**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Приглашенный преподаватель базовой кафедры ПАО Сбербанк  факультета компьютерных наук  А.И. Калинин  « » 2025 г. | УТВЕРЖДАЮ  Академический руководитель образовательной программы  «Программная инженерия» старший преподаватель департамента программной инженерии  Н.А. Павлочев  « » 2025 г. |

## Приложение для создания пешеходных маршрутов

|  |  |
| --- | --- |
| *Подп. и дата* |  |
| *Инв. № дубл.* |  |
| *Взам. инв. №* |  |
| *Подп. и дата* |  |
| *Инв. № подл* |  |

**Руководство программиста  
ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.05.06-01 33 01-1-ЛУ**

Исполнитель:

студент группы БПИ224

/ А. Евсюков /

« » 2025 г.

УТВЕРЖДЕНО RU.17701729.05.06-01 33 01-1-ЛУ

## Приложение для создания пешеходных маршрутов

**Руководство программиста**

**RU.17701729.05.06-01 33 01-1**

|  |  |
| --- | --- |
| *Подп. и дата* |  |
| *Инв. № дубл.* |  |
| *Взам. инв. №* |  |
| *Подп. и дата* |  |
| *Инв. № подл* |  |

**Листов 15**

# СОДЕРЖАНИЕ

[1. Назначение и условия применения программы 4](#_Toc198005539)

[1.1. Функциональное назначение 4](#_Toc198005540)

[1.2. Эксплуатационное назначение 4](#_Toc198005541)

[1.3. Состав выполняемых функций 4](#_Toc198005542)

[1.4. Технические средства 5](#_Toc198005543)

[1.5. Программные средства 5](#_Toc198005544)

[2. Характеристики программы 7](#_Toc198005545)

[2.1. Режим работы 7](#_Toc198005546)

[2.2. Временные характеристики 7](#_Toc198005547)

[2.3. Средства контроля правильности выполнения 7](#_Toc198005548)

[2.4. Средства самовосстановления 7](#_Toc198005549)

[3. Обращение к программе 8](#_Toc198005550)

[3.1. Процедуры вызова программы 8](#_Toc198005551)

[4. Входные и выходные данные 12](#_Toc198005552)

[4.1. Организация входных данных 12](#_Toc198005553)

[4.1. Организации выходных данных 12](#_Toc198005554)

[5. Сообщения……………………………………………………………………………………………13](#_Toc198005555)

[5.1. Коды ответов сервера 13](#_Toc198005556)

[5.1. Возвращаемые объекты (DTO) 13](#_Toc198005557)

[ПРИЛОЖЕНИЕ 1](#_Toc198005558) [ТЕРМИНОЛОГИЯ 15](#_Toc198005559)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 17](#_Toc198005560)

# Назначение и условия применения программы

## Функциональное назначение

Приложение для создания и обмена пешеходными маршрутами предназначено для помощи пользователям в планировании и организации прогулок. Основные функции приложения включают:

* Создание маршрутов: Пользователи могут создавать маршруты, добавлять в них важные точки и сохранять их в черновик для дальнейшего редактирования или публикации.
* Поиск маршрутов: Реализована система поиска с использованием фильтров и сортировки, что позволяет находить маршруты, соответствующие заданным критериям и предпочтениям.
* Прохождение маршрутов: Пользователи могут ставить маршруты на паузу, чтобы продолжить прогулку позже, не теряя прогресса.
* Сохранение маршрутов: Возможность добавлять понравившиеся маршруты в избранное для быстрого доступа в будущем.

Приложение ориентировано исключительно на пешие маршруты, предоставляя инструмент для планирования прогулок и исследования новых мест.

## Эксплуатационное назначение

Приложение для создания пешеходных маршрутов предназначено для пользователей, которые интересуются прогулками и исследованием новых мест. Оно предоставляет инструменты для планирования маршрутов, сохранения их для последующего использования, а также поиска и обмена маршрутами с другими пользователями.

Приложение ориентировано на улучшение качества прогулок, позволяя пользователям находить интересные маршруты, возвращаться к ранее сохранённым вариантам и делиться своими находками.

