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. Веб-сервис крипто-валют, используемый 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, Издательство Mango Media, 2018. ­ 408 c.
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:** 06.02.2023 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](#_Toc130131234)

[ВВЕДЕНИЕ 6](#_Toc130131235)

[1. АНАЛИЗ ЛИТЕРАТУРЫ И СМЕЖНЫХ ПРОЕКТОВ 8](#_Toc130131236)

[1.1. Понятие блокчейн 8](#_Toc130131237)

[1.2. Понятие крипто-валюты и их виды 9](#_Toc130131238)

[1.3. Биржи крипто-валют 10](#_Toc130131239)

[1.4. Приложения для отслеживания курсов крипто-валют 11](#_Toc130131240)

[1.5. Платформы для разработки веб-приложений 15](#_Toc130131241)

[2. Анализ требований к системе 18](#_Toc130131242)

[2.1. Описание системы «Cryptocurrency» 18](#_Toc130131243)

[2.2. Определение функциональных требований 18](#_Toc130131244)

[2.3. Определение нефункциональных требований 19](#_Toc130131245)

[2.4. Варианты использования системы 19](#_Toc130131246)

[3. ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ СИСТЕМЫ 22](#_Toc130131247)

[3.1. Flux архитектура 22](#_Toc130131248)

[3.2. Диаграмма компонентов веб-приложения Cryptocurrency 23](#_Toc130131249)

[3.3. Диаграмма деятельности для прецедента «Добавить в избранное» 24](#_Toc130131250)

[4. РЕАЛИЗАЦИЯ СИСТЕМЫ 27](#_Toc130131251)

[4.1. Программные средства 27](#_Toc130131252)

[4.2. Сборка и конфигурация приложения 28](#_Toc130131253)

[4.3. Реализация страницы списка крипто-валют 31](#_Toc130131254)

[4.4. Реализация модуля отрисовки графиков 35](#_Toc130131255)

[4.5. Реализация модуля добавления в избранное 36](#_Toc130131256)

[4.6. Реализация модуля загрузки графиков 37](#_Toc130131257)

[ЛИТЕРАТУРА 41](#_Toc130131258)

# 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 блокчейна 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 блокчейне 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 блокчейна, у множества пользователей возникла потребность хранить свои денежные сбережения в блокчейне. Вследствие 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].

Принято выделять следующие ключевые категории крипто-валют:

* Bitcoin (выделяется отдельно так как это одна из первых крипто-валют в мире с открытым исходным кодом [8]);
* альткоины;
* токены;
* стейблкоины;
* NFT;
* DeFi.

Альткоины (или альтернативные коины) – это любые крипто-валюты со своим блокчейном (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.

Токены – цифровые активы, не имеющие своего собственного блокчейна. 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) [10];
* Ethereum (ETH) [11];
* Binance Coin (BNB) [12].

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 [13].

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 [14];
* Exmo [15].

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 [8, 10];
* ETH [11];
* DogeCoin [16].

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 блокчейна, 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**

Онлайн трекер крипто-валют ForkLog [17] содержит в себе полный набор информации о крипто-валютах, биржах и блокчейне в целом. На главной странице приложения расположен список статей о блокчейне. 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 блокчейна. 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. Помимо этого, в приложении ForkLog отсутствует возможность добавления крипто-валюты в избранное, что не позволит пользователям в должной мере собирать по ней статистику простым и удобным способом. 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 «цены крипто-валют» на ForkLog

**BitInfoCharts**

Веб-приложение BitInfoCharts [18] 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 блокчейне, объем блокчейна 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 [2].

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 [2].

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 [19], 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 [19].
2. Vue [20].
3. Svelte [21].

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 [22]. 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) [23];
* Vuex [20];
* MobX [24];
* Zustand [25].

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 отдельных частей кода [23]. 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 и внешними сервисами [26] требуется строгая типизация, во избежание непредвиденных ситуаций с приведением типов данных. Также, будет использован Redux-toolkit и Redux-saga [23] 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 [27].

**Вывод по первой главе**

Подводя итоги по первой главе, нужно отметить, что мониторинг крипто-валют – очень важная и востребованная задача на данный момент, потому в рамках дипломной работы будет реализовано веб-приложение для отслеживания крипто-валют.

# 2. Анализ требований к системе

## 2.1. Описание системы «Cryptocurrency»

Целью данной работы является разработка веб-приложения для трекинга крипто-валют – Cryptocurrency.

Главное назначение для данного приложения – помощь в поиске и систематизации актуальных данных о крипто-валюте. Приложение Cryptocurrency позволит добавлять интересующие пользователя крипто-валюты и биржи в избранное для упрощенного мониторинга данных. Пользователь будет иметь возможность отсортировать крипто-валюты по росту, падению, популярности и прочим важным для него параметрам, также приложение предусматривает возможность поиска крипто-валют при помощи ввода названия или идентификатора крипто-валюты в специальную панель поиска. Вместе с этим пользователь будет иметь возможность найти крипто-валюту из большого списка всех возможных крипто-валют. На отдельной странице каждой крипто-валюты для удобства пользователя изменения котировок будут представлены на графиках. Также у пользователя будет отдельная страница избранного, на которой он сможет детальнее изучить котировки интересующей его крипто-валюты или биржи.

