Промежуточный отчет по программному проекту

1. Основные планы и этапы проекта

1.1 Краткое описание проекта:

«Сервис для стриминга и селфхостинга музыки. iOS-приложение» – это iOS-приложение, позволяющее пользователям управлять своей музыкальной библиотекой на личном сервере. Приложение обеспечивает удобный доступ к музыкальным композициям, их воспроизведение, создание плейлистов и поддержку оффлайн-режима.

Название проекта:

«Сервис для стриминга и селфхостинга музыки. iOS-приложение»

Цель проекта:

Разработать функциональное iOS-приложение для управления личной музыкальной библиотекой с поддержкой стриминга и оффлайн-доступа, обеспечивая полный контроль над музыкальными файлами. Это станет решением для тех, кто хочет сохранять доступ к своим музыкальным композициям и слушать их в любое время, независимо от ограничений стриминговых платформ.

Краткое описание задач:

1. Разработка технического задания, определение требований, определение архитектуры проекта.
2. Разработка iOS-приложения на языке программирования Swift.
3. Разработка программной документации: написание технического задания, пояснительной записки, программы и методики испытаний, текста программы, руководства оператора.
4. Испытания программы.
5. Защита проекта.

1.2 Планы и этапы выполнения проекта

Этап проекта	Описание работ	Ожидаемые результаты	Сроки выполнения
Обоснование необходимости разработки	Постановка задачи и сбор исходных теоретических материалов	Поставлены задачи и собрана литература и источники для теоретического решения задач	13.11.24
Научно-исследовательский этап разработки	Определение структуры входных и выходных данных; Предварительный выбор методов решения задач; Определение требований к техническим и программным средствам;	Определение требований для решения задачи	15.11.24 – 03.12.24

Этап проекта	Описание работ	Ожидаемые результаты	Сроки выполнения
	Обоснование возможности решения поставленной задачи		
Разработка и утверждение технического задания	Определение требований к программному продукту; Выбор языков программирования; Разработка и согласование технического задания с научным руководителем; Загрузка согласованного технического задания в SmartLMS	Готовое техническое задание для начала разработки приложения	15.11.24 – 04.12.24
Разработка программы	Предварительная разработка структуры программы; Программирование и отладка программы	Готовое приложение, реализующее все функции, определенные в техническом задании	05.12.24 – 27.03.25
Разработка программной документации	Разработка документов в соответствии с требованиями ГОСТ 19 ЕСПД (Единой системы программной документации)	Готовая программная документация	20.03.25 – 26.03.25
Испытания программы	Разработка, согласование и утверждение порядка в методики испытаний; Проведение испытаний программы в соответствии с утвержденными порядком и методикой; Корректировка программы и программной документации по результатам испытаний;	Тестирование програмы и исправление недочетов	27.03.25 – 05.04.25
Подготовка и передача программы	Подготовка программы и программной документации для презентации и защиты; Представление разработанного программного продукта научному руководителю и получение отзыва; Загрузка Пояснительной записки в систему Антиплагиат через ЛМС НИУ ВШЭ; Загрузка материалов курсового проекта в ЛМС, дисциплина «Курсовой проект, 2 курс, ПИ»;	Получение готового продукта, реализующее все функции, и зашита работы	05.04.25 – 30.04.25

Этап проекта	Описание работ	Ожидаемые результаты	Сроки выполнения
	Защита программного продукта комиссии		

2. Используемый технологический стек и его обоснование

2.1 Перечень используемых технологий

Технология/Инструмент	Описание	Причины выбора
Swift	Язык программирования для разработки приложений под операционную систему iOS	Актуальный язык для разработки под iOS
UIKit	Фреймфорк для создания интерфейса	Удобный и поддерживаемый фреймфорк для создания интерфейса под iOS
Combine	Фреймворк для обработки событий	Удобная и простая библиотека для работы с асинхронными событиями
HTTP REST API	Серверное взаимодействие	Для взаимодействия с сервером написанном на языке программирования Golang
Xcode	Среда разработки на языке программирования Swift	Бесплатная и удобная среда разработки. Нет лучших аналогов для языка программирования Swift
Git	Управление версиями	Удобство разработки в команде, сохранение версий разработки

2.2 Обоснование выбранного технологического стека

Использование языка программирования **Swift** и фреймворка **UIKit** для реализации iOS-приложения стриминга и селфхостинга музыки имеет следующие преимущества:

1. Простота и безопасность: Swift обеспечивает высокую читаемость и безопасность кода благодаря строгой типизации, что уменьшает количество ошибок на этапе компиляции.
2. Высокая производительность: Swift оптимизирован для работы с iOS, обеспечивая быстрые вычисления и отличное использование ресурсов устройства.
3. Документация: Обширные ресурсы и регулярные обновления делают разработку удобной и поддерживаемой.
4. Интеграция с iOS: UIKit обеспечивает доступ к базовым элементам интерфейса iOS (кнопкам, спискам, навигации), что упрощает создание стандартного пользовательского интерфейса.

5. Упрощенное асинхронное программирование: Combine позволяет управлять асинхронными событиями и обработкой данных, что упрощает разработку приложения.
6. Гибкость: Фреймворк UIKit позволяет создавать сложные пользовательские интерфейсы с кастомизацией.
7. Поддержка и стабильность: UIKit – устоявшийся инструмент, активно поддерживаемый Apple и совместимый со всеми актуальными версиями iOS.

Выбор серверного взаимодействия **HTTP REST API** связан по причинам:

1. Простота интеграции: REST API предоставляет стандартный способ взаимодействия между клиентом (приложением) и сервером, написанным на Golang.
2. Масштабируемость: REST API легко расширяется, добавляя новые конечные точки для поддержки дополнительных функций.
3. Кроссплатформенность: REST API может быть использован с любыми клиентскими приложениями, что делает серверную часть универсальной.

Xcode выбран в качестве среды разработки, потому что:

1. Предоставляет все необходимое для работы над проектом, включая редактор кода, дебаггер, симулятор устройств и инструменты для тестирования.
2. Это официальная IDE для разработки на Swift, что исключает проблемы совместимости.
3. Доступен бесплатно.

Git используется, так как:

1. Контроль версий: Git позволяет отслеживать изменения в проекте, что помогает разработчикам работать с различными версиями приложения.
2. Удобство командной работы: Git упрощает взаимодействие между членами команды, особенно при параллельной разработке функций.
3. Безопасность данных: Все изменения сохраняются, что позволяет откатиться к предыдущим версиям при необходимости.

3. Критерии оценивания проекта

Критерий	Описание
Использование адаптивного дизайна	Будет использовано/Не будет использовано
Функциональность – Процент выполнения функциональных требований	Выполненные требования в процентах от общего количества
Функциональность – Количество реализованных функций	Абсолютное количество функций, которые работают правильно
Использование технологического стека - Процент использования функциональности стека (%)	Процент использования функциональности выбранного стека технологий

Критерий	Описание
Документация и оформление – Полнота документации (%)	Процент от требуемого объема документации
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
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