

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

*Факультет*  
вычислительной техники

*Кафедра*  
МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

Направленность Программная инженерия

**БАКАЛАВРСКАЯ РАБОТА**

на тему

**Бизнес-проект «Клиентская часть  
информационной платформы трудоустройства  
и стажировки студентов в ИТ-компаниях»**

Студент \_\_\_\_\_ Поршнев Павел Александрович  
(подпись, дата) (ФИО полностью)

Руководитель \_\_\_\_\_ Д.Т.Н., профессор Козлов А.Ю.  
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр \_\_\_\_\_ Такташкин Д.В.  
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от \_\_\_\_\_ № \_\_\_\_\_)

Заведующий кафедрой \_\_\_\_\_ Козлов А.Ю.  
(подпись) (фамилия, инициалы)

Работа защищена с отметкой \_\_\_\_\_ (протокол заседания ГЭК от \_\_\_\_\_ № \_\_\_\_\_)

Секретарь ГЭК \_\_\_\_\_ Попова Н.А.  
(подпись) (фамилия, инициалы)

Пенза, 2026

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет  
Вычислительной техники

Кафедра  
МО и ПЭВМ

«Утверждаю»  
Заведующий кафедрой  
А.Ю. Козлов  
«26» декабря 2025 г

**ЗАДАНИЕ  
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ  
БАКАЛАВРА**

1. Студент Поршнев Павел Александрович гр. 22ВП1 факультета ВТ

<Фамилия, имя, отчество полностью>

2. Тема работы Бизнес-проект «Клиентская часть информационной платформы трудоустройства и стажировки студентов в ИТ-компаниях»

Тема утверждена приказом ПГУ №1583/о от "30" декабря 2025 г.

3. Руководитель работы д.т.н., профессор, заведующий кафедры «МО и ПЭВМ» Козлов А.Ю.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) Назначение: клиентская часть информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях для взаимодействия студентов, работодателей и преподавателей

2) Среда выполнения: операционная система Windows 10

Языковые средства разработки: JavaScript, JSX, HTML, CSS

Инструментальные средства: VS Code, Webstorm, Visual Paradigm

Технологии разработки: React, Vite, Vitest, Docker

3) Основные функции приложения:

а) регистрация и авторизация пользователей;

б) выбор роли пользователя при входе в систему;

в) просмотр и редактирование пользовательского профиля;

г) создание, редактирование и удаление вакансий работодателем;

д) просмотр, поиск и фильтрация вакансий студентом;

е) создание, редактирование, копирование и удаление резюме студентом;

ж) формирование рекомендаций вакансий для студента на основе резюме;

з) формирование рекомендаций резюме для работодателя на основе вакансии;

- и) отправка откликов студентов на вакансии;*
- к) просмотр и изменение статусов откликов работодателем;*
- л) просмотр студентом истории своих откликов;*
- м) управление учебными группами преподавателем;*
- н) обмен сообщениями между студентами и работодателями в режиме реального времени;*
- о) выход пользователя из системы.*

#### 5. Перечень подлежащих разработке вопросов

- 1) Анализ предметной области и требования к системе;*
- 2) Проектирование программных средств;*
- 3) Реализация программных средств;*
- 4) Тестирование программных средств;*
- 5) Оформление пояснительной записки;*

#### 6. Календарный график выполнения работы

| №<br>п/п | Наименование этапов работы                       | Объем<br>работы | Срок<br>выполнения         | Подпись<br>руководителя |
|----------|--------------------------------------------------|-----------------|----------------------------|-------------------------|
| 1        | Анализ предметной области и требований к системе | 10%             | 07.04.2026 –<br>13.04.2026 |                         |
| 2        | Проектирование программных средств               | 20%             | 14.04.2026 –<br>25.04.2026 |                         |
| 3        | Реализация программных средств                   | 30%             | 27.04.2026 –<br>11.05.2026 |                         |
| 4        | Тестирование программных средств                 | 20%             | 12.05.2026 –<br>21.05.2026 |                         |
| 5        | Оформление пояснительной записки                 | 20%             | 22.05.2026 –<br>02.06.2026 |                         |

Дата выдачи задания «26» декабря 2025г.

Руководитель бакалаврской работы \_\_\_\_\_  
(подпись, дата)

Задание к исполнению принял студент \_\_\_\_\_  
(подпись, дата)

Работу к защите допустить  
Декан ФВТ \_\_\_\_\_ Л.Р. Фионова  
(подпись, дата)

## Реферат

Пояснительная записка содержит 118 листов, 32 рисунка, 14 таблиц, 21 использованный источник и 3 приложения.

**ИНФОРМАЦИОННАЯ ПЛАТФОРМА, WEB-ПРИЛОЖЕНИЕ, КЛИЕНТСКАЯ ЧАСТЬ, REACTJS, JAVASCRIPT, JSX, HTML, CSS, REST API, SIGNALR, ROUTING, UI, UX, VITE, AXIOS, TAILWIND CSS, VITEST, DOCKER, NGINX, GIT.**

Объект исследования – клиентская часть информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях.

Цель выпускной квалификационной работы – разработать клиентскую часть информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях, обеспечивающую взаимодействие студентов, работодателей и преподавателей через веб-интерфейс.

Технологии разработки – разработка клиентской части приложения проводилась на языке программирования JavaScript с использованием библиотеки ReactJS в среде разработке Webstorm 2025.

Среда исполнения – Windows 10, web-браузер.

Результат работы – в ходе выполнения выпускной квалификационной работы была создана клиентская часть информационной платформы трудоустройства и стажировки студентов в ИТ-компаниях. Все требования к разработке приложения соблюдены.

ПГУ 09.03.04 – 08BP221.12 ПЗ

|           |      |                 |         |      |                                                                                                                 |              |      |        |
|-----------|------|-----------------|---------|------|-----------------------------------------------------------------------------------------------------------------|--------------|------|--------|
|           |      |                 |         |      | ПГУ 09.03.04 – 08BP221.12 ПЗ                                                                                    |              |      |        |
|           |      |                 |         |      |                                                                                                                 |              |      |        |
| Изм.      | Лист | № докум.        | Подпись | Дата | Бизнес-проект «Клиентская часть информационной платформы трудоустройства и стажировки студентов в ИТ-компаниях» | Лит.         | Лист | Листов |
| Разраб.   |      | Поршнев П. А.   |         |      |                                                                                                                 |              |      |        |
| Провер.   |      | Козлов А. Ю.    |         |      |                                                                                                                 |              | 4    | 118    |
| Реценз.   |      |                 |         |      |                                                                                                                 | Группа 22ВП1 |      |        |
| Н. Контр. |      | Такташкин Д. В. |         |      |                                                                                                                 |              |      |        |
| Утверд.   |      | Козлов А. Ю.    |         |      |                                                                                                                 |              |      |        |

## Содержание

|                                                                                                                                 |    |
|---------------------------------------------------------------------------------------------------------------------------------|----|
| Введение.....                                                                                                                   | 7  |
| 1 Стартап-часть.....                                                                                                            | 8  |
| 1.1 Описание существующей проблемы на рынке .....                                                                               | 8  |
| 1.2 Описание решения проблемы и продуктового решения .....                                                                      | 9  |
| 1.3 Объем, структура и развитие рынка .....                                                                                     | 10 |
| 1.4 Анализ конкурентов .....                                                                                                    | 12 |
| 1.5 Бизнес-модель проекта.....                                                                                                  | 14 |
| 1.6 Команда стартап-проекта.....                                                                                                | 15 |
| 1.7 Описание финансовой модели.....                                                                                             | 16 |
| 1.8 Развитие бизнес-проекта .....                                                                                               | 17 |
| 1.9 Описание рисков .....                                                                                                       | 18 |
| 2 Анализ требований и постановка задачи .....                                                                                   | 19 |
| 2.1 Анализ предметной области .....                                                                                             | 19 |
| 2.2 Анализ программных аналогов платформы трудоустройства....                                                                   | 20 |
| 2.2.1 Анализ общих платформ поиска работы.....                                                                                  | 20 |
| 2.2.2 Анализ специализированных платформ работы в ИТ-сфере.....                                                                 | 21 |
| 2.2.3 Анализ университетских информационных систем .....                                                                        | 21 |
| 2.2.4 Результаты анализа.....                                                                                                   | 22 |
| 2.3 Постановка задачи на разработку информационной платформы<br>трудоустройства и стажировок студентов в ИТ-компаниях .....     | 24 |
| 2.4 Анализ требований к разрабатываемой информационной<br>платформе трудоустройства и стажировок студентов в ИТ-компаниях ..... | 27 |
| 2.4.1 Анализ функциональных требований.....                                                                                     | 27 |
| 2.4.2 Анализ нефункциональных требований.....                                                                                   | 33 |
| 2.5 Выбор инструментальных средств.....                                                                                         | 34 |
| 2.6 Планирование проекта и оценка бюджета .....                                                                                 | 36 |
| 3 Проектирование клиентской части информационной платформы<br>трудоустройства и стажировок студентов в ИТ-компаниях .....       | 38 |
| 3.1 Проектирование архитектуры информационной платформы.....                                                                    | 38 |

|     |                                                                                                              |     |
|-----|--------------------------------------------------------------------------------------------------------------|-----|
| 3.2 | Проектирование интерфейса пользователя информационной платформы трудоустройства и стажировок студентов ..... | 41  |
| 3.3 | Проектирование алгоритма регистрации студента и отображения вакансий и резюме .....                          | 49  |
| 3.4 | Проектирование базы данных и способа доступа к ней .....                                                     | 52  |
| 3.5 | Проектирование взаимодействия клиентской части информационной платформы с серверной частью .....             | 55  |
| 4   | Реализация информационной платформы трудоустройства и стажировки студентов.....                              | 58  |
| 4.1 | Реализация клиентской части .....                                                                            | 58  |
| 4.2 | Реализация функции отображения вакансий .....                                                                | 59  |
| 4.3 | Реализация управления профилем компании .....                                                                | 64  |
| 4.4 | Реализация взаимодействия с клиентской частью .....                                                          | 68  |
| 4.5 | Реализация взаимодействия с серверной частью .....                                                           | 72  |
| 5   | Тестирование информационной платформы .....                                                                  | 77  |
| 5.1 | Функциональное тестирование клиентской части .....                                                           | 77  |
| 5.2 | Функциональная диаграмм и таблица решений .....                                                              | 77  |
| 5.3 | Тестирование SignalR-взаимодействия .....                                                                    | 81  |
| 5.4 | Модульное тестирование клиентской части .....                                                                | 82  |
| 5.5 | Оценка качества программного кода .....                                                                      | 84  |
| 5.6 | Метрики Холстеда .....                                                                                       | 85  |
| 5.7 | Метрика Чепина .....                                                                                         | 87  |
|     | Заключение .....                                                                                             | 89  |
|     | Список использованных источников .....                                                                       | 91  |
|     | Приложение А (рекомендуемое) Диаграммы вариантов использования .....                                         | 93  |
|     | Приложение Б (обязательное) Код клиентской части приложения .....                                            | 97  |
|     | Приложение В (рекомендуемое) Тестирование.....                                                               | 102 |

## Введение

Современное развитие информационных технологий способствует активному внедрению web-приложений в образовательную и профессиональную среду. Одной из актуальных задач является автоматизация взаимодействия между студентами, заинтересованными в прохождении стажировки или трудоустройстве, работодателями, размещающими вакансии, и преподавателями, сопровождающими учебные группы. Использование информационной платформы позволяет упростить процесс поиска работы, создания резюме, а также дает возможность удобной коммуникации между студентом и работодателем.

Клиентская часть представляет собой web-приложение для информационной платформы. Приложение обеспечивает удобный пользовательский интерфейс для взаимодействия с платформой. Для каждой роли предусмотрен отдельный набор интерфейсов и функциональных возможностей.

Целью проекта является создание удобной и функциональной клиентской части, обеспечивающей доступ пользователей к основным возможностям программы через web-интерфейс. Для реализации использовались современные технологии frontend-разработки, что обеспечило модульность структуры приложения, взаимодействие с серверной частью по API и корректную обработку пользовательских данных.

## **1     Стартап-часть**

### **1.1   Описание существующей проблемы на рынке**

Современный рынок труда в сфере информационных технологий характеризуется высоким спросом на молодых специалистов, однако процесс перехода студентов от обучения к профессиональной деятельности остаётся недостаточно формализованным. Студенты, заинтересованные в прохождении стажировки или поиске первого рабочего места, часто сталкиваются с разрозненностью информации о вакансиях, отсутствием понятного механизма взаимодействия с работодателями и высокой конкуренцией со стороны кандидатов, уже имеющих практический опыт.

Существующие сервисы поиска работы в основном ориентированы на широкий круг соискателей. Они позволяют размещать резюме, искать вакансии и отправлять отклики, однако не учитывают особенности студенческой аудитории. Начинающих специалистам важно знать не только сведения о вакансии, но и понимание требований работодателя, возможность представить учебные и проектные достижения, получить отклик от компании и выстроить дальнейшую коммуникацию. В массовых сервисах такие кандидаты часто оказываются менее заметными, поскольку их резюме конкурируют с анкетами более опытных специалистов.

Отдельную проблему составляет недостаточная связь между образовательными организациями и работодателями. Университеты заинтересованы в том, чтобы студенты проходили стажировки и практику в профильных компаниях, однако во многих случаях взаимодействие организуется через электронную почту, мессенджеры или отдельные внутренние системы. Такой подход затрудняет централизованное сопровождение студентов, контроль их активности и анализ результатов взаимодействия с компаниями.

Для работодателей также существует проблема поиска начинающих специалистов, подходящих под конкретные требования вакансии или стажировки. Компании вынуждены использовать общие платформы

трудоустройства, социальные сети или прямые обращения в образовательные организации.

Таким образом, на рынке существует потребность в информационной платформе, ориентированной на трудоустройство и стажировки студентов в ИТ-компаниях. Такая платформа должна объединять интересы студентов, работодателей и преподавателей, обеспечивать удобный пользовательский интерфейс, работу с вакансиями и резюме, обработку откликов, поддержку учебных групп и коммуникацию между участниками.

## **1.2 Описание решения проблемы и продуктового решения**

В качестве способа решения выявленной проблемы предлагается создание информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях. Продуктовое решение представляет собой web-приложение, объединяющее в единой цифровой среде студентов, работодателей и преподавателей. Такой подход позволяет заменить разрозненное взаимодействие через отдельные сервисы централизованной платформой, в которой основные действия выполняются через единый пользовательский интерфейс.

Одним из ключевых элементов решения проблемы является клиентская часть платформы. Именно она обеспечивает доступ пользователей к функциональным возможностям системы. Для каждой категории пользователей предусмотрен отдельный набор интерфейсов.

Предлагаемое решение ориентировано не только на поиск вакансий, но и на организацию полного цикла взаимодействия между участниками процесса. В отличие от универсальных сервисов трудоустройства, платформа учитывает специфику студенческой аудитории и образовательной среды. Наличие учебных групп, ролевого доступа и встроенной коммуникации позволяет использовать продукт как инструмент для сопровождения стажировок, практик и первых карьерных шагов студентов.

Продуктовое решение предполагает использование современной клиент-серверной архитектуры. Клиентская часть отвечает за отображение данных,

маршрутизацию, обработку пользовательских действий и взаимодействие с серверной частью через API. Серверная часть отвечает за хранение и обработку данных, а также выполнения основной бизнес-логики приложения. Такой подход позволяет разделить зоны ответственности, упростить развитие системы и обеспечить возможность дальнейшего расширения функциональности.