Ключевыми особенностями Cryptocurrency будут:

* возможность сравнивать две крипто-валюты на одном графике;
* добавление крипто-валюты или биржи в избранное;
* возможность сохранить картинку с графиком и информацией о крипто-валюте в формате png.

## 2.2. Определение функциональных требований

Веб-приложение Cryptocurrency должно соответствовать функциональным требованиям, приведенным ниже.

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

## 2.3. Определение нефункциональных требований

В рамках анализа требований к веб-приложению Cryptocurrency были выдвинуты нефункциональные требования, приведенные ниже.

1. Веб-приложение Cryptocurrency должно подключаться к внешнему веб-сервису крипто-валют и крипто-бирж [14].
2. Приложение должно быть адаптивным (поддерживать разрешения экрана всех современных устройств).
3. Приложение должно поддерживаться в последних версиях (последние 2) всех популярных браузеров (Google Chrome, Safari, Yandex).

## 2.4. 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.

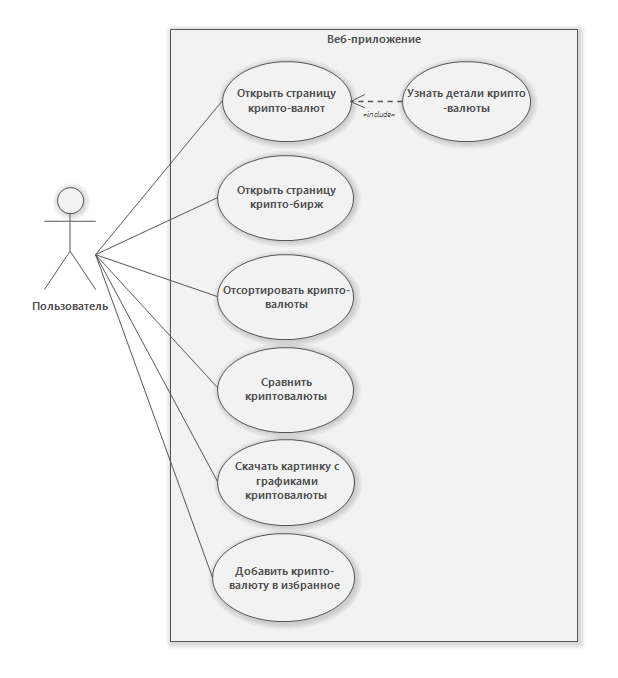


Рисунок 5 – Диаграмма вариантов использования для веб-приложения   
для отслеживания курсов крипто-валют

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.

**Выводы по второй главе**

В данной главе были рассмотрены основные требования к системе, была описана цель разработки программного продукта и реализована диаграмма вариантов использования веб-приложения Cryptocurrency.

# 3. ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ СИСТЕМЫ

В рамках выпускной квалификационной работы было проведено проектирование архитектуры веб-приложения для трекинга крипто-валют Cryptocurrency.

## 3.1. Диаграмма компонентов веб-приложения Cryptocurrency

На рисунке 7 представлена диаграмма компонентов веб-приложения Cryptocurrency.

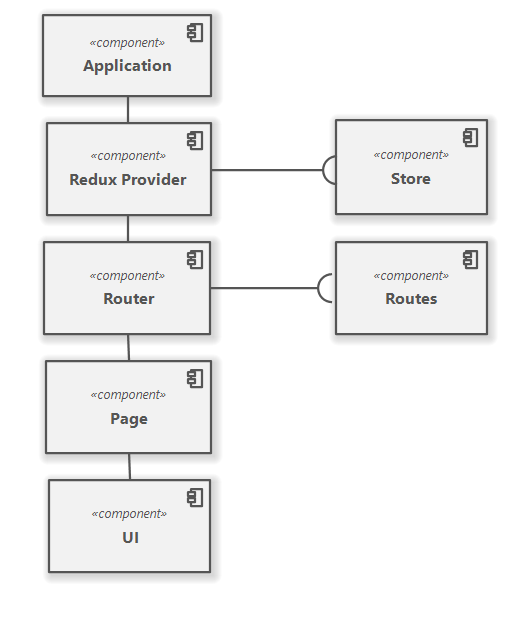


Рисунок 6 – Диаграмма компонентов веб-приложения

Приложение состоит из семи компонентов, описанных ниже.

1. Application – компонент, который является телом приложения, в нем находятся все провайдеры и потомки. Кроме этого, Application является точкой входа в приложение.
2. Redux Provider – компонент, при помощи которого в приложение добавляется возможность работы с хранилищем, описанным в пункте 3.1.
3. Store – компонент глобального хранилища приложения.
4. Router – компонент, при помощи которого в SPA-приложение добавляется мультистраничность посредством роутинга.
5. Routes – компонент хранящий в себе страницы SPA.
6. Page – компонент, в котором реализуется бизнес-логика приложения.
7. UI – компонент, не содержащий в себе бизнес-логики, отвечающий только за отрисовку данных переданных в него из родителя.

