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 «Электронные вычислительные машины»

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой ЭВМ
_____ Д.В. Топольский
«___»_____ 2025 г.

Разработка приложения для технического анализа рынка криптовалют

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУРГУ-090301.2025.405 ПЗ ВКР

Руководитель работы,
к. пед. н., доцент каф. ЭВМ
_____ М.А. Алтухова
«___»_____ 2025 г.

Автор работы,
студент группы КЭ-405
_____ А.А. Корнеев
«___»_____ 2025 г.

Нормоконтролёр,
ст. преподаватель каф. ЭВМ
_____ С.В. Сяськов
«___»_____ 2025 г.

Челябинск-2025

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 «Электронные вычислительные машины»

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ
_____ Д.В. Топольский
«___»_____2025 г.

ЗАДАНИЕ

на выпускную квалификационную работу бакалавра

студенту группы КЭ-405
Корнееву Александру Андреевичу
обучающемуся по направлению
09.03.01 «Информатика и вычислительная техника»

1. **Тема работы: «**Разработка приложения для технического анализа рынка криптовалют» утверждена приказом по университету от «21» апреля 2025 г. №648-13/12 (приложение №9).

2. **Срок сдачи студентом законченной работы:** 1 июня 2025 г.

3. **Исходные данные к работе:**

1. Приложение должно обеспечивать определение направлений ценового движения криптовалют и формирование торговых сигналов на покупку и продажу.
2. В рамках разрабатываемого программного модуля должны быть реализованы автоматическое извлечение и обработка данных с торговых платформ и криптовалютных бирж, таких как Investing, TradingView, Binance и др., для последующего анализа.

3. Разрабатываемый программный модуль (веб-приложение) должен реализовывать алгоритм построения интегрального индикатора валидности криптовалют, основанный на совокупности технических индикаторов с учётом кратковременных рыночных факторов;
4. Генерация торговых сигналов должна учитывать выбранный пользователем тип торговли: скальпинг, краткосрочную или долгосрочную стратегию.
5. Пользователю должна быть предоставлена возможность выбора инструментов технического анализа для анализа конкретных криптовалют.
6. В рамках разработанного веб-приложения должен быть реализован механизм регистрации пользователей с возможностью подписки на получение уведомлений по электронной почте о результатах проведённого технического анализа.

4. **Перечень подлежащих разработке вопросов:**

1. Аналитический обзор тематической литературы.
2. Проектирование архитектуры веб-приложения.
3. Разработка веб-приложения;
4. Проведение тестирования веб-приложения.

5. **Дата выдачи задания:** 2 декабря 2024 г.

Руководитель работы _____/ *М. А. Алтухова* /

Студент _____/ *А. А. Корнеев* /

КАЛЕНДАРНЫЙ ПЛАН

Этап	Срок сдачи	Подпись руководителя
Аналитический обзор тематической литературы	03.03.2025	
Проектирование архитектуры веб-приложения	22.03.2025	
Разработка веб-приложения	12.04.2025	
Проведение тестирования веб-приложения	26.04.2025	
Компоновка текста работы и сдача на нормоконтроль	22.05.2025	
Подготовка презентации и доклада	30.05.2025	

Руководитель работы _____/ *М. А. Алтухова* /

Студент _____/ *А. А. Корнеев* /

Аннотация

А.А. Корнеев. Разработка приложения для технического анализа рынка криптовалют. – Челябинск: ФГАОУ ВО «ЮУрГУ (НИУ)», ВШ ЭКН; 2025, 130 с., 33 ил., библиогр. список – 15 наим.

В рамках выпускной квалификационной работы проведен аналитический обзор современной научно-технической, нормативной и методической литературы, касающейся разработки приложений для технического анализа на рынке криптовалют.

Разработаны и реализованы алгоритмы и структуры данных, позволяющие эффективно анализировать рыночные данные и генерировать сигналы на покупку и продажу активов, что способствует более обоснованным инвестиционным решениям.

С использованием созданного приложения проведены тестовые испытания, в результате которых была проверена функциональность модуля технического анализа, что подтвердило его способность предоставлять точные и актуальные данные для пользователей.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В последние годы рынок криптовалют стал одним из самых динамично развивающихся сегментов финансового рынка, привлекая внимание как профессиональных трейдеров, так и обычных инвесторов. Высокая волатильность цен на криптовалюты создает уникальные возможности для получения прибыли, но одновременно и увеличивает риски. В условиях постоянных изменений и неопределенности, трейдеры нуждаются в надежных инструментах для анализа рыночной информации и принятия обоснованных решений. В этом контексте технический анализ, основанный на изучении исторических данных о ценах и объемах торгов, становится важным инструментом для прогнозирования ценовых движений.

Актуальность исследования обусловлена необходимостью создания эффективного приложения для технического анализа, которое сможет интегрировать современные методы анализа и предоставлять пользователям актуальную информацию о состоянии рынка. Существующие решения часто имеют ограничения по функциональности, удобству использования или доступности данных, что создает потребность в разработке нового инструмента, способного удовлетворить запросы трейдеров и инвесторов. Целью данной работы является создание приложения по техническому анализу для рынка криптовалют, которое будет включать в себя различные методы анализа и прогнозирования цен. Для достижения этой цели необходимо решить несколько задач: исследовать существующие методы технического анализа и их применение к криптовалютам; определить функциональные требования к приложению; разработать прототип приложения с использованием современных технологий; провести тестирование приложения на реальных данных и оценить его эффективность.

Объектом исследования является рынок криптовалют как высоко-волатильная финансовая среда, а предметом исследования выступает процесс

разработки приложения для технического анализа, включая выбор методов анализа, алгоритмов прогнозирования и интерфейса пользователя. Таким образом, данное исследование направлено на создание инновационного инструмента, который будет способствовать более точному анализу и прогнозированию цен на криптовалюты, а также повышению эффективности торговли для пользователей. Обзор существующих приложений на рынке и выявление их недостатков позволит определить ключевые особенности разрабатываемого приложения, что сделает его полезным инструментом как для опытных трейдеров, так и для новичков.

Для выполнения вышеназванной цели были поставлены следующие задачи:

– провести аналитический обзор тематической литературы;

– спроектировать архитектуру веб-приложения;

– сделать программную реализацию;

– провести тестирование реализации.

Разработанное приложение призвано стать инструментом, который не только автоматизирует анализ криптовалют, но и увеличит точность прогнозов.

1. АНАЛИТИЧЕСКИЙ ОБЗОР ТЕМАТИЧЕСКОЙ ЛИТЕРАТУРЫ

1.1. Определение понятия технический анализ

Технический анализ представляет собой систему методов и инструментов, предназначенных для анализа рыночных данных, которые позволяют трейдерам и инвесторам оценивать текущие и будущие ценовые движения активов. Он включает в себя использование графиков, индикаторов, уровней поддержки и сопротивления, а также различных паттернов, что помогает в принятии решений о входе и выходе из торговых позиций.

Технический анализ начал формироваться в конце 19 века, когда Чарльз Доу, один из основателей Dow Jones & Company, разработал теории, ставшие основой для анализа финансовых рынков. Его идеи о движении цен и рыночных трендах легли в основу Доу-теории, которая до сих пор используется трейдерами. В начале 20 века технический анализ продолжал развиваться благодаря работам таких авторов, как Эдвард Дж. Мэги и Джон Мэрфи, которые систематизировали методы анализа и представили их широкой аудитории. Они ввели понятия графиков, уровней поддержки и сопротивления, а также различные индикаторы. С развитием технологий в 1970-х годах началось использование компьютерных программ для технического анализа. Это позволило трейдерам анализировать большие объемы данных и применять сложные алгоритмы для прогнозирования ценовых движений.

В 1980-х и 1990-х годах технический анализ стал популярным среди розничных инвесторов благодаря доступности информации и росту числа онлайн-бирж. Появление новых индикаторов и методов анализа, таких как осцилляторы и волновая теория Эллиота, расширило возможности трейдеров. В последние годы наблюдается рост интереса к использованию машинного обучения и искусственного интеллекта в техническом анализе. Эти

технологии позволяют трейдерам анализировать данные более эффективно и выявлять сложные закономерности, что значительно улучшает точность прогнозов. Таким образом, история технического анализа демонстрирует его эволюцию от простых графиков до сложных алгоритмических моделей, что делает его важным инструментом для трейдеров на современных финансовых рынках.

Технический анализ включает в себя несколько основных видов, каждый из которых имеет свои уникальные методы и подходы. Каждый из этих видов технического анализа может быть использован отдельно или в комбинации с другими методами для более точного прогнозирования ценовых движений на финансовых рынках. Технический анализ делятся на несколько видов таких как графический, паттерн-аналитика, индикаторный, свечной, объемный.

1.2. Определение графического анализа

Графический анализ – это метод технического анализа, который основывается на визуальном представлении ценовых данных через графики. Этот подход позволяет трейдерам и инвесторам оценивать динамику цен, выявлять тренды, уровни поддержки и сопротивления, принимать обоснованные решения на основе исторических данных.

Трендами называют направления движения цен на рынке. Они могут быть восходящими (бычий тренд), нисходящими (медвежий тренд) или боковыми (консолидация). Определение тренда является ключевым аспектом графического анализа, поскольку позволяет трейдерам принимать решения о покупке или продаже активов.

Уровни поддержки в техническом анализе называют ценовые уровни, на которых спрос превышает предложение, что приводит к остановке падения цены. Уровни сопротивления – это уровни, где предложение превышает спрос,

что приводит к остановке роста цены. Эти уровни помогают трейдерам определить потенциальные точки входа и выхода из позиций.

Графический анализ является одним из самых популярных методов в техническом анализе благодаря своей простоте и наглядности. При графическом анализе используют разные виды графиков таких как линейные, барные, свечные:

Линейные графики представляют собой простое соединение цен закрытия за определенные периоды времени. Они позволяют быстро оценить общее направление движения цены, но не предоставляют информации о колебаниях цен в течение периода.

Барные графики – отображают цену открытия, закрытия, максимальную и минимальную цену за определенный период. Каждый бар представляет собой вертикальную линию с горизонтальными черточками, показывающими цены открытия и закрытия. Это позволяет трейдерам более детально анализировать ценовые движения.

Свечные графики (или японские свечи) – аналогичны барным, но представляют информацию в более наглядной форме. Каждая свеча отображает цену открытия, закрытия, максимальную и минимальную цену за период. Цвет тела свечи указывает на направление движения: зеленая (или белая) свеча означает рост цены, а красная (или черная) – падение.

1.3. Определение паттерн-аналитики

Паттерн-аналитикой называют нахождение повторяющихся формаций (паттернов) на графиках цен. Эти паттерны могут сигнализировать о возможных разворотах или продолжениях тренда, что позволяет трейдерам принимать более обоснованные решения о покупке или продаже активов. Паттерн-аналитика основана на предположении, что история имеет тенденцию повторяться, и что определенные формации могут предсказать

будущие движения цен. Паттерны делятся на паттерны разворота и паттерны продолжения.

Паттерны разворота – эти паттерны указывают на возможный разворот текущего тренда. К ним относятся: голова и плечи, двойная вершина и двойное дно, тройная вершина и тройное дно, клин, бриллиант.

Голова и плечи – состоит из трех последовательных пиков, где средний пик (голова) является самым высоким, а два боковых (плечи) – ниже. Обычно левое плечо немного ниже правого. Линия поддержки называется "шея", и при ее пробитии трейдеры открывают короткие позиции. Формирование левого плеча и головы происходит на фоне сильного тренда, однако при образовании правого плеча покупатели или продавцы не могут преодолеть уровень головы, что указывает на ослабление текущей тенденции и возможный разворот. На рисунке 1 представлен паттерн голова и плечи.

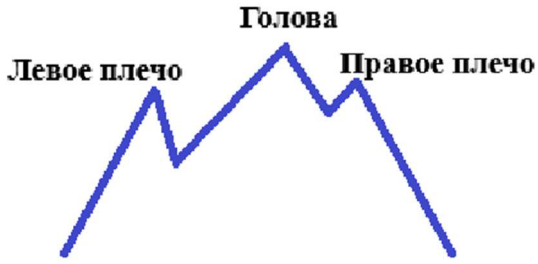

Рисунок 1 – Паттерн «Голова и плечи»

Двойная вершина и двойное дно – состоят из двух последовательных пиков или впадин на одном уровне цен. Эта модель возникает, когда цена сталкивается с сильным уровнем поддержки или сопротивления и не может его преодолеть, что приводит к развороту. Такие паттерны часто формируются на экстремумах, указывая на ослабление текущей тенденции. На рисунке 2 представлены паттерны двойная вершина и двойное дно.

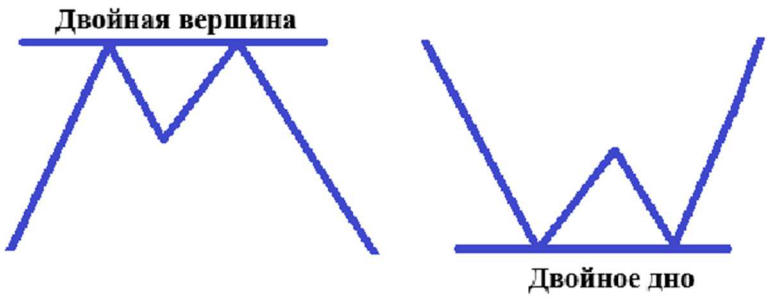

Рисунок 2 – Паттерны «Двойная вершина» и «Двойное дно»

Тройная вершина и тройное дно – похожи на двойную вершину или двойное дно. Отличие состоит в том, что эти фигуры представляют собой три последовательных вершины или основания, которые находятся на одном и том же ценовом уровне. На рисунке 3 представлены паттерны тройная вершина и тройное дно.

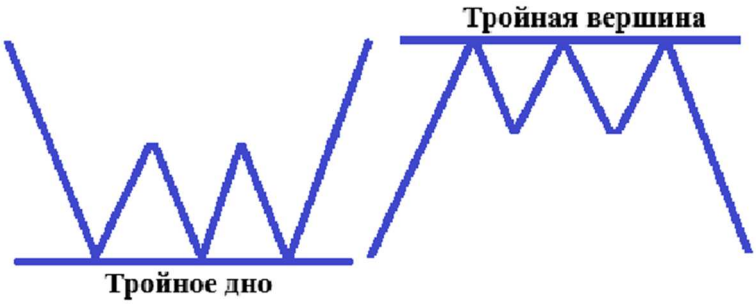

Рисунок 3 – Паттерны «Тройная вершина» и «Тройное дно»

Клин напоминает симметричный треугольник. На графике он выглядит растянутым и часто указывает на возможный разворот тренда. Существует два типа этого паттерна. Первый – восходящий, который формируется на рыночном максимуме и обычно сигнализирует о смене тенденции вниз при пробое нижней линии тренда. Второй – нисходящий клин, представляющий собой зеркальное отражение первого. Он возникает на минимуме рынка и указывает на разворот вверх, когда пробивается верхняя трендовая линия. В этом паттерне цена постепенно сужается к границам торгового канала. Когда это происходит, возникает сильное движение, часто в противоположную

сторону от текущего тренда. На рисунке 4 представлены паттерны восходящий клин и нисходящий.

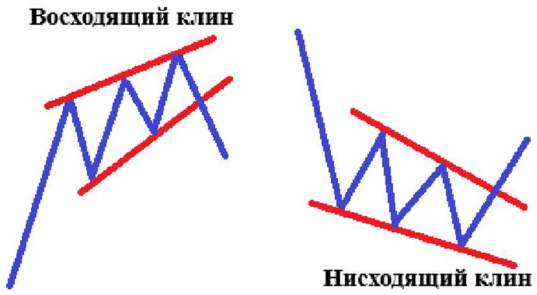

Рисунок 4 – Паттерны «Восходящий клин» и «Нисходящий клин»

Бриллиант – это крайне редкий разворотный паттерн, имеющий ромбовидную форму. Он образуется на максимумах или минимумах в процессе восходящего или нисходящего тренда. Эта графическая фигура сигнализирует о том, что текущая тенденция ослабла, и можно ожидать либо коррекции цены, либо полного разворота тренда. На рисунке 5 представлены паттерны бриллиант на падающем и растущем тренде.

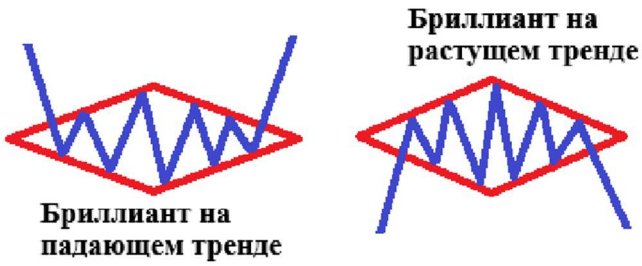

Рисунок 5 – Паттерны «Бриллиант» на падающем и растущем тренде

Паттерны продолжения – указывают на то, что текущий тренд, скорее всего, продолжится. К ним относятся: флаг; вымпел; нисходящий треугольник; восходящий треугольник; симметричный треугольник; расходящийся треугольник.

Один из самых популярных инструментов технического анализа, который предсказывает продолжение текущей рыночной тенденции – паттерн "флаг". Он состоит из двух частей: Флагшток – это сильное трендовое движение; полотнище – откат цены внутри небольшого восходящего или нисходящего канала. Флаги обычно сопровождаются резкими прорывами цены. Чтобы оценить масштаб прорыва, трейдер должен измерить глубину полотнища флага: чем более острым является угол наклона, тем сильнее будет прорыв. На основе практического опыта рекомендуется торговать по этой фигуре продолжения только после того, как произойдет прорыв. Резкие скачки цены часто связаны с паттерном "флаг", так как эта фигура представляет собой временную "передышку" участников рынка после резкого первоначального движения. Такая пауза позволяет другим трейдерам войти в позиции и спровоцировать дальнейшее движение в направлении тренда. На рисунке 6 представлены паттерны восходящий и нисходящий флаг.

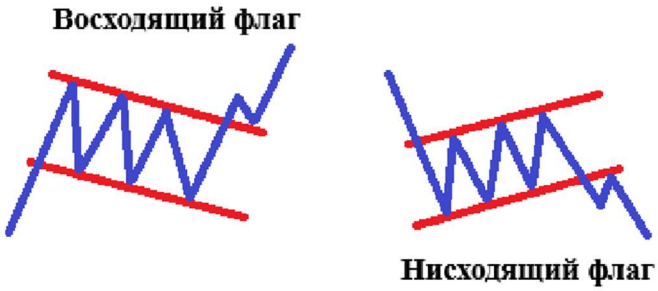

Рисунок 6 – Паттерны «Восходящий флаг» и «Нисходящий флаг»

Вымпел – паттерн напоминает флаг, но с небольшим симметричным треугольником на вершине. Основное отличие между вымпелом и флагом заключается в том, что вымпел движется в узком диапазоне и ему предшествует почти вертикальный рост цены. При формировании вымпела объемы торгов обычно уменьшаются, а затем начинают расти после прорыва. Эта фигура возникает в результате сильного трендового движения и

представляет собой аналогичную "передышку", как и у флага. На рисунке 7 представлены паттерны бычий и медвежий вымпел.

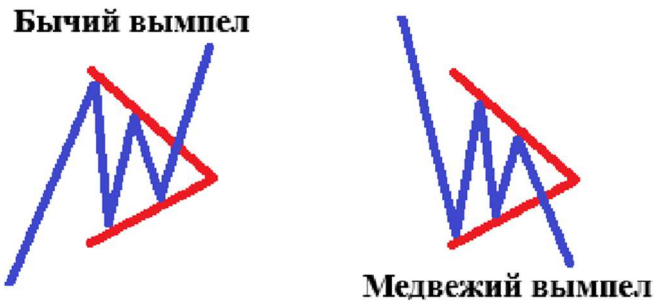

Рисунок 7 – Паттерны «Бычий вымпел» и «Медвежий вымпел»

Восходящий треугольник – эта фигура обычно наблюдается в восходящем тренде и указывает на его продолжение. Во время формирования восходящего треугольника объем торгов, как правило, снижается. Часто в направлении, где произойдет прорыв цены, объемы начинают увеличиваться. Если в рамках фигуры рост цены сопровождается увеличением объемов, то с высокой вероятностью восходящий треугольник будет пробит вверх. Определить восходящий треугольник можно по линии сопротивления, имеющую горизонтальное направление, а также по линии поддержки треугольника, она должна быть наклонена и поднята вверх. На рисунке 8 представлен паттерн восходящий треугольник.

Рисунок 8 – Паттерн «Восходящий треугольник»

Нисходящий треугольник – эта фигура схожа с "восходящим треугольником", но все характеристики являются противоположными. В этой фигуре линия поддержки треугольника имеет горизонтальное направление, а линия сопротивления треугольника наклонена и направлена вниз. На рисунке 9 представлен паттерн нисходящий треугольник.

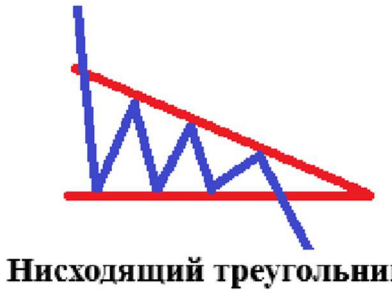

Рисунок 9 – Паттерн «Нисходящий треугольник»

Симметричный треугольник – паттерн, который не имеет четко выраженного направления, так как характеризуется балансом между спросом и предложением. Эта фигура формируется из двух сходящихся линий. Амплитуда колебаний внутри паттерна уменьшается на фоне постепенного снижения объема. Цена может двигаться как вверх, так и вниз, но с определенными нюансами. Вероятнее всего, цена пробьет уровень поддержки или сопротивления и продолжит движение в направлении тренда. Если в рамках треугольника рост цены сопровождается увеличением объема, то существует высокая вероятность, что фигура будет пробита вверх. На рисунке 10 представлены паттерны восходящий и нисходящий симметричный треугольник.

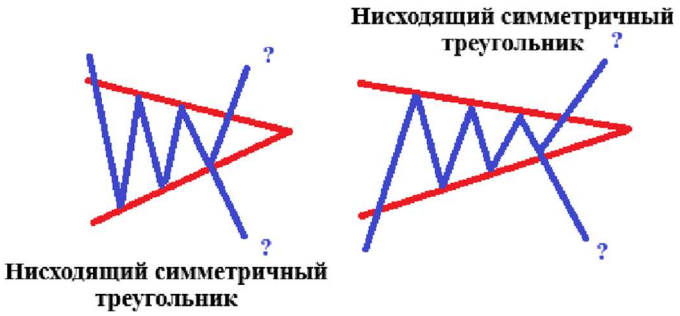

Рисунок 10 – Паттерны «Восходящий симметричный треугольник» и «Нисходящий симметричный треугольник»

Расходящийся треугольник – эта графическая формация, границы которой сходятся в одной точке в начале своего формирования и расходятся в разные стороны. Этот паттерн встречается довольно редко и не предоставляет четких сигналов для покупки или продажи, однако его классифицируют как паттерн продолжения тренда. На рисунке 11 представлены паттерны расходящегося треугольника на восходящим и нисходящим рынке.

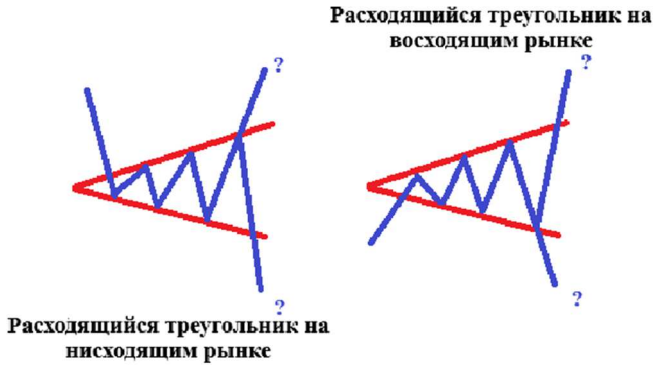

Рисунок 11 – Паттерны «Расходящийся треугольник на нисходящем рынке» и «Расходящийся треугольник на восходящем рынке»

1.4. Определение индикаторного метода

Индикаторный метод – это метод технического анализа, который использует математические формулы и алгоритмы для создания индикаторов, помогающих трейдерам оценивать состояние рынка и принимать

обоснованные решения о покупке или продаже активов. Индикаторы помогают выявлять тренды, уровни перекупленности или перепроданности, а также генерировать сигналы для входа и выхода из позиций. Индикаторы делятся на несколько типов таких как трендовые, осцилляторные, объемные, свечные.

Трендовыми индикаторами называют такие инструменты технического анализа, которые помогают трейдерам определить направление и силу текущего рыночного тренда. Они могут быть полезны для принятия решений о входе и выходе из сделок. К ним относятся: скользящие средние; полосы боллинджера; ADX; Parabolic SAR.

Скользящие это один из самых популярных и простых инструментов технического анализа, используемый для сглаживания ценовых данных и выявления направления тренда. Они помогают трейдерам и инвесторам лучше понять рыночные тенденции, устраняя шум и краткосрочные колебания цен. Основные типы скользящих средних являются SMA, EMA, WMA. Простая скользящая средняя (SMA) рассчитывается как среднее значение цен за определенный период. Например, 10-дневная SMA складывает цены закрытия за последние 10 дней и делит на 10. Экспоненциальная скользящая средняя (EMA) придает больший вес последним ценам, что делает ее более чувствительной к изменениям в цене по сравнению с SMA. Это достигается путем использования коэффициента сглаживания. Взвешенная скользящая средняя (WMA) - Похожая на EMA, но использует разные веса для каждой цены в расчетах. Более свежие данные имеют больший вес.

