**Министерство науки и высшего образования Российской Федерации**
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**
**(Университет ИТМО)**

Факультет **Прикладной информатики**

Направление подготовки **09.03.03 Прикладная информатика**

Образовательная программа **Мобильные и сетевые технологии**

# К У Р С О В О Й  П Р О Е К Т

Тема: «Разработка мобильного приложения на iOS для учета трат»

Обучающийся: Кретов Иван Алексеевич, K3139

Санкт-Петербург 2024

**СОДЕРЖАНИЕ**

# ВВЕДЕНИЕ

Современный мир становится все более цифровым, и многие аспекты нашей жизни переносятся в мобильные устройства. В условиях современных реалий, когда учет личных финансов, включая доходы и расходы, становится важной частью финансовой грамотности и планирования личного бюджета, Finance Manager предоставляет удобные и эффективные инструменты для отслеживания расходов, анализа бюджета и планирования. Наше приложение поможет пользователям повысить финансовую грамотность и лучше контролировать свои финансы. Результаты проекта могут быть полезны широкому кругу пользователей. К примеру, это могут быть индивидуальные пользователи, то есть люди, которым важно знать куда уходят деньги и важно контролировать этот процесс, также фрилансеры или предприниматели, которым также необходимо отслеживать свои доходы и расходы, для ведения отчетности и финансового планирования.

Целью проекта было разработать мобильное приложение для учета и контроля личных финансов и управления доходами и расходами, которое будет удобно и полезно для всех возрастных категорий. Приложение должно быть простым в использовании, с интуитивно понятным интерфейсом. Также оно должно обеспечивать пользователям возможность эффективно отслеживать свои финансовые потоки, анализировать расходы, предоставляя аналитику в удобном формате.

Для достижения поставленной цели необходимо было решить следующие задачи:

1) анализ потребностей целевой аудитории,
2) проектирование интерфейса,
3) разработка функционала.

## 1 Описание сути проекта

Проект представляет собой разработку мобильного приложения для учета личных финансов на платформе iOS, направленного на упрощение процесса контроля за доходами и расходами пользователей. Задача заключалась в создании интуитивно понятного инструмента, который поможет пользователям управлять своими финансами без необходимости глубоких знаний в области бухгалтерии и финансов.

Ключевая идея проекта – предоставить простой и доступный инструмент для учета расходов, который позволит пользователю не только фиксировать каждую транзакцию, но и анализировать свои финансовые потоки через визуализацию данных. Приложение должно было объединить функциональность ведения бюджета с простотой и удобством, чтобы даже те пользователи, кто не имеет опыта в использовании финансовых приложений, могли легко интегрировать его в повседневную жизнь.

Приложение Finance Manager актуально в условиях современного финансового управления и предоставляет пользователям инструменты для точного отслеживания расходов, анализа бюджета и планирования, что способствует улучшению финансовой грамотности и эффективному контролю за личными финансами.

## 2 Процессы работы

### 2.1 Изучение языка Swift и фреймворка SwiftUI

Разработка мобильных приложений для платформы iOS требует владения языком программирования Swift и фреймворком SwiftUI, которые являются ключевыми инструментами для создания современных и удобных пользовательских интерфейсов. На первом этапе работы над проектом основное внимание было уделено изучению этих технологий. Мною было прочитано руководство от Apple по SwiftUI[1], которое предоставил наш магистрант. В данном руководстве содержались описания основных компонентов фреймворка, их использование и принципы построения интерфейсов.

Основными темами изучения стали SwiftUI Essentials, Drawing and Animation, а также App Design and Layout. Эти разделы подробно освещали ключевые аспекты разработки интерфейсов, включая создание пользовательских компонентов, работу с анимацией и базовые подходы к дизайну и верстке экранов. Особое внимание я уделил разделу, посвященному созданию интерфейсов с использованием декларативного стиля программирования, так как это является одной из особенностей SwiftUI.

Поскольку ранее я имел некоторый опыт программирования на языке Python, освоение Swift далось мне сравнительно легко. Эти языки имеют схожую логику и подход к работе с синтаксическими конструкциями, что позволило мне быстрее освоиться с новой средой разработки.

В дополнение к руководству от Apple я также прошел курс по программированию на Swift[2], который охватывал базовые темы. Этот курс включал изучение основных синтаксических конструкций языка Swift, работу с встроенными типами данных и коллекциями, а также принципы объектно-ориентированного программирования. На изучение всех этих материалов у меня ушла примерно одна неделя. Чтобы закрепить полученные

знания, я написал небольшое тестовое приложение, по статье на хабре[3], в которой наглядно показывалось как создать приложение с нуля. Создание этого приложения помогло мне лучше понять, как компоненты SwiftUI взаимодействуют между собой, а также научиться строить интерфейс.