Выбор такого набора компонентов обусловлен современными подходами в архитектуре веб-приложений, описанными в следующем пункте данной главы.

## 3.2. Flux архитектура

Веб-приложение Cryptocurrency будет использовать популярный на данный момент подход к разработке программного обеспечения – Flux. Это подход, при котором решается проблема жесткой связанности компонентов, средствами реактивного программирования [28]. В такой архитектуре используются Flux хранилища, представляющие из себя глобальный объект, доступный на всех этапах жизненного цикла приложения. На эти хранилища подписываются отдельные модули или UI-компоненты, которые посредством диспетчеров и селекторов передают, изменяют или получают данные хранилища. Реактивность помогает отдельным компонентам избежать лишний рендер связанных компонентов, посредством получения обновлений данных напрямую из хранилища.

Веб-приложение будет иметь статический сервер и будет обращаться к стороннему веб сервису для получения необходимых данных. Логика работы будет реализована непосредственно на клиенте благодаря описанной выше Flux архитектуры.

В связи с этим, для прецедента номер 2 из пункта 2.4 второй главы работы – «Узнать детали крипто-валюты», была разработана диаграмма потоков данных [29]. Диаграмма представлена на рисунке 7.

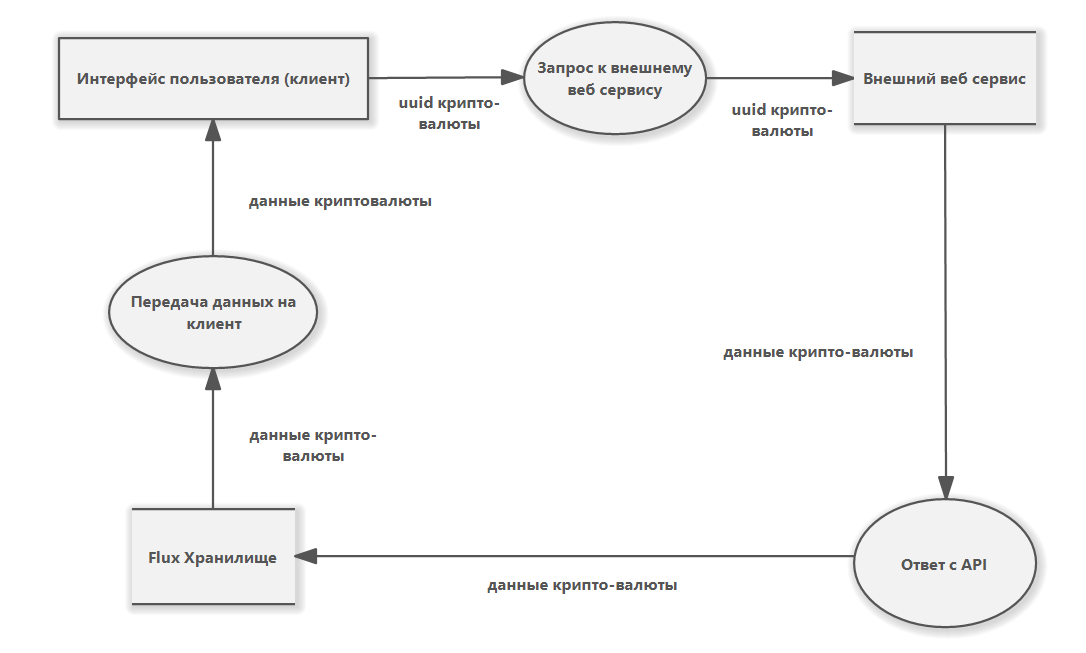


Рисунок 7 – Диаграмма потоков данных для прецедента   
«Узнать детали крипто-валюты»

Интерфейс пользователя инициирует отправку запроса к внешнему веб сервису при монтировании страницы, затем отправляет запрос к внешнему веб-сервису крипто-валют [26]. После обработки запроса, внешний веб-сервис возвращает веб-приложению данные по крипто-валюте, затем веб-приложение собирает данные и убирает их в глобальное хранилище, а затем передает его в интерфейс пользователя.

## 3.3. Диаграмма деятельности для прецедента «Добавить в избранное»

В силу того, что в рассмотренных, в четвертом пункте первой главы выпускной квалификационной работы, приложениях не было функционала для пометки интересных пользователю крипто-валют, было решено добавить такую отличительную возможность в разрабатываемое веб-приложение Cryptocurrency. В следствие этого была разработана диаграмма деятельности для прецедента «Добавить в избранное», упомянутого в четвертом пункте второй главы работы. На рисунке 8 представлена диаграмма деятельности для прецедента «Добавить в избранное».