Разрабатываемая платформа представляет собой специализированное решение для рынка студенческого трудоустройства в ИТ-сфере. Основная ценность проекта заключается в объединении вакансий, резюме, откликов и коммуникации в одном web-приложении. Это позволяет студентам быстрее находить подходящие предложения, работодателям – находить заинтересованных молодых специалистов, а преподавателям – сопровождать процесс взаимодействия студентов с компаниями.

### **1.3 Объем, структура и развитие рынка**

Основной целевой сегмент продукта составляют студенты ИТ-направлений, образовательные организации и компании, заинтересованные в привлечении начинающих специалистов на стажировки, практики и junior-позиции.

Российский рынок HR-tech демонстрирует устойчивый рост. По оценке Smart Ranking, в 2024 году совокупная выручка 80 крупнейших компаний в сфере автоматизации HR-функций составила 99,3 млрд рублей, что на 38% больше, чем годом ранее. При этом подбор персонала остаётся крупнейшим сегментом рынка: его выручка превысила 50 млрд рублей. Это подтверждает высокий спрос на цифровые инструменты, связанные с поиском, подбором и сопровождением кандидатов [1].

Дополнительным фактором развития рынка является сохраняющийся дефицит ИТ-специалистов. По данным отраслевых оценок, в России нехватка ИТ-кадров оценивается в сотни тысяч человек. В публикациях о состоянии ИТ-рынка указывается диапазон дефицита от 500 до 700 тыс. специалистов, а отдельные оценки называют ещё более высокие значения. Такая ситуация

формирует устойчивую потребность компаний в раннем поиске и подготовке молодых кадров, в том числе через стажировки, практики и работу с университетами [2].

Структурно рынок можно разделить на несколько групп решений. К первой группе относятся массовые платформы поиска работы, предназначенные для широкого круга соискателей и работодателей. Ко второй группе относятся специализированные ИТ-платформы, ориентированные на вакансии в сфере информационных технологий. Третью группу составляют внутренние университетские системы, используемые для сопровождения учебного процесса и взаимодействия со студентами. Отдельно можно выделить HR-tech-сервисы, автоматизирующие подбор персонала, аналитику, коммуникации и управление кандидатами.

Разрабатываемая информационная платформа занимает нишу между массовыми сервисами трудоустройства и внутренними университетскими системами. В отличие от универсальных job-сервисов, продукт ориентирован именно на студентов и начинающих ИТ-специалистов. Он включает инструменты взаимодействия с работодателями, работу с вакансиями, резюме, откликами и встроенную коммуникацию между участниками. За счёт этого платформа может рассматриваться как специализированное решение для сегмента студенческого трудоустройства и стажировок.

Перспективы развития рынка связаны с несколькими тенденциями. Во-первых, компании продолжают внедрять цифровые инструменты подбора персонала для сокращения времени найма и повышения качества отбора кандидатов. Во-вторых, образовательные организации заинтересованы в более прозрачном сопровождении практик и стажировок. В-третьих, студенты нуждаются в специализированных сервисах, где они могут представлять не только опыт работы, но и учебные проекты, навыки, резюме и карьерные интересы. Также сохраняется рост значимости электронного и дистанционного взаимодействия в образовательной среде: по данным НИУ ВШЭ, на начало 2023/24 учебного года электронное обучение использовали 51%

студентов программ высшего образования, а дистанционные образовательные технологии – 46% [3].

Таким образом, рынок имеет выраженный потенциал для развития специализированных решений, ориентированных на связь между университетами, студентами и ИТ-компаниями. Рост HR-tech-сегмента, кадровый дефицит в ИТ и потребность образовательных организаций в цифровизации процессов практики и трудоустройства создают условия для вывода на рынок продукта, обеспечивающего централизованное взаимодействие всех участников процесса.

#### **1.4 Анализ конкурентов**

Для оценки рыночного положения разрабатываемой информационной платформы необходимо рассмотреть основные конкурентные решения, которые уже используются студентами, соискателями, работодателями и образовательными организациями для поиска вакансий, стажировок и взаимодействия с кандидатами. К числу наиболее близких конкурентов относятся массовые сервисы поиска работы, специализированные платформы для ИТ-рынка и внутренние университетские системы.

Одним из крупнейших конкурентов является платформа HeadHunter. Сервис предоставляет пользователям возможность создавать резюме, искать вакансии и откликаться на предложения работодателей. Его сильной стороной является большая база вакансий, развитые механизмы поиска, наличие сервисов для соискателей и работодателей, а также высокая узнаваемость на рынке. Однако HeadHunter ориентирован на широкий круг пользователей и не выделяет студентов как отдельную целевую группу. В рамках такой платформы начинающие специалисты конкурируют с более опытными кандидатами, а взаимодействие с университетами, учебными группами и преподавателями не является основным сценарием работы сервиса [4].

К числу прямых конкурентов также можно отнести SuperJob. Данный сервис позволяет создавать резюме, размещать вакансии и получать

предложения от работодателей. Преимуществами платформы являются простота использования, наличие базы работодателей и ориентация на массовый подбор персонала. Вместе с тем SuperJob, как и другие универсальные сервисы трудоустройства, не решает задачу сопровождения студенческих стажировок и практик. В нём отсутствуют механизмы работы с учебными группами, преподавателями и университетской структурой [5].

Отдельное место среди конкурентов занимает Хабр Карьера. В отличие от универсальных платформ, данный сервис ориентирован именно на рынок информационных технологий. На платформе представлены вакансии по различным ИТ-специализациям, включая разработку, тестирование, администрирование, аналитику и другие направления. Это делает сервис более релевантным для студентов ИТ-направлений. Однако Хабр Карьера остаётся прежде всего карьерной платформой для ИТ-специалистов и не включает полноценные инструменты сопровождения учебного процесса, управления группами студентов и взаимодействия с преподавателями [6].

В качестве конкурента также можно рассматривать государственный портал «Работа России». Портал представляет собой федеральную базу вакансий и резюме, предоставляет функции поиска работы, добавления резюме, размещения вакансий работодателями и взаимодействия с кандидатами. Преимуществом данного решения является официальный статус, масштаб базы вакансий и ориентация на широкий рынок труда. Однако для задач студенческого трудоустройства в ИТ-компаниях такой сервис является слишком общим. Он не ориентирован на построение связки «студент – преподаватель – работодатель» и не предоставляет специализированного интерфейса для сопровождения стажировок в рамках образовательной организации [7].

Существующие конкуренты обладают сильными сторонами в отдельных направлениях: крупные сервисы имеют большую базу вакансий, ИТ-платформы обеспечивают отраслевую релевантность, а университетские системы связаны с образовательной средой. Однако ни одно из рассмотренных решений не объединяет в полной мере функциональность карьерной платформы и

инструменты сопровождения студентов внутри учебной организации. Это формирует рыночную возможность для разработки специализированной информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях.

### **1.5 Бизнес-модель проекта**

Бизнес-модель проекта основана на создании специализированной информационной платформы, предназначенной для взаимодействия студентов, работодателей и образовательных организаций в процессе поиска стажировок, практик и первых рабочих мест в ИТ-сфере. Основная ценность продукта заключается в объединении в одном web-приложении функций поиска вакансий, создания резюме, обработки откликов, сопровождения учебных групп и коммуникации между участниками.

Ключевыми потребителями продукта являются образовательные организации и ИТ-компании. Для университетов платформа представляет ценность как инструмент цифровизации процессов трудоустройства, сопровождения практик и взаимодействия с работодателями. Для компаний продукт полезен как канал поиска начинающих специалистов, заинтересованных в прохождении стажировок и трудоустройстве. Студенты выступают основной пользовательской аудиторией платформы, поскольку именно для них создаются инструменты поиска вакансий, управления резюме и взаимодействия с работодателями.

Основной источник дохода — подписка для работодателей и образовательных организаций. Компании могут получать доступ к расширенным возможностям платформы: публикации вакансий, просмотру резюме студентов, подбору кандидатов, обработке откликов и коммуникации с соискателями. Образовательные организации могут использовать платформу для подключения учебных групп, сопровождения студентов и контроля взаимодействия с работодателями.

К основным каналам продвижения продукта относятся прямое взаимодействие с университетами, участие в карьерных мероприятиях, партнёрство с ИТ-компаниями, размещение информации на образовательных порталах.

Ключевыми ресурсами проекта являются программная платформа, команда разработки, база пользователей, партнёрские связи с образовательными организациями и работодателями, а также накопленные данные о вакансиях, резюме и откликах. К основным видам деятельности относятся разработка и сопровождение клиентской и серверной частей, поддержка пользователей, развитие функциональности, обеспечение стабильной работы системы и продвижение продукта на рынке.

Бизнес-модель проекта строится на предоставлении цифрового сервиса, который решает задачи сразу нескольких групп пользователей. Студенты получают удобный инструмент для поиска стажировок и первых рабочих мест, работодатели – канал привлечения начинающих ИТ-специалистов, а образовательные организации – средство сопровождения учебных групп и взаимодействия с компаниями. Такая модель позволяет развивать продукт как специализированную платформу для студенческого трудоустройства в ИТ-сфере.

## **1.6 Команда стартап-проекта**

Команда стартап-проекта сформирована из двух участников, каждый из которых отвечает за отдельное направление разработки информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях. Такое распределение ролей позволяет разделить зоны ответственности между клиентской и серверной частью приложения, а также обеспечить выполнение полного цикла создания программного продукта.

Frontend-разработчик – Поршнев Павел Александрович отвечает за проектирование и реализацию клиентской части информационной платформы. В его задачи входит разработка пользовательского интерфейса, реализация

страниц для студентов, работодателей и преподавателей, настройка маршрутизации, создание форм, отображение данных, взаимодействие с серверной частью через API, а также обеспечение удобства работы пользователей с web-приложением.

Backend-разработчик – Годов Дмитрий Алексеевич отвечает за проектирование и реализацию серверной части платформы. В его задачи входит разработка API, реализация бизнес-логики, организация работы с базой данных, обработка пользовательских запросов, настройка авторизации и обеспечение корректного взаимодействия серверной части с клиентским приложением.

Разделение команды на frontend- и backend-направления позволяет параллельно выполнять разработку разных частей продукта. Клиентская часть обеспечивает взаимодействие пользователя с системой через web-интерфейс, а серверная часть отвечает за хранение, обработку и передачу данных. Подобное распределение обязанностей повышает эффективность разработки, позволяя каждому участнику команды сосредоточиться на своей зоне ответственности.

Команда стартап-проекта обладает необходимыми техническими навыками для создания и дальнейшего развития информационной платформы.

## **1.7 Описание финансовой модели**

Финансовая модель строится на поэтапном развитии информационной платформы. Задачей начального этапа проекта является проверка востребованности продукта, привлечение первых пользователей и формирование базы студентов, работодателей и образовательных организаций.

Основными затратами проекта выступают – разработка и сопровождение программного продукта, оплата доменного имени, хостинга, а также расходы на продвижение платформы.

Финансовая модель формируется за счет предоставления платного доступа для работодателей. Для студентов функции платформы будут бесплатны, так как наличие активной студенческой аудитории повышает ценность продукта для компаний.

Для работодателей предусмотрена подписочная модель. Компании получают возможность размещать вакансии и получать отклики от студентов.

На раннем этапе развития проекта целесообразнее использовать freemium-модель. Базовый функционал предоставляется бесплатно для привлечения пользователей. Такой подход позволяет постепенно формировать устойчивую клиентскую базу.

Окупаемость проекта предполагается за счет роста числа подключенных работодателей и образовательных организаций. Чем больше студентов использует платформу, тем выше её ценность для компаний. В свою очередь, рост числа работодателей повышает привлекательность платформы для студентов и вузов.

Финансовая модель предусматривает умеренные первоначальные затраты, бесплатный доступ для студентов и монетизацию за счет работодателей. Такой подход позволяет развивать продукт сохраняя его доступность для основной пользовательской аудитории.

## **1.8 Развитие бизнес-проекта**

Развитие бизнес-проекта осуществляется поэтапно. На первом этапе основной задачей является завершение проекта, включая работоспособность ключевых функций: регистрация и авторизация пользователей, работу с профилем, размещение и просмотр вакансий, создание резюме, обработку откликов, управление учебными группами и обмен сообщениями между студентами и работодателями. Данный этап необходим для проверки работы основного функционала системы и получения первичной обратной связи.

Вторым этапом является внедрение платформы в ограниченное число учебных организаций и компаний-партнеров. Это позволит удобно выявить наиболее востребованные функции и определить направления дальнейшего развития продукта. Особое внимание на данном этапе уделяется анализу активности работодателей и студентов, и эффективности взаимодействия между участниками платформы.

Важным этапом развития проекта является увеличение пользовательской базы. Для этого предполагается подключение новых образовательных организаций и ИТ-компаний, заинтересованных в поиске молодых специалистов. Расширение числа участников повысит ценность платформы: студенты получают больше релевантных предложений, работодатели – доступ к большему числу кандидатов, а преподаватели – более удобный инструмент контроля и сопровождения учебных групп.

Таким образом, развитие бизнес-проекта связано с ростом пользователей и их потребностей, увеличение функциональных возможностей платформы и наращивание клиентской базы.

### **1.9 Описание рисков**

Для снижения возможного негативного влияния риски необходимо учитывать на этапе планирования проекта и предусматривать меры по их минимизации.

Одним из основных рисков является недостаточный интерес со стороны целевой аудитории. Студенты, работодатели и образовательные организации уже используют привычные инструменты: сервисы поиска работы, мессенджеры, электронную почту и внутренние системы университета. Для снижения данного риска необходимо проводить внедрение платформы, собирать обратную связь от пользователей и дорабатывать функциональность с учетом потребностей студентов, преподавателей и компаний.

Следующим риском является сложность привлечь интерес со стороны целевой аудитории. Для компаний ценность платформы напрямую зависит от количества их резюме. Если пользовательская база будет небольшой, работодатели могут не увидеть достаточной выгоды от размещения вакансий. Для минимизации риска целесообразно начинать развитие проекта с ограниченного числа партнерских организаций, заинтересованных в работе со студентами ИТ-направлений.

Коммерческим риском является недостаточная готовность образовательных организаций и работодателей оплачивать использование платформы. На раннем этапе продукт может восприниматься как дополнительный сервис, а не как обязательный инструмент. Для снижения данного риска целесообразно использовать поэтапную модель монетизации: предоставить базовые функции бесплатно, а платные возможности вводить после подтверждения востребованности продукта.

Основные риски проекта связаны с привлечением пользователей и коммерческой устойчивостью продукта. Предусмотренные меры по их снижению позволяют повысить вероятность успешного вывода платформы на рынок и обеспечить дальнейшее развитие бизнес-проекта.

## **2 Анализ требований и постановка задачи**

### **2.1 Анализ предметной области**

Информационная платформа трудоустройства и стажировок студентов в ИТ-компаниях представляет собой web-систему, предназначенную для организации взаимодействия между студентами, работодателями и преподавателями. В рамках данной системы клиентская часть играет ключевую роль, поскольку именно она обеспечивает пользователям доступ к основным функциональным возможностям через графический web-интерфейс. Пользователь взаимодействует с платформой при помощи браузера, без необходимости установки дополнительного программного обеспечения на локальный компьютер.

