МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования

**«Южно-Уральский государственный университет**

**(национальный исследовательский университет)»**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

|  |
| --- |
| ДОПУСТИТЬ К ЗАЩИТЕ  Заведующий кафедрой, д.ф.-м.н., профессор  \_\_\_\_\_\_\_\_\_\_ Л.Б. Соколинский  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

**Разработка веб-приложения для отслеживания**

**курса крипто-валют**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ЮУрГУ – 02.03.02.2023.308-268.ВКР

|  |  |
| --- | --- |
|  | Научный руководитель,  доцент кафедры СП, к.ф.-м.н., доцент \_\_\_\_\_\_\_\_\_\_ Радченко Г.И.  Автор работы, студент группы КЭ-401  \_\_\_\_\_\_\_\_\_\_ В.А. Немцев  Ученый секретарь  (нормоконтролер)  \_\_\_\_\_\_\_\_\_\_\_\_\_ И.Д. Володченко  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

Челябинск, 2023 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования

**«Южно-Уральский государственный университет**

**(национальный исследовательский университет)»**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

УТВЕРЖДАЮ

Зав. кафедрой СП

\_\_\_\_\_\_\_\_\_\_ Л.Б. Соколинский

06.02.2023 г.

**ЗАДАНИЕ**

**на выполнение выпускной квалификационной работы бакалавра**

студенту группы КЭ-401

Немцеву Вячеславу Александровичу,

обучающемуся по направлению

02.03.02 «Фундаментальная информатика и информационные технологии»

1. **Тема работы** (утверждена приказом ректора от \_\_.\_\_.2023 г. № \_\_\_)

Разработка веб-приложения для отслеживания курса крипто-валют.

1. **Срок сдачи студентом законченной работы:** 05.06.2023 г.
2. **Исходные данные к работе2**
3. API для разработки веб-приложения. [Электронный ресурс] URL: https://rapidapi.com/Coinranking/api/coinranking1 (дата обращения: 30.01.2023 г.).
4. Веб-приложение forklog. [Электронный ресурс] URL: https://forklog.com/rates(дата обращения: 30.01.2023 г.).
5. Документация библиотеки состояний Redux. [Электронный ресурс] URL: https://redux-toolkit.js.org/ (дата обращения: 30.01.2023 г.).
6. Lewis Antony, The Basics of Bitcoins and Blockchains: An Introduction to Cryptocurrencies and the Technology that Powers Them. Лондон. изд: Mango Media.
7. Документация библиотеки языка JavaScript React. [Электронный ресурс] URL: https://reactjs.org/ (дата обращения: 30.01.2023 г.).
8. Документация библиотеки redux-saga. [Электронный ресурс] URL: redux-saga.js.org/docs/api/ (дата обращения: 30.01.2023 г.).
9. Документация CSS-in-JS библиотеки styled-components. [Электронный ресурс] URL: styled-components.com (дата обращения: 30.01.2023 г.).
10. **Перечень подлежащих разработке вопросов**
11. Выполнить анализ предметной области.
12. Спроектировать веб-приложение.
13. Реализовать веб-приложение.
14. Произвести тестирование разработанной системы.
15. **Дата выдачи задания:** 07.02.2023 г.

**Научный руководитель,**

доцент кафедры СП, к.ф.-м.н., доцент Г.И. Радченко

**Задание принял к исполнению** В.А. Немцев

# ГЛОССАРИЙ

1. *Майнинг* – добыча блоков в сети блокчейн для обеспечения крипто-валютных платформ [1].
2. *Реактивность* – способ автоматически обновлять систему в зависимости от изменения потока данных [2].
3. *Рендеринг* – процесс получения изображения по модели с помощью компиляции программы [3].
4. *DOM-дерево* – это интерфейс, обрабатывающий HTML-документ как древовидную структуру в которой каждый узел представляет собой объект, содержащий часть документа [3].
5. *Монтирование* – вставка элементов в DOM-дерево [3].

ОГЛАВЛЕНИЕ

[ГЛОССАРИЙ 4](#_Toc127771727)

[ВВЕДЕНИЕ 6](#_Toc127771728)

[1. АНАЛИЗ ЛИТЕРАТУРЫ И СМЕЖНЫХ ПРОЕКТОВ 8](#_Toc127771729)

[1.1. Понятие блокчейн 8](#_Toc127771730)

[1.2. Понятие крипто-валюты и их виды 9](#_Toc127771731)

[1.3. Биржи крипто-валют 10](#_Toc127771732)

[1.4. Приложения для отслеживания курсов крипто-валют 11](#_Toc127771733)

[1.5. Платформы для разработки веб-приложений 15](#_Toc127771734)

[2. ПРОЕКТИРОВАНИЕ 18](#_Toc127771735)

[2.1. Анализ требований к системе 18](#_Toc127771736)

[2.2. Варианты использования системы 18](#_Toc127771737)

[2.3. Архитектура системы 20](#_Toc127771738)

[ЛИТЕРАТУРА 22](#_Toc127771739)

# ВВЕДЕНИЕ

**Актуальность**

В современном мире существует множество видов хранения денежных активов – финитные валюты, недвижимость, фондовый рынок, драгоценные металлы и пр. Одним из таких видов хранения является крипто-валюта. Это достаточно популярный способ хранения денежных средств в современном мире, все благодаря его надежности. Однако, курсы отдельных крипто-валют постоянно меняются, что вводит неопытного пользователя в ступор [1].