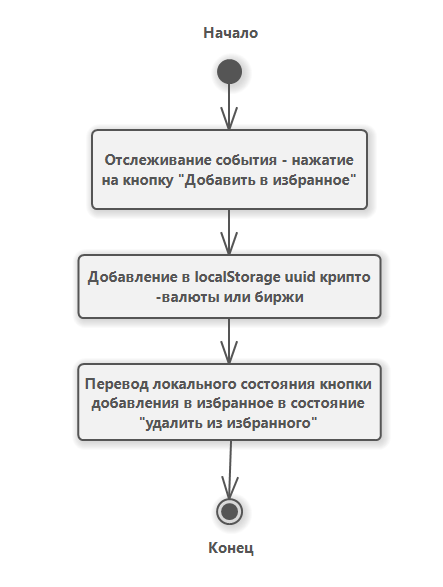


Рисунок 8 – Диаграмма деятельности для прецедента   
«Добавить в избранное»

После того, как пользователь нажмет на кнопку добавить в избранное, идентификатор криптовалюты (uuid) будет передан в локальное хранилище браузера – localStorage. Затем состояние кнопки, на котором находится слушатель события добавления крипто-валюты в избранное переключится на состояние удаления крипто-валюты из избранного пользователя. Стоит отметить, что удаление крипто-валюты из избранного происходит аналогичным образом.

**Выводы по третьей главе**

В данной главе была описана архитектура разрабатываемого веб-приложения для трекинга крипто-валют Cryptocurrency. Вместе с этим, были приведены диаграмма потоков данных для прецедента «узнать детали крипто-валюты», диаграмма компонентов и диаграмма деятельности для прецедента «добавить крипто-валюту в избранное».

# 4. РЕАЛИЗАЦИЯ СИСТЕМЫ

## 4.1. 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 [19].
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 [22].
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 [23].
4. Redux-saga версии 1.2.1 – библиотека JavaScript для связи серверной части приложения (API) непосредственно с клиентской частью приложения [30].
5. Styled-components версии 5.3.6 – библиотека, позволяющая писать CSS стили и анимации внутри JavaScript [27].
6. ChartJS версии 3.9.1 – библиотека JavaScript, позволяющая строить графики на основе структурированных данных [31].
7. ESLint версии 7.25 – линтер для JavaScript, для отслеживания и устранения синтаксических ошибок и предупреждений [32].
8. Vite версии 3.1 – сборщик JavaScript-приложений [33], улучшающий скорость сборки приложения в dev режиме.

# 4.2. 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 [35];
* ESBuild [36];
* Vite [33].

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 [35]. Высокое количество загрузок ESBuild обусловлено тем, что в качестве dev зависимости при сборке приложения при помощи Vite используется ESBuild.

Конфигурация Vite достаточно легковесная и простая, в отличие от конфигураций Webpack, в которых встречается очень много сложных и запутанных зависимостей [35]. Потому, для упрощения 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';

import path from 'path';

import { fileURLToPath } from 'url';

const \_\_filename = fileURLToPath(import.meta.url);

const \_\_dirname = path.dirname(\_\_filename);

export default defineConfig({

plugins: [

eslintPlugin({

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

}),

react(),

],

resolve: {

alias: {

'@': path.resolve(\_\_dirname, './src'),

},

},

server: {

port: 3002,

},

})

Vite поддерживает работу с готовыми популярными плагинами, так в приложении используются плагины для работы с React и ESLint. Помимо этого, в конфигурацию сборку были добавлены порт для запуска приложения локально и alias, необходимый для более простого обращения к абсолютным путям в коде.

**Настройка TypeScript**

Для разработки приложения 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 в себе набор правил компиляции кода в JavaScript, 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 отслеживания ошибок и предупреждений при разработке в языке JavaScript и TypeScript используются линтеры и форматировщики кода. Самым популярным является ESLint [36]. По данным npm он имеет 31 млн. загрузок за неделю [34]. 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.

# 4.3. 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 внешним веб-сервисом.

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 Dot = styled.div`

background-color: ${COLORS.duckEggBlue};

border-radius: 50%;

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>

);

};

Компонент пагинации принимает в себя слушатель события нажатия на кнопку (номер страницы), количество данных и текущую выбранную страницу. При помощи флага isActive определяется отрисовка выбранной и невыбранной страницы. Если флаг isActive истинный, событие onClick, отвечающее за выбор страницы не сработает.

Для получения данных из внешнего веб-сервиса используется библиотека redux-saga [30]. Эта библиотека позволяет накладывать побочные эффекты на actions, подключенных к redux редьюсерам [23], посредством такой конструкции, как генераторы в JavaScript [38]. В листинге 6 приведена логика получения данных с внешнего веб-сервиса.