### 2.2 Создание главного экрана приложения

Главный экран приложения является важнейшим элементом пользовательского интерфейса, так как именно с него начинается работа пользователя с приложением. На этом экране отображается основная информация о финансах: диаграмма распределения расходов по категориям, список всех транзакций (как доходов, так и расходов) и переключатель, позволяющий фильтровать отображаемые данные — показывать либо только доходы, либо только расходы. На рисунке 1 представлен дизайн главного экрана, который был разработан командой дизайнеров. Моя задача заключалась в том, чтобы сверстать этот дизайн с использованием фреймворка SwiftUI.

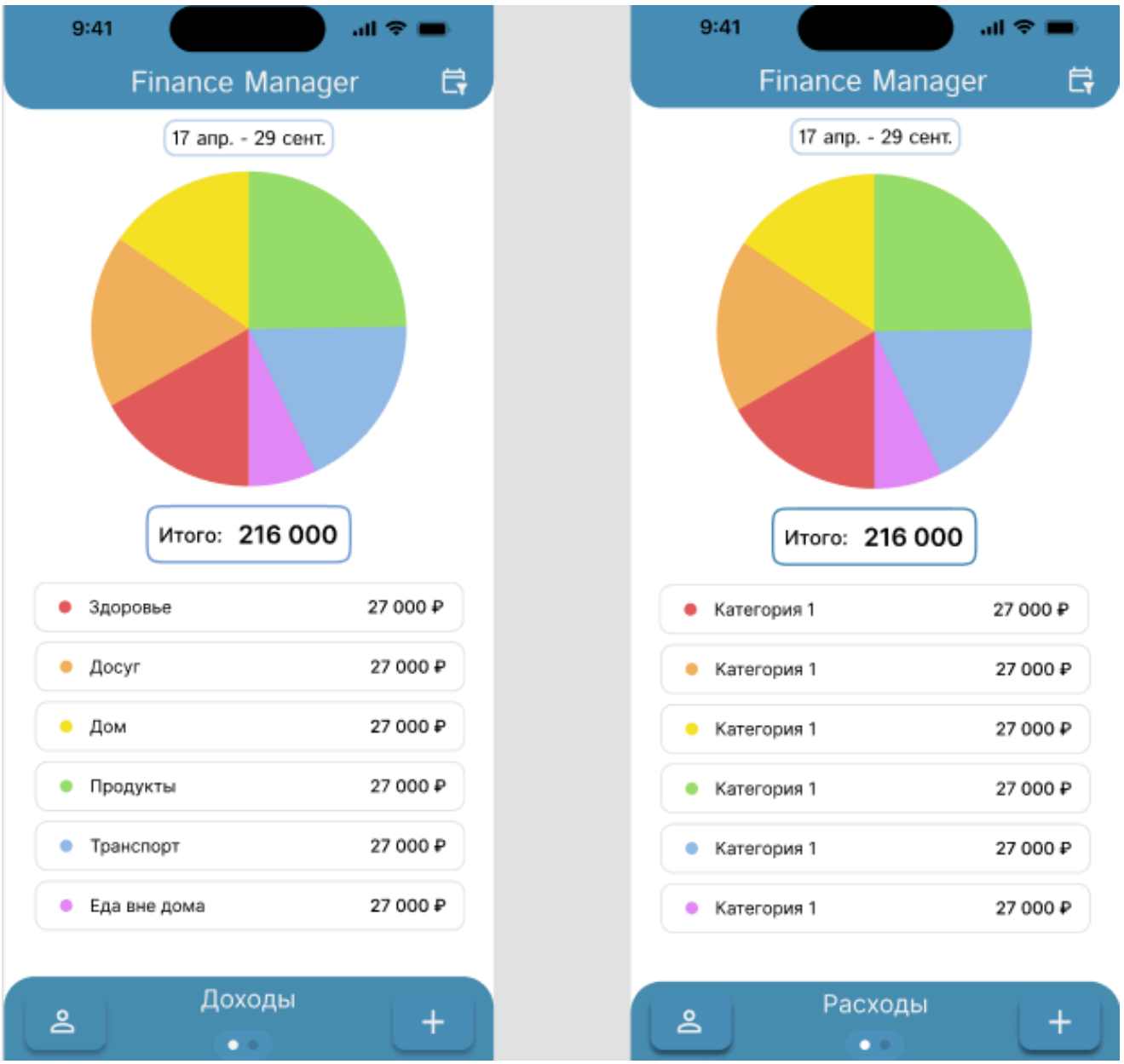


Рисунок 1 – Главный экран приложения с диаграммой расходов и списком транзакций

На реализацию главного экрана у меня ушло около недели. Основная работа включала верстку интерфейса с использованием компонентов SwiftUI. Я создал заголовки, списки и диаграмму, а также реализовал переключатель, позволяющий фильтровать данные по типу транзакции (доходы или расходы).

