TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 09.03.02 Информационные системы и технологии
Отделение школы (НОЦ) Отделение информационных технологий

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Торговый робот с возможностью интеграции в различные биржевые информационные системы

УДК 007.52:339:336.761

Студент

Группа	ФИО	Подпись	Дата
8И8А	Дорофеев Николай Сергеевич		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Цапко С.Г.	к.т.н., доцент		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН	Рыжакина Т.Г.	к.э.н, доцент		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Мезенцева И.Л.	-		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Цапко И.В.	к.т.н., доцент		

Томск – 2022г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ ООП

Код компетенции	Наименование компетенции
Универсальные компетенции	
УК(У)-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач
УК(У)-2	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений
УК(У)-3	Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде
УК(У)-4	Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(-ых) языке(-ах)
УК(У)-5	Способен воспринимать межкультурное разнообразие общества в социально-историческом, этическом и философском контекстах
УК(У)-6	Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни
УК(У)-7	Способен поддерживать должный уровень физической подготовленности для обеспечения полноценной социальной и профессиональной деятельности
УК(У)-8	Способен создавать и поддерживать в повседневной жизни и в профессиональной деятельности безопасные условия жизнедеятельности для сохранения природной среды, обеспечения устойчивого развития общества, в том числе при угрозе и возникновении чрезвычайных ситуаций и военных конфликтов
УК(У)-9	Способен проявлять предприимчивость в практической деятельности, в т.ч. в рамках разработки коммерчески перспективного продукта на основе научно-технической идеи
УК(У)-10	Способен принимать обоснованные экономические решения в различных областях жизнедеятельности
УК(У)-11	Способен формировать нетерпимое отношение к коррупционному поведению
Общепрофессиональные компетенции	
ОПК(У)-1	Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности
ОПК(У)-2	Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности
ОПК(У)-3	Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности

ОПК(У)-4	Способен участвовать в разработке технической документации, связанной с профессиональной деятельностью с использованием стандартов, норм и правил
ОПК(У)-5	Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем
ОПК(У)-6	Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий
ОПК(У)-7	Способен осуществлять выбор платформ и инструментальных программно-аппаратных средств для реализации информационных систем
ОПК(У)-8	Способен применять математические модели, методы и средства проектирования информационных и автоматизированных систем
Профессиональные компетенции	
ПК(У)-1	Способен выполнять интеграцию программных модулей и компонент
ПК(У)-2	Способен выполнять работы и управлять работами по созданию (модификации) и сопровождению информационных систем
ПК(У)-3	Способен создавать техническую документацию на продукцию в сфере информационных технологий, управлять технической информацией
ПК(У)-4	Способен выполнять работы по обеспечению функционирования баз данных и обеспечению их информационной безопасности
ПК(У)-5	Способен проводить, оценивать и следить за выполнением концептуального, функционального и логического проектирования систем малого и среднего масштаба и сложности

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки (специальность) 09.03.02 Информационные системы и технологии
Отделение школы (НОЦ) Отделение информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

бакалаврской работы

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	ФИО
8И8А	Дорофееву Николаю Сергеевичу

Тема работы:

Торговый робот с возможностью интеграции в различные биржевые информационные системы	
Утверждена приказом директора (дата, номер)	№34-61/с от 03.02.2022 г.

Срок сдачи студентом выполненной работы:	07.06.2022 г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Ошибки в торговых системах необходимо выявлять в первую очередь, так как это может привести к многим потерям как со стороны клиентов, так и со стороны владельцев систем. В результате анализа ошибок разных бирж потребность разработки специальных торговых роботов, симулирующих поведение реальных людей на рынке, с целью

	обнаружить уязвимости в торговой системе является особенно актуальной.
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	1. Анализ технологий и решений для тестирования торговых систем 2. Создание архитектуры и разработка робота 3. Апробация роботов 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение **5.** Социальная ответственность
Перечень графического материала *(с точным указанием обязательных чертежей)*	Презентация в формате *.pptx

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Рыжакина Татьяна Гавриловна
Социальная ответственность	Мезенцева Ирина Леонидовна

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	24.01.2022

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Цапко С.Г.	к.т.н., доцент		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8И8А	Дорофеев Николай Сергеевич		

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки (специальность) 09.03.02 Информационные системы и технологии
Уровень образования бакалавриат
Отделение школы (НОЦ) Отделение информационных технологий
Период выполнения весенний семестр 2021 /2022 учебного года

Форма представления работы:

бакалаврская работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	07.06.2022 г.

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
	Основная часть	75
	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	15
	Социальная ответственность	10

СОСТАВИЛ:
Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Цапко С.Г.	к.т.н., доцент		

СОГЛАСОВАНО:

Руководитель ООП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Цапко И.В.	к.т.н., доцент		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8И8А	Дорофеев Николай Сергеевич

Школа	Инженерная школа информационных технологий и робототехники	**Отделение школы (НОЦ)**	Отделение информационных технологий
Уровень образования	Бакалавриат	**Направление/специальность**	09.03.02 Информационные системы и технологии

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	Оклад руководителя (к.т.н., доцент) – 35000 руб. Оклад студента – 20000 руб.
2. *Нормы и нормативы расходования ресурсов*	Премиальный коэффициент 30%; Коэффициент доплат и надбавок: Научный руководитель - 40%; Студент - 20%; Районный коэффициент 30%; Коэффициент дополнительной заработной платы 15%; Накладные расходы 16%.
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	Коэффициент отчислений на уплату во внебюджетные фонды 30%
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения*	Оценка конкурентоспособности продукта Проведение SWOT анализа
2. *Планирование и формирование бюджета научных исследований*	Определение структуры работы. Расчет трудоемкости выполнения работ. Подсчет бюджета исследования
3. *Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования*	Оценка ресурсной, финансовой и экономической эффективности научно-исследовательского проекта.
Перечень графического материала *(с точным указанием обязательных чертежей)***:**	
1. *Оценка конкурентоспособности технических решений* 2. *Матрица SWOT* 3. *Альтернативы проведения НИ* 4. *График проведения и бюджет НИ* 5. *Оценка ресурсной, финансовой и экономической эффективности НИ*	

Дата выдачи задания для раздела по линейному графику	03.02.2022

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН	Рыжакина Татьяна Гавриловна	к.э.н.		03.02.2022

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8И8А	Дорофеев Николай Сергеевич		03.02.2022

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа		ФИО	
8И8А		Дорофеев Николай Сергеевич	
Школа	**Инженерная Школа Информационных технологий и Робототехники**	**Отделение (НОЦ)**	**Отделение Информационных Технологий**
Уровень образования	Бакалавриат	**Направление/ специальность**	***09.03.02 Информационные Системы и Технологии***

Тема ВКР:

Торговый робот с возможностью интеграции в различные биржевые информационные системы	
Исходные данные к разделу «Социальная ответственность»:	
Введение – Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика) и области его применения. – Описание рабочей зоны (рабочего места) при разработке проектного решения/при эксплуатации	*Объект исследования:* программное обеспечение (ПО) для автоматизации торговых операций в биржевой информационной системе. *Область применения:* биржевая информационная система. *Рабочая зона:* офис *Размеры помещения:* 5*6 м^2 *Количество и наименование оборудования рабочей зоны:* компьютер с доступом к целевой информационной системе (ИС). *Рабочие процессы, связанные с объектом исследования, осуществляющиеся в рабочей зоне:* настройка параметров ПО, контроль параметров и исправности ПО дистанционно.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности при эксплуатации: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	«Трудовой кодекс Российской Федерации» от 30.12.2001 N 197-ФЗ (ред. от 25.02.2022). ГОСТ 21889-76 Система «Человек-машина». Кресло человека-оператора. ГОСТ 12.2.032-78 Рабочее место при выполнении работ сидя.
2. Производственная безопасность при эксплуатации: – Анализ выявленных вредных и опасных производственных факторов	**Вредные факторы:** 1. Умственное перенапряжение, в том числе вызванное информационной нагрузкой; 2. Отсутствие или недостаток необходимого естественного освещения; **Опасные факторы:** 1. Производственные факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает работающий. **Требуемые средства коллективной и индивидуальной защиты от выявленных**

	факторов: электрическая изоляция элементов энергоснабжения и внутреннего оснащения ЭВМ.
3. Экологическая безопасность при эксплуатации	**Воздействие на литосферу:** утилизация отходов электрооборудования **Воздействие на гидросферу:** не выявлено **Воздействие на атмосферу:** углеродный след от использования электроэнергии
4. Безопасность в чрезвычайных ситуациях при эксплуатации	**Возможные ЧС:** землетрясения, бури, пожары, обрушение зданий, аварии. **Наиболее типичная ЧС:** пожар.
Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель ООД	Мезенцева Ирина Леонидовна			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8И8А	Дорофеев Николай Сергеевич		

РЕФЕРАТ

Выпускная квалификационная работа содержит: 100 страниц, 19 рисунков, 30 таблицу, 17 источников, 0 приложений.

Ключевые слова: биржа, робот, автоматизация, тестирование, node.js.

Актуальность работы: Ошибки в торговых системах необходимо выявлять в первую очередь, так как это может привести к многим потерям как со стороны клиентов, так и со стороны владельцев систем. В результате анализа ошибок разных бирж потребность разработки специальных торговых роботов, симулирующих поведение реальных людей на рынке, с целью обнаружить уязвимости в торговой системе является особенно актуальной.

Объектом исследования является программное обеспечение для симуляции клиентской активности в торговой системе.

Цель работы: анализ и выбор алгоритмов и разработка на их основе программных компонентов для симуляции действий пользователей в торговой системе для приведения ее в необходимое для тестирования состояние.

В результате исследования был спроектирован и разработан робот для симуляции активности в торговых системах.

Среда разработки: WebStorm.

Степень внедрения: частичная.

Область применения: тестирование торговых систем.

Оглавление

Введение

Компания ООО «Центр разработки для обнаружения ошибок» сотрудничает со многими биржами и проводит тестирование их систем на корректность работы. В результате анализа ошибок разных бирж потребность разработки специальных торговых роботов, симулирующих поведение реальных людей на рынке, с целью обнаружить уязвимости в торговой системе является особенно актуальной. Программные компоненты, разработанные в процессе, планируется в будущем встроить в микросервисный фреймворк th2, основанный на технологиях Docker и Kubernetes.

Необходимо отметить, что тестирование торговых систем – это необходимое мероприятие для их владельцев. Ведь когда система работает с финансами, ошибки могут привести к огромным убыткам как со стороны владельцев системы, так и со стороны их клиентов.

К примеру, в мае 2012 года на фоне первичного размещения акций компании Facebook на бирже NASDAQ [17] произошел сбой систем. На ценные бумаги крупной IT корпорации возник огромный единовременный спрос. Более того, из-за стратегических ошибок компании, размещающей акции, многие инвесторы стали сразу продавать купленные акции, не увидев желаемых темпов роста. Всё это создало серьезную нагрузку на системы и привело к финансовым потерям со стороны многих клиентов и, как следствие, к судебным искам в сторону биржи.

Для предотвращения подобных случаев, необходимо при тестировании биржи иметь возможность перевести систему в определенное состояние. Так как состояние торговых систем в значительной мере зависит от поведения ее участников, для задания различных состояний требуется симулировать поведение большого количества пользователей.

Цель исследования: анализ и выбор алгоритмов и разработка на их основе программных компонентов для симуляции действий пользователей в

торговой системе для приведения ее в необходимое для тестирования состояние.

Разработанные программные компоненты включают в себя торговых роботов, симулирующих действия пользователей в различных сценариях, и панель управления для централизованного управления этими роботами. В рамках данной работы акцент сделан на серверное приложение, симулирующее клиентов биржи. Разработка панели управления, способной подсоединяться и управлять роботами централизовано, рассмотрена в выпускной квалификационной работе Надежды Хромовой.

Задачи исследования:

- изучение процесса, специфики и технологий тестирования торговых систем;
- анализ алгоритмов для симуляции пользовательской активности в торговой системе;
- разработка программного обеспечения, симулирующего клиентскую активность в торговой системе;
- апробация приложения в сценарии управления им при помощи программных интерфейсов.

Система состоит из неограниченного числа запущенных копий роботов, симулирующих торговую активность клиентов определенной биржи, и панели управления, которая объединяет информацию обо всех роботах и отдает команды для выполнения различных действий.

Роботы должны обладать следующими свойствами:

- подключение к различным торговым системам;
- наличие возможности встраивания собственных алгоритмов.
- реализация на основе технологий, полностью совместимых с th2 [11].

Причина в возможности подключения к различным торговым системам следующая: предприятие занимается тестированием различных торговых систем, и для каждой системы предусмотрен свой способ взаимодействия. Необходимо предусмотреть, чтобы изменения робота для тестирования новой системы проходило с минимальными затратами труда.

Необходимость встраивания собственных алгоритмов объясняется тем, что тестирования может проводиться по множеству различных сценариев. Количество этих сценариев потенциально не ограничено и должны быть возможность их расширения.

В ходе работы были рассмотрены технологии и публикации Exactpro [9] [12], ведущей компании, специализирующейся на тестировании торговых систем. Были рассмотрены различные виды тестирования и шаблоны поведения людей на бирже, которые могут привести к проблемам в системе.

Была разработана архитектура программного комплекса для тестирования торговых систем и архитектура торговых роботов, в частности. Были спроектированы шаблонные абстрактные классы для удобного изменения робота под собственные нужды. Реализованы программные интерфейсы для возможности управления роботом при помощи другого приложения.

Также был рассмотрен процесс от изменения функционала робота под собственные цели до развертывания роботов в производственной системе и использования его через предоставленные программные интерфейсы.

ГЛАВА 1. АНАЛИЗ ТЕХНОЛОГИЙ И РЕШЕНИЙ ДЛЯ ТЕСТИРОВАНИЯ ТОРГОВЫХ СИСТЕМ

В ходе выполнения работы был проведен сбор информации о различных технологиях и подходах к тестированию торговых систем. Информация о тестировании торговых систем относительно недоступна в силу того, что специалисты этого сектора работают с критически важными данными.

Лидирующие позиции в тестировании торговых систем занимает компания Exactpro. В основном информация была взята из публикаций данной компании и из знаний о ее разработках.

1.1. Виды тестирования торговых систем

Существует два основных вида тестирования торговых систем:

1. функциональное;
2. нефункциональное.

Функциональное тестирование (рисунок 1.1.1) призвано проверить насколько правильно реализована та или иная функция системы.

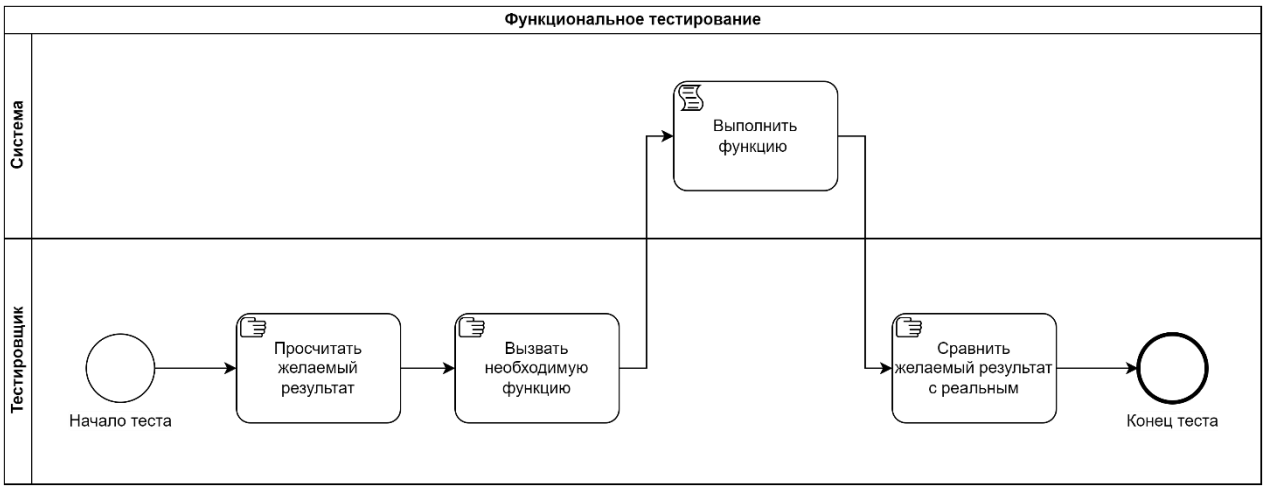

Рисунок 1.1.1. Процесс функционального тестирования в формате BPMN

Для этих целей производится тестовый вызов функции, при этом заранее просчитывается желаемый результат. В желаемый результат входит

как ответ системы на вызов функции, так и состояния данных, на которые должно производиться влияние.

Нефункциональное тестирование (рисунок 1.1.2) позволяет проверить, как влияет состояние системы на её работу.

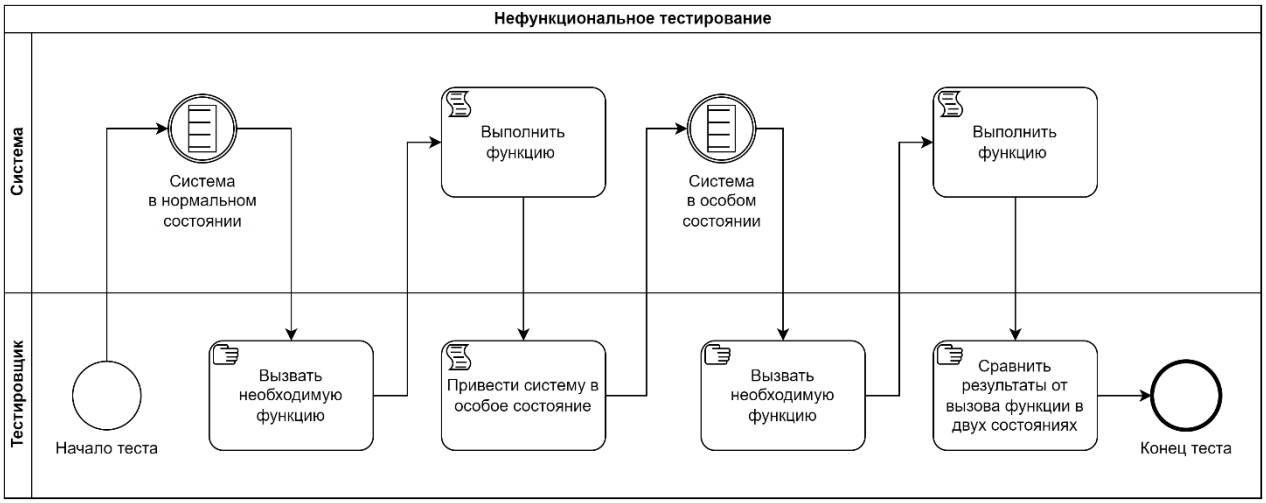

Рисунок 1.1.2. Процесс нефункционального тестирования в формате BPMN

Чтобы охарактеризовать состояние системы используются такие показатели, как нагрузка на систему или параллельный вызов функций, влияющих на одни и те же данные. Для нефункционального тестирования необходимо описать нормальное состояние системы. В процессе тестирования должны искусственно создаваться различные состояния системы и вызываться различные функции, результат которых должен сравниваться с результатом тех же функций в нормальном состоянии системы.

1.2. Имеющаяся инфраструктура для тестирования торговых систем

В настоящий момент на рынке тестирования торговых систем стандартом является фреймворк th2 [12] от компании Exactpro. Th2 является продуктом с открытым исходным кодом – исходный код всех его модулей можно найти на платформе GitHub. В будущем планируется интегрировать разрабатываемых в ходе данной работы роботов в данный фреймворк.

Основная идея th2 – это микросервисы [15]. Каждый кусочек логики тестирования является отдельным приложением. Реализованы микросервисы при помощи Kubernetes, популярного оркестратора контейнеров. Общую архитектуру данного фреймворка можно наблюдать на рисунке 1.2.1.

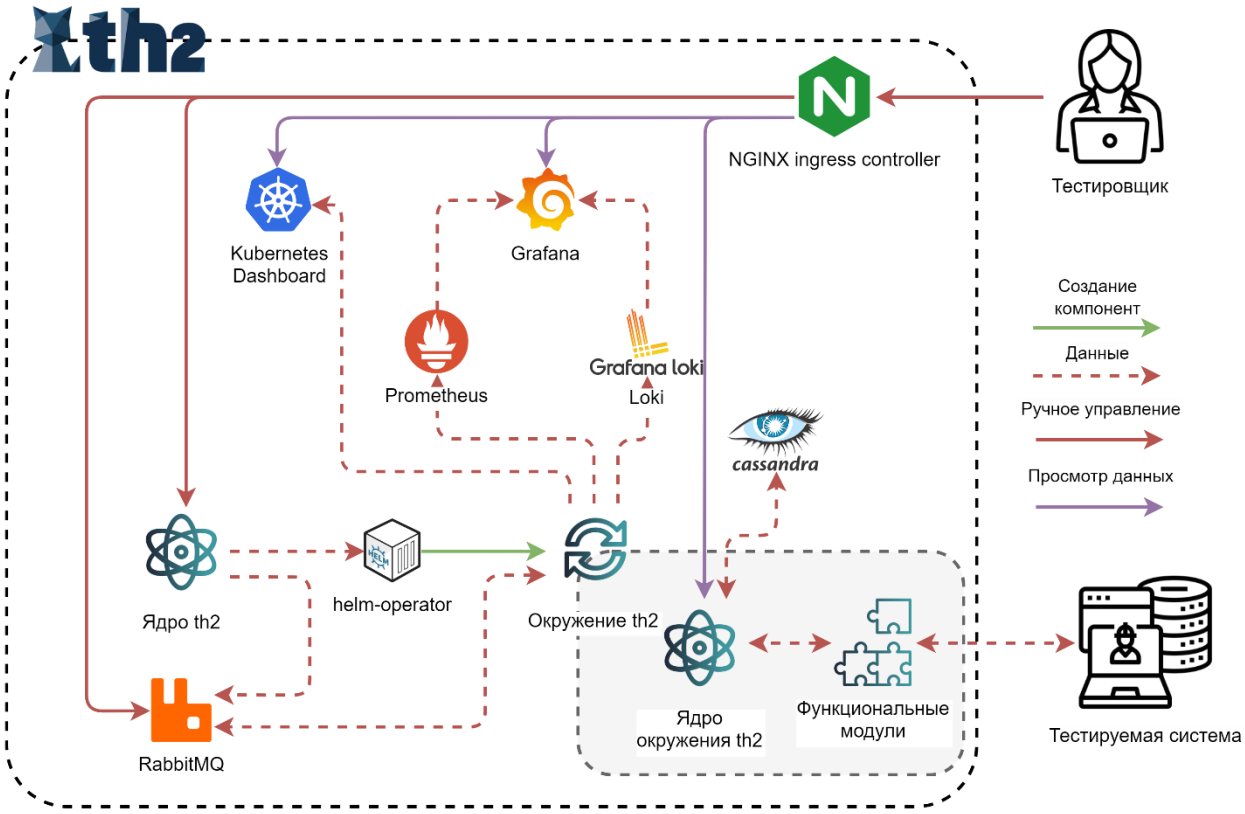

Рисунок 1.2.1. Архитектура микросервисного фреймворка th2

У компонентов данной системы следующие предназначения:

- ядро th2 – содержит в себе функционал для интерпретации и редактирования окружений th2, в том числе веб-интерфейс для отображения и редактирования окружений th2;
- helm-operator – сторонняя технология, предназначенная для автоматического создания объектов Kubernetes;
- окружение th2 – набор компонентов, связанных между собой и предназначенных для взаимодействия в системе;

- Kubernetes dashboard, Grafana, Prometheus, Loki – сторонние технология, предназначенные для сбора и отображения информации о различных компонентах системы в реальном времени;
- RabbitMQ – сторонняя технология, предназначенная для реализации очередей сообщений;
- ядро окружения th2 – набор необходимых для каждого окружения th2 компонентов, предназначенных для сохранения и просмотра событий и сообщений, созданных в системе;
- функциональные модули – компоненты окружения th2, выполняющие основные функции окружения;
- Cassandra – нереляционная база данных, выступающая физическим хранилищем событий и сообщений, созданных в системе;
- NGINX Ingress Controller – веб-сервер, предоставляющий доступ к веб-интерфейсам th2 извне Kubernetes кластера.