Популярность крипто-валют как вида хранения денежных активов побудила собой создание множества различных крипто-валют. В связи с этим обычному пользователю, недостаточно погруженному в блокчейн технологии, однако желающему прикоснуться к миру блокчейн, становится очень сложно отслеживать котировки крипто-валют, разбираться в их видах, а также начать зарабатывать на крипто-биржах.

**Постановка задачи**

Целью выпускной квалификационной работы является разработка веб-приложения для отслеживания курсов крипто-валют. Для достижения поставленной цели необходимо решить следующие задачи:

1. выполнить анализ предметной области;
2. спроектировать веб-приложение;
3. реализовать веб-приложение;
4. произвести тестирование разработанной системы.

**Структура и содержание работы**

Работа состоит из введения, четырех глав, заключения и списка литературы. Объем работы составляет 50 страниц, объем списка литературы – 21 источник.

В первой главе описывается …

Вторая глава посвящена …

В третьей главе …

В приложении А содержится …

Глоссарий является необязательной частью текста работы.

# 1. АНАЛИЗ ЛИТЕРАТУРЫ И СМЕЖНЫХ ПРОЕКТОВ

1.1. Понятие блокчейн

Блокчейн – это неизменяемая, децентрализованная база данных, доступ к которой имеют все узлы компьютерной сети [1]. Он хранит в себе данные в цифровом формате, при этом каждый последующий узел в сети имеет хэш предыдущего. Блокчейн гарантирует безопасность и точность в записи и хранении данных, без третьих лиц. Ключевое отличие блокчейн от стандартных баз данных заключается в определенном подходе в их структурировании [4].

Блокчейн собирает информацию в группы, так называемые «блоки», хранящие множества данных. «Блоки» имеют определенные возможности для их хранения, когда они переполняются, они закрываются и ссылаются на предыдущий заполненный узел (блок), вся последующая информация записывается по аналогичному принципу [4].

Блокчейн работает по принципу добычи цифровой информации для записи и распространения, но не для изменения. Все записи в блокчейн нельзя изменить, удалить или иным образом уничтожить. В следствие этого блокчейн принято называть DLT (с англ. Технология распределенного реестра).

В следствие высокой степени безопасности баз данных блокчейн, у множества пользователей возникла потребность хранить свои денежные сбережения в блокчейн. В следствие этого, блокчейн имеет ключевую роль в развитии крипто-валютных систем, например, таких как Bitcoin, Ethereum и других [5].

Таким образом, блокчейн – это неизменяемая и надежная база данных, благодаря которой, пользователи могут хранить свои сбережения и средства.

1.2. Понятие крипто-валюты и их виды

Крипто-валюта – особенный вид денежных средств в цифровой форме, использующий шифрование для защиты транзакций. Выпуск крипто-валюты происходит децентрализовано. Все участники сети равноправны и могут в любой момент переводить или получать крипто-валюту [7].

Крипто-валюты делятся на различные категории. В силу того, что Bitcoin – одна из самых первых валют [8], ее принято выделять в отдельную категорию, остальные делятся на несколько видов:

* альткоины;
* токены;
* стейблкоины;
* NFT;
* DeFi.

Альткоин (или альтернативный коин) – это любая крипто-валюта со своим блокчейн (за исключением Bitcoin). В связи с тем, что у Bitcoin открытый исходный код, разработчики альткойнов могут ускорять транзакции, оптимизировать процесс майнинга, создавать различные автоматизированные контракты, формировать базу для работы с крипто-приложениями.

Токен – цифровой актив, не имеющий своего собственного блокчейн. Вместо майнинга токены выпускаются в полной эмиссии. Зачастую такие активы выпускают различные компании с целью привлечения средств на развитие своих продуктов. Инвесторы, в свою очередь получают гарантии того, что компания выполнит перед ними свои обязательства.

Стейблкоины ­ цифровые деньги, цена которых привязана к материальным активам (обычная валюта, золото и пр.). Курс Bitcoin и альткоинов достаточно нестабилен и меняется каждый день, в то время как стейблкоины предельно стабильны, и несмотря на то, что их котировки могут меняться, происходит это гораздо реже и без резких скачков [9].

NFT – невзаимозаменяемые токены. Зачастую их используют для переноса блокчейн прав на владение уникальными активами. Например, произведения искусства, внутриигровые предметы в онлайн-играх и даже нотариально заверенные документы.