Полосы Боллинджера (Bollinger Bands) являются одним из самых популярным индикатор технического анализа, который используется для оценки волатильности рынка и определения потенциальных уровней поддержки и сопротивления. Индикатор состоит из трех линий: средняя, верхняя, нижняя. Средней линией является обычная скользящая средняя, которая служит центральной линией индикатора, она показывает среднюю

цену актива за определенный период. Верхняя полоса - Рассчитывается как простая скользящая средняя, плюс два стандартных отклонения, она показывает уровень, выше которого цена может считаться перекупленной. Нижняя полоса - Рассчитывается как SMA минус два стандартных отклонения, она показывает уровень, ниже которого цена может считаться перепроданной.

Индикатор ADX (Average Directional Index) – это инструмент технического анализа, который используется для измерения силы тренда на финансовых рынках. Индикатор ADX состоит из трех линий, таких как ADX, +DI, -DI. ADX является основной линией, которая измеряет силу тренда. Значения ADX колеблются от 0 до 100. При значении ниже 20 обычно указывают на слабый тренд или боковое движение, а при значении выше 40 могут свидетельствовать о сильном тренде. Значения от 20 до 40 указывают на умеренный тренд. Значения выше 40 могут свидетельствовать о сильном тренде. Линия +DI (Positive Directional Indicator) показывает силу восходящего тренда. Линия -DI (Negative Directional Indicator) показывает силу нисходящего тренда.

Parabolic SAR (Stop and Reverse) – является индикатором технического анализа, используется для определения направления тренда и потенциальных точек разворота. Он визуализируется в виде точек, расположенных над или под ценой актива. Если точки находятся ниже цены, это указывает на восходящий тренд; если выше – на нисходящий.

Осцилляторы в техническом анализе – это индикаторы, которые колеблются в определенном диапазоне значений и используются для оценки динамики цен, выявления условий перекупленности и перепроданности, а также для определения возможных точек разворота тренда. Они помогают трейдерам и аналитикам принимать решения на основе анализа ценовых движений. К ним относятся: MACD, RSI, Stochastic, Oscillator.

MACD (Moving Average Convergence Divergence) – это популярный индикатор технического анализа, который используется для определения

направления тренда, его силы и возможных точек разворота. Основные компоненты MACD являются: линия MACD, сигнальная линия, гистограмма. Линия MACD высчитывает разницу между двумя экспоненциальными скользящими средними (EMA) с разными периодами. Обычно используются 12-дневная и 26-дневная EMA. Сигнальная линия является 9-дневная EMA от MACD линии, она используется для генерации торговых сигналов. Гистограмма визуализирует силу тренда и направления, она высчитывается как разница между линией MACD и сигнальной линией.

RSI является осциллятором и колеблется в диапазоне от 0 до 100. Обычно уровни 70 и 30 используются в качестве стандартных порогов для определения перекупленности и перепроданности. Если значение выше 70 обычно указывает на то, что актив может быть перекуплен, что может предвещать коррекцию или разворот вниз. Если значение ниже 30 указывает на то, что актив может быть перепродан, что может сигнализировать о возможном развороте вверх.

Stochastic Oscillator (Стохастический осциллятор) – это индикатор технического анализа, используемый для определения условий перекупленности и перепроданности актива, а также для выявления возможных точек разворота тренда. Стохастический осциллятор основан на сравнении текущей цены закрытия актива с его ценовым диапазоном за определенный период. Основным компонентом является диапазон значений и компоненты. Диапазон осциллятора колеблется от 0 до 100. Уровни от 20 до 80 используются в качестве стандартных порогов для определения перекупленности и перепроданности. Компонентами стохастического осциллятора является %K линия и %D линия. Линия %K – это основная линия, которая показывает текущее положение цены относительно диапазона за определенный период. Линия %D – это сигнальная линия, которая представляет собой скользящее среднее %K линии.

Объемными индикаторами в технического анализе называют инструменты, которые используют данные об объеме торгов для оценки силы или слабости ценового движения. Они помогают трейдерам и аналитикам принимать решения на основе анализа активности рынка и могут служить дополнительными сигналами для входа или выхода из позиций. К ним относятся: OBV, A/D, AO, Alligator.

OBV (On-Balance Volume) – это объемный индикатор, который используется в техническом анализе для оценки силы тренда на основе объема торгов. Если цена закрытия текущего периода выше цены закрытия предыдущего периода, то к OBV добавляется объем текущего периода. Если цена закрытия текущего периода ниже цены закрытия предыдущего периода, то из OBV вычитается объем текущего периода. Если цена закрытия остается неизменной, значение OBV не изменяется.

A/D (Accumulation/Distribution Line) – это объемный индикатор, который используется в техническом анализе для оценки давления покупок и продаж на рынке. Он помогает трейдерам определить, накапливается ли актив (покупается) или распределяется (продается) в определенный период времени. Индикатор A/D рассчитывается на основе объема и положения цены закрытия относительно диапазона цен за день (максимум и минимум). Индикатор отображается в виде линии на графике, которая колеблется вверх и вниз в зависимости от изменений объема и цен. Если линия A/D растет вместе с ценой актива, это подтверждает силу восходящего тренда. Если линия A/D падает вместе с ценой, это подтверждает силу нисходящего тренда.

AO (Awesome Oscillator) – это индикатор технического предназначенный для измерения импульса и силы тренда, а также для выявления возможных разворотов на рынке. AO основан на разнице между двумя простыми скользящими средними (SMA) с разными периодами и помогает трейдерам оценить текущую рыночную динамику. AO рассчитывается как разница между 34-периодной и и 5-периодной простой

скользящей средней, взятой по средним ценам (обычно это средняя цена за день, которая рассчитывается как (максимум + минимум) / 2). Если АО находится выше нуля, это указывает на восходящий импульс. Если АО находится ниже нуля, это указывает на нисходящий импульс.

Alligator – это индикатор предназначенный для определения направления тренда и его силы, а также для выявления возможных точек входа и выхода из рынка. Индикатор Alligator состоит из трех линий, каждая из которых представляет собой скользящую среднюю с разными периодами и смещениями. Линия Jaw (Челюсть) – это 13-периодная простая скользящая средняя (SMA), смещенная на 8 баров в будущее, обозначается зеленым цветом. Линия Teeth (Зубы) – Это 8-периодная простая скользящая средняя (SMA), смещенная на 5 баров в будущее, обозначается красным цветом. Линия Lips (Губы) – Это 5-периодная простая скользящая средняя (SMA), смещенная на 3 бара в будущее, обозначается синим цветом.

1.5. Определение свечного анализа

Свечной анализ – это метод технического анализа, основанный на использовании свечных графиков для оценки ценового движения актива за определенный период времени. Каждая свеча на графике отображает четыре ключевых ценовых параметра: цену открытия, цену закрытия, максимальную и минимальную цену за заданный временной интервал. Если цена закрытия выше цены открытия, тело свечи обычно окрашивается в зеленый (или белый) цвет, что указывает на бычий тренд. Если цена закрытия ниже цены открытия, тело свечи окрашивается в красный (или черный) цвет, что указывает на медвежий тренд. Тени представляют собой линии, которые выходят за пределы тела свечи и показывают максимальные и минимальные цены за период: верхняя тень показывает максимальную цену, нижняя тень показывает минимальную цену. Свечной анализ включает в себя изучение

различных паттернов, которые могут сигнализировать о возможных разворотах или продолжениях тренда. Некоторые из наиболее распространенных паттернов: молот, повешенный, доджи, поглощение, три белых солдата, три черных ворона, падающая звезда, утренняя звезда, вечерняя звезда.

Молот (Hammer) – Свеча с коротким телом и длинной нижней тенью, которая в два или три раза длиннее тела. Указывает на возможный разворот вверх после нисходящего тренда. На рисунке 12 представлен свечной паттерн - молот.

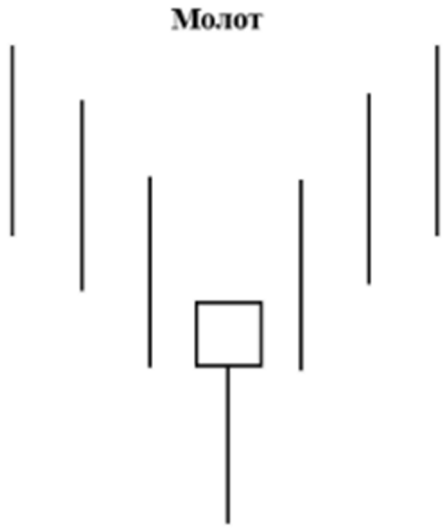

Рисунок 12 – Свечной паттерн «Молот»

Повешенный (Hanging Man) – Похож на молот, но появляется после восходящего тренда. Может сигнализировать о возможном развороте вниз. На рисунке 13 представлен свечной паттерн - повешенный.

Рисунок 13 – Свечной паттерн «Повешенный»

Доджи (Doji) - Свеча с очень коротким телом, где цена открытия и закрытия почти равны. Указывает на неопределенность на рынке; может предвещать разворот. На рисунке 14 представлен свечной паттерн - доджи.

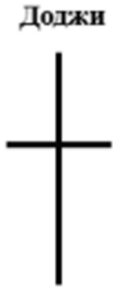

Рисунок 14 – Свечной паттерн «Доджи»

Поглощение (Engulfing). Данный паттерн делится на три вида. Первый вид это бычье поглощение, при нем вторая свеча полностью поглощает первую и закрывается выше. При медвежьем поглощении – вторая свеча полностью поглощает первую и закрывается ниже. Третий вид это – сигнал, он указывает на возможный разворот в сторону второй свечи. На рисунке 15 представлен свечной паттерн - поглощение.

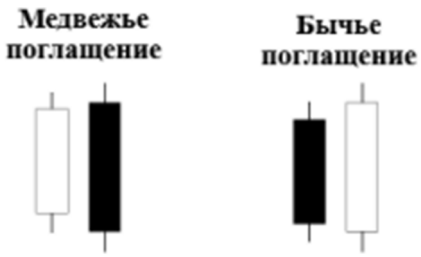

Рисунок 15 – Свечной паттерн «Поглощение»

Три белых солдата (Three White Soldiers) – это три последовательные бычьи свечи с растущими телами. Указывает на сильный восходящий тренд. На рисунке 16 представлен свечной паттерн - три белых солдата.

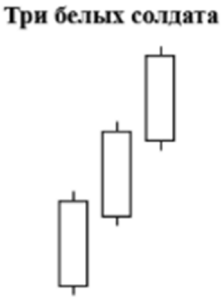

Рисунок 16 – Свечной паттерн «Три белых солдата»

Падающая звезда (Shooting Star) – это свеча с коротким телом и длинной верхней тенью, которая появляется после восходящего тренда, она может указывать на возможный разворот вниз. На рисунке 17 представлен свечной паттерн -падающая звезда.

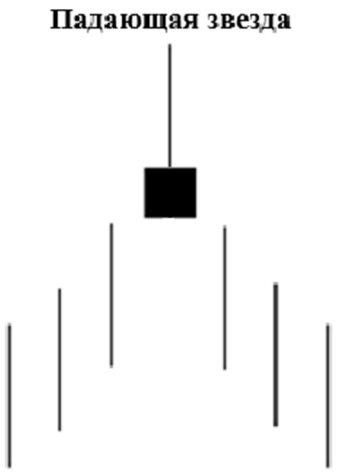

Рисунок 17 – Свечной паттерн «Падающая звезда»

Три черных ворона (Three Black Crows) – это три последовательных медвежьих свечи с падающими телами, они указывают на сильный нисходящий тренд. На рисунке 18 представлен свечной паттерн - три черных ворона.

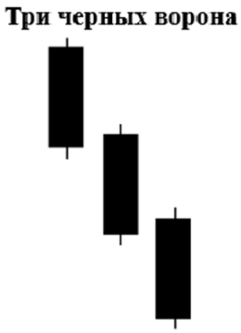

Рисунок 18 – Свечной паттерн «Три чёрных ворона»

Утренняя звезда (Morning Star) – паттерн, состоящий из трех свечей: медвежья свеча, маленькая свеча (доджи или маленькое тело) и бычья свеча. Данный паттерн может указывать на возможный разворот вверх после нисходящего тренда. На рисунке 19 представлен свечной паттерн - утренняя звезда.

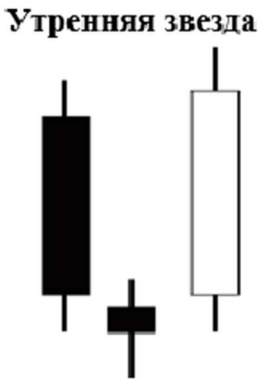

Рисунок 19 – Свечной паттерн «Утренняя звезда»

Вечерняя звезда (Evening Star) – это паттерн, состоящий из трех свечей: бычья свеча, маленькая свеча, медвежья свеча. Он может указывать на возможный разворот вниз после восходящего тренда. На рисунке 20 представлен свечной паттерн - вечерняя звезда.

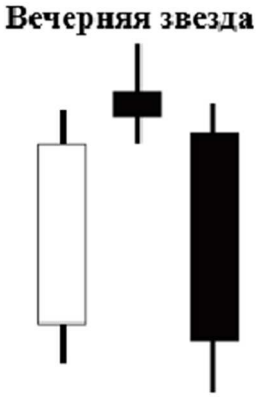

Рисунок 20 – Свечной паттерн «Вечерняя звезда»

1.6. Определение объёмного анализа

Объемный анализ – это метод технического анализа, который фокусируется на изучении объема торгов как важного индикатора для оценки силы и направления ценовых движений. Объем отражает количество акций, контрактов или других активов, которые были куплены и проданы за определенный период времени. Анализ объема помогает трейдерам лучше

понять динамику рынка и выявить ключевые моменты для принятия торговых решений. Объем торгов показывает, сколько единиц актива было куплено и продано в течение определенного временного интервала. Высокий объем может указывать на сильный интерес к активу, тогда как низкий объем может свидетельствовать о слабом интересе и неопределенности на рынке. Объем часто рассматривается в контексте изменения цен.

Рост цены с увеличением объема – это может сигнализировать о сильном бычьем тренде, подтверждая, что покупатели активно поддерживают рост.

Рост цены при низком объеме – это может указывать на слабость тренда и возможный разворот.

Падение цены с увеличением объема – это может свидетельствовать о сильном медвежьем тренде.

Падение цены при низком объеме – это может указывать на отсутствие интереса со стороны продавцов.

Кроме этого, используют различные индикаторы, которые помогают анализировать объем торгов. Например: Volume, A/d, CMF.

Объем (Volume) – это простой индикатор, показывающий количество проданных единиц актива за определенный период.

Индикатор накопления/распределения (A/D) – это индикатор учитывающий как объем, так и цену закрытия, чтобы определить давление покупок или продаж.

Chaikin Chaikin Money Flow (CMF) – это индикатор измеряющий объем по сравнению с ценами закрытия за определенный период, помогая определить общий поток денег в актив.

1.7. Применение технического анализа

Технический анализ (ТА) используется многими трейдерами не только для прогнозирования изменения цен криптовалют, но и для других

классических биржевых инструментов таких как акций, индексов, фьючерсов и так далее. Целью технического анализа является составление прогноза изменения цены актива на основании полученных данных за прошедшие периоды с учетом факторов, сопутствующих росту или падению стоимости. Берутся во внимание индикаторы, свечные модели, паттерны, уровни поддержки и сопротивления. ТА криптовалют проводится по стандартной методике. Но с учетом особых правил торговли данным активом оценка информации может быть иной, а от нее зависит решение, которое будет принято трейдером. Между техническим анализом для акций и криптовалют существуют различия. Основные различия заключаются в таких аспектах как волатильность, объем торговли, доступности данных, регулировании и использовании индикаторов.

Волатильность: Криптовалюты, как правило, более волатильны по сравнению с акциями. Это означает, что ценовые колебания на рынке криптовалют могут происходить гораздо быстрее и с большими амплитудами, что требует от трейдеров более гибких и адаптивных стратегий технического анализа.

Объем торговли: Рынок акций обычно имеет более высокие объемы торговли и более стабильные ликвидности, чем рынок криптовалют. Это может влиять на точность сигналов технического анализа, так как на менее ликвидных рынках может происходить резкое изменение цен при относительно небольших объемах торгов.

Доступность данных: Данные о ценах и объемах для акций часто более доступны и структурированы через официальные биржи и финансовые отчеты. В то время как данные о криптовалютах могут быть менее стандартизированы из-за большого количества различных бирж и платформ, что может усложнять анализ.

Регулирование: Рынок акций находится под строгим регулированием со стороны финансовых органов, что может влиять на его поведение и

предсказуемость. Криптовалютный рынок менее регулируем и может быть подвержен манипуляциям, что также должно учитываться при проведении технического анализа.

Использование индикаторов: хотя многие индикаторы технического анализа могут применяться как для акций, так и для криптовалют, некоторые из них могут давать разные результаты из-за специфики каждого рынка. Например, индикаторы, основанные на объеме, могут работать иначе на менее ликвидном рынке криптовалют.

Технический анализ на финансовых рынках, включая рынок криптовалют, широко используется трейдерами для принятия торговых решений. Однако точность и эффективность этого метода могут варьироваться в зависимости от различных факторов.

Точность сигналов, генерируемых техническим анализом, может варьироваться от 65% до 85%. Это означает, что трейдеры могут ожидать успешных сделок в половине или более случаев при использовании технического анализа.

На рынке криптовалют присуща высокая волатильность, которая может снижать точность прогнозов. Резкие колебания цен могут приводить к тому, что даже хорошо обоснованные сигналы могут не сработать.

Технический анализ часто комбинируется с фундаментальным анализом для повышения общей точности прогнозов. Это позволяет учитывать не только исторические данные, но и текущие новости и события.

Хотя точные цифры о процентной точности технического анализа могут варьироваться, он остается полезным инструментом для трейдеров на финансовых рынках. Понимание его возможностей и ограничений помогает трейдерам принимать более обоснованные решения и повышать эффективность своих торговых стратегий.

1.8. Процесс создания приложения по техническому анализу

Создание приложения для технического анализа криптовалюты представляет собой возможность трейдерам делать более обоснованные принятие решения. В этом приложении пользователи смогут получить доступ в реальном времени к разнообразным инструментам анализа, включая индикаторы, графики и аналитические отчеты, что позволит им более уверенно ориентироваться в волатильном мире криптовалют. В этом проекте будет стремление создать интуитивно понятный интерфейс, который будет доступен как новичкам, так и опытным трейдерам. Цель проекта – предоставить пользователям сигнал к покупке или к продаже актива, используя инструменты технического анализа.

Получение данных в режиме реальном времени является одним из самых главных факторов влияющих на прибыльность сделок. В данном приложении сбор данных с криптобирж будет осуществлен путем парсинга. Парсинг – это процесс извлечения данных из различных источников, таких как веб-страницы, документы или базы данных. Этот метод часто используется для сбора информации в структурированном виде, что позволяет анализировать и обрабатывать данные более эффективно. Парсинг может применяться в различных областях, включая веб-скрейпинг, анализ данных и автоматизацию задач. Существует несколько видов парсинга, мы рассмотрим два самых распространенных, веб-парсинг и парсинг с использованием API.

Веб-парсинг – это процесс автоматического извлечения данных с веб-страниц с помощью специальных программ или скриптов. Он позволяет получать структурированную информацию из HTML-кода сайтов для последующего анализа, обработки или использования в различных приложениях. Для парсинга веб-страниц потребуется библиотека для работы с HTML-кодом (Beautiful Soup или Scrapy для Python, Cheerio для JavaScript).

Пример кода на Java Script для парсинга цены Bitcoin с веб-страницы Binance приведен на рисунке 21.

```
const axios = require('axios');
const cheerio = require('cheerio');

async function getBitcoinPrice() {
  const url = https://www.binance.com/ru;

  try {
    // Выполняем GET-запрос к указанному URL
    const { data } = await axios.get(url);

    // Загружаем HTML-код в cheerio
    const $ = cheerio.load(data);

    // Предполагаем, что цена находится в элементе с классом 'price'
    const priceElement = $('.price'); // Замените на правильный селектор
    const price = priceElement.text().trim();

    return price;
  } catch (error) {
    console.error('Ошибка при получении данных:', error);
  }
}

// Получение цены Bitcoin
getBitcoinPrice().then(price => {
  if (price) {
    console.log(`Цена Bitcoin: ${price}`);
  }
});
```

Рисунок 21 – Пример кода на языке программирования JavaScript для парсинга цены криптовалюты Bitcoin с веб-страницы торговой площадки Binance

В данном коде на JavaScript, мы получает текущую цену Биткойна (BTC) по отношению к USDT с использованием веб-парсинга. Мы импортируем axios для выполнения HTTP-запросов и cheerio для парсинга HTML. Асинхронная функция getBitcoinPrice выполняет GET-запрос к указанному URL. После получения HTML-кода мы загружаем его в cheerio, далее происходит поиск элемента с классом (. price) и извлекаем текст. После функция возвращает цену Bitcoin.

Многие криптобиржи предоставляют API (интерфейсы программирования приложений), которые позволяют получать данные в структурированном формате (например, JSON). Использование API предпочтительнее, так как это более надежный и легкий способ получения данных.

Пример кода на Java Script для получения данных о ценах с Binance приведен на рисунке 22.

```
async function getBinancePrice(symbol) {
    const url = `https://api.binance.com/api/v3/ticker/price?symbol=${symbol}`;

    try {
        // Выполняем GET-запрос к API Binance
        const response = await fetch(url);

        // Проверяем, успешен ли ответ
        if (!response.ok) {
            throw new Error(`HTTP error! status: ${response.status}`);
        }

        // Получаем данные в формате JSON
        const data = await response.json();

        // Возвращаем цену
        return data.price;
    } catch (error) {
        console.error('Error fetching price:', error);
    }
}

// Пример использования функции
(async () => {
    const symbol = 'BTCUSDT';  // Пара для получения цены (Биткойн к USDT)
    const price = await getBinancePrice(symbol);

    if (price) {
        console.log(`Цена Bitcoin: ${price} USDT`);
    }
})();
```

Рисунок 22 – Код на языке программирования JavaScript
для получения данных о ценах криптовалют с торговой площадки Binance

В данном коде на JavaScript, мы получает текущую цену Биткойна (BTC) по отношению к USDT с использованием API Binance. Мы используем встроенный объект (fetch) для выполнения HTTP-запросов. Асинхронная функция (getBinancePrice) принимает символ торговой пары (например, 'BTCUSDT') и формирует URL для запроса. Если запрос успешен, мы

извлекаем данные в формате JSON и возвращаем цену. Если ответ не успешен, получаем сообщение об ошибки.

В качестве возможных бирж для парсинга можно выделить следующие: Binance, Coinbase, Kraken, Huobi и Gate.io.

Binance – Одна из крупнейших криптобирж в мире, предоставляет API для получения данных о ценах, объемах торгов и других параметрах.

Coinbase – Популярная платформа, предлагающая API для доступа к рыночным данным и информации о пользователях.

Kraken – Биржа, известная своей безопасностью и широким выбором криптовалют. Также предоставляет API для получения данных о торгах.

Huobi – Крупная криптобиржа с обширным набором инструментов и API для получения данных о торгах.

Gate.io – Биржа с большим количеством альткойнов и API для получения информации о ценах и объемах торгов.

Кроме криптобирж можно собирать данные и с других ресурсов, например с платформ, такие как: CoinGecko, CoinMarketCap, TradingView, Messari, CoinCheckup, CoinPaprika.

CoinGecko – эта платформа предлагает обширную информацию о криптовалютах, включая цены, объемы торгов и рыночную капитализацию. CoinGecko также предоставляет данные о разработке проектов и активности в социальных сетях.

CoinMarketCap – эта платформа фокусируется на событиях в мире криптовалют, таких как обновления проектов, форки и другие важные даты. Она помогает трейдерам учитывать влияние событий на рынок.

TradingView – эта платформа предлагает мощные инструменты для технического анализа финансовых рынков, включая интерактивные графики, широкий выбор индикаторов и возможность создания пользовательских скриптов на языке Pine Script. TradingView также предоставляет социальные

функции, позволяя пользователям делиться своими идеями и анализами, а также следить за активностью других трейдеров.

Messari – данная платформа предоставляет аналитическую информацию о криптовалютах, включая рыночные данные, новости и исследования. Платформа ориентирована на более глубокий анализ и понимание проектов.

CoinCheckup – данная платформа предоставляет подробную информацию о криптовалютах, включая технический анализ, новости и прогнозы. Платформа также предлагает инструменты для анализа проектов.

CoinPaprika – данная платформа предлагает информацию о ценах, объемах торгов и рыночной капитализации различных криптовалют, а также позволяет пользователям отслеживать свои портфели.

1.9. Обзор аналогов

С развитием технологий и увеличением популярности криптовалют, на рынке появились многочисленные приложения, которые помогают трейдерам в прогнозировании цен на основе технического анализа. Эти приложения используют алгоритмы и модели машинного обучения для обработки больших объемов данных, анализа исторических трендов и генерации торговых сигналов. Ниже представлены примеры приложений, помогающие трейдерам в принятие решении.