Весь функционал системы, организованной при помощи th2, составляется из множества различных модулей (приложений), соединенных между собой сервисом RabbitMQ и протоколом gRPC.

Th2 в базовой кмплектации предоставляет следующие возможности:

- отслеживание нагрузок различного вида в специальных веб-интерфейсах (Kubernetes Dashboard, Grafana Dashboard, RabbitMQ Dashboard);
- фиксация и хранение событий, возникших внутри системы;
- фиксация и хранение сообщений, возникших внутри системы;
- фильтрация и отображение событий и сообщений внутри специального веб-интерфейса, созданного командой th2.

В результате, можно отметить, что th2 предоставляет все возможности фиксации и обработки событий, необходимые для тестирования. Как следствие, в роботах не требуется подобного функционала и требуется реализовать лишь базовый функционал для хранения истории событий.

1.3. Сравнение торговых роботов для тестирования с торговыми роботами для трейдинга

Важно отметить, что торговые роботы для тестирования бирж и для трейдинга сильно отличаются (краткое сравнение приведено в таблице 1.3.1), хоть имеют схожие названия. Единственное, что их объединяет – это основной функционал в виде торговых алгоритмов, которые автоматически и в нужное время посылают определенные торговые поручения в систему.

Таблица 1.3.1 – Сравнение торговых роботов трейдинга и для тестирования

Параметр для сравнения	**Роботы для трейдинга**	**Роботы для тестирования**
Основной функционал	Алгоритмы для автоматической отправки торговых поручений	
Способ реализации	Использование клиента инструмента для трейдинга	Использование API торговой системы
Назначение алгоритмов	Заработок денег	Выявление уязвимостей в системе
Фактическое место исполнение алгоритмов	Торговая система	Робот
Полезное количество аккаунтов	Один	Множество

Для нужд трейдеров существует множество инструментов для облегчения их работы, такие как QUIK [8], MetaTrader. Недостатком таких инструментов является то, что для их поддержки в системе брокеру необходимо за свой счет настраивать и содержать серверную часть данных инструментов. Трейдерам же для использования нужно только скачать клиент инструмента (в большинстве случаев он поддерживается на множестве

платформ) и подключиться к серверу, предоставленному брокером. Биржи сами не предоставляют доступ к своим системам через данные инструменты. Соответственно, не во всех торговых системах реализованы данные инструменты.

В то же время биржи предоставляют прямой доступ к своим системам в частном порядке посредством API, подавляющее число брокеров использует API для автоматизации торговли в их системе. Таким образом, для тестирования непосредственно торговых систем бирж необходимо взаимодействовать с определенным API, что может быть проблематично в случае необходимости тестирования нескольких бирж.

Самым очевидным различием между роботами для тестирования и роботами для трейдинга будет назначение алгоритмов у данных типов роботов. Роботы для трейдинга призваны получить прибыль законным путем. Как правило, они анализируют множество факторов на рынке и составляют определенную математическую модель, на основе которой делают предсказания. В случае с роботами для тестирования необходимо проверить торговую систему на уязвимости. Причем в данном случае можно включить в понятие «система» не только техническую часть биржи, но и участников торгов, которые могут реагировать совершенно по-разному на ситуации на рынке. К примеру, в 2010 году произошел Flash Crash [16] – из-за ошибки промышленный индекс на ценную бумагу на рынке кратковременно сильно опустилась, однако многие участники запаниковали и начали продавать активы, что в итоге привело к сильному обвалу рынка.

Как уже говорилось ранее, роботы для трейдинга реализовываются при помощи специальных инструментов, сервера которых размещаются на стороне брокера. В эти инструменты так же обычно встроена поддержка создания алгоритмов при помощи специальных языков, и исполняются все эти алгоритмы на стороне сервера. Через клиентскую часть инструмента алгоритмы только фиксируются в системе и привязываются к определенному

аккаунту. Однако роботы для тестирования выполняют роль симулятора активности живого человека, и в данном случае необходимо производить расчеты для алгоритмов на стороне приложения-клиента системы.

Стоит отметить, что несколько аккаунтов, реализующих алгоритмы для трейдинга, заводить бессмысленно, так как алгоритмы можно без проблем организовать и на одном аккаунте. В то же время, чем больше параллельно роботов для тестирования будет работать, тем лучше. Ведь для создания нагрузки на систему, либо симуляции торговли большого количества людей необходимо как можно больше активных участников.

1.4. Торговые алгоритмы для тестирования системы

Далее будут рассмотрены торговые алгоритмы для тестовых роботов [9]. Условно алгоритмы можно разделить на 2 группы:

1. Алгоритмы, проверяющие техническую часть торговой системы (создающие нагрузку на систему):
    a. «нарезание»;
    b. «молоток».
2. Алгоритмы, влияющие на участников торговли / находящиеся под влиянием от участников торговли:
    a. «паникующий продавец/покупатель»;
    b. «расслоение»;
    c. «переполнение».

«Нарезание»

Алгоритм «Нарезание» (рисунок 1.4.1) способен отправить множество торговых поручений в систему за счет разделения одного крупного торгового поручения на несколько маленьких. В том числе при помощи этого алгоритма можно не привлекать внимания других участников торгов размещением крупного поручения.

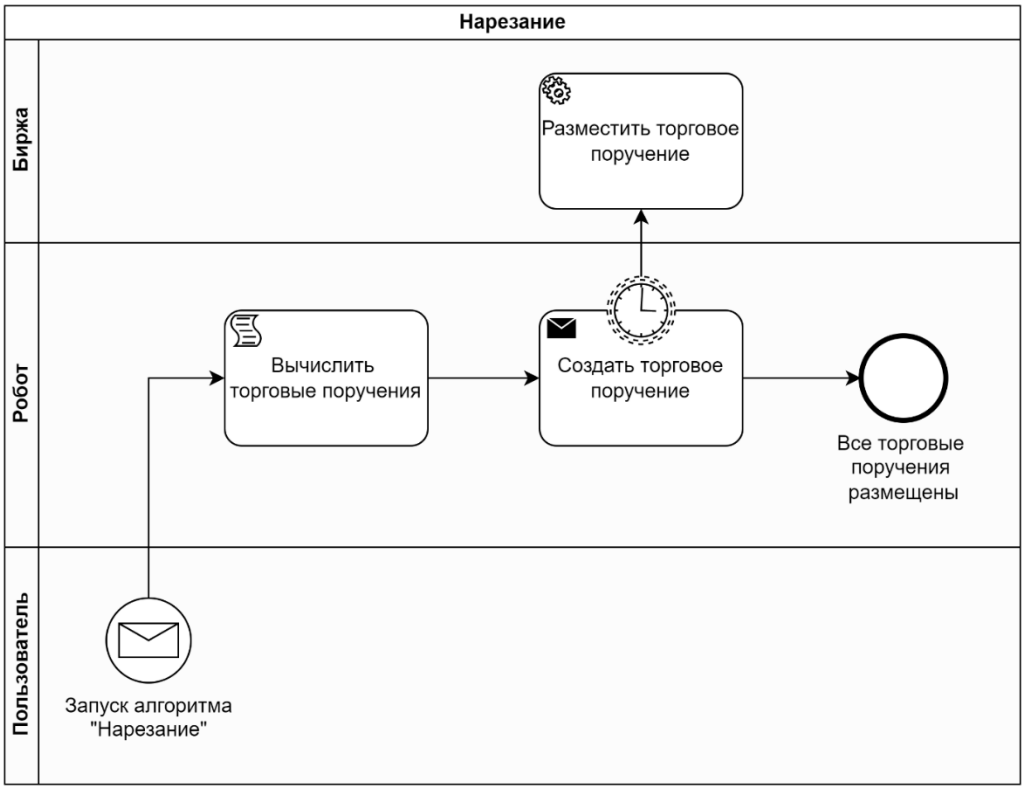

Рисунок 1.4.1. Процесс алгоритма «Нарезание» в формате BPMN

Алгоримт реализует механизм деления крупного торгового поручения на несколько маленьких торговых поручений с постепенной отправкой, чтобы не привлекать внимания остальных участников торгов.

«Молоток»

Алгоритм «Молоток» (рисунок 1.4.2) способен переполнить систему запросами на размещение торговых поручений в нерабочее время биржи. Размещение торгового поручения в самом начале дня торгов в том числе может дать некоторое преимущество.

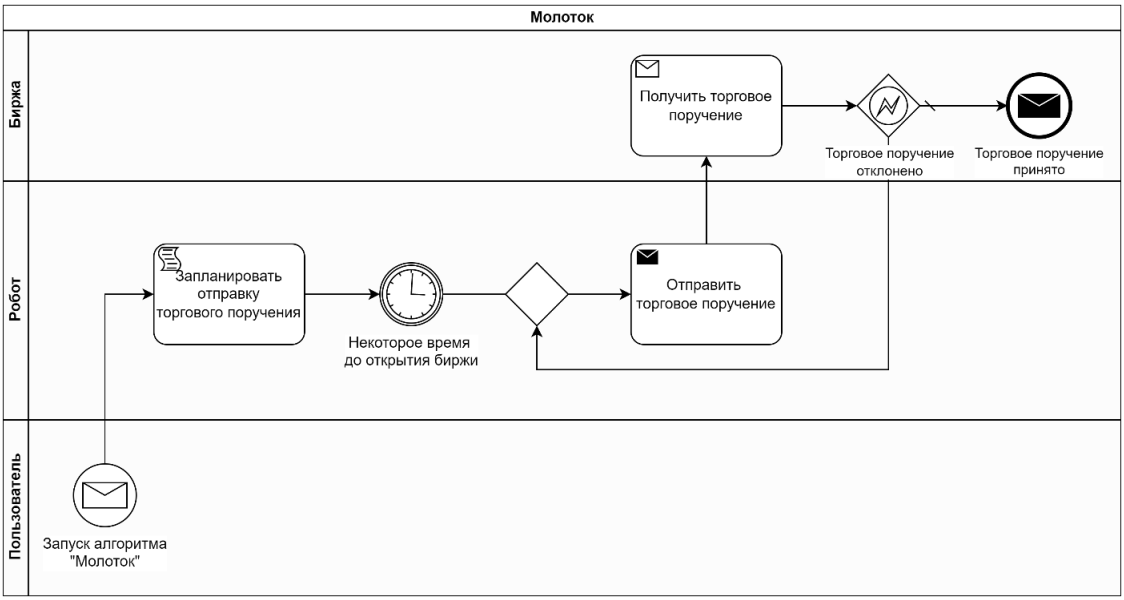

Рисунок 1.4.2. Процесс алгоритма «Молоток» в формате BPMN

Робот, действующий по алгоритму "Молоток", за некоторое время до открытия биржи начинает отсылать торговое поручение, которое поначалу отклоняется, так как биржа ещё не работает. Торговое поручение отправляется повторно после каждого отклонения, пока не будет удовлетворено.

«Паникующий продавец/покупатель»

При помощи алгоритма «Паникующий продавец/покупатель» (рисунок 1.4.3) робот симулирует неопытного трейдера, который принимает решения на эмоциях во время резких скачков котировок.

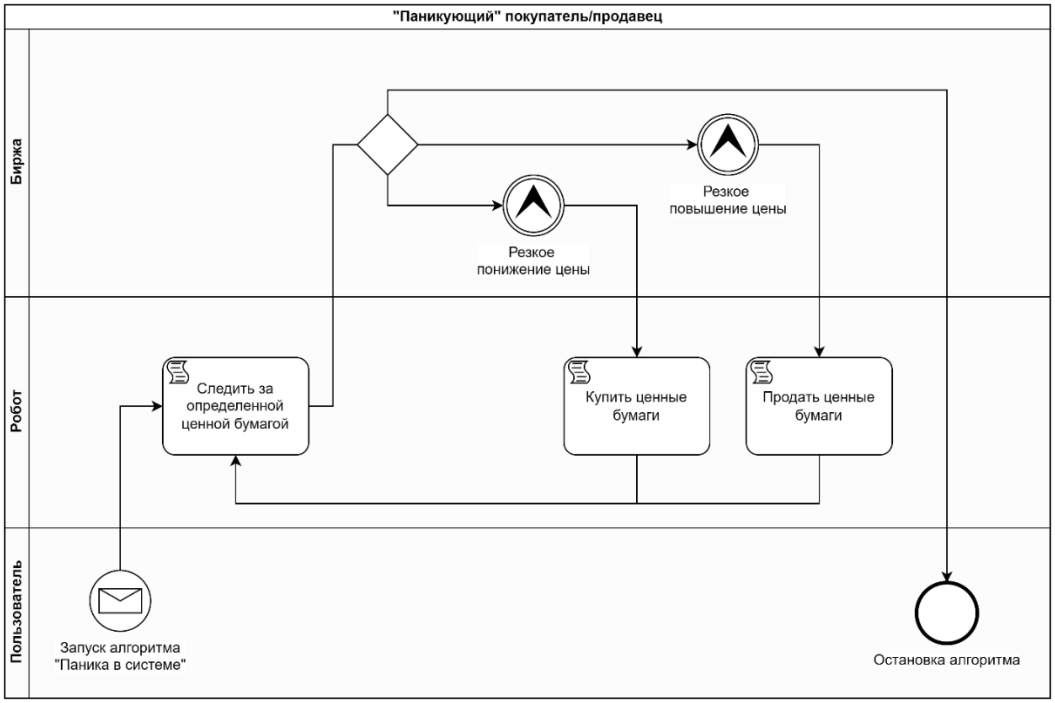

Рисунок 1.4.3. Процесс алгоритма «Паникующий продавец/покупатель» в формате BPMN

Алгоритм постоянно следит за курсом определенной ценной бумаги. При резком повышении цены робот покупает определенное количество ценных бумаг в надежде на то, что цена продолжит расти. При резком понижении цены – наоборот продает ценные бумаги, побоявшись понести большие убытки.

«Расслоение»

Алгоритм «Расслоение» (рисунок 1.4.4) создан для увеличения шанса взаимодействия других участников рынка с необходимыми торговыми поручениями за счет создания плохих условий в конкурирующих частях рынка.

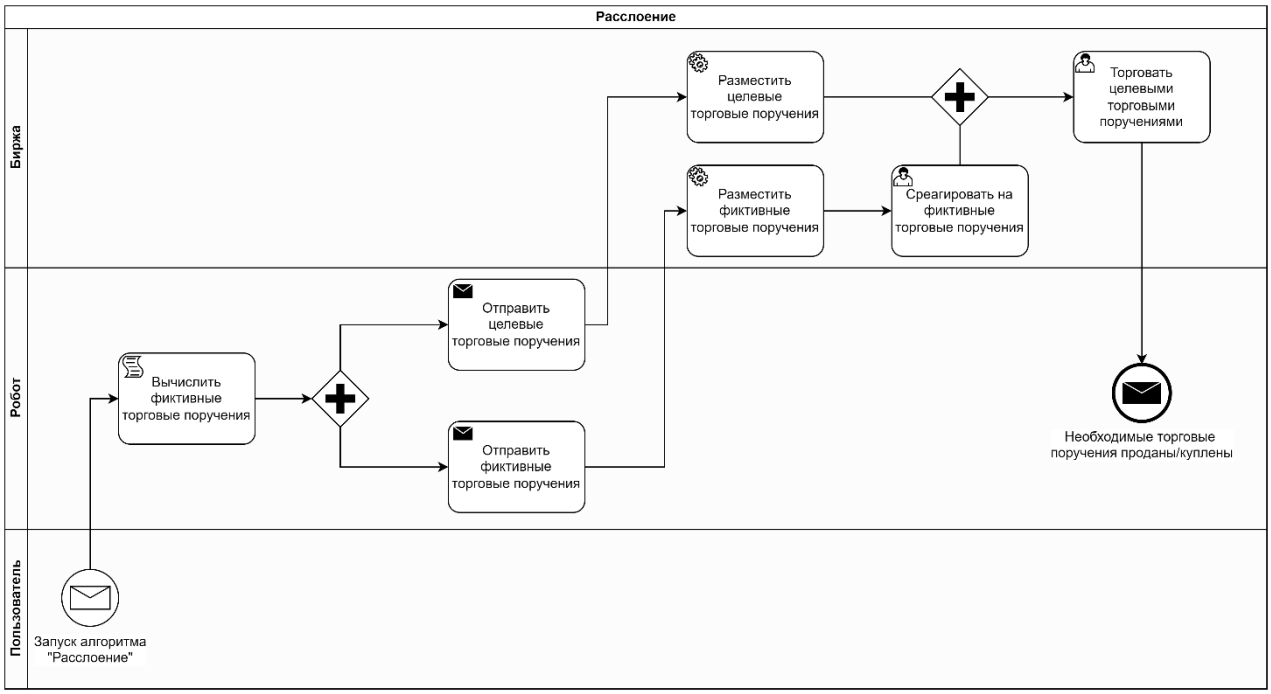

Рисунок 1.4.4. Процесс алгоритма «Расслоение» в формате BPMN

Робот, действующий по алгоритму "Расслоение", оставляет одновременно целевые торговые поручения в одной части рынка и фиктивные поручения - в другой, отпугивая пользователей от фиктивной части и провоцируя взаимодействовать с целевыми поручениями.

«Переполнение»

Алгоритм «Переполнение» (рисунок 1.4.5) призван захламить систему запросами для создания задержек. В теории, в тестируемой системе может с задержкой обновиться цена на ценную бумагу и с учетом того, что цена в других системах продолжает варьироваться, злоумышленники могут получить прибыль на разнице.

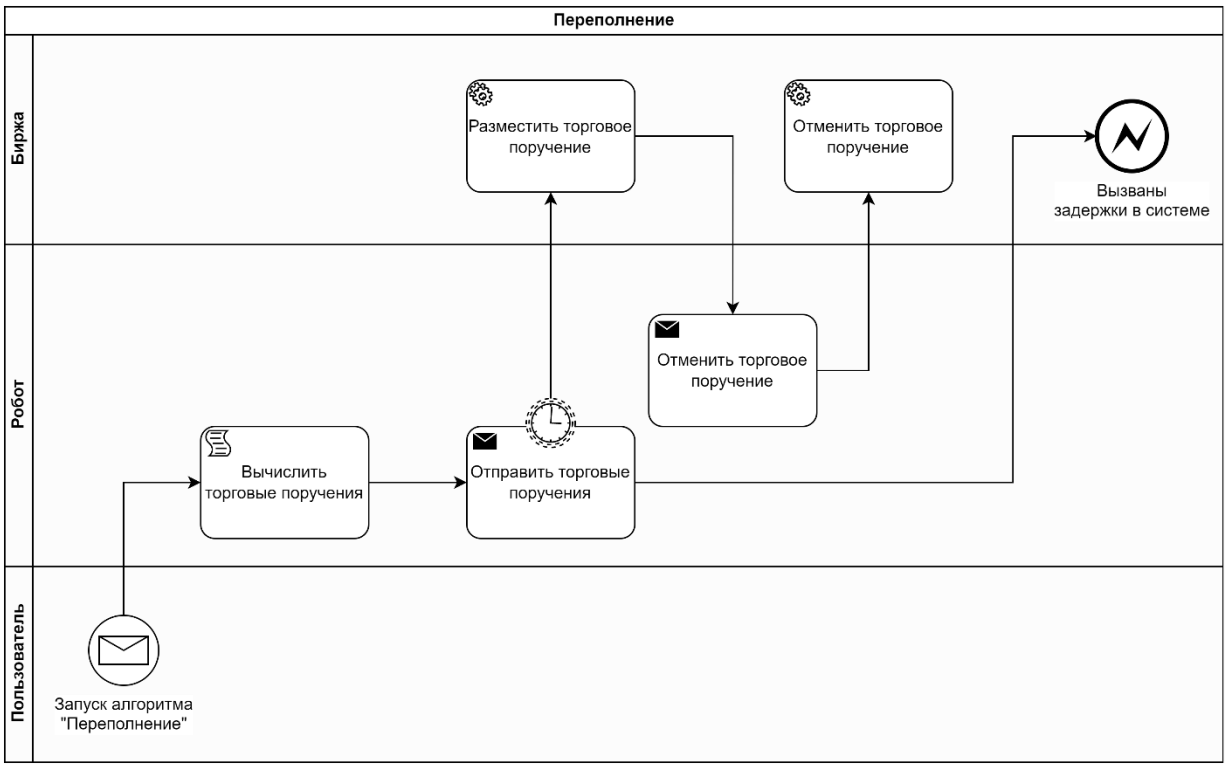

Рисунок 1.4.5. Процесс алгоритма «Переполнение» в формате BPMN

Робот постоянно отсылает и отменяет торговые поручения, вызывая задержки в системе.