DeFi – децентрализованные финансовые сервисы. Стоит отметить, что это не отдельные крипто-валюты, а платформы для объединения различных видов цифровых активов и их функций.

Самыми популярными крипто-валютами являются:

* Bitcoin (BTC);
* Ethereum (ETH);
* Binance Coin (BNB).

Таким образом, крипто-валюта – это один из способов хранить денежные активы, а также иметь возможность заработать на скачках и падениях их цен.

**1.3. Биржи крипто-валют**

Крипто-валютная биржа – это площадка для осуществления торговли и обмена одно цифровой валюты на другую либо на валюты определенной страны (доллары, рубли и пр.). Биржа, наряду с майнингом, один из способов получения крипто-валюты [10].

Благодаря высокой волатильности, крипто-валюты используют как спекулятивный инструмент, это позволяет извлечь из нее максимальную прибыль.

Биржи делятся на два типа.

1. Площадки, на которых предусмотрена возможность обменивать крипто-валюту на фиатные валюты.
2. Площадки, на которых можно обменивать только крипто-валюту на крипто-валюту.

На текущий момент существует множество бирж со своими достоинствами и недостатками. Ниже приведены самые популярные представители крипто-валютных бирж:

* Poloniex;
* Exmo.

Poloniex – американская крипто-валютная биржа, включающая в свой функционал помимо обменника крипто-валют детальную статистику.

Exmo – универсальный сервис, занимающийся торговлей самых популярных коинов. Ниже приведены самые популярные крипто-валюты, представленые на площадке «Exmo».

* BitCoin;
* ETH;
* DogeCoin.

Основным преимуществом этой биржи является возможность обменивать валюту на рубли, доллары, евро и др.

Таким образом, крипто-валютная биржа, это место, где пользователь может обменять или купить крипто-активы за финитную или крипто-валюту.

**1.4. Приложения для отслеживания курсов крипто-валют**

Для того, чтобы сделать правильный выбор крипто-валюты при инвестировании, а также узнать тенденции в мире блокчейн, пользователи используют веб-приложения с трекерами крипто-валют. Большинство таких приложений имеют подробную статистику с графиками, статистикой, возможностью поиска и сортировки по определенным параметрам (например, по типу крипто-валюты). Однако, зачастую они не имеют функционала аутентификации на платформе.

**Forklog**

Веб-приложение содержит в себе полный набор информации о крипто-валютах, биржах и блокчейн в целом. На главной странице приложения расположен список статей о блокчейн. Интерфейс главной страницы представлен на рисунке 1.



Рисунок 1 – Главная страница приложения «Forklog»

Помимо этого, forklog имеет страницы с котировками коинов, подробной информации о них, страницу со прогнозами тенденций в мире блокчейн. Внутри страницы нет возможности отсортировать коины по наибольшему изменению, а также узнать важные детали коина такие как история появления, или к какому типу он относится. Также, у forklog нет особенных возможностей, которые могли бы его выделить на фоне конкурентов – например, сравнение двух коинов на графиках. Однако, forklog предлагает пользователю детальную информацию на гибких и кастомизируемых графиках, дает возможность найти желаемый коин при помощи поля ввода. Интерфейс страницы котировок крипто-валют представлен на рисунке 2.



Рисунок 2 – Интерфейс страницы «курсы криптовалют» на «forklog»

**BitInfoCharts**

Веб-приложение содержит в себе детальную статистику по крипто-валютам, включающую в себя:

* капитализацию на бирже;
* цену на текущий момент в американском долларе;
* количество блоков, полученных майнингом;
* награду за блок, добытый майнингом;
* интересную информацию о коине (дата первого добытого блока, количество звезд на GitHub, последний коммит в GitHub репозиторий).

Так же, как и Forklog, BitInfoCharts в интерфейсе имеет поиск и сортировку коинов по различным полям. В приложении есть отдельная страница со списком всех крипто-валют, на которой отображена базовая информация о них, дополнительно при нажатии на любой из коинов, приложение перенаправит пользователя на отдельную ссылку с более подробной информацией о крипто-валюте. Интерфейс страницы о BitCoin представлен на рисунке 3.



Рисунок 3 – Интерфейс страницы о BitCoin в веб-приложении «BitInfoCharts»

На странице с детальной информацией о конкретной крипто-валюте отображается график изменения цены валюты за последний месяц, цена указана в долларах США. Помимо этого, отображается количество крипто-валюты в целом, капитализация крипто-валюты. Также, отображается информация о средней цене транзакций, количестве блоков в блокчейн, объем блокчейн и гистограмма комиссии крипто-валюты за 24 часа. Гистограмма BitCoin показана на рисунке 4.



Рисунок 4 – Гистограмма комиссии для BitCoin

Основное преимущество «BitInfoCharts» – это детальная статистика и подробная информация по крипто-валютам. В то время как основной недостаток – устаревший дизайн.

Таким образом, основная цель приложений для отслеживания курсов крипто-валют – отображение статистики с наглядными графиками и тенденциями роста или спада конкретного коина.