Серьезную проблему вызвала диаграмма, так как встроенные средства SwiftUI для ее создания поддерживаются только начиная с iOS 16, а наше приложение должно было работать на устройствах начиная с iOS 15. Чтобы

решить эту проблему, я выбрал альтернативный подход и реализовал диаграмму с использованием графических примитивов SwiftUI. Мною были прочитана статья на хабре, в которой подробно описывалась работа с графикой в SwiftUI[4]. Этот метод потребовал дополнительного времени и усилий, так как мне пришлось вручную рассчитывать размеры и позиции элементов диаграммы. Несмотря на трудности, этап был успешно завершен, и главный экран стал полностью функциональным.

### 2.3 Верстка экрана добавления транзакций

Следующим важным этапом разработки стало создание экрана для добавления новых транзакций. Данный экран является одним из ключевым элементом приложения, так как позволяет пользователю вносить данные о своих расходах и доходах. На рисунке 2 изображен дизайн экрана, подготовленный командой дизайнеров.

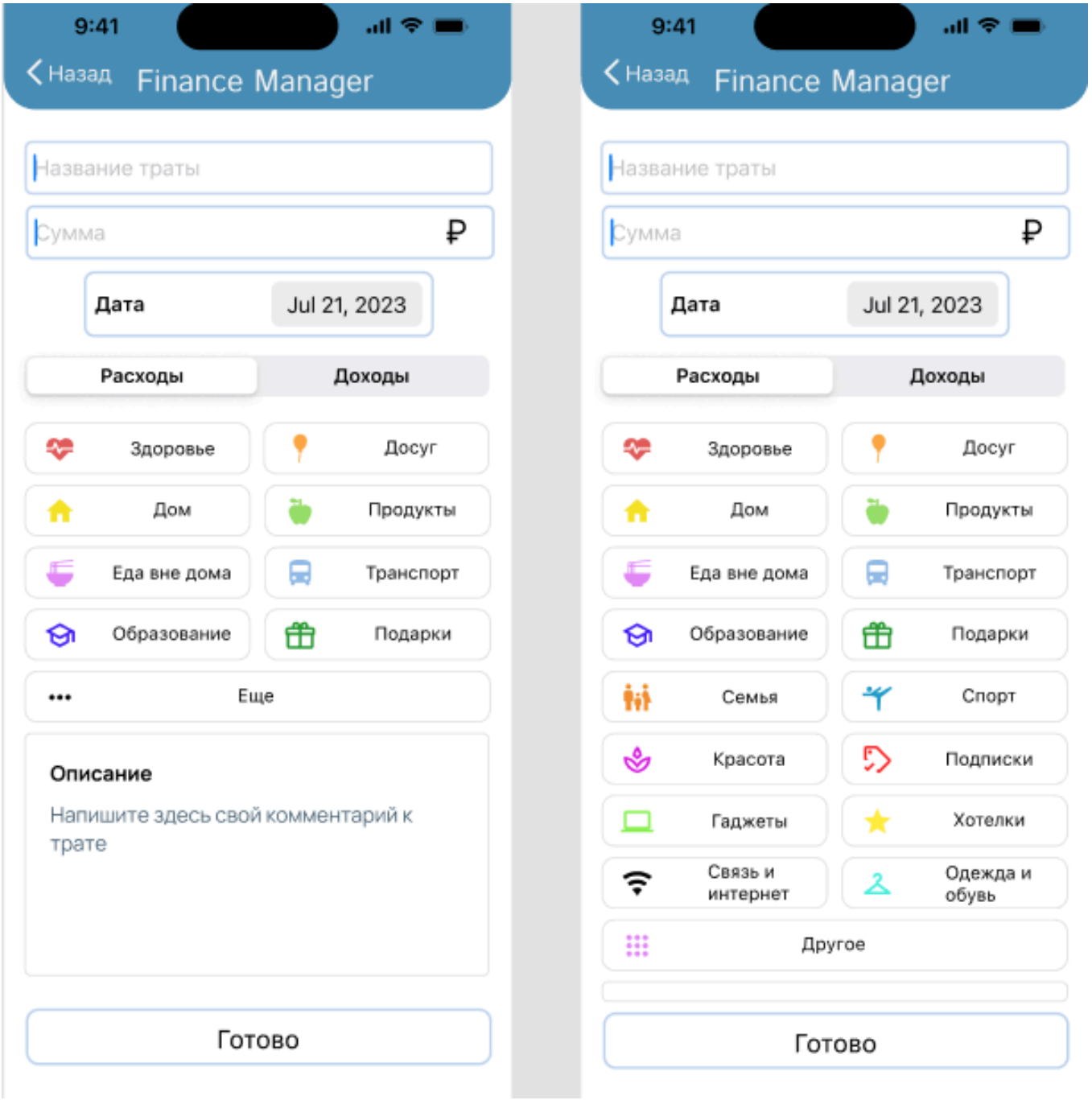


Рисунок 2 – Экран добавления новой транзакции

На экране добавления транзакции отображаются следующие элементы:

- название транзакции – поле ввода, где пользователь может указать краткое название,
- сумма транзакции – обязательное поле ввода, в которое пользователь вводит сумму операции.

- тип транзакции – слайдер для выбора типа транзакции (расходы или доходы)
- дата транзакции – компонент выбора даты с удобным интерфейсом. По умолчанию отображается текущая дата.
- категория транзакции – выпадающий список с предустановленными категориями,
- описание транзакции – необязательное текстовое поле для ввода дополнительной информации о транзакции.

Для верстки интерфейса я использовал компоненты SwiftUI, включая TextField для ввода суммы и названия, DatePicker для выбора даты, Picker для выбора типа транзакции, а TextEditor для ввода описания. Интерфейс был реализован в полном соответствии с предоставленным дизайном на рисунке 2.

Для временного хранения данных была создана простая архитектура, обеспечивающая сохранение информации о транзакциях в оперативной памяти до перезагрузки приложения. На данном этапе было важно обеспечить корректную работу экрана до интеграции с базой данных.