## Состав выполняемых функций

1. Основные функции:
   * 1. Сервис взаимодействия с базой данных:

* Хранение информации о пользователях, маршрутах и связанных с ними данных.
* Предоставление хранимых данных по средством создания RESTful API
  + 1. Сервис авторизации и аутентификации:
* Механизмы аутентификации и авторизации пользователей.
* Создание и валидация JWT токенов пользователей.
  + 1. Сервис управления взаимодействия фронтенда и бэкенда:
* Обеспечение централизованной точки входа в приложение.
* Валидация запросов к приложению.

1. Основные хранимые данные:

* Пользователи.
* Маршруты.
* Категории.
* Избранные маршруты.
* Пройденные маршруты.
* Черновики маршрутов.
* Отзывы о маршрутах.

## Технические средства

Для работы программы необходим следующий состав технических средств:

1. Доступ к сети интернет.
2. Серверная инфраструктура (или персональный компьютер), которая будет отвечать следующим минимальным требованиям:

* Процессор: 2 ядра
* Оперативная память: 4 Гб
* Публичный IP-адрес (при тестировании на серверной инфраструктуре)
* Дисковое хранилище: 20 Гб

## Программные средства

Во время испытаний должны быть использованы следующие программные средства:

1. Установленная Java версии 21 или выше;
2. Установленная система автоматической сборки Gradle 8.11.1 или выше
3. Docker 24.0.0 или выше.

# Характеристики программы

## Режим работы

Программа работает в круглосуточном режиме при условии наличия запущенных сервисов и подключённой базы данных. Предусмотрена работа в распределенной среде с использованием микросервисной архитектуры.

## Временные характеристики

Среднее время отклика для большинства эндпоинтов не превышает 500 мс при штатной нагрузке. Время обработки зависит от объема передаваемых данных и текущей нагрузки на систему.

## Средства контроля правильности выполнения

Для мониторинга и диагностики работы системы используются Prometheus и Grafana. Логирование осуществляется с помощью встроенных механизмов Spring Boot с поддержкой вывода в консоль и в файловую систему.

## Средства самовосстановления

При отказе одного из микросервисов предусмотрена возможность его автоматического перезапуска через Docker и систему оркестрации. Ошибки сохраняются в логах и могут быть проанализированы для исправления ошибок.

