

1. Общая информация об объекте закупки
   1. Объект закупки

Выполнение работ по развитию геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (4 этап).

* 1. Код и наименование позиции Классификатора предметов государственного заказа

Код позиции КПГЗ: 02.13.04.02

Наименование позиции КПГЗ: Развитие и модернизация автоматизированных информационных систем

* 1. Наименование позиции Справочника предметов государственного заказа

Развитие и/или модернизация автоматизированной информационной системы, усл.ед

* 1. Место выполнения работ

г. Москва, 2-й Лесной переулок д. 11, с.1.

* 1. Объем работ

Работы по развитию геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (4 этап) должны выполняться в 5 этапов.

В первом этапе должны быть выполнены работы:

* Техно-рабочее проектирование;
* Доработка системы;
* Предварительные испытания;
* Опытная эксплуатация;
* Приемочные испытания.

Во втором этапе должны быть выполнены работы:

* Техно-рабочее проектирование;
* Доработка системы;
* Предварительные испытания;
* Опытная эксплуатация;
* Приемочные испытания.

В третьем этапе должны быть выполнены работы:

* Техно-рабочее проектирование;
* Доработка системы;
* Предварительные испытания;
* Опытная эксплуатация;
* Приемочные испытания.

В четвертом этапе должны быть выполнены работы:

* Техно-рабочее проектирование;
* Доработка системы;
* Предварительные испытания;
* Опытная эксплуатация;
* Приемочные испытания.

В пятом этапе должны быть выполнены работы:

* Техно-рабочее проектирование;
* Доработка системы;
* Предварительные испытания;
* Опытная эксплуатация;
* Приемочные испытания.
  1. Срок выполнения работ

**Общий срок выполнения работ:** в течение 731 календарного дня с даты заключения Государственного контракта.

**Первый этап:** в течение 30 календарных дней с даты заключения Государственного контракта.

**Второй этап:** в течение 100 календарных дней с даты завершения первого этапа.

**Третий этап:** в течение 191 календарного дня с даты завершения второго этапа.

**Четвертый этап:** в течение 169 календарных дней с даты завершения третьего этапа.

**Пятый этап:** в течение 241 календарного дня с даты завершения четвертого этапа.

Подрядчик имеет право выполнить работы по Государственному контракту досрочно.

**Термины и определения**

В документ введены следующие термины, специальные обозначения и сокращения:

| **Наименование термина / сокращения** | **Определение** |
| --- | --- |
| АЗС | Автомобильная заправочная станция |
| АРМ | Автоматизированное рабочее место |
| АС | Автоматизированная система |
| БД | База данных |
| ГК | Государственный контракт |
| ГКУ ЦОДД | Государственное казенное учреждение города Москвы – Центр организации дорожного движения Правительства Москвы |
| ГЛОНАСС | Глобальная навигационная спутниковая система |
| ГОСТ | Межгосударственный стандарт — региональный стандарт, принятый Межгосударственным советом по стандартизации, метрологии и сертификации Содружества Независимых Государств |
| ГРЗ | Государственный регистрационный знак автомобиля |
| ЕСОО | Автоматизированная информационная система «Единая система обработки обращений» |
| ж\д | Железная дорога |
| ИЦ | Информационный центр |
| КПГЗ | Классификатор предметов государственного заказа |
| МО | Московская область |
| МЦД | Московские центральные диаметры |
| МЦК | Московское центральное кольцо |
| НГПТ | Наземный городской пассажирский транспорт |
| Общее программное обеспечение | Часть программного обеспечения автоматизированной системы, представляющая собой совокупность программных средств, разработанных вне связи с созданием данной АС |
| ОЗУ | Оперативное запоминающее устройство |
| ПО | Программное обеспечение |
| ППЗТ | Плоскостные парковки закрытого типа |
| ПЦ | Процессинговый центр |
| РД | Руководящий документ |
| Система | Геосоциальная система для пассажиров комплекса общественного транспорта города Москвы |
| Специальное программное обеспечение | Часть программного АС, представляющая собой совокупность программ, разработанных при создании данной АС |
| СУБД | Система управления базами данных |
| ТЗ | Техническое задание |
| ТС | Транспортное средство |
| ТСП | Торгово-сервисное предприятие (юридическое лицо, индивидуальный предприниматель, кредитная организация и т.п.), реализующее товары или услуги через интернет или другими способами, и использующее ПЦ для проведения операций в рамках своей деятельности |
| ЦППК | Акционерное общество «Центральная пригородная пассажирская компания» |
| API | Application programming interface – программный интерфейс приложения, интерфейс прикладного программирования |
| Dev-ops | Методология активного взаимодействия специалистов по разработке со специалистами по информационно-технологическому обслуживанию и взаимная интеграция их рабочих процессов друг в друга для обеспечения качества продукта |
| GPS | Global Positioning System – система глобального позиционирования |
| HDD | Накопитель на жёстких магнитных дисках (hard disk drive) |
| NFC | Технология беспроводной передачи данных малого радиуса действия, которая даёт возможность обмена данными между устройствами (Near field communication) |
| OSM | (Open Street Map) Некоммерческий веб-картографический проект |
| PIN | Аналог пароля. В ходе авторизации операции используется одновременно как пароль доступа держателя карты к терминалу и как секретный ключ для цифровой подписи запроса |
| PUSH-сообщения | Это короткие сообщения, которые веб-ресурс рассылает своим подписчикам на компьютеры и мобильные устройства |
| root-доступ | Полноценный доступ ко всем данным устройства |
| SSD | Компьютерное энергонезависимое немеханическое запоминающее устройство на основе микросхем памяти, альтернатива HDD (Solid-State Drive) |
| SOS | SOS – международный сигнал бедствия |
| Wi-Fi | Технология беспроводной локальной сети с устройствами на основе стандартов IEEE 802.11 |

1. Стандарт работ
   1. Общие принципы выполнения работ по развитию и настройке Системы

Основание для создания информационной системы - Договор № 99 хоз. 13 от 04 февраля 2015 г.

Развитие Системы не должно приводить к ухудшению функциональных и эксплуатационных характеристик действующей (дорабатываемой) Системы.

* 1. Общие требования к работам
     1. Требования к структуре и функционированию Системы

При выполнении работ по развитию Системы должны быть учтены и соблюдены перечисленные ниже требования.

Программное обеспечение для Системы должно представлять собой информационную систему, взаимодействующую с автоматизированными системами города Москвы и интегрируемую с другими информационными ресурсами.

Система должна иметь трехуровневую архитектуру:

* уровень базы данных (сервер СУБД);
* уровень сервера приложений;
* уровень клиентского приложения (мобильный интерфейс).

Архитектура Системы должна обеспечивать гибкое масштабирование производительности, наращивание объемов хранимой и обрабатываемой информации без длительных простоев в работе.

Механизмы управления должны обеспечивать управление Системой на всех уровнях архитектуры: инфраструктурном, функциональном, представления данных.

Все составляющие Системы должны использовать единую методологию и отвечать единым принципам взаимодействия, надежности и управления.

* + 1. Требования к способам и средствам связи для информационного обмена между компонентами Системы

Информационный обмен между компонентами Системы должен осуществляться:

* с использованием совместного доступа к базе данных Системы;
* с использованием вызовов веб-сервисов, реализованных в интерфейсах подсистем.
  + 1. Требования к характеристикам взаимосвязей Системы с внешними системами

Система взаимодействует со следующими внешними системами:

| **Обозначение** | **Наименование внешней системы** | **Направление потоков данных** | **Описание** |
| --- | --- | --- | --- |
| **Существующие интеграции со внешними системами** | | | |
| АИС КПП | Автоматизированная информационная система «Контроль пассажирских перевозчиков» | Входящий | Получаемые данные из смежной системы:   * маршруты НГПТ; * остановки НГПТ; * пути следования НГПТ; * расписание движения НГПТ; * прогнозы прибытия НГПТ; * вместимость ТС; * модели и номера машин каршеринга |
| РНИС | Государственная информационная система «Единая региональная навигационная система города Москвы» | Входящий | Получаемые данные из смежной системы:   * местоположение ТС; * местоположение машин каршеринга |
| РНИС МО | Государственная информационная система Московской области «Региональная навигационно-информационная система Московской области» | Входящий | Получаемые данные из смежной системы:   * маршруты НГПТ в МО; * остановки НГПТ в МО; * пути следования НГПТ в МО; * расписание движения НГПТ в МО |
| АИС ОВСБ | API оператора велопрокатов | Входящий | Получаемые данные из смежной системы:   * станции велопроката; * информация о свободных велосипедах и стойках |
| АСУ «Е-Автовокзал» | Автоматизированная система управления «Е-Автовокзал» | Входящий\Исходящий | Получаемые данные из смежной системы:   * станции городских автовокзалов; * станции назначения; * список рейсов на выбранную дату; * маршруты рейсов; * количество свободных мест   Отправляемые данные в смежную систему:   * станция отправления; * станция назначения; * дата |
| АИС ОЭСЮ | API оператора электросамокатов | Входящий | Получаемые данные из смежной системы:   * местонахождение электросамокатов; * уровень заряда; * местоположение парковок самокатов; * зоны работы сервисов |
| AИС ОЭСВ | API оператора электросамокатов | Входящий | Получаемые данные из смежной системы:   * местонахождение электросамокатов; * уровень заряда; * местоположение парковок самокатов; * зоны работы сервисов |
| АИС ОТЯ | API оператора такси | Входящий\Исходящий | Получаемые данные из смежной системы:   * список тарифов; * стоимость поездки по маршруту   Передаваемые данные в смежную систему:   * координаты точки отправления; * координаты точки назначения |
| АИС ОТСМ | API оператора такси | Входящий\Исходящий | Получаемые данные из смежной системы:   * список тарифов; * стоимость поездки по маршруту   Передаваемые данные в смежную систему:   * координаты точки отправления; * координаты точки назначения |
| АИС ПОД | Портал открытых данных Правительства города Москвы | Входящий | Получаемые данные из смежной системы:   * расположение киосков пополнения проездных; * информация о киоске |
| АИС ЯЭ | API сервиса расписаний электричек | Входящий | Получаемые данные из смежной системы:   * расположение ж\д станций; * расписание электричек; * маршруты электричек |
| КД ДОС | Интегрированные данные об организациях | Входящий | Получаемые данные из смежной системы:   * организации и филиалы организаций |
| АИС ДСТ | API Департамента спорта и туризма города Москвы | Входящий | Получаемые данные из смежной системы:   * местоположение значимых городских объектов; * информация о значимых городских объектах |
| АИС НС | Интеграция сообщений из Twitter-каналов ГУП «Мосгортранс», ГКУ «ЦОДД», ГУП «Московский метрополитен» | Входящий | Получаемые данные из смежной системы:   * сообщения из твиттер каналов |
| АИС ПС | API погодного сервиса | Входящий | Получаемые данные из смежной системы:   * получение прогнозов погоды |
| КС ОСМ | Картографический сервис | Входящий | Получаемые данные из смежной системы:   * картографические данные; * адреса |
| КС МБ | Картографический сервис | Входящий | Получаемые данные из смежной системы:   * картографические данные |
| АИС ЕСОО | Автоматизированная информационная система «Единого окна» обработки обращений граждан | Исходящий | Отправляемые данные в смежную систему:   * отправка обращений в части оценки ТС |
| АМ ВК | Авторизационный модуль социальной сети | Входящий | Получаемые данные из смежной системы:   * авторизационный токен; * данные о пользователе |
| АМ ФБ | Авторизационный модуль социальной сети | Входящий | Получаемые данные из смежной системы:   * авторизационный токен; * данные о пользователе |
| АМ ОК | Авторизационный модуль социальной сети | Входящий | Получаемые данные из смежной системы:   * авторизационный токен; * данные о пользователе |
| АМ ГС | Авторизационный модуль социальной сети | Входящий | Получаемые данные из смежной системы:   * авторизационный токен; * данные о пользователе |
| АМ НС | Авторизационный модуль социальной сети | Входящий | Получаемые данные из смежной системы:   * авторизационный токен; * данные о пользователе |
| ПС ПТК | Интеграция библиотеки платежной системы банка | Входящий | Получаемые данные из смежной системы:   * функционал работы с пополнением проездных   Отправляемые данные в смежную систему:   * идентификатор пользователя |
| АИС АПНС | API сервиса отправки уведомлений | Исходящий | Отправляемые данные в смежную систему:   * отправка PUSH-сообщений |
| АИС ГФ | API сервиса отправки уведомлений | Исходящий | Отправляемые данные в смежную систему:   * отправка PUSH-сообщений |
| АИС ССЯ | API сервиса по сбору статистики использования приложения | Входящий | Получаемые данные из смежной системы:   * события использования функций приложения |
| АИС ССА | API сервиса по сбору статистики использования приложения | Входящий | Получаемые данные из смежной системы:   * события использования приложения |
| **Модернизируемые взаимодействия со внешними системами** | | | |
| АИС ОВСБ | API оператора велопрокатов | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о пользователе; * список абонементов на покупку; * активные поездки; * история поездок; * история платежей;   Отправляемые данные в смежную систему:   * id пользователя; * регистрация пользователя; * запрос на оплату абонементов, задолженности, активацию промокода |
| АИС ОЭСЮ | API оператора электросамокатов | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о пользователе; * стоимость аренды; * статус бронирования; * статусы поездки; * история поездок   Отправляемые данные в смежную систему:   * запрос на бронирование; * запрос на начало поездки; * запрос на окончание поездки; * запрос на оплату поездки |
| AИС ОЭСВ | API оператора электросамокатов | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о пользователе; * статус бронирования   Отправляемые данные в смежную систему:   * запрос на бронирование |
| АИС ОТЯ | API оператора такси | Входящий\Исходящий | Получаемые данные из смежной системы:   * статусы заказа поездки; * статус оплаты поездки; * история поездок   Отправляемые данные в смежную систему:   * запрос заказа поездки; * выбранный способ оплаты |
| АИС ОТСМ | API оператора такси | Входящий\Исходящий | Получаемые данные из смежной системы:   * статусы заказа поездки; * статус оплаты поездки; * история поездок   Отправляемые данные в смежную систему:   * запрос заказа поездки; * выбранный способ оплаты |
| АИС ЕСОО | Автоматизированная информационная система «Единого окна» обработки обращений граждан | Входящий\Исходящий | Получаемые данные из смежной системы:   * идентификатор обращения; * статус обращения; * ответ на обращение   Отправляемые данные в смежную систему:   * запрос на создание обращения; * запрос на получение статуса обращения |
| **Создаваемые взаимодействия со внешними системами** | | | |
| АИС ЕПП | Автоматизированная информационная система «Единое парковочное пространство города Москвы» | Входящий\Исходящий | Получаемые данные из смежной системы:   * местонахождение парковок; * информация о парковках (места, стоимость, тип парковки); * стоимость парковочной сессии; * информация о парковочной сессии; * информация об автомобилях пользователя; * информация о балансе   Отправляемые данные из смежной системы:   * создание парковочной сессии; * завершение парковочной сессии; * изменение парковочной сессии; * пополнение баланса; * добавление автомобиля в систему |
| АИС УМП | API сервиса «Универсальная мультисервисная платформа» | Входящий | Получаемые данные из смежной системы:   * информация о входах и выходах пассажиров на остановках НГПТ |
| АИС ИЦ БТ | API сервиса загруженности ИЦ «Безопасный транспорт» | Входящий | Получаемые данные из смежной системы:   * информация о загруженности станций метро, маршрутов НГПТ, парковок |
| АИС ГКУ ММ | API сервиса ГКУ «Московский Метрополитен» | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о загруженности вагонов; * информация о транспортных картах пользователя; * получение баланса карты; * перенос баланса карты   Отправляемые данные в смежную систему:   * регистрация пользователя; * добавление транспортной карты; * удаление транспортной карты; * перенос баланса; * покупка абонемента |
| АИС ВВТ | API сервиса, предоставляющего информацию о водных видах транспорта | Входящий | Получаемые данные из смежной системы:   * информация о маршрутах водных видов транспорта; * информация о расписании водных видов транспорта; * информация о стоимости водных видов транспорта |
| АИС «Аэроэкспресс» | API компании ООО «Аэроэкспресс» | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о маршрутах аэроэкспресса; * информация о расписании аэроэкспресса; * информация о стоимости аэроэкспресса; * информация о купленных билетах   Отправляемые данные:   * запрос на покупку билетов аэроэкспресса; |
| АИС ЦППК | API компании АО «Центральная ППК» | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о маршрутах электричек; * информация о расписании электричек; * информация о стоимости поездок на электричках; * информация о купленных билетах   Отправляемые данные:   * запрос на покупку билетов |
| АИС «Россети» | API электрозарядных станций ПАО «Россети» | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о местонахождении электрозарядных станций; * информация о зарядной станции; * информация о статусе бронирования; * информация о статусе заряда   Отправляемые данные из смежной системы:   * запрос на бронирование; * запрос на старт зарядной сессии; * запрос на завершение бронирования; * запрос на завершение зарядной сессии |
| ИАС МПС | Информационно-аналитическая система мониторинга подвижного состава ГКУ «ЦОДД» Правительства Москвы – сервис перекрытий | Входящий | Получаемые данные из смежной системы:   * информация о перекрытиях |
| АИС ЗД | API сервиса загруженности дорог ГКУ «ЦОДД» | Входящий | Получаемые данные из смежной системы:   * информация о загруженности дорог |
| АИС «ВТБ» | API сервиса ПАО «ВТБ» | Входящий\Исходящий | Получаемые данные из смежной системы:   * список тарифов для покупки; * статус покупки тарифа; * остаток поездок; * баланс карты   Отправляемые данные во внешнюю систему:   * номер карты пользователя; * выбранный тариф; * стоимость; * запрос на оплату |
| АИС СБР | API системы биометрического распознавания для оплаты проезда | Входящий\Исходящий | Получаемые данные из смежной системы:   * информация о пользователе; * история проходов   Отправляемые данные во внешнюю систему:   * идентификатор пользователя; * фотография пользователя; * платежная информация |
| АИС ОМТС | API сервиса обмена мгновенными текстовыми сообщениями | Входящий | Получаемые данные из смежной системы:   * список сообщений;   Отправляемые данные во внешнюю систему:   * запрос на отправку сообщения; * запрос на получение списка сообщений |
| АИС «ЧБТС» | API сервиса чат-бота транспортной системы города Москвы | Входящий\Исходящий | Получаемые данные из смежной системы:   * список вариантов ответа   Отправляемые данные во внешнюю систему:   * запрос на отправку сообщения |
| АИС УИР | API сервиса уведомлений об изменении работы транспорта ИЦ «Безопасный транспорт» | Входящий | Получаемые данные из смежной системы:   * список сообщений об изменении работы НГПТ и станций метрополитена |
| АИС МКРТ | API сервиса микротранзита | Входящий\Исходящий | Получаемые данные из смежной системы:   * зона действия сервиса; * список возможных рейсов; * изменение статусов забронированного рейса   Отправляемые данные во внешнюю систему:   * запрос на поиск вариантов рейсов; * запрос на бронирование выбранного рейса; * запрос на получение статуса выбранного рейса; * запрос на верификацию пользователя; * запрос на отмену рейса |

Взаимодействие с внешними системами должно строиться с использованием технологий веб-сервисов.

* + 1. Требования к режимам функционирования

Система должна иметь возможность функционировать в следующих режимах:

* штатный режим работы;
* сервисный режим работы;
* аварийный режим работы.

Штатный режим работы должен являться основным режимом функционирования, обеспечивающим выполнение задач Системы.

Сервисный режим работы должен являться технологическим режимом и использоваться для сопровождения Системы – изменения конфигурации, параметров работы, настроек, выполнения регламентного обслуживания программно-технических средств.

Аварийный режим работы Системы должен характеризоваться отказом одного или нескольких компонентов программного и/или технического обеспечения.

* + 1. Перспективы развития Системы

Проектные решения, применяемые при выполнении работ по развитию Системы, должны обеспечивать возможность дальнейшего развития Системы.

Технические решения, используемые при проектировании и реализации Системы, должны позволять минимизировать трудозатраты по развитию и настройке, требуемые в связи с выпуском новых, обновлением существующих или отменой действующих нормативных актов, приводящих к изменению технологического процесса.

Система должна иметь структуру, позволяющую проводить обновления, усовершенствования и замену программно-аппаратных средств любой из подсистем без необходимости внесения изменений в другие подсистемы.

Должна быть предусмотрена возможность масштабирования Системы при увеличении нагрузки на Систему, должны учитываться требования к увеличению нагрузки, объемам информации и числу пользователей, а также последующему расширению функциональности.

Должна быть предусмотрена возможность дальнейшего развития Системы в следующих направлениях:

* расширение функциональных возможностей Системы;
* увеличение количества систем, участвующих в информационном взаимодействии с Системой.

Архитектура Системы должна позволять наращивать и модифицировать функциональные возможности Системы.

* + 1. Требования к численности и квалификации пользователей и персонала Системы и режиму его работы

Разрабатываемая Система должна быть рассчитана на следующие группы пользователей и персонала:

* Пользователи – граждане РФ и иностранные граждане (владеющие одним из языком пользовательского интерфейса: русским, английским, французским, немецким, испанским, китайским), использующие функционал Системы через предоставляемый пользовательский интерфейс;
* Модераторы, редакционная группа – специалисты эксплуатирующей организации, отвечающие за контроль и информационное наполнение ресурса;
* Администраторы – специалисты, выполняющие задачи по обеспечению функционирования разработанного программного обеспечения;
* Системные администраторы – специалисты, выполняющие работы по обслуживанию Системы: выполнение регламентных работ, настройка и диагностирование работы Системы, обеспечение функционирования системного программного обеспечения.

Требования к пользователям Системы представлены в таблице ниже:

| **Наименование пользователя** | **Квалификация** | **Численность** |
| --- | --- | --- |
| Пользователь | Пользователь низкой квалификации | Не предъявляется |
| Редакционная группа, Модератор | Пользователь высокой квалификации | 4 |
| Администратор | Квалифицированный специалист | 2 |
| Системный администратор | Квалифицированный специалист | 2 |

Требования к уровням квалификации пользователей Системы представлены в таблице ниже:

| **Уровень квалификации** | **Требования** |
| --- | --- |
| Пользователь низкой квалификации | Владение базовыми функциями операционной системы, без навыков настройки и администрирования данной операционной системы |
| Пользователь высокой квалификации | Уверенные пользователи персонального компьютера, со знанием операционных систем и общего программного обеспечения; посещение обучающих курсов по изучению разработанной Системы |
| Квалифицированный специалист | Высшее техническое образование, знание английского языка (технический перевод), теоретические знания и практические навыки по администрированию операционных систем, вычислительной сети, СУБД и приложений, а также владение способами архивации и аварийного восстановления базы данных; умение осуществлять настройку программной и аппаратной части Системы; обладание знаниями и умением классифицировать и устранять возникающие ошибки |

* + 1. Требования к надежности

При функционировании Системы должны выполняться следующие требования.

За отказ работоспособности Системы принимается неполучение пользователем ответа на запрос в течение времени, превышающего 15 секунд, без учета времени передачи информации по сети, при 2000 и более одновременно работающих пользователей.

Критерии отказа Системы и (или) ее компонентов определяются средним временем:

* наработки на отказ;
* восстановления работоспособности.

Средний показатель времени наработки на отказ должен составлять не менее 100 календарных дней.

Средний показатель времени восстановления работоспособности должен составлять:

* при развертывании специального ПО – не более 1 часа;
* при развертывании общего и специального ПО – не более 8 часов;
* общее время внеплановой недоступности Системы не должно превышать 72 часов в год.

К программным средствам сервера, на котором размещены компоненты Системы, должны предъявляться следующие требования по надежности:

* среднее время восстановления программных средств сервера должно быть не более 3-х часов в рабочее время и не более 8 часов в ночное время по будним дням, а также не более 8 часов в праздничные и выходные дни;
* коэффициент готовности к работе программных средств Системы (отношение времени исправной работы к сумме времен исправной работы и вынужденных простоев за календарный год) должен быть не менее 99,5%.

Технические средства сервера, на котором размещены компоненты Системы, должны удовлетворять следующим требованиям к надежности:

* суммарное время на восстановление работоспособности и регламентное обслуживание серверов – не более 48 часов в год.

Время восстановления работоспособности должно включать в себя:

* время на диагностирование отказа, замену или ремонт оборудования (без учета времени на заказ и поставку);
* конфигурирование оборудования и ПО;
* восстановление данных и тестирование работоспособности оборудования и ПО.

Для обеспечения устойчивости к отказам электроснабжения все устройства хранения и обработки информации Системы должны быть подключены к электросети через источники бесперебойного питания с функцией автоматического предупреждения персонала, управляющие корректным завершением операций и автоматическим сохранением данных при переходе на резервное питание.

* + - 1. Перечень аварийных ситуаций, приводящих к отказу Системы и (или) ее компонентов

Система должна обеспечивать круглосуточный режим функционирования 7 дней в неделю.

Время гарантированной работы для завершения серверных и прикладных приложений, обеспечиваемое источниками бесперебойного питания, должно быть не менее 3 часов.

Сохранность работоспособности и информации в Системе в пределах значений показателей надежности, приведенных в настоящем ТЗ, должна обеспечиваться при возникновении следующих аварийных ситуаций:

* отказы в Системе электроснабжения:
* отказы технических средств электропитания;
* полное отключение электроэнергии;
* отказы комплекса технических средств (аппаратных средств):
* отказы серверного оборудования;
* отказы подсистем;
* отказы сетевого, телекоммуникационного оборудования и каналов связи;
* отказы оборудования подсистемы резервного копирования информации;
* отказы программных средств:
* отказы общего ПО;
* отказы специального ПО;
* отказы в результате ошибок пользователей.
  + 1. Требования к эргономике и технической эстетике

Система должна иметь человеко-машинный интерфейс, удовлетворяющий следующим требованиям:

* Взаимодействие Системы и пользователя должно осуществляться на русском, английском, французском, немецком, испанском, китайском языках;
* Должно быть реализовано отображение на экране только тех возможностей, которые доступны пользователю в соответствии с его функциональной ролью в Системе;
* Представление управляющих элементов, экранных форм и их информационных элементов (окон, панелей и т.п.) должно быть унифицировано. Экранные формы должны полностью находиться в видимой площади экрана мобильного устройства.
  + 1. Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов Системы

Эксплуатация Системы должна производиться в соответствии с эксплуатационной документацией на Систему.

Эксплуатационная документация на систему должна быть передана Заказчику не позже 5 рабочих дней с момента заключения Государственного контракта в составе:

1. Руководство пользователя;
2. Руководство администратора.
   * 1. Требования к информационной безопасности и защите информации от несанкционированного доступа

Система относится к группе многопользовательских автоматизированных информационных систем с разными правами доступа. С учетом особенностей планируемой к обработке информации, Система должна соответствовать Федеральному закону от 27.07.2006 № 149-ФЗ "Об информации, информационных технологиях и о защите информации".

Аттестация Системы проводится Заказчиком самостоятельно.

* + - 1. Требования к функциям установления и подтверждения субъекта

Должна осуществляться идентификация и проверка подлинности субъектов доступа при входе в Систему по идентификатору и паролю.

* + - 1. Требования к защите от подбора идентификатора и пароля

Проверка введенной информации (идентификатор, пароль) осуществляется только после её полного ввода.

В случае обнаружения ошибки, Система не должна уточнять какие именно данные введены неправильно.

* + - 1. Требования к защите от утечки данных в процессе передачи данных между компонентами Системы

Хранение и передача пароля между клиентом и сервером аутентификации должны осуществляться в защищённом с помощью криптографических алгоритмов виде или передаваться с использованием защищённых каналов связи.

* + - 1. Требования к управлению правами доступа субъектов

В системе должна быть назначена роль, которая имеет возможность назначить права для пользователей, контролировать список пользователей системы (в том числе подключенных в настоящее время) и их права.

* + - 1. Требования по безопасности мобильного клиента

Авторизация пользователя в мобильном приложении Системы должна осуществляться путем ввода логина и пароля.

После авторизации Система не должна спрашивать у пользователя логин и пароль до момента выхода из Системы.

* + 1. Требования по сохранности информации при авариях
       1. Требования к организации процесса резервного копирования

Организация процесса резервного копирования разработанного функционала Системы осуществляется Подрядчиком.

* + - 1. Перечень аварий (событий), при которых должна быть обеспечена сохранность информации в Системе

В Системе должно предусматриваться автоматическое восстановление обрабатываемой информации в следующих аварийных ситуациях:

* программный сбой при операциях записи-чтения;
* разрыв связи с клиентской программой (терминальным устройством) в ходе редактирования/обновления информации.

В Системе должна предусматриваться возможность ручного восстановления обрабатываемой информации из резервной копии в следующих аварийных ситуациях:

* физический выход из строя дисковых накопителей, в результате которого произошла полная либо частичная потеря данных;
* ошибочные действия обслуживающего персонала.
* в Системе должно предусматриваться автоматическое восстановление работоспособности серверной части в следующих ситуациях:
* штатное и аварийное отключение электропитания серверной части;
* штатная перезагрузка Системы и загрузка после отключения;
* программный сбой общесистемного программного обеспечения, приведший к перезагрузке Системы.

В Системе должно предусматриваться полуавтоматическое восстановление работоспособности серверной части Системы в следующих аварийных ситуациях:

* физический выход из строя любого аппаратного компонента, кроме дисковых накопителей — после замены компонента и восстановления конфигурации общесистемного программного обеспечения;
* аварийная перезагрузка системы, приведшая к нарушению целостности файловой системы, с возможностью восстановления работоспособности после восстановления файловой системы.
  + 1. Требования к защите от влияния внешних воздействий

Требований к защите от внешних воздействий не предъявляется.

* + 1. Требования к правам на охраняемые результаты интеллектуальной деятельности

Исключительное право на Систему, а также на все результаты интеллектуальной деятельности, созданные при исполнении Государственного контракта, в том числе на техническую и рабочую документацию, в полном объеме принадлежат Заказчику и передаются Заказчику по Акту приема-передачи исключительных прав на программное обеспечение системы.

Сведения обо всех результатах интеллектуальной деятельности, созданные и/или использованные в процессе выполнения работ, подлежат отражению в технической документации.

Если при выполнении настоящего ТЗ будет создано программное обеспечение, база данных, изобретение, полезная модель или промышленный образец, создание которых прямо не предусмотрено настоящим ТЗ, исключительное право на указанную интеллектуальную собственность и право на получение патента на указанные изобретения, полезную модель или промышленный образец принадлежит Заказчику.

Подрядчик обязан путем заключения соответствующих договоров со своими работниками и третьими лицами приобрести все права или обеспечить их приобретение за свой счет для передачи соответственно Заказчику.