AlgosOne – это платформа для автоматической торговли криптовалютами, которая предоставляет пользователям возможность разрабатывать, тестировать и внедрять собственные алгоритмические торговые стратегии. Она предназначена как для новичков, так и для опытных трейдеров, предлагая инструменты для оптимизации процесса торговли.

Особенности AlgosOne включают интуитивно понятный интерфейс, позволяющий легко создавать и настраивать торговые стратегии без необходимости программирования. Платформа обеспечивает автоматизацию

торговых процессов с помощью заранее заданных алгоритмов и стратегий, что помогает минимизировать эмоциональные решения. Также она предоставляет возможность тестирования стратегий на исторических данных, что позволяет оценить их эффективность перед применением в реальной торговле. AlgosOne интегрируется с несколькими крупными криптобиржами, что дает пользователям возможность управлять активами из одного интерфейса. Работая в облаке, платформа обеспечивает круглосуточную торговлю без необходимости держать компьютер включенным. Кроме того, она предлагает сообщество трейдеров для обмена стратегиями и получения советов от более опытных пользователей.

Плюсы AlgosOne включают легкость в использовании благодаря интуитивному интерфейсу, возможность автоматизации торговли для снижения эмоциональной нагрузки и повышения эффективности, а также тестирование стратегий на исторических данных для снижения рисков при переходе к реальной торговле. Облачная работа платформы дает гибкость и мобильность, а широкий набор инструментов позволяет анализировать и оптимизировать торговые стратегии.

Однако у AlgosOne есть и минусы. Алгоритмические стратегии могут не показывать хороших результатов в условиях высокой волатильности или нестабильности рынка. Несмотря на простоту использования, пользователям все равно нужны базовые знания о техническом анализе для эффективного применения функций платформы. Также платформа может иметь ограниченную поддержку некоторых криптобирж, что может ограничить возможности отдельных пользователей.

Algoriz – это платформа, предназначенная для автоматизации торговли криптовалютами с использованием алгоритмических стратегий. Она предлагает пользователям возможность разрабатывать, тестировать и внедрять собственные торговые стратегии без необходимости глубоких знаний в программировании. Рассмотрим подробно.

Особенности Algoriz включают интуитивно понятный интерфейс, который позволяет легко создавать и настраивать торговые стратегии с помощью визуального редактора. Платформа обеспечивает автоматизацию торговых процессов, позволяя трейдерам использовать заранее заданные алгоритмы и стратегии. Также Algoriz предоставляет возможность тестирования торговых стратегий на исторических данных, что помогает оценить их эффективность перед использованием в реальной торговле. Кроме того, платформа поддерживает взаимодействие с сообществом трейдеров и обмен стратегиями, что способствует обучению и развитию навыков. Работая в облаке, Algoriz позволяет торговать 24/7 без необходимости держать компьютер включенным.

Плюсы Algoriz включают легкость в использовании благодаря интуитивному интерфейсу, широкий набор функций для автоматизации торговли, тестирования стратегий и обмена опытом с другими трейдерами. Облачная работа платформы дает гибкость и мобильность, а возможность настройки стратегий под индивидуальные потребности делает ее универсальным инструментом.

Однако у Algoriz есть и минусы. Алгоритмические стратегии могут не показывать хороших результатов в условиях высокой волатильности или нестабильности рынка. Несмотря на простоту использования, пользователям все равно нужны базовые знания о техническом анализе для эффективного применения платформы. Также возможны ошибки или баги в программном обеспечении, что может повлиять на работу торговых стратегий. Кроме того, поддержка некоторых криптобирж может быть ограничена, что сужает возможности для трейдеров.

Zignaly – это платформа для автоматической торговли криптовалютами, которая предлагает пользователям возможность копировать сделки успешных трейдеров и использовать алгоритмические стратегии для оптимизации своих торговых процессов. Рассмотрим подробно. Особенности Zignaly включают

функцию копи-трейдинга, позволяющую пользователям следовать за опытными трейдерами и автоматически копировать их сделки. Это особенно полезно для новичков, желающих учиться у более опытных участников рынка. Платформа также предоставляет инструменты для автоматизации торговли с помощью настройки торговых ботов, которые работают на основе заранее заданных стратегий. Zignaly интегрируется с несколькими крупными криптобиржами, такими как Binance, KuCoin и другими, что позволяет управлять активами из одного интерфейса. Работая в облаке, платформа обеспечивает круглосуточную торговлю без необходимости держать компьютер включенным. Кроме того, Zignaly предоставляет аналитические данные о производительности стратегий и сделок, что помогает принимать обоснованные решения. Интуитивно понятный интерфейс делает использование платформы доступным для пользователей с разным уровнем опыта.

Плюсы Zignaly включают простоту в использовании благодаря удобному интерфейсу, возможность следовать за успешными трейдерами через копи-трейдинг, автоматизацию торговли для снижения эмоциональной нагрузки и повышения эффективности, а также доступ к аналитике для оценки результатов. Облачное решение обеспечивает гибкость и мобильность.

Минусы платформы связаны с зависимостью от выбранных трейдеров: успех копи-трейдинга зависит от их решений, и неправильные действия могут привести к убыткам. Также пользователям необходимо иметь базовые знания о рынке криптовалют и техническом анализе для более эффективного использования платформы. Возможны программные ошибки или баги, которые могут повлиять на работу стратегий. Кроме того, некоторые функции могут требовать дополнительных затрат или комиссий, что снижает общую прибыльность торговли.

Проанализировав программы, выделив их особенности, плюсы, минусы была составлена таблица. Результаты сравнительного анализа сведены в таблицу 1.

Таблица 1 – Сравнение программ по анализу криптовалют.

Программа / Преимущество	Генерация сигналов	Собственные стратегии	Автоматическое исполнение сделок	Оповещение о сигнале на покупку/продажу	Поддержка различных форматов обучения	Генерация сигналов с различными весами для каждого индикатора
Algoriz	+	+	+	+	–	–
AlgosOne	+	-	+	+	-	–
Zignaly	+	+	+	+	–	–

В таблице 1 видно, что различные программы предлагают разные преимущества в области генерации сигналов и автоматизации торговли. Все три платформы, Algoriz, AlgosOne и Zignaly, поддерживают генерацию

сигналов и автоматическое исполнение сделок, что делает их подходящими для трейдеров, стремящихся к автоматизации своих стратегий. Algoriz и Zignaly также предлагают возможность создания собственных стратегий, что может быть важным для более опытных пользователей. Однако ни одна из платформ не поддерживает генерацию сигналов с различными весами для каждого индикатора, что может ограничивать гибкость в настройке торговых стратегий. Кроме того, ни одна из платформ не предлагает поддержку различных форматов обучения, что может затруднить процесс адаптации пользователей к новым условиям рынка.

1.10. Выбор языков программирования

Фронтенд веб-приложения будет создан на языке html с использованием css для стилизации.

Для бэкэнда требовалось выбрать язык, который оптимально подходит для Api-парсинга, выбор был осуществлен из таких языков как: JavaScript, C# и Python.

Результаты сравнительного анализа сведены в таблицу 2.

Таблица 2 – Сравнение языков программирования

Критерий	JavaScript	Python	C#
Простота использования	Очень высокая: богатый набор инструментов, легкость работы с DOM и асинхронностью	Очень высокая: простота синтаксиса, богатая экосистема	Высокая: мощные библиотеки, интеграция с Windows и .NET платформой

Продолжение таблицы 2

Критерий	JavaScript	Python	C#
Производительность	Отличная: высокая скорость выполнения, особенно в асинхронных задачах	Средняя: зависит от реализации, подходит для большинства задач	Хорошая: стабильная и надежная обработка больших данных
Библиотеки/инструменты	Puppeteer, Cheerio – мощные инструменты для парсинга HTML/XML и автоматизации браузера	BeautifulSoup, Scrapy – популярные библиотеки для парсинга	HtmlAgilityPack, AngleSharp – мощные инструменты для парсинга HTML/XML
Параллельность/Масштабируемость	Параллельность/Масштабируемость	Средняя: есть возможности многопоточности, но требуют дополнительных настроек	Высокая: встроенная поддержка многопоточности
Обучаемость и документация	Очень высокая благодаря большому сообществу и богатой документации	Очень высокая благодаря большому сообществу и простоте синтаксиса	Хорошо документирован, особенно в контексте .NET Framework/.NET Core

Преимущество JavaScript в высокой скорости выполнения, особенно при асинхронной обработке и автоматизации браузеров с помощью Puppeteer или Cheerio. Это делает его идеальным для парсинга динамических сайтов и работы с API, где важна скорость и возможность взаимодействия с веб-страницами в реальном времени. Благодаря кроссплатформенности и богатому инструментарию, JavaScript отлично подходит для масштабных проектов по сбору данных.

Python обладает простым синтаксисом и большим количеством библиотек (BeautifulSoup, Scrapy), что значительно ускоряет разработку парсеров. Недостаток – меньшая производительность по сравнению с другими языками. Python отлично подходит для быстрого прототипирования и обработки структурированных данных.

C# является языком высокого уровня, обладает мощными библиотеками (HtmlAgilityPack) и высокой производительностью, что делает его хорошим выбором для крупных корпоративных решений. Он особенно эффективен при парсинге статичных HTML/XML документов и интеграции с системами на базе .NET. Для работы с API C# обеспечивает стабильность и надежность, а его многопоточность ускоряет обработку больших объемов данных.

Вывод: после проведения сравнительного анализа было принято решение остановиться на языке JavaScript, поскольку его возможностей достаточно для выполнения поставленной задачи, что упростит процесс разработки и последующее сопровождение приложения.

1.11. Выводы по главе

В рамках проведённого аналитического обзора была всесторонне исследована природа и сущность технического анализа как одного из ключевых методов прогнозирования поведения финансовых инструментов. Особое внимание было уделено классификации существующих подходов к

техническому анализу, изучению их методологических основ, а также рассмотрению наиболее широко применяемых инструментов и индикаторов. В процессе анализа были выявлены как сильные стороны, так и ряд существенных ограничений применения технического анализа, особенно в условиях высокой волатильности, характерной для криптовалютного рынка.

Отдельное внимание в работе было направлено на исследование специфики функционирования криптовалютных активов и особенностей применения технического анализа к их торговле. Это позволило не только глубже понять поведенческие закономерности данного сегмента рынка, но и сформировать требования к разработке программного инструментария, способного эффективно работать в столь динамичной среде.

В ходе исследования был произведён обоснованный выбор стека технологий для реализации веб-приложения. В качестве основных языков программирования были выбраны HTML и JavaScript, что обеспечило необходимую гибкость при построении пользовательского интерфейса и реализации функциональности сбора, обработки и визуализации данных. Применение JavaScript позволило организовать динамическое взаимодействие с внешними веб-источниками и в значительной степени автоматизировать процессы извлечения и анализа информации в режиме реального времени.

Дополнительно был проведён сравнительный обзор существующих решений в области автоматизированного технического анализа и генерации торговых сигналов. Анализ функциональных возможностей аналогичных инструментов позволил не только выявить наиболее удачные решения, но и сформировать представление о конкурентных преимуществах разрабатываемого приложения, а также определить перспективные направления его развития.

2. ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ ВЕБ-ПРИЛОЖЕНИЯ

2.1. Определение требований

Перед проектированием архитектуры веб-приложения необходимо определить функциональные и нефункциональные требования в соответствии с заданием.

Функциональные требования:

– интеграция с API криптобирж для получения актуальных данных;

– индикаторы (скользящие средние, RSI, MACD и др.);

– динамические графики цен с возможностью настройки временных интервалов;

– предоставление обучающих материалов по техническому анализу и торговым стратегиям;

– возможность создания пользовательских торговых стратегий.

Нефункциональные требования:

– быстрая загрузка страниц и минимальная задержка при взаимодействии с графиками и визуализациями данных;

– интуитивно понятный интерфейс с адаптивным дизайном;

– возможность добавления новых инструментов анализа и образовательных материалов без значительных изменений в архитектуре приложения.

2.2. Математическое описание технических индикаторов

Технические индикаторы играют ключевую роль в процессе анализа финансовых рынков, предоставляя трейдерам и инвесторам инструменты для оценки текущих рыночных условий и прогнозирования будущих ценовых двиний. Эти индикаторы основаны на математических расчетах, которые

используют исторические данные о ценах и объемах торгов для выявления трендов, уровней поддержки и сопротивления, а также других значимых паттернов. Ниже будут рассмотрены основные математические модели и формулы, лежащие в основе популярных технических индикаторов, такие как: SMA, EMA, WMA, MACD, RSI, BB, Williams %R, CCI, PSAR, AD, FI, Stoch RSI, MFI, SMMA, Ichimoku Cloud.

SMA (Simple Moving Average) – это простая скользящая средняя, один из наиболее распространенных технических индикаторов, используемых в анализе финансовых рынков. Она помогает сгладить ценовые данные, чтобы выявить тренды и направления движения цены. Для расчета индикатора представлена формула 1, где A1 – самая последняя цена за взятый период времени; n – количество периодов (например, дней), за которые рассчитывается средняя.

$$\boldsymbol{SMA = \frac{A1 + A2 + \cdots + An}{n}}. \quad (1)$$

EMA (Exponential Moving Average) – это экспоненциальная скользящая средняя, которая является одним из популярных технических индикаторов, используемых в анализе финансовых рынков. EMA придает больший вес более недавним ценам, что позволяет ей быстрее реагировать на изменения в ценах по сравнению с простой скользящей средней (SMA). Для расчета индикатора представлена формула 2, где Pt – цена актива в текущий момент времени; EMAt−1 – значение EMA за предыдущий период; k=N+12 – коэффициент сглаживания, где N – количество периодов.

$$\boldsymbol{EMAt = kPt + (1 - k)(EMAt - 1)}. \quad (2)$$

WMA (Weighted Moving Average) – это взвешенная скользящая средняя, которая придает разный вес ценам в зависимости от их положения во времени. В отличие от простой скользящей средней (SMA), где все значения имеют одинаковый вес, WMA позволяет более значительное влияние на итоговое значение более недавним ценам. Для расчета индикатора представлена формула 3:

$$\boldsymbol{WMA} = \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}}(\boldsymbol{wi} * \boldsymbol{pi})}{\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{wi}}, \tag{3}$$

где Pi – цена актива в момент времени i; Wi – вес, присвоенный цене Pi; N – количество периодов.

MACD (Moving Average Convergence Divergence) – для расчёта линейного MACD из скользящей средней цены (обычно берётся экспоненциальная скользящая средняя с меньшим периодом) вычитается экспоненциальная скользящая средняя с большим периодом. В большинстве случаев полученный результат сглаживают при помощи экспоненциальной скользящей средней (EMA). Для расчета индикатора представлены формулы 4 и 5, где $EMA_s(P)$ – экспоненциальная скользящая средняя с коротким периодом от цены; $EMA_l(P)$ – экспоненциальная скользящая средняя с длинным периодом от цены; $EMA_a(P)$ – сглаживающая скользящая средняя с коротким периодом от разницы двух остальных скользящих; P – цена, обычно берётся цена закрытия периода Close, но возможны и другие варианты (Open, High, Low, Close, Median Price, Typical Price и т. д.).

$$\boldsymbol{Signal} = \boldsymbol{EMAa}\big(\boldsymbol{EMAs}(\boldsymbol{P}) - \boldsymbol{EMAl}(\boldsymbol{P})\big). \tag{4}$$

$$\boldsymbol{MACD} = \boldsymbol{EMAs}(\boldsymbol{P}) - \boldsymbol{EMAl}(\boldsymbol{P}). \tag{5}$$

RSI (Relative Strength Index) – это индекс относительной силы, который является популярным техническим индикатором, используемым в анализе финансовых рынков для оценки состояния перекупленности или перепроданности актива. Для расчета индикатора представлена формула 6:

$$\boldsymbol{RSI = 100 - \frac{100}{1 + RS}}, \tag{6}$$

где RS (относительная сила) – определяется как среднее значение приростов цен за определенный период, деленное на среднее значение убытков за тот же период.

BB (Bollinger Bands) – это индикатор, разработанный Джоном Боллинджером, который используется в техническом анализе для оценки волатильности рынка и определения уровней перекупленности и перепроданности актива. Индикатор состоит из трех линий:

– средняя скользящая (SMA): обычно используется 20-дневная простая скользящая средняя (SMA), но трейдеры могут настраивать этот период в зависимости от своих стратегий;

– верхняя полоса: рассчитывается как SMA плюс два стандартных отклонения;

– нижняя полоса: рассчитывается как SMA минус два стандартных отклонения.

Williams %R (Williams Percent Range) – это осциллятор, разработанный Ларри Уильямсом, который используется в техническом анализе для определения уровней перекупленности и перепроданности актива. Индикатор колеблется в диапазоне от -100 до 0 и помогает трейдерам выявлять потенциальные точки разворота на рынке. Для расчета индикатора представлена формула 7:

$$\%\boldsymbol{R} = \frac{\boldsymbol{highNdays - closetoday}}{\boldsymbol{highNdays - lowNdays}} * \boldsymbol{-100}, \quad (7)$$

где highNdays – максимальная цена за выбранный период (обычно 14 дней); lowNdays – минимальная цена за тот же период; closetoday – цена закрытия, текущего период.

CCI (Commodity Channel Index) – это технический индикатор, разработанный Дональдом Ламбертом, который используется для оценки уровня перекупленности и перепроданности актива. Хотя изначально он был создан для анализа товарных рынков, CCI также широко применяется на фондовых и валютных рынках. Для расчета индикатора представлена формула 8, где Typical Price (Типичная цена) – цена актива в определенный день в рассматриваемом периоде времени; Simple Moving Average (Простая скользящая средняя) – арифметическое среднее цены актива за определенный период времени; Mean Deviation (Среднее отклонение) – среднее абсолютных отклонений цены актива за определенный период времени.

$$\boldsymbol{CCI} = \frac{\boldsymbol{TypicalPrice - SimpleMovingAverage}}{\boldsymbol{0.015 * MeanDeviation}} \quad (8)$$

PSAR (Parabolic Stop and Reverse) – это технический индикатор, разработанный Дж. Уэллсом Уайлдером, который используется для определения направления тренда и потенциальных точек разворота на рынке. PSAR помогает трейдерам устанавливать уровни стоп-лоссов и принимать решения о входе и выходе из позиций. Для расчета индикатора представлена формулы 9 и 10, где AF (Acceleration Factor) – фактор ускорения, который обычно устанавливается на 0,02 и может увеличиваться до 0,2; EP (Extreme Point) – максимальная цена в восходящем тренде или минимальная цена в нисходящем тренде.

$$\boldsymbol{PSAR = Prior\ PSAR + Prior\ AF\ (Prior\ EP - Prior\ PSAR)} \quad (9)$$

$$\boldsymbol{PSAR = Prior\ PSAR - Prior\ AF\ (Prior\ PSAR - Prior\ EP)} \quad (10)$$

AD (Accumulation/Distribution) – это технический индикатор, который используется для оценки силы тренда и выявления возможных точек разворота на рынке. Индикатор AD был разработан Марком Чайкиным и помогает трейдерам понять, накапливается ли актив (покупается) или распределяется (продается) в определенный период времени. Для расчета индикатора представлена формула 11:

$$\boldsymbol{ADt = ADt - 1 + \left(\frac{(C - L)(H - C)}{H - L}\right) * Volume,} \quad (11)$$

где C – цена закрытия текущего периода; H – максимальная цена текущего периода; L – минимальная цена текущего периода; V – объем торгов текущего периода; AD предыдущий – значение индикатора AD за предыдущий период.

FI (Force Index) – это технический индикатор, разработанный Александром Элдером, который измеряет силу и направление тренда, основываясь на объеме и изменении цены. Индикатор помогает трейдерам оценить, насколько сильно покупатели или продавцы контролируют рынок. Для расчета индикатора представлена формула 12:

$$\boldsymbol{FI} = (\boldsymbol{C} - \boldsymbol{C}\text{предыдущий}) * \boldsymbol{V}, \quad (12)$$

где C – цена закрытия текущего периода; C предыдущий – цена закрытия предыдущего периода; V – объем торгов текущего периода.

Stoch RSI (Stochastic Relative Strength Index) Это технический индикатор, который сочетает в себе два популярных индикатора: Stochastic Oscillator и

Relative Strength Index (RSI). Он используется для определения уровней перекупленности и перепроданности, а также для выявления возможных точек разворота на рынке. Для расчета индикатора представлена формула 13:

$$\boldsymbol{Stoch\ RSI = \frac{RSI - RSImin}{RSImax - RSImin}}, \tag{13}$$

где RSI – текущее значение индикатора RSI; RSImin – минимальное значение RSI за определенный период; RSImax – максимальное значение RSI за тот же период.

MFI (Money Flow Index) Это технический индикатор, который используется для оценки давления покупок и продаж на рынке. Он основан на объеме торгов и ценах, что позволяет трейдерам определять уровни перекупленности и перепроданности актива. MFI может быть полезен для выявления возможных точек разворота тренда. MFI рассчитывается на основе цен и объема за определенный период (обычно 14 дней). Расчет состоит из следующих шагов:

Определение типичной цены (TP). Для расчета индикатора представлена формула (13).

Определение типичной цены (Определение денежного потока (MF). Если типичная цена текущего периода выше типичной цены предыдущего периода, то денежный поток считается положительным. Если типичная цена текущего периода ниже, то денежный поток считается отрицательным.

Расчет MFI. Для расчета индикатора представлена формула 14.

$$\boldsymbol{MFI = 100 - \left(\frac{100}{1 + \frac{Positive\ Money\ Flow}{Negative\ Money\ Flow}} \right)}. \tag{14}$$

Суммирование положительного и отрицательного денежного потока за выбранный период. Для расчета индикатора представлены формулы 15 и 16.

$$\boldsymbol{Positive\ Money\ Flow} = \sum(\boldsymbol{MF} \text{ при } \boldsymbol{TP} > \boldsymbol{TP}\text{предыдущий}). \tag{15}$$

$$\boldsymbol{Negative\ Money\ Flow} = \sum(\boldsymbol{MF} \text{ при } \boldsymbol{TP} < \boldsymbol{TP}\text{предыдущий}). \tag{16}$$

SMMA (Smoothed Moving Average) Это сглаженная скользящая средняя, которая используется в техническом анализе для определения трендов и сглаживания ценовых данных. SMMA является одним из видов, скользящих средних, и отличается от других типов (например, простых или экспоненциальных) тем, что она более плавно реагирует на изменения цен.

Расчет состоит из следующих шагов: для первого значения SMMA используется простая средняя за заданный период n. Для расчета индикатора представлена формула 17:

$$\boldsymbol{SMMA1} = \frac{\boldsymbol{C1} + \boldsymbol{C2} + \cdots + \boldsymbol{Cn}}{\boldsymbol{n}}, \tag{17}$$

где Ci – цена закрытия за период i.

Для последующих значений SMMA используется предыдущая SMMA и текущая цена. Для расчета индикатора представлена формула 18:

$$\boldsymbol{SMMAt} = \frac{\left(\boldsymbol{SMMAt} - \boldsymbol{1} * (\boldsymbol{n} - \boldsymbol{1})\right) + \boldsymbol{Ct}}{\boldsymbol{n}}, \tag{18}$$

где Ct – цена закрытия текущего периода.

Ichimoku Cloud. Это комплексный индикатор технического анализа, который используется для оценки трендов, уровней поддержки и сопротивления, а также для определения возможных точек входа и выхода из сделок. Он был разработан в Японии и стал популярным среди трейдеров по всему миру благодаря своей способности предоставлять полное представление о рыночной ситуации. Расчет состоит из следующих шагов:

Tenkan-sen рассчитывается как среднее значение максимума и минимума за последние 9 периодов. Для расчета индикатора представлена формула 10.

$$\boldsymbol{Tenkan - sen} = \frac{\boldsymbol{Max(High, 9) + Min(Low, 9)}}{\boldsymbol{2}} \quad (19)$$

Kijun-sen рассчитывается как среднее значение Tenkan-sen и Kijun-sen, сдвинутое вперед на 26 периодов. Для расчета индикатора представлена формула 20.

$$\boldsymbol{Kijun - sen} = \frac{\boldsymbol{Max(High, 26) + Min(Low, 26)}}{\boldsymbol{2}} \quad (20)$$

Senkou Span A рассчитывается как среднее значение Tenkan-sen и Kijun-sen, сдвинутое вперед на 26 периодов. Для расчета индикатора представлена формула 21.

$$\boldsymbol{Senkou\ Span\ A} = \frac{\boldsymbol{Tenkan - sen + Kijun - sen}}{\boldsymbol{2}} \quad (21)$$

2.3. Проектирование пользовательского интерфейса

Веб-приложение по техническому анализу криптовалют включает в себя несколько модулей, главные из них: генерация сигналов, лекционные материалы, модуль-помощь и авторизация.

Модуль генерации сигналов является основным для пользователей данного приложения, в этом модуле пользователь должен ввести или выбрать данные для генерации сигнала и получить результат в виде рекомендации.

Полями для ввода или выбора из списка являются:

– ввод названия криптовалюты - (например, Bitcoin, Ethereum), данный ввод будет проходить проверку на существующую криптовалюту;

– выбор типа сделки: долгосрочная, краткосрочная, скальпинг;

– выбор периода сделки – пользователю будет предоставлено три типа сделки: для долгосрочного сделки (1д -1 М), для краткосрочной (1ч -12ч) и для скальпинга (1м -45м);

– выбор типа торговли: Long (на повышение стоимости), Short (на понижение стоимости).