Одной из проблем, с которой я столкнулся при разработке экрана добавления транзакций, стало поведение экранной клавиатуры. Я пытался реализовать функцию, позволяющую скрывать клавиатуру при нажатии на пустую область экрана. Несмотря на то, что мне удалось частично реализовать этот функционал, он работал нестабильно: клавиатура скрывалась лишь в некоторых случаях, а в других оставалась на экране. Из-за этой непредсказуемости было решено отказаться от данной функции на данном этапе.

### 2.4 Подключение приложения к базе данных Firestore

Последним и самым сложным этапом разработки стало подключение приложения к базе данных Firestore. На этом этапе требовалось обеспечить хранение данных о транзакциях в удаленной базе данных, что позволило бы

пользователю сохранять свои данные и получать к ним доступ с любого устройства.

Для начала я изучил документацию Firebase[5], чтобы понять, как правильно работать с Firestore. В процессе изучения я научился добавлять новые записи в базу данных, редактировать и удалять существующие данные, а также загружать информацию из Firestore при запуске приложения.

Подключение Firestore потребовало значительной переработки архитектуры приложения, так как ранее использовалось только локальное хранилище данных. Мне пришлось переписать значительную часть кода, чтобы добавить поддержку удаленной базы данных и обеспечить корректную работу приложения.

Тестирование работы с Firestore проводилось вручную. Я проверял корректность сохранения данных в базе через консоль Firebase, а также регулярно устанавливал приложение на свой телефон и проверял его работу в реальных условиях. Все ключевые функции: добавление, редактирование и удаление транзакций – были успешно реализованы и протестированы. Благодаря поддержке руководителя проекта, который помогал нам с решением возникающих проблем, этап подключения базы данных был завершен в срок.

## 3 Проблемы и их решение

В процессе разработки приложения я столкнулся с несколькими значительными трудностями. Одной из первых проблем стала необходимость реализации диаграммы для отображения расходов. Изначально планировалось использовать встроенный инструмент SwiftUI, который отлично подходил для этой задачи, но он был доступен только для iOS 16. Так как наше приложение разрабатывалось для более старой версии – iOS 15, мне пришлось искать альтернативные методы. После анализа доступных решений было принято решение вручную отрисовывать диаграмму с использованием графических примитивов SwiftUI. Этот подход оказался сложным и трудоемким, но позволил достичь нужного результата и обеспечить совместимость с iOS 15. На рисунке 3 показана реализация рисования диаграммы.