Листинг 6 – Логика получения данных с внешнего веб-сервиса

export function\* getCoinsFromApi({ payload }: any): any {

try {

const {

limit,

offset,

} = payload;

const search = yield select(makeSelectSearchParams);

const orderBy = yield select(makeSelectOrderBy);

const direction = yield select(makeSelectOrderDirection);

const response = yield call(request, getCoinsApi(limit, offset, search, orderBy, direction), COINS\_REQUEST\_OPTIONS);

if (response) {

yield put(getCoinsResponse(response));

yield put(updateTotalCount(response?.data?.stats.total));

}

} catch(e) {

console.warn('coins request error:', e);

}

}

export default function\* coinsSaga() {

yield takeLatest(getCoins, getCoinsFromApi);

}

При доставке события getCoins в хранилище redux, срабатывает побочный эффект – getCoinsFromApi, создающий GET запрос к серверу. При помощи встроенного эффекта select из библиотеки redux-saga собираются query параметры [3] из хранилища redux. При помощи встроенного эффекта call создается запрос к внешнему веб-сервису с заданными параметрами, после того, как клиент получает асинхронный ответ с сервера, данные записываются в хранилище при помощи эффекта put и redux actions.

В результате данной реализации была получена страница списка крипто-валют, находящаяся на странице /coins. Интерфейс страницы списка крипто-валют представлен на рисунке 9.



Рисунок 9 – Интерфейс страницы списка крипто-валют

## 4.4. Реализация модуля отрисовки графиков

Модуль отрисовки графиков реализован посредством библиотеки ChartJS [31]. Среди UI компонентов этой библиотеки был выбран компонент Line, осуществляющий рендер графика с заданными опциями и данными. Верстка этого компонента представлена в листинге 7.

Листинг 7 – Верстка компонента Line

<coinStyles.ChartWrapper>

<Line

options={OPTIONS}

data={data}

/>

</coinStyles.ChartWrapper>

Свойство data в компоненте Line наполняется из redux хранилища. В хранилище данные, необходимые для построения графика, попадают после отправки запроса при монтировании страницы. При получении ответа с сервера данные собираются в нужный для ChartJS формат. Реализация маппинга данных для отрисовки графиков представлена в листинге 8.

Листинг 8 – Реализация маппинга данных для отрисовки графиков

const response = yield call(request, formatCoinRequest(uuid), options);

if (response) {

yield put(getCoinResponse(response));

const coin = yield select(makeSelectCoinData);

const labels = coin?.sparkline

.map((spark: string, sparkIndex: number) => `${sparkIndex}h`);

yield put(updateData({

labels,

datasets: [{

label: coin?.symbol,

data: coin?.sparkline.map((spark: string) => Number(spark)),

borderColor: COLORS.bioticGrasp,

backgroundColor: COLORS.bioticGrasp,

}],

}));

}

Для реализации сравнения двух графиков был реализован дополнительное событие, которое при его вызове добавляет в параметр datasets данные другой крипто-валюты, тем самым образуется наложение двух графиков. Пример такой ситуации изображен на рисунке 10.

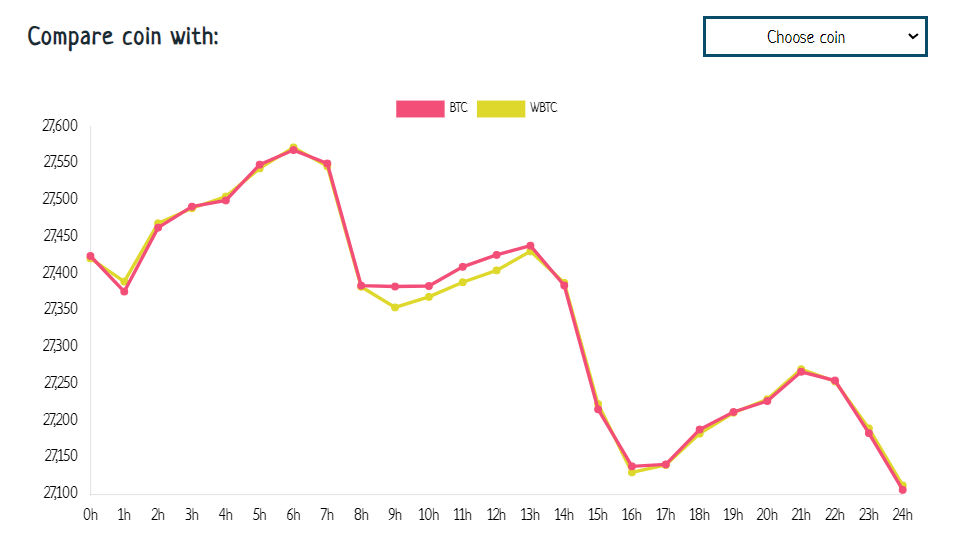


Рисунок 10 –График сравнения крипто-валют BTC и Wrapped BTC

## 4.5. Реализация модуля добавления в избранное

Для добавления в избранное был использован объект localStorage [38]. Это веб-хранилище, позволяющее сохранять данные внутри браузера. При помощи методов getItem и setItem осуществляется запись и чтение данных объекта. Реализация функций добавления в избранное и удаления биржи из избранного приведены в листинге 9.