Кроме ввода пользователю будет предоставлены кнопки – показать график, для отображения графика при выбранных параметров и кнопка анализировать -для старта анализа криптовалюты.

Результат анализа будет расположен под графиком в виде таблицы с отображением каждого индикатора и его личного сигнала. Общий сгенерированный сигнал будет расположен под таблицей, в виде кнопки с значением SOLD/BUY.

В результате был подготовлен макет данного модуля, представленного на рисунке 23.

Рисунок 23 – Макет экранной формы для модуля генерации сигналов

Модуль «Лекционный материал». В этом модуле пользователь будет имеет доступ к образовательным материалам, которые помогут разобраться в техническом анализе для лучшего понимания логики генерации сигналов.

Образовательные материалы будут даны в виде лекций в формате PDF с текстовой информацией, с формулами и иллюстрациями. Лекции будут представлены по следующим разделам технического анализа:

– японские свечи;

– уровни поддержки и сопротивления;

– линии тренда;

– фигуры графического анализа;

– технические индикаторы.

В результате был подготовлен макет данного модуля, представленного на рисунке 24.

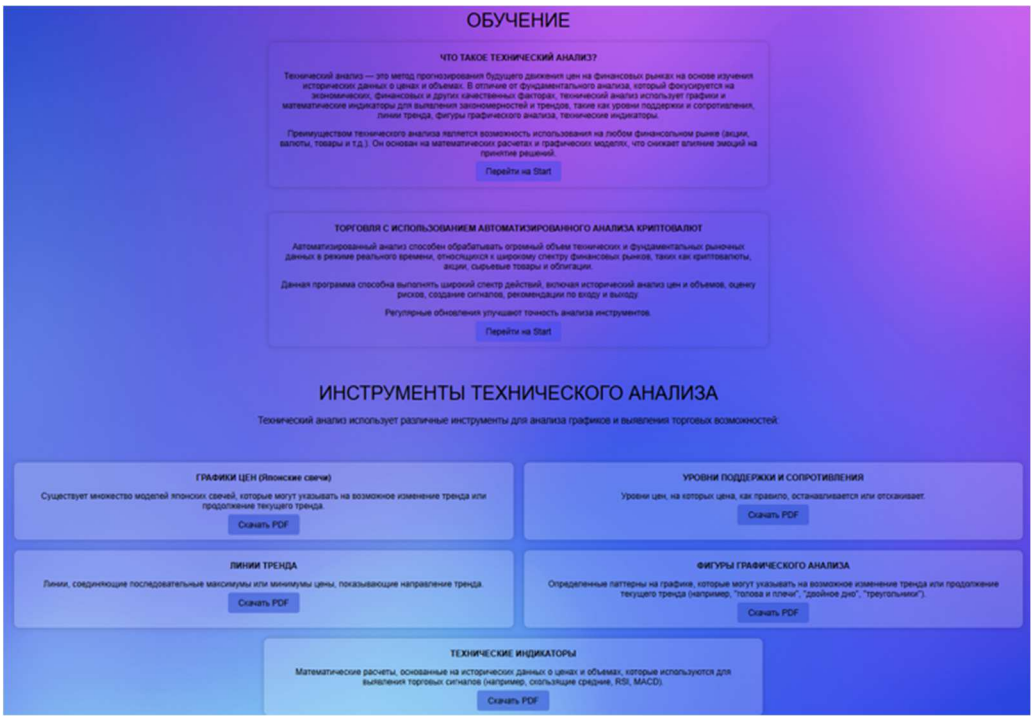

Рисунок 24 – Макет экранной формы обучающего раздела

Модуль «Помощь». Основной целью данного модуля является обратная связь между пользователями приложения и разработчиками. В нем представлены следующие разделы:

Отзывы. Пользователи могут оставить свой отзыв или ознакомиться с отзывами других пользователей данного приложения;

Разработки. Здесь можно узнать о последних изменениях и обновлениях в генерации сигналов;

Часто задаваемые вопросы. Раздел содержит ответы на популярные вопросы пользователей, а также возможность задать свой;

Справочный центр. В этом разделе представлена подробная информация о работе и использовании приложения;

Обратная связь. Позволяет связаться с консультантом для получения помощи и консультаций.

В результате был подготовлен макет данного модуля, представленного на рисунке 25.

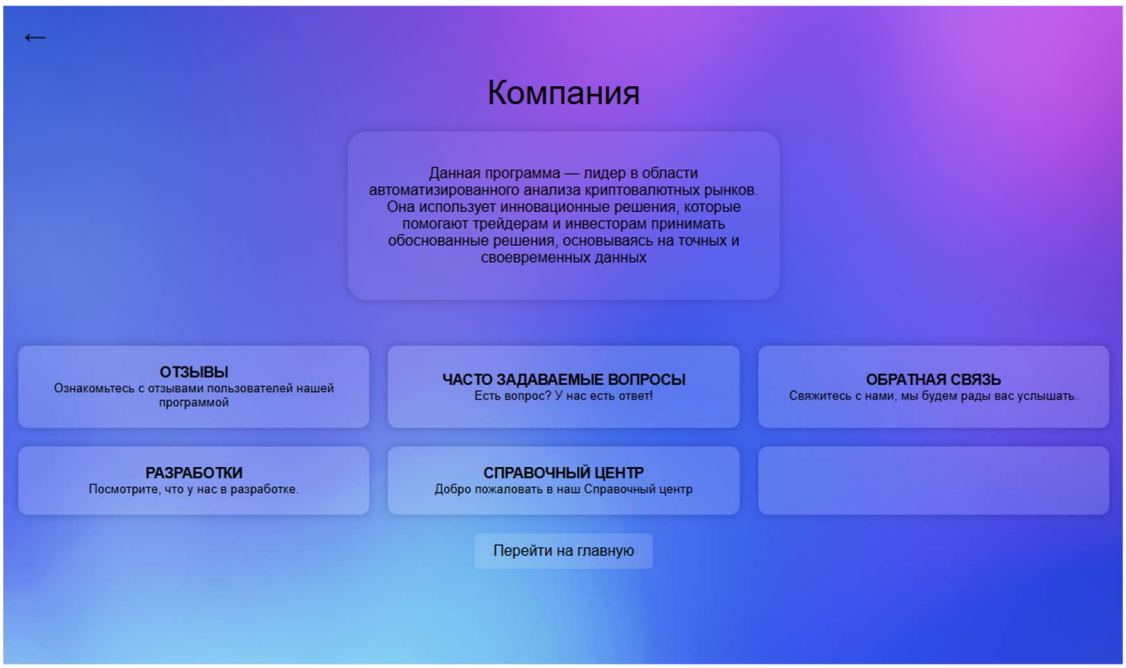

Рисунок 25 – Макет экранной формы к модулю «Помощь»

Модуль регистрации. В этом модуле пользователь может зарегистрироваться, для получения торговых сигналов на почту. Все поля ввода являются обязательными. Будут использоваться подсказки и маски ввода для удобств пользователя.

Поля для ввода:

– логин;

– пароль;

– электронная почта.

В результате был подготовлен макет данного модуля, представленного на рисунке 26.

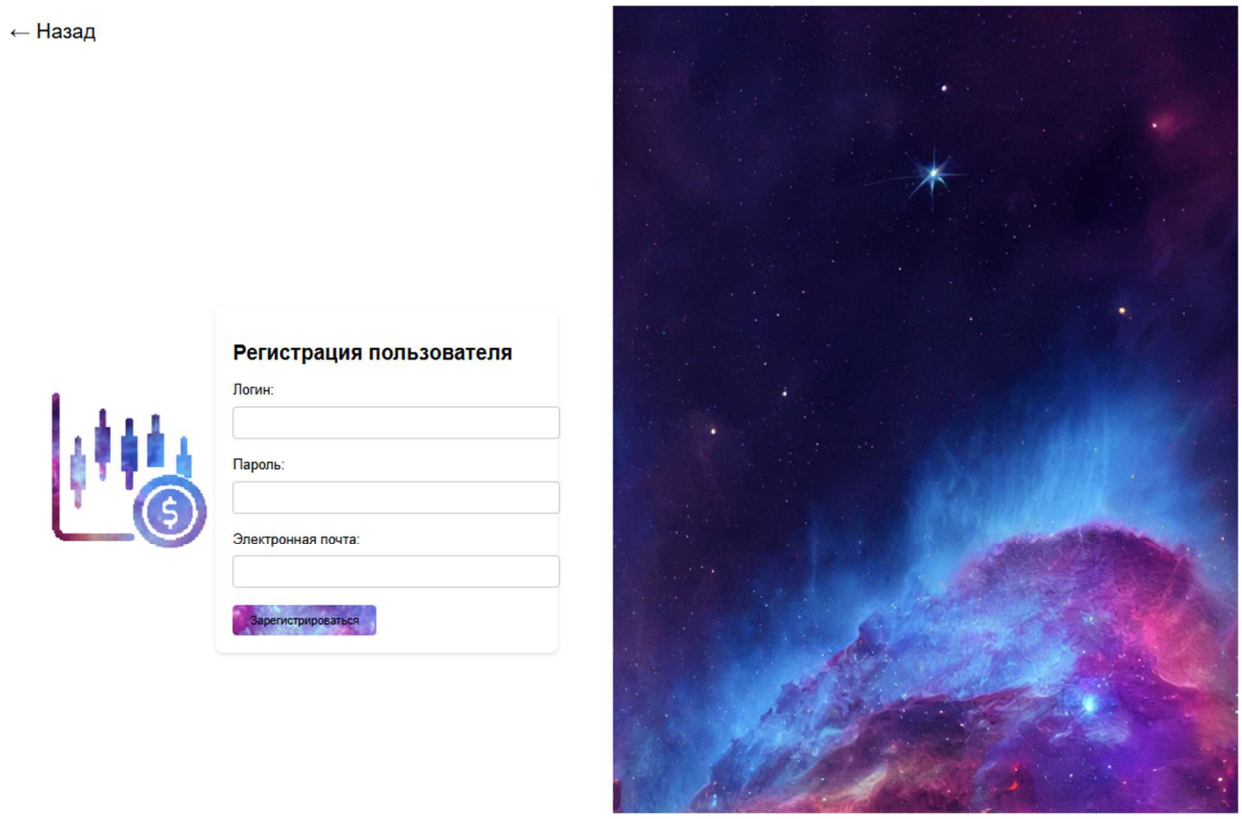

Рисунок 26 – Макет экранной формы к модулю регистрации

2.4. Выбор архитектуры

Разрабатываемое приложение для технического анализа криптовалютного рынка построено на основе клиент-серверной архитектуры. Данная модель предусматривает разделение системы на две основные части: клиентскую и серверную. В качестве серверной платформы выбран стек технологий Node.js с использованием веб-фреймворка Express.js, а в качестве базы данных – NoSQL СУБД MongoDB. Все компоненты развернуты локально на компьютере разработчика, что упрощает отладку и обеспечивает независимость от облачных сервисов. Ниже подробно обосновывается выбор такой архитектуры, описывается взаимодействие между фронтендом и бэкендом, структура модулей приложения, а также демонстрируются преимущества решения с точки зрения гибкости, масштабируемости, модульности, безопасности и удобства разработки.

Клиент-серверная архитектура была выбрана по причине ее естественного соответствия задаче веб-приложения. В данной архитектуре пользовательский интерфейс отделен от серверной части, которая обрабатывает данные и выполняет бизнес-логику. Клиентская часть отправляет запросы на сервер через HTTPS по REST API и отображает результаты для пользователя, в то время как серверная часть принимает и обрабатывает эти запросы, взаимодействует с базой данных и внешними сервисами по необходимости, а затем возвращает ответы клиенту.

Альтернативой могло бы быть монолитное приложение, выполняющее все функции на стороне клиента или сервера, однако такой подход был бы неэффективен. Например, реализация всей логики на стороне клиента привела бы к раскрытию исходного кода и невозможности обеспечить надлежащую защиту данных, а также переложила бы вычислительную нагрузку на устройство пользователя. Также выполнение всех задач в одном модуле без разделения усложнило бы поддержку и развитие системы. Таким образом, клиент-серверная архитектура является оптимальным выбором для веб-приложения. В составе решения присутствуют три ключевых компонента: пользовательский интерфейс, сервер с бизнес-логикой и API, а также база данных для хранения информации.

В качестве платформы для разработки серверной части был выбран Node.js с фреймворком Express.js. Node.js представляет из себя высокопроизводительную серверную платформу JavaScript, которая известна своей асинхронной моделью выполнения. Данная модель позволяет эффективно обрабатывать множество одновременных запросов без блокировки потоков, что особенно ценно для данного приложения, интенсивно работающего с вводом-выводом (запросы к внешним API, базам данных и т.д.). В данного приложения Node.js хорошо подходит, так как предполагается частая загрузка данных с внешних ресурсов (например, ценовые данные рынка криптовалют с бирж) и обмен данными с клиентом в

режиме реального времени. Асинхронность Node.js обеспечивает плавную обработку таких операций и высокую отзывчивость сервера.

Express.js был выбран как основной веб-фреймворк на стороне сервера, потому что он хорошо подходят для создания веб-приложений на Node.js. Express является гибким фреймворком, расширяющим базовые возможности Node.js и предоставляющим удобные инструменты для разработки веб-сервера. По своему назначению Express решает множество рутинных задач: настройку маршрутизации URL и обработку HTTP-запросов, управление сессиями, обработку ошибок и т.д. С его помощью можно легко определить набор RESTful маршрутов для API нашего приложения и связать их с соответствующей логикой обработки. Фреймворк нет жесткой архитектуры, позволяя реализовать собственную структуру каталогов и модулей. Это упростило организацию кода: в проекте выделены отдельные файлы/модули для маршрутов API, для бизнес-логики и для взаимодействия с базой данных. Кроме того, Express.js прекрасно поддерживает генерацию ответов в формате JSON, что совпадает с потребностями нашего приложения, так как фронтенд обменивается с сервером JSON-данными. Фреймворк также совместим с любыми популярными базами данных – можно подключать MongoDB, PostgreSQL, MySQL и др. В нашем случае использование Express в сочетании с MongoDB не вызывает затруднений благодаря наличию официальных драйверов и middleware для Node.js.

Для хранения данных приложения была выбрана документо-ориентированная база данных MongoDB. Данный выбор обусловлен несколькими факторами технического характера. Во-первых, MongoDB – это NoSQL СУБД, которая хранит информацию в виде документов JSON. Это обеспечивает большую гибкость в работе с данными по сравнению с традиционными реляционными СУБД: структура хранимых документов не жестко фиксирована схемой, каждый документ может иметь свой набор полей, что удобно при развитии приложения. Для нашего проекта это означает

возможность легко модифицировать структуру данных по мере добавления новых функций, например хранение различных показателей технического анализа, настройки пользователей и прочую информацию без необходимости сложных изменений схемы базы данных.

Во фронтенде нашего приложения реализован веб-интерфейс, позволяющий пользователю запрашивать технический анализ выбранных криптовалют и просматривать такие результаты как: графики, индикаторы и др. Фронтенд представляет собой набор веб-страниц, написанных с использованием HTML, CSS и JavaScript. Эта клиентская часть взаимодействует с сервером посредством HTTP-запросов к REST API, которые определены на сервере. Обмен данными осуществляется в формате JSON: клиент отправляет запросы GET или POST с параметрами в формате JSON или URL-параметрами, а сервер отвечает JSON-объектами, содержащими необходимые данные или результаты операций.

Архитектура серверной части спроектирована модульно, что облегчает поддержку и расширение кода. В проекте выделены следующие основные модули: Модуль маршрутов API, Модуль бизнес-логики, Модуль доступа к данным, Модуль аутентификации и сессий, Фронтенд-модуль.

Таким образом, архитектура приложения получилась многоуровневой и модульной. Каждый компонент отвечает за свою задачу и может разрабатываться и изменяться относительно независимо. Например, изменение структуры базы данных не требует переписывания фронтенда – достаточно скорректировать слой доступа к данным и при необходимости бизнес-логику. Добавление нового индикатора технического анализа затронет только сервисный слой и интерфейс для его отображения, но основные механизмы останутся прежними. Такая модульность соответствует принципам хорошей архитектуры и облегчает сопровождение проекта. Пример схемы архитектуры приложения приведен на рисунке 27.

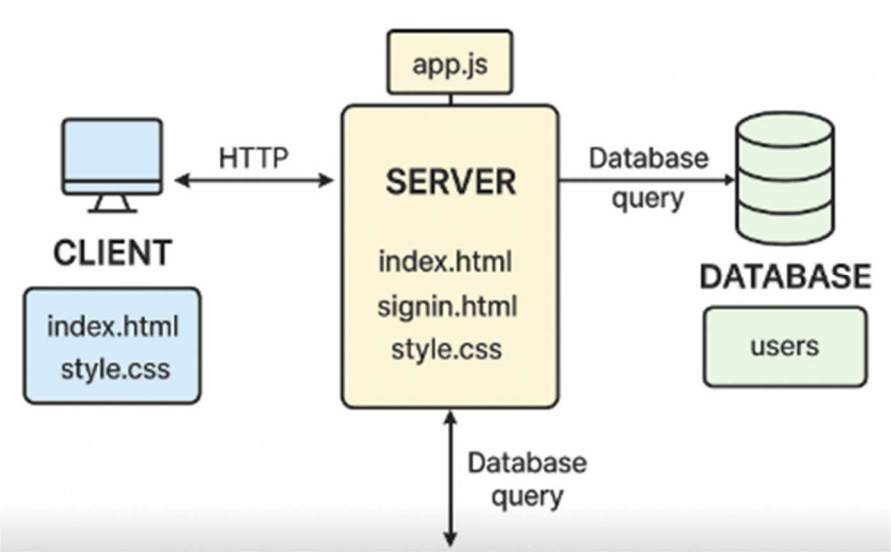

Рисунок 27 – Архитектура приложения (создано при помощи веб-приложения draw.io)

2.5. Выводы по главе

Во второй главе выпускной квалификационной работы подробно рассматриваются ключевые архитектурные решения, положенные в основу разрабатываемого веб-приложения, предназначенного для проведения технического анализа криптовалютного рынка. Основное внимание уделяется формированию общей концепции программной архитектуры, а также обоснованию выбора технологического стека, включающего в себя современный и широко применяемый веб-фреймворк Express.js, документоориентированную систему управления базами данных MongoDB, а также использование клиент-серверной модели, предусматривающей локальное развёртывание приложения на пользовательском или серверном оборудовании.

В рамках главы приводится описание общей структуры приложения и механизма взаимодействия его основных модулей. Особое внимание уделяется организации обмена данными между клиентской и серверной частями системы, что обеспечивает целостность, стабильность и

своевременную обработку запросов пользователей в процессе анализа криптовалют.

Отдельный раздел главы посвящён реализации алгоритмов расчёта основных технических индикаторов, традиционно используемых в аналитике финансовых рынков, таких как MACD (Moving Average Convergence Divergence), RSI (Relative Strength Index) и других. Подробно описаны принципы их интеграции в архитектуру разрабатываемого программного обеспечения, что позволяет обеспечить возможность автоматизированного анализа рыночных данных и формирование обоснованных аналитических выводов на их основе. Это, в свою очередь, создаёт прочную техническую основу для последующего формирования торговых сигналов и принятия решений пользователями приложения.

3. РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ

В данной главе подробно описывается процесс разработки веб-приложения по техническому анализу криптовалют, включая серверную часть, работу с базой данных, аутентификацию пользователей, вычисление технических индикаторов, взаимодействие фронтенда с сервером, а также запуск и используемые инструменты разработки. Реализация выполнена с соблюдением современных подходов к веб-разработке, ориентируясь на надёжность, масштабируемость и безопасность системы.

3.1. Пользовательский интерфейс

Пользовательский интерфейс приложения представлен веб-страницами, через которые пользователь взаимодействует с системой. На одной из страниц доступна форма регистрации нового пользователя: она содержит поля для ввода необходимых данных таких как: логин, пароль и почта (см. приложение А). После успешной регистрации пользователь получает доступ к основному окну приложения. Основное окно интерфейса предоставляет возможность выбора криптовалютной пары и отображает таблицу с актуальной информацией о рынке в реальном времени. Пользователь может ввести название интересующей криптовалюты (например, BTCUSDT) в соответствующее поле ввода. При выборе инструмента на странице появляется таблица, строки которой динамически обновляются по мере поступления новых данных (см. приложение А). Таблица включает колонки с ключевыми показателями: текущая цена, рассчитанные значения технических индикаторов и статус торгового сигнала. Обновление этих данных происходит без перезагрузки страницы благодаря встроенному клиентскому скрипту, который автоматически получает от сервера актуальные данные и обновляет содержимое таблицы (см. приложение А).

3.2. Серверная часть

Серверная часть приложения реализована на платформе Node.js с использованием фреймворка Express.js. Node.js был выбран благодаря его возможности обрабатывать большое количество входящих запросов в режиме реального времени за счёт событийно-ориентированной, не блокирующей I/O архитектуры, что особенно важно при работе с потоками финансовых данных. Express, в свою очередь, представляет собой популярный минималистичный веб-фреймворк для Node.js, написанный на JavaScript. Использование Express.js упростило создание веб-сервера и маршрутизацию HTTP-запросов, предоставив готовый набор функций для разработки API.

При инициализации серверного приложения создаётся экземпляр Express и настраивается базовый middleware. В частности, для разбора входящих JSON-данных применяется встроенный парсер express.json(), что позволяет серверу принимать запросы с телом в формате JSON. После настройки парсеров и других middleware, определяются маршруты (routes), соответствующие функциональности приложения – например, маршруты для регистрации, аутентификации и получения аналитических данных. Каждый маршрут обрабатывается определённым контроллером (функцией-обработчиком), который реализует необходимую логику. На рисунке 28 приведён фрагмент кода, демонстрирующий создание Express-приложения и базовую настройку сервера.

```
const express = require("express");
const connectToMongo = require("./db");  // модуль подключения к БД
const app = express();

connectToMongo();            // подключение к базе данных MongoDB
app.use(express.json());     // поддержка JSON в теле запросов
app.use("/auth", require("./auth"));  // подключение маршрутов аутентификации

const PORT = 3000;
app.listen(PORT, () => {
  console.log(`Server listening at http://localhost:${PORT}`);
});
```

Рисунок 28 – Программный код на языке программирования JavaScript для создания Express-приложения и базовой настройки сервера

В приведённом коде выполняется подключение к базе данных (через функцию connectToMongo()), включается middleware для JSON, а также монтируется роутер для маршрутов аутентификации по пути /auth. Запуск сервера на порту 3000 осуществляется вызовом app.listen(). Таким образом, после запуска сервер начинает принимать HTTP-запросы, обрабатывать их и перенаправлять в соответствующие модули логики. В данном случае маршрутизатор /auth будет обрабатывать все запросы, связанные с регистрацией и входом пользователей.

Стоит отметить, что Express.js также обеспечивает простое подключение дополнительных middleware, например, для логгирования запросов, обработки ошибок и настройки заголовков. В случае, если фронтенд приложения развернут на другом домене или порте, для корректного взаимодействия может понадобиться разрешить CORS (Cross-Origin Resource Sharing). В проекте при необходимости подключается пакет cors для Express, позволяющий открыто приниматьзапросы с внешнего клиентского приложения. Настройка CORS сводится к вызову app.use(cors()) с указанием требуемых параметров доступа. Это гарантирует, что браузерные запросы с фронтенда успешно достигнут нашего API.

3.3. База данных MongoDB

Для хранения данных приложения используется база данных MongoDB, благодаря её производительности и гибкости в работе с JSON-подобными документами. В качестве инструмента для взаимодействия с MongoDB выбран ORM/ODM Mongoose, который обеспечивает объектно-ориентированный интерфейс к базе данных и валидацию данных на уровне схем. Подключение к базе данных производится при старте сервера: выполняется соединение с локальным или удалённым экземпляром MongoDB по URI, после чего все необходимые коллекции становятся доступными через модели Mongoose.

Процесс подключения вынесен в отдельный модуль (db.js) для удобства и повторного использования. В этом модуле с помощью Mongoose создаётся подключение к MongoDB и обрабатываются возможные ошибки. На рисунке 29 приведён код функции подключения.

```
const mongoose = require("mongoose");
const mongoURI = "mongodb://127.0.0.1:27017/crypto_db";

async function connectMongo() {
  try {
    await mongoose.connect(mongoURI);
    console.log("Connected to MongoDB!");
  } catch (error) {
    console.error("MongoDB connection error:", error.message);
  }
}

module.exports = connectMongo;
```

Рисунок 29 – Программный код на языке программирования JavaScript для подключения к базе данных MongoDB

Данный код устанавливает соединение с базой данных по адресу mongodb://127.0.0.1:27017/crypto_db и выводит сообщение в консоль при успешном подключении. Используется асинхронная функция и оператор await для ожидания установления связи. В случае ошибки она будет перехвачена в блоке catch и залогирована. Подобная организация кода делает структуру проекта более удобной и масштабируемой, позволяя вызывать connectMongo() при запуске сервера и централизованно управлять соединением.