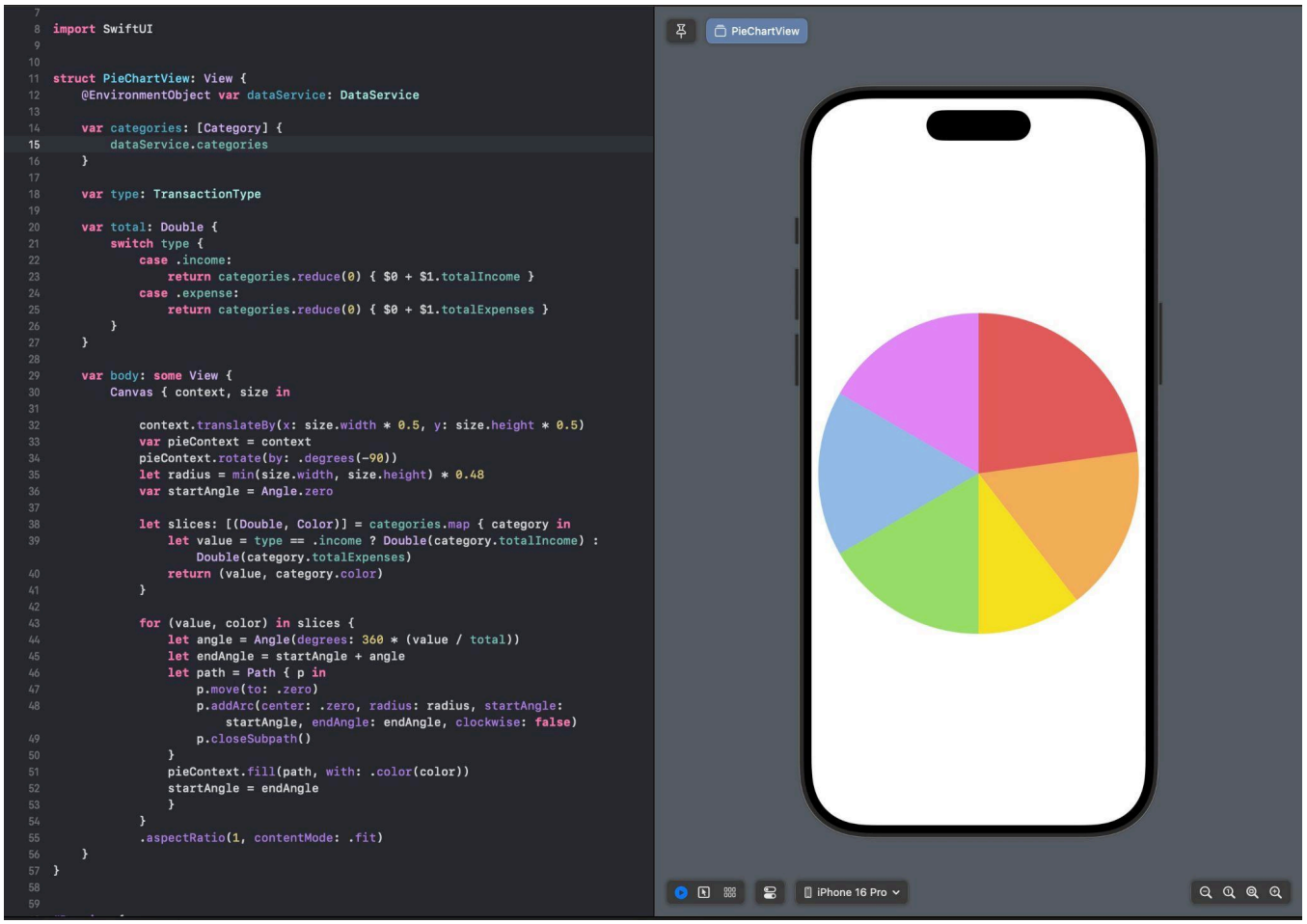

Рисунок 3 – Реализация диаграммы

Еще одной серьезной проблемой стало подключение приложения к удаленной базе данных Firestore. Для меня это был новый опыт, поэтому мне пришлось подробно изучать документацию Firebase и примеры работы с базой данных. После нескольких дней изучения мне удалось понять основные принципы работы с Firestore и интегрировать ее с приложением. Для полноценной работы приложения были реализованы функции добавления, редактирования и удаления транзакций, а также автоматическая загрузка данных при запуске.

Отдельного упоминания заслуживает проблема с клавиатурой на экране добавления транзакций. Во время тестирования я обнаружил, что клавиатура не всегда исчезает при нажатии на пустую область экрана, что создавало неудобства для пользователя. Я пробовали разные подходы к решению этой проблемы, но из-за нехватки времени было решено оставить текущую реализацию, так как эта проблема не критична и не мешает работе приложения. В целом, несмотря на все трудности, большинство возникающих проблем удалось решить благодаря обсуждениям с магистрантом и слаженной работе команды.