ГЛАВА 2. СОЗДАНИЕ АРХИТЕКТУРЫ И РАЗРАБОТКА РОБОТА

Первым этапом разработки торговых роботов было создание архитектуры. Так как робот – это серверное приложение, которое использует реляционную базы данных, были описаны объекты робота и взаимоотношения между ними. В том числе была разработана модель данных для БД. Таким образом, при реализации функционала был шаблон, на который можно было ориентироваться.

2.1. Архитектура разрабатываемого решения

Разрабатываемая система (рисунок 2.1.1) состоит из следующих компонентов:

- торговая система (Подключается через API интерфейс);
- неограниченное число запущенных роботов;
- панель управления роботами.

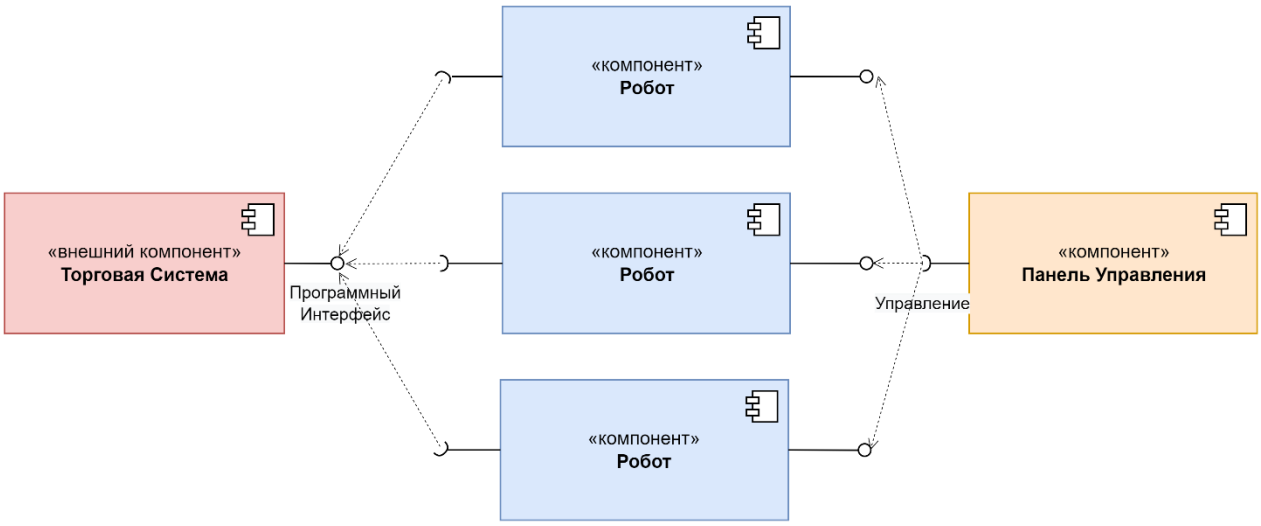

Рисунок 2.1.1. Архитектура системы роботов

Торговая Система – это система, реализованная брокером или биржей. Она должна иметь определенный программный интерфейс для подключения роботов.

Робот является серверным приложением, которое взаимодействует с торговой системой и предоставляет интерфейсы для управления. Он содержит в себе логику для подключения к торговой системе, логику преобразования структуры данных из формата торговой системы в локальный формат, который используется для хранения в базе данных и для обмена информацией с панелью управления. Для любой торговой системы достаточно одной совместимой версии приложения робота. Для запуска множества роботов запускается множество копий данного приложения, но с индивидуальными настройками доступа к аккаунту в торговой системе и настройками авторизации робота.

Панель управления является приложением для управления множеством торговых роботов. Данное приложение имеет возможность взаимодействовать с каждым роботом по отдельности и сразу с несколькими. Панель управления должна автоматически считывать алгоритмы из каждого робота и динамически создавать интерфейсы для каждого вызова.

Так как роботы и панель управления постоянно взаимодействуют между собой, для упрощения разработки было принято решение создавать их на основе одной и той же платформы и одного и того же языка программирования. Таким образом открывается возможность использовать общие библиотеки в обоих приложениях и поддерживать совместимость данных.

В процессе работы над ВКР был проведен анализ трех популярных платформ (таблица 2.1.1):

- Node.js, TypeScript [1][14];
- .NET Core, C# [2];
- Python.

Таблица 2.1.1 – Потенциальные платформы для реализации решения.

Платформа	**Возможности для создания робота**	**Возможности для создания панели управления**
Node.js, TypeScript	• Статическая типизация с возможностью отказаться от неё в некоторых местах • Много возможностей для манипуляций с типами • Широкий выбор библиотек • Средняя скорость работы программ	• Популярные SPA фреймворки: Vue, React, Angular • Создание кроссплатформенных приложений с интерфейсом: Electron
.NET Core, C#	• Статическая типизация • Широкий выбор библиотек • Высокая скорость работы программ	• SPA фреймворк: Blazor • Создание приложений с интерфейсом для Windows: Windows Presentation Foundation (WPF)
Python	• Динамическая типизация (в данном случае отрицательно повлияет на разработку) • Огромный выбор библиотек	• Создание кроссплатформенных приложений с интерфейсом: PyQT, PyGUI

	• Медленная скорость работы программ	

Платформа Node.js изначально была предназначена для создания серверных приложений при помощи языка программирования JavaScript. Данный язык является динамически типизированным и дает разработчикам слишком много свободы при манипуляциях с данными. Такая ситуация может плохо сказаться на разработке приложения, так как это может приводить к непредвиденным ошибкам уже во время работы. TypeScript – это надстройка над JavaScript, которая привносит статическую типизацию в разработку на Node.js. Однако TypeScript все равно сохраняет гибкость JavaScript за счет возможности отказаться от типизации в некоторых местах и широкого функционала для преобразования типов. Node.js в том числе довольно популярная платформа с обширным набором библиотек.

Платформа .NET Core, использующая язык C#, серьезно зарекомендовала себя. Статическая типизация позволяет избежать неожиданных ошибок после запуска программ, а программы работают быстрее чем большинство языков программирования высокого уровня. Стоит отметить большое количество библиотек для использования.

Python – это самый популярный язык программирования на данный момент, который имеет самую большую коллекцию библиотек для решения любой задачи. Однако отсутствие статической типизации делает затруднительным создание серьезных приложений. Также стоит отметить относительно медленную скорость работы программ, написанных на Python.

В итоге платформой для разработки приложений была выбрана Node.js с использованием языка TypeScript. Главной причиной стала гибкая работа с типами данных, что позволит разработать такую архитектуру программы, при

которой интеграция робота в новую торговую систему будет происходить наиболее быстро.

Также стоит отметить широкие возможности для создания веб-интерфейсов на основе SPA фреймворков, которые поддерживаются на всех устройствах, где есть браузер.

2.2. Выбор технологий для разработки

Необходимо было выбрать следующие технологии для разработки:

- База данных (БД) для кэша робота
- ORM для работы с БД для кэша
- Фреймворк для создания REST API
- Фреймворк для создания WebSocket

Кэш робота необходимо хранить в локальной и легковесной базе данных, чтобы не было необходимости настраивать отдельный элемент системы. На данный момент существует только одна технология, способная удовлетворить этим требованиям – SQLite [11]. Все данные хранятся в специальном файле с расширением “db” и считываются легковесным программным обеспечением, доступным на многих языках программирования. Для взаимодействия с базой данных нужна ORM библиотека. Сравнение самых популярных ORM технологий приведено в таблице 2.2.1.

Таблица 2.2.1 – Потенциальные ORM для работы с SQLite

Платформа и язык программирования	**Преимущества**	**Недостатки**
Sequelize [10]	Популярность, гибкость, поддержка Database First и Code First	Сложное создание модели через подход Code First

TypeORM [13]	Популярность, гибкость, поддержка Code First	Отсутствие поддержки подхода Database First, сложное создание модели
Prisma [7]	Популярность, гибкость, поддержка Database First и Code First, собственная разметка для создания моделей, простое создание моделей	Относительно новая технология

При принятии решения учитывалось, что база данных будет создаваться через подход Code First. В данном случае сначала будут описаны все модели, на основе которых ORM сгенерирует SQL скрипт для создания нужных таблиц в БД. Также рассматривалась возможность создания общих зависимостей, чтобы типы базы данных можно было использовать в других интегрируемых приложениях. По данному параметру больше всего подходила ORM Prisma. Данная технология содержит определения моделей не в файле, написанном пользователем, а в сгенерированном файле без зависимостей.

Таблица 2.2.2 – Потенциальные фреймворки для создания REST API

Платформа и язык программирования	**Преимущества**	**Недостатки**
NestJS [3]	Популярность, использование преимуществ TypeScript	Громоздкая реализация
Express [4]	Популярность, гибкость	

Fastify [5]	Гибкость, скорость работы	

Для создания REST API был выбран легковесный и гибкий фреймворк Express. Он также очень популярен, что облегчит поиск информации о нем в интернете.

Для создания WebSocket API есть две очень похожие технологии: “ws” и “socket.io”. Для реализации был выбран socket.io, так как он имеет больший функционал и предоставляет возможность запустить на одном порту с Express.

На основе проведенного анализа были выявлены технологии, которые были применены разработке (таблица 2.2.3).

Таблица 2.2.3 – Используемые технологии

№	Название	Версия	Назначение
1	Express.js [4]	4.17.2	REST сервер
2	socket.io [6]	4.5.0	WebSocket сервер
3	Prisma [7]	3.10.0	ORM для работы с SQLite
4	simple-node-logger	21.8.12	Логирование в файл
5	@tinkoff/invest-openapi-js-sdk	1.5.0	API торговой системы

2.3. Архитектура робота

В соответствии с изложенным выше описанием приложения, была разработана архитектура робота, представленная на рисунке 2.3.1 в формате UML диаграммы.

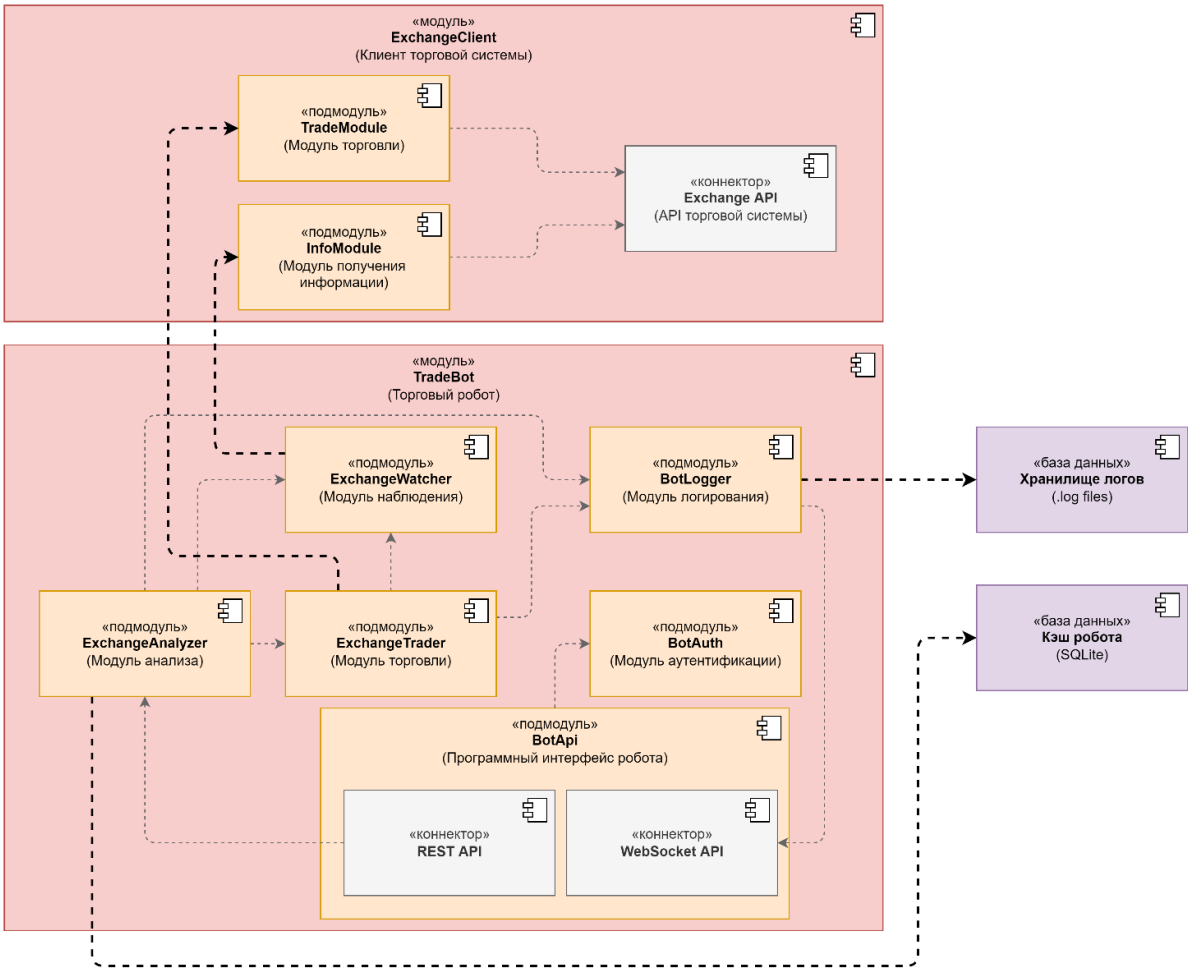

Рисунок 2.3.1. Архитектура торгового робота

Робот состоит из следующих элементов:

- хранилище логов – хранилище записей об операциях в виде папки с .log файлами;
- кэш робота – хранилище кэша робота в формате базы данных SQLite. Для обращений к базе данных используется ORM Prisma;
- клиент торговой системы (ExchangeClient) – модуль, который взаимодействует с торговой системой через API:
    - модуль торговли (TradeModule) – модуль для отправки торговых поручений на биржу. Получает команды на отправку поручений из **торгового модуля** робота;

- модуль получения информации (InfoModule) – модуль для получения различной информации с биржи по запросам **модуля наблюдения** робота;
- API торговой системы (ExhangeAPI) – коннектор, обеспечивающий связь с торговой системой (предполагается, что это будут SDK для работы с биржей).

- торговый робот (TradeBot) – модуль, реализующий функционал робота:
  - модуль наблюдения (ExchangeWatcher) – компонент, ответственный за преобразование данных из формата биржи в формат кэша робота. Получает информацию по з**апросам модуля анализа**. Для получения информации обращается к **модулю получения информации** в **клиенте торговой системы**;
  - ExchangeTrader (Модуль торговли) – компонент, ответственный за отправку торговых поручений. Получает команды на отправку поручений из **модуля анализа**. Для отправки поручений использует **модуль торговли** в **клиенте торговой системы**;
  - модуль анализа (ExchangeAnalyzer) – компонент, принимающий решения об отправке торговых поручений через **модуль торговли** робота на основе данных, полученных от **модуля наблюдения**. Именно этот модуль взаимодействует с **кэшем робота** посредством ORM Prisma. В том числе хранит и запускает алгоритмы. Вызов методов **модуля анализа** осуществляется через интерфейсы робота (REST API);

- модуль логирования (BotLogger) – специальный модуль для выведения и сохранения логов. Взаимодействует с **хранилищем логов** посредством библиотеки simple-node-logger. Отправляет новые логи в **Web Socket API** для трансляции в реальном времени;
- программный интерфейс робота (BotApi) – модуль, обеспечивающий интерфейс для управления роботом:
  - REST API – интерфейс для вызова методов **модуля анализа** робота. REST сервер создан на основе технологии Express.js;
  - Web Socket API – интерфейс для трансляции информации в реальном времени. WebSocket сервер создан на основе библиотеки Socket.io.
- модуль аутентификации (BotAuth) – модуль для аутентификации запросов на соединение с роботом. Используется REST и Web Socket интерфейсами для защиты от подделанных запросов. Проверка аутентификации осуществляется проверка Bearer токена в заголовках запросов.

Так как различные торговые системы могут иметь различную структуру данных, было принято решение преобразовывать типы данных в некую единую для всех роботов модель (рисунок 2.3.3), которую можно было бы использовать для интеграции сторонних приложений, чтобы те работали независимо от того, к какой торговой системе подключены роботы.

Так как названия типов для данных из торговой системы и для модели данных робота может совпадать, всем типам внутри робота были присвоены префиксы:

- C_ (от Client) – типы данных, полученных из торговой системы;

- D_ (от Database) – типы данных, используемые внутри робота.

Процесс получения, преобразования и сохранения данных представлен на рисунке 2.3.2 в нотации DFD.

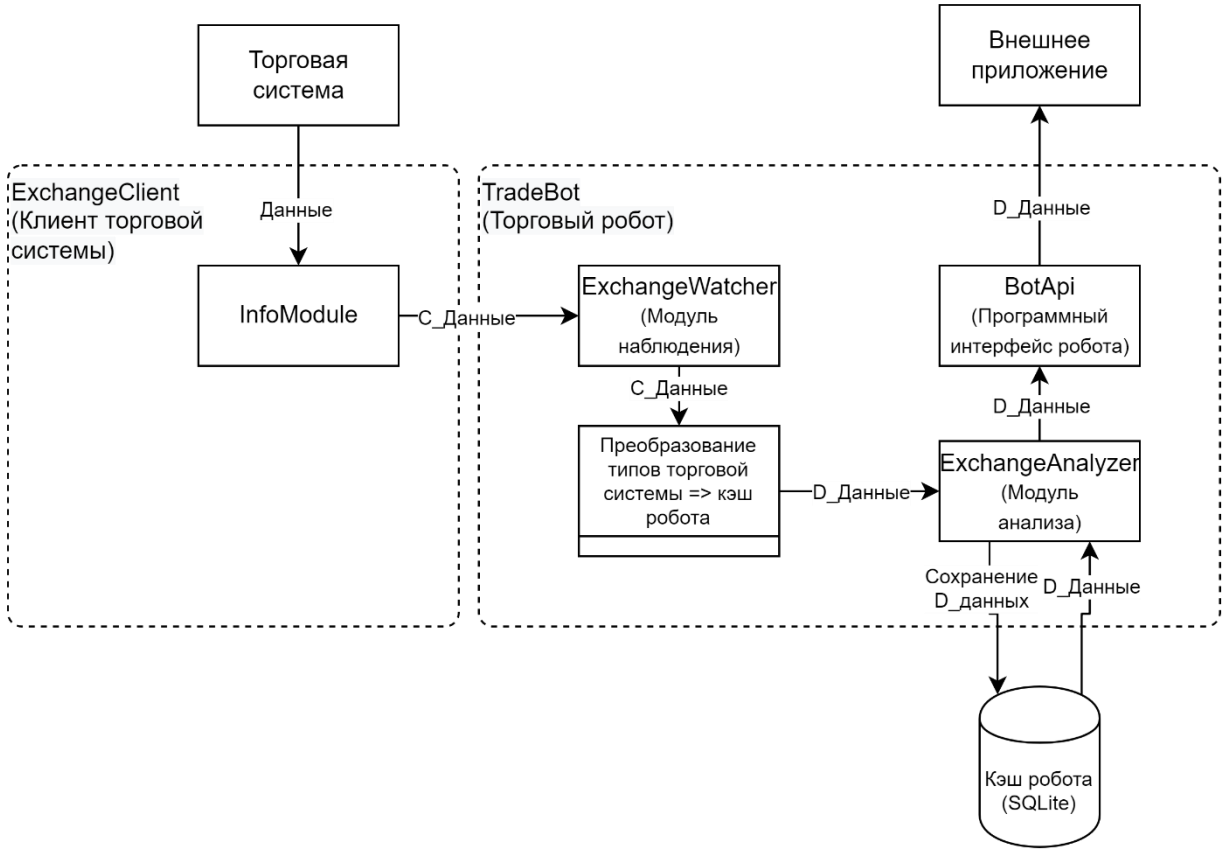

Рисунок 2.3.2. Процесс получения данных в нотации DFD

В зависимости от запроса внешнего приложения существует 2 сценария получения данных.

Первый сценарий – модуль анализа просто делает запрос к SQLite базе данных и возвращает полученные данные в ответ на запрос.

Второй сценарий – внешнее приложение запрашивает такую информацию, которую необходимо запросить в торговой системе. Сначала **модуль получения информации** клиента торговой системы получает нужную информацию в формате данных торговой системы (с префиксом C_). Полученная информация направляется в **модуль наблюдения** робота, где преобразуется в формат данных кэша робота (с префиксом D_). После чего

полученная информация фиксируется в кэше и отправляется во внешнее приложение.

На рисунке 2.3.3 представлена модель данных, используемая роботом для хранения и анализа информации.