В настоящее время процессы поиска стажировок и начального трудоустройства часто распределены между различными сервисами, мессенджерами, электронными таблицами и внутренними каналами образовательных организаций. Такой подход затрудняет централизованное хранение информации о вакансиях, резюме, откликах и учебных группах. Использование специализированной информационной платформы позволяет

объединить данные процессы в едином интерфейсе, упростить навигацию для пользователей и повысить удобство взаимодействия между участниками.

Целевыми пользователями платформы являются студенты, работодатели и преподаватели. Студенты получают возможность просматривать вакансии, создавать резюме, отправлять отклики и взаимодействовать с работодателями. Работодатели могут размещать вакансии, просматривать отклики и работать с профилем компании. Преподаватели используют систему для управления учебными группами и обработки заявок студентов. Клиентская часть приложения обеспечивает разделение пользовательских сценариев по ролям, защищённый доступ к страницам, обработку форм, отображение данных, полученных от серверной части, а также обмен сообщениями между участниками платформы.

## **2.2 Анализ программных аналогов платформы трудоустройства**

В рамках анализа предметной области необходимо разграничить классы систем, схожих по назначению с разрабатываемой платформой. Современные решения для поиска стажировок и трудоустройства можно разделить на несколько групп в зависимости от выполняемых задач: сервисы поиска вакансий, профессиональные социальные платформы, образовательные порталы с карьерными разделами и внутренние системы учебных организаций.

Далее приведён обзор решений, относящихся к каждой из указанных категорий, с выявлением их сильных и слабых сторон. Особое внимание уделяется функциональным возможностям клиентской части, удобству пользовательского интерфейса, поддержке различных ролей, работе с вакансиями, резюме, откликами и коммуникацией между участниками платформы.

### **2.2.1 Анализ общих платформ поиска работы**

К категории сервисов поиска работы относятся веб-сервисы, такие как HeadHunter, SuperJob, «Хабр Карьера» и другие. Данные платформы

предоставляют пользователям возможность размещать резюме, просматривать вакансии, направлять отклики и взаимодействовать с работодателями. Их преимуществом является наличие обширной базы вакансий и организаций, а также механизмы поиска и фильтрации предложений.

Указанные сервисы ориентированы на широкий круг соискателей и не учитывают специфику взаимодействия в образовательной среде. Кроме того, начинающие специалисты и студенты могут уступать более опытным кандидатам, что снижает эффективность поиска стажировок и первых рабочих мест.

### **2.2.2 Анализ специализированных платформ работы в ИТ-сфере**

Платформы данной категории, например «Хабр Карьера», ориентированы на подбор вакансий и специалистов именно в сфере информационных технологий. За счёт узкой направленности такие сервисы позволяют студентам ИТ-направлений находить более подходящие предложения о стажировках и работе, а работодателям – взаимодействовать с кандидатами, обладающими профильными навыками.

Однако подобные платформы также не решают задачи, характерные для образовательной среды. В них отсутствует сопровождение процессов, связанных с практикой или стажировкой в рамках учебной организации. Поэтому такие решения не обеспечивают полного взаимодействия между студентами, преподавателями и работодателями.

### **2.2.3 Анализ университетских информационных систем**

Отдельную категорию программных аналогов составляют внутренние университетские системы. Они применяются для организации учебного процесса, хранения сведений о студентах, учебных группах, расписании, успеваемости и документах. Основным преимуществом таких систем является их непосредственная связь со структурой образовательной организации.

В качестве примера можно рассмотреть модуль «ВУЗ + Работодатель» на базе электронной информационно-образовательной среды (ЭИОС) ПГУ [8]. Данный модуль предоставляет студентам возможность создавать резюме, откликаться на вакансии, а также подбирать компании для прохождения стажировки или практики. Однако, несмотря на ориентацию на университетскую среду, система имеет ряд ограничений. Взаимодействие с работодателями в основном осуществляется через контактные данные организации, например электронную почту или телефон, без встроенного обмена сообщениями внутри платформы. Кроме того, система ориентирована преимущественно на поиск вакансий и мест практики, но не обеспечивает полноценное сопровождение данного процесса: отсутствует просмотр активности студентов, контроля прохождения практики и отслеживания взаимодействия студентов с конкретными компаниями.

#### **2.2.4 Результаты анализа**

В результате анализа программных аналогов были выделены основные критерии сравнения, позволяющие оценить соответствие существующих решений задачам разрабатываемой платформы. При сравнении учитывались как базовые функции, необходимые для работы студентов и работодателей, так и специализированные возможности, связанные с образовательной средой: участие преподавателей, работа с учебными группами, сопровождение практики и контроль активности студентов.

Специализированные платформы поиска работы в IT-сфере являются более релевантными для студентов технических направлений, так как ориентированы на вакансии в области информационных технологий. Вместе с тем они также не обеспечивают взаимодействие с образовательной организацией и не предусматривают механизмов сопровождения студентов в рамках учебного процесса.

Университетские системы, напротив, связаны со структурой образовательной организации и могут использоваться для хранения информации

о студентах, группах и местах прохождения практики. Однако такие решения часто ограничены в части взаимодействия с работодателями и не всегда предоставляют удобные средства коммуникации, просмотра активности студентов и комплексной работы с откликами. Сравнение разрабатываемой платформы с представителями рассмотренных категорий приведено в таблице 1.

Таблица 1 – Сравнение разрабатываемой платформы с программными аналогами

| Критерий                                                    | Общие платформы поиска работы | Специализированные платформы поиска работы в IT-сфере | Университетские системы | Разрабатываемая платформа |
|-------------------------------------------------------------|-------------------------------|-------------------------------------------------------|-------------------------|---------------------------|
| Размещение и управление резюме                              | Да                            | Да                                                    | Да                      | Да                        |
| Поиск и просмотр вакансий                                   | Да                            | Да                                                    | Да                      | Да                        |
| Встроенная коммуникация между студентами и компаниями       | Да                            | Да                                                    | Нет                     | Да                        |
| Ориентация на IT-сферу                                      | Частично                      | Да                                                    | Частично                | Да                        |
| Участие преподавателей                                      | Нет                           | Нет                                                   | Да                      | Да                        |
| Работа с учебными группами                                  | Нет                           | Нет                                                   | Да                      | Да                        |
| Просмотр активности студентов                               | Нет                           | Нет                                                   | Нет                     | Да                        |
| Подбор места практики                                       | Нет                           | Нет                                                   | Да                      | Да                        |
| Запись на практику внутри платформы                         | Нет                           | Нет                                                   | Нет                     | Да                        |
| Просмотр информации о местах прохождения практик студентами | Нет                           | Нет                                                   | Нет                     | Да                        |

По результатам сравнения можно сделать вывод, что существующие программные аналоги решают отдельные задачи рассматриваемой предметной области, но не обеспечивают комплексного подхода к взаимодействию студентов, работодателей и преподавателей. Общие и специализированные платформы поиска работы удобны для размещения вакансий и резюме, однако

не учитывают организационные процессы образовательной среды. Университетские системы, в свою очередь, связаны с учебным процессом, но имеют ограничения в части взаимодействия с работодателями и сопровождения карьерных сценариев студентов.

Разрабатываемая платформа объединяет функции поиска вакансий, управления резюме, обработки откликов, обмена сообщениями, работы с учебными группами и сопровождения практики. За счёт этого она обеспечивает более целостное решение задач, связанных с трудоустройством и стажировками студентов в ИТ-компаниях.

### **2.3 Постановка задачи на разработку информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях**

На основе анализа предметной области и существующих программных аналогов была выявлена необходимость разработки информационной платформы, предназначенной для организации взаимодействия студентов и работодателей в едином цифровом пространстве.

Целью работы является разработка информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях. Система предназначена для автоматизации процессов поиска вакансий и стажировок, взаимодействия работодателей и студентов, а также управления пользовательскими профилями и информацией о компаниях.

В рамках платформы предусматриваются две основные роли пользователей: студент и работодатель. Для каждой роли реализуется собственный набор функциональных возможностей и пользовательских интерфейсов.

Базовый функционал, доступный всем пользователям:

- регистрация в системе;
- авторизация пользователей;
- управление личным кабинетом;
- редактирование персональных данных;

- изменение учетной информации;
- выход из системы.

#### Функционал студентов:

- управление личным профилем;
- добавление и редактирование информации о себе;
- указание информации об образовании и специализации;
- добавление профессиональных ссылок;
- загрузка аватара пользователя;
- поиск и просмотр вакансий;
- просмотр информации о компаниях;
- просмотр рекомендаций вакансий;
- взаимодействие с работодателями;
- просмотр откликов и предложений.

#### Функционал работодателей:

- управление профилем работодателя;
- создание и редактирование информации о компании;
- загрузка и удаление логотипа компании;
- публикация вакансий и стажировок;
- редактирование вакансий;
- просмотр профилей студентов;
- поиск кандидатов;
- взаимодействие со студентами;
- управление откликами и вакансиями.

Одной из ключевых особенностей платформы является разграничение доступа по ролям. В зависимости от типа пользователя система предоставляет доступ только к соответствующему функционалу и интерфейсам. Такой подход позволяет упростить взаимодействие с системой и повысить удобство использования платформы.

Основные бизнес-процессы платформы включают:

- регистрацию и авторизацию пользователей;
- создание и управление профилями;
- публикацию вакансий и стажировок;
- поиск вакансий студентами;
- взаимодействие между работодателями и студентами;
- управление информацией о компаниях.

Нефункциональные требования предполагают, что система должна обеспечивать стабильную работу при одновременном взаимодействии нескольких пользователей, поддерживать разграничение прав доступа, обеспечивать безопасность пользовательских данных и возможность дальнейшего расширения функциональности платформы.

Клиентская часть приложения должна обладать современным и адаптивным пользовательским интерфейсом, соответствующим UI/UX-подходам. Интерфейс системы должен обеспечивать удобство взаимодействия пользователей с платформой как на персональных компьютерах, так и на мобильных устройствах.

Для разработки клиентской части приложения используются современные веб-технологии:

- React;
- Vite;
- React Router;
- Axios;
- Tailwind CSS.

Серверная часть системы должна обеспечивать обработку запросов, реализацию бизнес-логики, аутентификацию пользователей и взаимодействие с базой данных. Для обеспечения безопасности предполагается использование JWT-аутентификации с хранением токенов в HTTP-only cookie.

Хранение данных должно осуществляться в отдельной базе данных, а развертывание и сопровождение системы должны быть упрощены за счет использования контейнеризации с применением Docker и Docker Compose.

Такой подход обеспечивает переносимость приложения, единообразие конфигураций и удобство развертывания системы в различных средах.

Отдельным требованием к разработке является выполнение проектирования программного обеспечения с подготовкой UML-диаграмм, диаграмм компонентов, диаграмм вариантов использования и других артефактов, описывающих структуру системы и взаимодействие ее компонентов.

Разрабатываемое программное обеспечение должно быть протестировано и подготовлено к дальнейшему развитию и масштабированию. Гибкая архитектура системы должна обеспечивать возможность последующего расширения функциональности платформы, включая внедрение аналитики, рекомендаций вакансий, системы откликов и коммуникации между пользователями.

Сформулированные требования являются основой для дальнейшего проектирования и разработки информационной платформы трудоустройства и стажировок студентов в IT-компаниях.

## **2.4 Анализ требований к разрабатываемой информационной платформе трудоустройства и стажировок студентов в IT-компаниях**

### **2.4.1 Анализ функциональных требований**

В соответствии с определёнными функциональными требованиями были разработаны диаграммы вариантов использования для основных ролей системы. Данные диаграммы позволяют наглядно представить состав пользовательских сценариев, доступных каждой роли, а также отразить характер взаимодействия пользователя с клиентской частью платформы.

Диаграмма вариантов использования для роли студента приведена на рисунке А.1.

Ниже приведены спецификации, основных пользовательских сценариев системы для роли студента.

Спецификация варианта использования «Вступить в учебную группу» приведена в таблице 2.

Таблица 2 – Спецификация варианта использования «Вступить в учебную группу»

| Поле                  | Содержание                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Наименование          | Вступить в учебную группу                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Актёры                | Студент, преподаватель                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Цель                  | Привязать студента к учебной группе                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Краткое описание      | Студент вводит код приглашения учебной группы. Система проверяет код и создаёт заявку на вступление. Преподаватель рассматривает заявку и принимает или отклоняет её. При принятии заявки система привязывает студента к учебной группе, обновляет его статус в системе и открывает ему доступ к функциональным возможностям платформы.                                                                                                                                                                                                                                                      |
| Предусловия           | Студент авторизован. Учебная группа создана преподавателем. Для группы сформирован код приглашения. Студент ещё не привязан к учебной группе.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Основной поток        | <ol style="list-style-type: none"> <li>1. Студент вводит код приглашения.</li> <li>2. Система проверяет существование учебной группы с введенным кодом приглашения.</li> <li>3. Система проверяет, что студент ещё не состоит в учебной группе.</li> <li>4. Система проверяет, что у студента нет заявки на вступление в другую группу.</li> <li>5. Система создаёт заявку на вступление в учебную группу.</li> <li>6. Преподаватель открывает заявку и принимает её.</li> <li>7. Система привязывает студента к учебной группе и открывает доступ ко всему функционалу студента.</li> </ol> |
| Альтернативные потоки | <p>A1: Код приглашения не найден – система выводит сообщение об ошибке, заявка не создаётся.</p> <p>A2: Студент уже состоит в учебной группе – система выводит сообщение об ошибке, заявка не создаётся.</p> <p>A3: У студента уже есть активная заявка – система выводит информацию о текущей заявке, новая заявка не создаётся.</p>                                                                                                                                                                                                                                                        |

Спецификация варианта использования «Создать резюме» представлена в таблице 3

Таблица 3 – Спецификация варианта использования «Создать резюме»

| Поле                  | Содержание                                                                                                                                                                                                                                                                     |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Наименование          | Создать резюме                                                                                                                                                                                                                                                                 |
| Актёры                | Студент                                                                                                                                                                                                                                                                        |
| Цель                  | Добавить новое резюме в профиль.                                                                                                                                                                                                                                               |
| Краткое описание      | Студент заполняет все поля резюме и сохраняет его                                                                                                                                                                                                                              |
| Предусловия           | Права доступа студента; студент привязан к учебной группе.                                                                                                                                                                                                                     |
| Основной поток        | 1. Студент заполняет все необходимые поля.<br>2. Студент указывает ключевые навыки, которыми он владеет.<br>3. Система проверяет корректность введенных данных.<br>4. Студент указывает сведения об опыте работы (если имеется).<br>5. Система сохраняет резюме в базе данных. |
| Альтернативные потоки | A1: Не заполнены обязательные поля или данные не проходят валидацию – система выводит сообщения об ошибках, вакансия не создаётся.                                                                                                                                             |
| Постусловия           | Резюме создано и доступно для просмотра на платформе.                                                                                                                                                                                                                          |

Спецификация варианта использования «Создать резюме» представлена в таблице 4

Таблица 4 – Спецификация варианта использования «Оставить отклик на вакансию»

| Поле             | Содержание                                                                                                                                              |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Наименование     | Оставить отклик на вакансию                                                                                                                             |
| Актёры           | Студент                                                                                                                                                 |
| Цель             | Отправить работодателю отклик на выбранную вакансию.                                                                                                    |
| Краткое описание | Студент выбирает из списка предложенных вакансий, подходящую ему по навыкам и отправляет отзыв с сопроводительным письмом работодателю на рассмотрение. |
| Предусловия      | Авторизация пользователя под ролью студента; проверка привязки студента к учебной группе                                                                |

Продолжение таблицы 4

| Поле                  | Содержание                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Основной поток        | 1. Студент выбирает вакансию.<br>2. Студент нажимает кнопку создания отклика.<br>3. Студент выбирает резюме и отправляет отклик.<br>4. Система проверяет принадлежность резюме студенту.<br>5. Система проверяет, что вакансия существует и активна.<br>6. Система проверяет отсутствие активного отклика студента на эту вакансию.<br>7. Система создаёт или находит чат между студентом и компанией и прикрепляет его к отклику.<br>8. Система создаёт отклик со статусом «Ожидание ответа». |
| Альтернативные потоки | A1: Вакансия не найдена или неактивна – система выводит ошибку, отклик не создаётся.<br>A2: Резюме не принадлежит студенту или не найдено – система выводит ошибку, отклик не создаётся.<br>A3: У студента уже есть активный отклик на вакансию – система выводит ошибку, повторный отклик не создаётся.                                                                                                                                                                                       |
| Постусловия           | Отклик создан и доступен для просмотра студенту и работодателю                                                                                                                                                                                                                                                                                                                                                                                                                                 |

На рисунке А.2 представлена диаграмма вариантов использования для роли работодателя.