# 1.5. Платформы для разработки веб-приложений

В современном мире веб-разработки существует 3 различных вида подхода к разработке приложений – SPA, MPA, PWA.

SPA (single page application) – одностраничное приложение, загружающееся на одну HTML-страницу, переход между страницами в таких приложениях реализован средствами языка JavaScript. Такой подход позволяет упростить разработку, а также увеличить скорость работы приложения. Однако, главным минусом у такого подхода является полное отсутствие SEO оптимизации, так как рендеринг страницы происходит на клиенте, в силу этого, поисковые роботы Google (и другие) видят страницу приложения пустой. Однако, в случаях, где контент статичен и важен для SEO, этот недостаток SPA, может закрыть серверный рендеринг, позволяющий писать собственный API для рендеринга на сервере [11].

MPA (multi page application) – многостраничное приложение, один из первых подходов в разработке приложений, при нем страница полностью обновляется при малейшем изменении состояний на клиенте. Главный плюс такого подхода, в отличие от SPA – отличная SEO оптимизация, т.к. рендер приложения происходит на сервере. Минусов у такого подхода много. Первый из них – сложность разработки. Для разработки MPA требуется использовать фреймворки, заточенные на фронтенд и бекенд одновременно, например, Angular, Flask или Ruby on rails, что усложняет разработку и делает ее менее поддерживаемой и вариативной. Второй большой минус MPA – плохая скорость работы, в силу того, что перерисовка страницы происходит каждый раз при изменении состояний в приложении [11].

PWA (progressive web applications) – прогрессивное веб приложение, позволяющее запускать себя как в браузере, так и в качестве десктопного или мобильного приложения. Главная особенность и преимущество PWA – кроссплатформенность, приложение, написанное под персональный компьютер может быть использовано в качестве мобильного или веб-приложения. Однако, минусом такого подхода является поддержка некоторых возможностей таких приложениях в некоторых браузерах (Safari, Edge).

На данный момент, веб-приложения пишутся на различных фреймворках и библиотеках языка JavaScript, в силу их удобства и скорости разработки. Они не требуют работы с DOM-деревом [12], вместо этого в них есть специальные методы и функции для работы с событиями, референсами и куки.

Самые популярные JavaScript-фрейморки на данный момент.

1. React (для серверного рендеринга React приложений используют Next) [13];
2. Vue (для серверного рендеринга Vue приложений используют Nuxt);
3. Svelte (для серверного рендеринга Svelte приложений используют SvelteKit).

Также, важнейшим трендом веб-разработки является типизация. Для типизации в веб-приложениях используют разработанный компанией Microsoft язык TypeScript. TypeScript компилируется в JavaScript и формально является полезной надстройкой над языком, делая его более строгим и стабильным [13]. Типизация очень полезна для командной разработки, а также в случаях, где нужно работать с большим количеством запросов и валидацией.

Помимо этого, в современных веб-приложениях используют библиотеки состояний для хранения данных и устранения проблемы сильной связанности (prop-drilling) между множеством компонентов. Самые популярные библиотеки для работы с состояниями:

* Redux (Redux Toolkit) [14];
* Vuex (библиотека состояний для Vue);
* MobX;
* Zustand.

Самая часто используемая из них – Redux, из-за хорошего отладчика в виде Redux DevTools. Однако в сообществе разработчиков многие скептически относятся к Redux из-за сильной нагроможденности [13]. Проблема была решена с выпуском Redux Toolkit, однако в некоторых проектах до сих пор используется старая версия Redux.

Таким образом, современная веб разработка имеет множество различных способов и подходов к реализации веб приложений. Каждый из них имеет право на существование, выбор инструмента и подхода к разработке зависит от конкретной задачи разработчика.

В силу того, что библиотека React позволяет разработчику не думать о работе с DOM-деревом и сконцентрировать усилия на работе с данными было принято использовать именно ее. Помимо этого, для работы с API требуется строгая типизация, во избежание непредвиденных ситуаций с приведением типов данных. Также, будет использован Redux-toolkit и Redux-saga [13] для работы с состояниями и асинхронными запросами к базе данных крипто-валют. Для работы со стилизацией HTML элементов и компонентов будет использована CSS-in-JS библиотека styled-components [15].

# 2. ПРОЕКТИРОВАНИЕ

# 2.1. Анализ требований к системе

Для разработки программного продукта обязательно выдвигать к нему требования, для получения четкой картины целей продукта и выбора оптимальных инструментов разработки. Потому, были выделены следующие функциональные и нефункциональные требования.

**Функциональные требования**

1. Веб сервис должен отображать список крипто-валют и крипто-бирж на отдельных страницах.
2. Веб сервис должен иметь функционал поиска и сортировки крипто-валют.
3. Веб сервис должен иметь функционал перехода на отдельную страницу с крипто-валютой для детального анализа.
4. Веб сервис должен иметь функционал отображения статистики крипто-валют на графиках.
5. Веб сервис должен иметь функционал загружать графики с выбранной крипто-валютой.
6. Веб сервис должен иметь функционал для добавления крипто-валют и бирж в «избранное».