По поручению Заказчика Подрядчик производит оформление документов, необходимых для регистрации прав на Систему, а также на все результаты интеллектуальной деятельности, созданные при исполнении Государственного контракта, в течение 10 рабочих дней с момента обращения Заказчика.

При использовании в Системе программ (программных комплексов или компонентов), разработанных третьими лицами, условия, на которых передается право на использование (исполнение) этих программ, не должны накладывать ограничений, препятствующих использованию Системы по ее прямому назначению, и нарушать права третьих лиц. Права на указанные программы (программные комплексы или компоненты) передаются Заказчику на срок действия исключительных прав.

Подрядчик гарантирует, что возместит Заказчику все документально подтвержденные убытки в полном объеме, которые понесет Заказчик в связи с несоблюдением Подрядчиком обязательств и гарантий, предусмотренных настоящим разделом.

Подрядчик предоставляет Заказчику право на обнародование и переработку результатов интеллектуальной деятельности, а также право не указывать Автора/Разработчика, передаваемого ПО/СПО.

Подрядчик не вправе использовать результат интеллектуальной деятельности, полученный по результатам работ, для собственных нужд любым способом, в том числе и изготавливать на базе исходного текста программного обеспечения аналогичные программы и базы данных элементов, использованных при формировании данной программы.

Все затраты Подрядчика по передачи исключительных прав входят в стоимость ГК.

* + 1. Требования к техническому обеспечению Системы

Заказчик предоставляет технические средства, необходимые для функционирования Системы, включая серверное оборудование, необходимое для функционирования Системы.

Подрядчик производит установку и настройку разрабатываемого ПО Системы в объёме необходимом для обеспечения функционирования всех функций Системы, по окончании доработки программного обеспечения, на серверном оборудовании, предоставляемом Заказчиком.

Требования к серверам Dev-ops:

­ кол-во ядер процессоров — не менее 4;

­ объем ОЗУ — не менее 16 Гб;

­ SSD диск — не менее 100 ГБ;

Требования к серверам приложений:

­ кол-во ядер процессоров — не менее 8;

­ объем ОЗУ — не менее 16 Гб;

­ SSD диск — не менее 100 ГБ.

Требования к серверам управления кластерами Docker:

­ кол-во ядер процессоров — не менее 4;

­ объем ОЗУ — не менее 16 Гб;

­ SSD диск — не менее 100 ГБ.

Требования к серверам хранения расписаний:

­ кол-во ядер процессоров — не менее 6;

­ объем ОЗУ — не менее 16 Гб;

­ SSD диск — не менее 300 ГБ.

Требования к серверам баз данных Системы:

­ кол-во ядер процессоров — не менее 10;

­ объем ОЗУ — не менее 24 Гб;

­ SSD диск — не менее 600 ГБ.

Требования к серверам баз данных обработки телеметрии:

­ кол-во ядер процессоров — не менее 10;

­ объем ОЗУ — не менее 24 Гб;

­ SSD диск — не менее 200 ГБ.

Требования к серверам баз данных для хранения данных вида «Ключ – значение»:

­ кол-во ядер процессоров — не менее 4;

­ объем ОЗУ — не менее 32 Гб;

­ HDD диск — не менее 50 ГБ.

Требования к серверам поиска:

­ кол-во ядер процессоров — не менее 6;

­ объем ОЗУ — не менее 24 Гб;

­ SSD диск — не менее 100 ГБ.

Требования к серверам брокеров очередей:

­ кол-во ядер процессоров — не менее 4;

­ объем ОЗУ — не менее 16 Гб;

­ HDD диск — не менее 50 ГБ.

Требования к серверам балансеров:

* кол-во ядер процессоров — не менее 6;
* объем ОЗУ — не менее 16 Гб;
* SSD диск — не менее 50 ГБ.

Требования к серверам выполнения периодических заданий:

* кол-во ядер процессоров — не менее 6;
* объем ОЗУ — не менее 24 Гб;
* SSD диск — не менее 150 ГБ.

Требования к серверам маршрутизации:

* кол-во ядер процессоров — не менее 6;
* объем ОЗУ — не менее 24 Гб;
* SSD диск — не менее 100 ГБ.

Требования к серверам телеметрии:

* кол-во ядер процессоров — не менее 6;
* объем ОЗУ — не менее 16 Гб;
* SSD диск — не менее 100 ГБ.

Требования к серверам хранения пользовательской информации:

* кол-во ядер процессоров — не менее 4;
* объем ОЗУ — не менее 8 Гб;
* HDD диск — не менее 1 ТБ.

Требования к серверам хранения логов:

* кол-во ядер процессоров — не менее 6;
* объем ОЗУ — не менее 24 Гб;
* SSD диск — не менее 600 ГБ.

Требования к тестовым серверам:

* кол-во ядер процессоров — не менее 6;
* объем ОЗУ — не менее 24 Гб;
* SSD диск — не менее 500 ГБ.

Требования к серверам для инструментов разработки:

* кол-во ядер процессоров — не менее 2;
* объем ОЗУ — не менее 8 Гб;
* HDD диск — не менее 500 ГБ.

Требования к мобильному устройству для работы приложения:

* поддержка операционной системы iOS 12.0 и выше (для мобильного приложения под iOS) или Android 6.0 и выше (для мобильного приложения под Android);
* наличие модуля GPS/ГЛОНАСС;
* поддержка Wi-Fi и/или мобильных сетей 3G/4G/5G;
* отсутствие root-доступа.
  + 1. Требования к программному обеспечению Системы

Программное обеспечение Системы должно обладать следующими характеристиками.

На сервере должны быть установлены следующее программные средства:

* Ubuntu 16.04;
* Haproxy;
* PostgreSQL 9.6 и Postgis 2.4;
* ElasticSearch;
* PgBouncer;
* MongoDb;
* Redis;
* RabbitMQ;
* Docker;
* PHP;
* Golang;
* Java;
* NodeJS;
* Prometheus;
* Backupninja;
* Nginx.

Указанное серверное программное обеспечение предоставляется по открытым лицензиям (без взимания платы), в настоящее время установлено на серверах Заказчика и используется в текущей версии Системы.

На мобильных устройствах для работы мобильного приложения должны использоваться мобильные платформы:

* Apple iOS 12.0 и выше;
* Android 6.0 и выше.
  + 1. **Требования к лингвистическому обеспечению Системы**

В пользовательском интерфейсе должны применяться русский, английский, французский, немецкий, испанский, китайский языки.

Содержание используемых в Системе справочников должно быть представлено на русском, английском, французском, немецком, испанском, китайском языках.

Документация должна быть выполнена на русском языке.

Специальных требований к языкам манипулирования данными, средствам описания предметной области, а также языкам программирования высокого уровня не предъявляется.

* + 1. Требования к телекоммуникационному обеспечению

Система должна дорабатываться из расчета использования общегородских каналов связи, предоставляемых в рамках городской мультисервисной транспортной сети Правительства Москвы, с учётом реализованных технологий и протоколов передачи данных.

* + 1. Требования к условиям эксплуатации объекта автоматизации и характеристикам окружающей среды

В соответствии с ГОСТ 15150-69 технические средства, используемые в составе автоматизированного рабочего места, серверного и коммуникационного оборудования и размещаемые в отапливаемых и кондиционируемых помещениях, должны быть выполнены:

* по категории размещения изделий (для эксплуатации в помещениях (объемах) с искусственно регулируемыми климатическими условиями, например, в закрытых отапливаемых или охлаждаемых и вентилируемых производственных и других, в том числе хорошо вентилируемых подземных помещениях (отсутствие воздействия прямого солнечного излучения, атмосферных осадков, ветра, песка и пыли наружного воздуха; отсутствие или существенное уменьшение воздействия рассеянного солнечного излучения и конденсации влаги): для эксплуатации в помещениях с кондиционированным или частично кондиционированным воздухом;
* в климатическом исполнении (изделия, предназначенные для эксплуатации на суше, реках, озерах): для макроклиматического района с умеренным климатом.

Соответственно, диапазон рабочих температур для технических средств, используемых в составе АРМ, серверного и коммуникационного оборудования и размещаемых должен составлять от +10° до +25°C, а диапазон предельных рабочих температур – от +1° до +40°C.

* 1. Требования к документированию

В качестве результатов работ должен быть представлен дистрибутив программного обеспечения (вместе с исходными кодами) на электронном носителе в 2 экземплярах с Актом передачи исходного кода и дистрибутива.

Состав отчетной документации на Систему представлен в Таблице 1 Приложения 1 к ТЗ в столбце «Результаты, отчетные документы».

Документы должны быть разработаны с учетом требований ГОСТ 34.201 и представлены Заказчику для согласования и утверждения в отпечатанном и сброшюрованном виде в двух экземплярах, а также в электронном виде в формате, совместимом с MS Word.

1. Состав работ
   1. Требования к развитию Системы

Работы по развитию Системы должны включать в себя развитие и разработку следующих функциональных компонентов:

1) Развиваемые подсистемы:

* Подсистема построения маршрутов;
* Подсистема карты и поиска;
* Подсистема персонализированных рекомендаций;
* Подсистема билетного сервиса;
* Подсистема авторизации;
* Подсистема обратной связи;
* Подсистема профиля пользователя;
* Подсистема обработки данных о пользователях;
* Подсистема интеграций

2) Разрабатываемые подсистемы:

* Подсистема загруженности дорог и пробок;
* Подсистема оплаты банковской картой;
* Подсистема карты метро;
* Подсистема загруженности транспорта;
* Подсистема перекрытий;
* Подсистема такси;
* Подсистема каршеринга;
* Подсистема парковок;
* Подсистема краудсорсинга;
* Подсистема лояльности и геймификации;
* Подсистема велопроката;
* Подсистема самокатов;
* Подсистема электрозарядок;
* Подсистема биометрического распознавания для оплаты проезда;
* Подсистема микротранзит.
  + 1. Требования к развитию подсистемы построения маршрутов

Подсистема должна поддерживать перечисленные ниже функции.

* + - 1. Выбор времени отправления

Функция 1: Выбор времени отправления (этап 1).

Функция должна позволять при построении маршрута устанавливать время отправления из начальной точки маршрута для дальнейшего подбора оптимального маршрута.

Функция 2: Выбор оптимального маршрута, удовлетворяющего времени отправления (этап 1).

Функция должна определять оптимальные маршруты, которые удовлетворяют времени отправления, а затем предлагать выбрать один из них пользователю.

Функция должна использовать параметр оптимальности, который был выбран пользователем.

Маршруты должны строиться для всех видов транспорта, которые удовлетворяют маршруту и времени отправления.

При построении маршрутов должно учитываться расписание движения транспорта.

При построении маршрута должно учитываться время пеших переходов до остановок транспорта.

Функция 3: Выбор времени прибытия (этап 1).

Функция должна позволять устанавливать время прибытия в конечную точку маршрута для дальнейшего подбора оптимального маршрута.

Функция 4: Выбор оптимального маршрута, удовлетворяющего времени прибытия (этап 1).

Функция должна определять оптимальные маршруты, которые удовлетворяют времени прибытия, а затем предлагать выбрать один из них пользователю.

Функция должна использовать параметр оптимальности, который был выбран пользователем.

Маршруты должны строиться для всех видов транспорта, которые удовлетворяют маршруту и времени прибытия. Если маршруты, которые подходят для данного времени прибытия, отсутствуют, то функция должна построить маршруты для более раннего времени прибытия, с указанием времени ожидания в конечной точке маршрута.

Функция должна предлагать пользователю маршруты, время прибытия по которым ближе всего к указанному времени, но не позже.

При построении маршрутов должно учитываться расписание движения транспорта.

При построении маршрута должно учитываться время пеших переходов до остановок транспорта.

* + - 1. Построение маршрутов с учетом экологичных видов транспорта

Функция 1: Построение маршрутов с учетом экологичных видов транспорта (этап 3).

Функция должна позволять строить маршруты с учетом следующих видов экологичного транспорта: метро, МЦК, МЦД, электрички, электробусы, велосипеды, самокаты, трамваи.

Функция должна формировать маршруты, которые используют только экологичные виды транспорта, и предлагать их пользователю, как экологичные маршруты.

Для построения экологичных маршрутов должна быть реализована следующая таблица для маршрутов в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| name | varchar(255) | Название |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время обновления |
| type | enum(‘bus’,’tram’,’subway’,’trolleybus’,’mcc’,’mcd’,’train’, ‘bicycle’, ‘scooter’) | Тип транспорта |
| number | varchar(255) | Номер маршрута |
| external\_id | varchar(255) | Внешний идентификатор |
| color | varchar(255) | Цвет маршрута |
| font\_color | varchar(255) | Цвет шрифта |
| active | boolean | Активность маршрута |
| active\_for\_routing | boolean | Активность для маршрутизатора |
| map\_icon | varchar(255) | Иконка на карте |
| low\_floor | boolean | Только низкопольные ТС |
| source | varchar(255) | Внешний источник получения данных |
| electrobus | boolean | Только электробусы |

Функция должна проверять экологичность маршрута для каждого шага маршрута. Шаг должен считаться экологичным, если тип транспорта соответствует одному из перечисленных типов:

* bus,
* tram,
* subway,
* trolleybus,
* mcc,
* mcd,
* train,
* bicycle,
* scooter.

Функция 2: Расчет количества выброса СО2 (этап 3).

Функция должна рассчитывать усредненное количество выброса CO2 за весь маршрут, выбрасываемое всеми видами транспорта, используемыми при прохождении построенного маршрута.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| name | varchar(255) | Название |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время обновления |
| type | enum(‘bus’,’tram’,’subway’,’trolleybus’,’mcc’,’mcd’,’train’) | Тип транспорта |
| emission\_per\_km | integer | Количественный показатель выброса CO2 за 1км |

Функция должна рассчитывать выбросы СО2 для каждого шага маршрута, умножая значение выброса СО2 для вида транспорта, используемого на данном шаге, время пути в минутах, а затем суммируя результаты всех шагов маршрута.

Данные, необходимые для расчета выбросов CO2, предоставляются Заказчиком.

Функция 3: Расчет калорий (этап 3).

Функция должна рассчитывать усредненное количество калорий, сжигаемых пользователем, при прохождении данного маршрута. Функция должна брать в расчет среднее количество калорий для установленного среднего веса пассажира.

* + - 1. Перестроение в реальном времени в зависимости от предпочтений пользователя

Функция 1: Смена вида транспорта в пути (этап 5).

Функция должна позволять пользователю менять вид транспорта на одном из шагов маршрута, когда пользователь находится в режиме «в пути», а затем перестраивать маршрут на основании выбранного вида транспорта.

При смене вида транспорта функция должна перестраивать маршрут от начальной точки указанного шага, с использованием указанного вида транспорта.

Функция должна обновлять данные о маршруте (время в пути, стоимость и т.д.) при смене вида транспорта на одном из шагов.

Функция 2: Предложение более быстрых маршрутов во время пути (этап 5).

Функция должна предлагать пользователю, находящемуся в режиме «в пути», более быстрые маршруты, если таковые имеются.

Функция должна строить новые маршруты, которые удовлетворяют параметрам текущего маршрута, раз в определенный промежуток времени, и сравнивать их расчетное время прибытия. Если время прибытия нового маршрута раньше, чем текущего, то такой маршрут должен предлагаться пользователю, с возможностью выбора его как текущего.

После смены маршрута на более быстрый, функция должна продолжать искать более быстрые маршруты.

При поиске более быстрого маршрута должны учитываться различные виды транспорта, а не только те, которые используются в текущем маршруте.

* + - 1. Использование проката велосипедов в мультимодальных маршрутах

Функция 1: Выбор проката велосипеда как вид транспорта в мультимодальных маршрутах (этап 1).

Функция должна позволять выбирать прокат велосипеда как один из видов транспорта при построении мультимодального маршрута, а построенный маршрут должен проходить через стойки проката или возврата велосипеда.

Функция должна строить маршруты от начальной точки до конечной с учетом всех доступных видов транспорта, затем формировать перечень оптимальных маршрутов.

Функция должна заменять тип используемого транспорта на велосипед в первом или последнем шаге маршрута.

Шаг маршрута на велосипеде может заменять шаг общественного наземного транспорта или шаг метро, если шаг по протяженности не больше трех станций или остановок.

Функция должна проверять наличие велостанций в начале шага на велосипеде и в конце.

Функция должна проверять наличие велосипедов на велостанции в начальной точке и наличие свободных стоек для велосипеда в конечной точке.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| global\_id | varchar(255) | Внешний идентификатор |
| total\_slots | integer | Общее кол-во стоек |
| free\_slots | integer | Кол-во свободных стоек |
| avl\_bikes | integer | Кол-во свободных велосипедов |
| address | varchar(255) | Адрес парковки |
| address\_en | varchar(255) | Локализованный адрес парковки |
| lon | double | Широта |
| lat | double | Долгота |
| geopoint | geometry | Геометрия точки |
| status | varchar(255) | Статус (работает/не работает) |
| type | varchar(255) | Тип (обычные/электровелосипеды) |
| total\_ordinary\_slots | integer | Кол-во стоек для обычных велосипедов |
| total\_elecric\_slots | integer | Кол-во стоек для электрических велосипедов |
| free\_ordinary\_slots | integer | Кол-во свободных стоек обычных велосипедов |
| free\_electric\_slots | integer | Кол-во свободных стоек для электрических велосипедов |
| avl\_ordinary\_bikes | integer | Кол-во свободных обычных велосипедов |
| avl\_electric\_bikes | integer | Кол-во свободных электровелосипедов |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время редактирования |

* + - 1. Использование проката самокатов в мультимодальных маршрутах

Функция 1: Выбор проката самоката как вида транспорта (этап 1).

Функция должна позволять выбирать прокат самоката как один из видов транспорта при построении мультимодального маршрута, а построенный маршрут должен проходить от места его стоянки, до места, в котором его можно поставить на стоянку.

Функция должна строить маршруты от начальной точки до конечной с учетом всех доступных видов транспорта, затем формировать перечень оптимальных маршрутов.

Функция должна заменять тип используемого транспорта на самокат в первом или последнем шаге маршрута.

Шаг маршрута на самокате может заменять шаг общественного наземного транспорта или шаг метро, если конечная точка шага находится в зоне действия данного самоката.

Функция должна проверять наличие самокатов или самокатных парковок в начале шага на самокате.

Функция должна проверять наличие самокатных парковок или возможность парковки самоката на улице в конце шага.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| company\_id | uuid | Идентификатор прокатчика самокатов |
| address | varchar(255) | Адрес |
| lat | double | Широта |
| lon | double | Долгота |
| geopoint | geometry | Геометрия точки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 2: Поиск подходящих самокатов (этап 1).

Функция должна предлагать подходящие для прохождения данного маршрута самокаты, учитывая их расположение, уровень заряда и зону действия.

Функция должна проверять наличие самокатов поблизости.

Функция должна проверять удовлетворяет ли зона действия самоката параметрам маршрута.

Функция должна проверять удовлетворяет ли значение заряда самоката расстоянию поездки.

При поиске самоката должны учитываться только самокаты с зарядом не ниже 10%.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| name | varchar(255) | Название |
| logo\_id | varchar(255) | Логотип |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время редактирования |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| code | varchar(255) | Идентификатор во внешней системе |
| company\_id | uuid | Идентификатор прокатчика самокатов |
| power | integer | Уровень заряда (от 0 до 100) |
| deep\_link | varchar(255) | Ссылка для перехода к самокату в приложении прокатчика |
| lat | double | Широта |
| lon | double | Долгота |
| geopoint | geometry | Геометрия точки |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время обновления |

* + - 1. Сохранение истории завершенных маршрутов

Функция 1: Сохранение истории завершенных маршрутов (этап 4).

Функция должна сохранять историю всех завершенных маршрутов пользователя.

Функция должна проверять текущее местоположение пользователя, который находится «в пути».

Функция должна сохранять данные о маршрутах, которые были завершены, то есть пользователь достиг конечной точки маршрута, находясь в режиме «в пути».

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| created\_at | timestamp | Время построения |
| finished\_at | timestamp | Время завершения |
| start\_point | geometry | Точка А |
| finish\_point | geometry | Точка Б |
| route\_params | json | Параметры маршрута |
| route\_index | integer | Индекс маршрута из общей коллекции построенных маршрутов |
| route | json | Краткое представление маршрута (без геометрии) |

Каждая завершенная поездка должна быть сохранена новой записью в таблице.

Функция 2: Отображение статистики завершенных маршрутов (этап 4).

Функция должна отображать статистику по всем завершенным маршрутам для дальнейшего отображения в личном кабинете пользователя.

* + - 1. Напоминания о выходе

Функция 1: Напоминание о выходе из дома (этап 4).

Функция должна отправлять PUSH-уведомление с напоминанием о выходе за некоторое время до отправления по одному из запланированных маршрутов. Уведомление должно учитывать время, необходимое чтобы дойти до ближайшей остановки, и запасное время.

Функция должна обновлять данные по маршруту (время в пути, пробки и т.д.) с периодичностью, указанной в данных о запланированном маршруте. Обновление данных должно выполняться до наступления времени выхода и отправки оповещения.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_id | uuid | Идентификатор маршрута |
| path\_geometry | geometry | Геометрия пути |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| type | enum(‘bus’,’tram’,’subway’,’trolleybus’,’mcc’,’mcd’,’train’) | Тип транспорта |
| name | varchar(255) | Название пути |
| number | varchar(255) | Номер пути |
| external\_id | varchar(255) | Внешний идентификатор |
| active | boolean | Активность |
| direction | integer | Направление маршрута |
| is\_circle | boolean | Признак кольцевого маршрута |
| color | varchar(255) | Цвет пути |
| start\_name | varchar(255) | Название начальной остановки |
| finish\_name | varchar(255) | Название конечной остановки |

Функция 2: Напоминание о выходе на остановке (этап 4).

Функция должна определять текущее местоположение пользователя, если он приближается к остановке общественного транспорта, на которой пользователю необходимо выйти, то должно отправляться PUSH-уведомление с напоминанием о выходе на данной остановке.

Функция должна работать только в режиме «в пути».

Функция должна определять на каком именно транспортном средстве передвигается пользователь.

Функция должна отслеживать продвижение транспортного средства по остановочным пунктам пути. Оповещение должно отправляться после начала движения от предпоследней остановки данного шага маршрута.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_id | uuid | Идентификатор маршрута |
| path\_geometry | geometry | Геометрия пути |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| type | enum(‘bus’,’tram’,’subway’,’trolleybus’,’mcc’,’mcd’,’train’) | Тип транспорта |
| name | varchar(255) | Название пути |
| number | varchar(255) | Номер пути |
| external\_id | varchar(255) | Внешний идентификатор |
| active | boolean | Активность |
| direction | integer | Направление маршрута |
| is\_circle | boolean | Признак кольцевого маршрута |
| color | varchar(255) | Цвет пути |
| start\_name | varchar(255) | Название начальной остановки |
| finish\_name | varchar(255) | Название конечной остановки |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| route\_path\_id | uuid | Идентификатор пути маршрута |
| stop\_id | uuid | Идентификатор остановки |
| weight | integer | Порядок следования остановки |
| end\_stop | boolean | Признак конечной остановки |
| distance | double | Расстояние от прошлой остановки |
| time | integer | Время в пути от прошлой остановки |
| path\_geometry | geometry | Геометрия отрезка от предыдущей остановки до текущей |
| prev\_stop\_id | uuid | Идентификатор предыдущей остановки |

* + - 1. Учет реальных прогнозов прибытия транспорта

Функция 1: Учет реальных прогнозов прибытия транспорта (этап 1).

Функция должна учитывать реальные прогнозы прибытия и расписание движения наземного транспорта при построении маршрута, а также выбирать оптимальные маршруты на основании учтенных данных. Функция должна быть опциональной.

Функция должна получать данные о прогнозах прибытия транспорта из сопряженных систем.

При построении маршрутов функция должна учитывать только тот транспорт, который прибудет на остановку, являющуюся началом шага маршрута, не менее чем через 4 минуты.

Если прогнозы прибытия недоступны, то функция должна использовать расписание движения транспорта для расчетов времени прибытия.

Для работы функции должна быть реализована следующая таблица БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| vehicle\_id | varchar(255) | ИД ТС |
| telemetry\_id | varchar(255) | ИД телематического датчика |
| external\_route\_id | varchar(255) | Идентификатор прогноза во внешней системе |
| external\_stop\_id | varchar(255) | Идентификатор остановки во внешней системе |
| route\_id | uuid | Идентификатор маршрута |
| route\_path\_id | uuid | Идентификатор пути маршрута |
| expected\_time | timestamp | Ожидаемое время прибытия |
| stop\_id | uuid | Идентификатор остановки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| is\_scheudle | boolean | Прогноз по расписанию |

* + - 1. Сервис маршрутизации

Функция 1: Построение маршрутов в фоновом режиме (этап 5).

Функция должна позволять строить и перестраивать маршруты в фоновом режиме маршрутизатора.

Фоновый режим маршрутизатора должен активироваться в тот момент, когда окно приложения перестает быть активным окном в течение не менее 5 минут.

Функция должна проверять наличие движения пользователя по маршруту в режиме «В пути». Если пользователь не начинает движение по одному из участков маршрута, то функция должна обновлять актуальность последующих шагов.

Функция должна работать при включенном режиме «С учетом расписания».

Функция должна предлагать актуализировать построенный маршрут не чаще, чем раз в 5 минут.

Функция 2: Сохранение маршрутов для переиспользования (этап 5).

Функция должна позволять сохранять построенные маршруты с целью повторного использования при схожих начальных условиях построения маршрута. Функция должна сохранять следующие данные о маршруте:

* начальные и конечные точки шагов маршрута;
* используемые виды транспорта на шагах маршрута, включенные пользователем в настройках;
* флаг доступности для маломобильных;
* тип проездного пользователя;
* геометрия шагов маршрута;
* время в пути;
* цена;
* маршруты общественного транспорта (номера автобусов, троллейбусов и т.п.).

Сохраненные маршруты должны сохраняться в базе данных.

Для работы функции сохранения маршрутов в БД должна быть реализована следующая таблица:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| start\_point | geometry | Точка А |
| finish\_point | geometry | Точка Б |
| transports | array | Типы используемого транспорта |
| for\_disabled | boolean | Для маломобильных |
| created\_at | timestamp | Время создания маршрута |
| route\_data | json | Данные маршрута |

Функция 3: Обновление данных (этап 2).

Функция должна позволять обновлять данные, используемые сервисом маршрутизации, без перезапуска сервиса.

Функция должна позволять загружать все данные, необходимые для работы сервиса маршрутизации, на координирующий узел.

Функция должна позволять поочередно обновлять данные сервисов маршрутизации, заменяя их на данные, хранящиеся на координирующем узле.

Функция должна обновлять не более одной копии сервиса маршрутизации за раз. Функция должна переходить к обновлению следующей копии после перезагрузки текущей обновленной копии.

Для работы функции в БД должны быть реализованы следующие таблицы:

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| lat | double | Широта |
| lon | double | Долгота |
| geopoint | geometry | Геометрия точки |
| name | varchar(255) | Название остановки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| type | enum(‘public\_transport’, ‘subway’, ‘mcc’, ‘mcd’, ‘train’, ‘airport\_express’) | Тип транспорта |
| bench | boolean | Наличие скамеек (для остановок метро) |
| elevator | boolean | Наличие лифта (для остановок метро) |
| active | boolean | Активность |
| zone\_id | integer | Тарифная зона (для электричек) |
| souvenir\_shop | boolean | Наличие магазина с сувенирами |
| repair\_works | boolean | Веду строительные работы |
| aero\_express\_pass | boolean | Наличие перехода на аэроэкспресс |
| closed | boolean | Открыта/Закрыта |
| mcd\_zone\_id | integer | Идентификатор тарифной зоны МЦД |
| source | varchar(255) | Источник данных |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| stop\_id | uuid | Идентификатор станции метро |
| lat | double | Широта |
| lon | double | Долгота |
| geopoint | geometry | Геометрия маршрута |
| name | varchar(255) | Название станции |
| active | boolean | Активность |
| active\_route | boolean | Активно для маршрутизации |
| number | integer | Номер входа |
| entrance\_type | enum(‘all’,’exit’,’entrance’) | Проходимость входа |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| exit\_to | text | Объекты рядом с выходом |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| stop\_id\_in | uuid | ИД станции перехода |
| stop\_id\_from | uuid | ИД станции перехода |
| transfer\_time | integer | Время перехода |
| is\_active | boolean | Активность |
| forward\_path\_optimal\_carriage | integer | Оптимальный вагон для пересадки (прямой путь) |
| reverse\_path\_optimal\_carriage | integer | Оптимальный вагон для пересадки (обратный путь) |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_id | uuid | Идентификатор маршрута |
| date | date | Дата отправления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| path\_stop\_id | integer | ИД остановки на пути маршрута |
| arrival | time | Прибытие |
| departure | time | Отправление |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| alias | varchar(255) | Идентификатор во внешней системе |
| android\_app\_id | varchar(255) | Идентификатор приложения в Google Play |
| ios\_app\_id | varchar(255) | Идентификатор приложения в App Store |
| site\_url | varchar(255) | Веб-сайт |
| contact\_number | varchar(255) | Контактный номер |
| map\_icon | uuid | Логотип на карте |
| active | boolean | Активность |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| minute\_cost | integer | Стоимость в минуту |
| color | varchar(255) | Цвет |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | ObjectID | Идентификатор |
| tm\_id | integer | ID телематического датчика |
| address | string | Адрес |
| addressTranslit | string | Адрес транслитеризированный |
| expire\_at | ISOTime | Время жизни |
| model | string | Модель машины |
| plate | string | Гос. номер |
| organisation | ObjectID | Организация каршеринга |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| external\_id | integer | Идентификатор во внешней системе |
| free\_spaces\_total | integer | Кол-во мест |
| lat | double | Широта |
| lon | double | Долгота |
| geopoint | geometry | Геометрия точки |
| active | boolean | Активность |
| description | string | Описание |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_scheudle\_active\_period\_id | uuid | Идентификатор периода действий расписания |
| route\_path\_id | uuid | Идентификатор пути маршрута |
| stop\_id | uuid | Идентификатор остановки |
| arrival | integer | Прибытие (в минутах от начала дня) |
| departure | integer | Отправление (в минутах от начала дня) |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_id | uuid | Идентификатор маршрута |
| monday | boolean | Ходит в понедельник |
| tuesday | boolean | Ходит во вторник |
| wednesday | boolean | Ходит в среду |
| thursday | boolean | Ходит в четверг |
| friday | boolean | Ходит в пятницу |
| saturday | boolean | Ходит в субботу |
| sunday | boolean | Ходит в воскресенье |
| holiday | boolean | Ходит по выходным |
| date\_range | date\_range | Временной интервал |
| description | varchar(255) | Описание |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 4: Отображение оптимальных маршрутов (этап 3).