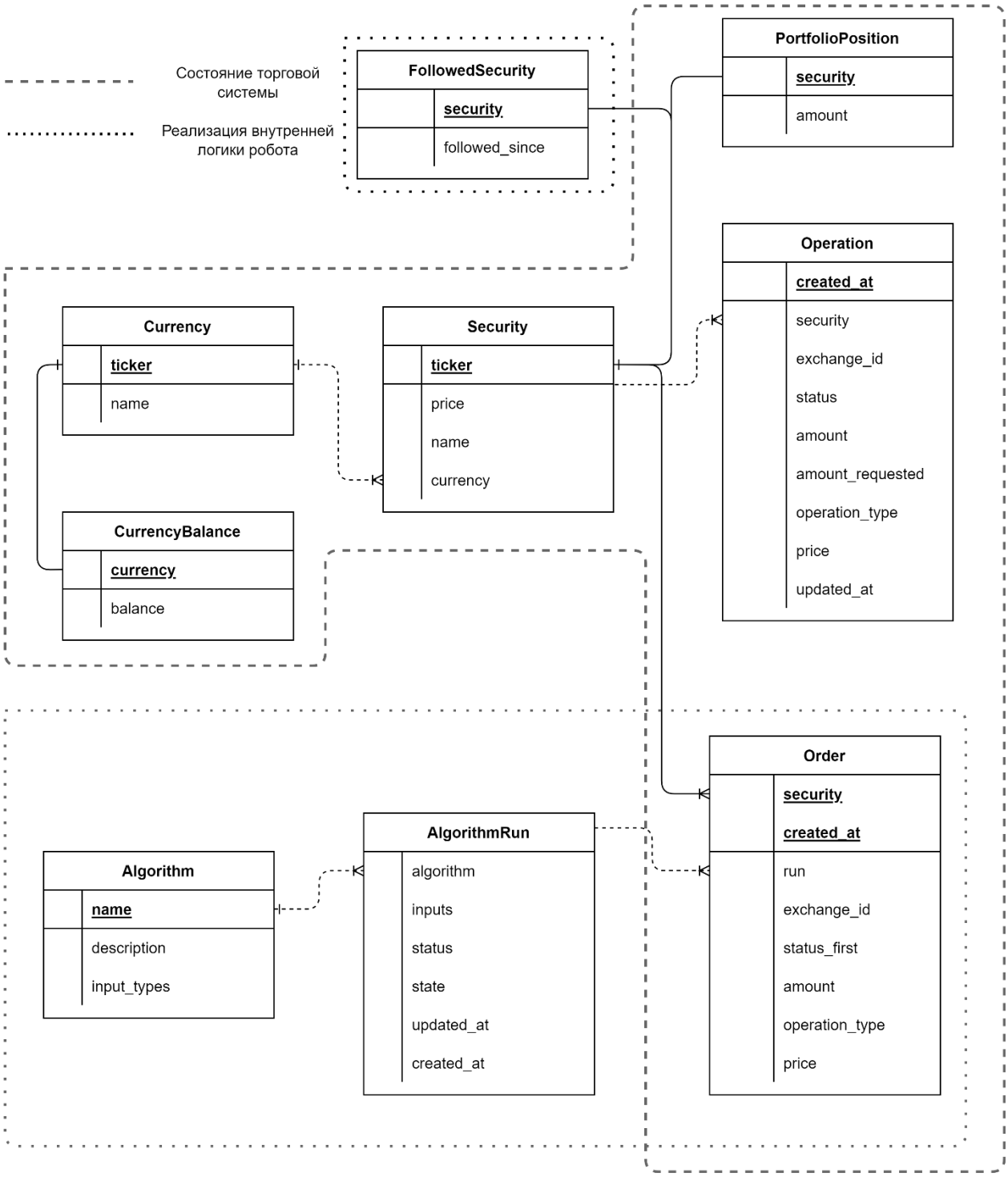

Рисунок 2.3.3. Логическая модель данных робота

В модели данных присутствуют следующие сущности:

- Currency – валюта, в которой происходит торговля:
  - ticker – уникальный идентификатор валюты в формате строки (например, USD для доллара);
  - name – наименование валюты.
- CurrencyBalance – баланс валюты на счете аккаунта торговой системы:
  - currency – привязанная валюта;
  - balance – количество валюты.
- Security – торговый инструмент (например, акция или облигация):
  - ticker – уникальный идентификатор инструмента в формате строки;
  - name – наименование торгового инструмента;
  - price – цена инструмента, измеряемая в привязанной валюте;
  - currency – валюта, в которой измеряется цена инструмента.
- FollowedSecurity – отслеживаемый инструмент. Используется для систематического обновления некоторых интсрументов:
  - security – инструмент, который необходимо отслеживать;
  - followed_since – момент начала отслеживания инструмента.
- PortfolioPosition – позиция в портфеле привязанного аккаунта. Таблица этих сущностей используется для хранения портфеля робота:

- security – инструмент в наличии;
- amount – количество инструментов.

- Operation – запись об операции с активами. Операции могут происходить не только из-за взаимодействий с торговыми инструментами, поэтому все поля, которые к ним относятся, необязательные:
  - instrument – инструмент, с которым совершалась операция (необязательный атрибут);
  - created_at – дата совершения операции;
  - exchange_id – id операции внутри торговой системы (необязательный атрибут);
  - status – статус операции (например, “опубликовано” или “исполнено”);
  - amount – количество лотов инструмента, с которым совершалась операция (необязательный атрибут);
  - amount_requested – количество лотов инструмента, с которым планировалось совершить операцию (необязательный атрибут);
  - operation_type – тип операции (например, “купить” или “продать”);
  - price – цена совершения сделки;
  - updated_at – дата обновления операции.
- Order – отправленные роботом торговые поручения:
  - security – инструмент, на взаимодействие с которым было отправлено поручение;
  - created_at – дата отправки поручения;

- run – запуск алгоритма, в ходе которого было отправлено поручение (необязательный атрибут);
- exchange_id – id поручения внутри торговой системы (необязательный атрибут);
- status_first – статус поручения на момент отправки;
- amount – количество лотов инструмента, с которым запрашивается операция;
- operation_type – тип операции (например, “купить” или “продать”);
- price – цена одного лота в сделке;

- Algorithm – метаинформация об алгоритме, который поддерживает робот:
  - name – имя алгоритма;
  - description – описание алгоритма;
  - input_types – описание входных параметров алгоритма в формате JSON.
- AlgorithmRun – запись о запуске и ходе работы алгоритма:
  - algorithm – запущенный алгоритм;
  - inputs – входные параметры алгоритма, использованные для запуска в формате JSON;
  - state – состояние алгоритма в формате JSON, индивидуально для каждого алгоритма, используется для запуска после сбоя;
  - updated_at – дата последнего обновления запуска алгоритма;
  - created_at – дата запуска алгоритма.

Сущности Currency, CurrencyBalance, Security, PortfolioPosition, Operation созданы для того, чтобы локально запомнить часть состояния торговой системы. Этих сущностей вполне достаточно для составления самых базовых алгоритмов робота. Если для алгоритмов понадобится дополнительный функционал, можно будет использовать напрямую методы для взаимодействия с торговой системой.

Сущности FollowedSecurity, Algorithm, AlgorithmRun служат для внутренней логики робота. Они сохраняют состояние определенных процессов, не принадлежащих к торговой системе.

Сущность Order находится на границе состояния робота и состояния торговой системы, так как о нем возможно получить достоверную информацию только в момент отправки поручения. После отправки поручения, затруднительно получать информацию о его статусе. По этой причине статус хранится в поле с название «status_first».

В соответствии с логической моделью данных (рисунок 2.3.3) была создана физическая модель данных для базы данных SQLite (рисунок 2.3.4).

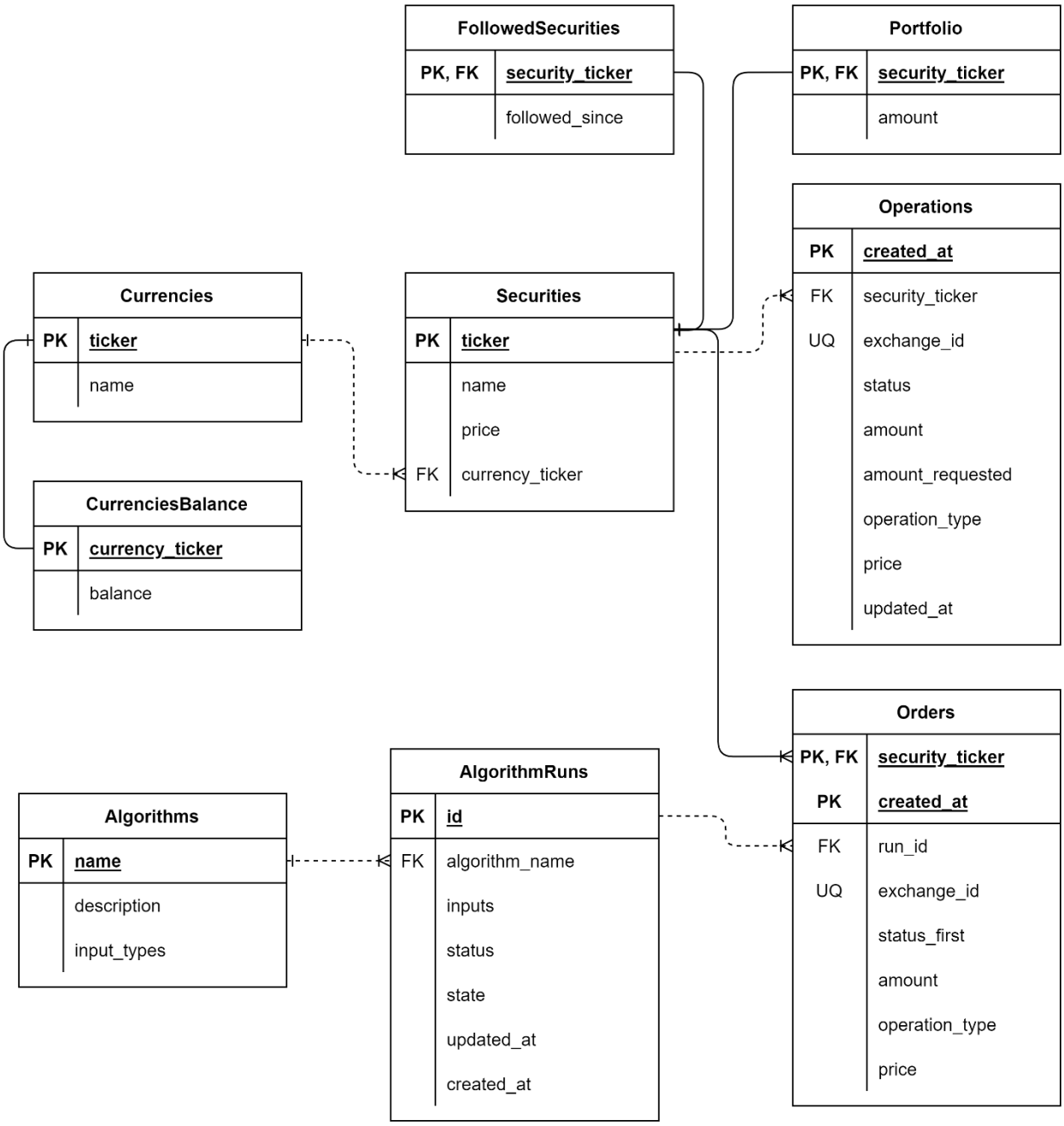

Рисунок 2.3.4. Физическая модель данных робота

В физической модели все сущности были переименованы во множественное число, так как теперь они представляют таблицы в базе данных. Также некоторые атрибуты были изменены в соответствии со спецификой SQL:

- Currencies – используемые валюты:
  - ticker – является первичным ключом.

- CurrenciesBalance – балансы валют на счете:
    - currency_ticker – является идентификатором связанной валюты. Является первичным ключом и используется во внешнем ключе.
- Security – торговые инструменты:
    - ticker – является первичным ключом;
    - currency_ticker – является идентификатором связанной валюты. Используется во внешнем ключе.
- FollowedSecurities – отслеживаемые инструменты:
    - security_ticker – является идентификатором связанного инструмента. Является первичным ключом и используется во внешнем ключе.
- Portfolio – портфель аккаунта, связанного с роботом:
    - security_ticker – является идентификатором связанного инструмента. Является первичным ключом и используется во внешнем ключе.
- Operations – записи о проведенных операциях:
    - created_at – время проведения операции. Является первичным ключом.
    - exchange_id – идентификатор операции внутри торговой системы. Создан для дополнительной возможности поиска нужной операции. Может оставаться пустым. Является частью уникального индекса, чтобы значения не повторялись.
- Orders – записи об отправленных торговых поручениях:

    - security_ticker – является идентификатором связанного инструмента. Является частью составного первичного ключа и используется во внешнем ключе.
    - created_at – время отправки поручения. Является частью составного первичного ключа.
    - exchange_id – идентификатор операции внутри торговой системы. Создан для дополнительной возможности поиска нужной операции. Может оставаться пустым. Является частью уникального индекса, чтобы значения не повторялись.
- Algorithms – алгоритмы, поддерживаемые роботом:
    - name – наименование алгоритма. Используется как первичный ключ.
- AlgorithmRuns – запуски алгоритмов:
    - id – суррогатный ключ для идентификации запусков. Является частью первичного ключа;
    - algorithm_name – имя запущенного алгоритма. Используется во внешнем ключе.

2.4. Структура проекта

Робот создавался без использования какого-либо шаблона от популярного фреймворка по той причине, что для подобных задач шаблоны в открытом доступе отсутствуют. Файловая структура проекта робота представлена на рисунке 2.4.1.

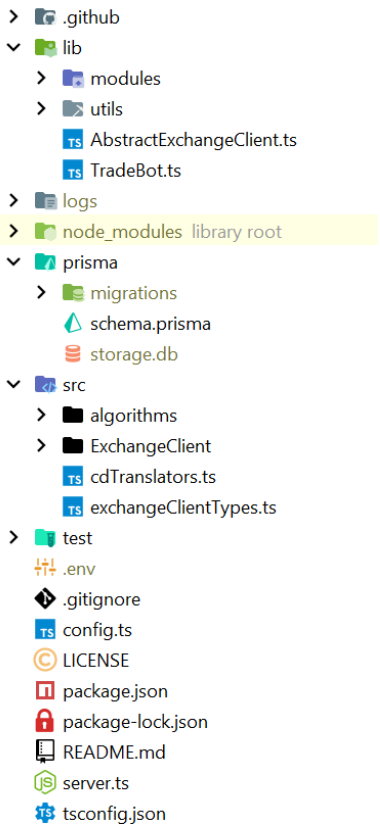

Рисунок 2.4.1. Файловая структура проекта

Папки и файлы проекта имеют следующее предназначение:

- .github – содержит в себе скрипты GitHub Actions для автоматического обновления пакета @badlabs/trade-bot__db-types, содержащего в себе описание структуры модели данных робота;
- lib – папка, содержащая основные файлы, описывающие логику робота:
    - modules – содержит программные реализации подмодулей для AbstractExchangeClient (шаблон клиента торговой системы) и TradeBot (торговый робот);
    - utils – содержит вспомогательные утилиты;
    - AbstractExchangeClient.ts – код шаблонного класса, реализующего клиент торговой системы;

    - TradeBot.ts – код класса, реализующего функционал торгового робота.

- src – хранит в себе файлы, в которых настраивается конкретная реализация робота;
- logs – папка, в которой хранятся логи робота в формате .log файлов;
- prisma – папка, содержащая в себе описание базы данных в формате файлов Prisma ORM и саму базу данных;
- test – сценарии для тестирования корректности работы робота. Применяются при разработке;
- .env – файл с приватными настройками робота, исключен из системы контроля версий;
- config.ts – файл настроек робота;
- server.ts – файл для запуска робота.

Основные файлы робота (lib)

Папка lib сосредотачивает в себе основную логику работы робота, которая не подразумевает изменение сторонним пользователем как при интеграции в новую торговую систему, так и при реализации своих собственных алгоритмов.

Для того, чтобы пользователь смог адаптировать робота для работы с новой торговой системой, созданы специальные шаблоны класса клиента торговой системы, представленные в виде абстрактных классов (рисунок 2.4.2). Наследуемый от абстрактного класс обязан переопределить все абстрактные методы и поля, используя то же имя и те же типы данных, что и в родительском абстрактном классе. Таким образом, используя шаблонные абстрактные классы, можно добиться того, чтобы пользователи создавали совместимые с торговым роботом классы для клиента торговой биржи.

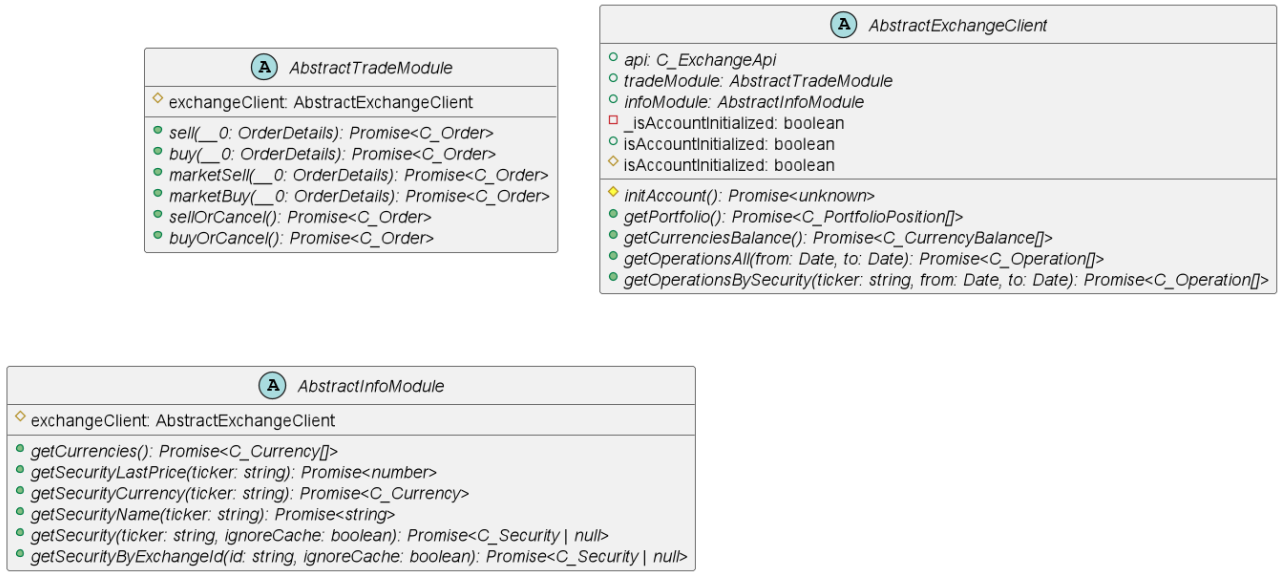

Рисунок 2.4.2. UML диаграмма шаблонных классов клиента торговой системы

Шаблон клиента торговой системы разделен на несколько модулей для того, чтобы логически разделить функционал на различные группы:

- AbstractExchangeClient – класс, который соединяет в себе все модули. Так же содержит методы для получения информации об аккаунте торговой системы (портфель, проведенные операции);
- AbstractInfoModule – модуль для получения информации о торговой системе в целом;
- AbstractTradeModule – модуль для отправки торговых поручений.

AbstractExchangeClient – шаблон непосредственно клиента торговой системы. Содержит в себе поля для работы с торговой системой и методы для получения информации о привязанном к роботу аккаунту системы.

Поле «api» представляет собой пространство библиотеки для интеграции с торговой системой, предоставляемой извне. Данное поле предполагается использовать во всех методах для обращения к торговой системе, однако также допускается прописывать логику взаимодействия с

нуля. Поля «tradeModule» и «infoModule» представляют собой ссылки на шаблоны модулей клиента.

Класс торгового робота (рисунок 2.4.3) представляет собой объединение функциональных модулей и ссылки на клиент торговой системы (рисунок).

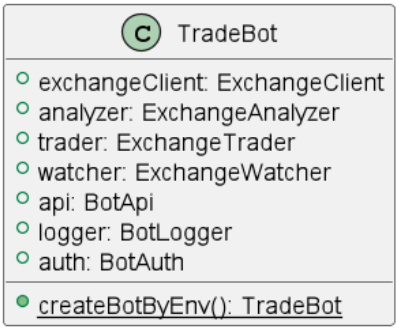

Рисунок 2.4.3. UML диаграмма класса торгового робота

Среди модулей торгового робота можно выделить 3 основных:

- Модуль наблюдения (ExchangeWatcher) – рисунок 2.4.4;
- Модуль торговли (ExchangeTrader) – рисунок 2.4.5;
- Модуль анализа (ExchangeAnalyzer) – рисунок 2.4.6.

И 3 вспомогательных модуля (рисунок 2.4.8):

- Программный интерфейс робота (BotApi);
- Модуль аутентификации (BotAuth);
- Модуль логирования (BotLogger).

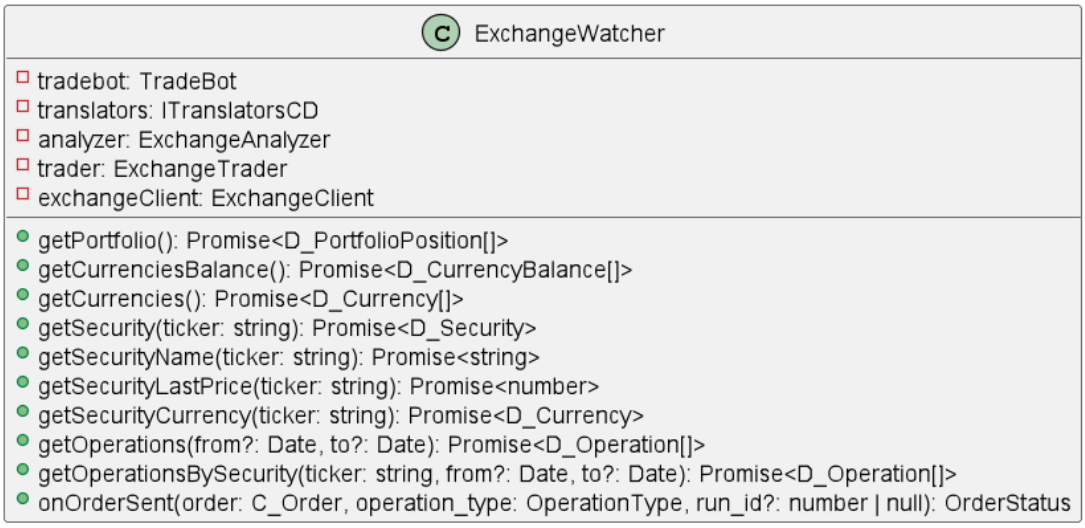

Рисунок 2.4.4. UML диаграмма класса ExchangeWatcher (Модуль наблюдения)

Модуль наблюдения содержит ссылки на торгового робота и некоторые его модули. Важной частью данного модуля является поле «translators», которое является коллекцией методов для перевода типов торговой системы (с префиксом “C_”) в типы кэша робота (с префиксом “D_”). В основном данный модуль содержит методы для запроса информации из торговой системы.