**Нефункциональные требования**

1. Веб сервис должен подключаться к открытому API крипто-валют и крипто-бирж [14].
2. Веб сервис должен быть адаптивным (поддерживать разрешения экрана всех современных устройств).
3. Веб сервис должен поддерживаться в последних версиях (последние 2) всех популярных браузеров (Google Chrome, Safari, Yandex).

# 2.2. Варианты использования системы

Для проектирования приложения был использован язык графического описания для объектного моделирования UML. По выдвинутым к системе требованиям, была составлена диаграмма вариантов использования. Диаграмма приведена на рисунке 5.

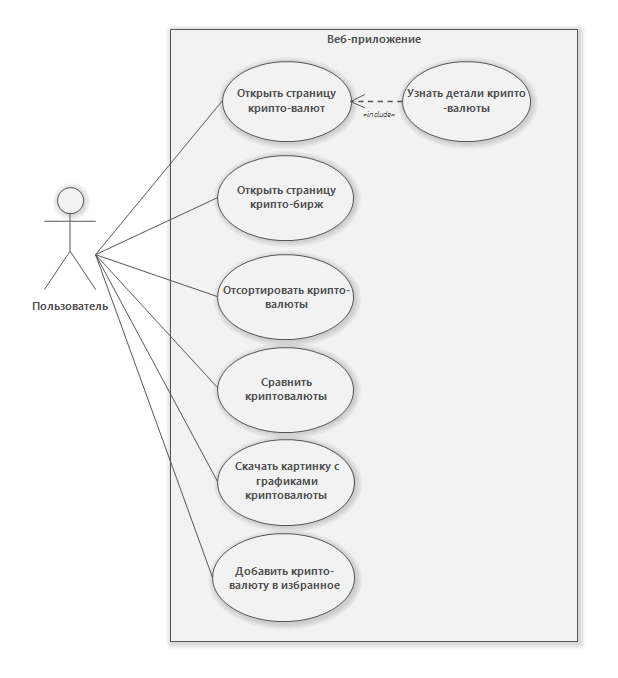


Рисунок 5 – Диаграмма вариантов использования для веб-приложения для отслеживания курсов крипто-валют

С системой взаимодействует один актер – непосредственный пользователь системы. Пользователь может выполнять действия, представленные ниже.

1. «Открыть страницу крипто-валют». Пользователь может открыть отдельную страницу со списком крипто-валют.
2. «Узнать детали крипто-валюты». Пользователь, зайдя на страницу крипто-валют, может узнать детали конкретного представителя из списка.
3. «Открыть страницу крипто-бирж». Пользователь может открыть отдельную страницу со списком крипто-бирж.
4. «Отсортировать крипто-валюты». Пользователь может отсортировать крипто-валюты на соответствующей странице.
5. «Сравнить крипто-валюты». Пользователь может на отдельной странице с выбранной крипто-валютой произвести сравнение крипто-валюты с любым другим представителем крипто-валют и увидеть разницу в их цене и изменении на графиках, а также в их деталях.
6. «Скачать картинку с графиками крипто-валюты». Пользователь может скачать картинку с графиком крипто-валюты, а также с графиком сравнения крипто-валют.
7. «Добавить крипто-валюту в избранное». Пользователь может добавить крипто-валюту в избранное, для отслеживания детальной информации о ней.

# 2.3. Архитектура системы

Архитектура приложения использует популярный на данный момент подход – Flux. Это подход, при котором решается проблема жесткой связанности компонентов, средствами реактивного программирования [16]. В такой архитектуре используются хранилища, на которые подписываются отдельные UI-компоненты, которые посредством диспетчеров и селекторов передают, изменяют или получают данные хранилища. Реактивность помогает отдельным компонентам избежать лишний рендер связанных компонентов, посредством обновления данных напрямую из хранилища.

В связи с этим, для прецедента номер 2 из пункта 2.2 второй главы работы – «Узнать детали крипто-валюты», была разработана диаграмма потоков данных [17]. Диаграмма представлена на рисунке 6.