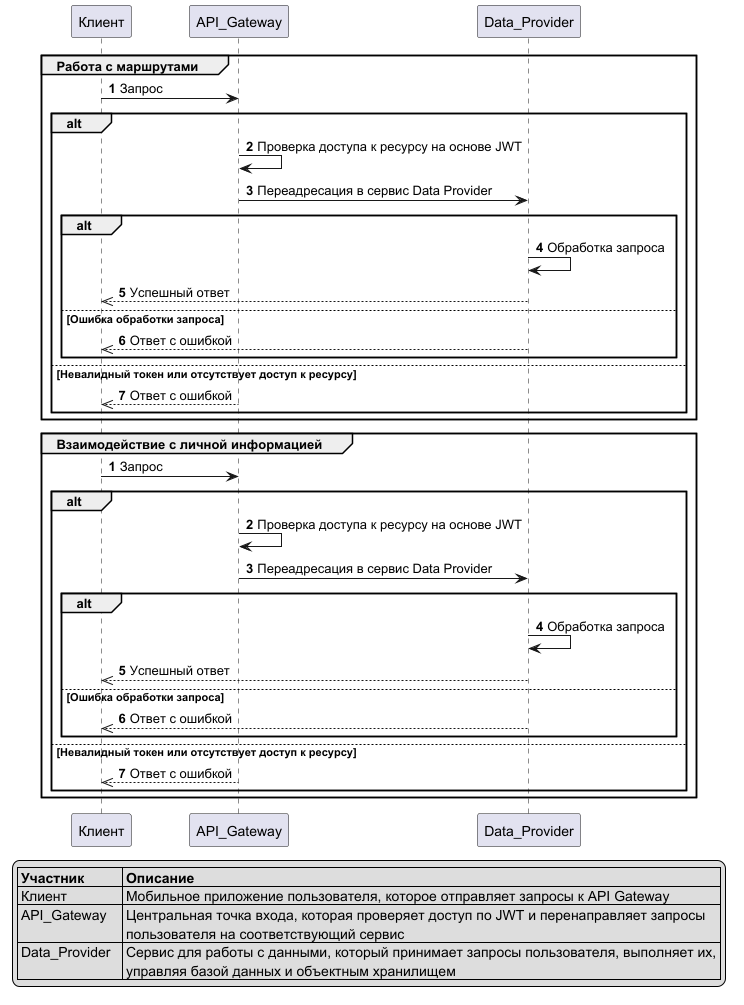
# Обращение к программе

## Процедуры вызова программы

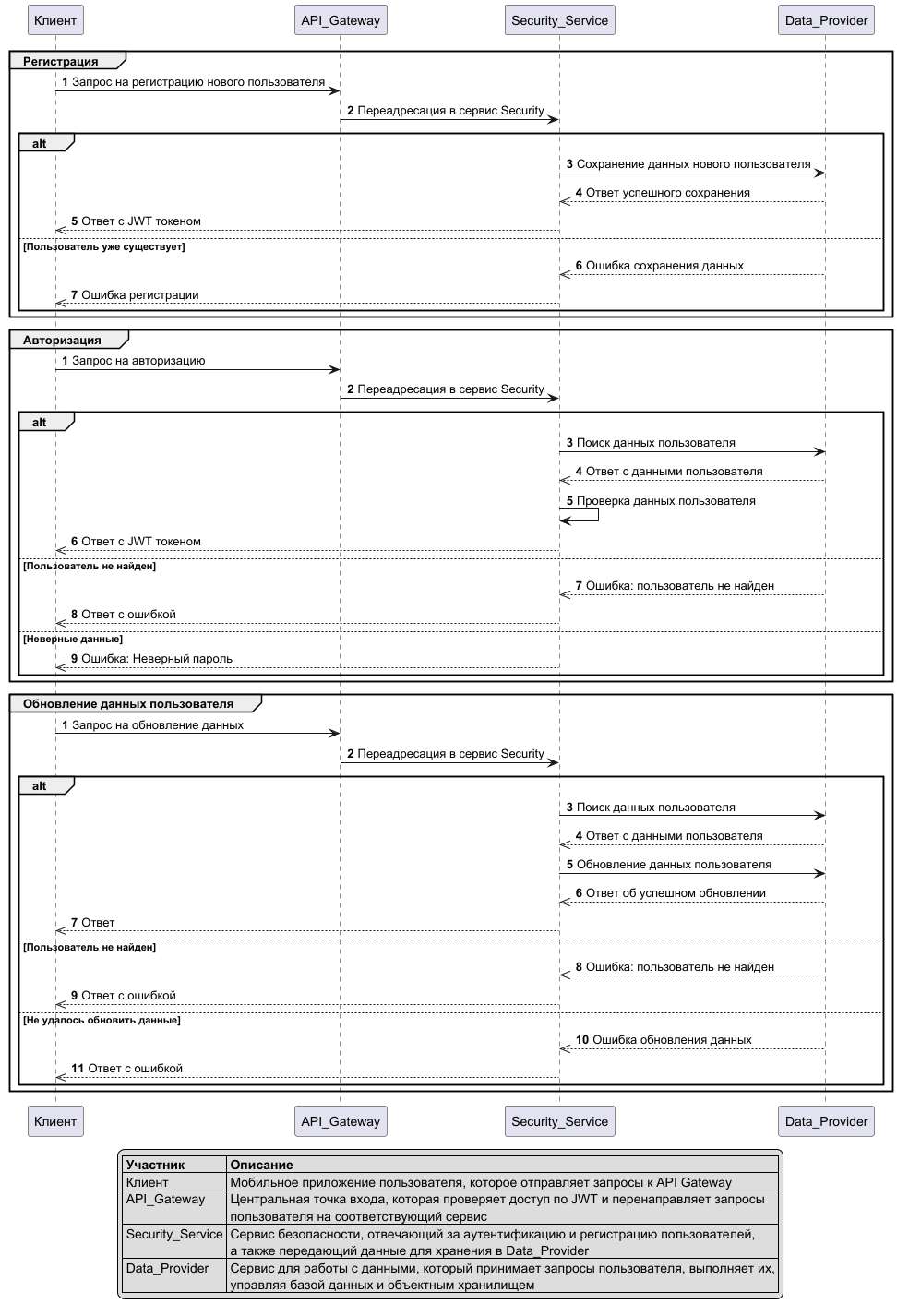
Для обмена данными между клиентом и сервером в системе используется REST API, который является широко распространённым решением в разработке серверных приложений благодаря своей простоте, гибкости и соответствию современным стандартам. REST обеспечивает удобный способ взаимодействия, позволяя передавать данные, такие как JSON, с помощью стандартных HTTP-запросов и ответов.