Функция должна предоставлять возможность пользователю выбирать оптимальные маршруты по заданным параметрам (включая маршруты для маломобильных и экомаршруты). В выборке похожие маршруты одного типа должны объединяться в группы.

Функция должна использовать следующие типы транспорта:

* только наземный транспорт;
* только метро;
* смешанный тип.

Функция 5: Построение маршрутов на метро в оффлайн (этап 5).

Функция должна позволять загружать данные, необходимые для построения маршрутов на метро, на устройство пользователя, а затем позволять строить на нем такие маршруты без подключения к сети интернет.

Функция должна загружать следующие данные:

* линии метро;
* остановки метро;
* маршруты;
* время перегонов;
* время переходов между станциями.

Данные должны загружаться только для следующих типов транспорта:

* Метро;
* МЦК;
* МЦД.

При наличии подключения к сети интернет функция должна обновлять сохраненные данные.

Функция 6: Учет перекрытий дорог при построении маршрута (этап 3).

Функция должна учитывать перекрытия дорог при построении маршрутов. Информация о перекрытиях должна обновляться не реже 1 раза в день.

При построении маршрутов функция должна искать другие оптимальные маршруты, обходящие перекрытия. Функция должна учитывать перекрытия при построении маршрутов для следующих видов маршрутов:

* каршеринг;
* личный автомобиль;
* мотоцикл.

Функция должна проверять наличие перекрытий на маршруте при каждом построении маршрута.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| polygon | geometry | Полигон перекрытия |
| color | varchar(255) | Цвет |
| time\_start | timestamp | Время начала перекрытия |
| time\_end | timestamp | Время окончания перекрытия |
| created\_at | timestamp | Время создания |
| name | varchar(255) | Название |
| show\_on\_map | boolean | Отображать на карте |

Данные о перекрытии дорог предоставляются Заказчиком.

Функция 7: Наложение собственных OSM данных на данные, получаемые из официального источника OSM (этап 5).

Функция должна позволять накладывать собственные данные в формате Open Street Map (OSM) на данные полученные из официального источника OSM. При обновлении собственных или официальных OSM данных они должны становится актуальными без перезагрузки сервиса маршрутизации.

Функция должна предоставлять пользовательский интерфейс, позволяющий работать с пользовательскими данными в формате OSM.

Функция должна позволять выполнять следующие операции с OSM данными:

* открывать карты OSM;
* отображать OSM файлы в виде карты;
* добавлять новые объекты на OSM карту;
* редактировать объекты на OSM карте;
* удалять объекты на OSM карте;
* сохранять OSM карту в файл в формате OSM.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | bigint | Идентификатор |
| tstamp | timestamp | Время редактирования |
| tags | hstore | Теги |
| geom | geometry | Геометрия |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | bigint | Идентификатор |
| tstamp | timestamp | Время редактирования |
| tags | hstore | Теги |
| nodes | array | Список нод |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | bigint | Идентификатор |
| tstamp | timestamp | Время редактирования |
| tags | hstore | Теги |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| relation\_id | bigint | Идентификатор relation |
| member\_id | bigint | Идентификатор связанного объекта |
| member\_type | char(1) | Тип связанного объекта - way(W), node(N), relation(R) |
| memeber\_role | text | Роль объекта в отношении |
| sequence\_id | integer | Идентификатор |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| way\_id | bigint | ID пути |
| node\_id | bigint | ID ноды |
| sequence\_id | integer | Идентификатор |

Функция 8: Геокодинг (этап 4).

Функция должна реализовывать иерархическую структуру адресных объектов в процедуре обратного геокодирования.

Иерархическая структура должна иметь следующие уровни:

* страна;
* город;
* улица;
* дом.

При указании точки геолокации функция должна определять к каким уровням иерархической структуры относится данная точка.

Каждый уровень должен представлять из себя полигон, который покрывает определенную область, в которую входит набор геолокационных данных.

Функция должна проверять вхождение точки геолокации в полигоны по возрастанию, т.е. начинать с самого нижнего уровня – дом, а затем проверять следующие по возрастанию полигоны, пока не дойдет до полигона уровня страна.

Каждый уровень полигона должен иметь ссылку на следующий по уровню возрастания полигон.

Функция должна формировать адрес на основании полигонов, в которые входит геолокационная точка в порядке уменьшения полигонов.

Функция 9: Автоматическое тестирование маршрутов (этап 4).

Функция должна позволять осуществлять интеграционное тестирование функционала сервиса маршрутизации, а также предоставлять возможность использовать различные эталонные выборки в качестве исходных данных для тестов.

Функция должна иметь возможность сохранить данные о текущих маршрутах, которые были сформированы маршрутизатором и хранятся в БД.

Должны сохраняться следующие данные:

* идентификатор маршрута;
* геометрия пути;
* дата создания маршрута;
* дата обновления маршрута;
* тип транспорта;
* название пути;
* номер пути;
* внешний идентификатор маршрута;
* активность маршрута;
* направление маршрута;
* признак кольцевого маршрута;
* название начальной остановки;
* название конечной остановки;
* расписание движения транспорта на маршруте;
* предполагаемое время прохождения маршрута.

Функция должна иметь возможность построить маршрут, используя данные сохраненных маршрутов как исходные. Функция должна иметь возможность сравнить полученный маршрут с сохраненными маршрутами. Функция должна отображать отличия между построенным и сохраненным маршрутом.

Функция должна иметь возможность запуска вручную и по расписанию.

Для работы функции необходимо реализовать следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| params | json | Параметры запроса к маршрутизатору |
| start | geometry | Точка А |
| finish | geometry | Точка Б |
| name | varchar(255) | Название плана |
| description | text | Описание плана |
| expected\_result | json | Ожидаемый результат выполнения плана |
| majority | enum(‘green’,’yellow’,’red’) | Важность соответствия результата ожиданиям |
| type | enum(‘isolated’,’common’) | Тип плана - где он должен выполняться - в изолированной среде или на продакшене |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| plan\_id | uuid | ID плана |
| testset\_id | uuid | Идентификатор запуска серии тестов |
| timestamp | timestamp | Время выполнения |
| result | json | Полученный результат |
| success | boolean | Успешность выполнения |
| conformity\_percent | integer | Процент соответствия ожиданиям |

Функция 10: Сбор статистики (этап 5).

Функция должна позволять собирать следующие статистические данные:

* время, за которое был выдан маршрут пользователю;
* список предложенных маршрутов пользователю, без фильтров с указанием видов транспорта;
* список предложенных маршрутов пользователю после фильтрации с указанием видов транспорта;
* какой маршрут выбрал пользователь;
* отправился ли пользователь по выбранному маршруту;
* какое время был открыт маршрут у пользователя;
* завершил ли данный маршрут пользователь.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор маршрута |
| timestamp | timestamp | Момент построения маршрута |
| time | integer | Время в миллисекундах, затраченное на построение маршрута |
| results\_raw | json | Результаты без фильтрации |
| results | json | Результаты после фильтрации |
| selected\_index | integer | Индекс маршрута в списке отфильтрованных результатов, который открыл пользователь |
| journey\_started | boolean | Отправился ли пользователь по маршруту |
| time\_active | integer | Время, пока маршрут был открыт |
| journey\_finished | boolean | Завершил ли пользователь маршрут |

Функция 11: Рейтинг маршрута (этап 5).

Функция должна позволять пользователю выставлять оценку пройденному маршруту.

Функция должна отображать среднее арифметическое рейтинга с точностью до десятых в списке предложенных маршрутов при построении маршрута.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| journey\_id | uuid | ID маршрута |
| comment | text | Комментарий к маршруту |
| rating | integer | Оценка (от 1 до 3) |
| user\_id | uuid | ID пользователя |
| timestamp | timestamp | Время создания оценки |
| selected\_index | integer | Индекс маршрута в списке результатов, который оценивает пользователь |

Функция 12: Учет особенностей инфраструктуры при построении веломаршрутов (этап 4).

Функция должна учитывать особенности городской инфраструктуры при построении маршрутов для велосипеда.

Данные об инфраструктуре для учета при построении веломаршрутов предоставляются Заказчиком по API.

Для работы функции должны быть реализованы следующие таблицы:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| object\_type | enum('underground','foreground') | Тип перехода (наземный, подземный) |
| geometry | geometry | Геометрия объекта |
| time | integer | Ориентировочное время перехода |
| osm\_nodes | array | Связанные с переходом ребра OSM |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| geometry | geometry | Геометрия велодорожки |
| osm\_nodes | array | Связанные OSM ребра дорог |

Функция 13: Объединение МЦД, ж\д станций, станций метро (этап 3).

Функция должна объединять станции МЦД с ж\д станциями, станциями метро или аэроэкспресса при построении маршрута, если они совмещены между собой.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| geopoint | geometry | Координаты станции |
| name | varchar(255) | Название |
| bench | boolean | Наличие скамеек |
| elevator | boolean | Наличие лифта |
| closed | boolean | Закрыта |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Поле** | **Поле** |
| id | integer | Идентификатор |
| union\_stop\_id | uuid | Идентификатор объединенной станции |
| type | enum('mcd','subway','train', 'airpot\_express') | Тип объекта |
| source\_stop\_id | uuid | ID остановки |

Функция 14: Поделиться местоположением (этап 5).

Функция должна передавать текущее местоположение пользователя с возможностью поделиться и отслеживать его передвижение на карте в режиме реального времени в течение заданного интервала времени.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Поле** | **Поле** |
| \_id | ObjetcID | Идентификатор |
| user\_id | string | ID пользователя |
| geopoint | geojson | Координата пользователя |
| device | string | Модель устройства |
| created\_at | ISODate | Дата создания |
| expire\_at | ISODate | Дата удаления точки (удаляем через месяц) |

Функция 15: Построение маршрута со значимыми объектами (этап 4).

Функция должна отображать значимые объекты на пути следования маршрута.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| icon\_id | uuid | ID иконки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| name | varchar(255) | Название |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| geopoint | geometry | Координаты |
| address | varchar(255) | Адрес |
| email | text[] | Email |
| phones | text[] | Телефоны |
| opening\_hours | text | Часы работы |
| web\_site | varchar(255) | Сайт |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| icon\_id | uuid | ID иконки |
| category\_id | uuid | ID категории |
| tags | hstore | Теги |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| poi\_id | uuid | ID POI |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| name | varchar(255) | Название |
| description | text | Описание |
| short\_description | varchar(255) | Короткое описание |

Функция 16: Отзыв о пройденном маршруте (этап 5).

Функция должна позволять публиковать отзыв о пройденном пользователем маршруте.

Отзыв должен сохраняться в базу данных Системы вместе с оценкой, которую оставил пользователь к данной поездке.

Отзывы должны содержать следующие данные:

* указанная пользователем оценка маршрута от 1 до 5;
* текст отзыва;
* дата отзыва.

Для работы функции в БД должна быть реализована следующая таблица:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| journey\_id | uuid | Идентификатор маршрута |
| rating | integer | Оценка маршрута |
| text | text | Отзыв пользователя |
| created\_at | timestamp | Дата создания |

* + - 1. Формирование маршрутов на основе истории поездок пользователей

Функция 1: Учет истории маршрутов при построении новых (этап 5).

Функция должна учитывать историю маршрутов пользователя при построении новых.

Функция должна сохранять маршруты, по которым пользователь перешел в режим «В пути».

Функция должна сохранять следующие данные о таком маршруте:

* начальная точка маршрута;
* конечная точка маршрута;
* граф маршрута;
* типы транспорта, использованные на маршруте.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| journey\_id | uuid | Идентификатор выбранного маршрута |
| start\_point | geometry | Точка А |
| finish\_point | geometry | Точка Б |
| transports | array | Типы используемого транспорта |
| created\_at | timestamp | Дата создания |

Функция 2: Отображение раннее выбранных маршрутов (этап 5).

При построении нового маршрута функция должна проверять наличие сохраненных в истории маршрутов с такими же начальными и конечными точками. Если таковые имеются, то функция должна учитывать эти маршруты, с использованием сохраненных параметров, и отображать эти маршруты вместе с остальными предлагаемыми маршрутами отдельной категорией.

* + - 1. Возможность построения маршрутов с двумя и более точками прибытия

Функция 1: Возможность добавления дополнительных точек на маршруте (этап 4).

Функция должна позволять добавлять новые точки на маршруте с последующим перестроением маршрута в режиме «в пути» в зависимости от добавленных точек.

Функция должна позволять добавить точку после конечной точки или в середину маршрута. При добавлении точки в середину маршрута, должны перестраиваться участки маршрута, в которые была добавлена новая точка, и последующие участки маршрута.

При добавлении точки в конец маршрута функция должна достроить текущий маршрут от прошлой конечной точки до добавленной.

При перестроении маршрутов должны обновиться следующие данные:

* граф маршрута;
* время в пути;
* стоимость;
* расписание движения;
* прогнозы загруженности;
* прогнозы прибытия транспортных средств.

Функция 2: Построение маршрута с двумя и более точками прибытия (этап 4).

Функция должна позволять указывать более одной точки прибытия при построении маршрута. Маршрут должен строиться от точки начала до первой точки прибытия, затем от первой точки прибытия до второй и так далее.

Функция должна позволять менять местами точки прибытия. При смене точек прибытия маршрут должен перестраиваться заново.

* + - 1. Построение маршрутов с новыми видами транспорта

Функция 1: Построение маршрутов с водным транспортом (этап 5).

Функция должна позволять выбирать водный транспорт, как вид транспорта при построении маршрута.

Должны быть реализованы маршруты, проходящие по воде между остановками. Данные маршруты должны учитываться маршрутизатором при построении маршрутов и предлагаться в виде альтернативных маршрутов.

Граф построенного маршрута должен отображаться на карте.

В карточке маршрута пользователям должна быть доступна следующая информация:

* начальная точка маршрута;
* конечная точка маршрута;
* стоимость поездки на водном виде транспорта;
* прогноз прибытия;
* список остановок водного транспорта на маршруте.

В карточке маршрута водного транспорта должна иметься возможность сообщить об ошибке, произошедшей при построении или прохождении маршрута.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| name | varchar(255) | Название остановки |
| address | varchar(255) | Адрес остановки |
| geopoint | geometry | Расположение остановки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| name | varchar(255) | Название маршрута |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| Поле | Тип данных | Описание |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| name | varchar(255) | Название маршрута |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| direction | enum('forward','backward') | Направление движения |
| geometry | geometry | Геометрия маршрута |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| stop\_id | uuid | Идентификатор остановки |
| route\_path\_id | uuid | Идентификатор пути маршрута |
| geometry | geometry | Отрезок пути до следующей остановки |
| index | integer | Номер остановки на маршруте |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_id | uuid | Идентификатор маршрута |
| dates | daterange | Время действия расписания |
| days | array | Массив индексов дней недели, когда расписание активно |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| schedule\_id | uuid | Идентификатор расписания |
| path\_stop\_id | integer | Идентификатор остановки на маршруте |
| arrival | time | Время прибытия |
| departure | time | Время отправления |

Функция 2: Использование водного транспорта в мультимодальных маршрутах (этап 5).

Функция должна позволять выбирать водный транспорт, как один из видов транспорта при построении мультимодального маршрута, а построенный участок маршрута на водном транспорте должен проходить по воде через остановки водного транспорта.

Функция 3: Отображение остановок водного транспорта (этап 5).

Для работы функции должен быть реализован новый вид объектов – остановка водного транспорта.

Остановка должна иметь краткую и полную карточку остановки, которые содержат информацию о работе остановки, маршрутах, проходящих через данную остановку, часах работы.

Функция 4: Сообщение об ошибке при прохождении маршрута на водном транспорте   
(этап 4).

Функция должна отображать окно для отправки сообщения об ошибке. Сообщение должно содержать в себе следующие данные:

* начальная точка маршрута;
* конечная точка маршрута;
* текст сообщения об ошибке.

Начальная и конечная точки маршрута должны заполняться автоматически. Данные поля должны быть неизменяемыми.

Текст сообщения должен иметь ограничение в 1000 символов включая пробелы.

Функция 5: Расчет стоимости поездки на водном виде транспорта (этап 5).

Функция должна рассчитывать стоимость поездки на водном транспорте. Рассчитанная стоимость должна отображаться в карточке маршрута для водного транспорта и в карточке мультимодального маршрута для участка маршрута на водном транспорте.

Функция 6: Построение маршрутов с канатными дорогами (этап 5).

Функция должна позволять выбирать канатные дороги как вид транспорта при построении маршрута. Должны быть реализованы маршруты, проходящие по между остановками канатной дороги. Данные маршруты должны учитываться маршрутизатором при построении маршрутов и предлагаться в виде альтернативных маршрутов.

Граф построенного маршрута должен отображаться на карте.

Пользователям должна быть доступна следующая информация:

* начальная точка маршрута;
* конечная точка маршрута;
* остановочные пункты канатной дороги на маршруте;
* расчет стоимости поездки на канатной дороге;

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| name | varchar(255) | Название остановки |
| address | varchar(255) | Адрес остановки |
| geopoint | geometry | Расположение остановки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| name | varchar(255) | Название маршрута |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| Поле | Тип данных | Описание |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| name | varchar(255) | Название маршрута |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| direction | enum('forward','backward') | Направление движения |
| geometry | geometry | Геометрия маршрута |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| stop\_id | uuid | Идентификатор остановки |
| route\_path\_id | uuid | Идентификатор пути маршрута |
| geometry | geometry | Отрезок пути до следующей остановки |
| index | integer | Номер остановки на маршруте |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| route\_id | uuid | Идентификатор маршрута |
| dates | daterange | Время действия расписания |
| days | array | Массив индексов дней недели, когда расписание активно |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| schedule\_id | uuid | Идентификатор расписания |
| path\_stop\_id | integer | Идентификатор остановки на маршруте |
| arrival | time | Время прибытия |
| departure | time | Время отправления |

Функция 7: Использование канатной дороги в мультимодальных маршрутах (этап 5).

Функция должна позволять выбирать канатные дороги, как один из видов транспорта при построении мультимодального маршрута, и строить маршрут на канатной дороге для одного из участков этого маршрута.

Функция 8: Отображение остановок канатной дороги (этап 5).

Для работы функции должен быть реализован новый вид объектов – остановка канатной дороги.

Остановка должна иметь краткую и полную карточку остановки, которые содержат информацию о работе остановки, маршрутах, проходящих через данную остановку, расписании.

Функция 9: Расчет стоимости поездки на канатной дороге (этап 5)

Функция должна рассчитывать стоимость поездки на канатной дороге. Рассчитанная стоимость должна отображаться в карточке маршрута для канатной дороги и в карточке мультимодального маршрута для участка маршрута на водном транспорте.

Функция 10: Сообщение об ошибке при прохождении маршрута на канатной дороге (этап 5).

Функция должна отображать окно для отправки сообщения об ошибке. Сообщение должно содержать в себе следующие данные:

* начальная точка маршрута;
* конечная точка маршрута;
* текст сообщения об ошибке.

Начальная и конечная точки маршрута должны заполняться автоматически. Данные поля должны быть неизменяемыми.

Текст сообщения должен иметь ограничение в 1000 символов включая пробелы.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| start\_point | geometry | Точка А |
| finish\_point | geometry | Точка Б |
| text | text | Коментарий пользователя |
| user\_id | uuid | Идентификатор пользователя |
| created\_at | timestamp | Время создания |

Функция 11: Выбор мотоцикла как вида транспорта (этап 5).

Функция должна позволять выбирать мотоцикл в качестве вида транспорта при построении маршрутов, при этом маршрут должен проходить через парковки.

При построении маршрутов пользователям, наряду с маршрутами на личном автомобиле, должны предлагаться маршруты с использованием мотоциклов с учетом особенностей транспортной системы.

* + - 1. Специальный маршрут на шаттле

Функция 1: Предоставление информации о шаттле (этап 1).

Функция должна предоставлять информацию о специальных шаттлах:

* маршрутах шаттлов;
* адресах остановок;
* режиме работы;
* условиях поездки.

Функция 2: Отображение остановок шаттлов (этап 1).

Для работы функции должен быть реализован новый вид объектов – остановка шаттла.

Остановка должна иметь краткую и полную карточку остановки, которые содержат:

* данные о маршрутах, проходящих через данную остановку;
* прогнозы прибытия шаттлов.

Функция должна позволять добавлять остановку шаттла в избранное, а также поделиться ей, построить маршрут до остановки и сообщить о проблеме.

Функция 3: Получение билета на шаттл, идущий по специальному маршруту (этап 1).

Функция должна позволять получить билет на шаттл и отображать маршрут шаттла.

Функция должна формировать QR-код, который является билетом на шаттл и действует в течение текущего дня.

Для работы функции должен быть реализован новый раздел мобильного приложения. Купленный пользователем билет на шаттл должен отображаться в виде QR-кода в реализованном разделе приложения.

Функция 4: Отображение маршрута на карте (этап 1).

Для работы функции должны быть реализованы маршруты шаттлов.

Функция должна отображать специальный маршрут шаттла на карте, проходящий через остановки шаттла.

Должна быть реализована короткая и полная карточка маршрута на шаттле. Короткая карточка должна отображать следующую информацию:

* тип транспорта;
* номер маршрута;
* конечная остановка.

Полная карточка должна отображать следующую информацию:

* остановки на маршруте;
* расписание движения;

Полная карточка должна позволять поделиться ссылкой на остановку и добавить карточку остановки в избранное, а также сообщить о проблеме.

Функция 5: Движения шаттла в режиме реального времени (этап 1).

Функция должна отображать движущийся шаттл, следующий по выбранному маршруту.

Данные для расчета движения предоставляются Заказчиком по API.

Функция 6: Сообщение об ошибке при прохождении маршрута на шаттле (этап 1).

Функция должна отображать окно для отправки сообщения об ошибке.

Функция должна позволять выбрать категорию ошибки и написать текстовое описание данной ошибки.

* + - 1. Построение маршрутов с учетом велодорожек

Функция 1: Построение маршрута на велосипеде с учетом велодорожек (этап 4).

Функция должна строить маршрут на велосипеде и участки мультимодальных маршрутов с использованием велосипеда, учитывая велодорожки, имеющиеся на маршруте.

Маршруты и участки составного маршрута должны проходить через велодорожки, если имеется такая возможность и это не ведет к значительному увеличению времени поездки.

Функция 2: Построение маршрута на самокате с учетом велодорожек (этап 4).

Функция должна строить маршрут на самокате и участки мультимодальных маршрутов с использованием самоката, учитывая велодорожки, имеющиеся на маршруте.

Маршруты и участки составного маршрута должны проходить через велодорожки, если имеется такая возможность и это не ведет к значительному увеличению времени поездки.

* + - 1. Отображение общего списка построенных маршрутов для всех видов транспорта

Функция 1: Отображение общего списка построенных маршрутов для всех видов транспорта (этап 3).

В карточке маршрута должен быть реализован раздел, отображающий маршруты на всех видах транспорта для указанных параметров маршрута.

Список маршрутов должен отображать ориентировочное время прохождения маршрута для каждого вида транспорта и его стоимость.

* + 1. Требования к разработке подсистемы загруженности дорог и пробок

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Отображение загруженности дорог (этап 3).

Функция должна отображать цветовое обозначение загруженности дорог на карте, а также индикатор общей загруженности дорог в городе в баллах.

Функция должна получать из подсистемы интеграции уровень загруженности в баллах для каждого ребра дороги и отображать его на карте отдельным слоем в различном цветовом кодировании.

Функция должна сохранять загруженность по всем участкам дороги во внутренней системе и обновлять данные не реже 1 раза в 5 минут.

Пользователь должен иметь возможность включать и выключать слой загруженности дорог.

Функция должна получать из подсистемы интеграции уровень средней загруженности по городу и отображать в виде отдельного значка.

Функция должна сохранять среднюю загруженность дороги во внутренней системе и обновлять данные не реже 1 раза в 5 минут.

При невозможности получения данных о загруженности дорог функция должна отображать загруженность на основании сохраненных данных.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| osm\_node\_id | bigint | ID ребра OSM |
| level | integer | Уровень загруженности |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 2: Учет загруженности дорог при построении маршрута на НГПТ (этап 4).

Функция должна учитывать загруженность дорог при построении маршрута на наземном городском транспорте.

Функция должна предлагать наиболее оптимальные по времени маршруты с учетом средней скорости движения по ним для наземного городского транспорта.

Функция 3: Учет загруженности дорог при построении маршрута на личном транспорте   
(этап 3).

Функция должна учитывать загруженность дорог при построении маршрута на личном автомобиле, каршеринге и мотоцикле.

Функция должна предлагать наиболее оптимальные по времени маршруты с учетом загруженности дорог.

Функция должна предлагать пользователю не менее 3 маршрутов на выбор при построении маршрута на личном автомобиле.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| osm\_node\_id | bigint | ID ребра OSM |
| avg\_speed | float | Средняя скорость |
| avg\_reverse\_speed | float | Средняя скорость в обратном направлении |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 4: Расчет времени в пути на НГПТ с учетом загруженности дорог (этап 4).

Функция должна рассчитывать время в пути, учитывая загруженность дорог на маршруте наземного городского транспорта. Для расчета времени должна использоваться средняя скорость наземного городского транспорта на маршруте, вычисляемая из телеметрических данных.

Телематические данные предоставляются Заказчиком через API.

Функция 5: Расчет времени в пути на личном транспорте с учетом загруженности дорог   
(этап 3).

Функция должна рассчитывать время в пути, учитывая загруженность дорог на маршруте личного транспорта. Для расчета времени должны использоваться данные о загруженности дорог и средней скорости движения по ним, получаемые из подсистемы интеграции.

Данные о загруженности дорог предоставляются Заказчиком через API.

* + 1. Требования к разработке подсистемы оплаты банковской картой

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Привязка банковской карты (этап 2).

Функция должна позволять привязывать банковскую карту в личном кабинете пользователя.

Функция должна открывать окно предпроцессинга, позволяющее ввести данные банковской карты. После ввода данных функция должна получать и сохранять полученный платежный токен пользователя.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| pan | varchar(255) | Маскированный номер карты |
| expiry\_date | varchar(255) | Дата окончания действия карты |
| processing | varchar(255) | Строковый идентификатор процессинга |
| cardholder\_name | varchar(255) | Фамилия и имя владельца карты |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 2: Выбор способа оплаты услуги (этап 2).

Функция должна позволять выбирать способ оплаты для безналичного расчета.

Функция должна предоставлять пользователю выбор из следующих способов оплаты при покупке услуг:

* оплата банковской картой, привязанной в приложении «Московский транспорт»;
* оплата банковской картой, привязанной в приложении «Аpple pay»;
* оплата банковской картой, привязанной в приложении «Google pay»;

Функция 3: Оплата привязанной картой (этап 2).

Функция должна позволять оплачивать услуги, доступные в модуле оплаты, при помощи привязанной банковской карты.

Функция должна позволять холдировать стоимость услуги с привязанной пользовательской банковской карты.

Функция должна позволять списывать захолдированные средства с привязанной банковской карты.

Функция должна позволять отменять холдирование средств на привязанной банковской карте.

Функция должна позволять списывать средства с привязанной карты без холдирования средств.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| external\_id | varchar(255) | Идентификатор во внешней системе |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| card\_id | uuid | Идентификатор карты |
| operation\_type | enum('purchase', 'authorize', 'complete', 'reverse') | Тип операции (списание, холдирование, списание захолдированных средств, отмена холдирования) |
| amount | int | Сумма в минимальных единицах валюты |
| description | text | Описание операции |
| currency | int | Цифровой код валюты операции |
| status | enum('pending', 'approved', 'rejected', 'timeout', 'error') | Статус операции (в ожидании, одобрена, отклонена, таймаут, ошибка системы) |

Функция 4: Отображение списка привязанных банковских карт (этап 2).

Функция должна отображать список привязанных пользователем банковских карт.

Список должен позволять просматривать информацию о прикрепленных картах и отображать последние 4 цифры банковской карты.

Функция 5: Удаление привязанной банковской карты (этап 2).

Функция должна позволять пользователю удалять привязанную банковскую карту.

При удалении банковской карты должна удаляться вся информация о карте, включая платежный токен.

Функция 6: История совершенных операций (этап 2).

Функция должна отображать список совершенных операций по покупке услуг и абонементов, совершенных с помощью безналичного расчета в приложении.

Функция должна отображать следующие данные об операции:

* последние 4 цифры привязанной карты, с которой проводилась оплата;
* дата проведения операции;
* статус проведения операции;
* сумма платежа.
  + 1. Требования к разработке подсистемы карты метро

Функция 1: Рекомендации по выбору вагона и выхода (этап 1).

Функция должна отображать информацию об удобных вагонах для пересадок и номерах вестибюлей для выхода в город в карточке маршрута.

Функция при построении маршрута на метро должна предлагать вагон, который находится ближе всего к переходу, в целях сокращения времени перехода между станциями.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| transfer\_id | uuid | Идентификатор пересадки |
| carriage\_number | integer | Оптимальный номер вагона для пересадки |
| reverse\_carriage\_number | integer | Оптимальный номер вагона для пересадки в обратную сторону |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 2: Карта и внутренняя навигация (этап 1).

Функция должна быть выполнена в виде отдельного режима с картой метро, позволяющего строить маршрут только на метро, МЦК, МЦД, и предоставляющего информацию о станциях, переходах и выходах из метро.

На карте должны отображаться все действующие станции метро, МЦК, МЦД, линии метро с соответствующей им цветовой индикацией, переходы между станциями метро, МЦК, МЦД.

Функция должна позволять обновлять карту метро в случае внесения в нее изменений:

* закрытие станций;
* изменений в работе станции (ограничений);
* добавление станций.

Функция должна отображать закрытые станции метро специальной пиктограммой.

Функция должна отображать станции с изменениями в работе специальной пиктограммой.

Должна быть реализована короткая и полная карточка станции метро, которая открывается при выборе станции метро.

Короткая карточка должна предоставлять информацию о загруженности станции, позволять выбрать станцию как начало или конец маршрута, добавить станцию в избранное. Короткая карточка должна отображать данные о закрытии или изменении в работе станции с возможностью получения более подробной информации о данных событиях.

Полная карточка должна содержать более подробную информацию о станции: подробная информация о загруженности, информация об инфраструктуре станции, количество станций и время движения до конечных станций от выбранной по текущей линии метро.

Функция должна позволять менять уровень приближения отображаемой карты метро.

Функция 3: Отображение ближайшей станции к моему местоположению (1 этап).