## 4 Самоанализ работы

Работа над проектом принесла мне много нового опыта и знаний. За время выполнения курсовой работы я освоил язык программирования Swift и научился создавать интерфейсы с использованием фреймворка SwiftUI. Несмотря на то что ранее я уже имел некоторый опыт программирования, работа над мобильным приложением для iOS оказалась для меня совершенно новым и интересным вызовом.

Самым сложным этапом для меня стало подключение к базе данных Firestore. На этом этапе пришлось глубоко погрузиться в изучение новой технологии и переработать большую часть существующего кода, чтобы приложение могло работать с удаленной базой данных. Это было непросто, но в результате я получил ценные знания, которые пригодятся мне в будущих проектах.

Также во время работы над проектом я научился лучше планировать свое время и более эффективно работать над сложными задачами. Некоторым функциям, таким как фильтрация транзакций по времени и поддержка оффлайн-режима, не удалось уделить достаточно времени, поэтому они остались нереализованными. Однако я считаю, что основная цель проекта была достигнута, а приложение полностью выполняет свои ключевые функции.

## 5 Работа в команде

Проект был командным, и мне посчастливилось работать в группе, состоящей из двух дизайнеров и трех разработчиков. Дизайнеры подготовили макеты интерфейса, которые мы должны были реализовать в приложении. Разработчики, включая меня, занимались созданием отдельных частей приложения: каждый из нас работал над определенными экранами и функциями. Задачи между нами распределял магистрант, который выполнял роль руководителя команды. Благодаря четкому распределению задач нам удалось избежать дублирования работы и эффективно справляться с возникающими трудностями.

Взаимодействие в команде происходило как очно, так и онлайн. Регулярные созвоны помогали нам поддерживать координацию и своевременно решать возникающие вопросы. На созвонах каждый участник отчитывался о проделанной работе, получал новые задачи и обсуждал проблемы, возникшие на этапе разработки. Такая организация работы позволила нам придерживаться установленного графика и завершить проект в срок.

Руководитель проекта оказывал значительную поддержку на всех этапах разработки. Он помогал нам разобраться с инструментами, рассказывал, как работать с Git через приложение Fork, и был всегда готов ответить на вопросы. Особенно ценной была его помощь на этапе подключения базы данных Firestore, где он дал полезные советы по оптимизации работы с базой. Благодаря его поддержке и готовности помочь мне удалось значительно улучшить свои навыки программирования и получить положительный опыт работы в команде. Я считаю, что взаимодействие с руководителем и коллегами по команде было организовано на высоком уровне, и это сыграло ключевую роль в успешном завершении проекта.

# ЗАКЛЮЧЕНИЕ

Я считаю, что проект выполнен успешно, поскольку приложение работает и выполняет свои основные функции – добавление, редактирование и удаление транзакций. Однако не все задачи удалось завершить из-за нехватки времени. В частности, не был реализован фильтр транзакций по времени, а также возможность работы приложения в оффлайн-режиме.

Мой вклад в проект заключался в изучении новых инструментов и технологий, создании части интерфейса приложения, разработке логики взаимодействия приложения с пользователем и подключении удаленной базы данных. Полученный мною опыт значительно улучшил мои навыки программирования и работы в команде, а также расширил мою область знаний.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Руководство от Apple по SwiftUI https://developer.apple.com/tutorials/swiftui/

2) Курс по языку программирования Swift

https://www.youtube.com/playlist?list=PLRJuPW6BGThue5qX6AkARnqc8o_80fbQA

3) Статья на хабре про создание приложения с нуля

https://habr.com/ru/articles/461645/

4) Статья на хабре про работу с графикой в SwiftUI

https://habr.com/ru/articles/805809/

5) Документация по FireStore

https://firebase.google.com/docs/firestore/quickstart?hl=ru

## ПРИЛОЖЕНИЕ

Техническое задание к проекту “Разработка мобильного приложения на iOS для учета трат”.

Цель: разработать мобильное приложение для учета и контроля личных финансов и управления доходами и расходами

Руководитель проекта: Сидаматов Жасур Илхомович

Термины и сокращения:

iOS - операционная система для мобильных устройств компании Apple.

SwiftUI – это декларативный фреймворк Apple для создания пользовательских интерфейсов на всех платформах Apple с использованием минимального кода и обновления UI в реальном времени.

Core Data – это фреймворк Apple для управления моделью данных в приложениях, обеспечивающий сохранение, выборку и управление объектами в удобной объектно-ориентированной форме.

GitHub/GitLab - это платформы для управления репозиториями исходного кода с использованием Git, предоставляющие инструменты для совместной разработки.

МП - мобильное приложение

Техническое задание (ТЗ) — это документ, в котором описываются требования и спецификации для выполнения проекта или разработки продукта.