После установки соединения посредством Mongoose можно работать с данными через модели. Mongoose модели определяются с помощью схем (Schema), описывающих структуру документов. Для нашего приложения важнейшей является модель пользователя, так как она необходима для регистрации и аутентификации. На рисунке 30 приведён код модели пользователя.

```
const mongoose = require("mongoose");
const UserSchema = new mongoose.Schema({
  email:    { type: String, required: true, unique: true },
  password: { type: String, required: true }
});
module.exports = mongoose.model("User", UserSchema);
```

Рисунок 30 – Программный код на языке программирования JavaScript для реализации модели пользователя

В этой схеме определены два поля: email и password. Поле email помечено как уникальное и обязательное для заполнения, что гарантирует невозможность регистрации двух учетных записей с одинаковым адресом почты. Поле password также обязательное; оно будет хранить хэш пароля пользователя. При компиляции схемы в модель через mongoose.model автоматически создаётся одноимённая коллекция в базе данных. Документы в

этой коллекции будут содержать только указанные поля с заданными типами, что вводит определённую степень валидации и структуры данных на уровне приложения.

Благодаря Mongoose, основные операции с базой данных значительно упрощаются. Для добавления нового пользователя используется метод User.create(), для поиска – User.findOne() или User.findById(), для обновления – User.updateOne() и т.д. Все эти методы возвращают промисы (или позволяют использовать async/await), что удобно для последовательной обработки результатов и ошибок. Например, чтобы найти пользователя по email, достаточно вызвать User.findOne({email: ...}) – Mongoose сам выполнит запрос к MongoDB и вернет объект пользователя, либо null если документ не найден. Таким образом, взаимодействие с коллекциями реализовано через высокоуровневые методы моделей, без необходимости вручную формировать запросы к базе.

Помимо модели пользователя, в базе данных могут храниться и другие данные, относящиеся к работе приложения. Например, может потребоваться хранение настроек пользователей (избранные торговые пары, период индикатора по умолчанию и т.п.) или результаты анализа, однако в данной работе основной акцент сделан на оперативном вычислении индикаторов по запрошенным данным без длительного сохранения результатов. Исторические ценовые данные криптовалют, необходимые для расчёта показателей, как правило, получают из внешних API и обрабатывают в режиме реального времени, поэтому нет строгой необходимости сохранять их в собственной базе. Но при желании структура базы данных может быть расширена моделями для хранения котировок, чтобы кэшировать данные и уменьшить количество внешних запросов.

3.4. Регистрация и аутентификация пользователей

Реализация системы регистрации и аутентификации пользователей является ключевым элементом приложения. Для управления учётными записями реализован модуль auth на базе Express Router, включающий маршруты регистрации (signup) и входа в систему (login).

Регистрация пользователя. При создании новой учётной записи клиентское приложение отправляет на сервер HTTP POST запрос на специальный маршрут (например, /auth/signup) с данными нового пользователя (e-mail и пароль). Сервер проверяет корректность данных и создаёт запись в базе. Критически важно, что пароли не хранятся в открытом виде – перед сохранением выполняется хеширование пароля. В проекте используется библиотека BcryptJS для безопасного хеширования. Bcrypt защищает от перебора и rainbow-атак. На рисунке 31 показано, как выполняется регистрация пользователя на стороне сервера.

```
// auth.js (маршрут регистрации)
router.post("/signup", async (req, res) => {
  // генерация соли для хеширования
  const salt = await bcrypt.genSalt(10);
  // хеширование пароля с использованием соли
  const securePass = await bcrypt.hash(req.body.password, salt);
  // создание нового пользователя в базе данных
  try {
    const user = await User.create({
      email: req.body.email,
      password: securePass
    });
    res.status(201).json({ message: "User registered", userId: user._id });
  } catch(err) {
    res.status(400).json({ error: "Registration failed" });
  }
});
```

Рисунок 31 – Программный код на языке программирования JavaScript, реализующий регистрацию пользователя

В приведённом фрагменте кода пароль пользователя сначала преобразуется: метод bcrypt.genSalt(10) генерирует крипографическую соль, а bcrypt.hash() вычисляет хеш пароля вместе с этой солью. Полученный хеш (строка из символов) сохраняется вместо оригинального пароля. Далее создаётся запись пользователя в коллекции MongoDB через User.create(). Если указанный email уже существует в базе, операция создаст ошибку из-за ограничения unique: true в модели; в блоке catch этот случай обрабатывается и возвращается соответствующий ответ с кодом 400 (Bad Request). Таким образом, в базе данных никогда не хранится открытый пароль – только безопасный хеш. Это соответствует современным требованиям безопасности: даже в случае компрометации базы данных злоумышленник не сможет напрямую узнать пароли пользователей и узнать их email, так как обратное восстановление из хеша практически невозможно без знания исходного ключа (пароля).

Для входа зарегистрированный пользователь также отправляет POST запрос (например, /auth/login) с своими учетными данными. Сервер по маршруту входа выполняет поиск пользователя в базе по email и затем проверяет соответствие введённого пароля сохранённому хешу. Проверка осуществляется с помощью метода bcrypt.compare(), который принимает введённый пароль и сохраненный хеш и возвращает результат сравнения. Если пользователь с данным email не найден или пароль неверный, сервер возвращает ошибку авторизации, например, HTTP 401 Unauthorized или 400 Bad Request с сообщением об ошибке. В противном случае – аутентификация считается успешной.

После успешной проверки логин/пароль сервер должен создать некую сессию или токен, позволяющий идентифицировать пользователя при последующих запросах. В данной реализации был выбран подход без состояния на сервере с использованием JWT-токенов (JSON Web Token). При успешном входе генерируется подписанный токен JWT, включающий

идентификатор пользователя и срок действия. Для этого применяется библиотека jsonwebtoken. На рисунке 32 показана генерация JWT после успешной аутентификации.

```
const jwt = require("jsonwebtoken");
// ... внутри обработчика /login, после проверки пароля:
const payload = { userId: user._id };
const token = jwt.sign(payload, JWT_SECRET_KEY, { expiresIn: "1h" });
res.json({ token });
```

Рисунок 32 – Программный код на языке программирования JavaScript для генерации JWT-токена

Здесь jwt.sign() создает токен, используя секретный ключ приложения и срок жизни токена (например, 1 час). Токен отправляется клиенту в ответе. Клиентское приложение сохраняет этот токен (в памяти или в хранилище, например LocalStorage, либо в http-only cookie) и прикрепляет его к последующим запросам. Таким образом, при обращении к защищённым ресурсам сервер может проверить подпись токена (методом jwt.verify) и извлечь из него закодированную информацию о пользователе. Данный метод аутентификации удобен, так как устраняет необходимость хранения сессии на сервере.

Альтернативой JWT могла бы быть классическая сессионная аутентификация с cookies. Однако в контексте раздельного фронтенда JWT оказался более подходящим решением благодаря простоте использования в AJAX-запросах и гибкости.

3.5. Реализация функций технического анализа криптовалют

Ключевая часть веб-приложения является модуль технического анализа, который выполняет вычисления индикаторов по данным о ценах криптовалют.

Технический анализ предполагает использование различных статистических индикаторов, основанных на ценовых рядах, для выявления трендов и прогнозирования движения рынка. В рамках проекта были реализованы наиболее распространённые технические индикаторы, такие как скользящие средние (MA) различных периодов, индекс относительной силы (RSI), полосы Боллинджера, MACD и другие подробно описанные в главе 2. Реализация этих индикаторов может быть выполнена двумя способами: с помощью подключения готовой библиотеки или посредством самостоятельного программирования формул.

В интересах скорости разработки был сделан упор на использовании

готовых решений, допуская при этом проверку их корректности на отдельных примерах. В частности, был использован пакет technicalindicators – популярная библиотека на JavaScript, предоставляющая набор функций для расчёта технических индикаторов. Эта библиотека позволила избежать ошибок при ручной реализации сложных формул и обеспечила оптимизированное вычисление индикаторов. Например, расчёт простой скользящей средней (SMA) с периодом N для массива цен values может быть выполнен вызовом встроенной функции библиотеки. На рисунке 33 показан, расчёт простой скользящей средней (SMA).

```
const { SMA } = require("technicalindicators");
const closingPrices = [ /* массив цен закрытия */ ];
const smaValues = SMA.calculate({ period: 5, values: closingPrices });
```

Рисунок 33 – Программный код на языке программирования
JavaScript для вызова функции,
ответственной за расчёт простой скользящей средней (SMA)

Вызов SMA.calculate(...) возвращает массив значений скользящей средней для заданного массива цен. Аналогично, библиотека предоставляет

методы для экспоненциальной скользящей средней (EMA), индекса относительной силы (RSI), а также более сложных индикаторов. Например, RSI вычисляется на основе средних значений приростов и потерь за определённый период и возвращает величину от 0 до 100, указывающую на перекупленность или перепроданность актива. Использование готовой функции RSI.calculate({period: 14, values: prices}) избавляет от необходимости вручную реализовывать алгоритм вычисления RSI, что снижает вероятность ошибок.

Для демонстрации принципов работы индикаторов в ходе разработки также были реализованы упрощённые прототипы функций вручную. Например, функция для расчёта простой скользящей средней может быть написана на JavaScript путем суммирования цен внутри скользящего окна и деления на длину окна. Такой подход позволил убедиться в правильности работы библиотечных методов. Однако в конечной версии приложения расчёты доверены библиотеке technicalindicators, прошедшей проверку сообществом, что повысило доверие к результатам анализа.

Важно отметить, что прежде чем вычислять какие-либо индикаторы, приложение должно получить исходные данные – исторические цены криптовалют. Для этого реализована интеграция с внешним API биржи. Была выбрана криптобиржа Binance, далее будет рассмотрен пример использования публичного API биржи Binance, предоставляющий данные о ценах в формате JSON. Серверная часть отправляет запрос к внешнему API для выбранного пользователем актива (например, BTC/USDT) и временного интервала, получает массив цен (например, временные свечи с ценами открытия, закрытия, максимума и минимума) и передаёт массив цен закрытия функциям индикаторов. Полученные результаты индикаторов затем используются для формирования ответа клиенту.

Для реализации функции отправки уведомлений был использован модуль nodemailer, являющийся стандартным инструментом для работы с

электронной почтой в среде Node.js. Настройка соединения с почтовым сервером осуществляется посредством протокола SMTP. В целях безопасности авторизация на стороне сервера производится с использованием специализированных паролей приложений, а конфиденциальные данные (например, логин и пароль от почтового ящика) хранятся в переменных окружения, что позволяет избежать их прямого включения в исходный код.

Все ключевые функции анализа были тщательно протестированы на корректность: сверялись результаты вычислений с эталонными значениями, полученными из сторонних источников, например, с TradingView и Investing для того, чтобы убедиться в правильности работы алгоритмов. Вычислительная нагрузка при обработке индикаторов невелика.

3.6. Взаимодействие фронтэнда и сервера

Фронтенд веб-приложения обеспечивает интерфейс, через который пользователь взаимодействует с системой: вводит запросы (например, выбирает криптовалюту и индикаторы для анализа), просматривает результаты в виде графиков и таблиц, а также осуществляет действия регистра-ции/входа. В данной работе фронтенд реализован как многостраничное веб-приложение, работа которого тесно связана с серверным API. Связь между фронтендом и сервером выстроена на основе обмена сообщениями по про-токолу HTTP с использованием формата данных JSON.

После запуска приложение фронтенда загружается в браузере пользователя и взаимодействует с сервером путем AJAX-запросов или через библиотеку для запросов (например, fetch API или Axios). Когда пользователь запрашивает определённую информацию – например, построить график индикаторов для BTC за последнюю неделю – фронтенд формирует соответствующий HTTP-запрос к серверу, такой запрос имеет следующий вид: /api/analysis?symbol=BTC&interval=7d-&indicators=SMA,RSI. Сервер, получив

запрос, проверяет токен пользователя (если данный ресурс требует авторизации), затем извлекает параметры запроса (в данном случае символ валюты, интервал, требуемые индикаторы) и вызывает соответствующие функции технического анализа на сервере. Результат вычислений сервер отправляет обратно в ответе в формате JSON.

Важно отметить, что взаимодействие происходит в режиме реального времени по запросу пользователя: при изменении настроек анализа (выбор другого индикатора или временного диапазона) фронтенд снова обращается к API сервера и обновляет отображение. Если бы требовалось обеспечить непрерывное обновление (например, поток котировок и индикаторов в реальном времени), стоило бы реализовать WebSocket-соединение для трансляции обновлений без постоянных опросов. Однако в данной реализации предполагается, что анализ выполняется по запросу, и пользователь обновляет данные вручную или через периодический опрос API.

Коммуникация фронтенда с сервером защищена стандартными механизмами веб-безопасности. Используется протокол HTTPS для шифрования трафика, особенно важного при передаче учетных данных во время логина. Также все запросы, требующие аутентификации, содержат JWT-токен пользователя. На сервере для таких запросов настроен middleware авторизации: он проверяет наличие и валидность токена. Только при успешной валидации токена запрос проходит дальше к основной логике (например, к функции расчёта индикаторов). Если токен отсутствует или недействителен, сервер возвращает ошибку (HTTP 401 Unauthorized), и фронтенд перенаправляет пользователя на страницу входа.

3.7. Выводы по главе

В ходе реализации веб-приложения для технического анализа криптовалют была последовательно воплощена архитектурная модель,

обеспечивающая эффективное взаимодействие между клиентской и серверной частями. Серверная логика на платформе Node.js с использованием Express.js позволила организовать масштабируемое и модульное API, а MongoDB с ODM-библиотекой Mongoose обеспечила гибкое и надёжное хранение данных пользователей. Ключевые аспекты безопасности были реализованы через систему аутентификации на основе JWT-токенов и хеширования паролей с использованием bcrypt.

Реализация вычислительных модулей технического анализа опиралась как на проверенные внешние библиотеки, так и на собственные алгоритмы, что обеспечило точность и воспроизводимость результатов. Клиентская часть взаимодействует с сервером через HTTP-интерфейс, получая данные в формате JSON и визуализируя их в интерактивной форме. Таким образом, в результате проделанной работы было создано функциональное, безопасное и расширяемое программное решение, способное выполнять аналитические задачи в режиме реального времени.

4. ПРОВЕДЕНИЕ ТЕСТИРОВАНИЯ ВЕБ-ПРИЛОЖЕНИЯ

В данной главе представлены результаты тестирования разработанного веб-приложения для технического анализа криптовалют. Цель тестирования – проверить корректность реализации основных функций приложения и убедиться, что система удовлетворяет требованиям технического задания. Были протестированы следующие аспекты: функциональность регистрации новых пользователей, получение актуальных котировок криптовалют с биржи Binance, правильность расчета технических индикаторов (с учетом допустимой погрешности ±2%), логика генерации торговых сигналов на основе рассчитанных индикаторов, а также корректность формирования текстовых рекомендаций для пользователя по результатам анализа. Кроме того, проведено кроссбраузерное тестирование работы клиентской части приложения в основных настольных браузерах (Google Chrome, Mozilla Firefox, Microsoft Edge и Safari) для подтверждения корректной работы интерфейса во всех целевых средах.

4.1. Тестирование функциональности регистрации пользователей

Регистрация нового пользователя является первым ключевым сценарием приложения, поэтому ее функциональность была проверена особенно тщательно. Тестирование регистрации включало проверку как клиентской, так и серверной частей процесса. В веб-интерфейсе проверялась корректность работы формы регистрации: приложение должно правильно обрабатывать ввод данных пользователя и выводить сообщения об ошибках при некорректном вводе. Было подтверждено, что при заполнении всех обязательных полей валидными данными и отправке формы происходит успешное создание учетной записи. В этом случае система добавляет нового

пользователя в базу данных, инициирует создание пользовательской сессии и перенаправляет на страницу профиля или главного интерфейса приложения (в зависимости от принятой логики после регистрации).

Были также предусмотрены тестовые случаи для неуспешной регистрации. Например, проверялась ситуация, когда пользователь пытается зарегистрироваться с уже существующим адресом электронной почты. Тестирование показало, что при вводе email, который уже присутствует в системе, попытка регистрации блокируется: сервер возвращает соответствующий ответ об ошибке, а клиентская часть отображает понятное уведомление (например, «Пользователь с таким email уже зарегистрирован»). Аналогично, проверялось поведение при некорректных данных: пустых или невалидных значениях в полях (например, слишком короткий пароль или неправильный формат email). В каждом случае система корректно обрабатывала ситуацию: фронтенд-валидатор предупреждал о проблеме до отправки данных либо серверная проверка возвращала сообщение об ошибке, которое отображалось на форме. Таким образом, функциональность регистрации пользователей прошла полное тестирование и подтвердила работоспособность как при корректном вводе, так и при ошибочных сценариях. Результат тестирования в таблице 3.

Таблица 3 – Результаты тестирования функциональности регистрации пользователей

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Регистрация с корректными данными	1. Перейти на страницу регистрации. 2. Ввести уникальный адрес электронной почты и валидный пароль 3. Нажать кнопку «Регистрация».	Пользователь успешно зарегистрирован; происходит перенаправление на страницу профиля либо отображается сообщение об успешной регистрации.	Да
Регистрация без указания E-mail	1. Перейти на страницу регистрации. 2. Оставить поле адреса электронной почты пустым, заполнив остальные обязательные поля. 3. Нажать «Регистрация».	Регистрация не выполняется; отображается сообщение об ошибке о том, что адрес электронной почты должен быть указан (обязательное поле).	Да
Регистрация без указания пароля	1. 1. Перейти на страницу регистрации. 2. Ввести валидный адрес электронной почты, оставить поле пароля пустым. 3. Нажать «Регистрация».	Регистрация не выполняется; появляется предупреждение о том, что пароль является обязательным для заполнения.	Да

Продолжение таблицы 3

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Регистрация с уже существующим E-mail	1. Убедиться, что в системе уже существует учетная запись с определенным адресом электронной почты. 2. Перейти на страницу регистрации и ввести тот же адрес электронной почты, заполнив остальные поля валидными данными. 3. Нажать «Регистрация».	Регистрация отклоняется; отображается сообщение о том, что пользователь с таким адресом электронной почты уже зарегистрирован (нельзя создать дубликат учетной записи).	Да
Регистрация с некорректным E-mail	1. Перейти на страницу регистрации. 2. Ввести некорректный адрес электронной почты (например, без символа «@»). 3. Нажать «Регистрация».	Регистрация не выполняется; отображается сообщение о неверном формате адреса электронной почты.	Да

4.2. Тестирование парсинга котировок криптовалют с Binance

Важной частью серверной логики приложения является модуль, отвечающий за парсинг актуальных котировок криптовалют с биржи Binance через API. Тестирование этого компонента проводилось для подтверждения того, что система корректно получает данные о ценах и обрабатывает их без ошибок. В рамках проверки выполнялись запросы к API Binance для различных популярных валютных пар (например, BTC/USDT, ETH/USDT) и проверялось, что полученные котировки соответствуют реальным рыночным значениям. Для верификации точности результаты, возвращаемые приложением, сравнивались с данными на самой бирже Binance в тот же момент времени. Отклонения цен оказались минимальными и находились в пределах нормы, что свидетельствует о правильной работе механизма парсинга. Следует учитывать, что незначительные расхождения в котировках

могут возникать из-за сетевых задержек или разной частоты обновления, однако в тестах зафиксировано отклонение не более долей процента.

Кроме того, проверялись ситуации с ошибками при получении котировок. Была смоделирована недоступность внешнего API (например, временное отсутствие сетевого соединения или использование неверного тикера валютной пары). Ожидаемым поведением системы в таких случаях является обработка исключения без прекращения работы всего приложения. Тестирование подтвердило, что при возникновении сетевой ошибки приложение корректно реагирует: выводит уведомление об отсутствии связи (либо записывает ошибку в журнал на сервере) и продолжает работу, предпринимая повторные попытки подключения при восстановлении связи. Если указывается несуществующая валютная пара, система тоже ведет себя правильно – API возвращает ошибку, которая перехватывается модулем парсинга, и пользователю выводится сообщение о невозможности получить данные для данной пары. Модуль запроса котировок с Binance продемонстрировал устойчивость к непредвиденным ситуациям и корректность функционирования в штатном режиме. Результат тестирования в таблице 4.

Таблица 4 – Результаты тестирования парсинга цен криптовалют с Binance

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Получение котировок для существующей криптовалюты	1. 1. Открыть раздел приложения, отображающий котировки криптовалют. 2. Выбрать валидную валютную пару и инициировать запрос котировки из API Binance. 3. Дождаться ответа от API.	Актуальные данные по выбранной валюте успешно получены с Binance.	Да

Продолжение таблицы 4

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Получение котировок для несуществующей валютной пары	1. Попытаться запросить котировку для несуществующей или некорректной валютной пары. 2. Отправить запрос к API Binance и ожидать ответа.".	Система не получает данных для неверного инструмента; отображается сообщение об ошибке, приложение не выходит из строя.	Да
Сбой при отсутствии подключения к сети/API	1. Отключить интернет-соединение или имитировать недоступность API Binance. 2. В приложении попытаться загрузить котировки для криптовалюты.	Запрос не выполняется; приложение выводит уведомление о невозможности получить данные.	Да
Загрузка исторических цен для расчёта индикаторов	1. В приложении запросить исторические данные цен выбранной криптовалюты через API Binance. 2. Дождаться завершения обработки полученных данных.	Исторические котировки успешно получены и сохранены; данные корректно формируются для дальнейшего использования в расчёте технических индикаторов.	Да

4.3. Тестирование расчёта технических индикаторов

Одним из ключевых требований приложения является правильный расчет технических индикаторов на основе исторических данных цен. В рамках тестирования данного функционального блока проводилась проверка алгоритмов расчета индикаторов с целью убедиться в их точности. Для этого были подготовлены контролируемые наборы входных данных (временные ряды цен за определенные периоды), по которым вычислялись целевые показатели (например, простая скользящая средняя (SMA), экспоненциальная

скользящая средняя (EMA), индекс относительной силы (RSI) и др.). Результаты, выдаваемые приложением, сравнивались с эталонными значениями, полученными альтернативным способом – либо с помощью расчета вручную по формулам, либо с использованием известных библиотек технического анализа.

Сравнение показало, что расхождения между значениями индикаторов, вычисленными системой, и эталонными значениями не превышают допустимую погрешность ±2%. Например, при тестовом расчете 14-дневного RSI на заранее подготовленном наборе данных приложение выдало значение, отличающееся менее чем на 1% от значения, рассчитанного вручную. Аналогично, для 50-дневной скользящей средней среднее отклонение от ожидаемого значения составило порядка 0,5%. Такие малые расхождения находятся в пределах допустимой погрешности и объясняются эффектами округления или небольшими отличиями в методике расчета. Таким образом, тестирование подтвердило корректность реализации формул технических индикаторов в приложении. Все проверенные индикаторы, включая различные виды скользящих средних, RSI и другие реализованные осцилляторы рассчитывались верно. Результат тестирования в таблице 5.

Таблица 5 – Результаты тестирования расчёта технических индикаторов

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Расчёт простой скользящей средней (SMA)	1. Загрузить известный набор исторических цен. 2. Включить в приложении индикатор SMA с заданным периодом для данного набора данных. 3. Получить рассчитанное значение SMA.	Значение SMA рассчитывается корректно; вычисленный приложением результат соответствует ожидаемому среднему значению для заданного периода.	Да
Расчёт индекса относительной силы (RSI)	1. Загрузить исторические данные, достаточные для расчёта RSI. 2. Активировать индикатор RSI в приложении для выбранных данных. 3. Дождаться отображения результата RSI.	Отображаемое значение RSI соответствует рассчитанному по формуле значению; индикатор RSI работает корректно.	Да
Расчёт экспоненциальной скользящей средней (EMA)	1. Загрузить набор цен за достаточный период. 2. Включить индикатор EMA с заданным периодом и дождаться вычисления значения. 3. Считать результат EMA, показанный приложением.	Рассчитанное значение EMA соответствует ожидаемому значению экспоненциальной средней по введённым данным.	Да
Расчёт индикатора при недостатке данных	1. Загрузить небольшой объём исторических данных. 2. Попытаться вычислить индикатор, требующий большего количества точек. 3. Отследить реакцию системы.	Система корректно обрабатывает ситуацию: индикатор не вычисляется для недостаточного набора данных; ошибок или сбоев не происходит.	Да

4.4. Тестирование логики генерации торговых сигналов

Логика формирования торговых сигналов, таких как рекомендации на покупку или продажу, была протестирована на соответствие заданным правилам стратегии. Поскольку сигналы формируются на основе комбинации технических индикаторов, для проверки использовались специальные сценарии со смоделированными входными данными. Вначале тестировался каждый отдельный тип сигнала в условиях, гарантирующих его возникновение. Например, для проверки сигнала на покупку применялся набор данных, в котором краткосрочная скользящая средняя устойчиво превышает долгосрочную (условие «золотого креста»), а RSI находится в зоне перепроданности (ниже порогового значения, например 30). Такой сценарий должен инициировать четкий бычий сигнал. В ходе теста убедились, что система при этих условиях генерирует сигнал на покупку и отмечает соответствующую ситуацию в данных. Аналогично, для сигнала на продажу использовался обратный случай: краткосрочная средняя пересекает долгосрочную сверху вниз (сигнал «мертвый крест»), а индикатор RSI достигает зоны перекупленности (выше порога, например 70). Ожидаемый результат – генерация сигнала на продажу – был успешно получен в процессе тестирования.