Функция должна получать текущую геолокацию пользователя с его устройства.

Функция должна определять и отображать ближайшую станцию метро к текущим координатам пользователя.

Функция 4: Построение маршрутов на метро (этап 1).

Функция должна позволять строить маршруты между станциями метро на интерактивной карте метро.

Функция должна предлагать несколько возможных маршрутов между начальной и конечной точкой маршрута. Первым маршрутом должен отображаться самый оптимальный маршрут.

При расчете времени прохождения маршрута должны учитываться: время, необходимое на прохождение перегонов между станциями, и время, необходимое на переход между станциями.

Должна быть реализована краткая и полная карточка маршрута на метро. Краткая карточка должна отображать следующую информацию:

* время на прохождение маршрута;
* время начала и конца движения по маршруту;
* количество пересадок;
* стоимость поездки по маршруту;
* линии метро, через которые проходит маршрут и пересадки между ними;

Полная карточка маршрута должна отображать дополнительную информацию о маршруте:

* время движения по участкам маршрута;
* количество станций на участках маршрутов;
* загруженность станций, на которых пользователь выходит или совершает пересадки;
* рекомендуемые вагоны.

Полная карточка должна предоставлять возможность отправки обращений в техническую поддержку.

Карточки маршрутов на метро должны позволять переключаться на другие построенные маршруты.

При построении маршрутов должны учитываться закрытия станций метро.

Должна быть реализована возможность сменить направление движения по построенному маршруту.

Функция 5: Добавление станций метро в избранное (этап 1).

Должна быть реализована функция добавления выбранных на карте метро станций метро в избранное.

Избранные станции должны предлагаться пользователю при построении маршрута отдельным списком. Для избранных станций в списке должна отображаться следующая информация:

* название станции;
* название линии метро;
* цвет линии метро;
* номер линии метро.

При выборе в списке избранной станции она должна отмечаться как начальная или конечная станция маршрута, в зависимости от того на каком этапе построения маршрута она была выбрана.

Функция должна позволять удалять станции метро из избранного повторным действием добавления в избранное.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| stop\_id | uuid | Идентификатор избранной остановки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 6: Последний маршрут (этап 1).

Функция должна сохранять маршруты, построенные пользователем на интерактивной карте метро.

При построении нового маршрута предыдущий построенный маршрут должен заменяться новым.

При построении маршрутов пользователю должен предлагаться последний построенный маршрут. Предложенный маршрут должен содержать следующую информацию:

* наименование начальной станции;
* цвет линии начальной станции;
* номер линии начальной станции;
* наименование конечной станции;
* цвет линии конечной станции;
* номер конечной линии.

Функция 7: Список недавних станций (этап 1).

Функция должна сохранять последние 15 станций, выбранные пользователем в качестве начальной или конечной точки маршрута.

Функция должна формировать список недавних станций из сохраненных станций. Первой в списке размещается последняя выбранная пользователем станция.

Список должен предоставлять следующие данные о сохраненных станциях:

* название станции;
* название линии метро;
* цвет линии метро;
* номер линии метро.

Функция 8: Поиск по станциям (этап 1).

Функция должна позволять выполнять поиск станций по их названию при построении маршрута. При выборе начальной или конечной точки маршрута должно открываться окно поиска станции, состоящее из:

* поискового поля;
* последнего маршрута (функция 6);
* списка избранных станций (функция 5);
* списка недавних станций (функция 7).

Поиск станций должен осуществляться по названиям станций метро, путем ввода их названий в поисковое поле. Поиск должен начинаться по первой введенной букве наименования. Список поисковых результатов должен быть отсортирован по линиям метро и отображать следующие данные:

* название станции;
* название линии метро;
* цвет линии метро;
* номер линии метро.

Функция 9: Закрытие станций (этап 1).

Функция должна позволять отмечать станции метро как закрытые через интерфейс администратора Системы.

В интерфейсе администратора должна быть реализована возможность выбора станций метро и присвоения им статуса закрытых или открытых. При присвоении статуса закрытия должен быть указан один из типов закрытия станции:

* закрыт переезд через станцию в одно направление;
* закрыт переезд через станцию в оба направления;
* переезд через станцию открыт, но станция закрыта в оба направления;
* переезд через станцию открыт, но станция закрыта в одном из направлений;
* станция открыта, переезд открыт, но закрыты какие-то определенные вестибюли (только на вход, только выход, оба включены или оба выключены);
* закрыт переход с этой станции на другую.

При закрытии станции, кроме типа закрытия, должна быть указана следующая информация о закрытии:

* дата начала закрытия;
* дата окончания закрытия;
* сообщение о работах.

Сообщение о закрытии будет отображать пользователям более подробную информацию о работах и состоять из следующих данных:

* дата начала показа;
* дата окончания показа;
* заголовок сообщения;
* текст сообщения.

Сообщение должно отображаться в указанном диапазоне дат.

Должна быть реализована возможность прикрепления своего сообщения для каждой локализации.

Должна иметься возможность прикрепить несколько разных сообщений для одного закрытия.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| path\_stop\_id | integer | Идентификатор остновки на маршруте |
| mode | integer | Режим закрытия остановки |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| closed\_from | timestamp | Время начала закрытия |
| closed\_to | timestamp | Время окончания закрытия |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| block\_id | uuid | Идентификатор перекрытия |
| text | text | Информационный текст |
| title | varchar(255) | Заголовок перекрытия |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| show\_from | timestamp | Время начала показа |
| show\_until | timestamp | Время окончания показа |

Функция 10: Учет закрытых станций при построении маршрута (этап 1).

Функция должна учитывать закрытые станции метро при построении маршрута на метро.

Функция должна не позволять строить маршруты до закрытых станций метро и через них, если они перекрывают дальнейшее движение по линии.

Функция должна строить маршруты в объезд закрытых станций, если имеется такая возможность.

* + 1. Требования к разработке подсистемы загруженности транспорта
       1. Отображение загруженности станций

Функция 1: Получение данных о загруженности станций и вагонов метро (этап 1).

Функция должна иметь возможность получать данные о загруженности станций и вагонов метрополитена.

Функция должна получать информацию о загруженности станций метро из внешнего источника и преобразовывать полученное количество баллов в 3-х бальную систему (низкая, средняя и высокая). Функция должна сохранять информацию о загруженности станций метро в интервалах за каждые 10 минут и вычислять максимальную загруженность часа.

Функция должна получать информацию о загруженности вагонов из внешнего источника в формате:

* идентификатор состава;
* предыдущая станция;
* следующая станция;
* время до прибытия в секундах;
* количество вагонов;
* уровень загруженности вагонов.

Функция должна определять направление движения поезда в пределах ветки метро.

Данные о загруженности станций предоставляются Заказчиком.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| subway\_id | uuid | Идентификатор станции |
| date | date | Календарный день |
| data | json | Данные о баллах загруженности по 10-минуткам |
| type | enum('entrance', 'exit', 'station') | Тип загруженности(на вход, на выход, на станции) |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| current\_subway\_id | uuid | Идентификатор станции |
| prev\_subway\_id | uuid | Идентификатор предыдущей станции |
| date | date | Календарный день |
| data | json | Данные о баллах загруженности по 10-минуткам |
| carriage\_number | integer | Номер вагона |

Функция 2: Отображение информации об уровне загруженности станций метрополитена и вагонов (этап 1).

Функция должна отображать данные об уровне загруженности станций метрополитена в виде индикатора загруженности.

Функция должна отображать в интерфейсе пользователя график загруженности станций, в течение дня с шагом 60 минут, начиная с 6:00 утра и заканчивая в 01:00 ночи.

Функция должна позволять выбрать точное время и день для отображения загруженности станции. Загруженность станции не в текущее время основывается на прогнозе загруженности для будущих часов, которая строится на основании статистики загруженности данной станции или на основе истории загруженности для уже прошедшего времени.

Функция должна отображать индикатор загруженности станции на текущий момент времени. Функция должна отображать загруженность станций как в карточке станции метро, так и при построении маршрута на метро. При построении маршрута на метро функция должна отображать загруженность на каждой станции, где пользователь входит или выходит со станции.

Функция должна отображать пользователю массив прибывающих поездов в формате: направление движения поезда, количество вагонов, время прибытия, загруженность каждого вагона.

Функция должна отображать все возможные направления движения поездов в карточке станции и только одно определенное направление при построении маршрута.

Функция должна обновлять данные о загруженности станций метро не реже 1 раза в сутки. Функция должна обновлять данные о загруженности прибывающих поездов не реже 1 раза в минуту.

Функция 3: Cбор обратной связи о загруженности метро (этап 1).

Функция должна получать от пользователей информацию о соответствии полученных данных по прогнозу загруженности и реальной загрузке в виде значения: соответствует или не соответствует. Функция должна ограничивать получение обратной связи о загруженности ТС от одного пользователя в количестве 5 значений в минуту.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| subway\_id | uuid | Идентификатор станции |
| match | boolean | Соответствие прогноза |
| created\_at | timestamp | Дата создания |

* + - 1. Информация о загруженности наземного транспорта

Функция 1: Получение данных о загруженности (этап 1).

Функция должна иметь возможность получать информацию об уровне загруженности маршрутов наземного транспорта и отображать в виде индикатора загруженности.

Функция должна получать данные из разных источников по уровню загруженности наземного транспорта в формате реального времени и прогноза по загруженности.

Для отображения загруженности в реальном времени функция должна сохранять следующие параметры:

* “bus\_id”: string;
* “eventTime”: string;
* “route\_id”: string;
* “stop\_id”: string;
* “total\_counter”: int32;
* “current\_in\_bus”: int32;
* “in”: int32;
* “out”: int32;
* “not\_payed\_counter”: int32;
* “latitude”: float64;
* “longitude”: float64.

Функция должна преобразовывать полученные данные по входам и выходам пассажиров на остановках в режиме реальном времени в 3-балльную шкалу загруженности (низкая, средняя и высокая) и учитывать данные о вместимости ТС при расчете. При отсутствии данных о вместимости ТС, принимать согласованное среднее значение вместимости.

Для отображения прогноза загруженности ТС функция должна сохранять следующую информацию:

* Количество пассажиров в 10-ти минутных интервалах на 7 календарных дней;
* ID маршрута;
* ID пути маршрута;
* начальная остановка;
* конечная остановка.

Функция должна преобразовывать полученные данные в 3-балльный формат (низкая, средняя и высокая) и учитывать данные о вместимости ТС при расчете.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| path\_id | uuid | Идентификатор пути маршрута |
| date | date | Календарный день |
| hour | integer | Час |
| minute | integer | Минутный интервал |
| data | json | Данные о загруженности по остановкам на пути |

Функция 2: Отображение данных о загруженности (этап 1).

Функция должна отображать данные о загруженности в карточке ТС, а также на карте, в виде индикатора загруженности.

Функция 3: Cбор обратной связи о загруженности ТС (этап 1).

Функция должна получать от пользователей информацию о соответствии полученных данных по прогнозу загруженности и реальной загрузке в виде значения: соответствует или не соответствует. Функция должна ограничивать получение обратной связи о загруженности ТС от одного пользователя в количестве 5 значений в минуту.

Данные о загруженности транспорта предоставляются Заказчиком.

* + - 1. Информация о загруженности парковок

Функция 1: Получение данных о загруженности (этап 1).

Функция должна получать информацию о загруженности городских парковок из внешнего источника и преобразовывать полученное количество баллов в 3-х балльную систему.

Для отображения прогноза загруженности парковок функция должна сохранять следующую информацию:

* ID парковки;
* ID зоны;
* время;
* количество мест;
* количество автомобилей.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| line\_id | uuid | Идентификатор парковочной линии |
| date | date | Календарный день |
| data | json | Данные о баллах загруженности по 10-минуткам |
| created\_at | timestamp | Дата создания |

Функция 2: Отображение данных о загруженности (этап 1).

Функция должна отображать в интерфейсе пользователя график загруженности парковок в течение дня и позволять выбирать другой день недели в пределах 7 календарных дней.

Функция должна отображать индикатор загруженности парковки на текущий момент времени.

Данные о загруженности парковок предоставляются Заказчиком.

Функция 3: Cбор обратной связи о загруженности парковки (этап 1).

Функция должна получать от пользователей информацию о соответствии полученных данных по прогнозу загруженности и реальной загрузке в виде значения: соответствует или не соответствует. Функция должна ограничивать получение обратной связи о загруженности ТС от одного пользователя в количестве 5 значений в минуту.

* + 1. Требования к разработке подсистемы перекрытий

Функция 1: Отображение геометрии перекрытий на карте (этап 3).

Функция должна отображать перекрытия на карте приложения в виде отдельного слоя на карте.

Функция должна позволять отображать следующую информацию о перекрытии:

* название перекрытия;
* тип перекрытия;
* геометрия перекрытия;
* ограничение скорости;
* срок действия перекрытия;
* описание.

При истечении срока действия перекрытия его отображение должно автоматически отключаться.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор перекрытия |
| polygon | geometry | Геометрия перекрытия |
| type | enum('full', 'partial') | Тип перекрытия (полное, частичное) |
| speed\_limit | integer | Ограничение скорости |
| date\_from | timestamp | Дата начала действия |
| date\_to | timestamp | Дата окончания действия |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| description | text | Описание |

* + 1. Требования к разработке подсистемы такси

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Выбор из нескольких операторов такси (этап 1).

Функция должна позволять выбирать одного из нескольких операторов при построении маршрута на такси.

Функция выбора при построении маршрутов на такси должна отображать:

* наименования операторов;
* минимальную стоимость поездки для выбранного оператора;
* наличие промокода для данного приложения;
* расстояние поездки;
* тарифы для выбранного оператора.

После выбора оператора, должна обновляться вся информация о поездке на такси согласно данным, получаемым от выбранного оператора.

Функция должна случайным образом менять порядок отображения операторов такси, представленных в приложении.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| \_id | ObjectID | Идентификатор |
| title | string | Название |
| alias | string | Строковый идентификатор |
| translations | object | Объект типа translations для поля title |
| tariffs | Array | Массив объектов типа tariff |
| options | Array | Массив объектов типа option |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| en | string | Английский |
| fr | string | Французский |
| zh | string | Китайский |
| de | string | Немецкий |
| es | string | Испанский |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| alias | string | Строковый идентификатор тарифа |
| title | string | Название тарифа |
| isDefault | boolean | Является тарифом по умолчанию |
| additional | object | Дополнительные данные тарифа (индивидуально для разных операторов) |
| translations | object | Объект типа translations для поля title |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| alias | string | Строковый идентификатор |
| title | string | Название |
| translations | object | Объект типа translations для поля title |

Функция 2: Отображение цены (этап 1).

Функция должна отображать цену поездки по выбранному пользователю маршруту для указанного оператора, рассчитанную на текущий момент.

Функция 3: Отображение маршрута такси (этап 1).

Функция должна отображать маршрут такси от его текущего местоположения до точки начала поездки на карте.

Функция должна отображать:

* маршрут между начальной и конечной точками поездки на карте, после того как пользователь сел в машину, и водитель подтвердил начало поездки;
* предполагаемое время в пути;
* расстояние до конечной точки маршрута.

Функция должна отображать текущее местоположение машины такси на карте на всех этапах поездки.

Функция 4: Отображение карточки поездки (этап 2).

Функция должна отображать карточку поездки, в которой представлена информация о текущем активном заказе такси. В карточке должна быть следующая информация о поездке:

* наименование оператора такси;
* адрес начальной точки маршрута;
* адрес конечной точки маршрута;
* время начала заказа;
* время подачи машины;
* стоимость поездки;
* предполагаемое время поездки;
* модель машины;
* номер машины;
* цвет машины.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| external\_id | varchar(255) | Идентификатор заказа во внешней системе |
| user\_id | uuid | ID пользователя |
| service\_alias | varchar(255) | Строковый идентификатор оператора такси |
| time\_start | timestamp | Время начала поездки |
| time\_finish | timestamp | Время окончания поездки |
| point\_start | geometry | Точка отправления |
| point\_finish | geometry | Точка прибытия |
| price | integer | Цена поездки |

Функция 5: Выбор карты для оплаты (этап 2).

Функция должна позволять выбирать карту для оплаты из списка привязанных банковских карт к профилю пользователя. Выбор должен производиться в момент заказа.

Функция 6: Заказ такси (этап 2).

Функция должна позволять заказывать услуги такси у выбранного оператора такси, не переходя в другие приложения.

Функция должна позволять указывать начальную и конечную точку маршрута.

Функция должна позволять указывать дополнительные параметры заказа, поддерживаемые выбранным оператором такси, например:

* детское кресло;
* перевозка животных;
* перевозка инвентаря;
* не курящий водитель;
* и прочие.

Полный список зависит от дополнительных услуг, предоставляемых оператором такси.

Функция должна позволять выбирать тариф поездки.

Функция должна позволять выбирать способ оплаты поездки (функция 5).

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор заказа |
| user\_id | uuid | Идентификатор пользователя |
| external\_id | uuid | Идентификатор заказа во внешней системе |
| service\_alias | varchar(255) | Строковый идентификатор оператора такси |
| time\_start | timestamp | Время начала поездки |
| time\_finish | timestamp | Время окончания поездки |
| point\_start | geometry | Точка отправления |
| point\_finish | geometry | Точка прибытия |
| address\_start | varchar(255) | Адрес отправления |
| address\_finish | varchar(255) | Адрес прибытия |
| price | integer | Цена поездки |
| params | array | Дополнительные параметры поездки |
| car\_description | varchar(255) | Описание машины |
| status | enum('pending', 'in\_progress', 'cancelled', 'finished') | Статус заказа |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| comment | text | Комментарий пользователя к заказу |
| card\_id | uuid | Идентификатор привязанной карты пользователя |
| payment\_type | enum('card', 'google\_pay', 'apple\_pay') | Способ оплаты |

Функция 7: Отмена такси (этап 2).

Функция должна позволять отменять текущий активный заказ такси.

Функция при отказе должна позволять указывать причину отмены заказа или комментарий, если данный оператор такси предоставляет такой функционал.

При отмене поездки должны сохраняться следующие данные об отмененной поездке:

* дата и время отмены;
* причина отмены;
* комментарий к отмененной поездке.

Функция 8: Завершение поездки на такси (этап 2).

Функция должна сохранять информацию о завершенной поездке.

Функция должна отображать информацию о завершенном заказе, включая информацию о поездке:

* адрес отправления;
* адрес прибытия;
* время в пути;
* стоимость поездки.

Функция 9: История поездок на такси (этап 2).

Функция должна отображать список завершенных поездок на такси.

Функция должна отображать информацию о завершенных поездках пользователя в его профиле:

* адрес отправления;
* адрес прибытия;
* время в пути;
* способ оплаты поездки;
* стоимость поездки.

Функция 10: Список отмененных поездок на такси (этап 2).

Функция должна отображать список отмененных поездок на такси.

Функция должна отображать информацию об отмененных поездках пользователя в его профиле:

* дата и время отмены;
* причина отмены;
* комментарий к отмененной поездке.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| order\_id | uuid | Идентификатор заказа |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| reason | varchar(255) | Причина отмены |
| comment | text | Комментарий отмены |

* + 1. Требования к разработке подсистемы каршеринга

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Выбор из нескольких операторов (этап 1).

Функция должна позволять выбирать одного из нескольких операторов при построении маршрута на каршеринге. Функция должна позволять фильтровать операторов каршеринга, предлагаемых в вариантах маршрутов, с учетом заданных пользователем параметров.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| alias | varchar(255) | Строковый идентификатор во внешней системе |
| name | varchar(255) | Название |
| android\_app\_id | varchar(255) | Идентификатор приложения Android |
| ios\_app\_id | varchar(255) | Идентификатор приложения iOS |
| site\_url | varchar(255) | Ссылка на сайт оператора |
| rules\_url | varchar(255) | Ссылка на правила сервиса |
| contact\_number | varchar(255) | Контактный номер телефона |
| logo | uuid | Идентификатор логотипа |
| active | boolean | Активность |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| minute\_cost | float | Средняя стоимость минуты аренды |
| waiting\_cost | float | Средняя стоимость минуты ожидания |
| id | integer | Идентификатор |

Функция 2: Отображение стоимости аренды (этап 1).

Функция должна позволять отображать в карточке каршеринга стоимость минуты аренды и стоимость минуты ожидания.

Функция 3: Переход в приложение оператора (этап 1).

Функция должна позволять генерировать ссылку перехода на приложение оператора с передачей параметров выбранного автомобиля.

* + 1. Требования к развитию подсистемы карты и поиска
       1. Оптимизация поискового алгоритма

Функция 1: Учет геолокации пользователя (этап 4).

Функция должна учитывать текущее местоположение пользователя при поисковом ранжировании.

Функция должна ранжировать объекты, находящиеся в радиусе не более 1000 метров от пользователя с наивысшим приоритетом, объекты в радиусе от 1000 до 5000 метров от пользователя должны ранжироваться в зависимости от их расстояния, а объекты в радиусе более 5000 метров должны иметь одинаковый приоритет ранжирования.

Функция должна отображать результаты ранжирования при поисковых запросах объектов пользователя.

Функция 2: Распознавание сокращений (этап 4).

Функция должна позволять распознавать общепринятые сокращения адресов (город, дом, улица, квартира и прочие сокращения).

Функция должна выделять из поискового запроса пользователя сущности, относящиеся к сокращениям, и учитывать их как эквивалент сокращаемым словам.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| full\_term | varchar(255) | Оригинальное слово |
| short\_terms | array | Сокращения |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 3: Разделение результатов поиска по категориям (этап 4).

Функция должна позволять разделять результаты поиска по заданным категориям, например: бары, рестораны, отели, заправки и т.д.

Функция 4: Учет истории запросов пользователя (этап 5).

Функция должна учитывать предыдущие запросы пользователя при выполнении новых поисковых запросов.

Функция должна сохранять поисковые запросы пользователя и результат, выбранный им по результатам поиска. Функция должна хранить историю поиска не менее 30 календарных дней. Функция должна сохранять поисковый запрос только при наличии выбранного пользователем результата поиска.

Функция должна при каждом поисковом запросе проверять его наличие в сохраненных запросах. Если в сохраненных запросах уже имеется такой запрос, то функция должна ранжировать выбранный пользователем результат с наивысшим приоритетом. Если пользователь выбрал результат, отличный от предыдущего, определенное количество раз, то функция должна обновить связку поисковый запрос – результат.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| query | varchar(255) | Итоговый запрос |
| query\_parts | array | Предварительные запросы |
| object\_id | uuid | Идентификатор выбранного объекта |
| hits | integer | Кол-во выборов объекта |

Функция 5: Определение именованных сущностей (этап 5).

Функция должна определять именованные сущности (слово или словосочетание, обозначающее предмет или явления определенной категории, например: наименования организаций, названия парков и т.п.) в поисковых запросах.

При введении поискового запроса функция должна проверять наличие слов, которые могут быть не только наименованиями объектов, но и являться именованными сущностями. При наличии таких сущностей функция должна предоставлять в результатах поиска не только объекты с таким названием, но и объекты, соответствующие данной именованной сущности.

Функция 6: Поиск маршрутов (этап 4).

Функция должна выделять маршруты из общего поискового результата, если поисковый запрос может указывать как на адрес, так и на маршрут. Также, при поисковом запросе маршрута должно учитываться местоположение пользователя, в качестве результата первым должен предлагаться маршрут, который находится ближе к пользователю. Например, если пользователь находится в МО, то первым ему должен быть предложен маршрут автобуса МО с таким номером, а затем уже другие маршруты.

Функция должна учитывать различные ключевые слова и отображать маршруты соответствующего типа, такие как:

* автобус;
* троллейбус;
* трамвай.

Функция 7: Поиск по категориям организаций (этап 5).

Функция должна позволять разделять результаты поиска по заданным категориям.

Функция 8: Хранение истории запросов пользователя (этап 4).

Функция должна иметь возможность хранить пользовательские поисковые запросы, для дальнейшего учета их при новых запросах.

Функция должна хранить всю поисковую цепочку:

* весь процесс написания поискового запроса до выбора первого результата;
* поисковый запрос на момент выбора первого результата;
* поисковый запрос на момент выбора следующего результата, если пользователь вернулся обратно;
* выбранный конечный результат.

Функция 9: Выделение введенного фрагмента текста в результатах поиска (этап 4).

Функция должна выделять фрагменты, которые совпадают с поисковым запросом, в результатах поиска.

Функция 10: Поиск объектов (этап 5).

Функция должна быть доработана таким образом, чтобы иметь возможность искать новые добавленные объекты, а именно: парковки, туалеты, парки, водный транспорт и другие.

Список типов, отображаемых в поиске, должен определяться в мобильном приложении.

Функция 11: Фильтры поиска организаций (этап 5).

Функция должна позволять накладывать фильтры на результаты поиска организаций, например: «Сейчас работает», «Для маломобильных», и прочие.

Функция 12: Рейтинг организаций (этап 5).

Функция должна позволять оценивать организации, определять посещение пользователем организации и отправлять PUSH-уведомление, с предложением оценить посещенное место.

Функция должна сохранять оценку, выставленную пользователем данному организации.

Функция 13: Учет рейтинга организаций в поиске (этап 5).

Функция должна учитывать рейтинг организаций при формировании результатов поиска по организациям.

Функция должна ранжировать результаты поиска организаций, предоставляя больший приоритет организации с наивысшим рейтингом. Функция должна выполнять ранжирование при поиске внутри категории объекта.

Функция 14: Передача полигона поискового объекта (этап 5).

Функция должна возвращать полигон искомого объекта. Полигон должен быть представлен в формате GeoJSON.

Передаваемый полигон может иметь мультиполигон в случае полигонов используемых на высоких уровнях геокодирования (города, страна).

Функция 15: Фильтрация в поиске объектов OSM (этап 5).

Функция должна позволять фильтровать объекты по настраиваемым параметрам при импорте из OSM.

В качестве настраиваемых параметров должны использоваться специальные теги для объектов OSM.

Функция 16: Локализация объектов в поиске (этап 5).

Функция должна позволять менять язык, на котором отображаются объекты OSM в поиске, их карточки, а также адреса, обозначения улиц, городов на карте.

* + - 1. Отображение объектов на карте

Функция 1: Слой с пунктами переработки мусора (этап 5).

Функция должна представлять собой отдельный активируемый слой объектов, отображающих на карте пункты переработки мусора, а также поддерживать кластеризацию объектов на разных уровнях приближения карты.

* Функция должна отображать информацию о пункте переработке мусора в карточке объекта:
* название объекта;
* тип объекта;
* адрес объекта;
* ближайшие станции метро;
* описание.

Информация предоставляется Заказчиком через API.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| type | varchar(255) | Тип объекта |
| address | varchar(255) | Адрес объекта |
| geopoint | geometry | Точка расположения объекта |
| description | text | Описание объекта |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 2: Слой с АЗС (этап 5).

Функция должна представлять собой отдельный, активируемый слой объектов, отображающих на карте АЗС, а также поддерживать кластеризацию объектов на разных уровнях приближения карты.

Функция должна отображать информацию о выбранной АЗС в карточке объекта:

* название объекта;
* тип объекта;
* адрес объекта;
* ближайшие станции метро;
* описание.

Информация предоставляется Заказчиком через API.

Функция 3: Слой с пунктами еды (этап 5).

Функция должна представлять собой отдельный активируемый слой объектов, отображающих на карте рестораны, кафе, бары и другие пункты общественного питания, а также поддерживать кластеризацию объектов на разных уровнях приближения карты.

Функция должна отображать информацию в карточке выбранного объекта:

* название объекта;
* тип объекта;
* адрес объекта;
* ближайшие станции метро;
* описание.

Функция 4: Слой с общественными туалетами (этап 5).

Функция должна представлять собой отдельный активируемый слой объектов, отображающих на карте общественные туалеты, а также поддерживать кластеризацию объектов на разных уровнях приближения карты.

Функция должна отображать информацию в карточке выбранного объекта:

* название объекта;
* тип объекта;
* адрес объекта;
* ближайшие станции метро;
* описание.

Информация предоставляется Заказчиком через API.

Полный перечень элементов интеграции должен быть уточнен на этапе ЧТЗ.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| type | varchar(255) | Тип объекта |
| address | varchar(255) | Адрес объекта |
| geopoint | geometry | Точка расположения объекта |
| description | text | Описание объекта |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 5: Слой велодорожек (этап 4).

Функция должна предоставлять собой отдельный активируемый слой объектов, отображающих на карте велодорожки, а также поддерживать кластеризацию объектов на разных уровнях приближения карты.

Функция 6: Слой велобоксов (этап 4).

Функция должна предоставлять собой отдельный активируемый слой объектов, отображающих на карте специализированные велосипедные парковки – велобоксы, а также поддерживать кластеризацию объектов на разных уровнях приближения карты

Функция 7: Слой сервисов велосипедов и специализированных велосипедных магазинов (этап 5)

Функция должна предоставлять собой отдельный активируемый слой объектов, отображающих на карте сервисы велосипедов и специализированные велосипедные магазины, а также поддерживать кластеризацию объектов на разных уровнях приближения карты

* + 1. Требования к разработке подсистемы парковок

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Отображение парковок на карте (этап 1).

Функция должна отображать на карте парковки с различным обозначением разных типов парковки (коммерческая или городская) и указанием стоимости для городских парковок.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| number | integer | Номер парковки |
| external\_id | integer | Идентификатор во внешней системе |
| description | text | Описание |
| price | integer | Цена в час |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| zone\_id | uuid | Идентификатор парковочной зоны |
| name | varchar(255) | Название |
| external\_id | integer | Идентификатор во внешней системе |
| address | varchar(255) | Адрес |
| total\_spaces | integer | Кол-во парковочных мест |
| hahdicapped\_spaces | integer | Кол-во мест для маломобильных |
| free\_spaces | integer | Кол-во свободных мест |
| location | geometry | Полигон парковочной линии |
| center | geometry | Центр парковочной линии |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| category\_id | integer | Идентификатор категории во внешней системе |
| description | text | Описание |

Функция 2: Отображение слоя городских платных парковок (этап 1).

Функция должна представлять собой отдельный, активируемый слой объектов, отображающихся на карте, и отображать информацию о выбранной парковке в формате: название парковки, тип парковки, стоимость парковок, время их работы, количество парковочных мест, ближайшие станции метро, а также поддерживать их учет при построении маршрутов на автомобиле и каршеринге.

Функция 3: Отображение доступности мест (этап 2).

Функция должна отображать доступность парковочных мест в карточке парковки, а также стоимость минуты парковки.

Функция 4: Расчёт стоимости парковки (этап 2).

Функция должна рассчитывать стоимость первого часа парковки на личном транспорте при построении маршрута.