Ниже представлены спецификации для наиболее важных вариантов использования.

Спецификация варианта использования «Создать вакансию» представлена в таблице 5.

Таблица 5 – Спецификация варианта использования «Создать вакансию»

| Поле         | Содержание                                                                      |
|--------------|---------------------------------------------------------------------------------|
| Наименование | Создать вакансию                                                                |
| Актёры       | Работодатель                                                                    |
| Цель         | Целью является размещение компанией новой вакансии на информационной платформе. |

Продолжение таблицы 5

| Поле                  | Содержание                                                                                                                                                                                                         |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Краткое описание      | Работодатель заполняет все поля вакансии и сохраняет её – система записывает созданную вакансию в базу данных.                                                                                                     |
| Предусловия           | Права доступа работодателя.                                                                                                                                                                                        |
| Основной поток        | 1. Работодатель заполняет все необходимые поля.<br>2. Работодатель указывает требуемые для вакансии навыки.<br>3. Система проверяет корректность введенных данных.<br>4. Система сохраняет вакансию в базе данных. |
| Альтернативные потоки | A1: Не заполнены обязательные поля или данные не проходят валидацию – система выводит сообщения об ошибках, вакансия не создаётся.                                                                                 |
| Постусловия           | Вакансия создана и доступна для просмотра на платформе.                                                                                                                                                            |

Спецификация варианта использования «Искать резюме» представлена в таблице 6.

Таблица 6 – Спецификация варианта использования «Искать резюме»

| Поле                  | Содержание                                                                                                                                                                                                                                        |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Наименование          | Искать резюме                                                                                                                                                                                                                                     |
| Актёры                | Работодатель                                                                                                                                                                                                                                      |
| Цель                  | Найти подходящие резюме для приглашения                                                                                                                                                                                                           |
| Краткое описание      | Работодатель указывает фильтры поиска. Система отображает подходящие резюме.                                                                                                                                                                      |
| Предусловия           | Права доступа работодателя.                                                                                                                                                                                                                       |
| Основной поток        | 1. Работодатель указывает параметры поиска.<br>2. Работодатель отправляет запрос на поиск.<br>3. Система фильтрует резюме в соответствии с переданными параметрами и возвращает полученный список резюме.<br>4. Система отображает список резюме. |
| Альтернативные потоки | A1: По заданным параметрам не найдено ни одного резюме – система отображает пустой список и сообщение об отсутствии результатов.                                                                                                                  |

## Продолжение таблицы 6

| Поле        | Содержание                                                                         |
|-------------|------------------------------------------------------------------------------------|
| Постусловия | Работодатель получил список резюме в соответствии с указанными параметрами поиска. |

На рисунке А.3 представлена диаграмма вариантов использования для роли преподавателя.

Ниже представлены основные спецификации вариантов использования для роли преподавателя.

Спецификация варианта использования «Создать учебную группу» представлена в таблице 7.

Таблица 7 – Спецификация варианта использования «Создать учебную группу»

| Поле                  | Содержание                                                                                                                                                                                                                                                                                                                                            |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Наименование          | Создать учебную группу                                                                                                                                                                                                                                                                                                                                |
| Актёры                | Преподаватель                                                                                                                                                                                                                                                                                                                                         |
| Цель                  | Создать новую учебную группу под образовательную программу.                                                                                                                                                                                                                                                                                           |
| Краткое описание      | Преподаватель указывает основные данные о учебной группе, затем система сохраняет группу и на основе полученных данных формирует для неё уникальный код приглашения.                                                                                                                                                                                  |
| Предусловия           | Пользователь авторизован с правами доступа преподавателя.                                                                                                                                                                                                                                                                                             |
| Основной поток        | 1. Преподаватель указывает образовательную программу и код поступления.<br>2. Система формирует название группы на основании предоставленных сведений.<br>3. Система генерирует уникальный код приглашения для группы.<br>4. Система сохраняет данные об учебной группе в базу данных.<br>5. Система отображает преподавателю код приглашения группы. |
| Альтернативные потоки | А1: Не заполнены обязательные поля или данные не проходят валидацию – система выводит сообщения об ошибках, учебная группа не создаётся.                                                                                                                                                                                                              |
| Постусловия           | Учебная группа создана с уникальным кодом приглашения.                                                                                                                                                                                                                                                                                                |

Спецификация варианта использования «Оценить решение студента по практике» представлена в таблице 8.

Таблица 8 – Спецификация варианта использования «Оценить решение студента по практике»

| Поле                  | Содержание                                                                                                                                                                                                  |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Наименование          | Оценить решение студента по практике                                                                                                                                                                        |
| Актёры                | Преподаватель                                                                                                                                                                                               |
| Цель                  | Выставить оценку решению студента и завершить практику.                                                                                                                                                     |
| Краткое описание      | Преподаватель выставляет оценку – система сохраняет оценку и помечает практику пройденной.                                                                                                                  |
| Предусловия           | Права доступа преподавателя; за преподавателем закреплена группа по практике.                                                                                                                               |
| Основной поток        | 1. Преподаватель выбирает студента, просматривает его решение и выставляет оценку.<br>2. Система проверяет, что практика принята работодателем.<br>3. Система сохраняет оценку по практике в БД.            |
| Альтернативные потоки | A1: Указана не корректная оценка – система выводит сообщение об ошибке, оценка не выставляется.<br>A2: Практика ещё не принята работодателем – система выводит сообщение об ошибке, оценка не выставляется. |
| Постусловия           | Учебная группа создана с уникальным кодом приглашения.                                                                                                                                                      |

По итогам анализа функциональных требований были определены основные варианты использования системы для каждой из ролей. Полученные диаграммы вариантов использования служат основой для дальнейшего проектирования платформы трудоустройства и стажировок в ИТ-компаниях.

#### 2.4.2 Анализ нефункциональных требований

Клиентская часть информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях должна соответствовать ряду нефункциональных требований, обеспечивающих удобную, стабильную и безопасную работу пользователей с web-приложением. Нефункциональные требования определяют качественные характеристики клиентского приложения, особенности его интерфейса, производительности, доступности и взаимодействия с серверной частью [9].

Были выделены следующие нефункциональные требования клиентской части:

- производительность интерфейса;
- удобство пользовательского интерфейса;
- адаптивность интерфейса;
- кроссбраузерность;
- надежность взаимодействия с серверной частью;
- единообразие визуального оформления.

Соблюдение указанных требований позволит создать интерфейс, обеспечивающий высокую эффективность и удобство работы для всех пользователей.

## **2.5 Выбор инструментальных средств**

При разработке информационной платформы трудоустройства и стажировок студентов в IT-компаниях был выбран современный стек технологий, обеспечивающий удобство разработки, высокую производительность, масштабируемость и простоту сопровождения приложения. Разработка программного обеспечения выполнялась с использованием современных средств frontend- и backend-разработки, а также инструментов контейнеризации и контроля версий.

Для разработки клиентской части приложения использовалась библиотека React. Данный фреймворк используется в разработке современных web-приложений. Использование React обеспечивает высокую производительность клиентской части, переиспользуемость компонентов и удобство дальнейшего развития платформы [10].

В качестве инструмента сборки и запуска frontend-приложения использовался Vite. Он обеспечивает высокую скорость запуска проекта, быструю пересборку и удобную среду разработки. Vite позволяет ускорить процесс разработки [11].

Для организации маршрутизации внутри клиентского приложения использовалась библиотека React Router. Это позволяет реализовать навигацию между страницами приложения без полной перезагрузки [12].

Чтобы организовать взаимодействие клиентской части с серверным API использовалась библиотека Axios. Она предоставляет удобный механизм отправки запросов и обработки ответов сервера, а также упрощает работу с аутентификацией пользователей [13].

В качестве инструмента стилизации интерфейса использовался Tailwind CSS. Этот CSS-фреймворк позволяет быстро создавать современные адаптивные интерфейсы и реализовывать UI/UX-подходы при разработке веб-приложения. Tailwind CSS обеспечивает гибкость настройки внешнего вида приложения и упрощает поддержку стилей проекта [14].

Разработка серверной части приложения выполнялась с использованием языка программирования C# и фреймворка ASP.NET Core [15].

В качестве системы управления базами данных используется PostgreSQL.

Для документирования и тестирования API используется Swagger.

В качестве системы контроля версий используется система Git. Она позволяет отслеживать изменения в проекте, организовывать совместную разработку и сохранять историю изменений программного кода.

Развертывание приложения осуществляется с использованием Docker и Docker Compose.

Стилем реализации проекта был выбран клиент-серверный подход. Клиентская часть отвечает за отображение пользовательского интерфейса и взаимодействие с пользователем, а серверная часть обеспечивает обработку запросов, хранение данных и реализацию бизнес-логики приложения.

Для проектирования структуры системы и описания взаимодействия компонентов используется нотация UML 2.0. С помощью UML-диаграмм описываются основные элементы архитектуры системы, пользовательские сценарии и взаимодействие между компонентами платформы.

Определенные инструментальные средства необходимы для разработки информационной платформы трудоустройства и стажировки студентов в ИТ-компаниях. Выбранный стек технологий полностью соответствует требованиям проекта, обеспечивает удобство разработки, масштабируемость приложения и возможность дальнейшего развития платформы.

## 2.6 Планирование проекта и оценка бюджета

В качестве инструмента для планирования проекта был выбран Microsoft Project 2007. Основные задачи проекта отображена на рисунке 1.

|    |    | Название задачи                            | Длительность   | Начало             | Окончание          | Предшественники |
|----|-------------------------------------------------------------------------------------|--------------------------------------------|----------------|--------------------|--------------------|-----------------|
| 1  |                                                                                     | Начало проекта                             | 0 дней         | Чт 04.06.26        | Чт 04.06.26        |                 |
| 2  |    | <b>Анализ предметной области и постано</b> | <b>5 дней?</b> | <b>Чт 04.06.26</b> | <b>Ср 10.06.26</b> |                 |
| 3  |                                                                                     | Анализ предметной области                  | 1 день?        | Чт 04.06.26        | Чт 04.06.26        |                 |
| 4  |                                                                                     | Анализ программных аналогов                | 1 день?        | Пт 05.06.26        | Пт 05.06.26        | 3               |
| 5  |                                                                                     | Постановка задачи на разработку            | 1 день?        | Чт 04.06.26        | Чт 04.06.26        |                 |
| 6  |                                                                                     | Анализ функциональных требований           | 2 дней         | Пт 05.06.26        | Пн 08.06.26        | 5               |
| 7  |                                                                                     | Анализ нефункциональных требовани          | 1 день?        | Вт 09.06.26        | Вт 09.06.26        | 6               |
| 8  |                                                                                     | Выбор инструментальных средств             | 1 день?        | Ср 10.06.26        | Ср 10.06.26        | 7               |
| 9  |  | <b>Проектирование системы</b>              | <b>6 дней</b>  | <b>Чт 11.06.26</b> | <b>Чт 18.06.26</b> | <b>2</b>        |
| 10 |                                                                                     | Проектирование архитектуры                 | 2 дней         | Чт 11.06.26        | Пт 12.06.26        |                 |
| 11 |                                                                                     | Проектирование пользовательского и         | 2 дней         | Пн 15.06.26        | Вт 16.06.26        | 10              |
| 12 |                                                                                     | Проектирование компонентов клиент          | 2 дней         | Ср 17.06.26        | Чт 18.06.26        | 11              |
| 13 |  | <b>Реализация системы</b>                  | <b>21 дней</b> | <b>Пт 19.06.26</b> | <b>Пт 17.07.26</b> | <b>9</b>        |
| 14 |                                                                                     | Реализация клиентской части                | 14 дней        | Пт 19.06.26        | Ср 08.07.26        |                 |
| 15 |                                                                                     | Реализация взаимодействия клиентс          | 7 дней         | Чт 09.07.26        | Пт 17.07.26        | 14              |
| 16 |  | <b>Тестирование</b>                        | <b>6 дней</b>  | <b>Пн 20.07.26</b> | <b>Пн 27.07.26</b> | <b>13</b>       |
| 17 |                                                                                     | Функциональное тестирование                | 2 дней         | Пн 20.07.26        | Вт 21.07.26        |                 |
| 18 |                                                                                     | Нефункционально тестирование               | 2 дней         | Ср 22.07.26        | Чт 23.07.26        | 17              |
| 19 |                                                                                     | Модульное тестирование                     | 2 дней         | Пт 24.07.26        | Пн 27.07.26        | 18              |
| 20 |  | <b>Завершение проекта</b>                  | <b>4 дней</b>  | <b>Вт 28.07.26</b> | <b>Пт 31.07.26</b> | <b>16</b>       |
| 21 |                                                                                     | Исправление выявленных ошибок              | 2 дней         | Вт 28.07.26        | Ср 29.07.26        |                 |
| 22 |                                                                                     | Составление документации                   | 2 дней         | Чт 30.07.26        | Пт 31.07.26        | 21              |
| 23 |                                                                                     | Развертывание системы                      | 2 дней         | Вт 28.07.26        | Ср 29.07.26        |                 |
| 24 |                                                                                     | Конец проекта                              | 0 дней?        | Пт 31.07.26        | Пт 31.07.26        | 20              |

Рисунок 1 – Основные задачи проекта

На рисунке 2 представлены ресурсы, выделенные для реализации клиентского приложения.

|   |  | Название ресурса | Тип      | Единицы измерения материалов | Краткое название | Группа | Макс. единиц | Стандартная ставка | Ставка сверхурочных | Затраты на исполыз. | Начисление       | Базовый календарь |
|---|-------------------------------------------------------------------------------------|------------------|----------|------------------------------|------------------|--------|--------------|--------------------|---------------------|---------------------|------------------|-------------------|
| 1 |                                                                                     | Аналитик         | Трудовой |                              | А                |        | 100%         | 600,00 Р/ч         | 800,00 Р/ч          | 0,00 Р              | Пропорциональное | Стандартный       |
| 2 |                                                                                     | Проектировщик    | Трудовой |                              | П-к              |        | 100%         | 500,00 Р/ч         | 700,00 Р/ч          | 0,00 Р              | Пропорциональное | Стандартный       |
| 3 |                                                                                     | Программист      | Трудовой |                              | П                |        | 100%         | 800,00 Р/ч         | 1 000,00 Р/ч        | 0,00 Р              | Пропорциональное | Стандартный       |
| 4 |                                                                                     | Тестировщик      | Трудовой |                              | Т                |        | 100%         | 400,00 Р/ч         | 600,00 Р/ч          | 0,00 Р              | Пропорциональное | Стандартный       |
| 5 |                                                                                     | Дизайнер         | Трудовой |                              | Д                |        | 100%         | 500,00 Р/ч         | 700,00 Р/ч          | 0,00 Р              | Пропорциональное | Стандартный       |

Рисунок 2 – Лист ресурсов проекта

Для каждой задачи были выделены необходимые для выполнения ресурсы. Распределение ресурсов позволяет определить, какие специалисты задействованы на отдельных этапах проекта.