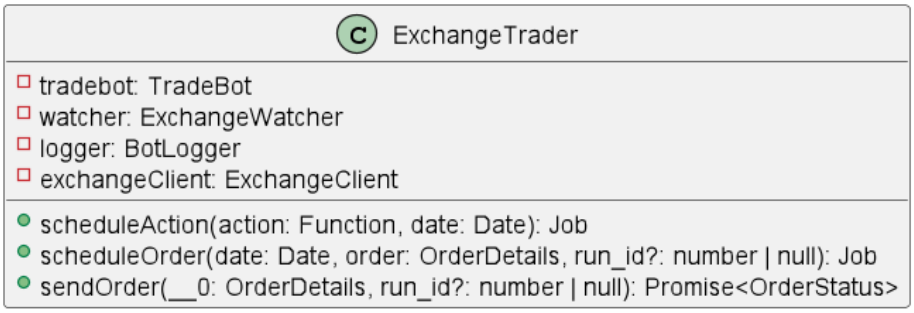

Рисунок 2.4.5. UML диаграмма класса ExchangeTrader (Модуль торговли)

Модуль торговли содержит методы для отправки торговых поручений в систему (поручения полностью описываются типом OrderDetails):

- sendOrder() – моментально высылается торговое поручение;

- scheduleOrder() – планируется отправка поручение на определенный момент времени в будущем;
- scheduleAction() – планируется определенное действие на определенный момент времени в будущем. Полезно в случаях, когда необходимо запланировать отправку поручения с несколькими сопутствующими действиями.

Рисунок 2.4.6. UML диаграмма класса ExchangeAnalyzer (Модуль анализа)

Модуль анализа содержит в себе методы для вызова функционала модулей наблюдения и торговлю с добавлением взаимодействия с кэшем робота (запись, обновление данных). При создании робота в модуле анализа запускаются процессы для периодического обновления информации об аккаунте при помощи метода initUpdaters().

Также в поле «tradeAlgos» класса TradeAlgorithms (рисунок 2.4.7) хранятся все алгоритмы робота, доступные для удаленного вызова, их исходный код и описание.

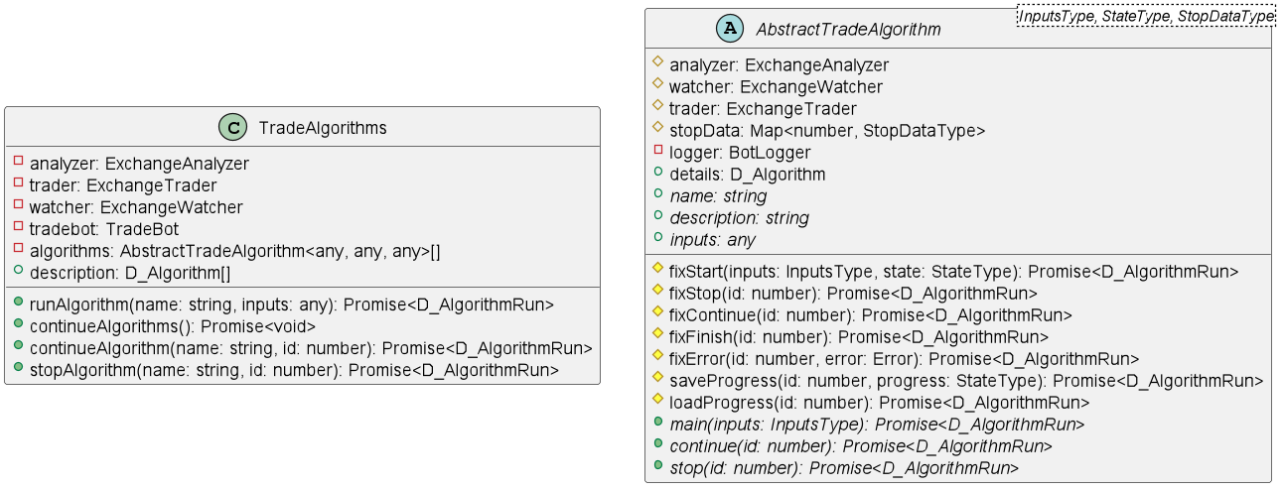

Рисунок 2.4.7. UML диаграмма классов торговых алгоритмов

Класс TradeAlgorithms создан для централизованного вызова алгоритмов при помощи специальных методов:

- runAlgorithm() – запускает определенный алгоритм по его имени;
- continueAlgorithm() – запускает процесс алгоритма после неожиданного сбоя, используя сохраненное состояние процесса из кэша.

Поле «algorithms» содержит в себе исходный код алгоритмов, созданный по шаблону алгоритма AbstractTradeAlgorithm, а свойство «description» предоставляет описание всех имеющихся алгоритмов.

AbstractTradeAlgorithm представляет собой абстрактный класс, на основе которого предполагается создавать торговые алгоритмы для робота.

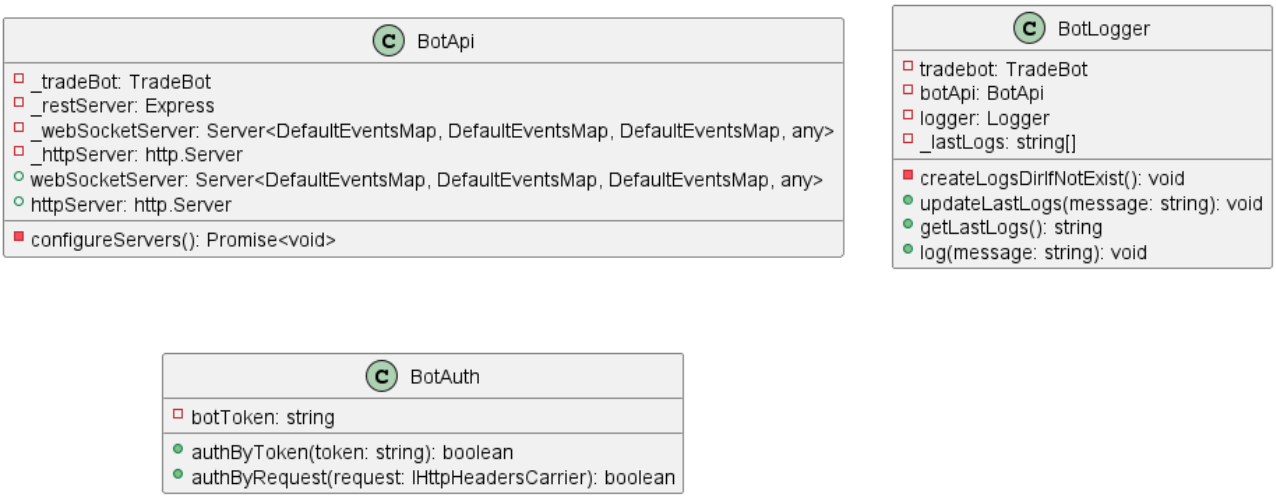

Рисунок 2.4.8. UML диаграмма классов вспомогательных модулей робота

Структура и предназначение вспомогательных модулей робота BotApi, BotLogger, BotAuth достаточно точно описаны на рисунке 2.3.1.

Файлы реализации робота (src)

Предполагается, что, изменяя папку «src» пользователь будет настраивать робота под свои нужды. Тестовая версия робота реализовывалась на основе SDK для Тинькофф Инвестиций. В данном разделе будут кратко описаны имеющиеся примеры реализации. Инструкции о процессе настройки робота под собственные нужды будут рассмотрены в другом разделе.

В папке «src» предусмотрено 4 концепции для определения пользователем при реализации шаблона:

- структуры данных торгового клиента;
- методы для перевода данных торговой системы в данные кэша робота;
- модуль клиента торговой системы на основе шаблона;
- алгоритмы для исполнения роботами.

Файл «exchangeClientTypes.ts» предназначен для того, чтобы определить структуры данных для клиента торговой системы. Это можно сделать стандартным методом, с нуля прописывая все структуры. Но гораздо быстрее использовать уже готовые типы (в данном случае из SDK Тинькофф Инвестиций), реэкспортировав их под новым именем. В том числе есть возможность частично поменять структуры этих типов. В том числе для этих возможностей для разработки был выбран язык TypeScript.

В файле «cdTranslators.ts» находится шаблон функции, которая возвращает набор методов, преобразующих структуры данных клиента торговой система в структуры данных кэша торгового робота.

В каталоге «ExchangeClient» реализуются 3 модуля клиента торговой системы:

- ExchangeClient – основной класс клиента торговой системы;
- InfoModule – модуль клиента торговой системы, обладающий методами для получения информации о системе;
- TradeModule – модуль клиента торговой системы, обладающий методами для отправки торговых поручений.

Каталоге «algorithms» предназначен для создания внутри неограниченного числа алгоритмов торгового робота, используя шаблон «AbstractTradeAlgorithm» и его интерфейсы. Коллекция алгоритмов будет использоваться модулем анализа для автоматизации.

Настройки робота (config.ts)

Файл «config.ts» содержит в себе настройки робота, которые могут различаться у различных его копий. Структура настроек приведена на рисунке 2.4.9.

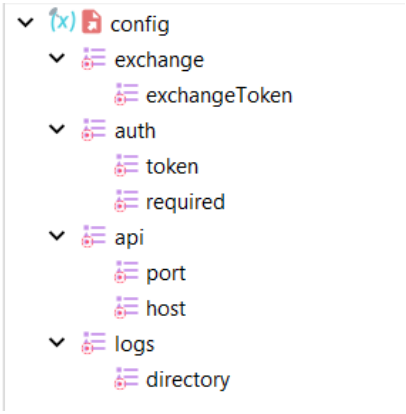

Рисунок 2.4.9 Структура настроек робота.

В структуре можно наблюдать следующие опции:

- Торговая система:
    - Токен доступа к торговой системе.
- Аутентификация:
    - Токен доступа к данному роботу;
    - Отключение/включение проверки запросов.
- Программные интерфейсы робота:
    - Порт прослушивания запросов;
    - Хост прослушивания запросов.
- Логи робота:
    - Папка для хранения файлов с логами.

В настройки можно добавлять новые опции и использовать при реализации собственного функционала. Когда робот будет полностью подготовлен для интеграции, можно будет во множестве его копий поменять данные настройки для доступа к разным аккаунтам и прослушивания по разным адресам и запустить.

2.5. Программные интерфейсы робота

Робот предоставляет программные интерфейсы для управления извне при помощи модуля BotApi. Данный модуль использует интеграции 2 типов:

- REST API - реализуется при помощи библиотеки Express;
- WebSocket – работает на основе socket.io.

Был разработан ряд REST запросов, позволяющих получить доступ к определенным методам модуля анализа робота. Условно можно разделить запросы на 3 группы:

- Алгоритмы – запросы для взаимодействия с алгоритмами (таблица 2.5.1);
- Аутентификация – запросы для проверки доступа к роботу (таблица 2.5.2);
- Робот – управление различными сущностями робота (таблица 2.5.3).

Таблица 2.5.1 – REST запросы робота для работы с алгоритмами.

Адрес	**Тип запроса**	**Предназначение**
`/api/algos/`	GET	Получение информации о всех алгоритмах
`/api/algos/:algorithm_name/runs`	GET	Получение информации о запусках определенного алгоритма
`/api/algos/:algorithm_name`	POST	Запуск определенного алгоритма

/api/algos/:algorithm_name/stop/:run_id	POST	Приостановка процесса алгоритма
/api/algos/:algorithm_name/continue/:run_id	POST	Продолжение процесса алгоритма

Таблица 2.5.2 – REST запросы робота для работы с аутентификацией.

Адрес	**Тип запроса**	**Предназначение**
/api/auth/check	GET	Проверка корректности аутентификации запроса.

Таблица 2.5.3 – REST запросы робота для работы с различными сущностями робота.

Адрес	**Тип запроса**	**Предназначение**
/api/state/currencies	GET	Получение всех известных валют
	POST	Обновление всех валют
/api/state/securities	GET	Получение всех известных торговых инструментов
	POST	Обновление всех известных торговых инструментов
/api/state/securities/followed	GET	Получение всех отслеживаемых торговых инструментов
	POST	Обновление всех отслеживаемых инструментов

/api/state/securities/follow	POST	Добавление торгового инструмента в отслеживаемые
/api/state/securities/unfollow	POST	Исключение торгового инструмента из отслеживаемых
/api/state/portfolio	GET	Получение портфеля, привязанного к аккаунту
	POST	Обновление портфеля, привязанного к аккаунту
/api/state/portfolio/clear	POST	Очистка портфеля робота в кэше
/api/state/portfolio/:ticker/average-buy-price	GET	Получение средней цены покупки позиции в портфеле
/api/state/orders	GET	Получение отправленных торговых поручений
/api/state/operations	GET	Получение операций, привязанных к аккаунту робота
	POST	Обновление операций, привязанных к аккаунту робота
/api/state/operations/:instrument-ticker	GET	Получение операций по конкретному инструменту

WebSocket соединение используется для передачи логов робота в реальном времени. Логи из всех модулей получает модуль BotLogger, который отправляет их в BotApi.

ГЛАВА 3. АПРОБАЦИЯ РОБОТОВ

Когда разработка была завершена, было необходимо удостовериться в том, что всё работает корректно. API интерфейсы создавались с расчетом на то, что будет создано стороннее приложение, интегрированное с роботом. Однако, пока интегрированных решений нет, для проверки функций было решено использовать HTTP клиент для отправки запросов.

3.1. Создание собственной реализации робота

Робот спроектирован так, чтобы его потенциально можно было использовать для любой торговой системы. По задумке адаптировать робота к определенной торговой системе должен человек, обладающий навыками программирования. Это могут быть как специалисты компании, так и энтузиасты из сообщества программистов.

Для использования робота в необходимой пользователю торговой системе и в соответствии с его нуждами необходимо:

1. Переопределить типы данных торговой системы в файле «exchangeClientTypes.ts» (рисунок 3.1.1);
2. Определить методы для перевода данных в формат кэша робота (рисунок 3.1.2);
3. Создать клиент торговой системы по шаблону (рисунки 3.1.3 – 3.1.5);
4. Создать собственные алгоритмы для действий робота (рисунок 3.1.6).

```
import OpenAPI, {
    Order,
    Portfolio,
    Currency,
    MarketInstrument,
    Operation, CurrencyPosition
} from "@tinkoff/invest-openapi-js-sdk";

export type C_ExchangeApi = OpenAPI
export type C_Order = Omit<Order, 'operation'> &
    { operation: 'Buy'| 'Sell' |
            'MarketBuy' | 'MarketSell' |
            'BuyOrCancel' | 'SellOrCancel' }
export type C_Portfolio = Portfolio
export type C_Currency = Currency
export type C_CurrencyBalance = CurrencyPosition
export type C_Security = MarketInstrument
export type C_Operation = Operation
```

Рисунок 3.1.1 Переопределение типов данных клиента торговой системы.

Типы данных торговой системы можно как написать с нуля, так и реэкспортировать из готового пакета для работы с торговой системой.

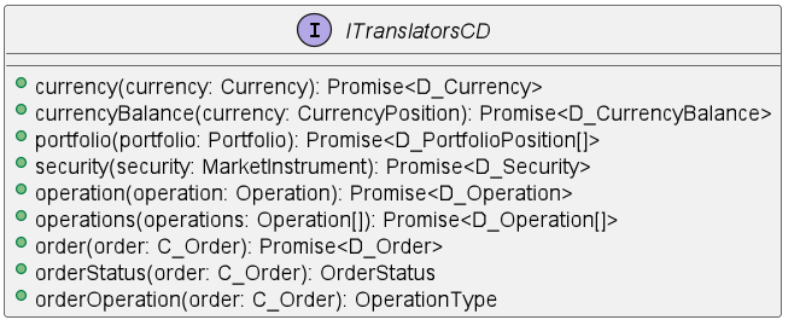

Рисунок 3.1.2. Интерфейс для перевода типов данных торговой системы в кэш робота.

Методы для преобразования данных должны содержаться в объекте, реализующем интерфейс ITranslatorCD. При создании данных методов доступны методы торговой системы для получения данных на случай, если для преобразования будет недостаточно входных данных.

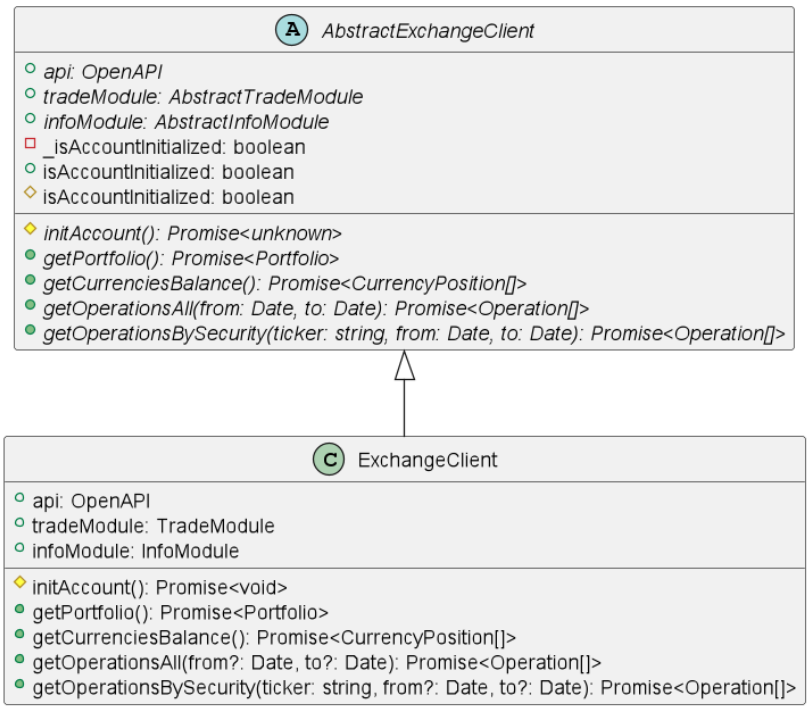

Рисунок 3.1.3 Класс торгового клиента, реализующий шаблон

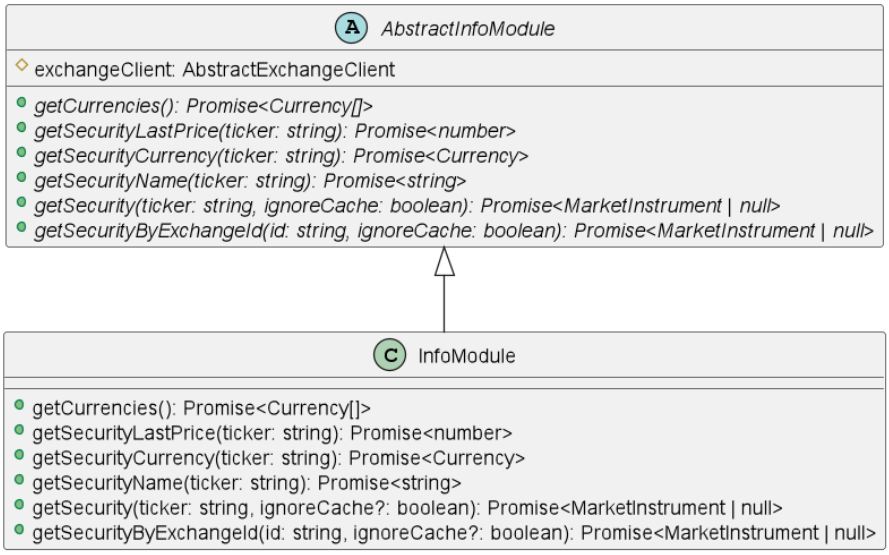

Рисунок 3.1.4 Класс информационного модуля, реализующий шаблон

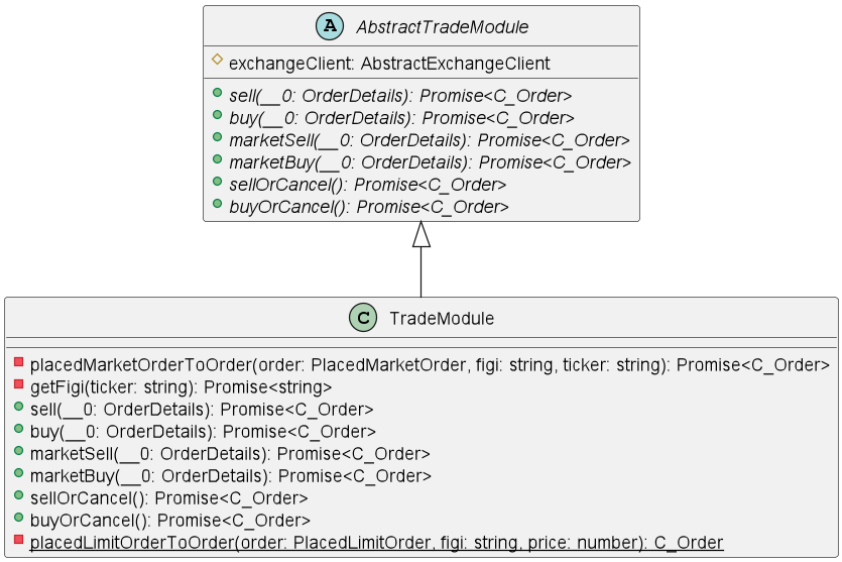

Рисунок 3.1.5 Класс торгового модуля, реализующий шаблон

Для торгового клиента в шаблонах уже подготовлены необходимые методы и ссылки на другие модули. Однако при реализации можно дополнять клиент своей логикой при необходимости, как сделано в торговом модуле (рисунок 3.1.5).