Функция 5: Оплата парковки (этап 2).

Функция должна позволять оплачивать парковку, выбранную в приложении. Функция должна осуществлять расчет стоимости парковки на выбранной зоне с выбранной длительностью.

Для формирования запроса на оплату, функция должна передавать следующие параметры:

* идентификатор парковочного места;
* номер паркующегося автомобиля;
* длина парковочной сессии в минутах.

В ответ на передаваемые параметры функция должна получать следующие параметры:

* идентификатор парковочной сессии;
* идентификатор парковочного места;
* время начала парковки в формате yyyy-mm-dd hh:mm:ss;
* ожидаемое время окончания парковки в формате yyyy-mm-dd hh:mm:ss;
* номер паркующегося автомобиля.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| line\_id | uuid | Идентификатор парковочной линии |
| parking\_place\_id | varchar(255) | Идентификатор парковочного места |
| user\_id | uuid | Идентификатор пользователя |
| car\_id | uuid | Идентификатор автомобиля |
| length | integer | Длительность в минутах |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| started\_at | timestamp | Время начала сессии |

Функция 6: Отображение текущей сессии парковки (этап 2).

Функция должна отображать текущую сессию парковки в личном кабинете пользователя.

Функция 7: Продление парковочной сессии (этап 2).

Функция должна позволять продлевать парковочную сессию и осуществлять доплату за ее продление.

Для продления парковочной сессии функция должна передавать параметр duration – длина парковочной сессии в минутах.

Функция 8: Завершение парковочной сессии (этап 2).

Функция должна позволять завершать парковочную сессию досрочно по желанию пользователя с перерасчетом стоимости.

Функция 9: Управление несколькими сессиями одновременно (этап 2).

Функция должна позволять осуществлять начало, продление и завершение нескольких парковочных сессий одновременно для автомобилей с разными ГРЗ.

Функция 10: Получение списка автомобилей пользователя (этап 2).

Функция должна позволять добавлять автомобили с разными ГРЗ в систему и получать список уже привязанных автомобилей пользователя.

Информация о парковках предоставляется Заказчиком через API.

Для работы функции должна быть реализована следующая таблица в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| number | varchar(255) | Номер автомобиля |
| model | varchar(255) | Модель автомобиля |
| color | varchar(255) | Цвет автомобиля |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |

Функция 11: Корректировка парковочной сессии (этап 2).

Функция должна позволять скорректировать парковочную сессию пользователя в течение установленного системой времени.

Функция 12: Оплата ППЗТ (этап 2).

Функция должна позволять осуществлять оплату парковочных сессий на ППЗТ по билету.

Функция 13: История парковочных сессий пользователя (этап 2).

Функция должна позволять отображать историю ранее запущенных сессий пользователя.

Функция 14: Пополнение парковочного счет пользователя (этап 2).

Функция должна позволять пополнять парковочный счет пользователя. Функция должна передавать следующие параметры для пополнения парковочного счета:

* сумма пополнения;
* идентификатор платежа.

В параметрах ответа функция должна получать следующие параметры:

* остаток на счету;
* идентификатор платежа;
* идентификатор пользователя;
* сумма пополнения;
* время транзакции.

Функция 15: Получение состояния парковочного счета (этап 2).

Функция должна передавать в параметрах запроса идентификатор пользователя и получать в ответ информацию об остатке средств на парковочном счете.

* + 1. Требования к разработке подсистемы краудсорсинга

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Размещение предложений (этап 5).

Функция должна позволять размещать пользователям предложения маршрутов или добавления остановок, изменения их расположения и прочие предложения по улучшению транспортной системы.

Функция 2: Модерация пользовательских предложений (этап 5).

Функция должна позволять осуществлять модерацию пользовательских предложений, размещенных через приложение.

Функция 3: Просмотр пользовательских предложений (этап 5).

Функция должна позволять просматривать предложения других пользователей приложения, а также голосовать за них.

Функция 4: Размещение заданий для пользователей (этап 5).

Функция должна позволять размещать в системе задания, с возможностью их последующего выполнения пользователем.

* + 1. Требования к развитию подсистемы обратной связи

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Чат со специалистом транспортной службы (этап 5).

Функция должна позволять общаться со специалистом при помощи живого чата.

Функция должна позволять создавать сессию чата со специалистом технической поддержки.

Функция должна позволять отправлять вопросы специалисту транспортной службы и получать письменные ответы на вопросы.

Функция должна позволять отображать ленту сообщений в хронологическом порядке в чате без перезагрузки экрана приложения.

Функция должна позволять прикреплять изображения в сообщении в чате.

Функция должна позволять сохранять историю переписки со специалистом в чате.

Функция 2: Чат-бот для оперативного решения проблем пользователя (этап 4).

Функция должна предоставлять функционал чат-бота, который упростит поиск ответов на часто задаваемые вопросы по работе транспортного комплекса, с возможность расширения списка вопросов и ответов.

Функция должна позволять отправлять вопросы пользователя и получать список предустановленных вариантов ответа.

Функция должна позволять добавлять варианты ответов в систему.

Функция 3: Ответы на часто задаваемые вопросы (этап 3).

Функция должна позволять просматривать и редактировать раздел часто задаваемых вопросов. Пользователям должен быть доступен только просмотр раздела.

Функция 4: Кнопка SOS (этап 5).

Функция должна позволять передавать оперативную информацию о ситуации в общественном транспорте в соответствующую службу.

* + - 1. Форма обратной связи/обращения

Функция 1: Подача обращений (этап 4).

Функция должна позволять отправлять обращения по вопросам работы транспорта.

Функция должна позволять выбирать категорию и подкатегории для отправки обращения. Функция должна позволять формировать список необходимых для заполнения полей в зависимости от выбранной категории и подкатегорий.

Функция должна позволять формировать текст обращения и прикреплять изображения к обращению. Функция должна позволять получать ID созданного обращения.

Функция 2: Отображение списка обращений (этап 4).

Функция должна отображать список обращений и их статусы внутри приложения. Функция должна позволять обновлять статусы обращений не реже 1 раза в сутки.

Функция 3: Получение ответов на обращения (этап 4).

Функция должна позволять получать ответы от сервиса обработки обращений и отображать их внутри приложения в виде вложения.

* + 1. Требования к разработке подсистемы лояльности и геймификации

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Программа лояльности и геймификация (этап 4).

Функция должна позволять начислять баллы за различные действия в приложении для участия в программе лояльности.

Функция 2: Начисление бонусных баллов для участников программы лояльности приложения (этап 4).

Функция должна позволять начислять бонусные баллы пользователю за определенные действия в приложении.

Список действий должен быть уточнен на этапе технического проектирования в ЧТЗ.

Функция 3: Управление бонусным балансом (этап 4).

Функция должна отображать количество накопленных пользователем баллов.

Функция должна позволять списывать бонусные баллы на предусмотренные программой лояльности вознаграждения.

Функция 4: Партнёрские программы лояльности (этап 4).

Функция должна позволять конвертировать бонусный баланс пользователя в одной программе лояльности в бонусный баланс приложения и в обратную сторону.

Функция 5: Хранение информации о бонусных баллах (этап 4).

Функция должна позволять хранить информацию о бонусных баллах пользователя, в том числе:

* идентификатор зачисления;
* идентификатор пользователя;
* действия пользователя, за которые были начислены бонусы;
* количество зачисленных бонусных баллов;
* количество потраченных бонусных баллов;
* количество конвертируемых баллов;
* детализация по зачисленным бонусным баллам;
* детализация по потраченным бонусным баллам;
* детализация по конвертируемым бонусным баллам;
* дата зачисления;
* дата списания;
* дата сгорания;

Функция 6: Настройка метрик (этап 4).

Функция должна позволять отслеживать и оценивать эффективность внедрения программы лояльности при помощи заданных метрик.

Список метрик должен быть уточнен на этапе технического проектирования в ЧТЗ.

* + 1. Требования к развитию подсистемы персонализированных рекомендаций

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Формирование персональных предложений на основе истории маршрутов/избранного (этап 5).

Функция должна формировать рекомендации по точкам интереса, по маршруту и контенту (городские мероприятия, события), с учетом истории маршрутов и избранного.

Функция 2: Оповещение в случае изменения в работе транспорта (этап 2).

Функция должна отправлять PUSH-уведомления об изменениях в маршрутах, добавленных пользователями в избранное.

Функция должна отправлять PUSH-уведомления об изменениях в работе транспорта, добавленного пользователями в избранное.

Функция должна позволять отключать и включать PUSH-уведомления.

Данные о изменениях в работе транспорта предоставляются Заказчиком через API.

* + 1. Требования к развитию подсистемы профиля пользователя

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Добавление в избранное маршрутов, остановок электричек, велостанций и др. (этап 1).

Функция должна предоставлять возможность добавить в избранное маршрутов, станций ж/д, аэроэкспрессов, велостанций.

Функция должна позволять добавлять в избранное маршруты наземного транспорта из карточки остановки, карточки маршрута и карточки ТС.

Функция 2: Просмотр списка избранных объектов (этап 1).

Функция должна позволять просматривать список объектов, отмеченных как избранное, а также отменять данную отметку.

Функция должна позволять отображать список избранных объектов с сортировкой по категориям (остановки, маршруты, адреса).

Функция должна позволять фильтровать метки движущегося наземного транспорта на карте, добавленные в избранное.

Функция 3: Отображение информации о пользователе (этап 4).

Функция должна отображать личную информацию, статистику, историю пройденных маршрутов, активных сессиях проката/зарядки/поездках, информацию о геймификации, информацию о подключенных сервисах, интегрированных в Систему.

* + 1. Требования к развитию подсистемы авторизации

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Авторизация по номеру телефона (этап 1).

Функция должна позволять выполнять авторизацию в приложении по номеру телефона.

Функция должна осуществлять проверку введенного номера телефона на уникальность.

Функция должна позволять регистрировать нового пользователя с указанным номером телефона, если такой еще не был сохранен в системе.

Функция должна позволять верифицировать номер телефона пользователя с помощью кода, отправленного в СМС сообщении на указанный номер телефона. Функция должна позволять ограничивать количество запрошенных СМС сообщений в установленный период времени.

Функция 2: Авторизация с помощью VK Connect (этап 4).

Функция должна позволять выбирать дополнительный способ авторизации в приложении по VK Connect.

Функция 3: Объединение аккаунтов (этап 1).

Функция должна позволять пользователю объединять созданные им аккаунты, с переносом и объединением сохраненных данных.

* + 1. Требования к развитию подсистемы билетного сервиса

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Добавление виртуальной карты для оплаты проезда на общественном транспорте (этап 4).

Функция должна позволять добавлять виртуальную карту оплаты проезда в нативный кошелек мобильного устройства, а также позволять осуществлять проход через турникеты общественного транспорта с применением виртуальной карты.

Функция 2: Оферта платежной системы (этап 3).

Функция должна отображать оферту платежной системы.

Функция должна позволять принять или отклонить условия оферты. Если пользователь отклонил условия оферты, то функционал, связанный с оплатой через платежную систему, должен быть ему недоступен, а при попытке воспользоваться оплатой через платежную систему, ему должно предлагаться ознакомиться с офертой и принять ее условия.

Функция 3: Покупка и оплата электронных и виртуальных карт оплаты проезда (этап 3).

Функция должна позволять пополнять кошельки или покупать абонемент для электронных и виртуальных карт оплаты проезда при помощи API.

Функция должна автоматически определять номер электронной карты оплаты проезда при считывании карты через NFC для смартфонов на базе OS Android.

Функция должна позволять выбрать один из доступных абонементов для покупки или пополнение кошелька.

Функция должна позволять указать сумму пополнения кошелька электронной или виртуальной карты оплаты проезда.

Функция должна отображать статус оплаты покупки. Если оплата произошла с ошибкой, то пользователю должен отобразится текст ошибки, а процедура оплаты вернуться на предыдущий шаг, до ошибки.

Функция 4: Запись билета на карту (этап 3).

Функция должна позволять записывать приобретенный абонемент или сумму пополнения кошелька на электронную карту оплаты проезда при помощи подключения через NFC.

Функция должна отображать статус записи данных на карту. Если запись данных на карту произошла с ошибкой, то пользователю должен отобразиться текст ошибки, а процедура записи начаться с начала.

Функция должна работать на устройствах под управлением OS Android.

Функция 5: История платежей (этап 3).

Функция должна отображать историю платежей, совершенных через билетный сервис.

Функция 6: Авторизация в личном кабинете Метро (этап 3).

Функция должна позволять регистрироваться и авторизовываться в сервисе Личный кабинет Метро.

Функция 7: Список привязанных транспортных карт (этап 3).

Функция должна отображать все привязанные транспортные карты.

Функция должна позволять привязывать новые транспортные карты.

Функция должна позволять удалять привязанные транспортные карты.

Функция 8: Перенос баланса с карты на карту (этап 3).

Функция должна позволять выбирать транспортную карту и переносить ее баланс на другую, выбранную транспортную карту.

Функция 9: История проходов (этап 3).

Функция должна отображать историю проходов через турникеты при помощи прикрепленных транспортных карт.

Функция 10: Верификация транспортной карты (этап 3).

Функция должна позволять верифицировать транспортную карту при помощи указания двух последних остановок, на которых она была использована для прохода (оплаты проезда).

Функция должна позволять проводить верификацию при помощи отложенного платежа на определенную сумму.

Функция 11: Запрос сопровождения (этап 3).

Функция должна позволять отправлять запрос на сопровождение для маломобильных граждан.

* + 1. Требования к разработке подсистемы велопроката

Для работы подсистемы должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| user\_id | uuid | ID пользователя |
| login | integer | Логин пользователя в системе Велобайк |
| pin | integer | PIN код пользователя в системе Велобайк |
| token | varchar(255) | Токен доступа во внешнее API |
| expires\_on | timestamp | Время жизни токена |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| user\_id | uuid | ID пользователя |
| bike\_type | enum(‘electric’, ‘common’) | Тип велосипеда |
| duration | integer | Длительность поездки в секундах |
| price | integer | Стоимость поездки |
| departure\_date | timestamp | Дата старта аренды велосипеда |
| departure\_address | varchar(255) | Адрес станции старта аренды велосипеда |
| return\_date | timestamp | Дата окончания аренды велосипеда |
| return\_address | varchar(255) | Адрес станции окончания аренды велосипеда |

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Авторизация пользователей в сервисе городского велопроката (этап 2).

Функция должна позволять пользователям авторизовываться в сервисе городского велопроката, не выходя из приложения.

Функция 2: Регистрация пользователей в сервисе городского велопроката (этап 2).

Функция позволять пользователям регистрироваться в сервисах городского велопроката, не выходя из приложения.

Функция 3: Отображение информации о пользователе (этап 2).

Функция должна отображать данные о пользователе. Состав данных может меняться в зависимости от сервиса, который предоставляет данные.

Функция 4: Отображение активной сессии аренды (этап 2).

Функция должна отображать информацию об активной сессии аренды велосипеда пользователем.

Функция 5: Выбор абонемента (этап 2).

Функция должна позволять выбирать абонемент сервиса городского проката с дальнейшей его оплатой.

Функция 6: Отображение истории совершенных поездок на велосипеде (этап 2).

Функция должна отображать историю совершенных поездок на велосипеде и платежей.

Функция 7: Отображение истории совершенных платежных операций (этап 2).

Функция должна позволять отображать историю платежей пользователя с указанием типа платежа и даты совершения платежа.

Функция 8: Функция активации промокода (этап 2).

Функция должна позволять вводить в приложении промокод, активация которого открывает дополнительные абонементы для приобретения. Функция должна проверять промокод на валидность и возвращать ошибку, если такого промокода не существует.

Функция 9: Погашение задолженности (этап 2)

Функция должна позволять отображать информацию о существовании задолженности на балансе пользователя и погашать эту задолженность.

Функция 10: Получение PIN в СМС сообщении (этап 2).

Функция должна позволять запрашивать отправку параметра PIN в СМС сообщении на телефон пользователя.

* + 1. Требования к разработке подсистемы самокатов

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Отображение самокатов на карте (этап 1).

Функция должна позволять отображать самокаты, доступные для аренды, на карте.

Функция должна позволять выбирать самокаты на карте.

Функция должна отображать карточку выбранного самоката, которая предоставляет информацию о транспортном средстве.

Функция должна отображать зону действия на карте для выбранного самоката.

Функция должна отображать доступные зоны завершения поездки для выбранного самоката.

Для работы функции должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| code | varchar(255) | Идентификатор во внешней системе |
| company\_id | uuid | Идентификатор оператора |
| power | integer | Уровень заряда |
| deep\_link | varchar(255) | Диплинк для перехода в приложение оператора |
| geopoint | geometry | Координаты самоката |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| minute\_cost | integer | Цена в минуту |
| activation\_cost | integer | Стоимость активации |
| booked | boolean | Забронирован ли самокат |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| name | varchar(255) | Название |
| logo | uuid | Логотип |
| created\_at | timestamp | Дата создания |
| updated\_at | timestamp | Дата обновления |
| booking | boolean | Поддерживает бронирование |
| booking\_cancel | boolean | Поддерживает отмену бронирования |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| user\_id | uuid | Идентификатор пользователя |
| scooter\_id | uuid | Идентификатор самоката |
| created\_at | timestamp | Время создания |
| expires\_at | timestamp | Время окончания |
| active | boolean | Активность |

Функция 2: Бронирование поездки (этап 1).

При первом бронировании функция должна отображать информацию о правилах бронирования, а также оферту о предоставлении услуг.

Функция должна проверять расстояние между пользователем и выбранным самокатом. Если расстояние между пользователем и выбранным самокатом больше определенного расстояния в км, то бронирование должно быть недоступно.

Функция должна проверять количество доступных бронирований в сутки у пользователя. Если пользователь исчерпал свои бронирования, то бронирование должно быть недоступно.

Функция должна проверять наличие активного бронирования. Если имеется активное бронирование, то бронирование должно быть недоступно.

Если активного бронирования нет, то функция должна позволять бронировать поездку на выбранном самокате.

После бронирования самоката функция должна запускать таймер до автоматической отмены бронирования.

Функция 3: Отмена бронирования поездки (этап 1).

Функция должна проверять наличие активного бронирования. Если активное бронирование имеется, то функция должна позволять отменять активную бронь самоката. Если активного бронирования нет, то функция должна быть недоступна.

При отмене бронирования функция должна отображать оставшееся количество бесплатных бронирований.

Функция 4: Аренда самоката (этап 3).

Функция должна проверять состояние баланса средств в сервисе аренды самоката. При нулевом балансе средств в сервисе аренды самоката функция должна открывать окно пополнения баланса. При ненулевом балансе функция должна позволять арендовать выбранный самокат.

Перед запуском поездки на самокате функция должна запрашивать код самоката.

После указания кода самоката функция должна предоставлять информацию о поездке.

Функция должна позволять активировать промокод для получения дополнительных привилегий, предусмотренных сервисом.

Функция должна запускать режим осмотра самоката до начала поездки. При запуске режима осмотра должен запускаться таймер, по истечению которого поездка запустится автоматически. Режим осмотра должен позволять показать код замка, сообщить о неисправности самоката, отменить поездку, начать поездку.

При наличии активной поездки функция должна отображать информацию о текущей активной аренде.

Функция 5: Приостановка аренды самоката (этап 3).

Функция должна проверять наличие активной поездки на самокате. Если имеется активная поездка, то функция должна позволять переводить самокат в режим ожидания. В режиме ожидания самокат должен быть недоступен для аренды другим пользователям.

Функция 6: Возобновление аренды самоката (этап 3).

Функция должна проверять наличие у пользователя самоката в режиме ожидания. Функция должна позволять возобновлять поездку на самокате.

Функция должна позволять отображать код от замка данного самоката пользователю, у которого он находится в режиме ожидания.

Функция 7: Завершение аренды самокатов (этап 3).

Функция должна проверять наличие активных поездок у пользователя. При наличии активной поездки функция должна позволять завершать поездку на самокате.

Функция должна проверять, находится ли самокат в зоне завершения поездки.

Функция должна отображать ближайшую к пользователю зону завершения поездки.

Функция должна позволять фотографировать самокат при завершении поездки в зоне завершения поездки. Функция должна проверять фотографию на соответствие требованиям сервиса.

* + 1. Требования к развитию подсистемы обработки данных о пользователях

Подсистема должна поддерживать перечисленные ниже функции.

Функция 1: Сбор данных (этап 5).

Функция должна собирать следующие данные о поездках пользователя:

1) Сбор данных о маршруте:

* маршрут;
* время, затраченное на прохождение всего маршрута;
* время, затраченное на прохождение участка мультимодального маршрута;
* время, в которое был пройден маршрут;
* время, в которое был пройден мультимодальный участок;
* стоимость маршрута;
* стоимость мультимодального участка маршрута;
* степень загруженности транспортных систем на момент прохождения маршрута и мультимодальных участков.

2) Сбор данных о маршрутах:

* маршруты пользователя;
* количество прохождений маршрутов;
* используемые виды транспорта;

3) Сбор данных о просматриваемых объектах, таких как:

* остановках общественного транспорта;
* машинах каршеринга;
* станциях велопроката;
* самокатах;
* парковках;
* достопримечательностях.
* прочих объектах, представленных в системе.
  + 1. Требования к разработке подсистемы электрозарядок

Для работы подсистемы должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| user\_id | uuid | Идентификатор пользователя |
| username | varchar(255) | Имя пользователя |
| password | varchar(255) | Хеш пароля пользователя |
| token | text | Токен доступа во внешнее API |
| expires\_on | timestamp | Время жизни токена |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время обновления |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| lat | double | Широта |
| lon | double | Долгота |
| name\_ru | varchar(255) | Название |
| name\_en | varchar(255) | Название на английском |
| charge\_rate | double | Скорость зарядки |
| enabled | boolean | Включена для отображения |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| station\_id | integer | ID станции |
| status | enum(‘available’, ‘preparing’, ‘charging’, ‘suspendedEV’, ‘suspendedEVSE’, ‘finishing’, ‘reserved’, ‘faulted’, ‘unavailable’) | Статус коннектора |
| amperage\_type | enum(‘AC’,’DC’) | Тип коннектора |
| charge\_rate | double | Скорость заряда |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | integer | Идентификатор |
| user\_id | uuid | ID пользователя |
| status | enum(‘accepted’, ‘rejected’, ‘pending’, ‘ended’, ‘reserved’) | Статус сессии |
| started\_at | timestamp | Время начала |
| stopped\_at | timestamp | Время окончания |
| duration | integer | Длительность сессии |
| connector\_id | integer | Идентификатор коннектора |
| energy | double | Объем энергии |
| cost | double | Цена зарядки |
| planned\_duration | integer | Предполагаемое время зарядки |
| created\_at | timestamp | Время создания |
| updated\_at | timestamp | Время обновления |

Подсистема должна поддерживать перечисленные ниже функции:

Функция 1: Отображение электрозарядок на карте (этап 2).

Функция должна отображать расположение и статус электрозарядок на карте.

Статусы электрозарядок должны обновляться не реже, чем раз в минуту, а также при каждом сдвиге карты.

Функция 2: Отображение карточки станции электрозарядки (этап 2).

Функция должна отображать карточки станций электрозарядок.

В карточке электрозарядки должен отображаться статус для каждого коннектора. Статусы коннекторов должны обновляться не реже, чем раз в минуту, и при каждом открытии карточки.

В карточке электрозарядки должны отображаться типы коннекторов, используемые на данной станции электрозарядок.

Функция 3: Старт сессии зарядки (этап 2).

Функция должна позволять запускать сессию зарядки для указанного или забронированного коннектора на станции.

Функция 4: Бронирование зарядки (этап 2).

Функция должна позволять выбирать свободный коннектор на станции зарядки и бронировать его.

Функция должна запускать таймер, отсчитывающий 15 минут с начала бронирования.

Функция должна отменять бронирование, если сессия зарядки не была запущена по истечению таймера.

Функция 5: Завершение зарядки (этап 2).

Функция должна завершать текущую сессию зарядки. Если текущей сессии зарядки нет, но имеется забронированный коннектор, то должна происходить отмена бронирования.

Функция 6: Оплата зарядки (этап 5).

Функция должна позволять оплачивать сессию электрозарядки.

* + 1. Требования к разработке подсистемы биометрического распознавания для оплаты проезда

Функция 1: Фотография для биометрического распознавания (этап 2).

Функция должна отображать прикрепленную фотографию, используемую для распознавания.

Функция должна позволять прикреплять фотографию для распознавания.

Функция 2: Банковская карта для оплаты проезда по биометрическому распознаванию   
(этап 2).

Функция должна отображать данные о прикрепленной к сервису биометрического распознавания карте, с которой будут списываться средства при проезде.

Функция должна позволять прикреплять банковскую карту к сервису биометрического распознавания.

Функция должна позволять изменять банковскую карту, прикр+епленную к сервису биометрического распознавания.

Функция 3: Отображение информации профиля биометрического сервиса (этап 2).

Функция должна отображать следующие данные о пользователе в профиле сервиса:

* фотография;
* статус;
* имя;
* фамилия;
* информация о банковской карте.

Функция 4: Включение и отключение функции оплаты при помощи биометрического распознавания (этап 2).

Функция должна позволять включать и отключать оплату проезда при помощи биометрического распознавания.

* + 1. Требования к разработке подсистемы микротранзита

Для работы подсистемы должны быть реализованы следующие таблицы в БД:

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | uuid | Идентификатор |
| trip\_id | uuid | Идентификатор поездки |
| user\_id | uuid | Идентификатор пользователя |
| passengers\_count | integer | Количество пассажиров |
| initial\_pickup\_eta | timestamp | Начальное время прибытия водителя на точку посадки |
| initial\_dropoff\_eta | timestamp | Начальное время прибытия водителя на точку высадки |
| pickup\_eta | timestamp | Обновленное время прибытия водителя на точку посадки |
| dropoff\_eta | timestamp | Обновленное время прибытия водителя на точку высадки |
| cost | integer | Стоимость поездки |
| cancellation\_cost | integer | Стоимость платной отмены |
| pickup\_point | geometry | Геометрия точки посадки |
| pickup\_point\_address | varchar(255) | Адрес места посадки |
| dropoff\_point | geometry | Геометрия точки высадки |
| starting\_point | geometry | Геометрия точки отправления |
| ending\_point | geometry | Геометрия точки завершения |
| boarding\_route\_time | integer | Длительность маршрута до точки посадки |
| boarding\_route\_geometry | geometry | Геометрия маршрута до точки посадки |
| dropoff\_route\_time | integer | Длительность маршрута от точки высадки до точки завершения |
| dropoff\_route\_geometry | geometry | Геометрия маршрута от точки высадки до точки завершения |
| rated | boolean | Оценил ли пользователь поездку |
| free\_cancellation\_seconds | integer | Количество секунд с момента бронирования, в течение которых отмена будет бесплатной |
| boarding\_waiting\_seconds | integer | Сколько секунд водитель будет ждать пользователя на точке посадки |
| soon\_arrival\_notified | boolean | Был ли пользователь уведомлен о скором прибытии водителя на место посадки |
| arrival\_notified | boolean | Был ли пользователь уведомлен о прибытии водителя на место посадки |
| no\_show\_notified | boolean | Был ли пользователь уведомлен о том, что водитель не дождался его на месте посадки |
| delay\_notified | boolean | Был ли пользователь уведомлен о том, что водитель задерживается |
| free\_cancellation\_notified | boolean | Был ли пользователь уведомлен о том, что отмена поедки бесплатна в связи с сильным опозданием водителя |
| vehicle\_changed\_notified | boolean | Был ли пользователь уведомлен об изменении ТС |
| boarding\_point\_changed\_notified | boolean | Был ли пользователь уведомлен об изменении точки посадки |
| free\_cancellation\_used | boolean | Использовал ли пользователь возможность бесплатной поздней отмены |
| free\_cancellations\_left | integer | Количество оставшихся бесплатных поздних отмен у пользователя |
| booked\_at | timestamp | Время бронирования |
| driver\_id | uuid | Идентификатор водителя |
| vehicle\_id | uuid | Идентификатор ТС |
| status | varchar(255) | Текущий статус поездки |
| arrived\_at | timestamp | Время прибытия на точку посадки |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| first\_name | varchar(255) | Имя водителя |
| last\_name | varchar(255) | Фамилия водителя |
| id | uuid | Идентификатор водителя |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| license\_plate | varchar(255) | Гос. номер |
| model | varchar(255) | Модель |
| current\_location | geometry | Текущее местоположение |

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| trip\_id | uuid | Идентификатор поездки |
| status | varchar(255) | Установленный статус |
| set\_in | timestamp | Время установки статуса |

Функция 1: Отображение зон действия сервиса микротранзита (этап 2).

Функция должна отображать зоны действия сервиса микротранзита на карте.

Зоны действия Подрядчик должен получать из внешнего сервиса, в виде полигонов зон.

К каждой зоне должен прикрепляться массив доступных остановок, позволяющих остановиться автобусу микротранзита.

Функция 2: Построение маршрута на микротранзите (этап 2).

Функция должна позволять строить маршруты на микротанзите.

Функция должна позволять выбирать начальную и конечную точку маршрута на микротранзите из списка доступных остановок, или на карте приложения внутри одной из зон действия микротанзита.

Функция должна не позволять строить маршрут, если выбранная точка начала не относится к зоне действия микротранзита.

Функция должна позволять передвигать пиктограмму начала и конца маршрута по карте. При передвижении пиктограмм данные о геолокации точек должны обновляться.

После выбора точки начала маршрута функция должна отображать список конечных точек, доступных для данной начальной точки. Выбор конечной точки должен быть доступен только при помощи данного списка.

После выбора начальной точки, на карте должны отображаться только те остановки, которые доступны в качестве конечной точки маршрута для данной начальной точки.

После выбора обеих точек пользователю должен быть доступен выбор рейса микротранзита.

Функция должна отображать построенный маршрут на карте. Функция должна прокладывать пеший маршрут до начальной точки маршрута.

Функция 3: Поиск остановок микротранзита (этап 2).

Функция должна позволять осуществлять поиск в поисковой строке приложения.

При указании адреса остановки микротранзита функция должна отображать место нахождение остановки на карте, отображая соответствующий индикатор (пин) остановки.

Функция должна отображать список доступных остановок микротранзита.

Функция 4: Отображение доступных остановок (этап 2).

При построении маршрута функция должна отображать доступные для выбора остановки микротранзита.

Функция должна отображать остановки, возможные для выбора в качестве начальной точки маршрута.

Функция должна определять зону и подзону сервиса микротранзита, к которой относится выбранная точка начала маршрута.

Функция должна отображать список остановок, доступных для данной подзоны или зоны в качестве конечной точки маршрута.