Результат распределения ресурсов приведен на рисунке 3.

|    | 📌 | Название задачи                                       | Длительность | Начало      | Окончание   | Предшественники | Названия ресурсов                    |
|----|---|-------------------------------------------------------|--------------|-------------|-------------|-----------------|--------------------------------------|
| 1  |   | Начало проекта                                        | 0 дней       | Чт 04.06.26 | Чт 04.06.26 |                 |                                      |
| 2  | ▢ | Анализ предметной области и постановка задачи         | 7 дней?      | Чт 04.06.26 | Пт 12.06.26 |                 |                                      |
| 3  |   | Анализ предметной области                             | 1 день?      | Чт 04.06.26 | Чт 04.06.26 |                 | Аналитик                             |
| 4  |   | Анализ программных аналогов                           | 1 день?      | Пт 05.06.26 | Пт 05.06.26 | 3               | Аналитик                             |
| 5  |   | Постановка задачи на разработку                       | 1 день?      | Пн 08.06.26 | Пн 08.06.26 | 4               | Аналитик                             |
| 6  |   | Анализ функциональных требований                      | 2 дней       | Вт 09.06.26 | Ср 10.06.26 | 5               | Аналитик[70%]; Программист[30%]      |
| 7  |   | Анализ нефункциональных требований                    | 1 день?      | Чт 11.06.26 | Чт 11.06.26 | 6               | Аналитик[70%]; Программист[30%]      |
| 8  |   | Выбор инструментальных средств                        | 1 день?      | Пт 12.06.26 | Пт 12.06.26 | 7               | Проектировщик                        |
| 9  | ▢ | Проектирование системы                                | 6,2 дней     | Пн 15.06.26 | Вт 23.06.26 | 2               |                                      |
| 10 |   | Проектирование архитектуры                            | 2 дней       | Пн 15.06.26 | Вт 16.06.26 |                 | Проектировщик                        |
| 11 |   | Проектирование пользовательского интерфейса           | 2,2 дней     | Ср 17.06.26 | Пт 19.06.26 | 10              | Проектировщик[70%]; Дизайнер[30%]    |
| 12 |   | Проектирование компонентов клиентской части           | 2 дней       | Пт 19.06.26 | Вт 23.06.26 | 11              | Проектировщик[70%]; Программист[30%] |
| 13 | ▢ | Реализация системы                                    | 21 дней      | Вт 23.06.26 | Ср 22.07.26 | 9               |                                      |
| 14 |   | Реализация клиентской части                           | 14 дней      | Вт 23.06.26 | Пн 13.07.26 |                 | Программист                          |
| 15 |   | Реализация взаимодействия клиентской части с сервером | 7 дней       | Пн 13.07.26 | Ср 22.07.26 | 14              | Программист                          |
| 16 | ▢ | Тестирование                                          | 6 дней       | Ср 22.07.26 | Чт 30.07.26 | 13              |                                      |
| 17 |   | Функциональное тестирование                           | 2 дней       | Ср 22.07.26 | Пт 24.07.26 |                 | Тестировщик                          |
| 18 |   | Нефункциональное тестирование                         | 2 дней       | Пт 24.07.26 | Вт 28.07.26 | 17              | Тестировщик                          |
| 19 |   | Модульное тестирование                                | 2 дней       | Вт 28.07.26 | Чт 30.07.26 | 18              | Тестировщик                          |
| 20 | ▢ | Завершение проекта                                    | 8,67 дней    | Чт 30.07.26 | Вт 11.08.26 | 16              |                                      |
| 21 |   | Исправление выявленных ошибок                         | 2 дней       | Чт 30.07.26 | Пн 03.08.26 |                 | Программист                          |
| 22 |   | Составление документации                              | 6,67 дней    | Пн 03.08.26 | Вт 11.08.26 | 21              | Программист[70%]; Аналитик[30%]      |
| 23 |   | Развертывание системы                                 | 2 дней       | Чт 30.07.26 | Пн 03.08.26 |                 | Аналитик                             |
| 24 |   | Конец проекта                                         | 0 дней?      | Вт 11.08.26 | Вт 11.08.26 | 20              |                                      |

Рисунок 3 – Задачи проекта с распределенными ресурсами

В рамках календарного плана были отображены основные этапы проекта: анализ предметной области, проектирование клиентской части, реализация пользовательских интерфейсов и тестирование разработанного функционала.

На рисунке 4 представлена диаграмма Ганта для проекта.

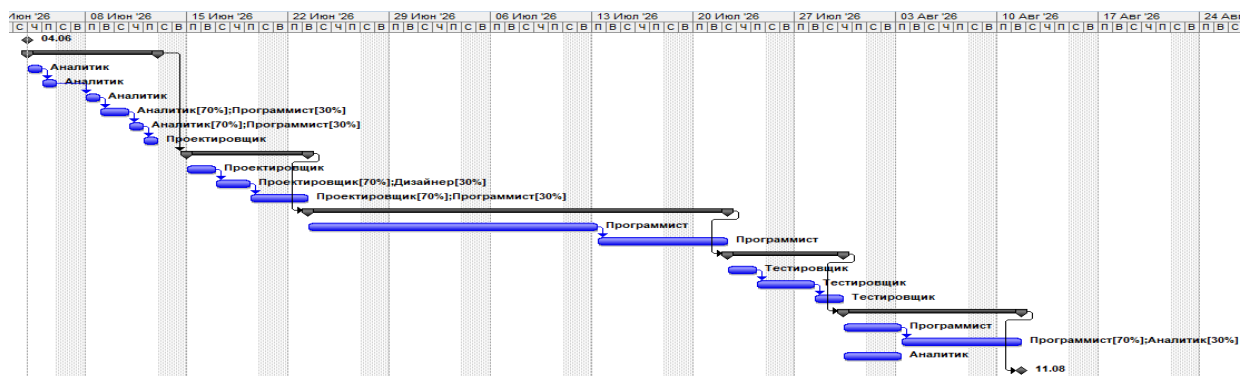


Рисунок 4 – Диаграмма Ганта для проекта

Средствами Microsoft Project 2007 была сформирована статистика проекта (рис. 5).

| Статистика проекта для 'Проект1.mpp'   |              |              |              |
|----------------------------------------|--------------|--------------|--------------|
|                                        | Начало       |              | Окончание    |
| Текущее                                | Чт 04.06.26  |              | Вт 11.08.26  |
| Базовое                                | НД           |              | НД           |
| Фактическое                            | НД           |              | НД           |
| Отклонение                             | 0д           |              | 0д           |
|                                        | Длительность | Трудозатраты | Затраты      |
| Текущие                                | 48,86д?      | 384ч         | 258 080,00 Р |
| Базовые                                | 0д?          | 0ч           | 0,00 Р       |
| Фактические                            | 0д           | 0ч           | 0,00 Р       |
| Оставшиеся                             | 48,86д?      | 384ч         | 258 080,00 Р |
| Процент завершения                     |              |              |              |
| Длительность: 0%      Трудозатраты: 0% |              |              |              |
|                                        |              |              | Заккрыть     |

Рисунок 5 – Статистика проекта

Из представленной статистики видно, что длительность реализации проекта составляет около 49 дней (текущая длительность – 48,86 дня) при плановых денежных затратах в 258 080 рублей.

### 3 Проектирование клиентской части информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях

#### 3.1 Проектирование архитектуры информационной платформы

Информационная платформа трудоустройства и стажировок студентов в ИТ-компаниях проектируется как web-приложение, обеспечивающее взаимодействие студентов, работодателей и преподавателей в едином информационном пространстве. Основное назначение платформы заключается в предоставлении студентам средств поиска вакансий и стажировок, работодателям – средств размещения предложений и обработки откликов, а преподавателям – инструментов сопровождения учебных групп.

Архитектура платформы строится по клиент-серверному принципу. При таком подходе пользователь взаимодействует с системой через клиентскую часть, а обработка данных, хранение информации и выполнение основных бизнес-операций осуществляются серверной частью. В рамках данной выпускной квалификационной работы основное внимание уделяется проектированию клиентской части, поэтому серверная часть рассматривается

как внешний источник данных и набор программных интерфейсов, необходимых для работы web-приложения.

Выбор клиент-серверной архитектуры обусловлен необходимостью разделения ответственности между частями системы. Клиентское приложение отвечает за отображение данных, обработку действий пользователя, маршрутизацию между страницами, проверку форм, управление состояниями интерфейса и отправку запросов к серверу. Серверная часть обеспечивает регистрацию и авторизацию пользователей, хранение данных, обработку вакансий, резюме, откликов, учебных групп и сообщений.

Основным способом взаимодействия клиентской и серверной частей является REST API. Клиентское приложение отправляет HTTP-запросы при выполнении пользователем действий: входе в систему, изменении профиля, просмотре вакансий, создании резюме, отправке отклика или обработке заявки. После выполнения операции серверная часть возвращает ответ, который используется клиентским приложением для обновления пользовательского интерфейса.

Для обмена сообщениями между студентами и работодателями предусматривается механизм взаимодействия в режиме реального времени. В клиентской части для этого используется подключение к серверному хабу сообщений. Такой подход позволяет получать новые сообщения без ручного обновления страницы и обеспечивает более удобную коммуникацию между участниками платформы.

В составе информационной платформы выделены следующие основные компоненты:

- клиентское приложение;
- серверное приложение;
- база данных;
- модуль авторизации;
- модуль работы с вакансиями и резюме;
- модуль обработки откликов;

- модуль учебных групп;
- модуль обмена сообщениями.

На рисунке 6 представлена диаграмма компонентов информационной платформы.

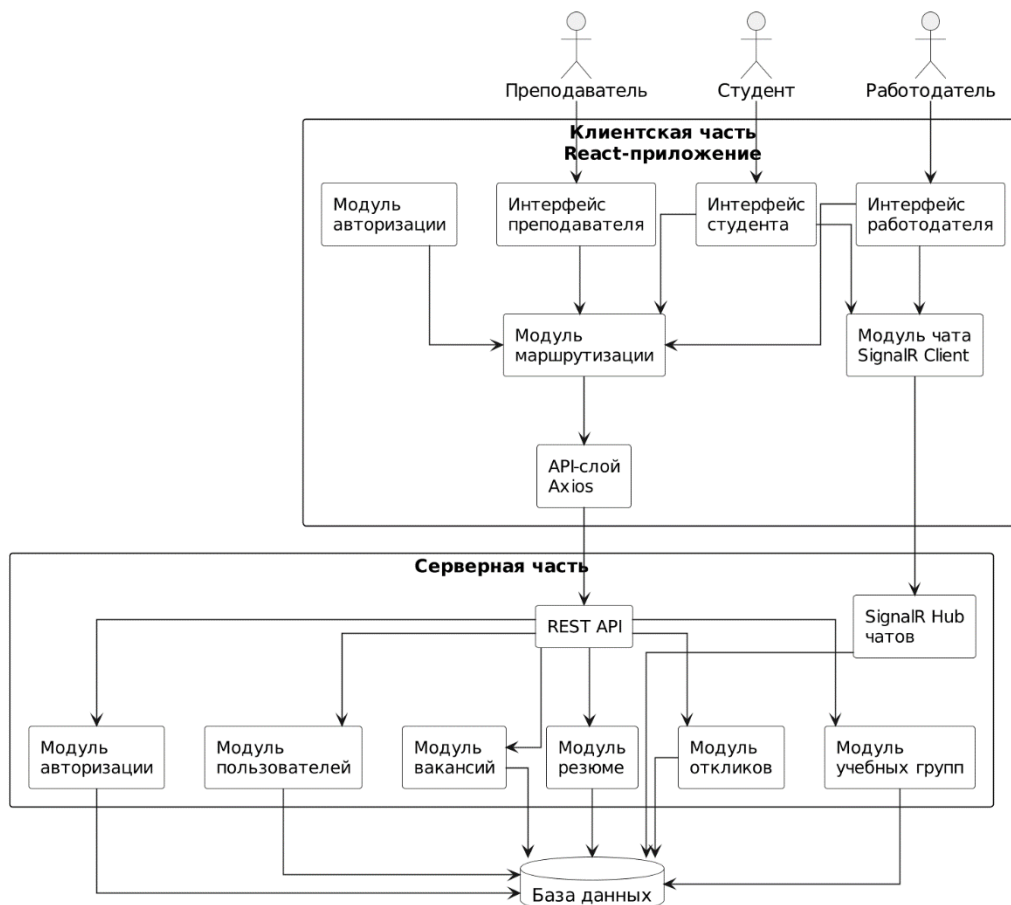


Рисунок 6 – Диаграмма компонентов информационной платформы

Клиентское приложение является основным объектом проектирования в рамках данной работы. Оно предоставляет пользователям интерфейсы для регистрации, авторизации, просмотра и изменения профилей, работы с вакансиями, резюме, откликами, учебными группами и сообщениями. Клиентская часть не обращается к базе данных напрямую, а получает и передает данные только через серверное API.

Серверная часть является внешним по отношению к клиентскому приложению компонентом. Она принимает запросы от клиента, выполняет необходимые проверки, обращается к базе данных и возвращает результат

выполнения операции. Подробная внутренняя реализация серверной части в данной работе не рассматривается, однако ее интерфейсы учитываются при проектировании клиентской части.

Проектируемая архитектура позволяет разделить пользовательский интерфейс и обработку данных. Это обеспечивает независимое развитие клиентского приложения, упрощает сопровождение frontend-кода и позволяет организовать единый механизм взаимодействия пользователя с основными функциями платформы.

### **3.2 Проектирование интерфейса пользователя информационной платформы трудоустройства и стажировок студентов**

Клиентская часть информационной платформы должна обеспечивать пользователям удобный и понятный доступ к функциональным возможностям системы через web-интерфейс. При проектировании интерфейса необходимо учитывать, что платформа предназначена для нескольких категорий пользователей. В системе предусмотрены роли студента, работодателя и преподавателя, для каждой из которых требуется отдельный набор страниц, форм и действий. Для студента приоритетными являются вакансии, резюме и отклики, для работодателя – управление компанией и вакансиями, а для преподавателя – работа с учебными группами. Кроме того, ролевое разделение интерфейса упрощает навигацию, поскольку пользователь видит только те разделы, которые относятся к его задачам.