Pull request (PR) — это предложение изменений в репозитории, которое отправляется на рассмотрение для слияния с основной веткой проекта после ревью и одобрения.

Firebase - это платформа от Google, предоставляющая набор инструментов и сервисов для разработки мобильных и веб-приложений, включая базы данных, аутентификацию, хостинг, аналитику и уведомления в реальном времени.

Верстка экрана — это процесс разработки и расположения элементов интерфейса (текстов, кнопок, изображений и других компонентов) на экране приложения или веб-страницы с учетом дизайна, функционала и удобства пользователя.

Flow пользователя в приложении (или пользовательский поток) — это последовательность шагов и действий, которые пользователь выполняет внутри приложения для достижения своей цели, например, регистрации, совершения покупки или выполнения задачи. Это включает в себя взаимодействие с интерфейсом, переходы между экранами и обработки действий, направленных на выполнение конкретной функции.

Технические требования

Таблица 1 - функциональные и нефункциональные требования

| Техническое требование | Язык разработки | СУБД | Потребители |
|---|---|---|---|
| Минимальная версия поддержки iOS 15 | Swift | | iOS разработчики |
| Технологии для разработки МП | Xcode, Swift, SwiftUI, URLSession, SPM. | CoreData, Firebase | iOS разработчики |
| Система контроля версий GitHub/GitLab | GitHub/GitLab | | iOS разработчики |

| | | | |
|---|---|---|---|
| Дизайн приложения должен соответствовать Apple HIG, обеспечивая интуитивный интерфейс, минимализм, адаптацию под разные экраны мобильного устройства iOS | Figma | | Дизайнеры |
| Возможность регистрации/авторизации | Firebase | | iOS разработчики |
| Реализация flow пользователя в макетах Figma | Figma | | |
| Поддержка оффлайн режима МП. В оффлайн режиме пользователь остается авторизованным на своей учетной записи и может добавлять новые транзакции, они сохраняются в локальную базу данных. Как только появляется интернет система должна обновить данные в Firebase. | Swift | | iOS разработчики |

| | | | |
|---|---|---|---|
| Выбрать архитектуру для МП с учетом дальнейшего расширения функциональности, учитывая что используется фреймворк SwiftUI. | Swift | | iOS разработчики |

Содержание работы

Этапы задач:

1) разработка технического задания. 1-10 ноября. Результат - Текстовый форма,

2) проектирование дизайна приложения. 10-30 ноября. Результат - Проект в Figma,

3) разработка МП. 20 ноября - 10 декабря. Результат - Проект на GitHub,

4) формирование отчета и презентации к защите. 10 декабря - 15 декабря. Результат - Отчет и презентация.

Задачи

Таблица 2 - Задачи

| Название задачи | Описание | Technical Specification | Этап |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Создать техническое задание | Создать техническое задание в соответсвии с требованиями представленные на лекция по предмету "Управление проектами по разработке мобильных и сетевых приложений".<br><br>Требования и шаблон - https://docs.google.com/document/d/1P38OYJhoECpPF4sMvkvrdekUI7h06dvl/edit.<br><br>Итог: готовое ТЗ заполненное на странице проекта. | Разработка мобильного приложения на iOS для учета трат | Разработка технического задания. 1-10 ноября. Результат - Текстовый формат. |
| Создать начальный файл с дизайном и дать доступ к нему | Создать исходный файл с дизайном и предоставить его команде разработки.<br><br>Итог: ссылка на проект в Figma в чате группы | Разработка мобильного приложения на iOS для учета трат | Проектирование дизайна приложения. 10-30 ноября. Результат - Проект в Figma |
| Подобрать цветовую гамму МП | По результатам анализа конкурентов и изучению цветовых гамм, определить основные цвета для МП.<br><br>Итог: цветовая гамма будущего МП | Разработка мобильного приложения на iOS | Разработка технического задания. 1-10 ноября. |

| | | | |
|---|---|---|---|
| | | для учета трат | Результат - Текстовый формат. |
| Спроектировать дизайн экрана "Авторизация" | Создать визуальный дизайн экрана для входа пользователя в систему. Экран должен включать поля для ввода логина и пароля, кнопки входа и регистрации нового пользователя. Дизайн должен быть интуитивно понятным, обеспечивать легкость восприятия и соответствовать общей стилистике приложения.<br><br>Итог: дизайн экрана в Figma | Разработка мобильного приложения на iOS для учета трат | Проектирование дизайна приложения. 10-30 ноября. Результат - Проект в Figma |
| Спроектировать экран "Главная" | На главном экране необходимо отобразить<br>• Траты в списочном виде<br>• Кнопка добавления новой транзакции<br>• Кнопка открытия профиля<br>• Диаграмма трат<br><br>Итог: дизайн экрана в Figma | Разработка мобильного приложения на iOS для учета трат | Проектирование дизайна приложения. 10-30 ноября. Результат - Проект в Figma |

| | | | |
|---|---|---|---|
| Спроектировать дизайн экрана "Добавление транзакции" | Создать удобный и понятный интерфейс для добавления новой финансовой транзакции. Экран должен включать следующие элементы:<br><br>1. Поля для ввода суммы транзакции и выбора категории (например, доход или расход).<br>2. Возможность указать дату и время транзакции.<br>3. Поле для ввода вида траты<br>4. Поле для добавления заметки или комментария.<br>5. Кнопку "Сохранить" для подтверждения добавления транзакции.<br><br>Дизайн должен быть минималистичным, с легким доступом ко всем функциям, и обеспечивать быстрое добавление информации | Разработка мобильного приложения на iOS для учета трат | Проектирование дизайна приложения. 10-30 ноября. Результат - Проект в Figma |
| Спроектировать дизайн экрана "Профиля" | Экран должен отображать основную информацию о пользователе и возможность выйти из аккаунта.<br><br>Итог: дизайн экрана в Figma | Разработка мобильного приложения на iOS для | Проектирование дизайна приложения. 10-30 ноября. Результа |

| | | | |
|---|---|---|---|
| | | учета трат | т - Проект в Figma |
| Подключить к проекту возможность работы с Firebase | Добавить зависимость через SPM в проект.<br>Добавить возможность авторизации и хранения данных в Firebase - создать отдельные сервисы, которые в дальнейшем будут использованы системой.<br><br>Итог: Итог: PR с описанием выполненной задачи | Разработка мобильного приложения на iOS для учета трат | Разработка МП. 20 ноября - 10 декабря. Результат - Проект на GitHub |
| Спроектировать архитектуру МБ | Подобрать архитектуру приложения под фреймворк SwifUI. | Разработка мобильного приложения на iOS для учета трат | Разработка МП. 20 ноября - 10 декабря. Результат - Проект на GitHub |

| | | | |
|---|---|---|---|
| Сверстать экран авторизации | Выполнить верстку экрана "Авторизации" и "Регистрации" по спроектированному макету Figma.<br><br>Макет: https://www.figma.com/design/wSfx7SNCRwGH70y4Oh9OdG/MoneyManager?node-id=0-1&node-type=canvas&t=J1zlDQ8N4hlThmp9-0<br><br>Итог: Итог: PR с описанием выполненной задачи | Разработка мобильного приложения на iOS для учета трат | Разработка МП. 20 ноября - 10 декабря. Результат - Проект на GitHub |
| Сверстать экран "Главная" | Сверстать экран "Главная" по макету Figma.<br><br>Макет - https://www.figma.com/design/wSfx7SNCRwGH70y4Oh9OdG/MoneyManager?node-id=0-1&node-type=canvas&t=J1zlDQ8N4hlThmp9-0<br><br>Итог: PR с описанием выполненной задачи | Разработка мобильного приложения на iOS для учета трат | Разработка МП. 20 ноября - 10 декабря. Результат - Проект на GitHub |
| Сверстать экран "Добавление транзакции" | Сверстать экран "Добавление транзакции" по Figma.<br><br>Макет - https://www.figma.com/design/wSfx7SNCRwG | Разработка мобильного приложения на iOS для | Разработка МП. 20 ноября - 10 декабря. Результат - |

| | | | |
|---|---|---|---|
| | H70y4Oh9OdG/MoneyManager?node-id=0-1&node-type=canvas&t=J1zlDQ8N4hlThmp9-0<br><br>Итог: PR с описанием | учета трат | Проект на GitHub |
| Сформировать отчет и презентацию для защиты проекта | Сформировать отчет и презентацию для защиты проекта.<br>Отчет должен соответствовать ГОСТ 7-32. 2017.<br><br>Итог: отчет и презентация | Разработка мобильного приложения на iOS для учета трат | Формирование отчета и презентации к защите. 10 декабря - 15 декабря. Ответственный - Атрашкевич Д.Д. Результат - Отчет и презентация |

Основные результаты работы

Мобильное приложение для учета доходов и расходов с функционалом анализа трат.

В результате разработки был создан прототип приложения, который включает:

- мобильное приложение: полностью функциональное приложение для учета доходов и расходов с возможностью анализа трат,

- проект на GitHub: исходный код приложения, доступный для просмотра и дальнейшего развития,

- отчет по разработке: подробный документ, описывающий этапы разработки приложения, использованные технологии и методы,

- презентация: подготовлена презентация для защиты проекта, включающая описание функционала и результатов работы.