Функция 5: Выбор рейса (этап 2).

Функция должна формировать список доступных рейсов для указанных начальной и конечной точек. Для получения списка рейсов пользователь должен быть авторизован в Системе и иметь прикрепленный номер телефона.

Если функция не находит рейсов, подходящих для указанных точек, то она должна отображать сообщение об отсутствии рейсов с возможностью повторного поиска рейсов или возвратом к выбору параметров для построения маршрута.

Список рейсов должен отображать карточки подходящих рейсов. В карточках рейсов должны отображаться следующие данные:

* прогнозируемое время прибытия на начальную точку;
* прогнозируемое время прибытия на конечную точку;
* стоимость поездки.

Функция 6: Оплата поездки на микротранзите (этап 2).

Функция должна позволять оплачивать поездку на микротранзите.

Функция должна позволять выбирать один из сохраненных в подсистеме оплаты банковской картой способов оплаты.

Если сохраненные способы оплаты отсутствуют, то функция должна предлагать их добавить.

После выбора способа оплаты, функция должна запрашивать у подсистемы оплаты банковской карты холдирования цены поездки.

Если функция получает отрицательный ответ на запрос холдирования, то процедура оплаты должна отменяться с предоставлением соответствующей ошибки пользователю.

После успешного холдирования должна запускаться процедура бронирования рейса. Если бронирование было неуспешным, то функция должна запрашивать расхолдирование средств у подсистемы оплаты банковской картой.

Функция 7: Бронирование рейса (этап 2).

Функция должна осуществлять бронирование места в выбранном рейсе.

Функция должна проверять доступность выбранного рейса и наличие на нем свободных мест. Если рейс доступен и имеются свободные места, то функция должна забронировать необходимое количество мест, а если рейс недоступен или на нем нет необходимого количества свободных мест, то функция должна отображать оповещение об ошибке бронирования и возвращать пользователя к списку рейсов.

Функция 8: Отображение данных о поездке на транзите (этап 2).

Функция должна отображать следующие данные о текущей поездке на микротранзите до прибытия автобуса на точку посадки:

* точка посадки, адрес посадки и время прибытия ТС;
* точка отправления и время в пути до точки посадки;
* текущая геопозиция пользователя;
* местонахождение ТС;
* время до прибытия ТС на точку посадки;
* модель и номер транспортного средства;
* примерное время в пути в диапазоне.

Функция должна отображать кнопку отмены поездки.

Функция должна позволять запросить обратную связь со службой технической поддержки микротранзита.

После прибытия автобуса на точку посадки должна отображаться следующая информация:

* местоположение автобуса на точке посадки с информацией;
* пеший участок пути и информация о времени в пути от точки отправления до точки посадки в автобус;
* номер автобуса;
* время ожидания автобуса на точке посадки;

Функция должна отправлять сообщение пользователю о прибытии автобуса на точку посадки.

После прибытия автобуса на точку посадки функция должна позволять пользователю подтвердить статус посадки.

После подтверждения посадки функция должна отображать следующие данные о поездке:

* текущее местоположение автомобиля;
* точка прибытия на карте;
* адрес прибытия;
* модель и номер автомобиля;
* время в пути.

Функция 9: Подтверждение статуса посадки (этап 2).

Функция должна предоставлять возможность отсканировать специальный QR-код, размещаемый на входе в автобус микротранзита, для подтверждения посадки в автобус.

Функция должна запрашивать разрешение на доступ к камере устройства. В случае отказа функция должна возвращать пользователя на предыдущий экран.

Функция должна проводить валидацию QR-кода автобуса на соответствие данным из заказа микротранзита. При ошибке валидации пользователю должно выводиться соответствующее сообщение. При успешной валидации статус посадки должен быть подтвержден.

Функция 10: Отмена поездки (этап 2).

Функция должна позволять выполнять отмену поездки двух типов:

1. отмена поездки системой;
2. отмена поездки пользователем.

При отмене поездки Системой функция должна отправлять запрос на расхолдирование стоимости поездки в подсистему оплаты банковской картой, за исключением автоматической отмены поездки из-за опоздания пользователя на посадку.

После автоматической отмены поездки Системой функция должна отображать пользователю сообщение об отмене и возвращать его к списку рейсов.

При отмене поездки пользователем функция должна отображать пользователю окно с информацией о правилах отмены.

Функция должна запрашивать подтверждение отмены.

При подтверждении пользователем отмены, функция должна отображать пользователю окно, с возможностью указать причину отмены.

Функция 11: Оценка поездки по завершению (этап 2).

По завершению поездки функция должна отображать окно с возможность оценки поездки по пятибалльной шкале.

Функция должна позволять оставить отзыв о поездке.

Функция 12: Сохранение истории поездок (этап 2).

Функция должна сохранять остановки, выбранные в качестве начала маршрута.

Функция должна сохранять остановки выбранные в качестве конца маршрута.

Функция должна сохранять маршрут, состоящий из выбранных начальных и конечных остановок.

Функция должна сохранять историю завершенных поездок.

Функция 13: Отображение последних выбранных остановок и маршрутов (этап 2).

Функция должна предлагать пользователю остановки в окне выбора точек маршрута для построения маршрута на микротранзите.

Функция должна предлагать начальную точку из тех, которые пользователь использовал в предыдущих маршрутах или выбирал в качестве начала маршрута.

Функция должна предлагать конечную точку из тех, которые пользователь использовал в предыдущих маршрутах или выбирал в качестве конца маршрута.

Первыми должны предлагаться те точки, которые использовались в построенных маршрутах, а после них те, которые выбирались пользователем как одна из точек маршрута (точки без пары).

После выбора начальной точки функция должна первыми предлагать те конечные точки, которые уже выбирались вместе с данной начальной точкой.

Функция 14: Отображение истории завершенных поездок(этап 2).

Функция должна отображать историю поездок. История поездки должна содержать следующие данные:

* адрес точки отправления;
* адрес точки прибытия;
* время отправления;
* время прибытия;
* дата;
* стоимость;
* тип оплаты.
  + 1. Требования к развитию подсистемы интеграций

Подсистема интеграций должна осуществлять обмен данными со смежными системами при помощи API. На основе внутренних запросов от других подсистем, подсистема интеграции должна формировать подходящие запросы в смежные системы, отправлять их, получать и обрабатывать ответы от смежных систем, и переводить их во внутренний формат данных, передавая данные в подсистему, от которой поступил внутренний запрос.

API для реализации интеграций предоставляются Заказчиком.

Требования к функционалу подсистемы зависят от интегрируемых систем и представлены ниже.

* + - 1. Требования к функциям интеграции с сервисами такси

Функция 1: Запрос поддерживаемых тарифов (этап 2).

Функция должна запрашивать у смежной системы список поддерживаемых тарифов у оператора такси.

Запрос должен содержать данные о координатах места посадки и высадки пассажира.

Функция должна получать и обрабатывать ответ от смежной системы.

Ответ должен содержать следующие данные:

* список тарифов;
* список опций для тарифов.

Список тарифов и опций Подрядчик должен получать для каждого сервиса такси отдельно.

Список отправляемых и получаемых данных может меняться в зависимости от сервиса такси, с которым взаимодействует Система.

Функция 2: Запрос цены (этап 2).

Функция должна формировать и отправлять запрос на расчет цены поездки с указанными параметрами.

Для расчета, в смежную систему должны передаваться следующие данные:

* номер мобильного телефона пользователя;
* координаты начала маршрута;
* координаты конца маршрута;
* выбранный тариф;
* выбранные опции;
* id номер расчета.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующую информацию:

* расстояние поездки;
* время поездки;
* наличие скидки;
* цена до скидки;
* цена после скидки;
* стоимость подачи машины.

Список отправляемых и получаемых данных может меняться в зависимости от сервиса такси, с которым взаимодействует Система.

Функция 3: Запрос на создание заказа (этап 2).

Функция должна формировать и отправлять запрос на создание заказа в смежную систему, с указанием пользовательских параметров заказа.

В запросе должна предоставляться следующая информация:

* номер мобильного телефона пользователя;
* тариф;
* тип оплаты;
* координаты посадки;
* координаты высадки.

Функция должна получать и обрабатывать ответ от смежной системы, в котором содержится следующая информация:

* hashid заказа;
* статус заказа.

Список отправляемых и получаемых данных может меняться в зависимости от сервиса такси, с которым взаимодействует Система.

Функция 4: Получение информации о заказе по hashid (этап 2).

Функция должна формировать и отправлять запрос на получение информации о заказе в смежную Систему. Запрос должен содержать hashid заказа.

Функция должна получать и обрабатывать ответ от смежной системы. Ответ должен содержать следующие данные:

* hashid заказа;
* статус заказа.

Список отправляемых и получаемых данных может меняться в зависимости от сервиса такси, с которым взаимодействует Система.

Функция 5: Запрос на отмену заказа (этап 2).

Функция должна формировать и отправлять запрос на отмену текущего заказа такси. Запрос должен содержать следующие данные:

* hashid заказа;
* причина отмены.

Функция должна получать и обрабатывать ответ от смежной системы, подтверждающий отмену заказа.

Функция 6: Получение маршрута для текущего заказа (этап 2).

Функция должна формировать и отправлять запрос на получение данных для отображения маршрута поездки на карте приложения. В запросе должен содержаться hashid маршрута.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит геометрию маршрута для построения на карте, а также текущие координаты такси.

Функция 7: Обработка ошибок (этап 2).

Функция должна получать коды ошибок, полученные при запросах к смежной системе, и обрабатывать их. Должны обрабатываться следующие коды ошибок:

|  |  |
| --- | --- |
| **Код ошибки\ответа** | **Значение** |
| 200 | Запрос успешно выполнен |
| 400 | Некорректный запрос |
| 401 | Требуется авторизация |
| 403 | Пользователь не найден или нет прав |

Список ошибок и их значений может меняться в зависимости от смежной системы, с которой выполняется интеграция.

* + - 1. Требования к функциям интеграции с каршерингом

Функция 1: Интеграция с операторами каршеринга (этап 3).

Функция должна обеспечивать интеграцию с различными операторами каршеринга. Полный перечень элементов интеграции прорабатывается на этапе ЧТЗ.

* + - 1. Требования к функциям интеграции с сервисом электричек и сервисом ж\д сообщения с аэропортами

Функция 1: Интеграция с сервисом электричек (этап 4).

Функция должна реализовывать интеграцию с сервисом электричек и позволять получать расписание электричек в реальном времени, включая изменения, а также покупать б илеты, доступные к покупке через интернет.

Функция 2: Интеграция с сервисом ж\д сообщения с аэропортами (этап 4).

Функция должна реализовывать интеграцию с сервисом ж\д сообщения с аэропортами г. Москва и позволять получать расписание аэроэкспресса в реальном времени, включая изменения, а также покупать билеты.

* + - 1. Требования к функциям интеграции с сервисом велопроката

Функция 1: Авторизация в сервисе городского велопроката (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на авторизацию в сервисе городского велопроката. Запрос должен содержать номер телефона пользователя.

Функция должна получать и обрабатывать ответ от смежной системы с результатом авторизации. Ответ должен содержать карточку пользователя, которая содержит следующие данные:

* ID пользователя;
* статус пользователя;
* разрешена ли аренда;
* наличие долга;
* размер долга;
* статус аренды;
* наличие суммы к удержанию;
* сумма к удержанию;
* сумма бонусов;
* PIN код;
* имя пользователя;
* фамилия пользователя;
* номер телефона;
* email.

Список отправляемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 2: Регистрация пользователей в сервисе городского велопроката (этап 2).

Функция должна формировать и отправлять запрос на регистрацию в сервисе, включающий в себя:

* номер телефона пользователя,
* email;
* имя;
* фамилию.

Функция должна получать и обрабатывать ответ от смежной системы с результатом регистрации.

Список отправляемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 3: Запрос данных о пользователе (этап 2).

Функция должна формировать запрос в смежную систему на получение данных о конкретном пользователе. Запрос должен включать номер телефона пользователя.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит данные о пользователе.

Функция должна получать следующие данные:

* id пользователя;
* имя;
* фамилия;
* номер телефона;
* email;
* логин;
* дата создания аккаунта;
* день рождения;
* данные о прикрепленной карте;
* данные об арендах;
* данные об удержанных суммах;
* сумма бонусов;
* баланс;
* данные о задолженностях.

Список данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 4: Запрос данных о станциях велопроката (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение списка станций велопроката.

Функция должна получать и обрабатывать ответы от смежной системы, которые содержат списки станций со следующими данными:

* координаты станции;
* статус работы станции;
* наличие свободных велосипедов;
* наличие парковочных стоек.

Список данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 5: Получение профиля пользователя с информацией о текущей поездке (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение данных о пользователе и его текущей поездке.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* id пользователя;
* логин;
* дата создания аккаунта;
* день рождения;
* данные об арендах;
* данные об удержанных суммах;
* сумма бонусов;
* баланс;
* данные о задолженностях;
* id аренды;
* id велосипеда;
* тип велосипеда;
* дата отправления;
* id станции отправления;
* номер стойки на станции отправления;
* данные о последней аренде.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 6: История совершенных поездок (этап 2).

Функция должна формировать и отправлять запрос на получение истории совершенных поездок пользователя.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

– список аренды велосипедов, каждая позиция которого содержит:

* id аренды;
* id велосипеда;
* тип велосипеда;
* дата отправления;
* id станции отправления;
* номер стойки на станции отправления;
* адрес станции отправления;
* адрес станции отправления на английском языке;
* дата возврата;
* id станции возврата;
* номер стойки на станции возврата;
* адрес станции возврата;
* адрес станции возврата на английском языке;
* пройденное расстояние;
* длительность аренды;
* стоимость аренды;
* данные по оплате;
* статус платежа.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 7: Инициализация оплаты покупки (этап 2).

Функция должна формировать и отправлять запрос на оплату аренды или задолженности по аренде. Запрос должен содержать следующие данные:

* id тарифа для покупки, если совершается покупка тарифа (аренды);
* флаг оплаты задолженности, если оплата назначается для погашения долга;
* тип оплаты.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* тип платежа;
* ссылка на платеж;
* параметры платежного сервиса.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 8: Данные об аренде (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение данных о контракте аренды. Запрос должен содержать id контракта аренды.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* количество арендуемых велосипедов;
* стоимость аренды;
* стоимость страховки;
* id страховки.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 9: Активация промокода (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на активацию промокода. Запрос должен содержать промокод.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результат активации промокода.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 10: Активация бонусного кода (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на активацию бонусного кода. Запрос должен содержать бонусный код.

Функция должна получать и обрабатывать запрос от смежной системы, который должен содержать результат активации бонусного кода.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 11: Получение списка тарифов (этап 2).

Функция должна формировать и обрабатывать запрос в смежную систему на получение списка поддерживаемых тарифов.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* список тарифов, каждая позиция в котором содержит следующие данные:
* id тарифа;
* название тарифа;
* бесплатные минуты;
* тип аренды;
* тип длительности;
* длительность;
* стоимость;
* сумма залога;
* тип страховки;
* стоимость страховки.

Функция 12: Обработка ошибок (этап 2).

Функция должна собирать и обрабатывать ответы от смежных систем, которые содержат в себе следующие коды ошибок.

|  |  |
| --- | --- |
| **Код** | **Значение** |
| 0 | Запрос выполнен успешно |
| 100 | Не указан бонус-код |
| 101 | Не указан промокод\Указан неверный бонус-код |
| 102 | Некорректный номер промокода\Бонус-код не активный |
| 103 | Активация промокодов недоступна для клиентов, имеющих задолженности. Пополните ваш баланс\Бонус-код просрочен |
| 104 | Бонус-код неактивный |
| 200 | Промокод не активный |
| 201 | Промокод просрочен |
| 203 | Для промокода не найден тариф |
| 204 | У тарифа по промокоду закончился срок действия |
| 205 | Промокод не активный |
| 206 | Тип тарифа для активации промокода не верный |
| 207 | Тариф по промокоду уже доступен для клиента |
| 20106 | Клиент с телефоном «номер телефона» уже зарегистрирован в системе |
| 20107 | Ошибка привязки карты Тройка. Данная карта не поддерживается шлюзом Тройки. |
| 20108 | Ошибка привязки карты Тройка. Такая карта уже привязана в системе. |
| 20109 | Карта Тройка с указанным номером уже привязана к другому клиенту |

Список кодов ошибок и действий, связанных с этим кодом, должны быть составлены для каждого сервиса отдельно.

* + - 1. Требования к функциям интеграции с самокатами

Функция 1: Список транспортных средств (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на получение списка действующих транспортных средств в данной зоне. Запрос должен содержать id зоны действия самокатов и флаг фильтрации списка по наличию Bluetooth у устройства.

Функция должна получать и обрабатывать ответ смежной системы, который содержит список транспортных средств, действующих в данной зоне. Каждая позиция списка должна содержать в себе следующие параметры:

* id транспорта,
* состояние транспорта;
* координаты;
* тип транспорта.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 2: Данные о транспортном средстве (этап 1).

Функция должна формировать и отравлять запрос в смежную систему на получение данных о транспортном средстве. Запрос должен содержать следующие параметры:

* широта;
* долгота;
* id транспортного средства.

Функция должна принимать и обрабатывать ответы от смежной системы, которые должны включать следующие данные:

* id транспортного средства;
* координаты;
* id модели транспортного средства;
* тип транспортного средства;
* нужна ли полная верификация для доступа к этому транспортного средства;
* цена за период в минутах;
* цена в копейках;
* форматированная цена;
* id ценового правила;
* название ценового правила;
* период времени, за который снимается оплата (в минутах);
* есть ли минимальная цена поездки;
* минимальная цена;
* сколько денег проверять на балансе карты перед поездкой;
* использовать ли проверку карты;
* цена активации после осмотра;
* использовать ли оплату за активацию;
* в какой зоне парковки находится ТС;
* в какой зоне использования находится ТС;
* процент зарядки;
* оставшееся расстояние;
* сколько активна батарея;
* сколько часов отключена батарея;
* статус батареи;
* название модели;
* ссылка на иконку модели;
* ошибки;
* статус подключения к ТС.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 3: Зоны завершения поездки (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение списка зон завершения (парковки). Запрос должен содержать в себе id зоны покрытия.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит список зон завершения поездки. Для каждого элемента списка должны быть представлены следующие атрибуты:

* id правила;
* уникальные правила для этой модели;
* идентификатор модели;
* процент, который будет начислен;
* дата изменения;
* название правила;
* процент, который будет начислен;
* тип начисление или списание;
* id правила завершения поездки;
* дата изменения зоны завершения;
* название правила зоны завершения;
* бонусная или штрафная парковка;
* цена парковки в этой зоне;
* цена парковки в копейках;
* форматированная цена;
* какой процент от поездки вернется бонусом;
* название модели с исключением из правил начисления;
* сумма бонуса;
* валюта;
* сумма бонуса в копейках;
* форматированная сумма бонуса;
* процент бонуса от поездки;
* координаты центра зоны парковки;
* точки координат полигона парковки;
* должно ли быть уведомление при пересечении зоны;
* к какой зоне использования относится эта зона завершения.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 4: Зона использования (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение зоны использования для самоката. Запрос должен содержать в себе данные о долготе и широте самоката, для которого нужна зона использования.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* идентификатор зоны;
* название зоны;
* id зоны аренды;
* иконка зоны;
* какие модели можно оставлять;
* иконка пина для зоны;
* координаты центра;
* массив точек-координат полигона;
* тип зоны;
* должно ли быть уведомление при пересечении зоны.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 5: Получение модели самоката (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на получение данных о модели самоката. Запрос должен содержать в себе данные о долготе и широте самоката, для которого получается модель.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* id модели;
* название модели;
* тип транспорта;
* картинка модели;
* уровень заряда для снятия с доступа;
* уровень заряда для отключения;
* уровень заряда пустой батареи;
* средняя скорость;
* уровни заряда для получения бесплатного времени для завершения;
* количество сигналов для изменения уровня заряда;
* расстояние, которое можно проехать на полной батарее.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 6: Статус бронирования (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на получение статуса бронирования. Запрос должен включать в себя долготу и широту самоката, статус которого запрашивается.

Функция должна принимать и обрабатывать ответ от смежной системы, который включает в себя следующие данные:

* время начала заказа;
* дата перехода в статус «ожидание»;
* доступное число бронирований в день;
* сколько минут доступно клиенту для бронирования;
* id арендованного самоката;
* идентификатор модели самоката;
* наличие бонусных денег;
* валюта;
* сумма;
* сумма в копейках;
* сумма в валюте;
* количество оставшихся аренд;
* время брони, если заказ в статусе забронирован;
* сколько часов активна батарея;
* сколько часов батарея в пассивном режиме;
* процент заряда;
* сколько километров осталось;
* статус батареи;
* список зон завершения, в которых был закрыт замок/выключен самокат;
* список зон использования, в которых был закрыт замок/выключен самокат;
* название структуры с локацией самоката;
* код замка;
* тип замка;
* тип закрытия замка;
* сколько времени прошло с момента начала аренды;
* идентификатор аккаунта;
* номер телефона на аккаунте;
* идентификатор активированного самоката.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 7: Бронирование самоката (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на бронирование самоката. Запрос должен содержать в себе id самоката, долготу и широту.

Функция должна получать и обрабатывать ответ от смежной системы. В качестве ответа на запрос должен поступать статус забронированного самоката. Перечень получаемых данных приведен в функции 6.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 8: Отмена бронирования (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на отмену бронирования. Запрос должен включать в себя id самоката, долготу и широту.

Функция должна получать и обрабатывать ответ от смежной системы с результатом отмены бронирования.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 9: Аренда самоката (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на начало аренды самоката. Запрос должен содержать:

* id самоката;
* широту;
* долготу;
* наличие страховки.

Функция должна получать и обрабатывать ответ от смежной системы с результатом начала аренды самоката.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 10: Приостановка сессии аренды (этап 2).

Функция должна позволять формировать и отправлять запрос в смежную систему на приостановление сессии аренды. Запрос должен содержать в себе:

* id самоката;
* широту;
* долготу.

Функция должна получать и обрабатывать ответ от смежной системы с результатом приостановки сессии.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 11: Возобновление сессии аренды (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на возобновление приостановленной сессии аренды. Запрос должен содержать:

* id самоката;
* широту;
* долготу.

Функция должна принимать и обрабатывать ответ от смежной системы с результатом возобновления сессии аренды.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 12: Завершение сессии аренды (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на завершение аренды. Запрос должен содержать:

* id самоката;
* широту;
* долготу;
* фотографии припаркованного самоката.

Функция должна принимать и обрабатывать ответ от смежной системы с результатом завершения сессии аренды.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 13: Обработка ошибок (этап 2).

Функция должна собирать и обрабатывать ответы от смежных систем, которые содержат в себе следующие коды ошибок.

|  |  |
| --- | --- |
| **Код** | **Значение** |
| 200 | Запрос успешно выполнен |
| 400 | Неверный запрос |
| ActionNotAvailable | Операция отклонена. Повторите операцию позднее |
| BookingsLimit | Достигнуто максимальное число одновременных броней. Аренда самоката возможна только на месте |
| DistanceTooFar | Расстояние до самоката слишком большое |
| HasNoBookingsToday | Достигнуто максимальное число броней за сутки. Аренда самоката возможна только на месте |
| ScooterAlreadyBooked | Самокат уже забронирован |
| ScooterAlreadyOrdered | Самокат уже арендован |
| ScooterAvailableViaBluetooth | Этот самокат недоступен |
| ScooterBusy | Самокат недоступен |
| ScooterNotFound | Такой самокат не найден |
| ScooterOutOfZones | Самокат находится вне зоны парковки |
| ScooterNotAvailable | Самокат недоступен |
| ScooterIsNotOnline | Нет связи с самокатом |
| TransportInRestrictedZone | Самокат находится в запретной зоне |
| AccountNotFoundError | Пользователь не найден |
| BookingNotFound | Бронь не найдена. |

Список кодов ошибок и действий, связанных с этим кодом, должны быть составлены для каждого сервиса отдельно.

* + - 1. Требования к функциям интеграции с электрозарядками

Функция 1: Авторизация в сервисе электрозарядок (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на авторизацию по протоколу oauth2.

Функция должна получать и обрабатывать ответ о результатах авторизации.

Функция должна сохранять ключ (токен), полученный после авторизации.

Функция 2: Бронирование сервиса электрозарядки (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на бронирование коннектора, находящегося на станции электрозарядки.

Функция должна получать и обрабатывать ответ о результатах запроса бронирования.

Функция 3: Старт зарядки (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на старт сессии зарядки для указанного коннектора.

Функция должна получать и обрабатывать ответ о результатах запроса от смежной системы.

Функция 4: Остановка зарядки (этап 2).

Функция должна формировать и отправлять запрос в смежную систему о завершении сессии зарядки.

Функция должна получать и обрабатывать ответ о результатах запроса завершения сессии зарядки.

Функция 5: Отмена бронирования (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на отмену бронирования электрозарядки.

Функция должна получать и обрабатывать ответ на запрос отмены бронирования.

Функция 6: Получение статуса сессии (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение статуса сессии.

Функция должна получать и обрабатывать ответ на запрос статуса сессии.

Функция 7: Получение списка станций (этап 2).

Функция должна формировать и отправлять запрос на получение списка станций электрозарядок, включая тип электрозарядочной станции и типа коннекторов на станции.

Функция должна получать и обрабатывать ответ на запрос получения списка станций.

Функция 8: Запрос оплаты сессии зарядки (этап 5).

Функция должна формировать и отправлять запрос в смежную систему на оплату указанной сессии зарядки.

Функция должна получать и обрабатывать ответ на запрос оплаты сессии зарядки.

* + - 1. Требования к функциям интеграции с билетным сервисом

Функция 1: Инициализация покупки (этап 3).

Функция должна формировать и отправлять в смежную систему запрос на инициализацию покупки. Запрос должен включать в себя следующие данные:

* Идентификатор сессии;
* Идентификатор транспортной карты;
* Номер транспортной карты;
* Идентификатор билета;
* Количество транспортных единиц для кошелька, в рублях;
* Тип оплаты;
* Платежный токен;
* Идентификатор карты;
* Сумма покупки;
* URL, на который клиента будет переадресовывать эквайринг по завершении оплаты.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* Идентификатор сессии;
* URL для перехода на страницу авторизации стороннего сервиса;
* URL на который сторонний сервис перенаправит пользователя, после завершения авторизации.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 2: Инициализация покупки для виртуальной карты оплаты проезда (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на инициализацию покупки для виртуальной карты оплаты проезда. Запрос должен включать в себя:

* Условный идентификатор карты оплаты проезда;
* Идентификатор билета;
* Сумма покупки или количество транспортных единиц для кошелька, в рублях;
* Тип оплаты;
* Платежный токен;
* URL, на который клиента будет переадресовывать эквайринг по завершении оплаты;
* Wallet-данные устройства для хранения билета/токена Google/Samsung/Apple;
* Информация об устройстве.

Для виртуальной карты оплаты проезда должен быть доступен только один тип оплаты – банковской картой.

Функция должна получать и обрабатывать ответ от смежной системы, который должен включать следующие данные:

* Идентификатор сессии;
* URL для перехода на страницу авторизации стороннего сервиса;
* URL, на который сторонний сервис перенаправит пользователя, после завершения авторизации.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 3: Получение статуса оплаты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение статуса оплаты. Запрос должен включать в себя идентификатор сессии.

Функция должна получать и обрабатывать ответ от смежной системы, который должен включать в себя:

* Статус оплаты;
* Идентификатор операции.

Функция должна позволять получать статус оплаты для электронной и виртуальной карты оплаты проезда.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 4: Получение доступных для покупки билетов для носителя (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение доступных для покупки билетов для носителя. Запрос должен включать в себя номер транспортной карты.

Функция должна получать и обрабатывать ответ от смежной системы, который должен включать в себя следующие данные:

* Идентификатор билета;
* Название билета;
* Краткое описание;
* Название типа продукта;
* Идентификатор типа продукта;
* Ссылка на иконку;
* Ссылка на изображение;
* Минимальная сумма;
* Максимальная сумма.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 5: Получение доступных для покупки билетов для виртуальной карты оплаты проезда (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение доступных для покупки билетов для виртуальной карты оплаты проезда. Запрос должен включать в себя следующие данные:

* Wallet-данные устройства для хранения билета/токена Google/Samsung/Apple;
* Информация об устройстве.

Функция должна получать и обрабатывать ответ от смежной системы, который должен включать в себя следующие данные:

* Идентификатор билета;
* Название билета;
* Краткое описание;
* Название типа продукта;
* Идентификатор типа продукта;
* Ссылка на иконку;
* Ссылка на изображение;
* Минимальная сумма;
* Максимальная сумма.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 6: Удаление привязанной банковской карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на удаление привязанной банковской карты. Запрос должен включать в себя идентификатор привязанной банковской карты.

Функция должна получать и обрабатывать ответ от смежной системы, который сообщает о результатах удаления привязанной банковской карты.

Функция 7: Получение списка банковских карт (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение списка банковских карт.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* Маскированный номер карты;
* Тип платёжной системы;
* Идентификатор привязанной карты;
* Альтернативное название привязываемой карты для дальнейшего отображения пользователю на страницах выбора карты;
* Данные о последней операции.

Функция 8: Получение формы привязки банковской карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение формы привязки банковской карты.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* Идентификатор сессии;
* URL для перехода на страницу авторизации стороннего сервиса;
* URL, на который сторонний сервис перенаправит пользователя, после завершения авторизации.

Функция 9: Подтверждение привязки банковской карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на подтверждение привязки банковской карты. Запрос должен содержать в себе следующие данные:

* идентификатор сессии;
* произвольное название карты.

Функция должна получать и обрабатывать ответ от смежной системы, который сообщает о статусе привязки банковской карты.

Функция 10: Получение данных о поездке (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение данных о поездке. Запрос должен содержать идентификатор поездки.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* Идентификатор поездки;
* Отображаемое название поездки;
* Информация о поездке;
* Информация об операциях по транспортной карте при поездке;
* Информация о транспортной карте.

Функция 11: Получение истории поездок (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение истории поездок. Запрос должен включать следующие данные:

* Количество записей на странице;
* Идентификатор транспортной карты;
* Период с;
* Период по.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит список поездок, со следующей информацией о каждой поездке:

* Идентификатор поездки;
* Отображаемое название поездки;
* Информация о поездке;
* Информация об операциях по транспортной карте при поездке;
* Информация о транспортной карте.

