Промежуточный отчет по программному проекту

1. Основные планы и этапы проекта

1.1 Краткое описание проекта:

Приложение для создания и обмена пешеходными маршрутами предоставит пользователям возможности для планирования прогулок и исследования новых мест.

Основная функция приложения — создание персонализированных маршрутов, где пользователи могут добавлять важные точки и сохранять маршрут в черновик для последующего редактирования. Кроме того, в приложении будет реализован поиск маршрутов с использованием фильтров и сортировки, что позволит находить маршруты, соответствующие предпочтениям пользователя. При прохождении маршрутов будет предусмотрена возможность ставить их на паузу и возвращаться к прогулке позже. Дополнительно будет реализована функция сохранения маршрутов в избранное, чтобы пользователи могли быстро вернуться к понравившимся вариантам.

В отличие от стандартных картографических сервисов, предлагающих маршруты для транспорта или спортивных приложений, ориентированных на поиск маршрутов для бега, наше приложение фокусируется именно на пеших маршрутах, позволяя пользователям создавать, сохранять и делиться маршрутами, что делает продукт инструментом для планирования прогулок.

Название проекта:

Приложение для создания пешеходных маршрутов

Цель проекта:

Разработать бэкенд-часть приложения для создания пешеходных маршрутов

Краткое описание задач:

- Выбор инструментов для написания бэкенд-части
- Спроектировать предварительную структуру бэкенд-части
- Спроектировать архитектуру реляционной базы данных
- Разработать микросервис взаимодействия с базой данных
- Разработать микросервис авторизации и аутентификации пользователей
- Разработать микросервис для управления запросами и взаимодействия с фронтенд-частью приложения
- Поднять и настроить S3 хранилище для хранения картинок
- Настроить CI/CD для проекта и развернуть бэкенд-часть на сервере
- Провести интеграцию с фронтенд-частью
- Доработка, исправление ошибок, тестирование

1.2 Планы и этапы выполнения проекта:

Этап проекта	**Описание работ**	**Ожидаемые результаты**	**Сроки выполнения**
Выбор инструментов и архитектуры для разработки бэкенд-части приложения	Выбор инструментов для написания бэкенд-части	Определен основной стек технологий	15.10.2024 – 10.11.2024
	Спроектировать предварительную структуру бэкенд-части	Определена структуру бэкенд-части приложения	10.11.2024 – 30.11.2024
Разработка проекта	Спроектировать архитектуру реляционной базы данных	Определена архитектура базы данных	20.11.2024 – 30.11.2024
	Разработать микросервис взаимодействия с базой данных	Разработан микросервис взаимодействия с базой данных	01.12.2024 – 20.12.2024
	Разработать микросервис авторизации и аутентификации пользователей	Разработан микросервис авторизации и аутентификации пользователей	21.12.2024 – 10.01.2025
	Разработать микросервис для управления запросами и взаимодействия с фронтенд-частью приложения	Разработан микросервис управления запросами и взаимодействия с фронтенд-частью приложения	11.01.2025 – 31.01.2025
Настройка сервера	Поднять и настроить S3 хранилище для хранения картинок	Настроено хранилище для хранения картинок	31.01.2025 – 10.02.2025
	Настроить CI/CD для проекта и развернуть бэкенд-часть на сервере	Настроен сервер с развернутой на нем бэкенд-частью	31.01.2025 – 10.02.2025

Тестирование, исправление ошибок	Провести интеграцию с фронтенд-частью	Проверена корректность взаимодействия с фронтенд-частью	10.02.2025 – 15.02.2025
	Доработка, исправление ошибок, тестирование	Обеспечена стабильная работа, исправлены ошибки и подготовлена релизная версия	10.02.2025 – 02.03.2025

2. Используемый технологический стек и его обоснование

2.1 Перечень используемых технологий:

Технология/Инструмент	Описание	Причины выбора
Kotlin	Современный язык программирования для JVM, сочетающий лаконичность и безопасность.	Удобен для разработки читаемого и безопасного кода, с полной совместимостью с Java.
Spring Boot	Фреймворк для упрощенной разработки приложений с минимальной конфигурацией.	Один из самых популярных фреймворков для разработки бэкенда, подходит для быстрой разработки микросервисной архитектуры.
Liquibase	Инструмент для управления миграциями базы данных.	Обеспечивает версионирование и автоматизацию изменений в схемах базы данных, помогает контролировать изменения БД.
PostgreSQL	Реляционная СУБД с поддержкой JSONB и расширенными функциями.	Обеспечивает надежность, масштабируемость и удобную работу с данными.
PostGIS	Расширение PostgreSQL для работы с геопространственными данными.	Поддерживает мощный анализ геолокационных данных и их визуализацию.

IntelliJ IDEA	IDE для разработки приложений на Java и Kotlin.	Популярнейшая среда разработки на Java и Kotlin, предоставляет мощные инструменты для написания, отладки и анализа кода.
DataGrip	Инструмент для работы с базами данных и SQL-запросами.	Удобен для анализа, отладки и написания сложных SQL-запросов.
Docker	Платформа для создания, развертывания и запуска контейнеров.	Упрощает управление зависимостями и обеспечивает согласованность среды, помогает в процессе локальной разработки.
Git, GitHub	Система контроля версий и платформа для совместной разработки.	Позволяют эффективно управлять изменениями кода и организовать командную работу.

2.2 Обоснование выбранного технологического стека:

Выбранный технологический стек является одним из наиболее популярных и актуальных решений. Он обеспечивает надежность, масштабируемость и удобство разработки благодаря использованию современных инструментов. Spring Boot упрощает создание микросервисов за счет минимальной конфигурации и гибкости в интеграции с другими технологиями. Интеграция Docker и GitHub значительно упрощает развертывание и совместную работу, а Liquibase и PostGIS предоставляют расширенные возможности для управления миграциями и обработки геоданных.

3. Критерии оценивания проекта

Критерий	Описание
Использование инструмента автоматической документации API	Будут использованы/Не будут использованы
Работа с базой данных	Будут использованы/Не будут использованы
Управление доступом	Будут использованы/Не будут использованы

Функциональность - Процент выполнения функциональных требований	Выполненные требования в процентах от общего количества
Функциональность - Количество реализованных функций	Абсолютное количество функций, которые работают правильно
Качество кода - Количество обнаруженных ошибок (bugs/KLOC)	Количество ошибок на 1000 строк кода
Совместимость и кроссплатформенность - Количество интеграций	Число успешных интеграций с другими системами или сервисами
Соблюдение сроков и плана - Процент выполнения работы в срок (%)	Процент задач, выполненных в срок
Соблюдение сроков и плана - Количество дней отклонения от плана	Общее число дней отклонения от плана
Использование технологического стека - Процент использования функциональности стека (%)	Процент использования функциональности выбранного стека технологий
Оценка командной работы - Среднее время коммуникации (в часах)	Среднее время, потраченное на обсуждение задач и решение вопросов
Оценка командной работы - Количество завершенных задач на каждого участника	Общее число задач, выполненных каждым членом команды