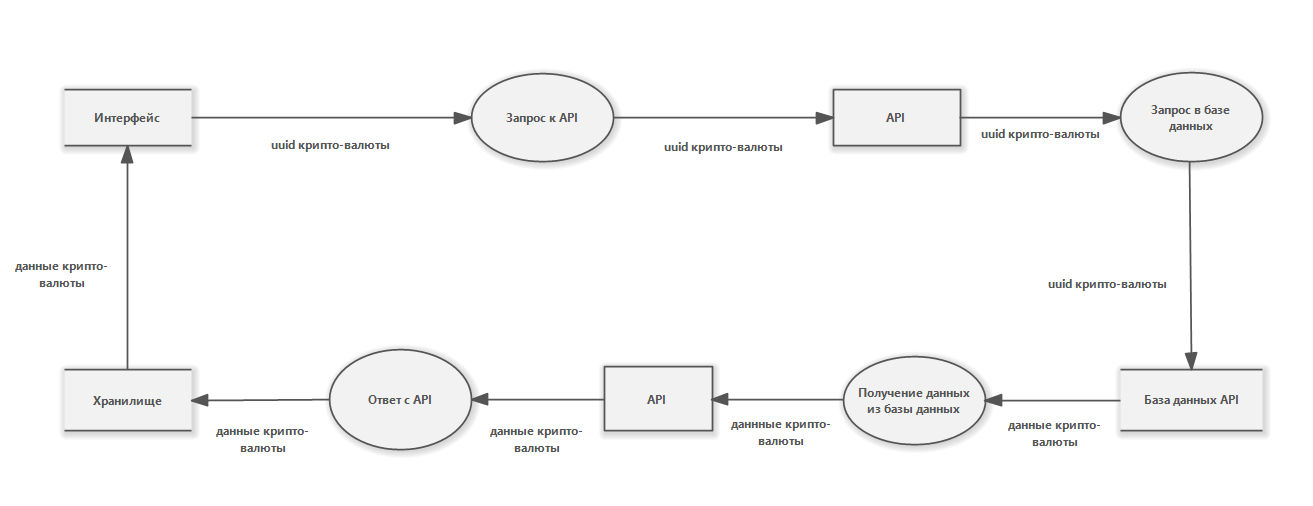


Рисунок 6 – Диаграмма потоков данных для прецедента «Узнать детали крипто-валюты»

При монтировании страницы веб-приложение получает из URL страницы uuid крипто-валюты, затем отправляет запрос к API [14]. После обработки запроса API возвращает веб-приложению данные по крипто-валюте, веб-приложение кладет данные в хранилище, а затем передает его в Интерфейс пользователя.

В данной главе были выдвинуты требования к системе, а также проведен их анализ. Помимо этого, были реализованы диаграмма вариантов использования и диаграмма потоков данных для одного из прецедентов.

# 3. РЕАЛИЗАЦИЯ СИСТЕМЫ

# 3.1. Программные средства

Реализация веб-приложения для отслеживания курсов крипто-валют производилась с помощью следующего набора инструментов:

1. React версии 18.2. – библиотека языка JavaScript, позволяющая с простотой разрабатывать SPA, декомпозировать UI компоненты, обращаться к узлам DOM дерева и работать с большими объемами данных.
2. TypeScript версии 4.8.4 – язык программирования, позволяющий добавлять типизацию и ООП в JavaScript. TypeScript при компиляции собирается в нативный JavaScript код, который в свою очередь успешно запускается во всех современных браузерах
3. Redux toolkit версии 1.8.4 – библиотека состояний JavaScript, позволяющая хранить состояния в глобальном хранилище и отделять логику работы с данными от UI компонентов.
4. Redux-saga версии 1.2.1 – библиотека JavaScript для связи серверной части приложения (API) непосредственно с клиентской частью приложения.
5. Styled-components версии 5.3.6 – библиотека, позволяющая писать CSS стили внутри JavaScript.
6. ChartJS версии 3.9.1 – библиотека JavaScript, позволяющая строить графики на основе структурированных данных.
7. ESLint версии 7.25 – линтер для JavaScript, для отслеживания и устранения синтаксических ошибок и предупреждений.
8. Vite версии 3.1 – сборщик JavaScript-приложений.

# 3.2. Сборка и конфигурация приложения

**Настройка сборщика веб-приложения**

Современные браузеры способны собирать и анализировать информацию только из HTML, CSS и JavaScript в чистом виде, в силу этого для работы TypeScript, jsx (шаблонизатор HTML, используемый в React), а также CSS-in-JS используются сборщики, преобразующие их в понятный для браузера формат.

Самыми популярными сборщиками JavaScript-приложений являются:

* Webpack;
* ESBuild;
* Vite.

По данным npm, количество загрузок Webpack порядка 27 млн. загрузок в неделю, в то время как у ESBuild и Vite количество загрузок сильно меньше – у ESBuild 10 млн. загрузок, а у Vite 3 млн [19]. Однако, количество загрузок у Vite и ESBuild достаточно быстро растет, обусловлено это высокой производительностью и скоростью сборки. Vite использует модули, написанные на языке Go, что позволяет быстро собирать и запускать приложение, а конфигурация Vite достаточно легковесная и простая, в отличие от конфигураций Webpack, в которых встречается очень много сложных зависимостей [20]. Потому, для разработки системы был выбран сборщик Vite. Конфигурация сборки представлена в листинге 1.

Листинг 1 – Конфигурация сборки веб-приложения

import { defineConfig } from 'vite';

import react from '@vitejs/plugin-react';

import eslintPlugin from 'vite-plugin-eslint';

export default defineConfig({

plugins: [

eslintPlugin({

exclude: ["/@react-refresh", "\*\*/\*.css"]

}),

react(),

]

});

Vite поддерживает работу с готовыми популярными плагинами, так в приложении используются плагины для работы с React и ESLint.