В архитектуре приложения используется API Gateway, через который проходят все внешние запросы. Он выполняет функции маршрутизации, предварительной валидации и, при необходимости, авторизации, после чего перенаправляет запросы в соответствующие внутренние сервисы.

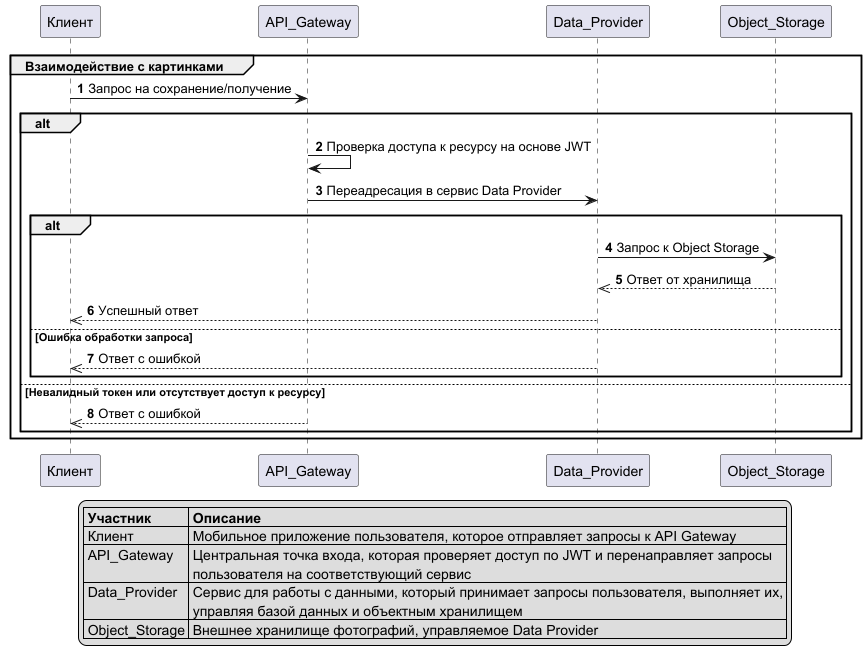
Для наглядной демонстрации логики обработки пользовательских запросов и последовательности взаимодействия компонентов ниже приведены диаграммы последовательностей для некоторых из них. Остальные запросы обрабатываются аналогичным образом.



*Рисунок 1 – Диаграммы последовательности для взаимодействия с данными о маршрутах*



*Рисунок 2 – Диаграммы последовательности для взаимодействия с данными пользователя*



*Рисунок 3 – Диаграммы последовательности для взаимодействия с объектным хранилищем*

# Входные и выходные данные

## Организация входных данных

Обработка POST запросов от фронтенд-части приложения по добавлению новых пользователей, маршрутов, оценок и отзывов.

## Организации выходных данных

Обработка GET запросов при выводе информации отображающей основной функционал из пункта 4.1.1 технического задания.

# Сообщения

## Коды ответов сервера

Программа может возвращать следующие HTTP коды, которые должен обрабатывать мобильный клиент:

* 200 — запрос успешно выполнен;
* 201 — объект успешно создан;
* 400 — некорректный запрос (ошибка со стороны клиента);
* 401 — пользователь не авторизован;
* 404 — запрашиваемый ресурс не найден;
* 500 — внутренняя ошибка сервера.