Функция 12: Получение параметров для чтения с карты через NFC (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение параметров для чтения с карты через NFC. Запрос должен включать в себя уникальный идентификатор.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* Идентификатор сессии
* Параметры для считывания данных с карты

Функция 13: Получение возможных для покупки продуктов, записанных билетов и отложенных пополнений (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение возможных для покупки продуктов, записанных билетов и отложенных пополнений. Запрос должен включать следующие данные:

* Идентификатор сессии. Передается в случае карт Mifare;
* Считанные данные с карты Ultralite в формате MIME base64. Передается в случае карт Ultralite;
* Запрашивать ли набор параметров сектора для записи, если есть отложенные билеты либо отложенное пополнение баланса. По умолчанию true. Передается в случае карт Mifare.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* Печатный номер карты;
* Баланс;
* Список записанных билетов;
* Отложенные билеты;
* Возможные для покупки продукты.

Функция 14: Подтверждение записи на носитель отложенного пополнения и отложенных билетов (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на подтверждение записи на носитель отложенного пополнения и отложенных билетов. Запрос должен включать в себя:

* Уникальный идентификатор;
* Идентификатор сессии.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит подтверждение записи.

Функция 15: Получение параметров при переносе баланса/билетов транспортных карт с блокировкой (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение параметров при переносе баланса/билетов транспортных карт с блокировкой. Запрос должен включать в себя идентификатор сессии.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит параметры при переносе баланса/билетов транспортных карт с блокировкой.

Функция 16: Подтверждение переноса билетов/баланса с блокировкой карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на подтверждение переноса билетов/баланса с блокировкой карты. Запрос должен включать в себя:

* Идентификатор сессии;
* Уникальный идентификатор;
* Параметры для считывания данных с карты.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит:

* Идентификатор сессии;
* Набор параметров сектора записи.

Функция 17: Привязка транспортной карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на привязку транспортной карты. Запрос должен включать в себя:

* Номер привязываемой транспортной карты;
* Название привязываемой транспортной карты;
* Перечень идентификаторов станций, по которым будет проведена проверка возможности привязки;
* Типы привязок.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит:

* Информацию по привязанной карте (носителе);
* Информацию о транспортной карте, которая ожидает привязки.

Функция 18: Изменение названия привязанной транспортной карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на изменение названия привязанной транспортной карты. Запрос должен в себя включать:

* Идентификатор транспортной карты;
* Новое название карты.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результаты изменения названия.

Функция 19: Отмена привязки носителя (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на отмену привязки носителя. Запрос должен содержать идентификатор транспортной карты.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результаты отмены привязки.

Функция 20: Получение списка носителей (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение списка носителей.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит:

* Перечень привязанных карт;
* Перечень карт, ожидающих привязки.

Функция 21: Просмотр информации о носителе пользователя (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на просмотр информации о носителе пользователя. Запрос должен содержать идентификатор транспортной карты.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит:

* Номер транспортной карты;
* Название транспортной карты;
* Тип носителя;
* Статус носителя;
* Информация по балансу карты;
* Информация о билетах;
* Отложенные записи;
* Дата и время в UTC, с которых доступен просмотр истории проходов и операций по карте.

Функция 22: Перенос в случае утраты / поломки носителя (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на перенос баланса в случае утраты / поломки носителя. Запрос должен содержать:

* Идентификатор транспортной карты;
* Идентификатор привязанной транспортной карты, на который необходимо выполнить перенос.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результаты переноса.

Функция 23: Валидация транспортной карты (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на валидацию транспортной карты. Запрос должен содержать номер транспортной карты и цель валидации карты.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит данные валидированной карты:

* Номер транспортной карты;
* Название транспортной карты;
* Тип носителя;
* Статус носителя;
* Информация по балансу карты;
* Информация о билетах;
* Отложенные записи;
* Дата и время в UTC, с которых доступен просмотр истории проходов и операций по карте.

Функция 24: Перенос баланса в случае, когда обе карты активны (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на перенос баланса в случае, когда обе карты активны. Запрос должен содержать:

* Идентификатор транспортной карты, с которой выполняется перенос;
* Идентификатор транспортной карты, на которую выполняется перенос;
* Сумма для переноса.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результат переноса.

Функция 25: Получение операции по привязанной карте (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение операции по привязанной карте. Запрос должен содержать идентификатор операции.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит:

* Идентификатор операции;
* Отображаемое название операции;
* Тип операции;
* Статус операции;
* Информация о транспортной карте;
* Дата и время операции;

Функция 26: Получение истории операции (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение истории операции. Запрос должен включать параметры фильтра операций.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит историю операций со следующими данными:

Идентификатор операции;

Отображаемое название операции;

* Тип операции;
* Статус операции;
* Информация о транспортной карте;
* Дата и время операции;
  + - 1. Требования к функциям интеграции с картографической системой

Функция 1: Интеграция с картографической системой (этап 4).

Функция должна осуществлять интеграцию с картографической системой и позволять получать данные об организациях.

* + - 1. Требования к функциям интеграции с сервисом биометрического распознавания для оплаты проезда

Функция 1: Регистрация учетной записи (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на регистрацию в сервисе. Запрос должен включать следующие данные:

* email;
* пароль.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать результат регистрации.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 2: Проверка статуса учетной записи (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на проверку статуса учетной записи и диагностического сообщения. Запрос должен включать в себя:

* email;
* пароль.

Функция должна получать и обрабатывать ответ от смежной системы, который должен содержать следующие данные:

* тип диагностического сообщения;
* диагностическое сообщение.

Функция должна обрабатывать следующие типы диагностических сообщений:

* информационные сообщения;
* предупреждения.

Функция должна выявлять следующие ситуации при помощи диагностических сообщений:

1. Нет фотографии. Пожалуйста, добавьте фотографию для извлечения биометрического вектора;
2. Проблемы с обработкой платежа с банковской карты;
3. Нет привязанной банковской карты;
4. Ожидается окончание обработки платежа;
5. Неожидаемо долгая обработка платежа. Обновите холдирование или дождитесь окончания обработки;
6. Не захолдированы средства с карты, вероятно возникли проблемы с обработкой платежа. Пожалуйста, обновите холдирование либо добавьте другую банковскую карту;
7. Рассинхронизация состояния готовности аккаунта;
8. Невозможно извлечь биометрический вектор из фотографии. Пожалуйста, установите другую фотографию;
9. Идёт процесс извлечения биометрического вектора из фотографии;
10. Обновление холдирования;
11. Блокировка повторного прохода.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 3: Получение клиентских настроек и ключевых данных учётной записи (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение и сохранение клиентских настроек и ключевых данных учётной записи. Запрос должен содержать следующие данные:

* email;
* пароль;
* настройки фильтра настроек и данных.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* пользовательские настройки;
* данные о пользователе.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 4: Сохранение пользовательских настроек (этап 2).

Функция должна формировать и отправлять запрос на сохранение пользовательских настроек. Запрос должен содержать следующие данные:

* email;
* пароль;
* настройки и данные из клиентского приложения.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результат сохранения данных и настроек.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 5: Изменение пароля учётной записи (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на изменение пароля учетной записи. Запрос должен содержать следующие данные:

* email;
* пароль;
* новый пароль.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит результаты изменения пароля

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 6: Начало процесса регистрации банковской карты (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на начало процесса регистрации банковской карты. Запрос должен содержать следующие данные:

* email;
* пароль.

Функция должна принимать и обрабатывать ответ, который содержит следующие данные:

* идентификатор платежа;
* авторизационный URL;
* параметры холдирования.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 7: Завершение процесса регистрации банковской карты (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на завершение процесса регистрации банковской карты. Запрос должен содержать следующие данные:

* email;
* пароль;
* авторизационный URL.

Функция должна принимать и обрабатывать ответ от смежной системы, который содержит результат завершения процесса регистрации.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 8: Отмена регистрации банковской карты (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на отмену регистрации банковской карты. Запрос должен содержать следующие данные:

* email;
* пароль;
* id платежа.

Функция должна принимать и обрабатывать ответ от смежной системы, который содержит результат отмены регистрации.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 9: Перехолдирование банковской карты клиентом (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на перехолдирование банковской карты. Запрос должен включать в себя следующие данные:

* email;
* пароль.

Функция должна получать и обрабатывать ответ от смежной системы с результатом перехолдирования банковской карты.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 10: Получение истории успешных проходов через турникеты (этап 2).

Функция должна формировать и отправлять запрос в смежную систему на получение истории успешных проходов через турникеты. Запрос должен включать в себя следующие данные:

* email;
* пароль.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* номер прохода;
* название места прохода;
* время прохода.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

* + - 1. Требования к функциям интеграции с сервисом перекрытий

Функция 1: Запрос данных о перекрытиях (этап 3).

Функция должна формировать и отправлять запрос в смежную систему на получение данных о перекрытиях дорог. Запрос должен содержать следующие данные:

* токен доступа;
* параметры фильтрации.

Функция должна получать и обрабатывать ответ от смежной системы, который включает в себя следующие данные:

* id перекрытия в системе;
* id типа перекрытия;
* id статуса перекрытия;
* описание перекрытия;
* время начала;
* время окончания;
* ограничение скорости движения, км/ч;
* количество полос для движения;
* снижение пропускной способности;
* отметка о согласовании перекрытия;
* номер комиссии;
* заказчик;
* координаты перекрытия;
* вид (описание) проводимых работ;
* адрес;
* количество перекрытых полос;
* создатель записи;
* время создания записи;
* автор последнего изменения;
* время последнего изменения;
* id портала открытых данных Правительства Москвы (data.mos.ru);
* id типа проводимых работ;
* подрядная организация;
* номер телефона;
* id административного округа;
* путь к прикрепленному файлу.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

* + - 1. Требования к функциям интеграции обратной связи

Требования к функциям интеграции обратной связи должны быть уточнены на стадии технического проектирования в ЧТЗ.

Функция 1: Чат со специалистом транспортной службы (этап 5).

Функция должна осуществлять интеграцию с готовым сервисом, позволяющим общаться со специалистом при помощи живого чата.

Функция 2: Чат-бот для оперативного решения проблем пользователя (этап 4).

Функция должна осуществлять интеграцию с готовым сервисом, позволяющим реализовать функционал чат-бота, который упростит поиск ответов на часто задаваемые вопросы по работе транспортного комплекса с возможность расширения списка вопросов и ответов.

Функция 3: Интеграция с сервисом обработки обращений (этап 4).

Функция должна реализовывать интеграцию с сервисом обработки обращений и позволять отправлять обращения по вопросам работы различных категорий транспорта.

Функция должна позволять отправлять обращения следующих категорий:

* метро;
* велосипеды;
* электрички;
* парковки;
* остановки.

Функция 4: Получение ответов на обращения (этап 4).

Функция должна позволять получать ответы от сервиса обработки обращений через интеграцию.

* + - 1. Требования к функциям интеграции с сервисом загруженности дорог и пробок

Функция 1: Индикатор загруженности дорог (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на получение данных о загруженности дорог.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит следующие данные:

* текущий индекс загруженности;
* текущий балл;
* индикатор поведения балла;
* дата и время, в которое обновился сервис.

Функция 2: Средняя скорость на участке дороги (этап 1).

Функция должна формировать и отправлять запрос в смежную систему на получение средней скорости на участке дороги.

Функция должна получать и обрабатывать ответ от смежной системы, который содержит список, состоящий из следующих данных:

* номер участка дороги в системе OSM;
* средняя скорость на участке.
  + - 1. Требования к функциям интеграции с сервисом системы лояльности и геймификации

Функция 1: Функция интеграции с сервисом системы лояльности и геймификации (этап 4).

Требования к функциям интеграции должны быть определены на стадии технического проектирования в ЧТЗ.

* + - 1. Требования к функциям интеграции с сервисом микротранзита

Функция 1: Получение рейсов.

Функция должна формировать и отправлять запрос в смежную систему на получение списка рейсов, который может содержать следующие параметры:

* точка начала поездки;
* точка завершения поездки;
* информация о пассажире;
* количество пассажиров;
* настройки рейса.

В качестве настройки рейса функция должна позволять передавать информацию о необходимости предоставления транспорта для людей на инвалидных колясках.

Точки начала и завершения поездки должны содержать следующие параметры:

* долгота;
* широта.

Информация о пассажире должна содержать следующие параметры:

* ID пассажира ;
* 4 последние цифры номера телефона.

Функция должна позволять получать и обрабатывать ответы от смежной системы через интеграцию. Ответ должен содержать список рейсов. Рейс может содержать следующие параметры:

* идентификатор рейса;
* тип рейса;
* точка начальной остановки;
* время прибытия на начальную остановку;
* точка конечной остановки;
* время прибытия на конечную остановку;
* расстояние до начальной остановки;
* предположительное время пешего перехода до начальной остановки;
* стоимость.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 2: Бронирование рейса.

Функция должна формировать и отправлять запрос в смежную систему на бронирование рейса, который может содержать следующие параметры:

* идентификатор рейса.

Функция должна позволять получать и обрабатывать ответы от смежной системы через интеграцию. Ответ должен содержать следующие параметры:

* идентификатор забронированного рейса;
* статус рейса.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 3: Отмена рейса.

Функция должна формировать и отправлять запрос в смежную систему на отмену рейса, который может содержать следующие параметры:

* идентификатор рейса.

Функция должна позволять получать и обрабатывать ответы от смежной системы через интеграцию. Ответ должен содержать следующие параметры:

* идентификатор забронированного рейса;
* статус рейса.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 4: Получение информации о рейсе.

Функция должна формировать и отправлять запрос в смежную систему на получение информации о рейсе, который может содержать следующие параметры:

* идентификатор рейса.

Функция должна позволять получать и обрабатывать ответы от смежной системы через интеграцию. Ответ должен содержать следующие параметры:

* идентификатор забронированного рейса;
* информация о рейсе.

Информация о рейсе должна содержать следующие параметры:

* статус рейса;
* причина отмены рейса, если он был отменен;
* точка начальной остановки;
* точка конечной остановки;
* время прибытия на начальную остановку;
* время прибытия на конечную остановку;
* информация об автобусе;
* информация о водителе;
* настройки рейса;

Информация об автобусе должна содержать следующие параметры:

* лицензия;
* цвет автобуса;
* модель автобуса;
* идентификатор наемного автобуса;
* текущее местоположение;

Информация о водителе должна содержать следующие параметры:

* имя;
* фамилия.

Функция 5: Получение зоны работы сервиса.

Функция должна формировать и отправлять запрос в смежную систему на получение зоны работы сервиса.

Функция должна позволять получать и обрабатывать ответы от смежной системы через интеграцию. Ответ должен содержать следующие параметры:

* идентификатор зоны;
* полигон зоны.
  + - 1. Требования к функциям интеграции с сервисом оплаты банковской карты

Функция 1: Регистрация в процессинговом центре

Функция должна формировать и отправлять запрос в смежную систему на регистрацию заказа в процессинговом центре (ПЦ), который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* сумма заказа в минимальных единицах валюты;
* код валюты по ISO4217;
* описание Заказа;
* цифровая подпись Запроса;
* адрес Плательщика;
* название населённого пункта Плательщика;
* язык отображения Платёжных страниц;
* номер Заказа на стороне ТСП;
* время жизни Заказа в секундах — период, в течение которого разрешена оплата Заказа;
* эквайринговая комиссия в минимальных единицах валюты;
* адрес страницы на стороне ТСП, на которую по завершении Операции переводится Плательщик вместо показа ему типового чека ПЦ;
* адрес страницы на стороне ТСП, на которую переводится Плательщик в случае неуспешного окончания проведения Операции;
* комментарий к заказу;
* адрес электронной почты Плательщика;
* телефон Плательщика;
* адрес электронной почты получателя денежных средств;
* телефон получателя денежных средств;
* двухсимвольный код страны Плательщика в соответствии с ISO 3166-1
* название региона (области, края) Плательщика;
* наименование банка Плательщика;
* имя Плательщика;
* фамилия Плательщика;
* отчество Плательщика;
* фамилия имя отчество получателя денежных средств;
* номер счета получателя денежных средств;
* номер договора получателя денежных средств;
* идентификатор клиента на стороне ТСП;
* платёжная система, используемая для оплаты;
* признак, отвечающий за необходимость отправки Плательщику email-уведомления о зарегистрированном Заказе со ссылкой на оплату;
* месяц срока действия Карты Плательщика;
* год срока действия Карты Плательщика;
* ID Операции в ПЦ;
* значение для передачи данных обо всех дополнительных платежах;
* опция, отвечающая за необходимость отправки sms-уведомления (счета) со ссылкой на оплату.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 2: Оплата по карте.

Функция должна формировать и отправлять запрос в смежную систему на оплату по карте, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* цифровая подпись;
* проверочный код Карты Плательщика CVV2/CVC2/ППК2;
* запрос на предоставление использованной в Операции платёжной карты в составе уведомления на url;
* токен ранее зарегистрированной Платёжной карты Плательщика;
* действие, выполняемое ПЦ;
* идентификатор клиента на стороне ТСП;
* значение для передачи номера карты, по которой будет проводиться операция;
* значение хеша, взятого по алгоритму sha256 от номера карты, по которой будет проводиться операция;
* настройка сохранения данных карты.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 3: Холдирование средств на карте.

Функция должна формировать и отправлять запрос в смежную систему на холдирование средств, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* уникальный идентификатор Заказа в ПЦ;
* цифровая подпись;
* проверочный код Карты Плательщика CVV2/CVC2/ППК2;
* запрос на предоставление токена;
* токен ранее зарегистрированной Платёжной карты Плательщика;
* действие, выполняемое ПЦ;
* идентификатор клиента на стороне ТСП
* значение для передачи номера карты, по которой будет проводиться операция;
* значение хеша, взятого по алгоритму sha256 от номера карты, по которой будет проводиться операция;
* настройка сохранения данных карты.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 4: Завершение оплаты по карте.

Функция должна формировать и отправлять запрос в смежную систему на списание захолдированных средств, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* уникальный идентификатор Заказа в ПЦ;
* сумма списания
* код валюты по ISO4217;
* цифровая подпись;
* эквайринговая комиссия в минимальных единицах валюты.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 5: Отмена операции оплаты

Функция должна формировать и отправлять запрос в смежную систему на отмену последней успешной операции, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* уникальный идентификатор Заказа в ПЦ;
* сумма Возврата или Отмены в минимальных единицах валюты;
* код валюты по ISO421;
* цифровая подпись.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 6: Получение информации об операции

Функция должна формировать и отправлять запрос в смежную систему на получение информации об операции, который может содержать следующие параметры:

* уникальный идентификатор Заказа в ПЦ;
* уникальный идентификатор Операции в ПЦ;
* цифровая подпись;
* запрос на предоставление токена использованной в Операции Платёжной карты в составе уведомления на url.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 7: Регистрация карты и получение токена.

Функция должна формировать и отправлять запрос в смежную систему на регистрацию карты и получение токена, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* уникальный идентификатор Заказа в ПЦ;
* цифровая подпись.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 8: Оплата и холдирование по токену карты.

Функция должна формировать и отправлять запрос в смежную систему на оплату и холдирование по токену карты, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* уникальный идентификатор Заказа в ПЦ;
* токен ранее зарегистрированной карты;
* цифровая подпись;
* CVV2/CVC2.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 9: Оплата через Google Pay.

Функция должна формировать и отправлять запрос в смежную систему на оплату через сервис Google Pay, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* уникальный идентификатор Заказа в ПЦ;
* полученные от сервера Google платежные реквизиты Плательщика: преобразованное в текст base64-кодированное значение параметра;
* цифровая подпись;
* тип выполняемой Операции;
* адрес электронной почты Плательщик.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

Функция 10: Оплата через Apple Pay.

Функция должна формировать и отправлять запрос в смежную систему на оплату через сервис Apple Pay, который может содержать следующие параметры:

* уникальный идентификатор учетной записи ТСП в ПЦ;
* код валюты;
* сумма Заказа в минимальных единицах валюты;
* платежные реквизиты Плательщика, полученные от устройства Apple в виде значения платежного токена;
* платежные реквизиты Плательщика, полученные от устройства Apple в виде значения параметра;
* номер заказа на стороне ТСП;
* контактные данные плательщика;
* уникальный идентификатор заказа в ПЦ;
* формат данных результата операции;
* эквайринговая комиссия в минимальных единицах валюты;
* описание заказа;
* язык отправки уведомлений;
* адрес для отправки чека и почтового уведомления по операции.

Функция должна позволять получать ответы от смежной системы через интеграцию.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

* + - 1. Требования к функциям интеграции с сервисом загруженности вагонов

Функция 1: Получение загруженности вагонов

Функция должна формировать и отправлять запрос в смежную систему на получение загруженности вагонов метро.

Функция должна позволять получать ответы от смежной системы через интеграцию. Ответ на запрос должен содержать следующие параметры:

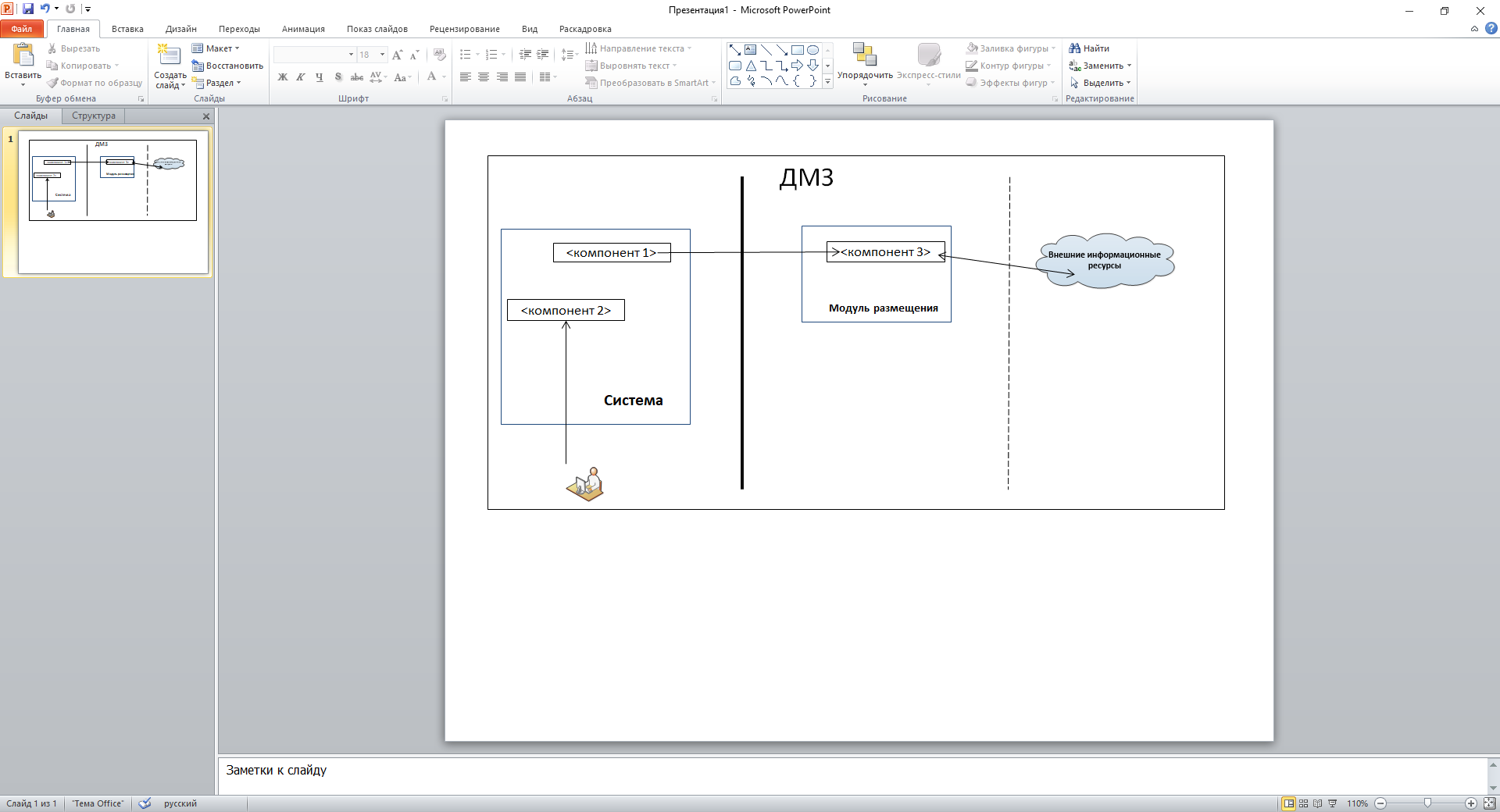
* метка успешности обработки запроса;
* ключ – id предыдущей или следующей станции;
* идентификатор состава (номер головного вагона и последнего);
* номер пути;
* предыдущая станция id;
* следующая станция id;
* время до прибытия в секундах;
* индекс поезда;
* загруженность конкретного вагона.

Список передаваемых и получаемых данных может быть изменен в зависимости от сервиса, с которым производится интеграция.

* + 1. Требования к функциям подсистемы безопасности

1. **Требования к схеме размещения компонентов информационного взаимодействия**

Компоненты модуля информационного взаимодействия должны обеспечивать возможность получения данных из новых внешних источников и передачу компонентам Системы, обеспечивающим их дальнейшую обработку и использование в соответствии с требованиями настоящего Технического задания, по следующей схеме.



Архитектурные и технологические решения организации ДМЗ должны быть выполнены с учетом требований действующего законодательства Российской Федерации в сфере защиты информации и согласованы с Заказчиком на стадии проектирования функционала Системы.

Требования к функциям модуля размещения компонентов информационного взаимодействия.

1. **Требования к функциям обработки выявленных ошибок**

Функции подсистемы должны обеспечить реализацию процесса передачи сведений о выявленных ошибках в качестве предоставляемой информации в информационные системы, являющиеся поставщиками данных. В рамках данного процесса должно быть обеспечено однозначное соответствие между данными, поступающими из источников, и информацией об ошибках, возвращаемой в источник. Данное соответствие достигается путем присвоения уникального идентификатора каждому экземпляру сущности, передаваемой из источников. При выявлении проблем с качеством данных для каких-то экземпляров сущностей, информация об этом передается обратно в источник вместе с вышеописанными идентификаторами экземпляров сущностей. Уникальность идентификатора должна соблюдаться в рамках одного типа сущностей для конкретной информационной системы. Т.е. экземпляры сущности в разных системах могут иметь одинаковые идентификаторы, но в рамках одной системы идентификаторы уникальны.

Соответствующие требования должны выполняться для всех функций интеграционного взаимодействия с информационными ресурсами, являющимися источником данных для Системы:

АИС КПП;

РНИС;

РНИС МО;

АИС ОВСБ;

АСУ «Е-Автовокзал»;

АИС ОЭСЮ;

AИС ОЭСВ;

АИС ОТЯ;

АИС ОТСМ;

АИС ПОД;

АИС ЯЭ;

КД ДОС;

АИС ДСТ;

АИС НС;

АИС ПС;

КС ОСМ;

КС МБ;

АИС ЕСОО;

АМ ВК;

АМ ФБ;

АМ ОК;

АМ ГС;

АМ НС;

ПС ПТК;

АИС АПНС;

АИС ГФ;

АИС ССЯ;

АИС ССА.

* + - 1. Требования к функциям взаимодействия с внешними информационными ресурсами в ДМЗ

Должны быть доработаны функции модуля размещения компонентов информационного взаимодействия, обеспечивающие передачу сведений между подсистемой интеграции и внешними информационными ресурсами, в части взаимодействия со следующими ресурсами:

АИС КПП – получение;

РНИС – получение;

РНИС МО – получение;

АИС ОВСБ – получение данных и передача данных;

АСУ «Е-Автовокзал» – получение;

АИС ОЭСЮ – получение данных и передача данных;

AИС ОЭСВ – получение данных и передача данных;

АИС ОТЯ – получение данных и передача данных;

АИС ОТСМ – получение данных и передача данных;

АИС ПОД – получение;

АИС ЯЭ – получение;

КД ДОС – получение;

АИС ДСТ – получение данных и передача данных;

АИС НС – получение данных и передача данных;

АИС ПС – получение;

КС ОСМ – получение;

КС МБ – получение;

АИС ЕСОО – получение данных и передача данных;

АМ ВК – получение данных и передача данных;

АМ ФБ – получение данных и передача данных;

АМ ОК – получение данных и передача данных;

АМ ГС – получение данных и передача данных;

АМ НС – получение данных и передача данных;

ПС ПТК – получение данных и передача данных;

АИС АПНС – получение данных и передача данных;

АИС ГФ – получение данных и передача данных;

АИС ССЯ – получение данных и передача данных;

АИС ССА – получение данных и передача данных.

Должны быть разработаны функции логирования процедур обмена данными для каждой из перечисленных функций взаимодействия с внешними ресурсами. Параметры логирования должны быть согласованы с Заказчиком на стадии технического проектирования

* 1. Требования к интерфейсу приложения

Должен быть выполнен редизайн приложения с учетом нового и доработанного функционала Системы. Редизайн должен включать в себя следующие работы:

– создание новой стилистики приложения;

– обновление структуры приложения с учетом нового функционала;

– разработка макетов интерфейса;

– разработка нового интерфейса на основании макетов, в т.ч. «темная тема».

Также установлены требования к пользовательским сценариям.

Пользовательские сценарии должны быть доработаны с учетом обновленного и разработанного функционала и обновленной структуры приложения.

* 1. Сопутствующие работы

Цена ГК включает в себя стоимость всех работ, предусмотренных настоящим ТЗ, а также все затраты, издержки и иные расходы Подрядчика, в том числе сопутствующие, связанные с исполнением настоящего ТЗ.

* 1. Порядок сдачи-приемки
     1. Порядок выполнения работ

График выполнения работ по развитию Системы приведен в Приложении 1 к ТЗ.

В рамках выполнения работ по развитию Системы, на срок выполнения работ в рамках данного ТЗ, Заказчик обеспечивает передачу исходного кода программного обеспечения Подрядчику в течение 5 календарных дней со дня заключения Государственного контракта. Права на передаваемое ПО в полном объеме принадлежат городу Москве, от имени которого выступает Заказчик.

Подрядчик вправе привлечь к выполнению работ по настоящему ТЗ других лиц - субподрядчиков, обладающих специальными знаниями, навыками, квалификацией, специальным оборудованием и т.п., по видам (содержанию) работ, предусмотренных в настоящем ТЗ. При этом Подрядчик несет ответственность перед Заказчиком за исполнение и надлежащее качество работ.

Заказчик в праве запросить информацию о компаниях-субподрядчиках, привлекаемых Исполнителем на каждом этапе выполнения работ. Исполнитель в срок не позднее 5 календарных дней с момента получения запроса, предоставляет список привлеченных компаний-субподрядчиков и информацию о них.