Кроме этого, в приложении используется язык TypeScript, который в свою очередь является оберткой над языком JavaScript. Для TypeScript используются специальные конфигурации для компилирования языка в JavaScript. Конфигурация TypeScript в приложении представлена в листинге 2.

Листинг 2 – Конфигурация TypeScript в приложении

{

"compilerOptions": {

"target": "ESNext",

"useDefineForClassFields": true,

"lib": ["DOM", "DOM.Iterable", "ESNext"],

"allowJs": false,

"skipLibCheck": true,

"esModuleInterop": false,

"allowSyntheticDefaultImports": true,

"strict": true,

"forceConsistentCasingInFileNames": true,

"module": "ESNext",

"moduleResolution": "Node",

"resolveJsonModule": true,

"isolatedModules": true,

"noEmit": true,

"jsx": "react-jsx"

},

"include": ["src"],

"references": [{ "path": "./tsconfig.node.json" }]

}

Конфигурация в себе содержит точку входа в приложение – src, включаемые библиотеки и специальные параметры, исключающие возможность нестрогой типизации из JavaScript.

**Настройка кроссбраузерности веб-приложения**

Каждый современный браузер содержит в себе свои особенные CSS-стили, потому, для поддержки кроссбраузерности верстки используются два подхода:

* Нормализация CSS;
* Сброс стилей.

Нормализация CSS – подход, при котором задаются стили поверх браузерных, этот подход позволяет использовать нужные стили во всем приложении, без написания дополнительных.

Сброс стилей – подход, при котором все реализованные внутри браузера стили сбрасываются до стилей из спецификации HTML, такой подход позволяет делать верстку более гибкой и предсказуемой.

Внутри приложения используется сброс стилей, так как этот подход является более предсказуемым и масштабируемым.

**Настройка линтера веб-приложения**

Для удобства разработки в языке JavaScript и TypeScript используются линтеры и форматеры кода. Самым популярным является ESLint [21]. По данным npm он имеет 31 млн. загрузок за неделю [19]. ESLint позволяет отслеживать ошибки и предупреждения по заданным правилам в конфигурации, а также устранять некоторые из них. Конфигурация линтера представлена в листинге 3.

Листинг 3 – Конфигурация линтера в приложении

module.exports = {

env: {

node: true,

es2020: true

},

globals: { React: true },

root: true,

plugins: ["@typescript-eslint"],

extends: [

"eslint:recommended",

"plugin:@typescript-eslint/recommended",

"plugin:react/recommended",

"plugin:react-hooks/recommended"

],

settings: {

react: {

version: "detect"

}

},

};

Таким образом, в данном пункте были описаны конфигурации языка TypeScript, а также были выбраны инструменты и подходы для сборки, сброса стилей и форматирования кода приложения.

# 3.3. Реализация страницы списка крипто-валют

Для удобства разработки, задача по реализации страницы списка крипто-валют была декомпозирована на следующие пункты:

* Верстка UI компонентов на странице;
* Реализация логики хранения данных на клиенте;
* Реализация логики отображения данных на клиенте;
* Реализация логики взаимодействия с API.

В рамках первого пункта была реализована анимация загрузки данных на клиент, реализовано поле ввода текста, переключатель опции, компонент крипто-валюты и пагинация. Детально в рамках выпускной квалификационной работы будут рассмотрены компоненты пагинации и загрузки данных. Верстка и CSS компонента анимации загрузки данных на клиент приведена в листинге 4.

Листинг 4 – Верстка компонента анимации

export const Animation = keyframes`

0%, 100% {

margin-top: 0;

}

50% {

margin-top: 2rem;

}

`;

export const LoadingWrapper = styled.div`

display: flex;

justify-content: center;

position: fixed;

top: 0;

padding-top: 30vh;

Продолжение листинга 4

height: 100vh;

width: 100vw;

background-color: ${COLORS.carrierPigeonBlue};

color: ${COLORS.white};

`;

export const Dot = styled.div`

background-color: ${COLORS.duckEggBlue};

border-radius: 50%;

width: 0.75rem;

height: 0.75rem;

margin: 0 0.25rem;

animation: ${Animation} 1s infinite;

animation-delay: ${(props: DotProps) => props.delay};

`;

const DOTS = [

'0s',

'0.05s',

'0.1s',

'0.15s',

'0.2s',

'0.25s',

'0.3s',

];

const Loader = () => (

<LoadingWrapper>

{DOTS.map((delay, delayIndex) => (

<Dot key={delayIndex} delay={delay} />

))}

</LoadingWrapper>

);

Для реализации функции анимации был использован CSS контроллер keyframes, позволяющий задавать правила анимирования компонентов [3]. Для масштабируемости компонента Loader, отрисовка компонента Dot, помещена в итератор массива DOTS.