Листинг 9 – Функции добавления биржи в избранное и удаления биржи из избранного.

const [isFavorite, setIsFavorite] = useState(localStorage.getItem('favoriteExchange') === uuid);

const handleAddInFavorites = () => {

localStorage.setItem('favoriteExchange', uuid);

setIsFavorite(true);

}

const handleRemoveFromFavorites = () => {

localStorage.removeItem('favoriteExchange');

setIsFavorite(false);

}

При рендере элемента списка бирж задается значение состояния useState избранной биржи из объекта localStorage, при срабатывании события handleAddInFavorites добавляется uuid выбранной биржи, а также состояние отображения кнопки отображается переключается на true. При срабатывании handleRemoveFromFavorites из объекта localStorage уничтожается поле favoriteExchange, а состояние кнопки переключается на false.

Избранное отображается на домашней странице приложения. Интерфейс домашней страницы с добавленными в избранное биржей и крипто-валютой представлены на рисунке 11.

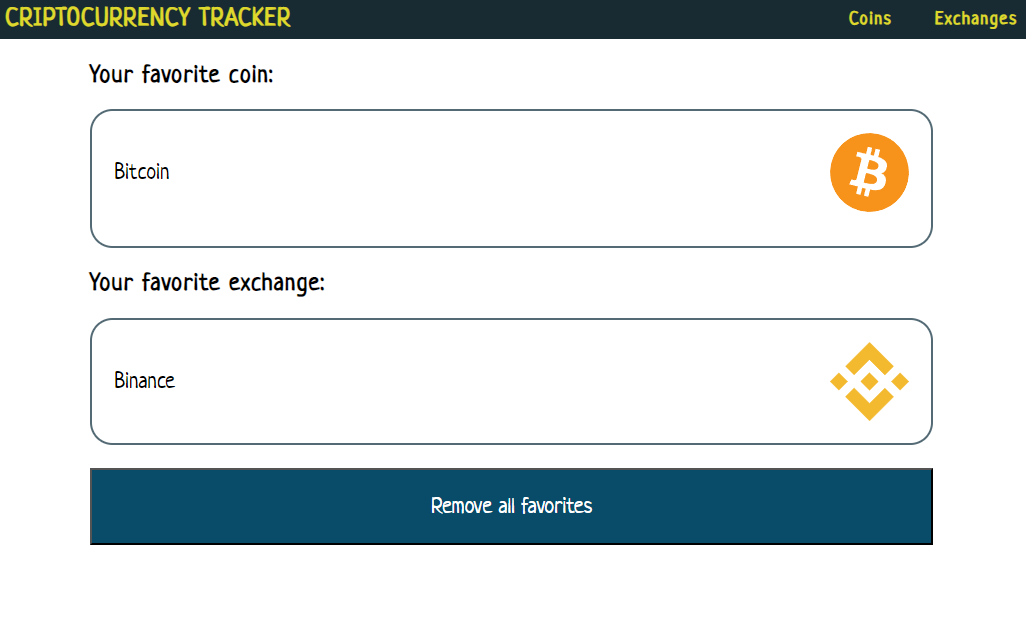


Рисунок 11 – Интерфейс домашней страницы

## 4.6. Реализация модуля загрузки графиков

Для реализации загрузки графиков используется npm пакет html-to-image [39]. Он позволяет при помощи конструкции Promise [38] асинхронно создавать скриншот DOM-узла. Реализация события загрузки графика в формате png приведена в листинге 10.

Листинг 10 – Реализация события загрузке графика

const handleDownloadGraphs = () => {

if (!graphRef) return;

setIsDisabledButton(true);

toPng(graphRef.current, { cacheBust: true })

.then((url) => {

const link = document.createElement('a');

const date = new Date();

const currentTime = date.getHours() + ':' + date.getMinutes() + ':' + date.getSeconds();

link.download = compareCoin

? `${currentTime}-${coin.name}-compared-${compareCoin.name}.png`

: `${currentTime}-${coin.name}.png`;

link.href = url;

link.click();

setIsDisabledButton(false);

})

.catch((e) => {

setIsDisabledButton(false);

alert('error', e);

});

};

Для реализации подключения к узлу DOM дерева, был использован хук useRef, позволяющий получить представление элемента верстки [19]. При отлавливании события handleDownloadGraphs создается ссылка, затем асинхронно вызывается загрузка скриншота узла, после этого пользователю загружается файл, в названии которого указано наименование криптовалюты и дата загрузки. Пример загруженного файла представлен на рисунке 12.