При проектировании пользовательского интерфейса учитывались принципы модульности и повторного использования элементов. Общие элементы приложения применяются в разных разделах приложения. Это позволяет обеспечить единообразие визуального оформления и снизить объем повторяющегося кода при дальнейшей реализации. Кроме того, единый подход к построению интерфейса упрощает сопровождение клиентской части и делает поведение приложения более предсказуемым для пользователя. При изменении или расширении функциональности отдельные компоненты могут быть

доработаны без полной переработки связанных страниц. Повторное использование компонентов также позволяет быстрее добавлять новые разделы приложения, сохраняя общий стиль и логику взаимодействия. При проектировании учитываются состояния загрузки, ошибки, успешного выполнения действий и отсутствия данных, так как они напрямую влияют на удобство работы пользователя.

Интерфейс клиентского приложения включает следующие основные группы страниц:

- публичные страницы, доступные без авторизации;
- страницы регистрации и авторизации пользователей;
- страницы студента, предназначенные для работы с профилем, вакансиями, резюме, откликами, учебной группой и чатами;
- страницы работодателя, предназначенные для управления профилем компании, вакансиями, откликами и коммуникацией со студентами;
- страницы преподавателя, предназначенные для управления учебными группами и заявками студентов;
- страницы обмена сообщениями между студентом и работодателем.

На рисунке 7 представлена общая структура клиентской части приложения.

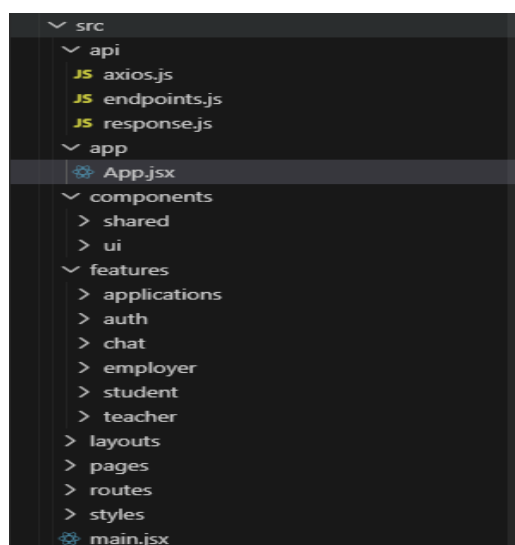


Рисунок 7 – Общая структура клиентской части приложения

Представленная структура отражает разделение клиентского приложения на страницы, функциональные модули, общие компоненты, маршруты и API-слой.

Публичная часть интерфейса предназначена для доступа пользователя к системе. Она включает главную страницу, страницу выбора роли, форму авторизации и формы регистрации. При проектировании форм регистрации и авторизации необходимо предусмотреть ввод учетных данных, проверку обязательных полей, обработку ошибок и переход пользователя к соответствующему разделу после успешного входа.

На рисунке 8 представлен интерфейс формы авторизации пользователя.

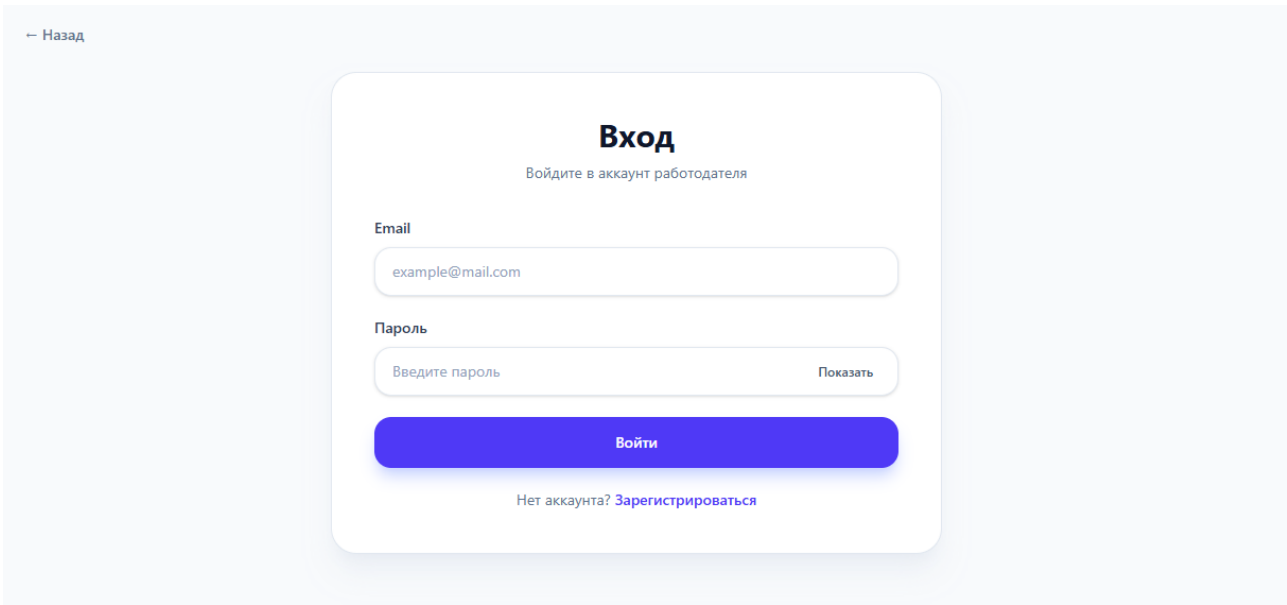


Рисунок 8 – Форма авторизации пользователя

Форма авторизации содержит поля для ввода электронной почты и пароля, кнопку входа в систему. После нажатия кнопки «войти», клиентская часть формирует запрос на сервер, содержащий введенные пользователем данные, после чего сервер формирует ответ, который отправляет и клиенту. При корректно введенных данных, пользователь авторизируется в системе и получает доступ к основным функциям, присущем его роли. Если введенные данные не корректны или пользователя с такими данными — нет, то он получает соответствующее сообщение об ошибке.

На рисунке 9 представлен пример формы регистрации пользователя.

Рисунок 9 – Пример формы регистрации пользователя

Форма регистрации проектируется с учетом выбранной роли.

В личном кабинете студент получает доступ к персональным данным, учебной группе, резюме, списку вакансий, откликам и чатам.

На рисунке 10 представлен интерфейс личного кабинета студента.

Рисунок 10 – Личный кабинет студента

Личный кабинет студента объединяет основную информацию о пользователе и предоставляет доступ к ключевым разделам платформы. Через данный раздел студент может перейти к редактированию профиля, просмотру вакансий, управлению резюме, истории откликов, информации об учебной группе и сообщениям.

Страница просмотра вакансий предназначена для отображения предложений, опубликованных работодателями. При проектировании данной страницы предусматриваются поиск, фильтрация, отображение краткой информации о вакансии и переход к подробному описанию.

На рисунке 11 представлена страница просмотра вакансий.

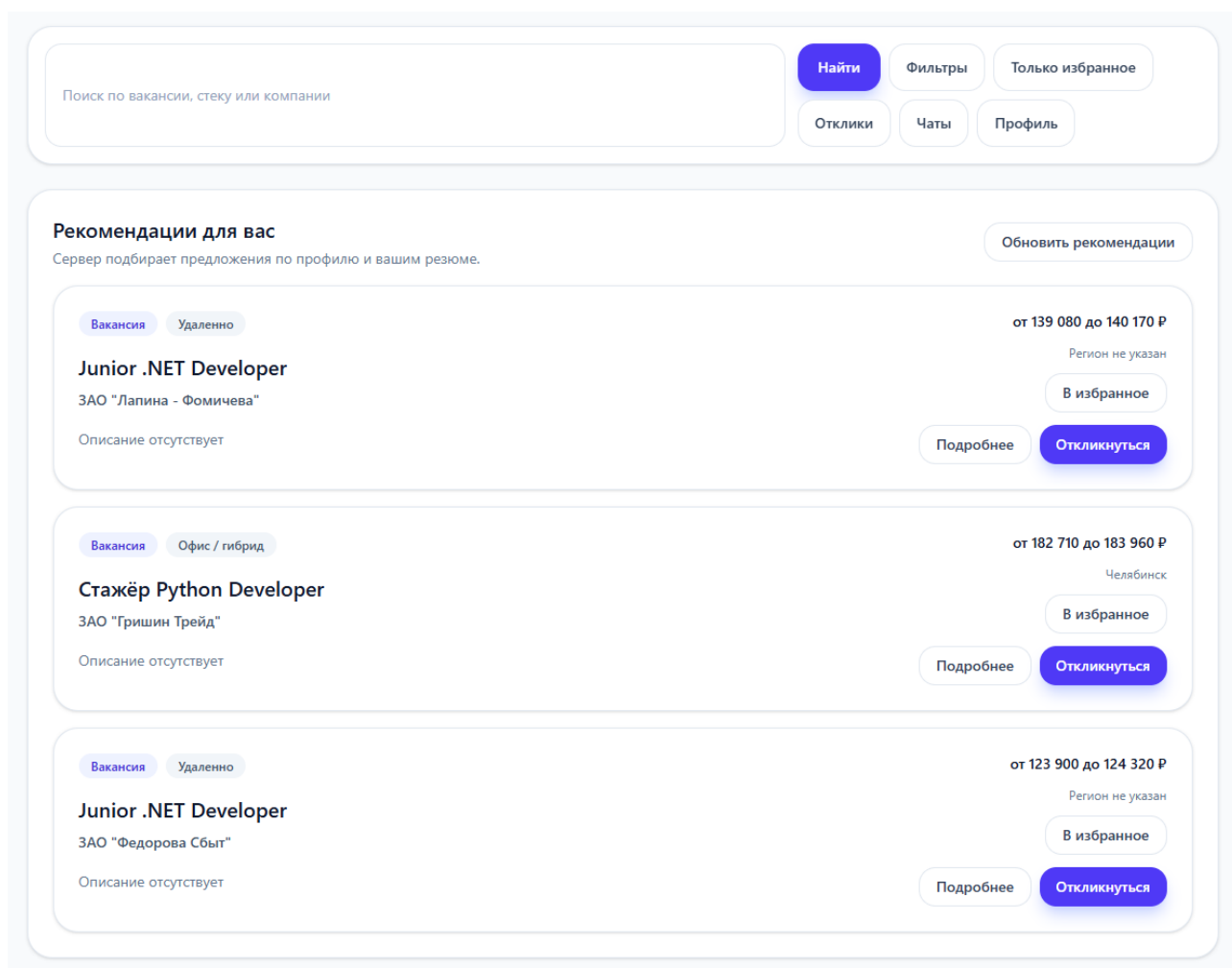


Рисунок 11 – Страница просмотра вакансий

Раздел управления резюме предназначен для создания, редактирования, копирования и удаления резюме студента.

На рисунке 12 представлен раздел управления резюме студента.

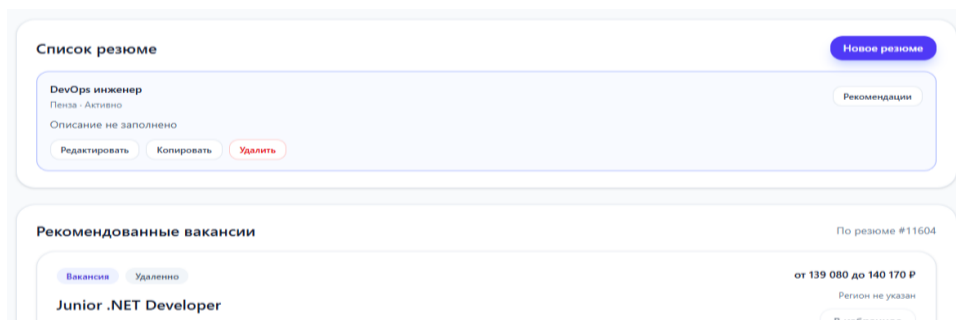


Рисунок 12 – Раздел управления резюме студента

Интерфейс работодателя проектируется с учетом задач размещения вакансий и обработки откликов студентов.

На рисунке 13 представлен интерфейс личного кабинета работодателя.

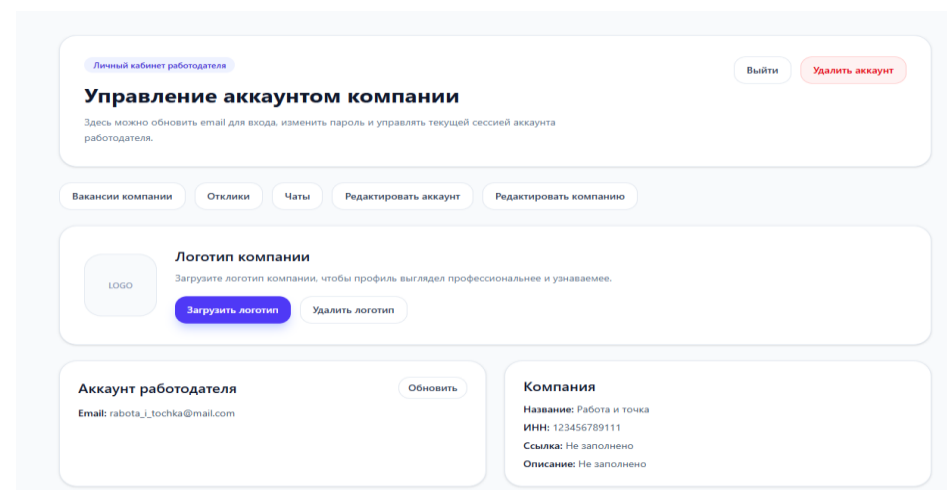


Рисунок 13 – Личный кабинет работодателя

Форма создания вакансии позволяет работодателю добавить новую вакансию в систему. В форме указывается название и описание должности, требования, предъявляемые кандидату, условия труда, специализацию, формат работы, необходимый технологический стек и другие параметры. После нажатия кнопки «Сохранить», клиентская часть формирует запрос на сервер для добавления новой вакансии в систему. При правильно заполненных данных, вакансия появляется в списке вакансий и в рекомендациях у студентов. Если данные были заполнены неверно или вовсе отсутствуют, клиент получит соответствующее сообщение об ошибке.

На рисунке 14 представлена форма создания вакансии.

Рисунок 14 – Форма создания вакансии

Страница откликов работодателя предназначена для просмотра заявок студентов на опубликованные вакансии

На рисунке 15 представлена страница просмотра откликов на вакансию.

Рисунок 15 – Страница просмотра откликов на вакансию

Интерфейс преподавателя предназначен для сопровождения учебных групп. Преподаватель должен иметь возможность создавать группы, просматривать их состав, обновлять код приглашения и обрабатывать заявки студентов на вступление

На рисунке 16 представлен личный кабинет преподавателя.