Помимо граничных случаев, проверялись и нейтральные сценарии, когда ни один четкий сигнал не должен формироваться. В таких ситуациях, когда индикаторы показывают разнонаправленные тенденции или находятся в нейтральных зонах, приложение не должно выдавать ни рекомендации на покупку, ни на продажу. Тестирование подтвердило корректность и этой логики: при отсутствии выраженного тренда система не формирует торговых сигналов, и пользователю не предлагаются ошибочные рекомендации. Таким образом, модуль генерации сигналов работает согласно заданным правилам: генерирует сигналы только в случае выполнения необходимых условий и

воздерживается от выдачи сигналов при их отсутствии. Все проверенные сценарии отработали правильно, что свидетельствует о надежности реализованного механизма принятия решений.

В проекте реализована возможность отправки уведомлений пользователю по электронной почте. При получении сигнала на покупку или продажу криптовалюты, формируется и отправляется соответствующее письмо с рекомендацией по действию. Результат тестирования в таблице 6.

Таблица 6 – Результаты тестирования логики генерации торговых сигналов

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Генерация сигнала «Покупка»	1. Смоделировать рыночные условия, при которых стратегия должна выдать сигнал на покупку. 2. Запустить алгоритм анализа/генерации сигналов.	Система формирует торговый сигнал типа «Покупка» для выбранного актива;	Да
Генерация сигнала «Продажа»	1. Смоделировать условия, при которых должен возникнуть сигнал на продажу. 2. Выполнить запуск генерации торговых сигналов на заданных данных.	Система генерирует сигнал «Продажа»; в интерфейсе отображается соответствующее оповещение.	Да
Отсутствие торгового сигнала	1. Предоставить алгоритму данные, не удовлетворяющие условиям ни для покупки, ни для продажи. 2. Выполнить запуск алгоритма генерации сигналов.	Торговый сигнал не генерируется; интерфейс приложения остаётся без новых сигналов, указывая на нейтральное состояние.	Да
Отправка письма при сигнале BUY или SOLD	1. Выбрать криптовалюту 2. Дождаться сигнала BUY или SOLD.	Пользователь получает письмо с рекомендацией	Да

4.5. Тестирование демонстрации текстовых рекомендаций

Завершающим этапом анализа данных в приложении является предоставление пользователю текстовых рекомендаций на основе обнаруженных сигналов и общей рыночной ситуации. Тестирование этого функционала было направлено на проверку того, что выводимые советы точно соответствуют результатам анализа и понятны по формулировке. Для каждого из сценариев, рассмотренных при тестировании сигналов, дополнительно оценивался корректный вывод текстового сообщения. Например, в ситуации с сигналом на покупку (бычий сценарий) приложение должно отобразить рекомендацию вида: «Золотой крест», при расчете индикатора SMA. В ходе тестов убедились, что при генерации сигнала на покупку пользователь действительно видит соответствующее пояснение с позитивной оценкой рынка.

Для противоположного сценария (сигнал на продажу) проверялось, что система выдает предупреждающее сообщение, например: «Крест смерти», при расчете индикатора SMA.». Это сообщение информирует пользователя о медвежьем сигнале. Тестирование подтвердило, что и этот случай обрабатывается верно: при условиях для сигнала на продажу приложение выдает именно такую рекомендацию. Также рассматривались нейтральные ситуации, когда явных сигналов нет – проверялось, что приложение выводит сбалансированное нейтральное уведомление, например: Отсутствие сигнала. Во всех проверенных случаях текстовые рекомендации точно соответствовали логике анализа: ни одна ситуация не привела к неправильному или противоречивому совету. Формулировки сообщений получились понятными и отражающими соответствующее состояние рынка. Таким образом, функциональность генерации пользовательских рекомендаций на основе анализа данных продемонстрировала свою корректность. Результат тестирования в таблице 7.

Таблица 7 – Результат тестирования демонстрации рекомендаций

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Вывод рекомендации «Покупать»	1. Обеспечить состояние, при котором торговая стратегия генерирует сигнал на покупку. 2. Открыть раздел с рекомендациями или дождаться отображения рекомендации после анализа.	Пользователю выводится текстовая рекомендация приобретать выбранную криптовалюту, что соответствует сигналу «Покупка».	Да
Вывод рекомендации «Продавать»	1. Смоделировать ситуацию, при которой стратегия выдаёт сигнал на продажу. 2. Выполнить процедуру анализа и перейти к отображению рекомендаций.	На экране появляется текстовая рекомендация продать выбранную криптовалюту в соответствии с сигналом «Продажа».	Да
Вывод нейтральной рекомендации	2. 1. Установить условия, при которых отсутствуют чёткие торговые сигналы. 2. Запустить анализ и перейти к просмотру текстовой рекомендации.	Отображается нейтральная рекомендация, указывающая на отсутствие активных действий, что соответствует нейтральному состоянию рынка.	Да

4.6. Кроссбраузерное тестирование

В дополнение к функциональным проверкам было проведено кроссбраузерное тестирование, чтобы убедиться в корректной работе приложения во всех основных браузерах. В данной серии тестов приложение запускалось в среде настольных браузеров Google Chrome, Mozilla Firefox, Microsoft Edge (Chromium) под операционной системой Windows 11, а также в браузере Safari под macOS. Проверялись корректность отображения

интерфейса и работоспособность ключевых функций в каждом из этих браузеров.

Для каждого браузера вручную проходился ключевой пользовательский сценарий: регистрация нового пользователя, загрузка и отображение страницы с графиком и техническими индикаторами, генерация торговых сигналов (смоделированные тестовые условия) и вывод рекомендаций. Особое внимание уделялось визуальному оформлению и взаимодействию: элементы интерфейса (формы, кнопки, графики) должны выглядеть единообразно и функционировать без сбоев во всех браузерах.

Результаты кроссбраузерного тестирования показали, что разработанное приложение корректно работает во всех заявленных средах. В браузерах Chrome, Firefox и Edge интерфейс приложения отображался идентично, без заметных различий в стилях или поведении скриптов. Графические компоненты (например, график цен с нанесёнными на него индикаторами) корректно отрисовывались и обновлялись. В Safari на macOS не было выявлено критических проблем: все функции (регистрация, получение данных, расчет индикаторов и вывод сигналов) работали исправно, хотя потребовалась небольшая корректировка CSS-стилей для полного соответствия отображения некоторой типографии особенностям движка WebKit. После внесения этих правок интерфейс в Safari стал аналогичен другим браузерам. Ни в одном из протестированных браузеров не обнаружено ошибок в функциональности. Таким образом, кроссбраузерное тестирование подтвердило совместимость и корректную работу веб-приложения в различных браузерах, что гарантирует всем пользователям стабильный опыт независимо от выбранной платформы. Результат тестирования в таблице 8.

Таблица 8 – Результаты кроссбраузерного тестирования

Название теста	**Шаги**	**Ожидаемый результат**	**Тест пройден?**
Работа приложения в Google Chrome	1. Открыть веб-приложение в браузере Google Chrome. 2. Последовательно проверить основные функции: регистрацию пользователя, загрузку и отображение котировок, расчёт индикаторов, генерацию сигналов и вывод рекомендаций.	Во всех проверенных функциях приложение работает корректно в Chrome: интерфейс отображается без искажений, элементы управления функционируют, данные загружаются и обновляются, никаких ошибок или отклонений в работе нет.	Да
Работа приложения в Mozilla Firefox	1. Открыть веб-приложение в браузере Mozilla Firefox. 2. Выполнить аналогичный набор действий: регистрация нового пользователя, просмотр котировок и графиков, получение технических индикаторов, проверка сигналов и рекомендаций.	Приложение полнофункционально в Firefox: все страницы отображаются правильно, стиль и разметка сохранены; функциональность работает без сбоев.	Да
Работа приложения в Safari	1. Запустить приложение в браузере Safari. 2. Проверить отображение интерфейса и выполнение ключевых функций приложения.	В браузере Safari веб-приложение работает штатно: внешний вид соответствует ожиданиям, верстка не нарушена; все основные возможности функционируют корректно.	Да

Продолжение таблицы 8

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
4	Работа приложения в Microsoft Edge	1. Открыть приложение в браузере Microsoft Edge. 2. Провести тестирование основных разделов и функций.	В Edge приложение отображается и работает аналогично другим браузерам: совместимость полная, различий в поведении нет, все функции исполняются правильно, проблемы совместимости не обнаружены.	Да

4.7. Выводы по главе

Результаты тестирования подтверждают, что разработанное веб-приложение для технического анализа криптовалют соответствует требованиям, предъявленным к его функциональности, надёжности и совместимости. В ходе комплексной проверки были успешно протестированы ключевые модули системы: регистрация пользователей, получение данных с биржи Binance, вычисление технических индикаторов, логика генерации торговых сигналов и формирование пользовательских рекомендаций. Все предусмотренные тестовые сценарии завершились положительно.

Также было проведено кроссбраузерное тестирование, показавшее полную совместимость интерфейса и функциональности в современных настольных браузерах, включая Google Chrome, Mozilla Firefox, Microsoft Edge и Safari.

Таким образом, приложение прошло все этапы функционального и визуального тестирования, подтвердив свою готовность к реальному использованию и дальнейшему расширению.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной выпускной квалификационной работы достигнута поставленная цель – разработано веб-приложение для технического анализа криптовалют, удовлетворяющее современным требованиям к функциональности и удобству использования. Все поставленные задачи были успешно решены: проведен анализ предметной области и существующих решений, обоснован выбор оптимального стека технологий (Node.js, MongoDB и др.), спроектирована клиент-серверная архитектура приложения, реализован прототип веб-приложения с основными функциями технического анализа и выполнено его тестирование на реальных данных. Разработанное приложение позволяет интерактивно отображать исторические ценовые данные криптовалют, вычислять ключевые технические индикато-ры и визуализировать результаты в удобной для пользователя форме.

Созданный программный продукт обладает практической ценностью, поскольку предоставляет трейдерам и инвесторам инструмент для оперативного анализа рынка криптовалют. Веб-интерфейс приложения доступен через браузер, что делает использование системы возможным с любого устройства без установки специализированного ПО. Реализация серверной части на Node.js и хранение данных в MongoDB обеспечивают достаточную производительность и масштабируемость, а использование JavaScript-библиотек для визуализации позволяет наглядно представить результаты анализа. Таким образом, итоговый результат работы отвечает заявленным требованиям и подтверждает актуальность выбранной темы: пользователи получают возможность выполнять технический анализ криптовалютных рынков быстро и эффективно с помощью разработанного веб-приложения.

Перспективы дальнейшего развития проекта включают следующие направления:

Расширение функциональности анализа. Внедрение других разделов технического анализа позволит углубить аналитические возможности приложения.

Поддержка новых данных и источников. Интеграция данных с разных криптовалютных бирж и поддержка большего числа криптовалют расширят охват приложения.

Усовершенствование пользовательского интерфейса. Адаптация интерфейса под мобильные устройства, создание настраиваемых панелей и виджетов, повысит удобство работы пользователей с приложением.

Сбор информации из базы данных для улучшения точности сигналов, на основе пользовательских настроек инструментов технического анализа.

Применение методов ИИ и машинного обучения. В будущем возможно внедрение модулей прогнозирования на основе исторических данных (например, с помощью нейронных сетей), что даст пользователям инструменты для предсказания движения цен и более глубокой аналитики.

Реализация перечисленных улучшений позволит значительно повысить ценность и конкурентоспособность разработанного веб-приложения. В целом, проведенная работа заложила основу для дальнейших исследований и разработок в области приложений технического анализа криптовалют, а полученные результаты могут быть использованы как базис для создания более сложных аналитических платформ и сервисов.

.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Эдвардс, Р.Д. Технический анализ трендов на фондовом рынке / Р.Д. Эдвардс, Дж. Маги. – Спрингфилд (шт. Массачусетс): Stock Trend Service, 1948. – 431 с.
2. Элдер, А. Как играть и выигрывать на бирже / А. Элдер. – СПб.: Питер, 2020. – 368 с.
3. Мэрфи, Дж.Дж. Технический анализ фьючерсных рынков: теория и практика / Дж. Дж. Мэрфи. – М.: Сокол, 1996. – 592 с.
4. Соловьев, А.Ю. Информационные технологии в анализе и прогнозировании финансовых рынков / А.Ю. Соловьев. – М.: Финансы и статистика, 2022. – 272 с.
5. AlgosOne [Электронный ресурс]. – Режим доступа: https://algosone.ai, свободный. – Дата обращения: 18.01.2025.
6. Algoriz [Электронный ресурс]. – Режим доступа: https://algoriz.com, свободный. – Дата обращения: 19.01.2025.
7. Zignaly [Электронный ресурс]. – Режим доступа: https://zignaly.com, свободный. – Дата обращения: 20.01.2025.
8. TradingView [Электронный ресурс]. – Режим доступа: https://www.tradingview.com, свободный. – Дата обращения: 21.01.2025.
9. Binance API [Электронный ресурс]. – Режим доступа: https://developers.binance.com/docs/binance-spot-api-docs, свободный. – Дата обращения: 26.02.2025.
10. technicalindicators [Электронный ресурс]. – Режим доступа: https://www.npmjs.com/package/technicalindicators, свободный. – Дата обращения: 28.02.2025.
11. JavaScript: Полное руководство [Электронный ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Web/JavaScript, свободный. – Дата обращения: 01.03.2025.

12. W3Schools: HTML, CSS, JavaScript Tutorials [Электронный ресурс]. – Режим доступа: https://www.w3schools.com/, свободный. – Дата обращения: 02.03.2025.
13. Node.js: Документация [Электронный ресурс]. – Режим доступа: https://nodejs.org/en/docs, свободный. – Дата обращения: 03.03.2025.
14. Express.js: Документация [Электронный ресурс]. – Режим доступа: https://expressjs.com, свободный. – Дата обращения: 04.03.2025.
15. MongoDB: Документация [Электронный ресурс]. – Режим доступа: https://www.mongodb.com/docs, свободный. – Дата обращения: 05.03.2025.

ПРИЛОЖЕНИЕ А

Листинг А.1 – HTML и JavaScript страницы home.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Страница с кнопками и логотипом</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <div id="loader">
    <div id="loader-content">
        <img  src="C:\\Users\\Александр\\OneDrive\\Рабочий  стол\\крипта\\Фотографии  для
проекта сайта\\IndicanorLoader.gif" alt="Loader Image">
      <br>
      <span id="progress">0%</span>
    </div>
  </div>

  <div class="header">
     <img  src="C:\Users\Александр\OneDrive\Рабочий  стол\крипта\Фотографии  для  проекта
сайта\эмблема.png" alt="Логотип" class="logo" width="100">
    <div class="button-container">
      <button class="button"><a href="trading.html" style="text-decoration: none; color:
black;">ОБУЧЕНИЕ</a></button>
       <button  class="button"><a  href="technologies.html"  style="text-decoration:  none;
color: black;">ТЕХНОЛОГИИ</a></button>
      <button class="button"><a href="company.html" style="text-decoration: none; color:
black;">КОМПАНИЯ</a></button>
       <button class="button"><a href="singin.html" style="text-decoration: none; color:
black;">РЕГИСТРАЦИЯ</a></button>
       <button class="button"><a href="start.html" style="text-decoration: none; color:
black;">НАЧАТЬ</a></button>

        <form>
          <select style="border: none;">
            <option>
              <button>RUS</button>
            </option>
            <option>
              <button>ENG</button>
            </option>
            <option>
              <button>DEU</button>
            </option>
            <option>
              <button>FRA</button>
            </option>
            <option>
              <button>CHS</button>
            </option>
            <option>
              <button>TRK</button>
```

Продолжение приложения А

Продолжение листинга А.1

```
     </option>
          </select>

        </form>
      </button>
    </div>
  </div>
  <!-- Основное содержимое страницы -->

  <div class="footer">
    <div class="footer-content">
      <div class="footer-left">
         <img src="C:\Users\Александр\OneDrive\Рабочий стол\крипта\Фотографии для проекта
сайта\apple.png" alt="Логотип" width="100">
         <img src="C:\Users\Александр\OneDrive\Рабочий стол\крипта\Фотографии для проекта
сайта\goooogle_play.png" alt="Фотография 1" width="100">
        <pre>
 Наша компания занимается разработкой
 инновационных технологий для улучшения
 жизни людей.
        </pre>
      </div>
      <div class="footer-right">
        <div class="footer-right-block">
          <h2>Технологии</h2>
          <button class="button">Узнать больше</button>
          <button class="button">Присоединиться</button>
          <button class="button">Скачать</button>
          <button class="button">Контакты</button>
        </div>
        <div class="footer-right-block">
          <h2>Компания</h2>
          <button class="button">О нас</button>
          <button class="button">Карьера</button>
          <button class="button">Партнеры</button>
          <button class="button">Новости</button>
        </div>
        <div class="footer-right-block">
          <h2>Поддержка</h2>
          <button class="button">FAQ</button>
          <button class="button">Контакты</button>
          <button class="button">Форум</button>
          <button class="button">Техподдержка</button>
        </div>
      </div>
    </div>
  </div>
  <script>
    let progress = 0;
    const loader = document.getElementById('loader');
    const progressSpan = document.getElementById('progress');

    function updateProgress() {
      progress += 5; // Увеличение прогресса на 5%
      progressSpan.textContent = `${progress}%`;

      if (progress < 100) {
        setTimeout(updateProgress, 100); // Обновление прогресса каждые 0.1 секунды
      } else {
```

Продолжение приложения А

Окончание листинга А.1

```
        // Скрытие индикатора загрузки после полной загрузки страницы

        loader.style.display = 'none';
      }
    }

    updateProgress();

    // Альтернативно, можно использовать событие window.onload для скрытия индикатора
    // window.onload = function() {
    // loader.style.display = 'none';
    // };
  </script>

</body>
</html>
```

Листинг А.2 – HTML и JavaScript страницы company.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Компания</title>
    <link rel="stylesheet" href="companyStyle.css" />
</head>
<body>
    <div id="loader">
        <div id="loader-content">
            <img src="C:\\Users\\Александр\\OneDrive\\Рабочий стол\\крипта\\Фотографии для
проекта сайта\\IndicanorLoader.gif" alt="Loader Image">
            <br>
            <span id="progress">0%</span>
        </div>
    </div>
    <div id="loading-curtain"></div>
                                        <button                    class="back-button"
onclick="window.location.href='index.html'">&#8592;</button>
    <div class="container">
        <div class="title">Компания</div>

        <div class="transparent-block-full">
           <p>Данная программа – лидер в области автоматизированного анализа криптовалютных
рынков.
              Она использует инновационные решения, которые помогают трейдерам и инвесторам
                  принимать обоснованные решения, основываясь на точных и своевременных
данных</p>
        </div>
        <div class="blocks-container">
            <button class="block-button">
                <a href="reviews.html"  style="text-decoration: none; color: black;">
                <h3>ОТЗЫВЫ</h3>
                <p>Ознакомьтесь с отзывами пользователей нашей программой</p>
                </a>
            </button>

            <button class="block-button">
```

Продолжение приложения А

Окончание листинга А.2

```
                <a href="questions.html"  style="text-decoration: none; color: black;">
                <h3>ЧАСТО ЗАДАВАЕМЫЕ ВОПРОСЫ</h3>
                <p>Есть вопрос? У нас есть ответ!</p>
                </a>
            </button>
            <button class="block-button">
                <a href="feedback.html"  style="text-decoration: none; color: black;"></a>
                <h3>ОБРАТНАЯ СВЯЗЬ</h3>
                <p>Свяжитесь с нами, мы будем рады вас услышать.</p>

            </button>
            <button class="block-button">
                <a href="developments.html"  style="text-decoration: none; color: black;">
                <h3>РАЗРАБОТКИ</h3>
                <p>Посмотрите, что у нас в разработке.</p>
                </a>
            </button>
            <button class="block-button">
                      <a  href="help_center.html"   style="text-decoration:  none;  color:
black;"></a>
                <h3>СПРАВОЧНЫЙ ЦЕНТР</h3>
                <p>Добро пожаловать в наш Справочный центр</p>
            </button>
            <button class="block-button">
                <a href="news.html"  style="text-decoration: none; color: black;">
                <h3></h3>
                <p></p>
                </a>
            </button>
        </div>
        <a href="index.html" class="button">Перейти на главную</a>
    </div>

    <script>
        let progress = 0;
        const loader = document.getElementById('loader');
        const progressSpan = document.getElementById('progress');
        const loadingCurtain = document.getElementById('loading-curtain');

        function updateProgress() {
            progress += 5; // Увеличение прогресса на 5%
            progressSpan.textContent = `${progress}%`;
            if (progress < 100) {
               setTimeout(updateProgress, 100); // Обновление прогресса каждые 0.1 секунды
            } else {
                // Скрытие индикатора загрузки и шторки после полной загрузки страницы
                loader.style.display = 'none';
                loadingCurtain.classList.remove('show');
            }
        }

        // Показываем шторку при загрузке страницы
        document.addEventListener('DOMContentLoaded', function() {
            loadingCurtain.classList.add('show');
            updateProgress();
        });
    </script>
</body>

</html>
```

Листинг А.3 – HTML и JavaScript страницы register.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Регистрация</title>
    <link rel="stylesheet" href="singingStyle.css" />
</head>
<body>
    <div id="loader">
        <div id="loader-content">
            <img src="IndicanorLoader.gif" alt="Loader Image" />
            <br />
            <span id="progress">0%</span>
        </div>
    </div>

       <button   class="back-button"   onclick="window.location.href='index.html'">&#8592;
Назад</button>

    <div class="container">
        <div class="left-side">
            <img src="logo.png" alt="Логотип компании" />
            <div class="form-container">
                <h2>Регистрация пользователя</h2>
                <form id="registrationForm">
                    <label for="login">Логин:</label>
                    <input type="text" id="login" name="login" required />

                    <label for="password">Пароль:</label>
                    <input type="password" id="password" name="password" required />

                    <label for="email">Электронная почта:</label>
                    <input type="email" id="email" name="email" required />

                    <button type="submit">Зарегистрироваться</button>
                </form>
                <div id="message"></div>
            </div>
        </div>
        <div class="right-side"></div>
    </div>

    <script src="register.js"></script>
</body>
</html>

document.getElementById('registrationForm').addEventListener('submit', function(e) {
    e.preventDefault();

    const login = document.getElementById('login').value.trim();
    const password = document.getElementById('password').value.trim();
    const email = document.getElementById('email').value.trim();
    const messageDiv = document.getElementById('message');

    if (!login || !password || !email) {
        messageDiv.textContent = 'Пожалуйста, заполните все поля.';
        messageDiv.style.color = 'red';
```

Продолжение приложения А

Окончание листинга А.3

```
 return;
    }

    fetch('/register', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ login: login, password: password, email: email })
    })
      .then(response  =>  response.json().then(data  =>  ({status:  response.status,  body:
data})))
    .then(({status, body}) => {
        if (status === 200 && body.status === 'success') {
            messageDiv.textContent = body.message;
            messageDiv.style.color = 'green';
            document.getElementById('registrationForm').reset();
        } else {
            messageDiv.textContent = body.message || 'Ошибка при регистрации.';
            messageDiv.style.color = 'red';
        }
    })
    .catch(error => {
        console.error('Ошибка:', error);
        messageDiv.textContent = 'Произошла ошибка. Попробуйте позже.';
        messageDiv.style.color = 'red';
    });
});
```

Листинг А.4 – HTML и JavaScript страницы technologies.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Технологии</title>
    <link rel="stylesheet" href="technologiesStyle.css">
</head>
<body>

    <div id="loader">
        <div id="loader-content">
            <img src="C:\\Users\\Александр\\OneDrive\\Рабочий стол\\крипта\\Фотографии для
проекта сайта\\IndicanorLoader.gif" alt="Loader Image">
            <br>
            <span id="progress">0%</span>
        </div>
    </div>

<button class="back-button" onclick="window.location.href='index.html'">&#8592;</button>

<div class="content">
    <div class="text-block">
        <h2>ПРОРУСК НА РЫНОК КРИПТОВАЛЮТ</h2>
        <p>У каждого должна быть возможность обеспечить свое финансовое будущее.
        Глобальные рынки должны быть открыты для всех, а не только для тех,
        кто обладает обширными финансовыми знаниями, часами свободного времени
        для мониторинга рыночной активности или значительными ресурсами,
```