## Возвращаемые объекты (DTO)

Программа возвращает следующие объект, которые должен обрабатывать мобильный клиент:

* RouteCartDto

|  |
| --- |
| data class RouteCartDto(  var id: UUID,  var routeName: String?,  var duration: Double?,  var length: Double?,  var routePreview: String?,  var distanceToUser: Double?,  var categories: List<RouteDto.Categories>? ) |

* RoutePageDto

|  |
| --- |
| data class RoutePageDto(  var id: UUID,  var routeName: String?,  var description: String?,  var duration: Double?,  var length: Double?,  var startPoint: String?,  var endPoint: String?,  var routePreview: String?,  var isFavourite: Boolean?,  var routeCoordinate: List<RouteDto.RouteCoordinate>?,  var categories: List<RouteDto.Categories>? ) |

* RouteSessionDto

|  |
| --- |
| data class RouteSessionDto(  var id: UUID? = null,  var routeId: UUID,  var isFinished: Boolean?,  var startedAt: LocalDateTime?,  var endedAt: LocalDateTime?,  var userCheckpoint: List<UserCheckpoint> ) {   data class UserCheckpoint(  var coordinateId: UUID,  var createdAt: LocalDateTime?  ) } |

* ReviewCartDto

|  |
| --- |
| data class ReviewDto(  var curUserId: UUID? = null,  var reviews: List<ReviewInfoDto>, ) {   data class ReviewInfoDto(  var userId: UUID,  var userName: String,  var photoUrl: String?,  var reviewText: String?,  var rating: Int,  val createdAt: LocalDateTime  ) } |

# ПРИЛОЖЕНИЕ 1

# ТЕРМИНОЛОГИЯ

Таблица 1

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| База данных | Совокупность данных, организованных в соответствии с концептуальной структурой, описывающей характеристики этих данных и взаимоотношения между ними, которая поддерживает одну или более областей применения |
| Бэкенд | Часть веб-приложения, отвечающая за обработку данных и бизнес-логику, которая скрыта от пользователя. |
| Программное обеспечение | Совокупность программных и документальных средств для  создания и эксплуатации систем обработки данных средствами  вычислительной техники. |
| RESTful API | Архитектурный стиль взаимодействия компонентов приложения с использованием стандартных HTTP-запросов для управления ресурсами. |
| Микросервис | Независимые компоненты приложения, каждый из которых выполняет определённую бизнес-задачу и взаимодействует с другими через API. |
| JWT-токен | Компактный формат передачи данных, используемый для аутентификации и авторизации, обеспечивающий безопасность взаимодействия между клиентом и сервером. |
| Программное обеспечение | Совокупность программных и документальных средств для  создания и эксплуатации систем обработки данных средствами  вычислительной техники. |
| RESTful API | Архитектурный стиль взаимодействия компонентов приложения с использованием стандартных HTTP-запросов для управления ресурсами. |
| Дашборд (панель мониторинга) | Интерактивная визуальная панель, отображающая ключевые метрики системы или приложения в реальном времени, используемая для анализа состояния и выявления аномалий. |
| JSON (JavaScript Object Notation) | Лёгкий текстовый формат обмена данными, используемый для передачи информации между клиентом и сервером. |
| Load Balancer | Механизм распределения входящего сетевого трафика между несколькими серверами для обеспечения отказоустойчивости и производительности. |
| Producer (производитель) | Компонент, отправляющий сообщения в Kafka-топик. |
| Consumer (потребитель) | Компонент, подписывающийся на Kafka-топик и обрабатывающий поступающие сообщения. |
| Топик (Topic) | Категория или канал в Apache Kafka, куда публикуются и из которого читаются сообщения. |
| Микросервисная архитектура | Архитектурный стиль, при котором приложение состоит из набора мелких, независимых сервисов, взаимодействующих друг с другом через API. |
| R-дерево | Структура данных для индексирования многомерной информации, такой как географические координаты, используемая в PostGIS. |
| Эндпоинт | Конечная точка веб-сервиса, к которой клиентское приложение обращается для выполнения определённых операций или получения данных. |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в  докум.) | №  документа | Входящий  № сопроводит ельного  докум. и дата | Подп. | Дата |
| Изм. | Изменен ных | Заменен ных | Новых | Аннули рованх |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |