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: «iOS приложение CryptoTracker»

Обучающийся: Востров Илья Анатольевич, K3139

Санкт-Петербург, 2024 г.

СОДЕРЖАНИЕ

Введение

В наши дни криптовалюты прочно вошли в структуру мировой экономики и инвестиционной деятельности. Многие пользователи ищут удобные средства для мониторинга свежих данных о рынке криптовалют. CryptoTracker предоставляет именно это, предлагая интуитивно понятный интерфейс и возможность получения развернутой информации о любой цифровой валюте.

Проект направлен на создание приложения, которое показывает список криптовалют с текущими ценами и динамикой за последние 24 часа, а также предоставляет доступ к детальной информации об интересующей монете.

Основные задачи проекта:

- построение базовой архитектуры приложения [1],
- реализация функционала получения данных через REST API [2],
- создание экрана списка криптовалют и экрана подробной информации о монете,
- интеграция сетевого слоя с пользовательским интерфейсом.

1 Описание проекта

CryptoTracker — это iOS-приложение, созданное для того, чтобы пользователи могли легко получать актуальные сведения о криптовалютах. В приложении представлен список цифровых валют с их текущими ценами и процентным изменением за последние сутки, а также имеется возможность более детального изучения данных о каждой конкретной монете. Интерфейс разработки прост для понимания и полностью соответствует современным стандартам UX/UI дизайна. Основой проекта выступает REST API, обеспечивающий доступ к актуальной информации о криптовалютном рынке в режиме реального времени.

Главная задача проекта - облегчить пользователям доступ к актуальным данным о криптовалютах, чтобы приложение стало полезным инструментом как для новичков в инвестициях, так и для опытных трейдеров.

2 Ход выполнения работ над проектом

Работа над проектом началась с этапа планирования, на котором задачи были распределены между членами команды. В качестве основной архитектурной модели выбрали MVVM (Model-View-ViewModel). Этот подход способствует четкому разделению бизнес-логики и логики представления данных, благодаря чему упрощается поддержка и расширение функциональности приложения.

Для создания пользовательского интерфейса использовались инструменты: фреймворк UIKit[3] для создания интерфейса и библиотека SnapKit [4] для настройки зависемостей между отображениями. Взаимодействие с API и обработка данных реализованы с помощью URLSession, а управление асинхронными операциями осуществлялось через замыкания. Процесс командной работы был организован с применением системы задач, каждая из которых имела определенные временные рамки. Регулярные встречи и проведение код-ревью помогали обнаруживать возможные ошибки и находить наилучшие решения.

3 Поставленная задача

В мои задачи входило настроить проект и разработать экран списка монет. Под настройку входит интегрирование .gitignore файл, а также установление зависимостей библиотеки SnapKit в проект, через cocoapods.

Главные сложности при работе в разработке экрана списка монет заключались в следующем:

- оптимизация производительности для больших объемов данных: Необходимо обеспечить эффективное управление и отображение больших списков криптовалют, чтобы приложение оставалось отзывчивым и быстрым даже при обработке тысячи элементов,
- поддержка различных форматов экранов и устройств:

  Учитывая разнообразие устройств iOS, важно было сделать интерфейс адаптивным, чтобы обеспечить корректное отображение информации как на iPhone, так и на iPad, независимо от их размеров,
- реализация функции фильтрации:

  Разработка интуитивно понятной системы фильтрации для быстрого нахождения нужных криптовалют, что требует продуманного подхода к индексации и взаимодействию с интерфейсом,

4 Задача разработки экрана списка всех криптовалют

Экран со списком информации был реализован в классе CoinsListViewController. Этот экран предоставляет пользователю данные о всех криптовалютах и краткую информацию о них. Например, название криптовалюты, ее текущую стоимость, процентное изменение стоимости и дату последнего обновления, рисунок 1.

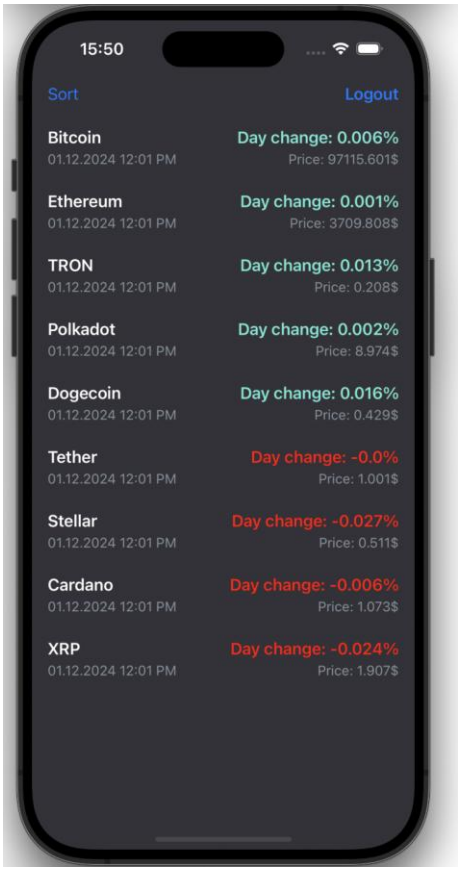

Рисунок 1 – Экран CoinsListViewController

Контроллер списка криптовалют был реализован в классе CoinsListViewController, инкапсулирующем логику отображения и взаимодействия с ViewModel. Для работы с таблицей использовались UITableView и настраиваемые ячейки, что позволило эффективно отображать данные о криптовалютах.

Основная функциональность включала отображение списка монет, сортировку по изменениям цен и обновление данных. Для обновления таблицы использовался UIRefreshControl, а состояния отображения (загрузка, успешное обновление, ошибка) обрабатывались с помощью реактивных подписок на события из ViewModel.

Для обработки пользовательских действий, таких как выбор сортировки, реализован UIAlertController, позволяющий переключаться между сортировкой по возрастанию и убыванию. Навигация на экран авторизации и детальной информации о криптовалюте выполнялась через методы ViewModel, обеспечивая четкое разделение обязанностей.

Особенностью реализации стало использование протокола ICoinsListViewModel, что обеспечило слабую связность между View и ViewModel. Это упростило модульное тестирование и позволило легко заменять ViewModel при масштабировании функционала приложения.

5 Анализ работы и выводы

Проект позволил мне углубить знания в области работы с сетевыми запросами, использования архитектуры MVVM и библиотек Swift. В процессе работы возникли трудности, связанные с обработкой данных API и корректным отображением информации при отсутствии подключения к сети. Эти проблемы удалось решить за счет тестирования и применения асинхронных механизмов обработки данных.

Несмотря на возникающие трудности, удалось планомерно выполнять поставленные задачи, следуя дедлайнам. Работа над проектом стала ценным опытом, позволившим не только освоить новые технологии, но и улучшить навыки работы в команде.

6 Взаимодействие с командой и руководителем проекта

Работа с командой была структурирована эффективно: задачи были равномерно распределены между участниками на основе их текущих знаний, и каждый занимался своей специализацией. Благодаря регулярным совещаниям удавалось координировать изменения и оперативно решать возникающие трудности.

Руководитель проекта сыграл важную роль, поддерживая команду в решении сложных задач и проверке архитектурных решений. Его участие существенно способствовало успешному выполнению проекта и достижению целей.

7 Оценка руководителя команды

Руководитель команды зарекомендовал себя как уверенный и опытный руководитель, который отлично справлялся с координацией усилий всех участников проекта. Он грамотно распределял обязанности, обеспечивал своевременную обратную связь и следил за тем, чтобы работа соответствовала намеченным целям.

Благодаря его методам, в команде сохранилась позитивная атмосфера, и все понимали свою роль и обязанности. Руководитель умело поддерживал гармонию между индивидуальными инициативами и общей стратегией, что облегчало решение возникающих вопросов.

Особо стоит отметить его способность видеть проект целостно, предлагая оптимальные решения для интеграции различных модулей. Это позволяло каждому сосредоточиться на своей зоне ответственности, не теряя общий фокус.

В сумме, руководитель команды проявил высокий профессионализм, и его усилия стали ключевыми для успешного выполнения проекта.