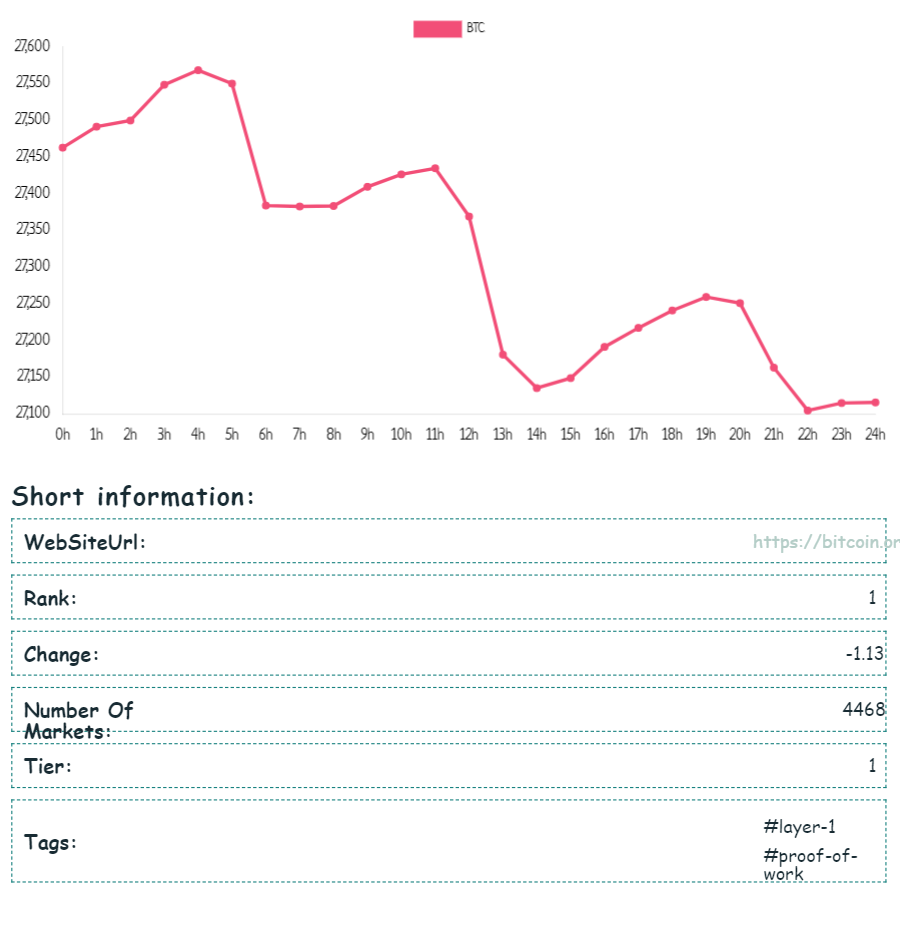


Рисунок 12 – Загруженный файл в формате png

**Выводы по четвертой главе**

В данной главе были описаны средства и подходы к реализации веб-приложения Cryptocurrency. Вместе с этим были описаны детали реализации некоторых ключевых модулей системы, а также приведены промежуточные результаты реализации приложения.

# ТЕСТИРОВАНИЕ

В соответствии с требованиями, выдвинутыми во втором пункте второй главы было решено провести функциональное мануальное тестирование. Для реализации поставленной задачи было принято решение разместить приложение с помощью GitHub Pages [40]. Для этого в конфигурацию сборки vite было добавлено поле base: '/graduate work' с названием репозитория кодовой базы веб-приложения [41]. Вместе с этим в конфигурацию проекта были добавлены скрипты для сборки и размещения приложения. Скрипты представлены в листинге 11.

Листинг 11 ­ Скрипты для размещения приложения на GitHub Pages

“predeploy”: “npm run build”,

“deploy”: “gh-pages –d dist”,

Скрипт deploy отвечает за публикацию сборки приложения на сервере GitHub Pages. Перед публикацией срабатывает скрипт predeploy, отвечающий за сборку приложения. После сборки приложения в папку dist, ее содержимое публикуется на сервер GitHub Pages.

Исходя из требований к системе был составлен набор тестов. В таблице 1 представлен набор тестов для проверки корректной работы веб-приложения.

Таблица 1 – Набор необходимых тестов для проверки работы веб-приложения

| **№** | **Название теста** | **Шаги** | **Ожидаемый результат** |
| --- | --- | --- | --- |
| 1 | Тестирование пагинации на странице крипто-валют | С домашней страницы нажать на ссылку «Coins». В открывшейся странице нужно нажать на цифру 2 среди страниц. | Содержимое страницы переключится с 1 на 2 страницу. |
| 2 | Тестирование поиска на странице крипто-валют | На странице /coins в поле ввода «Search coin» ввести название крипто-валюты. | В списке крипто-валют появятся только крипто-валюты с таким названием (либо включающие это слово в свое название) |
| 3 | Тестирование сортировки крипто-валют | На странице /coins сделать выбор одной из опций сортировки и упорядоченности. | Список крипто-валют сортируется по заданным правилам. |
| 4 | Тестирование открытия страницы конкретной крипто-валюты |  |  |
| 5 | Тестирование добавления крипто-валюты в избранное |  |  |
| 6 | Тестирование загрузки данных по графикам конкретной крипто-валюты |  |  |
| 7 | Тестирование сравнения двух крипто-валют на графиках |  |  |
| 8 | Тестирование отображения данных по избранной крипто-валюте на домашней странице |  |  |

Вместе с этим, был выбран набор сред для тестирования приложения. Он представлен в таблице 2.

Таблица 2 – Набор сред тестирования веб-приложения

Основываясь на данных из 1 и 2 таблицы работы, было проведено мануальное функциональное тестирование веб-приложения Cryptocurrency. Результаты тестирования приведены в таблице 3.