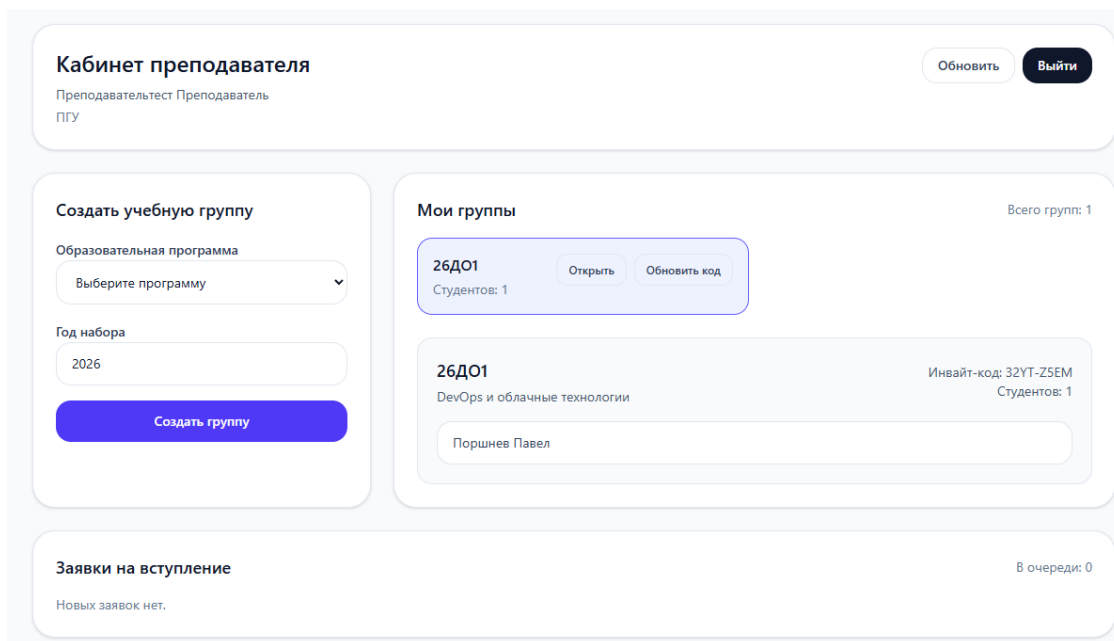


Рисунок 16 – Личный кабинет преподавателя

Раздел обмена сообщениями предназначен для коммуникации между студентом и работодателем.

На рисунке 17 представлен интерфейс обмена сообщениями.

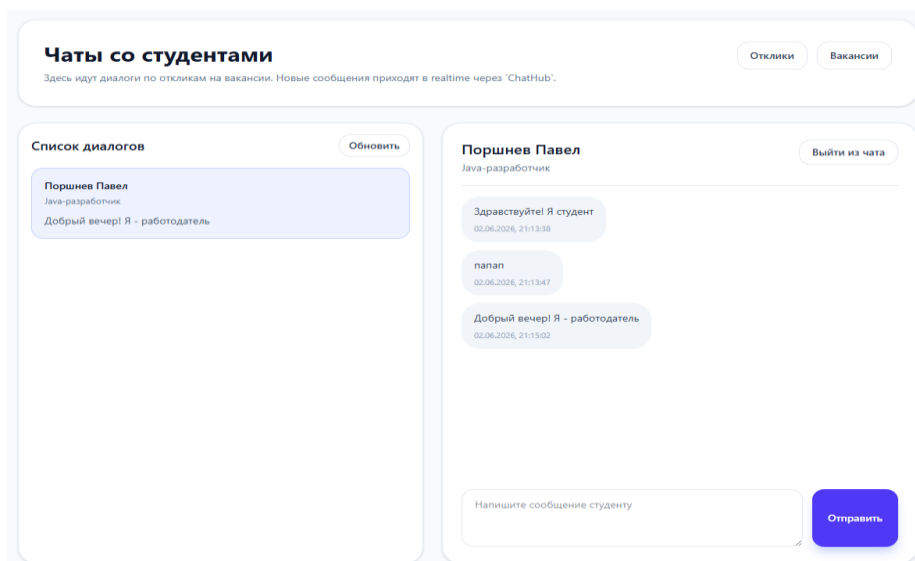


Рисунок 17 – Интерфейс чата студента и работодателя

Проектирование интерфейса пользователя направлено на создание единой и понятной системы экранов, адаптированной под разные роли. Разделение интерфейса по пользовательским сценариям позволяет снизить сложность работы с платформой и обеспечить доступ каждого пользователя только к тем функциям, которые соответствуют его задачам.

### 3.3 Проектирование алгоритма регистрации студента и отображения вакансий и резюме

В рамках данного раздела рассматриваются алгоритмы регистрации студента и отображения вакансий и резюме.

Алгоритм регистрации студента начинается с выбора соответствующей роли. После этого пользователь переходит к форме, где вводит персональные данные. На стороне клиентской части выполняется первичная проверка заполнения обязательных полей, корректности электронной почты и пароля. Это позволяет обнаружить часть ошибок до отправки запроса на сервер.

На рисунке 18 представлена схема алгоритма регистрации студента.

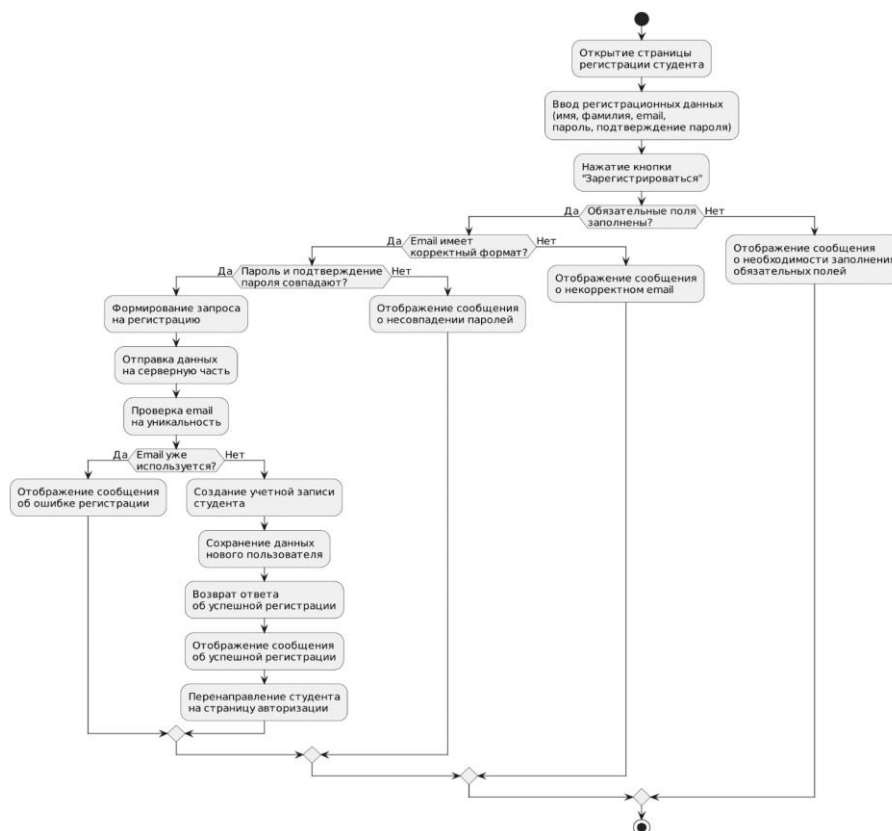


Рисунок 18 – Алгоритм регистрации студента

Проектирование данного алгоритма обеспечивает единый порядок обработки регистрационных данных. Первичная проверка на стороне клиента повышает удобство пользователя, а окончательная проверка на сервере обеспечивает корректность и безопасность создания учетной записи.

Алгоритм отображения вакансий предусматривает загрузку данных с серверной части, их фильтрацию и представление в удобном виде. При открытии страницы клиентское приложение формирует запрос на получение вакансий с учетом параметров поиска: текстового запроса, региона, специализации, формата работы и минимальной заработной платы.

После получения ответа данные преобразуются к формату карточек интерфейса. Каждая карточка содержит сведения о вакансии и доступные действия: просмотр информации, добавление в избранное и отправку отклика. Если подходящие вакансии не найдены, пользователю отображается соответствующее сообщение

На рисунке 19 представлена схема алгоритма отображения вакансий.

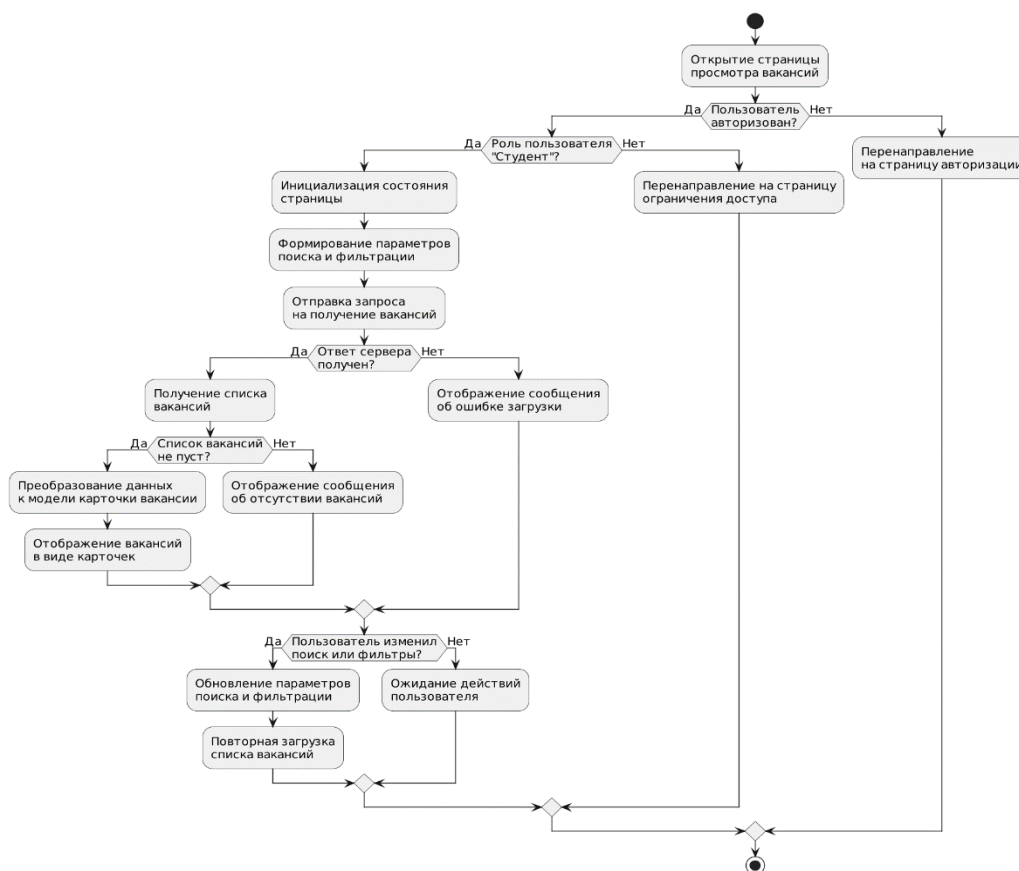


Рисунок 19 – Алгоритм отображения вакансий

Проектирование отображения резюме выполняется аналогичным образом. При переходе в раздел резюме клиентская часть запрашивает список резюме студента. Полученные данные приводятся к формату, удобному для представления в интерфейсе.

При выборе вакансии и нажатии кнопки отклика клиентская часть должна предоставить студенту возможность выбрать одно из созданных резюме. После выбора резюме данные передаются серверной части для создания отклика. Если у студента нет резюме, интерфейс отображает соответствующее сообщение и предлагает перейти к созданию резюме. Такой подход позволяет связать раздел резюме с процессом отклика на вакансию и обеспечить последовательность действий пользователя.

На рисунке 20 представлена схема алгоритма отображения резюме студента.

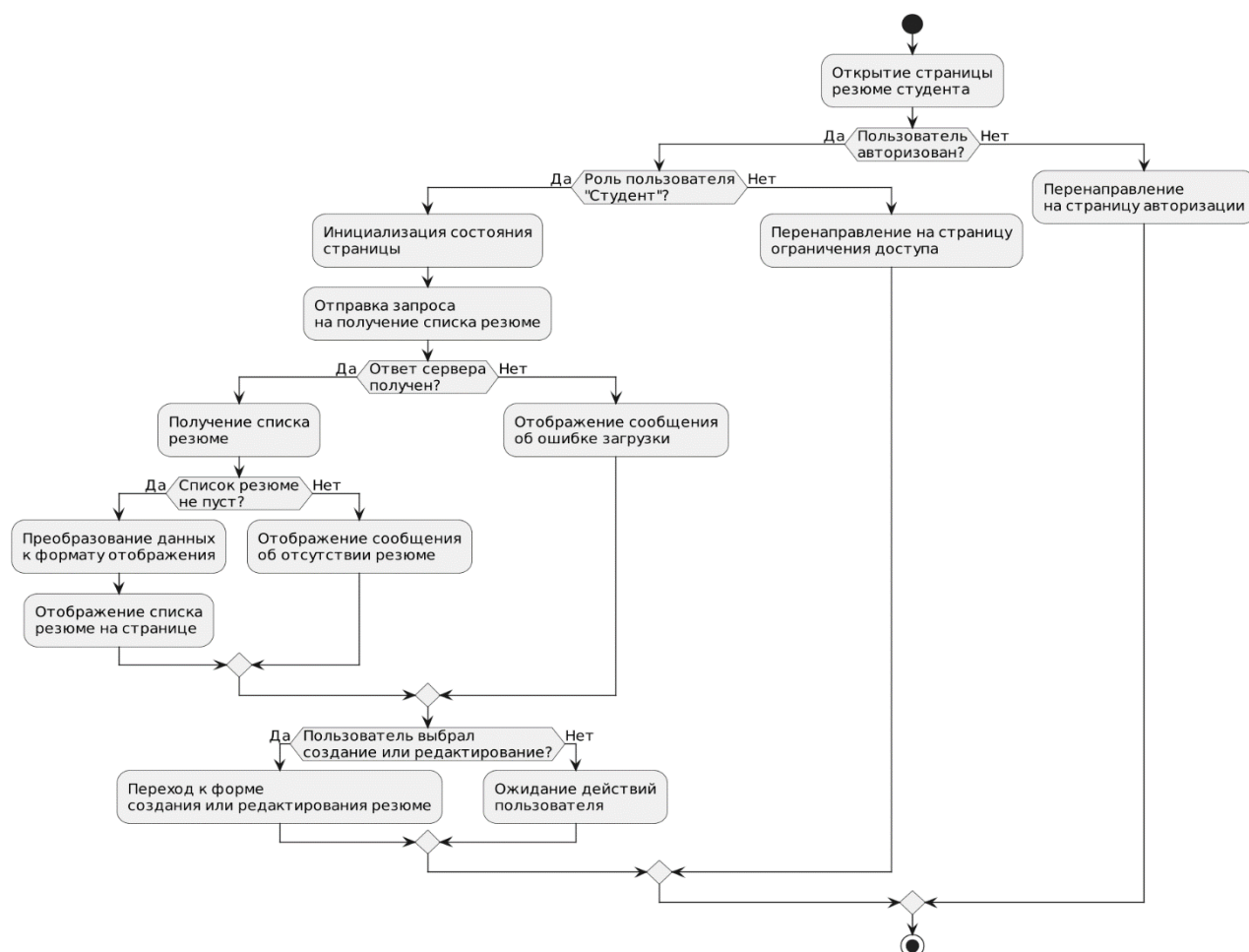


Рисунок 20 – Алгоритм отображения резюме студента

Проектирование алгоритмов регистрации, отображения вакансий и отображения резюме позволяет заранее определить порядок действий пользователя и реакции клиентского приложения. Это обеспечивает согласованность интерфейса, упрощает дальнейшую реализацию компонентов и снижает вероятность ошибок при выполнении основных пользовательских сценариев.

