

# 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	<p>Определение требований к программному продукту;</p> <p>Выбор языков программирования;</p> <p>Разработка и согласование технического задания с научным руководителем;</p> <p>Загрузка согласованного технического задания в SmartLMS</p>	Готовое техническое задание для начала разработки приложения	15.11.24 – 04.12.24
Разработка программы	<p>Предварительная разработка структуры программы;</p> <p>Программирование и отладка программы</p>	Готовое приложение, реализующее все функции, определенные в техническом задании	05.12.24 – 31.01.25
Разработка программной документации	Разработка документов в соответствии с требованиями ГОСТ 19 ЕСПД (Единой системы программной документации)	Готовая программная документация	05.12.24 – 31.01.25
Испытания программы	<p>Разработка, согласование и утверждение порядка в методики испытаний;</p> <p>Проведение испытаний программы в соответствии с утвержденными порядком и методикой;</p> <p>Корректировка программы и программной документации по результатам испытаний;</p>	Тестирование программы и исправление недочетов	01.02.25 – 28.02.25
Подготовка и передача программы	<p>Подготовка программы и программной документации для презентации и защиты;</p> <p>Представление разработанного программного продукта научному руководителю и получение отзыва;</p> <p>Загрузка Пояснительной записки в систему Антиплагиат через ЛМС НИУ ВШЭ;</p> <p>Загрузка материалов курсового проекта в ЛМС, дисциплина «Курсовой проект, 2 курс, ПИ»;</p>	Получение готового продукта, реализующее все функции, и защита работы	01.03.25 – 01.04.25

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	Фреймворк для создания интерфейса	Удобный и поддерживаемый фреймворк для создания интерфейса под iOS
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. Гибкость: Фреймворк UIKit позволяет создавать сложные пользовательские интерфейсы с кастомизацией.

6. 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
Документация и оформление – Полнота документации (%)	Процент от требуемого объема документации
Документация и оформление – Процент закомментированных строк (%)	Процент закомментированных строк относительно общего количества строк кода
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

Критерий	Описание
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