Таблица 3 – Результаты тестирования веб-приложения Cryprocurrency

// ЗАКЛЮЧЕНИЕ

# ЛИТЕРАТУРА

1. Lewis A. The Basics of Bitcoins and Blockchains: An Introduction to Cryptocurrencies and the Technology that Powers Them. // Mango Media, 2018.  ­ 408 c.
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. – 252 с.
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. – 298 с.
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. Веб-сайт крипто-валюты BitCoin. [Электронный ресурс] URL: https://bitcoin.org/en/ (дата обращения: 15.03.2023 г.).
11. Веб-сайт крипто-валюты Ethereum. [Электронный ресурс] URL: https://ethereum.org/en/ (дата обращения: 15.03.2023 г.).
12. Веб-сайт крипто-валюты Binance. [Электронный ресурс] URL: https://www.binance.com/en (дата обращения 15.03.2023 г.).
13. 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.).
14. Крипто-валютная биржа Poloniex. [Электронный ресурс] URL: https://poloniex.com (дата обращения: 15.03.2023 г.).
15. Крипто-валютная биржа Exmo. [Электронный ресурс] URL: https://exmo.com (дата обращения: 15.03.2023 г.).
16. Веб-сайт крипто-валюты DogeCoin. [Электронный ресурс] URL: https://dogecoin.com (дата обращения 15.03.2023 г.).
17. Веб-приложение ForkLog. [Электронный ресурс] URL: https://forklog.com/ (дата обращения 15.03.2023 г.).
18. Веб-приложение BitInfoCharts. [Электронный ресурс] URL: https://bitinfocharts.com (дата обращения 15.03.2023 г.).
19. Документация JavaScript библиотеки React. [Электронный ресурс] URL: https://reactjs.org/ (дата обращения: 10.02.2023 г.).
20. Документация JavaScript фреймворка Vue. [Электронный ресурс] URL: https://vuejs.org/ (дата обращения: 15.03.2023 г.).
21. Документация JavaScript фреймворка Svelte. [Электронный ресурс]: https://svelte.dev/ (дата обращения: 15.03.2023 г.).
22. Документация языка TypeScript. [Электронный ресурс] URL: https://www.typescriptlang.org/docs/ (дата обращения: 10.02.2023 г.).
23. Документация библиотеки Redux. [Электронный ресурс] URL: https://redux-toolkit.js.org/ (дата обращения: 10.02.2023 г.).
24. Документация библиотеки MobX. [Электронный ресурс] URL: https://mobx.js.org/ (дата обращения: 15.03.2023 г.).
25. Документация библиотеки Zustand. [Электронный ресурс] URL: https://docs.pmnd.rs/zustand/getting-started/introduction (дата обращения: 15.03.2023 г.).
26. Документация энд-поинтов внешнего веб-сервиса Coinranking. [Электронный ресурс] URL: https://rapidapi.com/Coinranking/api/coinranking1 (дата обращения: 12.02.2023 г.).
27. Документация CSS-in-JS библиотеки styled-components. [Электронный ресурс] URL: URL: styled-components.com (дата обращения: 30.01.2023 г.).
28. 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.).
29. Статья «DFD диаграммы – зачем они нужны и какие бывают». [Электронный ресурс] URL: https://habr.com/ru/post/668684/ (дата обращения: 18.02.2023 г.).
30. Документация бибилиотеки Redux-saga. [Электронный ресурс] URL: https://redux-saga.js.org/ (дата обращения: 15.03.2023 г.).
31. Документация библиотеки ChartJS. [Электронный ресурс] URL: https://www.chartjs.org/ (дата обращения: 15.03.2023 г.).
32. Документация линтера ESLint. [Электронный ресурс] URL:

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

1. Документация сборщика Vite. [Электронный ресурс] URL: https://vitejs.dev/ (дата обращения: 26.02.2023 г.).
2. Пакетный менеджер npm.js. [Электронный ресурс] URL:

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

1. Документация сборщика Webpack. [Электронный ресурс] URL: https://webpack.js.org/ (дата обращения: 26.02.2023 г.).
2. Документация сборщика ESBuild. [Электронный ресурс] URL:

https://esbuild.github.io/ (дата обращения: 26.02.2023 г.).

1. Документация Rollup. [Электронный ресурс] URL: https://rollupjs.org/ (дата обращения: 26.02.2023 г.).
2. Flanagan D. JavaScript: The Definitive Guide, 7th Edition. // O’Really Media Inc 2020. – 1032 с.
3. Пакет html-to-image. [Электронный ресурс] URL: https://www.npmjs.com/package/html-to-image (дата обращения: 26.02.2023 г.).
4. Сервис для размещения веб-страниц GitHub Pages. [Электронный ресурс] URL: https://pages.github.com/ (дата обращения: 02.04.2023 г.)
5. Репозиторий веб-приложения на GitHub. [Электронный ресурс] URL: https://github.com/ViaChessLove/graduate-work