Компонент пагинации не содержит в себе логики, а используется лишь как UI компонент и вся логика происходит без его участия, компонент лишь отвечает за рендер и монтирование содержимого на странице. Компонент пагинации представлен в листинге 5.

Листинг 5 – Компонент пагинации

const Pagination = (props: PaginationProps) => {

const {

onClick

} = props;

const totalCount = useSelector(makeSelectTotalCount);

const currentPage = useSelector(makeSelectCurrentPage);

const pagination: any[] = [];

for (let i = 0; i < Math.floor(totalCount/10); i++) {

pagination.push(i + 1);

}

return (

<PaginationWrapper>

{pagination.map((page) => {

const isActive = page === currentPage;

return (

<PaginationItem

key={page}

isActive={isActive}

onClick={!isActive && onClick}

>

{page}

</PaginationItem>

);

})}

</PaginationWrapper>

);

};

…

Интерфейс полученной страницы представлен на рисунке 7.



Рисунок 7 – Интерфейс веб-приложения

(Рассказать про то, как передаются данные, как отображаются, листинг саги, пагинация поиск, листинг стилей, листинг верстки, скриншот, описание подключение к апи)

3.4. Реализация модуля отрисовки графиков

3.5. Реализация модуля добавления в избранное

# ЛИТЕРАТУРА

1. Lewis A., The Basics of Bitcoins and Blockchains: An Introduction to Cryptocurrencies and the Technology that Powers Them. // Mango Media. 2018.
2. Статья «SPA vs. MPA: Pros, Cons & How To Make Final Choice». [Электронный ресурс] URL: https://www.simicart.com/blog/spa-vs-mpa/ (дата обращения: 10.02.2023 г.).
3. Документация браузерного API – MDN. [Электронный ресурс] URL: https://developer.mozilla.org/en-US/ (дата обращения: 10.02.2023 г.)
4. Прасти Н., Блокчейн. Разработка приложений. С.-Петерб. изд. BHV. 2018.
5. Статья «Что такое биткоин и блокчейн». [Электронный ресурс] URL: https://www.kaspersky.ru/blog/bitcoin-easy-explanation/12668/ (дата обращения: 05.02.2023 г.)
6. Статья «What is hashing in Blockchain». [Электронный ресурс] URL: https://learn.bybit.com/blockchain/what-is-hashing-in-blockchain/ (дата обращения: 06.02.2023 г.)
7. Статья «Что такое криптовалюта и как она применяется?». [Электронный ресурс] URL: https://www.kaspersky.ru/resource-center/definitions/what-is-cryptocurrency (дата обращения: 06.02.2023 г.)
8. Antonopoulos A., Mastering Bitcoin. // O’Really Media. 2014.
9. Статья «Stablecoins: definition, how they work, and types». [Электронный ресурс] URL: https://www.investopedia.com/terms/s/stablecoin.asp (дата обращения: 06.02.2023 г.)
10. Статья «Crypto airdrop season: Why people are making thousands for 'free'». [Электронный ресурс] URL: https://www.cnet.com/personal-finance/crypto/crypto-airdrop-season-why-people-are-making-thousands-for-free/ (дата обращения: 10.02.2023 г.)
11. Документация JavaScript библиотеки React. [Электронный ресурс] URL: https://reactjs.org/ (дата обращения: 10.02.2023 г.)
12. Документация языка TypeScript. [Электронный ресурс] URL: https://www.typescriptlang.org/docs/ (дата обращения: 10.02.2023 г.)
13. Документация библиотеки Redux. [Электронный ресурс] URL: https://redux-toolkit.js.org/ (дата обращения: 10.02.2023 г.)
14. Документация энд-поинтов API «Coinranking». [Электронный ресурс] URL: https://rapidapi.com/Coinranking/api/coinranking1 (дата обращения: 12.02.2023 г.)
15. Документация CSS-in-JS библиотеки styled-components [Электронный ресурс] URL: URL: styled-components.com (дата обращения: 30.01.2023 г.).
16. Статья об архитектуре веб-приложений «Flux» [Электронный ресурс] URL: https://medium.com/@marina.kovalyova/flux-the-react-js-application-architecture-773f515d068d (дата обращения: 18.02.2023 г.)
17. Статья «DFD диаграммы – зачем они нужны и какие бывают». [Электронный ресурс] URL: https://habr.com/ru/post/668684/ (дата обращения: 18.02.2023 г.)
18. Flanagan D., JavaScript: The Definitive Guide, 7th Edition // O’Really Media Inc 2020.
19. Пакетный менеджер «npm.js». [Электронный ресурс] URL:

https://www.npmjs.com/ (дата обращения 26.02.2023 г.)

1. Документация сборщика «Vite». [Электронный ресурс] URL: https://vitejs.dev/ (дата обращения: 26.02.2023 г.)
2. Документация линтера «ESLint». [Электронный ресурс] URL:

https://eslint.org/ (дата обращения: 26.02.2023 г.)