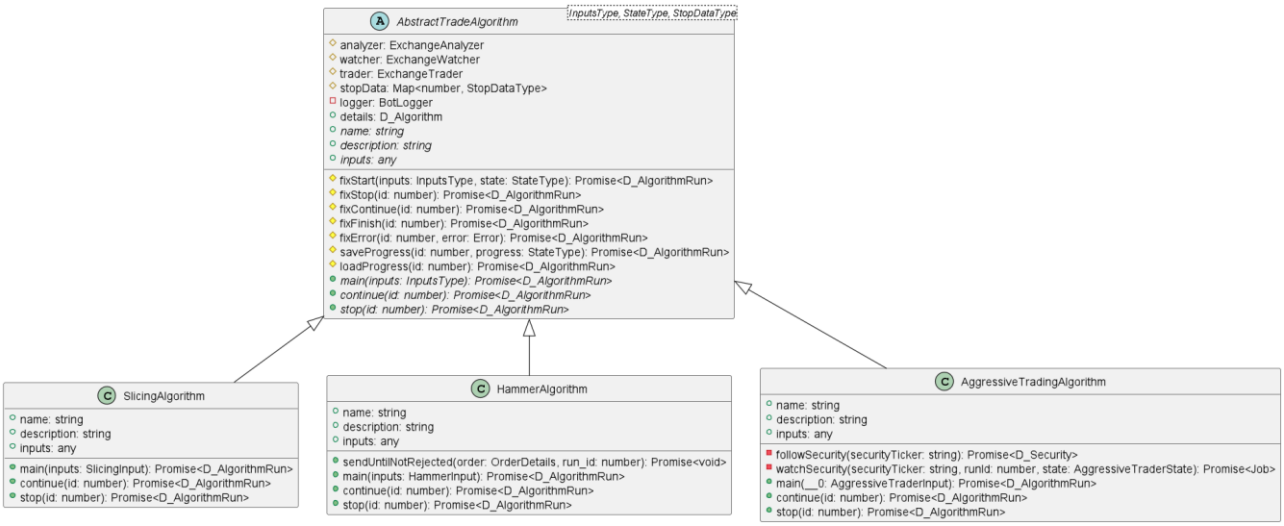

Рисунок 3.1.6 Классы торговых алгоритмов, реализующих единый шаблон

В шаблонном классе торговых алгоритмов уже заранее подготовлены ссылки на необходимые для взаимодействия объекты и методы для манипуляции с состоянием алгоритма в базе данных:

- fixStart() – зафиксировать запуск алгоритма а базе данных;

- fixStop() – зафиксировать остановку алгоритма в базе данных;
- fixContinue() – зафиксировать продолжение алгоритма в базе данных;
- fixFinish() – зафиксировать завершение алгоритма;
- fixError() – зафиксировать ошибку;
- saveProgress() – сохранение прогресса алгоритма;
- loadProgress() – загрузка прогресса алгоритма.

Пользователю необходимо только описать:

- метаданные алгоритма (название, описание, формат входных данных);
- выполнение алгоритма при запуске и после сбоя;
- остановку алгоритма.

В случае необходимости в алгоритме можно использовать напрямую API торговой системы, однако в данном случае алгоритм не сможет работать с другими системами.

В предоставленном примере реализовано 3 алгоритма:

- «Нарезание» (рисунок 1.5.1);
- «Молоток» (рисунок 1.5.2);
- «Паникующий продавец/покупатель» (рисунок 1.5.3).

3.2. Инициализация и подготовка к работе

Запустить один экземпляр робота для собственных нужд достаточно просто. Однако роботы задумывались так, чтобы они работали во множественных экземплярах. В таком случае, необходимы навыки системного администрирования.

Роботов рекомендуется разворачивать в приватной сети, чтобы минимизировать риски их взлома. Роботов должно быть множество и, как следствие, администрировать сертификаты безопасности для каждого адреса затруднительно. Поэтому рекомендуется использовать внешние веб-клиенты без SSL шифрования, иначе запросы к незашифрованным адресам роботов будут блокироваться.

Создание множества копий проекта робота и редактирование настроек вручную очень трудозатратно при разворачивании и отнять много времени при обновлении (рисунок 3.2.1).

Рисунок 3.2.1. Процесс ручного разворачивания роботов в системе

Поэтому рекомендуется создать docker образ, зафиксировать настройки для каждого робота в специальном манифесте и развернуть роботов при помощи docker-compose или Kubernetes (рисунок 3.2.2).

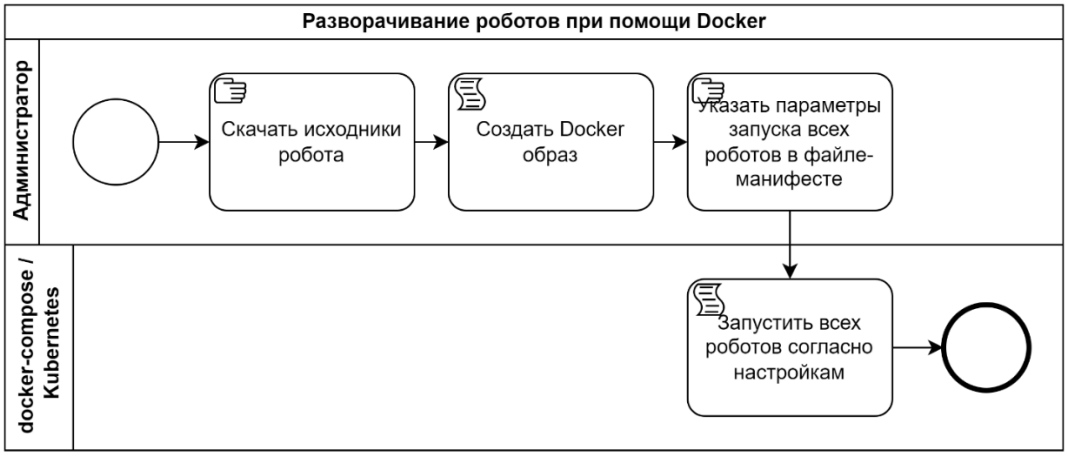

Рисунок 3.2.2. Процесс разворачивания роботов в системе при помощи Docker образов

3.3. Использование робота для симуляции пользовательской активности

Робот предоставляет программные интерфейсы для создания приложения, которое сможет управлять всеми роботами централизовано. Ведь наибольшую ценность данная разработка представляет, когда роботы оказывают серьёзное влияние на определенную часть системы. А это проще организовать, когда запущено множество экземпляров.

Однако, есть возможность управлять данным роботом. Для этих целей опубликована коллекция запросов для клиента Postman. Здесь есть возможность поменять параметры адреса робота и параметры аутентификации. Рассматривался так же вариант со Swagger для документации запросов, однако он не поддерживает соединение по веб-сокету и оказывает дополнительную нагрузку на робота.

Далее будет рассмотрен сценарий управления через Postman:

Сначала идет подключение к веб-сокету во вкладке «Logs» (рисунок 3.3.1).

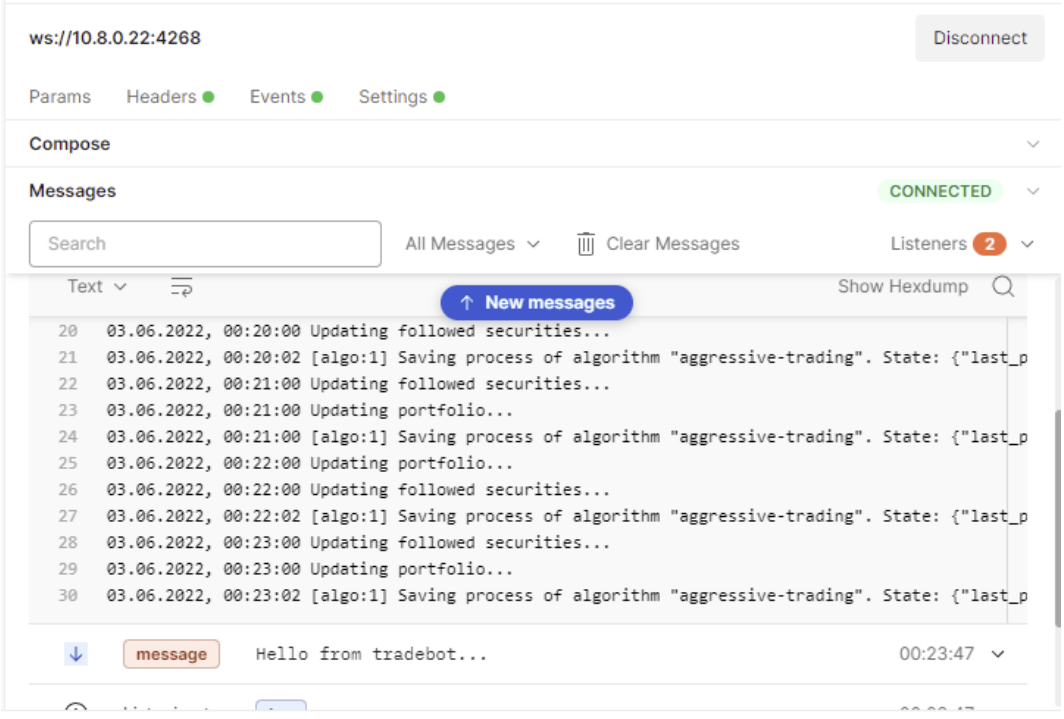

Рисунок 3.3.1. Логи при подключении к роботу

При подключении нам передается приветственное сообщение и недавняя история логов. Далее логи доставляются по мере их появления (рисунок 3.3.2).

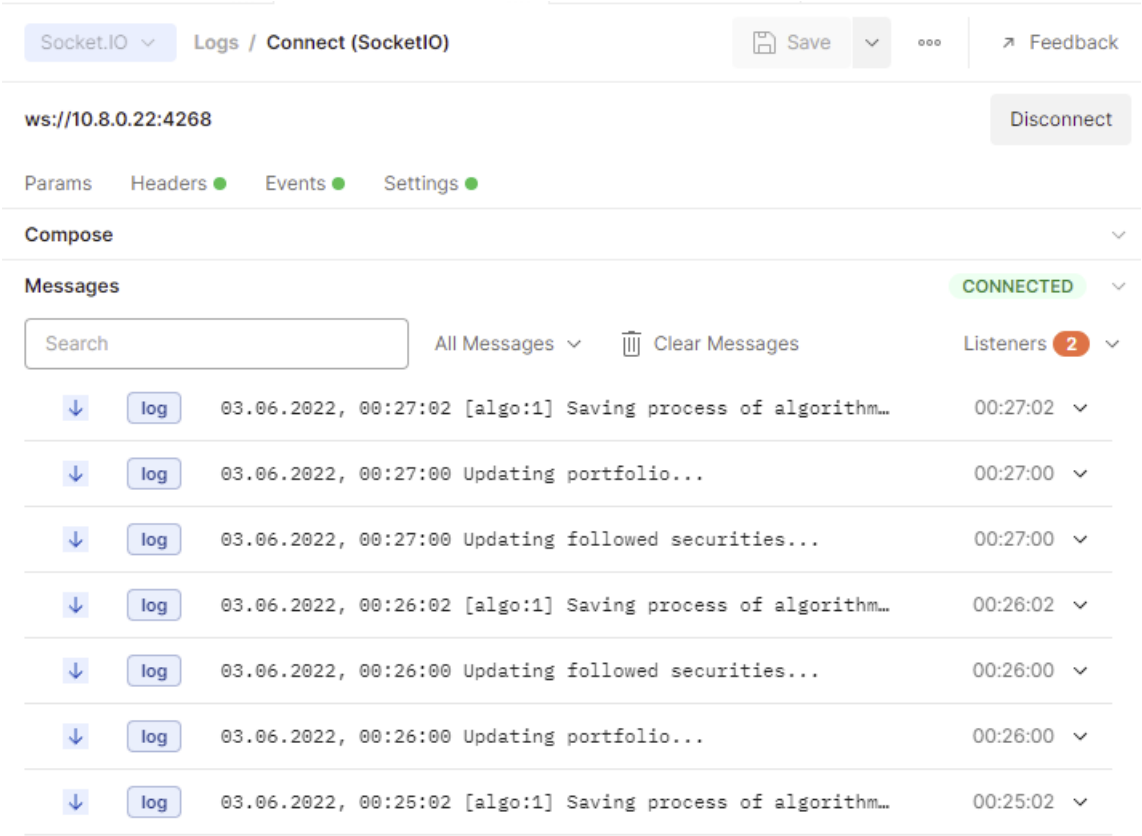

Рисунок 3.3.2. Логи робота в реальном времени

Теперь, когда возможно просматривать реакцию робота на различные действия, нужно отправить запрос для получения состояния портфеля (рисунок 3.3.3).

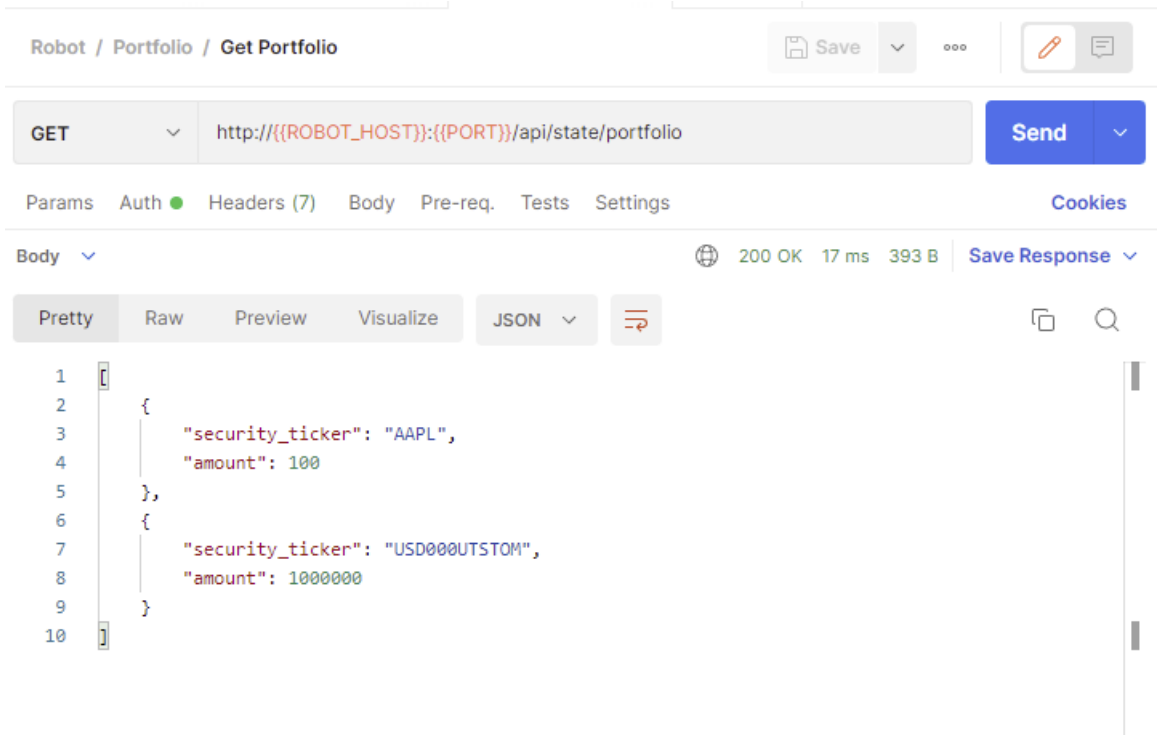

Рисунок 3.3.3. Состояние портфеля робота

На счету робота есть 100 акций Apple и 1 миллион долларов. Применим алгоритм нарезания, результаты действия которого можно увидеть моментально. Для этого необходимо запросить информацию об алгоритмах и отправить на исполнение нужный алгоритм с правильными параметрами (рисунок 3.3.4).

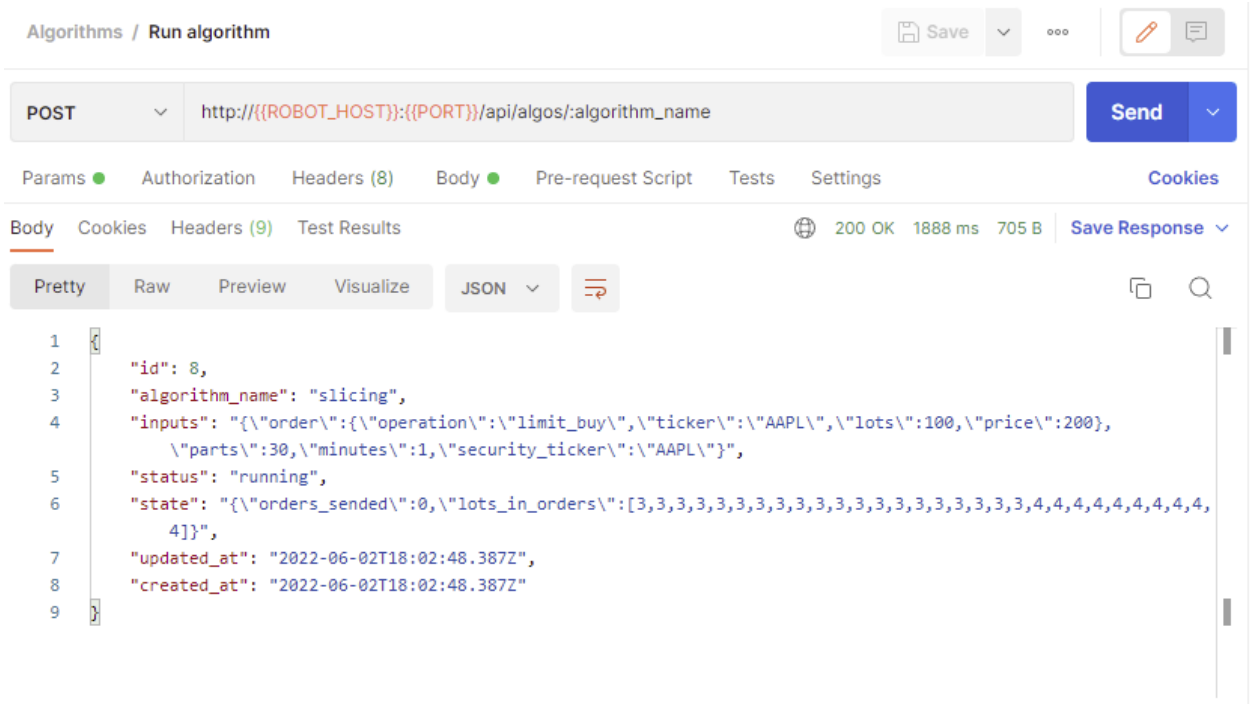

Рисунок 3.3.4. Результат запуска алгоритма «Нарезания»

Алгоритм заранее просчитал все торговые поручения, которые нужно будет отправить и отобразил это в поле state в пришедшем ответе. Реакцию также можно наблюдать в логах (рисунок 3.3.5).

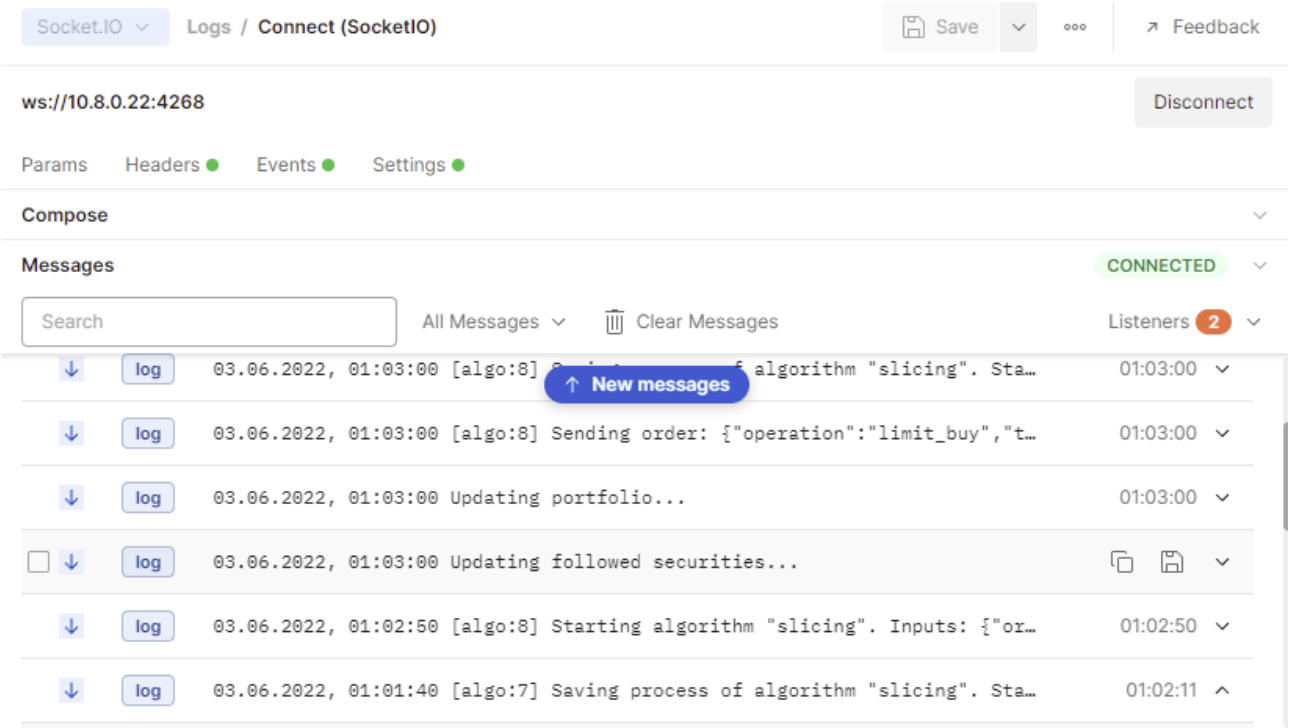

Рисунок 3.3.5. Реакция на запуск алгоритма в логах

ГЛАВА 4 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

Разработка НИ производится группой работников, состоящей из трех человек – руководителя и двух студентов.

Данная выпускная квалификационная работа заключается в разработке программного решения позволяющего планировать и автоматически осуществлять отправку заявок на торговых биржах. Робот будет предоставлять необходимые интерфейсы для подключения к нему панели управления. Исходный код робота можно дополнить собственными алгоритмами торговли, что увеличивает круг потенциальных пользователей.

Целью раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» является определение перспективности и успешности НИ, оценка его эффективности, уровня возможных рисков, разработка механизма управления и сопровождения конкретных проектных решений на этапе реализации.

Для достижения обозначенной цели необходимо решить следующие задачи:

1. Оценить коммерческий потенциал и перспективность разработки НИ;
2. Осуществить планирование этапов выполнения исследования;
3. Рассчитать бюджет затрат на исследования;
4. Произвести оценку научно-технического уровня исследования и оценку рисков.

4.1 Потенциальные потребители результатов исследования

В качестве потенциальных потребителей результатов разработки программного продукта «Биржевой торговый робот» выступают торговые биржи, где автоматизация большого количества действий помогла бы с тестированием их информационных систем. В том числе продуктом могут заинтересоваться трейдеры для автоматизации своих стратегий.

Примером компании потребителя является компания ПАО «Московская Биржа».

4.2 Анализ конкурентных технических решений

Анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения позволяет провести оценку сравнительной эффективности научной разработки и определить направления для ее будущего повышения.

Целесообразно проводить данный анализ с помощью оценочной карты. Оценочная карта для сравнения конкурентных технических решений представлена в таблице 4.1.

Таблица 4.1 – Оценочная карта для сравнения конкурентных технических решений

Критерии оценки	Вес критерия	Баллы			Конкурентоспособность		
		$Б_Б$	$Б_А$	$Б_Р$	$К_Б$	$К_А$	$К_Р$
1	2	3	4	5	6	7	8
Технические критерии оценки ресурсоэффективности							
1. Возможность настройки под разные биржи	0,137	10	0	0	1,373	0,000	0,000
2. Возможность реализации собственных алгоритмов	0,196	9	2	7	1,235	0,275	0,961
3. Возможность реализации распределенной архитектуры	0,059	10	2	2	1,373	0,275	0,275
4. Централизованное управление несколькими аккаунтами	0,118	10	0	0	1,373	0,000	0,000
5. Качество интеграции в систему отдельной биржи	0,196	6	10	8	0,824	1,373	1,098
Экономические критерии оценки эффективности							
6. Стоимость интеграции с новой биржей	0,196	10	5	5	1,373	0,686	0,686
7. Затраты на обучение пользованию технологией	0,098	7	10	5	0,961	1,373	0,686
Итого	**1**	**62**	**29**	**27**	**8,510**	**3,980**	**3,706**

Где $Б_Б$ – торговые роботы «Badlabs», производимые в рамках дипломной работы;

$Б_А$– торговые роботы от Альфа-Банка;

$Б_Р$– торговые роботы «RobotAtentis».

Анализ конкурентных технических решений определяется по формуле:

$$К = \sum В_i \times Б_i \quad (4.1)$$

где К – конкурентоспособность вида;

$В_i$– вес критерия (в долях единицы);

$Б_i$ – балл i-го показателя.

По данным оценочной карты можно увидеть, что торговые роботы «Badlabs», разрабатываемые в рамках данной работы, имеют конкурентоспособность, в разы превышающую другие решения. В основном это вызвано возможностью решения работать с любой биржей. В то время как все остальные решения созданы только для одной конкретной системы.

4.3 SWOT-анализ

Произведем также в данном разделе SWOT – анализ НИ, позволяющий оценить факторы и явления, способствующие или препятствующие продвижению проекта на рынок.

На первом этапе SWOT анализа в таблице 4.2 были описаны сильные и слабые стороны проекта, выявлены возможности и угрозы реализации НИ.

Таблица 4.2 – Матрица SWOT анализа

Сильные стороны	Возможности во внешней среде
С1. Возможность интеграции в различные биржевые системы. С2. Возможность создания распределенной системы роботов. С3. Возможность централизованного управления множеством роботов. С4. Популярный стек технологий.	В1. Внедрение продукта в новые биржевые системы. В2. Продажа исходного кода физическим лицам. В3. Доработка продукта для совместимости с популярными микросервисными фреймворками.
Слабые стороны	Угрозы внешней среды
Сл1. Относительно низкое качество интеграции с одной биржевой системой по сравнению со специализированными продуктами. Сл2. Необходимость доработки исходного кода для внедрения собственного функционала.	У1. Устаревание используемого стека технологий. У2. Создание более конкурентоспособного аналога продукта. У3. Крах фондового рынка.

Сл3. Необходимость обладания навыками, напрямую не относящимися к торговле, для создания полноценной системы роботов.	

Второй этап состоит в выявлении соответствия сильных и слабых сторон научно-исследовательского проекта внешним условиям окружающей среды. Это соответствие или несоответствие должны помочь выявить степень необходимости проведения стратегических изменений. В рамках данного этапа необходимо построить интерактивную матрицу проекта. Ее использование помогает разобраться с различными комбинациями взаимосвязей областей матрицы SWOT. Возможно использование этой матрицы в качестве одной из основ для оценки вариантов стратегического выбора. Каждый фактор помечается либо знаком «+» (означает сильное соответствие сильных сторон возможностям), либо знаком «-» (что означает слабое соответствие); «0» – если есть сомнения в том, что поставить «+» или «-». Интерактивная матрица проекта представлена в табл. 4.3.

Таблица 4.3 - Интерактивная матрица сильных и слабых сторон и возможностей

	Сильные стороны					Слабые стороны		
Возможности проекта		С1	С2	С3	С4	Сл1	Сл2	Сл3
	В1	+	+	+	+	+	+	+
	В2	+	-	-	+	+	+	0
	В3	+	+	+	-	-	+	-

Таблица 4.4 - Интерактивная матрица сильных сторон и слабых сторон и угроз

	Сильные стороны					Слабые стороны		
Угрозы проекта		С1	С2	С3	С4	Сл1	Сл2	Сл3
	У1	0	-	-	+	-	+	-
	У2	0	-	-	-	0	0	0

	У3	+	-	-	-	-	-	-

Самой большой угрозой для проекта является устаревание используемого стека технологий. При осуществлении этой угрозы будет затруднительно распространять продукт с условием, что нужна доработка исходного кода – всё меньше специалистов смогут внедрять собственный функционал.

Хочется отметить, что критических слабых сторон не наблюдается. Необходимость доработки исходного кода нивелируется популярным стеком технологий, а относительно низкое качество интеграции и необходимость обладания базовыми навыками построения информационных сетей потенциально могут быть решены по мере развития продукта.

В рамках третьего этапа составляется итоговая матрица SWOT-анализа, представленная в таблице 4.5.

Таблица 4.5 - Итоговая матрица SWOT-анализа

	Сильные стороны научно-исследовательского проекта: С1. Возможность интеграции в различные биржевые системы. С2. Возможность создания распределенной системы роботов. С3. Возможность централизованного управления множеством роботов. С4. Популярный стек технологий.	Слабые стороны научно-исследовательского проекта: Сл1. Относительно низкое качество интеграции с одной биржевой системой по сравнению со специализированными продуктами. Сл2. Необходимость доработки исходного кода для внедрения собственного функционала. Сл3. Необходимость обладания навыками, напрямую не относящимися к торговле, для создания полноценной системы роботов.
Возможности:	Торговые роботы разработаны таким	Внедрение продукта в большое количество

В1. Внедрение продукта в новые биржевые системы. В2. Продажа исходного кода физическим лицам. В3. Доработка продукта для совместимости с популярными микросервисными фреймворками.	образом, чтобы можно было без труда внедрять их в различные системы.	информационных систем может привести к образованию сообщества пользователей роботами и базы знаний по продукту, что нивелирует слабые стороны.
Угрозы: У1. Устаревание используемого стека технологий. У2. Создание более конкурентоспособного аналога продукта. У3. Крах фондового рынка.	Устаревание используемого стека технологий может усложнить интеграцию с новыми системами.	Новый продукт от конкурентов может решить проблемы торговых роботов «badlabs» и стать более успешным на рынке.

Исходя из результатов SWOT-анализа, можно сделать вывод, что, несмотря на угрозы и слабые стороны проекта, проект можно считать перспективным и успешным.

4.4 Планирование работ по научно-техническому исследованию

4.4.1 Структура работ в рамках научного исследования

Планирование комплекса предполагаемых работ осуществляется в следующем порядке:

- определение структуры работ в рамках научного исследования;
- определение участников каждой работы;
- установление продолжительности работ;
- построение графика проведения научных исследований.

Перечень этапов и работ, распределение исполнителей по данным видам работ приведен в таблице 4.6.

Таблица 4.6 – Перечень этапов, работ и распределение исполнителей

Основные этапы	№ раб	Содержание работ	Должность исполнителя
Разработка технического задания	1	Составление и утверждение технического задания	Руководитель
Создание архитектуры проекта	2	Разработка общей архитектуры торговых роботов	Бакалавр
	3	Подбор и изучение стека технологий для реализации проекта	Бакалавр
	4	Календарное планирование работ	Руководитель Бакалавр
Разработка торговых роботов	5	Разработка робота как серверного приложения	Бакалавр
	6	Интеграция робота и панели управления	Бакалавр
Обобщение и оценка результатов	7	Оценка эффективности полученных результатов	Бакалавр

Оформление отчета по НИР	8	Составление пояснительной записки	Бакалавр

4.4.2 Определение трудоемкости выполнения работ

Трудовые затраты в большинстве случаях образуют основную часть стоимости разработки, поэтому важным моментом является определение трудоемкости работ каждого из участников научного исследования.

4.4.3 Разработка графика проведения научного исследования

Наиболее удобным и наглядным представлением проведения научных работ является построение ленточного графика в форме диаграммы Ганта.

Для удобства построение графика, длительность каждого из этапов работ из рабочих дней следует перевести в календарные дни. Для этого необходимо воспользоваться следующей формулой:

$$\mathrm{T_{ki}} = \mathrm{T_{pi}} \cdot \mathrm{k_{кал}}\,, \qquad (4.1)$$

где T_{ki} – продолжительность выполнения i-й работы в календарных днях;

T_{pi} – продолжительность выполнения i-й работы в рабочих днях;

$k_{\text{кал}}$– коэффициент календарности.

Коэффициент календарности определяется по следующей формуле:

$$k_{\text{кал}} = \frac{T_{\text{кал}}}{T_{\text{кал}} - (T_{\text{вых}} + T_{\text{пр}})}\,, \qquad (4.2)$$

где $T_{\text{кал}}$ – количество календарных дней в году;

$T_{\text{вых}}$ – количество выходных дней в году;

$T_{\text{пр}}$ – количество праздничных дней в году.

Расчет коэффициента календарности:

$$\mathrm{k_{кал}} = \frac{\mathrm{T_{кал}}}{\mathrm{T_{кал} - (T_{вых} + T_{пр})}} = \frac{365}{365 - 118} = 1{,}48$$

Таблица 4.7 – Временные показатели проведения научного исследования

Название работы	Трудоёмкость работ									Исполнители	Длительность работ в рабочих днях T_{pi}			Длительность работ в календарных днях T_{ki}		
	T_{min}, чел–дни			T_{max}, чел–дни			$T_{ожi}$, чел– дни									
	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3		Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3
Выбор темы ВКР	1	1	1	2	2	2	1,4	1,4	1,4	Студент, научный руководитель	1	1	1	2	2	2
Составление и утверждение плана работ	1	1	1	2	2	2	1,4	1,4	1,4	Научный руководитель	2	2	2	3	3	3
Разработка общей архитектуры торговых роботов	2	2	3	4	4	5	2,8	2,8	3,8	Студент	3	3	4	5	5	6
Подбор стека технологий для реализации проекта	1	1	1	3	2	2	1,8	1,4	1,4	Студент, научный руководитель	1	1	1	2	2	2
Календарное планирование работ	1	1	1	3	4	4	1,8	2,2	2,2	Студент, научный руководитель	1	2	2	2	3	3
Изучение стека технологий для реализации проекта	3	7	10	7	11	15	4,6	8,6	12	Студент	5	9	12	8	14	18

Разработка робота как серверного приложения	14	17	21	21	26	30	16,8	20,6	24,6	Студент	17	21	25	26	32	37
Интеграция робота и панели управления	3	4	4	7	9	10	4,6	6	6,4	Студент	5	6	7	8	9	11
Оценка эффективности полученных результатов	2	2	3	4	4	5	2,8	2,8	3,8	Студент, научный руководитель	2	2	2	3	3	3
Написание раздела «Финансовый менеджмент»	4	4	4	6	6	6	4,8	4,8	4,8	Студент	5	5	5	8	8	8
Написание раздела «Социальная ответственность»	1	1	1	2	2	2	1,4	1,4	1,4	Студент	2	2	2	3	3	3
Оформление ВКР	5	4	6	7	7	8	5,8	5,2	6,8	Студент	6	6	7	9	9	11

Составлен план научного исследования, в котором разработан календарный план выполнения работ. Для построения таблицы временных показателей проведения НИ был рассчитан коэффициент календарности. С помощью показателей в табл. 4.7 был разработан календарный план-график проведения НИ по теме «Торговый робот с возможностью интеграции в различные биржевые информационные системы», представленный на рисунке 4.1. Для иллюстрации календарного плана была использована диаграмма Ганта, указывающая на целесообразность проведения данного исследования.

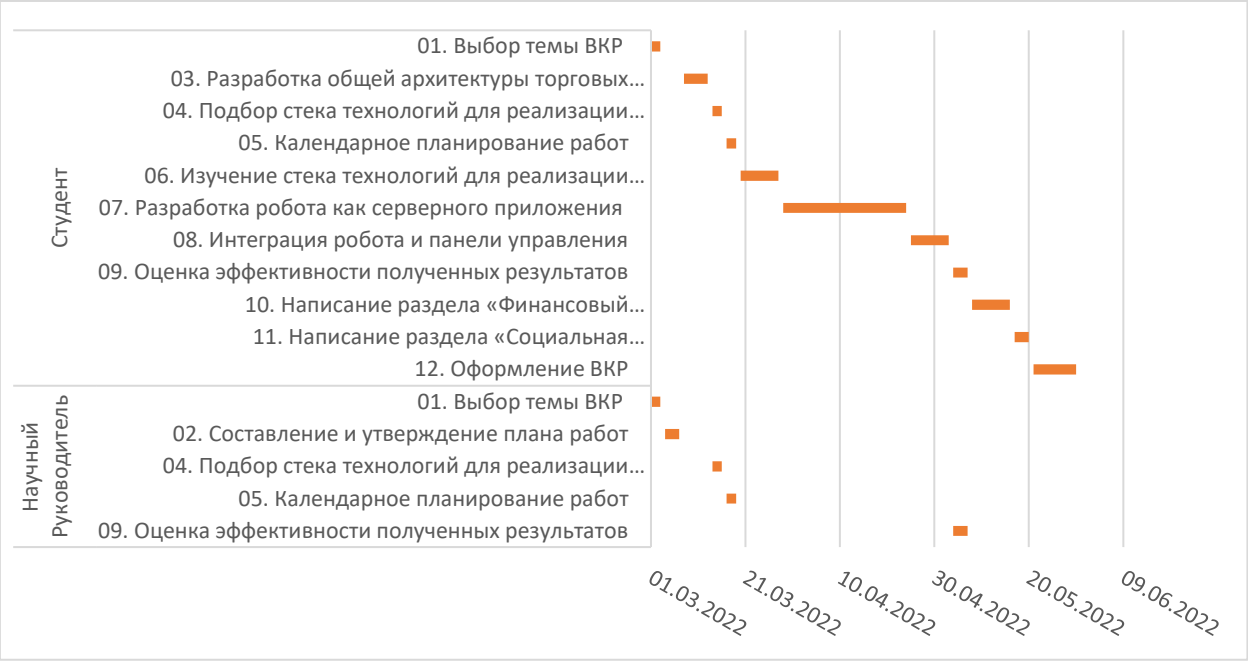

Рисунок 4.1 Календарный план-график проведения научного исследования

4.5 Бюджет научно-технического исследования (НТИ)

1. Материальные затраты.
2. Затраты на спец.оборудование
3. Основная и дополнительная ЗП.
4. Социальные отчисления.
4. Прямые затраты.
5. Накладные расходы.

4.5.1 Расчет материальных затрат НТИ

При планировании бюджета научно-техническое исследование должно быть обеспечено полное и достоверное отражение всех видов расходов, связанных с его выполнением.

В последующих расчетах будут фигурировать 3 возможных исполнения:

1. Робот и панель управления разрабатываются полностью на языке TypeScript в экосистеме Node.js. Для реализации интерфейса используется фреймворк Vue.js.
2. Робот и панель управления разрабатываются полностью на языке C# в экосистеме dotNET. Для реализации интерфейса используется фреймворк Blazor.
3. Роботы разрабатываются на языка C# в экосистеме dotNET, а панель управления – на языке TypeScript в экосистеме Node.js с использованием фреймворка Vue.js.

Таблица 4.8 – Материальные затраты

Наименование	Единица измерения	Количество			Цена за ед., руб.	Затраты на материалы, (Зм), руб.		
		Исп.1	Исп.2	Исп.3		Исп.1	Исп.2	Исп.3
Электроэнергия	кВт*ч	49,920	60,320	70,720	3,50₽	174,72₽	211,12₽	247,52₽
Итого, руб.						174,72₽	211,12₽	247,52₽

Общие материальные затраты составили 175 руб.

4.5.2 Расчет затрат на специальное оборудование для научных работ

В данную статью включают все затраты, связанные с приобретением специального оборудования (персональный компьютер) и программного обеспечения, необходимого для проведения работ по конкретной теме. Определение стоимости производится по действующим прейскурантам. Расчет затрат по данной статье представлен в таблице 4.9.

Таблица 4.9 – Расчет бюджета затрат на приобретение спецоборудования для научных работ

Наименование	Единица измерения	Количество			Цена за ед., руб.	Затраты на материалы, ($З_М$), руб.		
		Исп. 1	Исп. 2	Исп. 3		Исп. 1	Исп. 2	Исп. 3
Персональный компьютер	Шт.	1	1	1	60 000₽	60 000₽	60 000₽	60 000₽
Программное обеспечение (WebStorm) **студенческая лицензия*	Шт.	1	1	1	-₽	-₽	-₽	-₽
Итого:						60 000₽	60 000₽	60 000₽

4.5.3 Основная заработная плата исполнителя темы

В настоящую статью включается основная заработная плата научных работников и программистов. Величина расходов по заработной плате определяется исходя из трудоемкости выполняемых работ и действующей системы окладов и тарифных ставок. В состав основной заработной платы включается премия, выплачиваемая ежемесячно из фонда заработной платы в размере 30 % от тарифа или оклада. Расчет основной заработной платы приводится в таблице 4.10.

Таблица 4.10 – Расчет основной заработной платы

№ п/п	Наименование этапов	Исполнители по категориям	Трудоемкость, чел.-дн.			Зар. плата на 1 чел-дн.			Всего заработная плата по тарифу (окладам), тыс. руб.		
			Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3
1	Выбор темы ВКР	Студент, научный руководитель	1	1	1	6,8			6,8	6,8	6,8
2	Составление и утверждение плана работ	Научный руководитель	2	2	2	4,4			8,8	8,8	8,8
3	Разработка общей архитектуры торговых роботов	Студент	3	3	4	2,4			7,2	7,2	9,6

4	Подбор стека технологий для реализации проекта	Студент, научный руководитель	1	1	1	6,8	6,8	6,8	6,8
5	Календарное планирование работ	Студент, научный руководитель	1	2	2	6,8	6,8	13,6	13,6
6.	Изучение стека технологий для реализации проекта	Студент	5	9	12	2,4	12	21,6	28,8
7.	Разработка робота как серверного приложения	Студент	17	21	25	2,4	40,8	50,4	60
8.	Интеграция робота и панели управления	Студент	5	6	7	2,4	12	14,4	16,8
9.	Оценка эффективности полученных результатов	Студент, научный руководитель	2	2	2	6,8	13,6	13,6	13,6
10.	Написание раздела «Финансовый менеджмент»	Студент	5	5	5	2,4	12	12	12
11	Написание раздела «Социальная ответственность»	Студент	2	2	2	2,4	4,8	4,8	4,8
12	Выбор темы ВКР	Студент	6	6	7	2,4	14,4	14,4	16,8
Итого							146	174,4	198,4

Таблица 4.11 – Баланс рабочего времени

Показатели рабочего времени	Руководитель	Студент
Календарное число дней	365	365
Количество нерабочих дней - выходные дни - праздничные дни	118	118
Потери рабочего времени - отпуск - невыходы по болезни	48 0	72 0
Действительный годовой фонд рабочего времени	199	175

где $З_{тс}$ – заработная плата по тарифной ставке, руб.;

$k_{пр}$ – премиальный коэффициент, равный 0,3 (т.е. 30 процентов от $З_{тс}$);

$k_д$ – коэффициент доплат и надбавок составляет примерно 0,2 – 0,5;

$k_р$ – районный коэффициент, равный 1,3 (для Томска).

Расчет основной заработной платы представлен в таблице 4.13

Таблица 4.12 – Расчет основной заработной платы

Исполнители	Разряд	$З_{тс}$, руб.	$k_{пр}$	$k_д$	$k_р$	$З_м$, руб.	$З_{дн}$, руб.	$Т_р$, раб. дн.	$З_{осн}$, руб.
Научный руководитель	Доцент	35 000₽	0,3	0,4	1,3	77 350₽	4 353₽	7	30 474₽
Студент	Программист	20 000₽	0,3	0,2	1,3	39 000₽	2 318₽	48	111 250₽
Итого									141 724₽

4.5.4 Дополнительная заработная плата и отчисления во внебюджетные фонды

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органам государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников.

В соответствии с Федеральным законом от 24.07.2009 №212-ФЗ установлен размер страховых взносов равный 30%.

Отчисления во внебюджетные фонды представлены в таблице 4.13.

Таблица 4.13 – Отчисления во внебюджетные фонды

Исполнитель	Основная заработная плата, руб.			Дополнительная заработная плата, руб.		
	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3
Руководитель проекта	30 474₽	34 827₽	34 827₽	4 571₽	5 224₽	5 224₽
Студент	111 250₽	134 427₽	157 605₽	16 688₽	20 164₽	23 641₽
Коэффициент отчислений во внебюджетные фонды	0,3					
Итого						
Исполнение 1	48 895₽					
Исполнение 2	58 393₽					

Исполнение 3	66 389₽

4.5.5 Накладные расходы

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов.

Величину коэффициента накладных расходов взят в размере 16%.

Таблица 4.14 – Расчет накладных расходов

Накладные расходы	Сумма, руб.		
	Исп.1	Исп.2	Исп.3
1. Материальные затраты НТИ	28₽	34₽	40₽
2. Затраты на специальное оборудование для научных (экспериментальных) работ	9 600₽	9 600₽	9 600₽
3. Затраты по основной заработной плате исполнителей темы	23 360₽	27 904₽	31 744₽
4. Затраты по дополнительной заработной плате исполнителей темы	3 401₽	4 576₽	5 115₽
5. Отчисления во внебюджетные фонды	7 823₽	9 343₽	10 622₽
Итого	44 212₽	51 457₽	57 121₽

4.5.6 Формирование бюджета затрат научно-исследовательского проекта

Рассчитанная величина затрат научно–исследовательской работы является основой для формирования бюджета затрат проекта. Определение бюджета затрат на научно–исследовательский проект приведено в таблице 4.15.

Таблица 4.15 –Расчет бюджета затрат НТИ

Наименование статьи	Сумма, руб.			Примечание
	Исп.1	Исп.2	Исп.3	
1. Материальные затраты НТИ	175₽	211₽	248₽	Пункт 4.5.1
2. Затраты на специальное оборудование для научных (экспериментальных) работ	60 000₽	60 000₽	60 000₽	Пункт 4.5.2
3. Затраты по основной заработной плате исполнителей темы	146 000₽	174 400₽	198 400₽	Пункт 4.5.3
4. Затраты по дополнительной заработной плате исполнителей темы	21 259₽	28 602₽	31 968₽	Пункт 4.5.4
5. Отчисления во внебюджетные фонды	48 895₽	58 393₽	66 389₽	Пункт 4.5.5
6. Затраты на научные и производственные командировки	-₽	-₽	-₽	Отсутствуют
7. Контрагентские расходы	-₽	-₽	-₽	Отсутствуют
8. Накладные расходы	44 212₽	51 457₽	57 121₽	Пункт 4.5.6
9. Бюджет затрат НТИ	320 541₽	373 063₽	414 125₽	

4.6 Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования

Определение эффективности происходит на основе расчета интегрального показателя эффективности научного исследования. Его нахождение связано с определением двух средневзвешенных величин: финансовой эффективности и ресурсоэффективности.

Таблица 4.16 – Сравнительная оценка характеристик вариантов исполнения проекта

Объект исследования / Критерии	Весовой коэффициент параметра	Исп.1	Исп.2	Исп.3
1. Сложность развертывания	0,1	8	10	6
2. Удобство в эксплуатации (соответствует требованиям потребителей)	0,3	9	7	6
3. Удобство интеграции в новые биржи	0,15	10	4	8
4. Защита от последствий внезапных отключений	0,15	8	8	8

5. Безопасность данных	0,2	7	8	7
6. Совместимость библиотек в панели управления и роботе	0,1	10	8	3
Итого	1	8,6	7,3	6,5

Сравнение интегрального показателя эффективности вариантов исполнения разработки позволит определить сравнительную эффективность проекта и выбрать наиболее целесообразный вариант из предложенных.

Таблица 4.17 – Сравнительная эффективность разработки

№	Показатели	Исп.1	Исп.2	Исп.3
1	Интегральный финансовый показатель разработки	0,774	0,901	1,000
2	Интегральный показатель ресурсоэффективности разработки	8,6	7,3	6,5
3	Интегральный показатель эффективности	11,111	8,103	6,500
4	Сравнительная эффективность вариантов исполнения	1,000	0,729	0,585

Сравнив значения интегральных показателей эффективности, можно сделать вывод, что реализация проекта в первом исполнении является более эффективным вариантом решения задачи, поставленной в данной работе с позиции финансовой и ресурсной эффективности.

ГЛАВА 5 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Введение

Целью выпускной квалификационной работы является разработка серверного приложения - торгового робота, способного осуществлять автоматическую отправку множества заявок для осуществления операций на бирже.

Взаимодействия пользователя с системой будет производиться в помещении размера 5*6 м^2 и рабочей зоной, представляющей собой офис. Оборудованием рабочей зоны является один компьютер, имеющий доступ к целевой информационной системе. Через рассматриваемый объект будет производиться настройка параметров ПО, контроль параметров и исправности ПО дистанционно.

Потенциальными пользователями разработки являются сотрудники компаний, осуществляющих тестирование информационных систем бирж.

Взаимодействия пользователя с системой будет производиться через ПК, следовательно, в разделе будут рассмотрены правовые, организационные и производственные вопросы обеспечения безопасности человека при работе с системой. Будут выявлены и проанализированы потенциальные вредные и опасные факторы, возникающие в ходе разработки и использования конечного решения, а также будут предложены мероприятия по их предотвращения. Будут рассмотрены вопросы негативного влияния на окружающую среду и возникновения потенциальных чрезвычайных ситуаций на рабочем месте при работе с системой.

5.1 Правовые и организационные вопросы обеспечения безопасности

5.1.1 Специальные правовые нормы трудового законодательства

Регулирование взаимоотношений работника и работодателя осуществляется согласно Трудовому кодексу РФ – устанавливается режим рабочего времени и время отдыха работника, оплата, выходные и прочее.

Рабочее время работника не должна превышать 40 часов в неделю. Для работников в возрасте до 16 лет рабочее время составляет не более 24 часов в неделю; в возрасте от 16 лет до 18 лет и инвалидов I и II группы – не более 35 часов в неделю.

В течение смены работнику должен выделяться перерыв для отдыха и питания продолжительностью не менее 30 минут и не более двух часов.

Работодатель обязан предоставлять работнику ежегодный основной оплачиваемый отпуск длительностью в 28 календарных дней.

5.1.2 Организационные мероприятия при компоновке рабочей зоны

Конструкцией рабочего места должно быть обеспечено выполнение работы на ПК в пределах зоны легкой досягаемости и оптимальной зоны моторного поля из-за «частого» и «очень частого» взаимодействия с ПК.

Рабочее место пользователя ПК в нашем случае не подразумевает регулировку высоты рабочей поверхности и подставки для ног. Согласно ГОСТ 12.2.032-78 «Рабочее место при выполнении работ сидя» высота рабочей поверхности при организации рабочего места для работ с ПК должна быть равна 655 мм для мужчин и женщин.

Согласно ГОСТ 21889-76 Система «Человек-машина», кресло человека-оператора Кресло пользователя ПК должно создавать условия для поддержания корпуса человека в физиологически рациональном положении и не затруднять рабочих движений; должна регулироваться высота поверхности сиденья и угол наклона спинки.

5.2 Производственная безопасность

Для обеспечения производственной безопасности работника в ходе разработки и эксплуатации системы необходимо выявить и проанализировать возможные вредные и опасные факторы и их воздействие на организм человека, привести допустимые нормы, предложить средства индивидуальной и коллективной защиты для минимизации воздействия фактора.

Вредным фактором считается такое негативное воздействие на организм человека, приводящее к ухудшению самочувствия или развитию заболевания. Вредный фактор, оказывающий длительное и умеренное негативное воздействие, становится опасным.

Опасным фактором считается такое негативное воздействие на организм человека, приводящее к травмам или резкому ухудшению здоровья.

В ходе разработки и при эксплуатации системы были выявлены возможные вредные и опасные факторы (таблица 5.1).

Таблица 5.1– Возможные опасные и вредные производственные факторы при работе за ПК

Факторы (ГОСТ 12.0.003-2015)	Нормативные документы
Умственное перенапряжение, в том числе вызванное информационной нагрузкой	Р 2.2.2006–05. «Руководство, по гигиенической оценке, факторов рабочей среды и трудового процесса. Критерии и классификация условий труда.»
Отсутствие или недостаток необходимого естественного освещения	СП 52.13330.2016 «Естественное и искусственное освещение»
Производственные факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает работающий	ГОСТ 12.1.038-82 ССБТ. «Электробезопасность. Предельно допустимые уровни напряжений прикосновения и токов»

5.2.1 Умственное перенапряжение, в том числе вызванное информационной нагрузкой

В стандарте Р 2.2.2006–05. «Руководство, по гигиенической оценке, факторов рабочей среды и трудового процесса. Критерии и классификация условий труда.» интеллектуальные нагрузки классифицированы на оптимальные, допустимые и вредные (таблица 5.2).

Таблица 5.2 – Классы условий труда по показателям напряженности трудового процесса для интеллектуальных нагрузок

Показатели напряженности трудового процесса	Класс условий труда			
	Оптимальный	Допустимый	Вредный	
	Напряженность труда легкой степени	Напряженность труда средней степени	Напряженный труд	
			1 степени	2 степени
	1	2	3.1	3.2
Содержание работы	Отсутствует необходимость принятия решения	Решение простых задач по инструкции	Решение сложных задач с выбором по известным алгоритмам (работа по серии инструкций)	Эвристическая (творческая) деятельность, требующая решения алгоритма, единоличное

				руководство в сложных ситуациях
Характер выполняемой работы	Работа по индивидуальному плану	Работа по установленному графику с возможной его коррекцией по ходу деятельности	Работа в условиях дефицита времени	Работа в условиях дефицита времени и информации с повышенной ответственностью за конечный результат

5.2.2 Отсутствие или недостаток необходимого естественного освещения

Недостаточная освещенность рабочей зоны также является одним из важнейших потенциально вредных и опасных факторов. Работа при недостаточном освещении приводит к появлению усталости глаз, головным болям и переутомлению, снижается производительность труда, а при продолжительном воздействии может привести к снижению зрительной работоспособности.

Согласно СП 52.13330.2016 «Естественное и искусственное освещение», характеристику зрительной работы за ПК можно отнести к различению объектов при фиксированной и нефиксированной линии зрения средней точности, при которой наименьший или эквивалентный размер объекта различения (буквы и символы на дисплее ПК) составляет более 0,5 мм. Требования к искусственному и естественному освещению представлены в таблице 5.3.

Таблица 5.3 – Требования к естественному освещению помещений

Характеристика зрительной работы	Наименьший или эквивалентный размер объекта различения, мм	Разряд зрительной работы	Подразряд зрительной работы	Относительная продолжительность зрительной работы при направлении зрения на рабочую поверхность, %	Естественное освещение	
					КЕО $e_н$, %, при	
					верхнем или комбинированном	боковом
Различение объектов при	Более 0,5	В	1	Не менее 70	2,0	0,5

фиксированной и нефиксированной линии зрения: - средней точности			2	Менее 70	2,0	0,5

5.2.3 Производственные факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает работающий

При взаимодействии человека с ПК следует учесть вероятность поражения электрическим током, контакт с которым может привести к электротравме, а в тяжелых случаях – к гибели человека. Поражение электрическим током может произойти вследствие прикосновения к открытым токоведущим частям, находящимся под напряжением, из-за плохой изоляции токоведущих частей компьютера, при работе за ПК влажными руками. Поэтому особенно важно обеспечить пользователя ПК электробезопасностью.

Согласно ГОСТ 12.1.038-82 ССБТ. «Электробезопасность. Предельно допустимые уровни напряжений прикосновения и токов» значения напряжение прикосновения и токи при работе с ПК должны быть не выше значений, указанных в таблице 5.4.

Таблица 5.4 – Предельно допустимые напряжения прикосновения и токи

Род тока	U, В	I, мА
	не более	
Переменный, 50 Гц	2,0	0,3
Переменный, 400 Гц	3,0	0,4
Постоянный	8,0	1,0

Для обеспечения пользователя ПК электробезопасностью необходимо установить дополнительные оградительные устройства, обеспечивающие недоступность токоведущих частей для прикосновения, обеспечить защитное заземления или зануления (защитного отключения) электрооборудования.

Перед работой с ПК необходимо убедиться в целостности вилки и провода электропитания, в отсутствии видимых повреждений аппаратуры. При

работе с ПК запрещается прикасаться к задней панели системного блока и переключать разъемы периферийных устройств работающего устройства.

5.3 Экологическая безопасность

Использование системы конечным пользователем также необходимо рассмотреть с точки зрения экологической безопасности.

Во время использования системы загрязнение гидросферы не происходит, так как отсутствуют прямые выбросы. Однако стоит отметить косвенные выбросы парниковых газов за счет потребления электроэнергии, а также негативное воздействие использования системы на литосферу за счет утилизации отходов электрооборудования по причине поломок или из-за несоответствия производственным требованиям по причине технологического устаревания.

5.3.1 Утилизация отходов электрооборудования

В соответствие с приказом Минприроды России от 30.09.2011 N 792 «Об утверждении Порядка ведения государственного кадастра отходов» и федеральным классификационным каталогом отходов (ФККО) компьютеры и периферийное оборудование, использовавшиеся при разработке и эксплуатации системы и утратившие потребительские свойства, относятся к IV-ому классу опасности, ртутные и люминесцентные лампы, использовавшиеся для создания искусственного освещения, относятся к I-ому классу опасности, использованная бумага и канцелярия относятся к IV-ому и V-ому классам опасности. Отходы должны быть пройти этапы технологического цикла отходов, подлежащих ликвидации, и утилизированы в соответствие со своим классом опасности согласно ГОСТ Р 53692-2009 «Ресурсосбережение. Обращение с отходами. Этапы технологического цикла отходов».

5.3.2 Углеродный след от использования электроэнергии

Согласно ГОСТ Р ИСО 14067-2021 «Газы парниковые. Углеродный след продукции. Требования и руководящие указания по количественному

определению» выбросы парниковых газов от использования электроэнергии должны рассчитываться поставщиком электроэнергии при наличии выделенной сети электропередач. При отсутствии информации относительно конкретного поставщика электроэнергии следует использовать выбросы парниковых газов, связанные с соответствующими электросетями, поставляющими электроэнергию. Соответствующая сеть должна определять потребление электроэнергии в соответствующем регионе за исключением любой другой ранее заявленной электроэнергии. Если система учета электроэнергии отсутствует, то выбранная сеть будет определять потребление электроэнергии в определенном регионе.

Компьютер с необходимыми для запуска роботов характеристиками потребляет около 150 Ватт, что является относительно маленьким числом. При этом персональный компьютер для управления роботами потребляет около 100 Ватт. Если ещё учесть тот факт, что роботы спроектированы так, чтобы их можно было запускать на одном компьютере во множественных экземплярах, можно сказать, что были предусмотрены все возможные меры для уменьшения углеродного следа.

5.4 Безопасность в чрезвычайных ситуациях

При разработке и эксплуатации системы возможно возникновение следующих чрезвычайных ситуаций различного происхождения:

- природного (землетрясение, бури);
- техногенного (пожары, взрывы, внезапное обрушение зданий, аварии на коммунальных системах жизнеобеспечения, аварии на электростанциях);
- биолого-социального (эпидемии).

Наиболее вероятной чрезвычайной ситуацией на рабочем месте в помещении, оборудованном электронно-вычислительными машинами, является возникновение пожара. Это может происходить по причине близкого расположения элементов электронных систем ПК относительно друг друга – из-за нагревания током изоляция проводов может оплавиться и оголить провода,

что приведет к короткому замыканию и искрению. Согласно Федеральному закона от 22.07.2008 N 123-ФЗ (ред. от 30.04.2021) "Технический регламент о требованиях пожарной безопасности", возможный пожар принадлежит к классу Е - пожары горючих веществ и материалов электроустановок, находящихся под напряжением. Неисправность аппаратуры, неправильное использование электрооборудования, незнание мер безопасности и их пренебрежение также могут привести к возникновению пожара.

Для обеспечения пожарной безопасности необходимо устранить потенциальные причины возникновения пожара в электрооборудованиях – это предупреждение замыкания грамотными выбором, монтажом и эксплуатацией сетей, работа только с исправным оборудованием и электропроводкой. Не следует допускать накопления мусора, ненужных бумаг и прочих вещей, не используемых в работе.

Помещение должно быть оснащено рабочими порошковыми или углекислотными огнетушителями. Недопустимо применение в качестве средств пожаротушения электроприборов воды или пены из-за опасности поражения электрическим током. Необходимо обеспечить возможность беспрепятственного движения людей по эвакуационным путям. Перед работой необходимо провести для персонала инструктаж по технике безопасности.

В случае возникновения пожара необходимо обратиться в пожарную службу. Если воспламенение небольшое и его по силам потушить имеющимися средствами пожаротушения, то это необходимо сделать. В ином случае следует приступить к эвакуации из здания в соответствии с планом эвакуации при пожарах и других ЧС.

Выводы

В ходе написания раздела были рассмотрены правовые, организационные и производственные вопросы обеспечения безопасности человека при работе с системой. Были проанализированы вредные и опасные факторы, предложены мероприятия по их предотвращению. Также были рассмотрены вопросы

негативного влияния на окружающую среду, были предложены меры по обеспечению экологической безопасности. Было проанализировано возможное возникновение ЧС на рабочем месте, предложены рекомендации по профилактике возникновения наиболее вероятной ЧС – пожара.

Таблица 5.5 – Соответствие фактических значений потенциально возможных факторов нормативным значениям

Производственный фактор	Показатель	Реальное значение	Нормативное значение
Умственное перенапряжение, в том числе вызванное информационной нагрузкой	Класс условий труда	3.1 – напряженный труд 1 степени	2 - допустимый
Отсутствие или недостаток необходимого естественного освещения	КЕО $e_н$, %, при боковом освещении	0,6	От 0,5
Производственные факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает работающий	Напряжение (U) и сила тока (I) при прикосновении	При целости оборудования удары током невозможны	U не более 2 В, I не более 0,3 мА

- Согласно «Правилам устройства электроустановок» рабочее помещение является нормальным, без повышенной опасности.
- Согласно «Правилам по охране труда при эксплуатации электроустановок» персонал относится к I группе по электробезопасности.
- Согласно СанПиН 1.2.3685-21 "Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания", тяжесть труда принадлежит к категории I.
- Согласно СП 12.13130.2009 «Определение категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности», взрывопожарная и пожарная опасность к категории В4 – пожароопасность.
- Согласно критериям отнесения объектов, оказывающих негативное воздействие на окружающую среду, к объектам I, II, III и IV категорий, объект относится к IV категории.

Заключение

В ходе работы над ВКР был создан прототип торгового робота, который можно интегрировать в торговые системы через API интерфейсы и позволяющий реализовывать шаблоны поведения пользователей через вызов алгоритмов для создания различных ситуаций на бирже. Робот реализует функционал отправки торговых поручений, получения и хранения различной мета информации (например, операции со счетом или информация о ценных бумагах). Также робот предоставляет интерфейсы для управления из другого приложения, что позволит создать панель управления, оперирующую со множеством роботов.

При разработке корректность реализованных функций проверялась через специальный сценарий, вызывающий методы приложения и выводящий результаты в консоль. Качество исполнения программных интерфейсов робота было опробовано через клиент Postman. Все результаты работы соответствуют ожиданиям, ошибок не выявлено.

Одна запущенная копия робота не окажет значительного влияния на систему. Однако, множество роботов с одновременно запущенным алгоритмом способны выявить проблемы системы. При централизованном управлении роботы уже смогут создавать интересные торговые ситуации и серьезные нагрузки. Postman может быть использован только для управления одной копией робота, что противоречит идее проекта. Поэтому рекомендуется использовать специальные инструменты, такие как панель управления, созданная Надеждой Хромовой в рамках ее выпускной квалификационной работы.

Роботов рекомендуется запускать, используя инструменты для автоматического запуска множества копий программ (например, docker-compose или Kubernetes). У каждой копии необходимо только сменить настройки, отвечающие за связь с определенным аккаунтом торговой системы. Таким образом, каждый робот будет представлять собой симулятор отдельного пользователя с отдельным счетом.

На данном этапе роботы пригодны для апробации в условиях реальных производственных задач. Для этого только необходимо получить доступ к API интерфейсам торговой системы и реализовать собственные настройки робота.

Перечень информационных источников

1. About | Node.js [Электронный ресурс] // Режим доступа: https://nodejs.org/en/about/
2. ASP.NET | Open-source web framework for .NET [Электронный ресурс] // Режим доступа: https://dotnet.microsoft.com/en-us/apps/aspnet
3. Documentation | NestJS - A progressive Node.js framework [Электронный ресурс] // Режим доступа: https://docs.nestjs.com/
4. Express - Node.js web application framework [Электронный ресурс] // Режим доступа: https://expressjs.com/
5. Fastify, Fast and low overhead web framework, for Node.js [Электронный ресурс] // Режим доступа: https://www.fastify.io/
6. Introduction | Socket.IO [Электронный ресурс] // Режим доступа: https://socket.io/docs/v4/
7. Prisma Documentation | Concepts, Guides, and Reference [Электронный ресурс] // Режим доступа: https://www.prisma.io/docs/
8. QUIK — ARQA Technologies [Электронный ресурс] // Режим доступа: https://arqatech.com/ru/products/quik/
9. Reference test harness for algorithmic trading platforms | EXACTPRO [Электронный ресурс] // Режим доступа: https://exactpro.com/ideas/research-papers/reference-test-harness-algorithmic-trading-platforms
10. Sequelize | Feature-rich ORM for modern TypeScript & JavaScript [Электронный ресурс] // Режим доступа: https://sequelize.org/
11. SQLite Home Page [Электронный ресурс] // Режим доступа: https://www.sqlite.org/index.html

12. th2 - Test Automation Framework for Financial Markets | EXACTPRO [Электронный ресурс] // Режим доступа: https://exactpro.com/test-tools/th2
13. TypeORM - Amazing ORM for TypeScript and JavaScript (ES7, ES6, ES5). Supports MySQL, PostgreSQL, MariaDB, SQLite, MS SQL Server, Oracle, WebSQL databases. Works in NodeJS, Browser, Ionic, Cordova and Electron platforms. [Электронный ресурс] // Режим доступа: https://typeorm.io/
14. TypeScript: JavaScript With Syntax For Types. [Электронный ресурс] // Режим доступа: https://www.typescriptlang.org/
15. What are microservices? [Электронный ресурс] // Режим доступа: https://microservices.io/
16. День, который вошел в историю. Прошло 10 лет после Flash Crash [Электронный ресурс] // Режим доступа: https://bcs-express.ru/novosti-i-analitika/den-kotoryi-voshel-v-istoriiu-proshlo-10-let-posle-flash-crash
17. Первые итоги IPO Facebook: страхи, надежды и большой скандал | Forbes.ru [Электронный ресурс] // Режим доступа: https://www.forbes.ru/tehno/82475-pervye-dni-posle-ipo-facebook-strahi-i-nadezhdy