Продолжение приложения А

```
        необходимыми для того, чтобы стать аккредитованным инвестором в
        мире криптовалют.</p>
        <p> Объединяя машинную логику с реальным финансовым
        опытом, наши клиенты получают лучшее из обоих миров – скорость,
        эффективность и результативность.</p>
        <p>  Наша программа работает круглосуточно,
        отслеживая монеты которыми вы интересуетесь, чтобы гарантировать,
        что если произойдет непредвиденное рыночное событие, вы сможете
        преумножить или сберечь ваши активы.
        </p>
    </div>
    <div class="image-block">
          <img  src="C:\\Users\\Александр\\OneDrive\\Рабочий  стол\\крипта\\Фотографии  для
проекта сайта\\paint_traiding_lockbig.png" alt="Фотография 1">
    </div>
</div>

<div class="content">
    <div class="text-block">
        <h2>МОЩНАЯ ТЕХНОЛОГИЯ С ИНДИВИДУАЛЬНЫМ ПОДХОДОМ</h2>
        <p>Программа предназначена для автоматизации методов технического анализа.
             Она автоматически распознаёт графические модели, уровни поддержки и
             сопротивления, линии тренда и технические индикаторы такие как “MACD”,
             “RSI”  “BB”. Эти знания в области технического анализа  позволяет
             определить не только направление движения цены, но и его цели. В отличии
             от ручного, при автоматическом анализе криптовалют его качество не зависит
              от опыта трейдера и не подвержено влиянию психологии.</p>
        <p>  Процесс технического анализа в программе полностью автоматизирован и
            происходит по жестким критериям, которые так же можно скорректировать
            для более успешной торговли с помощью редактора. При появлении точки
            входа в монету программа сразу же выдаёт торговый сигнал и отправляет
             вам его на почту.</p>
        <p>  Программа позволяет работать с любыми инструментами и таймфреймами,
             поэтому она подойдет как внутридневным спекулянтам, так и долгосрочным
              инвесторам. </p>
    </div>
    <div class="image-block">
          <img  src="C:\\Users\\Александр\\OneDrive\\Рабочий  стол\\крипта\\Фотографии  для
проекта сайта\\paint_traiding_lock2upgreit.png" alt="Фотография 3">
    </div>
</div>

<div class="text-center">
    <h2>ПОЛУЧИТЕ ПРЕИМУЩЕСТВО НА РЫНКЕ</h2>
</div>

<div class="advantage-block">
    <div class="advantage-item">
        <h3>ИНВЕСТИЦИИ ДЛЯ ВСЕХ</h3>
        <p>Для получения прибыли не нужны знания в области технического
             анализа и криптовалют. Рынок инвестирования будет открыты
              для всех, а не только для тех, кто обладает обширными финансовыми знаниями.
</p>
    </div>
    <div class="advantage-item">
        <h3>СКОРОСТЬ И ЭФФЕКТИВНОСТЬ</h3>
        <p> Программа анализирует огромный объем информации для нескольких криптовалют
             одновременно, используя большое количество методом технического анализа ,
             что позволяет нам быстро выявлять новые рыночные тенденции и движущие силы</p>
```

Окончание листинга А.4

```
</div>
    <div class="advantage-item">
        <h3>ВЫСОКАЯ ТОЧНОСТЬ</h3>
        <p>Процесс анализа полностью автоматизирован и происходит по жестким критериям,
             которые были проверены на реальном рынке. Регулярные обновления и улучшения
                  алгоритмов в программе, с каждым разом будут увеличивать точность
прогнозов.</p>
    </div>
</div>

<script>
    let progress = 0;
    const loader = document.getElementById('loader');
    const progressSpan = document.getElementById('progress');

    function updateProgress() {
        progress += 5; // Увеличение прогресса на 10%
        progressSpan.textContent = `${progress}%`;

        if (progress < 100) {
            setTimeout(updateProgress, 100); // Обновление прогресса каждые 0.05 секунды
        } else {
            // Скрытие индикатора загрузки после полной загрузки страницы
            loader.style.display = 'none';
        }
    }
    updateProgress();

    // Альтернативно, можно использовать событие window.onload для скрытия индикатора
    // window.onload = function() {
    //     loader.style.display = 'none';
    // };
</script>

</body>
</html>
```

Листинг А.5 – HTML и JavaScript страницы trading.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Траидинг</title>
  <link rel="stylesheet" href="tradingStyle.css">
</head>
<body>
  <div id="loader">
    <div id="loader-content">
         <img  src="C:\\Users\\Александр\\OneDrive\\Рабочий  стол\\крипта\\Фотографии  для
проекта сайта\\IndicanorLoader.gif" alt="Loader Image">
      <br>
      <span id="progress">0%</span>
    </div>
  </div>
  <button class="back-button" onclick="window.location.href='index.html'">&#8592;</button>
```

Продолжение приложения А

Продолжение листинга А.5

```
<div class="container">

    <div class="title">ОБУЧЕНИЕ</div>

    <div class="transparent-block">
        <b>ЧТО ТАКОЕ ТЕХНИЧЕСКИЙ АНАЛИЗ?</b>
       <p>Технический анализ — это метод прогнозирования будущего движения цен на финансовых рынках на основе изучения исторических данных о ценах и объемах. В отличие от фундаментального анализа, который фокусируется на экономических, финансовых и других качественных факторах, технический анализ использует графики и математические индикаторы для выявления закономерностей и трендов, такие как уровни поддержки и сопротивления, линии тренда, фигуры графического анализа, технические индикаторы.</p>
        <p>Преимуществом технического анализа является возможность использования на любом финансольном рынке (акции, валюты, товары и т.д.). Он основан на математических расчетах и графических моделях, что снижает влияние эмоций на принятие решений.</p>

        <a href="start.html" class="button">Перейти на Start</a>
      </div>



    <div class="transparent-block" style="margin-top: 50px;">
      <b>ТОРГОВЛЯ С ИСПОЛЬЗОВАНИЕМ АВТОМАТИЗИРОВАННОГО АНАЛИЗА КРИПТОВАЛЮТ</b>
       <p>Автоматизированный анализ способен обрабатывать огромный объем технических и фундаментальных рыночных данных в режиме реального времени, относящихся к широкому спектру финансовых рынков, таких как криптовалюты, акции, сырьевые товары и облигации.</p>
      <p> Данная программа способна выполнять широкий спектр действий, включая исторический анализ цен и объемов, оценку рисков, создание сигналов, рекомендации по входу и выходу.</p>
       <p> Регулярные обновления улучшают точность анализа инструментов.</p>
      <a href="start.html" class="button">Перейти на Start</a>
    </div>
    <div class="title"></div>
    <div class="title" style="margin-top: 50px;">ИНСТРУМЕНТЫ ТЕХНИЧЕСКОГО АНАЛИЗА</div>

     <div class="title" style="text-align: center; font-size: 18px;">Технический анализ использует различные инструменты для анализа графиков и выявления торговых возможностей:</div>
    <div class="blocks-container">
      <div class="block">
        <b>ГРАФИКИ ЦЕН (Японские свечи)</b>
          <p>Существует множество моделей японских свечей, которые могут указывать на возможное изменение тренда или продолжение текущего тренда.</p>
             <a  href="C:\Users\Александр\OneDrive\Рабочий  стол\крипта\обучение\pdf  для Трейдинг\Японские свечи.docx" download class="download-button">Скачать PDF</a>
      </div>
      <div class="block">
        <b>УРОВНИ ПОДДЕРЖКИ И СОПРОТИВЛЕНИЯ</b>
        <p>Уровни цен, на которых цена, как правило, останавливается или отскакивает.</p>
            <a  href="C:\Users\Александр\OneDrive\Рабочий  стол\крипта\обучение\pdf  для Трейдинг\УРОВНИ ПОДДЕРЖКИ И СОПРОТИВЛЕНИЯ.docx" download class="download-button">Скачать PDF</a>
      </div>
      <div class="block">
        <b>ЛИНИИ ТРЕНДА</b>
         <p>Линии, соединяющие последовательные максимумы или минимумы цены, показывающие направление тренда.</p>
             <a  href="C:\Users\Александр\OneDrive\Рабочий  стол\крипта\обучение\pdf  для Трейдинг\ЛИНИИ ТРЕНДА.docx" download class="download-button">Скачать PDF</a>
```

Продолжение приложения А

Окончание листинга А.5

```
      </div>
      <div class="block">
        <b>ФИГУРЫ ГРАФИЧЕСКОГО АНАЛИЗА</b>
        <p>Определенные паттерны на графике, которые могут указывать на возможное изменение тренда или продолжение текущего тренда (например, "голова и плечи", "двойное дно", "треугольники").</p>
            <a href="C:\Users\Александр\OneDrive\Рабочий стол\крипта\обучение\pdf для Трейдинг\Фигуры технического анализа.docx" download class="download-button">Скачать PDF</a>
      </div>
      <div class="block">
        <b>ТЕХНИЧЕСКИЕ ИНДИКАТОРЫ</b>
        <p> Математические расчеты, основанные на исторических данных о ценах и объемах, которые используются для выявления торговых сигналов (например, скользящие средние, RSI, MACD).</p>
            <a href="C:\Users\Александр\OneDrive\Рабочий стол\крипта\обучение\pdf для Трейдинг\ТЕХНИЧЕСКИЕ ИНДИКАТОРЫ.docx" download class="download-button">Скачать PDF</a>
      </div>
      <div class="title"></div>
    </div>
  </div>
  <script>
    let progress = 0;
    const loader = document.getElementById('loader');
    const progressSpan = document.getElementById('progress');

    function updateProgress() {
      progress += 5; // Увеличение прогресса на 5%
      progressSpan.textContent = `${progress}%`;

      if (progress < 100) {
        setTimeout(updateProgress, 100); // Обновление прогресса каждые 0.1 секунды
      } else {
        // Скрытие индикатора загрузки после полной загрузки страницы
        loader.style.display = 'none';
      }
    }
    updateProgress();

    // Альтернативно, можно использовать событие window.onload для скрытия индикатора
    // window.onload = function() {
    // loader.style.display = 'none';
    // };
  </script>

</body>
</html>
```

Листинг А.6 – HTML и JavaScript главной страницы index.html

```
<!doctype html>
<html lang="ru">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="Кто ты, воин?... Ахиллес, сын Пелея!">
    <meta name="generator" content="Hugo 0.84.0">
```

Продолжение приложения А

Продолжение листинга А.6

```
    <title>Криптовалюты</title>
    <!-- Bootstrap core CSS -->
     <link  href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"                                                              integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">

    <style>
      .bd-placeholder-img {
        font-size: 1.125rem;
        text-anchor: middle;
        -webkit-user-select: none;
        -moz-user-select: none;
        user-select: none;
      }

      @media (min-width: 768px) {
        .bd-placeholder-img-lg {
          font-size: 3.5rem;
        }
      }
    </style>


    <!-- Custom styles for this template -->
    <link href="index.css" rel="stylesheet">
  </head>
  <body>

<header class="navbar navbar-dark sticky-top bg-dark flex-md-nowrap p-0 shadow">
  <a class="navbar-brand col-md-3 col-lg-2 me-0 px-3" href="#">Криптовалюты</a>
  <!-- Кнопка меню при маленьклм разрешении экрана -->
  <button class="navbar-toggler position-absolute d-md-none collapsed" type="button" data-
bs-toggle="collapse"   data-bs-target="#sidebarMenu"   aria-controls="sidebarMenu"   aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

    <input   id="cryptName"   class="form-control   form-control-dark   w-100"   type="text"
placeholder="Введите название криптовалюты (например, BTCUSDT)" aria-label="Search">
  <div class="navbar-nav">
    <div class="nav-item text-nowrap">
      <a id="LinkViewChart" class="nav-link px-3" role="button">Показать график</a>
    </div>
  </div>
</header>

<div class="container-fluid">
  <div class="row">
    <nav id="sidebarMenu" class="col-md-3 col-lg-2 d-md-block bg-light sidebar collapse">
      <div class="p-1">

        <h6 class="sidebar-heading d-flex justify-content-between align-items-center px-3
mt-4 mb-1 text-muted">
          <span>Выберите тип сделки</span>
        </h6>

       <select id="TransactionType" class="form-select btn-outline-secondary form-select-
lg mb-3" aria-label="Large select example">
```

Продолжение приложения А

Продолжение листинга А.6

```
          <option value="1" selected>Долгосрочная</option>
          <option value="2">Краткосрочная</option>
          <option value="3">Скальпинг</option>
        </select>

        <h6 class="sidebar-heading d-flex justify-content-between align-items-center px-3
mt-4 mb-1 text-muted">
          <span>Выберите время сделки</span>
        </h6>

       <select id="TransactionTime" class="form-select btn-outline-secondary form-select-
lg mb-3" aria-label="Large select example">

        </select>

        <h6 class="sidebar-heading d-flex justify-content-between align-items-center px-3
mt-4 mb-1 text-muted">
          <span>Выберите тип торговли</span>
        </h6>

           <select class="form-select btn-outline-secondary form-select-lg mb-3" aria-
label="Large select example">
          <option value="1" selected>Short</option>
          <option value="2">Long</option>
        </select>
        <div class="d-grid gap-2">

                    <button    id="analyze"    type="button"    class="btn    btn-outline-
secondary">Анализировать</button>
      </div>
      </div>
    </nav>

    <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
       <div class="d-flex justify-content-between flex-wrap flex-md-nowrap align-items-
center pt-3 pb-2 mb-3 border-bottom">
        <h1 id="TitleChart" class="h2">Dashboard</h1>
      </div>

      <canvas class="my-4 w-100" id="myChart" width="900" height="380"></canvas>

      <div id="resuilTable" class="container d-none">
        <h2>Результат анализа</h2>
        <table class="table align-middle">
          <thead>
            <tr>
              <th scope="col">Тип</th>
              <th scope="col">Аргументы</th>
              <th scope="col">Результат</th>
              <th scope="col"></th>
            </tr>
          </thead>
          <tbody>

          </tbody>
        </table>
        <div class="row mb-3">
               <button  type="button"  class="btn  btn-outline-secondary  btn-lg"><h4>No
Signal</h4></button>
```

Продолжение листинга А.6

```
        </div>
      </div>

    </main>
  </div>
</div>


                                                                                <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
      <script src="https://cdn.jsdelivr.net/npm/feather-icons@4.28.0/dist/feather.min.js"
integrity="sha384-uO3SXW5IuS1ZpFPKugNNWqTZRRglnUJK6UAZ/gxOX80nxEkN9NcGZTftn6RzhGWE"
crossorigin="anonymous"></script>
           <script   src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4/dist/Chart.min.js"
integrity="sha384-zNy6FEbO50N+Cg5wap8IKA4M/ZnLJgzc6w2NqACZaK0u0FXfOWRRJOnQtpZun8ha"
crossorigin="anonymous"></script>

      <script src="request.js"></script>
      <script src="chart.js"></script>
      <script src="table.js"></script>
      <script src="updateTimeFrame.js"></script>
      <script src="index.js"></script>

<div class="container mt-4" style="margin-left: 0px;">
  <h5>Выбор индикаторов для итоговой рекомендации:</h5>
     <div    id="indicatorToggleGroup"    class="btn-group    mb-4"    role="group"    aria-
label="Индикаторы">
          <button    type="button"    class="btn    btn-outline-success    active"    data-
indicator="macd">MACD</button>
          <button    type="button"    class="btn    btn-outline-success    active"    data-
indicator="rsi">RSI</button>
          <button    type="button"    class="btn    btn-outline-success    active"    data-
indicator="sma">SMA</button>
          <button    type="button"    class="btn    btn-outline-success    active"    data-
indicator="ema">EMA</button>
          <button    type="button"    class="btn    btn-outline-success    active"    data-
indicator="wma">WMA</button>

  </div>
</div>

<script>
  const indicatorSelection = {
    macd: true,
    rsi: true,
    sma: true,
    ema: true,
    wma: true
  };

  document.addEventListener('DOMContentLoaded', () => {
    document.querySelectorAll('#indicatorToggleGroup button').forEach(btn => {
      btn.addEventListener('click', () => {
        const ind = btn.dataset.indicator;
        indicatorSelection[ind] = !indicatorSelection[ind];
        btn.classList.toggle('active');
        if (indicatorSelection[ind]) {
```

Продолжение приложения А

Продолжение листинга А.6

```
          btn.classList.remove('btn-outline-danger');
          btn.classList.add('btn-outline-success');
        } else {
          btn.classList.remove('btn-outline-success');
          btn.classList.add('btn-outline-danger');
        }
      });
    });
  });
</script>

</body>
</html>
var intervalId=null;
var timeRequest = GetTimeRequest("1d");
var visibleTable = false;
// Добавление обработчиков
document.getElementById("TransactionType").addEventListener('change', (element) =>{
  updateTimeFrameOptions(element.target.value)
});
document.getElementById("LinkViewChart").addEventListener('click',async () =>{
    var cryptoSymbol = document.getElementById("cryptName").value;
    //Если строка поиска пуста, то берём последней поиск
    if(cryptoSymbol===""){
      cryptoSymbol = document.getElementById('TitleChart').textContent;
    }

  var timeFrame = document.getElementById("TransactionTime").value;
  timeRequest= GetTimeRequest(timeFrame);
  const isCreate = createChart(cryptoSymbol,timeFrame);

  if(!visibleTable||!isCreate){

    return;
  }

  var isFail  =await analyze(cryptoSymbol,timeFrame);

  if(isFail){
    return;
  }
  //удадление интервала
  if(intervalId){
    clearInterval(intervalId);
  }
  intervalId = setInterval(analyze,timeRequest,cryptoSymbol,timeFrame);
});

document.getElementById("analyze").addEventListener('click',async () =>{
  var cryptoSymbol = document.getElementById('TitleChart').textContent;
  var timeFrame = document.getElementById("TransactionTime").value;
  timeRequest= GetTimeRequest(timeFrame);
  var isFail  =await analyze(cryptoSymbol,timeFrame);
  if(isFail){
    return;
  }
  //удадление интервала
  if(intervalId){
    clearInterval(intervalId);
```

Продолжение приложения А

Продолжение листинга А.6

```
  }

  intervalId = setInterval(analyze,timeRequest,cryptoSymbol,timeFrame);
  visibleTable=true;
});

  async function analyze(cryptoSymbol,timeFrame){
    console.log(`analyze - (${cryptoSymbol}, ${timeFrame})`);
    var data = await AnalyzeRequest(cryptoSymbol,timeFrame);
    if(!data){
      console.log(`analyze - (${cryptoSymbol}, ${timeFrame}) - fail`);
      return true;
    }
    createTable(data);
return false;
  }

//инициализация временных рамок
updateTimeFrameOptions(typeTimeFrames.longTerm);
//график
createChart('BTCUSDT','1m');
```

Листинг А.7 – код JavaScript выполняющий GET-запроса к API для получения данных о криптовалюте из request.js

```
const GetRequest = async () =>{
    const response = await fetch("api/crypto", {
        method: "GET",
        headers: { "Accept": "application/json", "Content-Type": "application/json" },
    });
    if (!response.ok) throw new Error('Network response was not ok');
    const data = await response.json();
    console.log(data);
};

const AnalyzeRequest = async (cryptoSymbol,timeFrame) =>{
  const response = await fetch("api/crypto/analyze", {
      method: "POST",
      headers: { "Accept": "application/json", "Content-Type": "application/json" },\
      body: JSON.stringify({
        crypto: cryptoSymbol,
        time: timeFrame
      })
  })
  .then((response)=>{
    if (response.ok) {
      return response.json();

    throw new Error('Ошибка запроса анализа!');
})
  .catch((error)=>{
    console.log(error);
  });
  return response;
};
```

Листинг А.8 – код JavaScript для создания и обновления графика из chart.js

```
let chart;
const createChart = async (cryptoSymbol, timeFrame) => {
                        var           response           =           await           fetch(
`/api/crypto/klines?crypto=${cryptoSymbol}&time=${timeFrame}`);
    if (!response.ok){
        var mes = await response.text();
        alert("Криптовалюта не найдена");
    return false;
    }
    if (chart) {
        chart.destroy(); // Уничтожаем предыдущий график, если он существует
    }

    var cryptoData = await response.json();

    var title =document.getElementById('TitleChart');
    title.textContent=cryptoSymbol.toUpperCase();

    var ctx = document.getElementById('myChart').getContext('2d');
    chart = new Chart(ctx, {
        type: 'line',
        data: {
            labels: cryptoData.labels,
            datasets: [{
                label: `Цена ${cryptoSymbol} - ${timeFrame}`,
                data: cryptoData.dataPoints,
                borderColor: 'rgba(75, 192, 192, 1)',
                backgroundColor: 'rgba(75, 192, 192, 0.2)',
                borderWidth: 1,
                fill: true,
            }]
        },
        options: {
            scales: {
                y: {
                    beginAtZero: false,
                    position: 'right' // Перемещаем ось Y на правую сторону
                },
                x: {
                    ticks: {
                        autoSkip: true,
                        maxTicksLimit: 20
                    }
                }
            }
        }
    });
    return true;
};
```

Листинг А.9 – код JavaScript для формирования таблицы с результатами анализа из table.js

```
var typeBtn = 0;

const createTable = (data) =>{
```

Продолжение приложения А

Продолжение листинга А.9

```
    var container = document.getElementById("resuilTable");

    var body = container.querySelector("tbody");

    const mcad = createMCAD(data.macd);
    const rsi =createRSI(data.rsi);
    const sma =createSMA(data.sma);
    const wma =createWMA(data.wma);
    var rowsHTML="";
    rowsHTML+=mcad.row;
    rowsHTML+=rsi.row;
    rowsHTML+=sma.row;
    rowsHTML+=wma.row;

    var indexs =
    [
        0,// 0 BUY
        0,// 1 SOLD
        0 // 2 No Signal
    ];

       if (typeof indicatorSelection === "undefined" || indicatorSelection.macd)
indexs[mcad.index]++;
       if (typeof indicatorSelection === "undefined" || indicatorSelection.rsi)
indexs[rsi.index]++;
       if (typeof indicatorSelection === "undefined" || indicatorSelection.sma)
indexs[sma.index]++;
       if (typeof indicatorSelection === "undefined" || indicatorSelection.wma)
indexs[wma.index]++;
    typeBtn = indexOf(indexs);
            rowsHTML += `<tr><th scope="row" style="padding-left: 24px; text-align:
left;"><strong style="font-size: 2rem;">+</strong></th><td></td><td></td></tr>`;
body.innerHTML = rowsHTML;

    var btnHTML = createButton(typeBtn);

    var divBTN = container.querySelector("div");

    divBTN.innerHTML =btnHTML;

    container.classList.remove("d-none");
};

const createMCAD = (data)=>{
    var rowData;
    if(data?.error){
        rowData = `<td colspan="3">${data.error}</td>`
    }else{
        var col1=
        `${data.args.fastPeriod} ${data.args.slowPeriod} ${data.args.signalPeriod}`;
        var col2=
        `MACD: ${data.data.macd}\n`+
        `Signal: ${data.data.signal}\n`+
        `Histogram: ${data.data.histogram}\n`+
        `Проверка: ${data.data.isEvent}`;
        var col3;
        var index;
        if(data.indicator=="BUY"){
            index = 0;
```

Продолжение приложения А

Продолжение листинга А.9

```
            col3=`<h3 class="text-success">BUY</h3>`;
        }else if (data.indicator=="SOLD"){
            index = 1;
            col3=`<h3 class="text-danger">SOLD</h3>`;
        }else{
            index = 2;
            col3=`<h3 class="text-secondary"><pre>No Signal</pre></h3>`;
        }
        rowData =
        `<td>${col1}</td>`+
        `<td><pre>${col2}</pre></td>`+
        `<td>${col3}</td>`;
    }
    var row = `<tr><th scope="row">MACD</th>${rowData}</tr>`;
    return {row,index};
};
const createRSI = (data)=>{
    var rowData;
    if(data?.error){
        rowData = `<td colspan="3">${data.error}</td>`
    }else{
        var col1=
        `${data.args.period}`;
        var col2=
        `RSI: ${data.data.rsi}`;
        var col3;
        var index;
        if(data.indicator=="BUY"){
            index = 0;
            col3=`<h3 class="text-success">BUY</h3>`;
        }else if (data.indicator=="SOLD"){
            index = 1;
            col3=`<h3 class="text-danger">SOLD</h3>`;
        }else{
            index = 2;
            col3=`<h3 class="text-secondary"><pre>No Signal</pre></h3>`;
        }
        rowData =
        `<td>${col1}</td>`+
        `<td><pre>${col2}</pre></td>`+
        `<td>${col3}</td>`;
    }
    var row = `<tr><th scope="row">RSI</th>${rowData}</tr>`;
    return {row,index};
};
const createSMA = (data)=>{
    var rowData;
    if(data?.error){
        rowData = `<td colspan="3">${data.error}</td>`
    }else{
        var col1=
        `${data.args.period1} ${data.args.period2} ${data.args.period3}`;
        var col2=
        `SMA 50: ${data.data.sma50}\n`+
        `SMA 100: ${data.data.sma100}\n`+
        `SMA 200: ${data.data.sma200}\n`+
        `Новый уровень поддержки / сопротивления:\n`+
                    `Поддержка    -    ${data.data.newSupportLevel},    Сопротивление    -
${data.data.newResistanceLevel}\n`+
```

Продолжение приложения А

Продолжение листинга А.9

```
        `${data.data.message}`;
        var col3;
        var index;
        if(data.indicator=="BUY"){
            index = 0;
            col3=`<h3 class="text-success">BUY</h3>`;
        }else if (data.indicator=="SOLD"){
            index = 1;
              col3=`<h3 class="text-danger">SOLD</h3>`;
        }else{
            index = 2;
            col3=`<h3 class="text-secondary"><pre>No Signal</pre></h3>`;
        }
        rowData =
        `<td>${col1}</td>`+
        `<td><pre>${col2}</pre></td>`+
        `<td>${col3}</td>`;
    }
    var row = `<tr><th scope="row">SMA</th>${rowData}</tr>`;
    return {row,index};
};
const createButton = (index)=>{
    var but;
    switch(index){
        case 0:
                    but  =`<button  type="button"  class="btn  btn-outline-success  btn-
lg"><h4>BUY</h4></button>`;
            break;
        case 1:
                     but  =`<button  type="button"  class="btn  btn-outline-danger  btn-
lg"><h4>SOLD</h4></button>`;
            break;
        default:
             but =`<button type="button" class="btn btn-outline-secondary btn-lg"><h4>No
Signal</h4></button>`;
            break;
    }
    return but;
}
const  indexOf=(a) =>{
    // 0 BUY
    // 1 SOLD
    // 2 No Signal
    if(a[0]===0&&a[1]===0){
        return 2;
    }

    if(a[0]===a[1]&&a[0]===a[2]){
        return typeBtn;
    }

    if(a[0]>a[1]){
        return 0;
    }else{
        return 1;
    }
}
const createWMA = (data)=>{
    var rowData;
```

Продолжение приложения А

Окончание листинга А.9

```
    if(data?.error){
        rowData = `<td colspan="3">${data.error}</td>`;
    } else {
        var col1 = `${data.args.shortPeriod} ${data.args.longPeriod}`;
        var col2 =
        `WMA ${data.args.shortPeriod}: ${data.data.wmaShort}\n` +
        `WMA ${data.args.longPeriod}: ${data.data.wmaLong}`;
        var col3;
        var index;
        if(data.indicator == "BUY"){
            index = 0;
            col3 = `<h3 class="text-success">BUY</h3>`;
        } else if(data.indicator == "SOLD"){
            index = 1;
            col3 = `<h3 class="text-danger">SOLD</h3>`;
        } else {
            index = 2;
            col3 = `<h3 class="text-secondary"><pre>HOLD</pre></h3>`;
        }
        rowData = `<td>${col1}</td><td><pre>${col2}</pre></td><td>${col3}</td>`;
    }
    var row = `<tr><th scope="row">WMA</th>${rowData}</tr>`;
    return {row, index};
};
```

Листинг А.10 – код JavaScript для выбора и обновления временных интервалов из updateTimeFrame.js

```
const typeTimeFrames = {
    scalping: '3',
    shortTerm: '2',
    longTerm: '1'
};

const timeFrames = {
    scalping: ['1m', '3m', '15m', '30m'],
    shortTerm: ['1h', '2h', '4h', '6h', '8h', '12h'],
    longTerm: ['1d', '3d', '1w', '1M']
  };

function updateTimeFrameOptions(typeTime){
    var timeFrame;

    switch (typeTime) {
        case '1':
            timeFrame = timeFrames.longTerm;
            break;
        case '2':
            timeFrame = timeFrames.shortTerm;
            break;
        case '3':
            timeFrame = timeFrames.scalping;
            break;
        default:
            console.log('Не удалось обновить время, входящий параметр: ', typeTime);
            return;
    }
    updateTimeFrame(timeFrame);
```

Продолжение приложения А

Окончание листинга А.10

```
}

function updateTimeFrame(timeframe){
    var selectTransactionTime = document.getElementById("TransactionTime");

    while (selectTransactionTime.options.length > 0) {
        selectTransactionTime.remove(0);
    }

    timeframe.forEach(element => {
        var newOptions = new Option(element,element);
        selectTransactionTime.append(newOptions);
    });
}

function GetTimeRequest(time){
    var symbol = time[time.length-1];
    var value = parseInt(time.substr(0,time.length-1));
    var timeRequest=0;
    switch(symbol){
        case 'm':
            timeRequest = value*60;
            break;
        case 'h':
            timeRequest = value*3600;
            break;
        case 'd':
            timeRequest = value*24*3600;
            break;
        case 'w':
            timeRequest = value*7*24*3600;
            break;
        case 'M':
            timeRequest = value*30*24*3600;
            break;
        default:
            timeRequest = 60;
    }
    return timeRequest*1000;
}
```

Листинг А.11 – код JavaScript для реализации регистрации и входа пользователя из authController.js

```
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');
const User = require('../models/User');

const JWT_SECRET = 'my_super_secret_key'; // Желательно хранить в переменной окружения

const signup = async (req, res) => {
  try {
    const { email, password } = req.body;

    if (!email || !password)
      return res.status(400).json({ status: 'error', message: 'Заполните все поля.' });

    const existingUser = await User.findOne({ email });
```

Продолжение приложения А

Окончание листинга А.11

```
    if (existingUser)
      return res.status(400).json({ status: 'error', message: 'Email уже зарегистрирован.'
});

    const salt = await bcrypt.genSalt(10);
    const hashedPassword = await bcrypt.hash(password, salt);
    const user = await User.create({ email, password: hashedPassword });
    res.status(200).json({ status: 'success', message: 'Регистрация успешна.' });
  } catch (error) {
    console.error('Ошибка регистрации:', error);
    res.status(500).json({ status: 'error', message: 'Ошибка сервера.' });
  }
};

const login = async (req, res) => {
  try {
    const { email, password } = req.body;

    const user = await User.findOne({ email });
    if (!user)
      return res.status(400).json({ status: 'error', message: 'Неверный email или пароль.'
});

    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch)
      return res.status(401).json({ status: 'error', message: 'Неверный email или пароль.'
});

    const token = jwt.sign({ id: user._id }, JWT_SECRET, { expiresIn: '1h' });
    res.status(200).json({ status: 'success', token });
  } catch (error) {
    console.error('Ошибка входа:', error);
    res.status(500).json({ status: 'error', message: 'Ошибка сервера.' });
  }
};

module.exports = { signup, login };
```

Листинг А.12 – код JavaScript для получения цены криптовалюты и преобразование в формат json из cryptoController.js

```
const binance = require("../services/apiBinance");
const macd = require("../services/macd");
const rsi = require("../services/rsi");
const sma = require("../services/sma");
const wma = require("../services/wma");

var symbolCrypto="";

exports.analyze = async function (request, response){
    try{
        const cryptoSymbol = request.body.crypto;
        const timeFrame = request.body.time;
        console.log(`Получены данные для анализа: ${cryptoSymbol} ${timeFrame}`);

        const symbol = await binance.GetAllSymbols(cryptoSymbol);
        const interval = timeFrame || '5m';
```

Продолжение приложения А

Окончание листинга А.12

```
        const limit = 200;// Получаем последние периоды

        const res = await binance.GetHistory(symbol,interval,limit);

        // kline[4] - это цена закрытия
        const closingPrice = res.data.map(kline => parseFloat(kline[4]));
        const macd_value = await macd.main(closingPrice,symbol);
        const rsi_value = await rsi.main(closingPrice,symbol);
        const sma_value = await sma.main(closingPrice,symbol);
        const wma_value = await wma.main(closingPrice,symbol);

        const result = {macd: macd_value,
                    wma: wma_value,
                    rsi:rsi_value,
                    sma:sma_value}

        response.send(JSON.stringify(result));
    }catch(error){
        response.status(404).send(error);
    }
};

// обработка данных для графика вынес на сервер, чтобы избавиться от ошибки CORS
exports.klines = async function (request, response){
    var cryptoSymbol = request.query.crypto;
    var timeFrame = request.query.time;
    try{
        const symbol = await binance.GetAllSymbols(cryptoSymbol);
        const interval = timeFrame || '5m';
        const limit = 100;// Получаем последние периоды

        var res;

       res = await binance.GetHistory(symbol,interval,limit);
    }catch(error){
         console.error(`Ошибка при получении данных, входящие параметры: ${cryptoSymbol}
${timeFrame}`);
        response.status(404).send(error);
        return;
    }

    // Обработка данных
    dataPoints = [];
    labels = [];
    res.data.forEach(item => {
        const timestamp = new Date(item[0]).toLocaleString(); // Преобразование времени
        const closePrice = parseFloat(item[4]); // Закрывающая цена
        labels.push(timestamp);
        dataPoints.push(closePrice);
    });
    response.send(JSON.stringify({dataPoints,labels}));
};
```

Листинг А.13 – код JavaScript для реализации middleware и проверки JWT-токена

```
const jwt = require('jsonwebtoken');
const JWT_SECRET_KEY = 'my_super_secret_key';
const verifyToken = (req, res, next) => {
  const authHeader = req.headers['authorization'];
  if (!authHeader) return res.status(401).json({ error: 'Token missing' });
  const token = authHeader.split(' ')[1];
  if (!token) return res.status(401).json({ error: 'Token format invalid' });

  try {
    const decoded = jwt.verify(token, JWT_SECRET_KEY);
    req.user = decoded;
    next();
  } catch (err) {
    res.status(403).json({ error: 'Invalid or expired token' });
  }
};
module.exports = verifyToken;
```

Листинг А.14 – код JavaScript для создания модели пользователя для хранения в базе данных MongoDB из verfyTocken.js

```
const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({
  email: {
    type: String,
    required: true,
    unique: true
  },
  password: {
    type: String,
    required: true
  }
});

const User = mongoose.model('User', userSchema);
module.exports = User;
```

Листинг А.15 – код JavaScript создание роутера Express для обработки запросов к API из apiRouter.js

```
const express = require("express");
const apiRouter = express.Router();
const cryptoRouter = require("../routes/cryptoRouter.js");

// Обработчик http://localhost:3000/api
// Обращение к http://localhost:3000/api или http://localhost:3000/api/
apiRouter.get("/", (req, res) => {
    res.send('Добро пожаловать в API для анализа криптовалют!');
});
// В cryptoRouter будут поступать все маршруты http://localhost:3000/api/crypto
apiRouter.use("/crypto", cryptoRouter);
module.exports = apiRouter;
```

Листинг А.16 – код JavaScript для создания маршрута Express для регистрации пользователей с безопасным хэшированием пароля из auth.js

```
const express = require('express');
const router = express.Router();
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');

const User = require('../models/User');

const JWT_SECRET_KEY = 'my_super_secret_key';

// Регистрация
router.post("/signup", async (req, res) => {
  const salt = await bcrypt.genSalt(10);
  const securePass = await bcrypt.hash(req.body.password, salt);

  try {
    const user = await User.create({
      email: req.body.email,
      password: securePass
    });
    res.status(201).json({ message: "User registered", userId: user._id });
  } catch (err) {
    res.status(400).json({ error: "Registration failed" });
  }
});
// Вход
router.post("/login", async (req, res) => {
  try {
    const user = await User.findOne({ email: req.body.email });
    if (!user) return res.status(400).json({ error: "Invalid credentials" });

    const isMatch = await bcrypt.compare(req.body.password, user.password);
    if (!isMatch) return res.status(401).json({ error: "Invalid credentials" });

    const payload = { userId: user._id };
    const token = jwt.sign(payload, JWT_SECRET_KEY, { expiresIn: "1h" });
    res.json({ token });
  } catch (err) {
    res.status(500).json({ error: "Server error" });
  }
});
module.exports = router;
const express = require("express");
const cryptoRouter = express.Router();
const cryptoController= require("../controllers/cryptoController.js");
// Обработчик http://localhost:3000/api/crypto
// Обращение к http://localhost:3000/api/crypto или http://localhost:3000/api/crypto/
cryptoRouter.get("/", (req, res) => {
    res.send(JSON.stringify('Добро пожаловать в API для анализа криптовалют!'));
});
//http://localhost:3000/api/crypto/analyze
cryptoRouter.post("/analyze", cryptoController.analyze);
//http://localhost:3000/api/crypto/klines
cryptoRouter.get("/klines",cryptoController.klines);
module.exports = cryptoRouter;
```

Листинг А.17 – код JavaScript для получения названия криптовалюты и исторических данных из apiBinance.js

```
const axios = require('axios');

// Функция для получения списка всех доступных символов
exports.GetAllSymbols = async (cryptoSymbol) => {
    try {
        const response = await axios.get('https://api.binance.com/api/v3/exchangeInfo');
        const symbols = response.data.symbols;
        const cryptoSymbols = {};

        symbols.forEach(symbol => {
            if (symbol.quoteAsset === 'USDT') {
                const baseAsset = symbol.baseAsset.toLowerCase();
                cryptoSymbols[baseAsset] = symbol.symbol;
            }
        });
        cryptoSymbol = cryptoSymbol.toLowerCase();
        if(cryptoSymbol.endsWith('usdt')){
            cryptoSymbol = cryptoSymbol.substr(0,cryptoSymbol.length-4);
        }
        const cryptoName = cryptoSymbol; // Приводим к нижнему регистру для сопоставления
        // Получаем символ на основе полного названия криптовалюты
        const symbol = cryptoSymbols[cryptoName]; // Если не найдено, по умолчанию BTCUSDT
        if(!symbol){
            throw new Error("Такой криптовалюты нет!");
        }
        return symbol;
    } catch (error) {
        throw new Error(error);
    }
};

exports.GetHistory = async(symbol,interval,limit)=>{
    try{
        const response = await axios.get('https://api.binance.com/api/v3/klines', {
            params: {
                symbol: symbol,
                interval: interval,
                limit: limit
            }
        });
        return response;

    }catch(error){
        throw new Error('Ошибка при получении исторических данных: '+error.response);
    }
};
```

Листинг А.18 – код JavaScript для расчета сигнала по индикатору MACD из macd.js

```
const { MACD } = require('technicalindicators');

// Переменные для хранения предыдущих значений MACD и сигнала
let previousMACD = null;
let previousSignal = null;
```

Продолжение приложения А

Продолжение листинга А.18

```
let previousPrice = null;
let symbol = null;

const fastPeriod = 12;
const slowPeriod = 26;
const signalPeriod = 9;

// Функция для получения исторических данных и расчета MACD
const getHistoricalData = async (prices) => {
    const macdInput = {
        values: prices,
        fastPeriod: fastPeriod,
        slowPeriod: slowPeriod,
        signalPeriod: signalPeriod,
        SimpleMAOscillator: false,
        SimpleMASignal: false
    };

    const macdValues = MACD.calculate(macdInput);

    // Получаем последнее значение MACD и добавляем его в массив
    if (macdValues.length > 0) {
        const latestValue = macdValues[macdValues.length - 1];
        var indicator = "Нет сигнала";
        // Проверка на пересечение
        if (previousMACD !== null && previousSignal !== null) {
            if (latestValue.MACD > latestValue.signal && previousMACD <= previousSignal) {
                indicator = "BUY";
                } else if (latestValue.MACD < latestValue.signal && previousMACD >=
previousSignal) {
                indicator = "SOLD";
            }
        }

        // Проверка на дивергенцию/конвергенцию
        var isEvent="Ничего";
        if (previousPrice !== null) {
                if ((latestValue.MACD > previousMACD && prices[prices.length - 1] <
previousPrice) ||
                    (latestValue.MACD < previousMACD && prices[prices.length - 1] >
previousPrice)) {
                    isEvent = "Дивергенция";
             } else if ((latestValue.MACD > previousMACD && prices[prices.length - 1] >
previousPrice) ||
                        (latestValue.MACD < previousMACD && prices[prices.length - 1] <
previousPrice)) {
                    isEvent = "Конвергенция";
            }
        }

        // Обновляем предыдущие значения
        previousMACD = latestValue.MACD;
        previousSignal = latestValue.signal;
        previousPrice = prices[prices.length - 1];

        return {args:{
                    fastPeriod,
                    slowPeriod,
```

Окончание листинга А.18

```
                    signalPeriod

                    },
                data:{
                    isEvent,
                    macd: latestValue.MACD,
                    signal: latestValue.signal,
                    histogram:latestValue.histogram
                    },
                indicator
                }
    }

    throw new Error("Недостаточно данных для расчета текущего MACD.");
};

const defaultInitValue = ()=>{
    previousMACD = null;
    previousSignal = null;
    previousPrice = null;
};

// Основная функция
exports.main = async (prices,cryptoSymbol) => {
    try{
        if(cryptoSymbol!=symbol){
            defaultInitValue();
        }
        // Первоначальный вызов функции для получения данных и расчета MACD
        var data = await getHistoricalData(prices);

        console.log(`MACD: успешно`);

        return data;
    }catch(error){
        console.log("ERROR MACD:\n"+error);
        return{error}
    }
};
```

Листинг А.19 – код JavaScript для расчета сигнала по индикатору RSI из rsi.js

```
const { RSI, SMA } = require('technicalindicators');

// Массивы для хранения значений RSI и SMA
let rsiValuesArray = [];
let smaValuesArray = [];

const period = 14;
// Переменная для хранения предыдущего сигнала
let previousSignal = null;

let symbol = null;

// Функция для получения исторических данных и расчета RSI и SMA
const getHistoricalData = async (prices) => {
    // Рассчитываем RSI по ценам
    const rsiInput = {
```

Продолжение приложения А

Продолжение листинга А.19

```
        values: prices,
        period: period
    };

    const rsiValuesFull = RSI.calculate(rsiInput);

    // Нужно минимум 15 значений, чтобы взять iRSI и iRSI-1
    if (rsiValuesFull.length < 15){
        throw new Error("Недостаточно данных для расчета RSI.");
    }

    // Обновляем массив последних 15 значений RSI (для удобства)
    rsiValuesArray = rsiValuesFull.slice(-15);

      // Рассчитываем SMA по последним 14 значениям RSI (текущие последние 14 из
rsiValuesArray)
    const smaInputCurrent = {
        values: rsiValuesArray.slice(1), // берем последние 14 значений, т.к. всего 15 в
массиве
        period: period
    };

    smaValuesArray = SMA.calculate(smaInputCurrent); // будет один элемент

    if (!smaValuesArray || smaValuesArray.length === 0) {
        throw new Error("Недостаточно данных для расчета SMA.");
    }

    // Для предыдущего значения SMA пересчитаем на сдвинутом наборе данных:

    const smaInputPrev = {
        values: rsiValuesArray.slice(0, -1), // первые 14 элементов из последних 15
        period: period
    };

    let prevSmaArr = SMA.calculate(smaInputPrev);

    if (!prevSmaArr || prevSmaArr.length === 0) {
        throw new Error("Недостаточно данных для расчета SMA.");
    }

    // Берем текущие значения:
    const iRSI = rsiValuesArray[rsiValuesArray.length -1];      // текущий RSI
    const iRSI_1 = rsiValuesArray[rsiValuesArray.length -2];    // предыдущий RSI

    const iSMA = smaValuesArray[smaValuesArray.length -1];      // текущий SMA
    const iSMA_1 = prevSmaArr[prevSmaArr.length -1];           // предыдущий SMA

    return checkConditions(iRSI, iRSI_1, iSMA, iSMA_1);
};

// Функция проверки условий согласно заданию с выводом массивов значений RSI и SMA,
// а также выводом сообщений только при смене сигнала
const checkConditions = (iRSI, iRSI_1, iSMA, iSMA_1) => {
    let signalMessage = "Нет сигнала"; // локальная переменная для текущего сигнала

    //console.log("Массив последних значений RSI:");
    //console.log(rsiValuesArray.map(v => v.toFixed(2)));
```

Продолжение листинга А.19

```
    //console.log("Массив последних значений SMA:");
    //console.log(smaValuesArray.map(v => v.toFixed(2)));

    //console.log(`Текущий RSI: ${iRSI.toFixed(2)}, Предыдущий RSI: ${iRSI_1.toFixed(2)}`);
    //console.log(`Текущий SMA: ${iSMA.toFixed(2)}, Предыдущий SMA: ${iSMA_1.toFixed(2)}`);
    if(iRSI >= 70 && iRSI_1 >= 70)
    {
        signalMessage = "SOLD";
    }
    if (iRSI >= 70 && iRSI_1 < 70)
    {
        signalMessage = "SOLD";
    }else if (iRSI <= 30 && iRSI_1 <= 30)
    {
        signalMessage = "BUY";
    }else if (iRSI <= 30 && iRSI_1 > 30)
    {
        signalMessage = "BUY";
    }else if (
        iRSI > 32 && iRSI < 68 &&
        iRSI_1 > 32 && iRSI_1 < 68)
    {
        if (iRSI > iSMA && iRSI_1 > iSMA_1) {
            signalMessage = "BUY";
        } else if (iRSI < iSMA && iRSI_1 < iSMA_1) {
            signalMessage = "SOLD";
        }
    }

    // Выводим сообщение только если оно изменилось относительно предыдущего сигнала
    if (signalMessage !== null && signalMessage !== previousSignal) {
        //console.log(signalMessage);
        previousSignal = signalMessage;
    }

    return {args:{
                period: period
                },
            data:{
                rsi: iRSI
                },
            indicator: signalMessage
            }
};

const defaultInitValue = ()=>{
    rsiValuesArray = [];
    smaValuesArray = [];
    previousSignal = null;
};

// Основная функция
exports.main = async (prices,cryptoSymbol) => {
    try{
        if(cryptoSymbol!=symbol){
            defaultInitValue();
        }
        // Первоначальный вызов функции для получения данных и расчета MACD
```

Продолжение приложения А

Окончание листинга А. 19

```
        var data = await getHistoricalData(prices);

        console.log(`RSI: успешно`);

        return data;
    }catch(error){
        console.log("ERROR RSI:\n"+error);
        return{error}
    }
};
```

Листинг А.20 – код JavaScript для расчета сигнала по индикатору SMA из sma.js

```
const { SMA } = require('technicalindicators');

// Массив для хранения значений SMA
let supportLevel = null;
let resistanceLevel = null;
let symbol = null;

const period1=50;
const period2=100;
const period3=200;

// Функция для получения исторических данных и расчета SMA
const getHistoricalData = async (prices) => {

    // Рассчитываем SMA для периодов 50, 100 и 200
    const sma50 = SMA.calculate({ values: prices, period: period1 });
    const sma100 = SMA.calculate({ values: prices, period: period2 });
    const sma200 = SMA.calculate({ values: prices, period: period3 });

    // Получаем последние значения SMA
    if (sma50.length > 0 && sma100.length > 0 && sma200.length > 0) {
        const latestSMA50 = sma50[sma50.length - 1];
        const latestSMA100 = sma100[sma100.length - 1];
        const latestSMA200 = sma200[sma200.length - 1];

        //console.log(`SMA 50 - ${latestSMA50}`);
        //console.log(`SMA 100 - ${latestSMA100}`);
        //console.log(`SMA 200 - ${latestSMA200}`);
        // Получаем текущую цену
        const currentPrice = prices[prices.length - 1];
        //console.log(`Текущая цена - ${currentPrice}`);
        // Определяем уровень поддержки и сопротивления
        let newSupportLevel = null;
        let newResistanceLevel = null;

        // Находим уровень поддержки (максимальное значение SMA ниже текущей цены)
        [latestSMA50, latestSMA100, latestSMA200].forEach(sma => {
            if (sma < currentPrice) {
                if (newSupportLevel === null || sma > newSupportLevel) {
                    newSupportLevel = sma;
                }
            } else if (sma > currentPrice) {
```

Продолжение приложения А

Продолжение листинга А.20

```
                if (newResistanceLevel === null || sma < newResistanceLevel) {
                    newResistanceLevel = sma;
                }
            }
        });

        if (newSupportLevel !== supportLevel || newResistanceLevel !== resistanceLevel) {
            //console.log(`Новый уровень поддержки / сопротивления:`);
                     //console.log(`Поддержка   -   ${newSupportLevel},   Сопротивление   -
${newResistanceLevel}`);
            supportLevel = newSupportLevel;
            resistanceLevel = newResistanceLevel;
        }

        // Проверка на золотой крест и крест смерти между SMA
        let message = '';
        let recomendazia = "Нет сигнала";
        // Золотой крест и крест смерти между SMA 50 и SMA 100
        if (latestSMA50 > latestSMA100 && latestSMA100 > latestSMA200) {
            message += `Золотой крест между SMA 50 и SMA 100.\n`;
            recomendazia='BUY';
        } else if (latestSMA50 < latestSMA100 && latestSMA100 < latestSMA200) {
            message += `Крест смерти между SMA 50 и SMA 100.\n`;
            recomendazia='SOLD';
        }

        // Золотой крест и крест смерти между SMA 50 и SMA 200
        if (latestSMA50 > latestSMA200) {
            message += `Золотой крест между SMA 50 и SMA 200.\n`;
            recomendazia='BUY';
        } else if (latestSMA50 < latestSMA200) {
            message += `Крест смерти между SMA 50 и SMA 200.\n`;
            recomendazia='SOLD';
        }

        // Золотой крест и крест смерти между SMA 100 и SMA 200
        if (latestSMA100 > latestSMA200) {
            message += `Золотой крест между SMA 100 и SMA 200.\n`;
            recomendazia='BUY';
        } else if (latestSMA100 < latestSMA200) {
            message += `Крест смерти между SMA 100 и SMA 200.\n`;
            recomendazia='SOLD';
        }

        if (message) {
           // console.log(message);
        }

        return {args:{
                    period1,
                    period2,
                    period3
                    },
                data:{
                    newSupportLevel,
                    newResistanceLevel,
                    message,
                    sma50:latestSMA50,
                    sma100:latestSMA100,
```

Продолжение приложения А

Продолжение листинга А.20

```
                    sma200:latestSMA200
                    },
                indicator:recomendazia
                }
    }
    throw new Error("Недостаточно данных для расчета текущего SMA.");
};

const defaultInitValue = ()=>{
    supportLevel = null;
    resistanceLevel = null;
};

// Основная функция
exports.main = async (prices,cryptoSymbol) => {
    try{
        if(cryptoSymbol!=symbol){
            defaultInitValue();
        }
        // Первоначальный вызов функции для получения данных и расчета MACD
        var data = await getHistoricalData(prices);

        console.log(`SMA: успешно`);

        return data;
    }catch(error){
        console.log("ERROR SMA:\n"+error);
        return{error}
    }
};
```

Листинг А.21 – код JavaScript для подключения к базе данных MongoDB

```
const mongoose = require('mongoose');

const connectMongo = async () => {
  try {
    await mongoose.connect('mongodb://127.0.0.1:27017/crypto_db', {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    });
    console.log('MongoDB connected successfully');
  } catch (error) {
    console.error('MongoDB connection error:', error);
  }
};

module.exports = connectMongo;
```