* + 1. Порядок оплаты

Заказчик в течение 30 календарных дней от даты выставления счета Подрядчиком выплачивает аванс в размере 20% (двадцати процентов) от стоимости этапа выполнения работ.

Счет на выплату аванса предоставляется Подрядчиком по первому этапу - в течение 10 календарных дней с даты заключения Государственного контракта и в течение 10 календарных дней с начала каждого последующего этапа выполнения работ.

Оплата по Государственному контракту производится Заказчиком по завершению каждого этапа выполнения работ, на основании надлежаще оформленных и подписанных обеими сторонами Акта сдачи-приёмки выполненных работ по этапу, а за пятый этап – на основании надлежаще оформленных и подписанных обеими сторонами Акта сдачи-приемки выполненных работ по этапу, Акт об исполнении обязательств и Акта приема-передачи исключительных прав на программное обеспечение Системы, за вычетом пропорционально выплаченного ранее аванса, не более чем в течение 30 календарных дней с даты подписания Акта сдачи-приемки выполненных работ по этапу, а за пятый этап с даты подписания Акта сдачи-приемки выполненных работ по этапу, Акт об исполнении обязательств и Акта приема-передачи исключительных прав на программное обеспечение Системы.

Счёт и счет-фактура выставляется Подрядчиком в течение 2 (двух) рабочих дней после подписания Заказчиком Акта сдачи-приёмки выполненных работ по этапу, а за пятый этап – Акта об исполнении обязательств и Акта приема-передачи исключительных прав на программное обеспечение Системы.

В случае неисполнения или ненадлежащего исполнения Подрядчиком обязательств, предусмотренных Государственным контрактом, Заказчик производит оплату по Государственному контракту за вычетом соответствующего размера неустойки.

* + 1. Порядок контроля и приемки работ

По окончании каждой работы, предусмотренной Графиком выполнения работ (Приложение 1), Подрядчик представляет Заказчику документы, подтверждающие их выполнение.

Подрядчик в срок не позднее 5 (пяти) рабочих дней после окончания завершения этапа выполнения работ письменно уведомляет Заказчика о факте завершения выполнения соответствующих работ.

Приемка результатов выполнения работ осуществляется приемочной комиссией, назначаемой Заказчиком. В состав приемочной комиссии включаются представители Заказчика и Подрядчика, а также специалисты, привлекаемые Заказчиком. Приемочная комиссия формируются Заказчиком на основании распорядительного документа, который должен определять состав комиссии, порядок их работы, место и сроки проведения приемки выполненных работ.

Приемка результатов выполнения работ оформляется Актом сдачи-приёмки выполненных работ по каждому этапу и Актом приема-передачи исключительных прав на программное обеспечение Системы, разработанное в результате работ по каждому этапу. Основанием для составления и подписания Акта сдачи-приемки выполненных работ по этапу является передача Подрядчиком результатов выполненных работ в соответствии с условиями Государственного контракта и отчетных документов, указанных в Графике выполнения работ по развитию Системы.

Приемка результатов выполненных работ составляет 15 (пятнадцать) рабочих дней после получения от Подрядчика документов, Заказчик рассматривает результаты и осуществляет приемку выполненных работ на предмет соответствия их объема и качества требованиям, изложенным в настоящем ТЗ, и направляет заказным письмом с уведомлением, либо отдает Подрядчику подписанный Заказчиком 1 (один) экземпляр Акта сдачи-приемки выполненных работ по этапу, либо запрос о предоставлении разъяснений касательно результатов выполненных работ, или мотивированный отказ от принятия результатов выполненных работ, или Акт с перечнем выявленных недостатков, необходимых доработок и сроком их устранения. В случае отказа Заказчика от принятия результатов выполненных работ в связи с необходимостью устранения недостатков и/или доработки результатов выполненных работ, Подрядчик обязуется в срок, установленный в Акте, составленном Заказчиком, устранить указанные недостатки/произвести доработки за свой счет.

Для проверки предоставленных Подрядчиком результатов, предусмотренных Государственным контрактом, в части их соответствия условиям Государственного контракта, Заказчик обязан провести экспертизу. Экспертиза результатов, предусмотренных ГК, может проводиться Заказчиком своими силами или к ее проведению могут привлекаться эксперты и/или экспертные организации.

* + 1. Общие требования к приемке работ

Для проверки предоставленных Подрядчиком результатов, предусмотренных ГК, в части их соответствия условиям ГК Заказчик обязан провести испытания Системы.

Испытания должны быть организованы и проведены в соответствии с ГОСТ 34.603-92 «Информационная технология. Виды испытаний автоматизированных систем».

Должны быть проведены следующие виды испытаний в соответствии с Графиком выполнения работ:

* предварительные испытания;
* опытная эксплуатация;
* приемочные испытания.

Виды, состав, объем, и методы испытаний определяются в документах «Программа и методика предварительных испытаний», «Программа и методика опытной эксплуатации» и «Программа и методика приемочных испытаний» с указанием контрольных тестов, проводимых для каждого вида испытаний. Указанный документ разрабатывается Подрядчиком в соответствии с ГОСТ 34.603-92 «Информационная технология. Виды испытаний автоматизированных систем» и предоставляется Заказчику не позднее, чем за 5 календарных дней до проведения испытаний.

Результаты проведения испытаний должны быть зафиксированы в соответствующих Протоколах испытаний. Недостатки реализации оформляются как выявленные отклонения от ТЗ.

Сроки проведения предварительных испытаний устанавливаются отдельно по каждому из этапов выполнения работ, в соответствии с Графиком выполнения работ (Приложение 1).

После завершения предварительных испытаний, в случае наличия недостатков и замечаний, Подрядчик представляет и согласовывает с Заказчиком Протокол разногласий с перечнем замечаний и сроками их устранения.

Подрядчик в течение согласованного времени устраняет указанные в Протоколе недостатки. Факт устранения недостатков подтверждается Протоколом устранения недостатков. По результатам оформляется Акт о приемке Системы в опытную эксплуатацию.

Длительность проведения опытной эксплуатации – 15 календарных дней, в соответствии с Графиком выполнения работ (Приложение 1).

На этапе опытной эксплуатации Заказчиком ведется Журнал опытной эксплуатации, в который заносятся выявленные Заказчиком замечания (несоответствие Системы требованиям настоящего ТЗ) и пожелания к Системе или технической/рабочей документации. В ходе опытной эксплуатации замечания к Системе и документации устраняются Подрядчиком. Пожелания к Системе могут быть реализованы Подрядчиком в ходе опытной эксплуатации или отнесены к развитию Системы вне рамок работ настоящего ТЗ. Не менее чем за неделю до опытной эксплуатации Подрядчиком проводится консультирование Модераторов, Администраторов и Системных администраторов Системы.

Результаты опытной эксплуатации отражаются в документе «Журнал опытной эксплуатации» и рассматриваются в ходе приемочных испытаний.

После завершения опытной эксплуатации и устранения недостатков оформляется Акт о завершении опытной эксплуатации. Условием для передачи Системы в эксплуатацию является устранение всех замечаний на уже проведенных испытаниях, опытной эксплуатации и подписанный Заказчиком Акт о готовности Системы к эксплуатации.

Длительность проведения заключительных, приемочных испытаний, устанавливается отдельно по каждому из этапов выполнения работ, в соответствии с Графиком выполнения работ (Приложение 1).

В случае несоответствия Системы требованиям, предъявляемым в настоящем ТЗ, сроки проведения испытаний могут быть перенесены или расширены Заказчиком в пределах сроков выполнения работ в соответствии настоящим ТЗ.

* + 1. Порядок оформления и предъявления Заказчику результатов работ по доработке Системы

Документация на доработанную Систему передается на бумажном, оптическом или электронном носителях информации. Текстовые документы, передаваемые на оптическом или электронном носителях информации, должны быть представлены в форматах Microsoft Office. Все материалы передаются с сопроводительными документами Подрядчика.

Документация на бумажном носителе передается в 2-х экземплярах, по одному для Заказчика и Подрядчика.

Дистрибутив и исходные коды доработанной Системы должны передаваться на оптическом или электронном носителях информации и должны соответствовать требованиям к программному обеспечению.

Вместе с исходными кодами Системы должны передаваться:

* проектный файл, описывающий структуру файлов исходных текстов, файлов ресурсов, правил компиляции и компоновки и т.п.;
* используемые сторонние библиотеки.
  + 1. Требования к документированию

Корректируемая и/или разрабатываемая в рамках настоящего ТЗ документация должна быть выполнена в соответствии с требованиями ГОСТ 34.201-89.

Подрядчиком должны быть разработаны (актуализированы) и согласованы с Заказчиком документы в соответствии с Приложением 1 к настоящему ТЗ.

Подрядчик должен подготовить актуализированный паспорт Системы для регистрации в Реестре информационных систем и ресурсов города Москвы (в соответствии с постановлением Правительства Москвы от 24.06.2003 № 496-ПП «О Единой системе классификации и кодирования и Едином реестре информационных ресурсов и систем города Москвы») в соответствии с инструкцией по заполнению, предоставляемой Заказчиком Подрядчику не позднее 3-х рабочих дней с момента заключения ГК.

Дистрибутивы и исходные коды Системы должны передаваться на оптическом или электронном носителях информации в соответствии с Графиком выполнения работ (Приложение 1 к настоящему ТЗ) и должны соответствовать требованиям к программному обеспечению. Все материалы передаются с сопроводительными документами Подрядчика. Документация должна быть выполнена на русском языке. Оформление документации с рамками не требуется.

1. Объем и сроки гарантий качества
   1. Требования по объему и сроку гарантий качества выполнения работ

В целях обеспечения взаимодействия в рамках гарантийных обязательств, Подрядчик одновременно с подписанием Государственного контракта обязан предоставить Заказчику контактную информацию (телефонный номер, адрес электронной почты, либо адрес для доставки почтовых отправлений (телеграмм)) и указать ответственное контактное лицо. В случае изменения контактной информации Подрядчик обязан в течение 5 рабочих дней с момента изменения уведомить об этом Заказчика.

Гарантийные обязательства Подрядчика включают в себя принятие претензий и исправление отклонений от требований настоящего ТЗ, которые в том числе не удалось выявить в процессе приемки результатов выполненных работ.

Гарантийное обслуживание проводится Подрядчиком в течение 12 месяцев с момента подписания Акта сдачи-приёмки выполнения работ по Государственному контракту и распространяется на развитую и настроенную в соответствии с настоящим ТЗ Систему.

В случае обнаружения Заказчиком сбоя в работе Системы, Заказчик обязан уведомить об этом Подрядчика по предоставленному телефонному номеру и/или по адресу электронной почты Подрядчика. Подрядчик обязан устранить сбои за свой счет в сроки, согласованные и установленные Подрядчиком и Заказчиком в Акте о недостатках с перечнем выявленных недостатков, необходимых доработок и сроков их устранения. Гарантийный срок в данном случае продлевается на период устранения выявленных недостатков.

При отказе Подрядчика от составления или подписания Акта о недостатках, обнаруженных в период гарантийного срока, Заказчик проводит квалифицированную экспертизу с привлечением экспертов в порядке, установленном Государственным контрактом.

Подрядчик обязуется обеспечить консультирование специалистов Заказчика в течение всего периода гарантийного обслуживания. Для этого Подрядчик обеспечивает доступность консультаций по телефону и электронной почте. Ответ на обращение Заказчика должен быть предоставлен Подрядчиком в течение 24 часов с момента получения обращения от Заказчика.

Подрядчик по запросу Заказчика должен оказывать содействие при внесении изменений по Системе в Реестр информационных систем и ресурсов города Москвы, сформировать и предоставить Заказчику комплект документов, необходимый для регистрации изменений в соответствии с п. 4 Приложения 2 к постановлению Правительства Москвы от 24.06.2003 № 496-ПП «О Единой системе классификации и кодирования и Едином реестре информационных ресурсов и систем города Москвы».

1. Требования к безопасности выполнения работ
   1. Требования по обеспечению информационной безопасности

Подрядчиком должны быть реализованы все необходимые мероприятия, определенные Статьей 2 Постановления Правительства Российской Федерации от 06.07.2015 № 676 «О требованиях к порядку создания, развития, ввода в эксплуатацию, эксплуатации и вывода из эксплуатации государственных информационных систем, и дальнейшего хранения содержащейся в их базах данных информации» (далее – Требования 676-ПП). Реализация мероприятий, определенных Статьей 3 Требований 676-ПП, осуществляется Заказчиком.

Работы должны быть выполнены с учетом требований государственного стандарта Российской Федерации ГОСТ 34.602-89 «Информационная технология. Комплекс стандартов на автоматизированные системы. ТЗ на создание автоматизированной системы», а также требований к созданию систем безопасности значимых объектов КИИ РФ и обеспечению их функционирования, утвержденных приказом ФСТЭК России от 21.12.2017 №235 «Об утверждении Требований к созданию систем безопасности значимых объектов КИИ РФ и обеспечению их функционирования».

В соответствии с произведенными доработками Подрядчиком должны быть актуализированы Акт классификации Системы, Акт категорирования Системы и Модель угроз и нарушителя информационной безопасности Системы.

Аттестация Системы проводится Заказчиком самостоятельно в течение 30 календарных дней после подписания Акта сдачи-приемки выполненных работ по Государственному контракту. В случае, если в процессе проведения аттестации необходимы доработки Системы, которые входят в гарантийные обязательства по Государственному контракту, их производит Подрядчик.

Информационное обеспечение должно быть достаточным для выполнения всех автоматизированных функций Системы.

Должна быть обеспечена защита целостности хранимых данных на уровне операций базового ПО.

Средства администрирования базового программного обеспечения должны обеспечивать:

* возможность полного или частичного восстановления БД в случае возникновения аварийных ситуаций;
* наличие средств сопряжения с системами дублирования на резервные устройства хранения с последующим восстановлением.

Уровень хранения данных в Системе должен быть построен на основе объектно-реляционных СУБД. Для обеспечения логической и физической целостности данных должны использоваться встроенные механизмы СУБД.

Средства СУБД, а также средства используемых операционных систем должны обеспечивать документирование и протоколирование обрабатываемой в системе информации. СУБД должна обеспечивать диагностику производительности своими внутренними средствами.

1. Перечень нормативных правовых и нормативых технических актов
   1. При выполнении работ по развитию и настройке Системы следует учитывать следующие государственные стандарты:

* ГОСТ 34.201-89. Государственный стандарт Союза ССР. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем (утв. Постановлением Госстандарта СССР от 24.03.1989 N 664).
* ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. ТЗ на создание автоматизированной системы (утв. Постановлением Госстандарта СССР от 24.03.1989 N 661).
* ГОСТ 34.601-90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы стадии создания (утв. Постановлением Госстандарта СССР от 29.12.1990 N 3469).
* ГОСТ 34.003-90. Межгосударственный стандарт. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения (утв. и введен в действие Постановлением Госстандарта СССР от 27.12.1990 N 3399).
* РД 50-682-89. Руководящий документ по стандартизации. Методические указания. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Общие положения (утв. и введены в действие Постановлением Госстандарта СССР от 17.02.1989 N 254).
* ГОСТ Р ИСО/МЭК 14764-2002. Информационная технология. Сопровождение программных средств (принят и введен в действие Постановлением Госстандарта РФ от 25.06.2002 N 248-ст).
  1. При выполнении работ по развитию Системы следует учитывать следующие федеральные и региональные нормативные правовые акты:
* Федеральный закон от 27.07.2006 № 149-ФЗ "Об информации, информационных технологиях и о защите информации".
* Федеральный закон от 27.07.2006 № 152-ФЗ "О персональных данных".
* Указ Президента РФ от 05.12.2016 № 646 "Об утверждении Доктрины информационной безопасности Российской Федерации".
* Приказ Минэкономразвития России от 16.11.2009 № 470 "О Требованиях к технологическим, программным и лингвистическим средствам обеспечения пользования официальными сайтами федеральных органов исполнительной власти".
* Постановление Правительства Москвы от 07.02.2012 № 26-ПП "Об утверждении Положения об эксплуатации автоматизированных информационных систем и ресурсов города Москвы".
* Распоряжение Департамента информационных технологий г. Москвы от 27.12.2011 № 64-16-1045/11 "Об утверждении Регламента реализации мероприятий по организации выдачи сертификатов ключей электронных цифровых подписей".
* Распоряжение Департамента экономической политики и развития г. Москвы и Департамента информационных технологий г. Москвы от 01.07.2016 № 18-Р/64-16-264/16 "Об утверждении Порядка расчета планируемой стоимости работ (услуг) по эксплуатации информационных систем и ресурсов города Москвы".
* Распоряжение Департамента информационных технологий г. Москвы от 01.07.2016 № 64-16-265/16 "Об утверждении Порядка взаимодействия участников по организации и осуществлению эксплуатации информационных систем и ресурсов города Москвы".
* Распоряжение Департамента информационных технологий г. Москвы от 01.07.2016 № 64-16-266/16 "Об утверждении Порядка определения и актуализации показателей и коэффициентов Порядка расчета планируемой стоимости работ (услуг) по эксплуатации информационных систем и ресурсов города Москвы".
* Распоряжение Департамента информационных технологий г. Москвы от 01.07.2016 № 64-16-267/16 "Об утверждении Порядка ведения Паспортов информационных систем и ресурсов города Москвы".

# Приложение 1 к Техническому заданию

**График выполнения работ**

| **№**  **п/п** | **Содержание выполняемых работ** | **Результаты, отчетные документы** | **Сроки выполнения работ** | **Стоимость работ, руб. с НДС 20 %** |
| --- | --- | --- | --- | --- |
| **1.** | **Развитие геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (1-этап)** | * **Акт сдачи-приемки выполненных работ по этапу 1** * **Акт приема-передачи исключительных прав на программное обеспечение Системы, разработанное в результате этапа 1** | **30 (тридцать) календарных дней с даты заключения Государственного контракта** | **Заполняется на стадии заключения Контракта** |
| 1.1. | Техно-рабочее проектирование | * Технический проект в составе: * Ведомость документов; * Пояснительная записка Технического проекта; * Частное техническое задание (1 этап); * Общее описание Системы; * Описание программного обеспечения; * Схема функциональной структуры; * Описание комплекса технических средств; * Описание информационного обеспечения Системы; | 2 (два) календарных дня с момента заключения Государственного контракта |
| 1.2. | Доработка системы | * Отчет о проведении пусконаладочных работ; * Программное обеспечение Системы, включая исходные коды на машинном носителе (CD/DVD);   Актуализированная рабочая документация в составе:   * Руководство пользователя; * Руководство администратора; * Программа и методика предварительных испытаний; * Программа и методика опытной эксплуатации. | 8 (восемь) календарных дней с момента окончания техно-рабочего проектирования |
| 1.3. | Предварительные испытания | * Протокол предварительных испытаний; * Протокол устранения недостатков, выявленных в ходе предварительных испытаний (при необходимости); * Акт о приемке Системы в опытную эксплуатацию. | 3 (три) календарных дня с момента окончания доработки системы |
| 1.4. | Опытная эксплуатация | * Журнал опытной эксплуатации; * Акт о готовности Системы к приемочным испытаниям; * Отчет о проведении опытной эксплуатации с приложением журнала опытной эксплуатации; * Акт о завершении опытной эксплуатации; * Программа и методика приемочных испытаний (с приложением проекта протокола приемочных испытаний и проекта акта о готовности Системы к эксплуатации); | 15 (пятнадцать) календарных дней с момента окончания предварительных испытаний |
| 1.5. | Приемочные испытания | * Протокол приемочных испытаний; * Протокол разногласий приемочных испытаний (при необходимости) * Акт о готовности Системы к эксплуатации; * Программное обеспечение модернизированной Системы, включая исходные коды, на машинном носителе (CD/DVD); * Протокол развертывания; * Акт передачи исходного кода и дистрибутива; * Акт о готовности Системы к эксплуатации; * Накладная на передачу исполнительной документации. | 2 (два) календарных дня с даты завершения опытной эксплуатации |
| **2.** | **Развитие геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (2-этап)** | **Акт сдачи-приемки выполненных работ по этапу 2**  **Акт приема-передачи исключительных прав на программное обеспечение Системы, разработанное в результате этапа 2** | **100 (сто) календарных дней с даты завершения 1 этапа** | **Заполняется на стадии заключения Контракта** |
| 2.1. | Техно-рабочее проектирование | * Технический проект в составе: * Ведомость документов; * Пояснительная записка Технического проекта; * Частное техническое задание (2 этап); * Описание программного обеспечения; * Описание информационного обеспечения Системы; | 10 (десять) календарных дней с момента завершения 1 этапа |
| 2.2. | Доработка системы | * Отчет о проведении пусконаладочных работ; * Программное обеспечение Системы, включая исходные коды на машинном носителе (CD/DVD);   Актуализированная рабочая документация в составе:   * Руководство пользователя; * Руководство администратора; * Программа и методика предварительных испытаний; * Программа и методика опытной эксплуатации; * Разработанный дизайн-концепт Мобильного приложения; * Разработанный дизайн экранов и пользовательских сценариев Мобильного приложения для функционала, реализованного в рамках этапа 2; * Разработанное руководство по использованию элементов дизайна мобильного приложения. | 55 (пятьдесят пять) календарных дней с момента окончания техно-рабочего проектирования 2 этапа |
| 2.3. | Предварительные испытания | * Протокол предварительных испытаний; * Протокол устранения недостатков, выявленных в ходе предварительных испытаний (при необходимости); * Акт о приемке Системы в опытную эксплуатацию. | 15 (пятнадцать) календарных дней с момента окончания доработки системы 2 этапа |
| 2.4. | Опытная эксплуатация | * Журнал опытной эксплуатации; * Акт о готовности Системы к приемочным испытаниям; * Отчет о проведении опытной эксплуатации с приложением журнала опытной эксплуатации; * Акт о завершении опытной эксплуатации; * Программа и методика приемочных испытаний (с приложением проекта протокола приемочных испытаний и проекта акта о готовности Системы к эксплуатации); | 15 (пятнадцать) календарных дней с момента окончания предварительных испытаний 2 этапа |
| 2.5. | Приемочные испытания | * Протокол приемочных испытаний; * Протокол разногласий приемочных испытаний (при необходимости) * Акт о готовности Системы к эксплуатации; * Программное обеспечение модернизированной Системы, включая исходные коды, на машинном носителе (CD/DVD); * Протокол развертывания; * Акт передачи исходного кода и дистрибутива; * Акт о готовности Системы к эксплуатации; * Накладная на передачу исполнительной документации. | 5 (пять) календарных дней с даты завершения опытной эксплуатации 2 этапа |
| **3.** | **Развитие геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (3-этап)** | **Акт сдачи-приемки выполненных работ по этапу 3**  **Акт приема-передачи исключительных прав на программное обеспечение Системы, разработанное в результате этапа 3** | **191 (сто девяносто один) календарный день с даты завершения 2 этапа** | **Заполняется на стадии заключения Контракта** |
| 3.1. | Техно-рабочее проектирование | * Технический проект в составе: * Ведомость документов; * Пояснительная записка Технического проекта; * Частное техническое задание (3 этап); * Описание программного обеспечения; * Описание информационного обеспечения Системы; | 20 (двадцать пять) календарных дней с момента завершения 2 этапа |
| 3.2. | Доработка системы | * Отчет о проведении пусконаладочных работ; * Программное обеспечение Системы, включая исходные коды на машинном носителе (CD/DVD);   Актуализированная рабочая документация в составе:   * Руководство пользователя; * Руководство администратора; * Программа и методика предварительных испытаний; * Программа и методика опытной эксплуатации; * Разработанный дизайн экранов и пользовательских сценариев Мобильного приложения для функционала, реализованного в рамках этапа 3. | 136 (сто тридцать шесть) календарных дней с момента окончания техно-рабочего проектирования 3 этапа |
| 3.3. | Предварительные испытания | * Протокол предварительных испытаний; * Протокол устранения недостатков, выявленных в ходе предварительных испытаний (при необходимости); * Акт о приемке Системы в опытную эксплуатацию. | 15 (пятнадцать) календарных дней с момента окончания доработки системы 3 этапа |
| 3.4. | Опытная эксплуатация | * Журнал опытной эксплуатации; * Акт о готовности Системы к приемочным испытаниям; * Отчет о проведении опытной эксплуатации с приложением журнала опытной эксплуатации; * Акт о завершении опытной эксплуатации; * Программа и методика приемочных испытаний (с приложением проекта протокола приемочных испытаний и проекта акта о готовности Системы к эксплуатации); | 15 (пятнадцать) календарных дней с момента окончания предварительных испытаний 3 этапа |
| 3.5. | Приемочные испытания | * Протокол приемочных испытаний; * Протокол разногласий приемочных испытаний (при необходимости) * Акт о готовности Системы к эксплуатации; * Программное обеспечение модернизированной Системы, включая исходные коды, на машинном носителе (CD/DVD); * Протокол развертывания; * Акт передачи исходного кода и дистрибутива; * Акт о готовности Системы к эксплуатации; * Накладная на передачу исполнительной документации. | 5 (пять) календарных дней с даты завершения опытной эксплуатации 3 этапа |
| **4.** | **Развитие геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (4-этап)** | * **Акт сдачи-приемки выполненных работ по этапу 4** * **Акт приема-передачи исключительных прав на программное обеспечение Системы, разработанное в результате этапа 4** | **169 (сто шестьдесят девять) календарных дней с даты завершения 3 этапа** | **Заполняется на стадии заключения Контракта** |
| 4.1. | Техно-рабочее проектирование | * Технический проект в составе: * Ведомость документов; * Пояснительная записка Технического проекта; * Частное техническое задание (4 этап); * Описание программного обеспечения; * Описание информационного обеспечения Системы; | 20 (двадцать) календарных дней с момента завершения 3 этапа |
| 4.2. | Доработка системы | * Отчет о проведении пусконаладочных работ; * Программное обеспечение Системы, включая исходные коды на машинном носителе (CD/DVD);   Актуализированная рабочая документация в составе:   * Руководство пользователя; * Руководство администратора; * Программа и методика предварительных испытаний; * Программа и методика опытной эксплуатации; * Разработанный дизайн экранов и пользовательских сценариев Мобильного приложения для функционала, реализованного в рамках этапа 4. | 114 (сто четырнадцать) календарных дней с момента окончания техно-рабочего проектирования 4 этапа |
| 4.3. | Предварительные испытания | * Протокол предварительных испытаний; * Протокол устранения недостатков, выявленных в ходе предварительных испытаний (при необходимости); * Акт о приемке Системы в опытную эксплуатацию. | 15 (пятнадцать) календарных дней с момента окончания доработки системы 4 этапа |
| 4.4. | Опытная эксплуатация | * Журнал опытной эксплуатации; * Акт о готовности Системы к приемочным испытаниям; * Отчет о проведении опытной эксплуатации с приложением журнала опытной эксплуатации; * Акт о завершении опытной эксплуатации; * Программа и методика приемочных испытаний (с приложением проекта протокола приемочных испытаний и проекта акта о готовности Системы к эксплуатации); | 15 (пятнадцать) календарных дней с момента окончания предварительных испытаний 4 этапа |
| 4.5. | Приемочные испытания | * Протокол приемочных испытаний; * Протокол разногласий приемочных испытаний (при необходимости) * Акт о готовности Системы к эксплуатации; * Программное обеспечение модернизированной Системы, включая исходные коды, на машинном носителе (CD/DVD); * Протокол развертывания; * Акт передачи исходного кода и дистрибутива; * Акт о готовности Системы к эксплуатации; * Накладная на передачу исполнительной документации. | 5 (пять) календарных дней с даты завершения опытной эксплуатации 4 этапа |
| **5.** | **Развитие геосоциальной системы для пассажиров комплекса общественного транспорта города Москвы (5-этап)** | * **Акт сдачи-приемки выполненных работ по этапу 5** * **Акт приема-передачи исключительных прав на программное обеспечение Системы, разработанное в результате этапа 5** * **Акт сдачи-приемки выполненных работ по Государственному контракту** | **241 (двести сорок один) календарный день с даты завершения 4 этапа** | **Заполняется на стадии заключения Контракта** |
| 5.1. | Техно-рабочее проектирование | * Технический проект в составе: * Ведомость документов; * Пояснительная записка Технического проекта; * Частное техническое задание (5 этап); * Описание программного обеспечения; * Описание информационного обеспечения Системы. * Модель угроз и нарушителя информационной безопасности Системы; * Акт классификации Системы; * Акт категорирования Системы; * Отчет о выполнении анализа влияния функций разрабатываемых и развиваемых подсистем на классы защищенности информации, обрабатываемой в Системе; * Проектные решения по системе защиты информации, содержащейся в Системе. | 25 (двадцать пять) календарных дней с момента завершения 4 этапа |
| 5.2. | Доработка системы | * Отчет о проведении пусконаладочных работ; * Программное обеспечение Системы, включая исходные коды на машинном носителе (CD/DVD);   Актуализированная рабочая документация в составе:   * Руководство пользователя; * Руководство администратора; * Программа и методика предварительных испытаний; * Программа и методика опытной эксплуатации; * Разработанный дизайн экранов и пользовательских сценариев Мобильного приложения для функционала, реализованного в рамках этапа 5. | 181 (сто восемьдесят один) календарный день с момента окончания техно-рабочего проектирования 5 этапа |
| 5.3. | Предварительные испытания | * Протокол предварительных испытаний; * Протокол устранения недостатков, выявленных в ходе предварительных испытаний (при необходимости); * Акт о приемке Системы в опытную эксплуатацию. | 15 (пятнадцать) календарных дней с момента окончания доработки системы 5 этапа |
| 5.4. | Опытная эксплуатация | * Журнал опытной эксплуатации; * Акт о готовности Системы к приемочным испытаниям; * Отчет о проведении опытной эксплуатации с приложением журнала опытной эксплуатации; * Акт о завершении опытной эксплуатации; * Программа и методика приемочных испытаний (с приложением проекта протокола приемочных испытаний и проекта акта о готовности Системы к эксплуатации); | 15 (пятнадцать) календарных дней с момента окончания предварительных испытаний 5 этапа |
| Ф | Приемочные испытания | * Протокол приемочных испытаний; * Протокол разногласий приемочных испытаний (при необходимости) * Акт о готовности Системы к эксплуатации * Программное обеспечение модернизированной Системы, включая исходные коды, на машинном носителе (CD/DVD); * Паспорт информационной системы; * Протокол развертывания; * Акт передачи исходного кода и дистрибутива; * Акт о готовности Системы к эксплуатации; * Накладная на передачу исполнительной документации. | 5 (пять) календарных дней с даты завершения опытной эксплуатации 5 этапа |