Заключение

В заключение можно сказать, что цель проекта была успешно достигнута. Основные задачи, включая разработку экрана со списком информацией о всех валютах, были успешно реализованы. Предстоит лишь выполнить небольшие улучшения, касающиеся оптимизации пользовательского интерфейса и повышения производительности приложения.

Я внёс значительный вклад в проект, занимаясь созданием экрана со списком информацией о всех валютах. Работа над проектом позволила мне приобрести ценный опыт использования современных технологий, таких как SnapKit, UIKit и применение архитектурного паттерна MVVM. Кроме того, я научился более эффективно планировать свои задачи и решать сложные технические задачи.

Список использованных источников

1. Статья “MVVM архитектруа приложения” - https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93viewmodel
2. Документация об использовании API приложения: Messary IO - https://messari.io/api
3. Официальная документация Apple об фреймворке UIKit - https://developer.apple.com/documentation/uikit/
4. Официальная документация об библотеки Snapkit - https://snapkit.github.io/SnapKit/docs/

Приложение. Техническое задание к проекту.

Этапы задач:

- Анализ предметной области,
- Проектирование,
- Разработка,
- Ручное тестирование.

Задачи:

- Название: Разработать экран авторизации
  Ответственный: Баташов Богдан Александрович
  Описание: Первый вью контроллер для авторизации (AuthViewController), просто 2 текс Филда (UITextField) и кнопка (UIButton).

- Название: Разработать экран списка монет
  Ответственный: Востров Илья Анатольевич
  Описание: Список всех монет сделанный через tableview.
  На ячейке укажите название монеты, ее стоимость, и ее изменение за сутки или час. (Данные брать из API)
  Пока список монет грузится, отображается спинер (UIActivityIndicator).
  После тапа на ячейку, осуществляется переход на 3-тий вью контроллер.

- Название: Разработка экрана подробной информации о монете
  Ответственный: Титор Матвей Андреевич
  Описание: 3 - ий вью контроллер. Просто детальная информация о монете.
  (Можно в ряд добавить лейблы с любой дополнительной информацией о монете)
  При нажатии на ячейку доставать название монеты и передавать на этот экран. Дальше использовать название для запроса

- Название: Внедрить координатор для навигации между контроллерами
  Ответственный: Мелихов Андрей Юрьевич
  Описание: Нужно добавить Координатор
  - может пушить и попать вьюконтроллер

- Название: Разработать класс StorageService для сохранения состояния авторизации пользователя

  Ответственный: Мелихов Андрей Юрьевич

  Описание: Класс который инкапсулирует в себе логику (методы) для сохранения состояния isAuth: Bool переменной в UserDefaults

- Название: Разработать класс NetworkLayer, инкапсулирующий логику взаимодействия с сетью

  Ответственный: Титор Матвей Андреевич

  Описание: - Построить сетевой слой. (Network Layer in Swift)
  - Разобраться как использовать Codable = Encodable & Decodable
  - GET/POST/PATCH - REST API
  - URLSeesion, URL, URLRequest
  - JSONDecoder

- Название: Разработать сервис икапсулирующий логику работы с CoreData (Опционально)

  Ответственный: Востров Илья Анатольевич

  Описание: Создание класса CoreDataManager. Если юзер не был авторизован

- Название: Добавить кнопки фильтрации в NavigationBar

  Ответственный: Попов Артемий Альбертович

  Описание: В навигейшен бар слева добавьте кнопку с возможностью сортировки изменения цены за сутки или за час

- Название: Настроить проект

  Ответственный: Востров Илья Анатольевич

  Описание: Интегрировать .gitignore файл, а также установить зависимости в проект (SnapKit), через cocoapods

- Название: Создать кастомные UI элементы

  Ответственный: Попов Артемий Альбертович

  Описание: - Наследник UITextField, который умеет менять бордер, если произошла ошибка и отображать внизу error message (по сути надо сделать наследника CustomTextField: UITextField, настроить его делегаты, а потом его вместе с CustomLabel: UILabel поместить в класс TextFieldView: UIView и при верстке использовать только его)

  - Наследник UIButton