### **3.4 Проектирование базы данных и способа доступа к ней**

Хотя тема выпускной квалификационной работы связана преимущественно с клиентской частью информационной платформы, при проектировании web-интерфейса необходимо учитывать структуру данных, с которыми работает приложение. Клиентская часть отображает сведения, полученные от серверной части, поэтому состав основных сущностей предметной области влияет на структуру экранов, форм, карточек и API-запросов.

База данных проектируется как централизованное хранилище информации платформы. В ней должны храниться данные пользователей, их ролей, профилей студентов и работодателей, компаний, вакансий, резюме, откликов, учебных групп, заявок на вступление в группы, чатов и сообщений. Подробная физическая реализация базы данных относится к серверной части и в рамках данной работы не рассматривается детально. Однако для проектирования клиентской части важно определить, какие данные должны отображаться и передаваться через API.

К основным информационным объектам, используемым клиентской частью, относятся:

- пользователь, содержащий учетные данные и роль в системе;
- профиль студента, включающий персональные сведения, контакты, информацию об образовании и специализации;
- профиль работодателя, связанный с учетной записью представителя компании;

The diagram illustrates the database structure for a university career portal. It features several main entities and their relationships:

- Образовательная программа** (Educational Program): Attributes include Name, Program Code, and Duration in years.
- Учебная группа** (Study Group): Attributes include Name, Admission Year, Graduation Year, and Invitation Code.
- Учитель** (Teacher): Attributes include Name, Surname, Patronymic, Phone, VK Link, Telegram Link, Max Link, and Profile Photo.
- Студент** (Student): Attributes include Name, Surname, Patronymic, Birth Date, Phone, VK Link, Telegram Link, Max Link, GitHub Link, and Profile Photo.
- Компания** (Company): Attributes include Name, INN, Website Link, Description, and Logo.
- Вакансия** (Vacancy): Attributes include Title, Description, Salary Range, Remote Work Option, Region, Availability for View, and Required Experience.
- Предложение по практике** (Practice Offer): Attributes include Title, Description, Remote Work Option, Region, Availability for View, and Number of Places.
- Резюме** (Resume): Attributes include Description, Availability for View, Desired Salary, Region, and Last Update Date.
- Опыт работы** (Work Experience): Attributes include Company Name, Profession, Start Date, End Date, and Description.

Key relationships and cardinalities:

- Учитель - Студент**: One-to-many relationship.
- Студент - Компания**: One-to-many relationship.
- Компания - Вакансия**: One-to-many relationship.
- Вакансия - Предложение по практике**: One-to-many relationship.
- Предложение по практике - Резюме**: One-to-many relationship.
- Резюме - Опыт работы**: One-to-many relationship.

Other entities and relationships include:

- Период практики** (Practice Period): Attributes include Course Number, Start Date, Start Date of Practice, and End Date of Practice.
- Отчетность по практике** (Practice Report): Attributes include Path to Report File, Path to Solution Archive, Solution Link, Last Update Time, View Time, and Rating.
- Заявка на практику** (Practice Application): Attribute is Creation Date.
- Пользователь** (User): Attributes include Email and Password.
- Роль** (Role): Attribute is Name.
- Работодатель** (Employer): Attribute is Name.
- Избранная вакансия** (Selected Vacancy): Attribute is View Date.
- Просмотр вакансии** (Vacancy View): Attribute is View Date.
- Сообщение** (Message): Attributes include Content and Send Date.
- Чат** (Chat): Attribute is Availability for Send.
- Статус отклика** (Response Status): Attribute is Name.
- Отклик на вакансию** (Vacancy Response): Attribute is Status Update Date.
- Навык** (Skill): Attribute is Name.
- Навык в резюме** (Skill in Resume): Attribute is Name.
- Специализация** (Specialization): Attribute is Name.

Рисунок 21 – Концептуальная модель данных информационной платформы

С точки зрения клиентской части база данных является недоступной напрямую. Клиентское приложение не выполняет SQL-запросы и не управляет соединением с базой данных. Все операции с данными выполняются только через серверное API. Такой подход обеспечивает безопасность, так как клиент не получает доступа к внутренней структуре хранения данных и не может напрямую изменять записи в базе.

Способ доступа клиентской части к данным строится через набор серверных методов. Например, для получения списка вакансий клиентская часть отправляет запрос к методу работы с вакансиями, для получения резюме – к методу работы с резюме, для изменения профиля компании – к методу управления компанией. Серверная часть принимает запрос, обращается к базе данных и возвращает клиенту результат в формате, пригодном для отображения.

На стороне клиентского приложения для работы с полученными данными используются модели отображения и преобразователи данных. Преобразователи данных позволяют приводить сведения, полученные от серверной части, к структуре, удобной для конкретного компонента интерфейса. Например, серверный объект вакансии может содержать множество технических полей, тогда как карточке вакансии требуются только название, компания, описание, формат работы, регион и информация о заработной плате.

При проектировании форм необходимо учитывать обязательность отдельных полей и ограничения данных. Например, при создании вакансии работодатель должен указать название и описание предложения, при создании резюме студент должен заполнить основные сведения о себе и навыках, а при регистрации требуется корректный адрес электронной почты и пароль. Первичная проверка данных выполняется на стороне клиента, однако окончательное сохранение и проверка целостности выполняются серверной частью.

Проектирование базы данных в рамках данной работы рассматривается с позиции клиентского приложения: определяется состав данных, необходимых для отображения и ввода пользователем, а также способ доступа к этим данным

через серверное API. Такой подход позволяет сохранить акцент на клиентской части и при этом обеспечить согласованность интерфейса с общей структурой информационной платформы.

### 3.5 Проектирование взаимодействия клиентской части информационной платформы с серверной частью

Взаимодействие клиентской части с серверной частью проектируется как отдельный уровень клиентского приложения. Его назначение заключается в организации обмена данными между клиентом и сервером. Выделение данного уровня позволяет не размещать сетевые запросы непосредственно в визуальных компонентах, а использовать специальные API-модули для каждой предметной области.

Для взаимодействия клиентской и серверной части приложения предусмотрен API-слой. Компоненты интерфейса вызывают функции API-слоя, передают им необходимые параметры и получают результат для отображения. снижает связанность интерфейса с серверными адресами и упрощает сопровождение приложения.

На рисунке 22 представлена схема взаимодействия клиентской части с серверной частью.

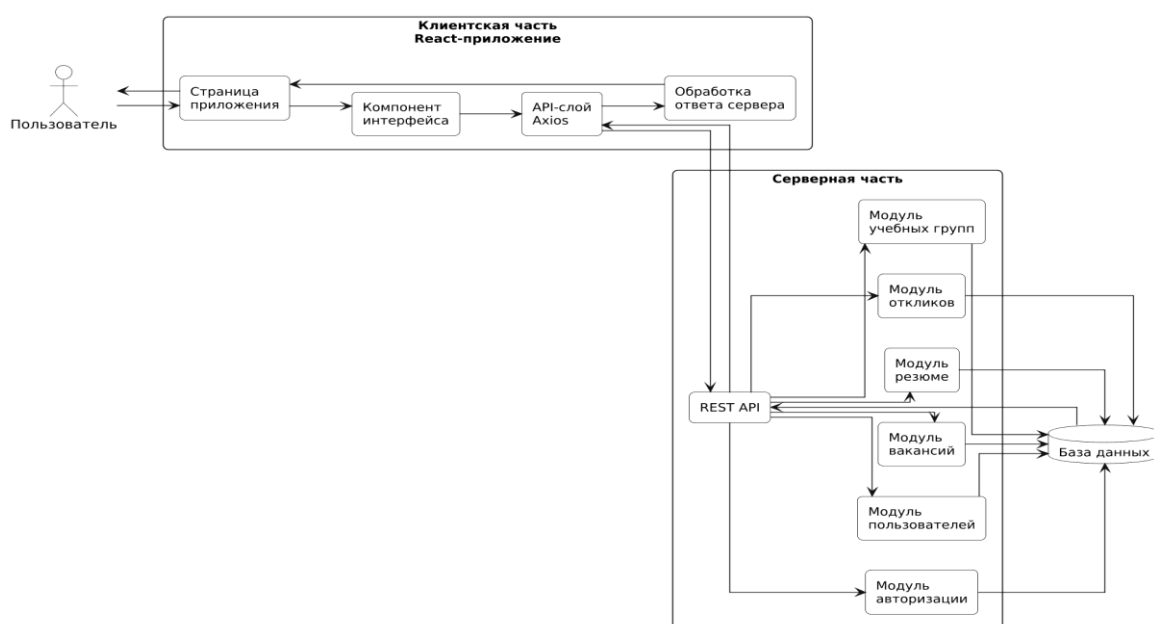


Рисунок 22 – Схема взаимодействия клиентской части с серверной частью

Сценарий взаимодействия начинается с действия пользователя. Например, пользователь отправляет форму авторизации. Компонент интерфейса передает данные в соответствующую функцию API-слоя. API-слой формирует HTTP-ответ и отправляет его на серверную часть.

После обработки запроса серверная часть возвращает HTTP-ответ. Клиентское приложение анализирует полученные данные и обновляет состояние интерфейса. При успешном выполнении операции пользователь видит актуальные данные или сообщение об успешном сохранении. При возникновении ошибки интерфейс должен вывести понятное сообщение и не допустить потери введенных данных.

Для защищенных запросов предусматривается передача данных авторизации. После входа в систему клиентская часть получает сведения о пользовательской сессии и использует их при последующих запросах. При обращении к защищенным маршрутам выполняется проверка авторизации и роли пользователя. Если пользователь не авторизован, он перенаправляется на страницу входа. Если роль пользователя не соответствует запрошенному разделу, доступ к странице ограничивается.

На рисунке 23 представлена схема маршрутизации клиентского приложения.

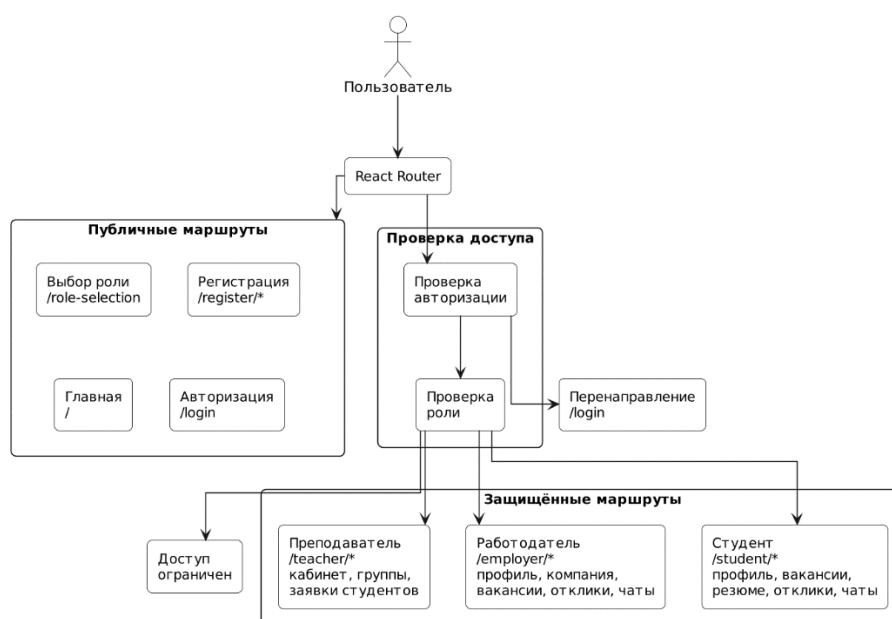


Рисунок 23 – Схема маршрутизации клиентского приложения

Схема маршрутизации показывает разделение страниц приложения на публичные и защищенные маршруты. Публичные маршруты доступны всем пользователям и включают главную страницу, выбор роли, регистрацию и авторизацию. Защищенные маршруты открываются только после проверки пользовательской сессии и роли. В зависимости от роли пользователь получает доступ к разделу студента, работодателя или преподавателя.

Для обмена сообщениями между студентами и работодателями проектируется канал реального времени. Клиентская часть подключается к серверному хабу сообщений, что позволяет отправлять сообщения и получать обновления выбранного диалога без постоянного опроса сервера.

Взаимодействие с серверной частью должно учитывать состояния интерфейса: загрузку, успешное выполнение операции, ошибку запроса, отсутствие результатов и ограничение доступа. Для каждого состояния предусматривается визуальная реакция: индикатор загрузки, сообщение об ошибке, пустой список, уведомление или перенаправление пользователя.

На рисунке 24 представлена схема взаимодействия клиента с серверным хабом сообщений.

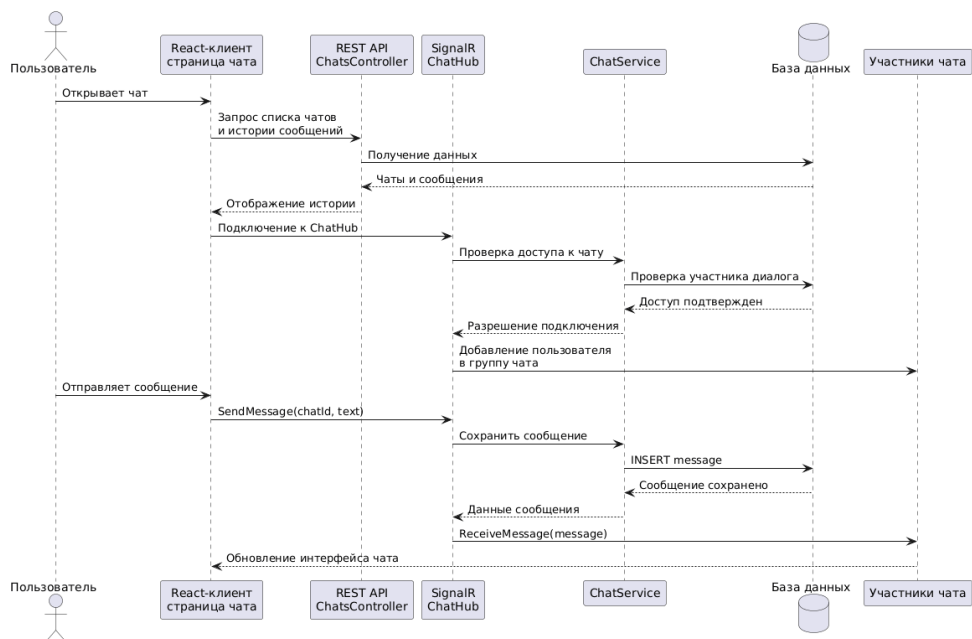


Рисунок 24 – Схема взаимодействия клиентской части с серверным хабом сообщений

Таким образом, взаимодействие клиентской части с серверной частью проектируется как совокупность REST-запросов, защищенной маршрутизации, API-слоя и real-time соединения для чатов. Такое построение позволяет отделить визуальные компоненты от сетевой логики, обеспечить корректную обработку пользовательских действий и создать основу для дальнейшей реализации клиентской части информационной платформы.

## 4 Реализация информационной платформы трудоустройства и стажировки студентов

### 4.1 Реализация клиентской части

Для разработки клиентской части информационной платформы трудоустройства и стажировки студентов в ИТ-компаниях была выбрана среда разработки кода Webstorm 2025, на языке программирования JavaScript с использованием библиотеки ReactJS. Код клиентской части приложения приведен в приложении А.

На рисунке 25 представлена диаграмма компонентов клиентской части приложения.

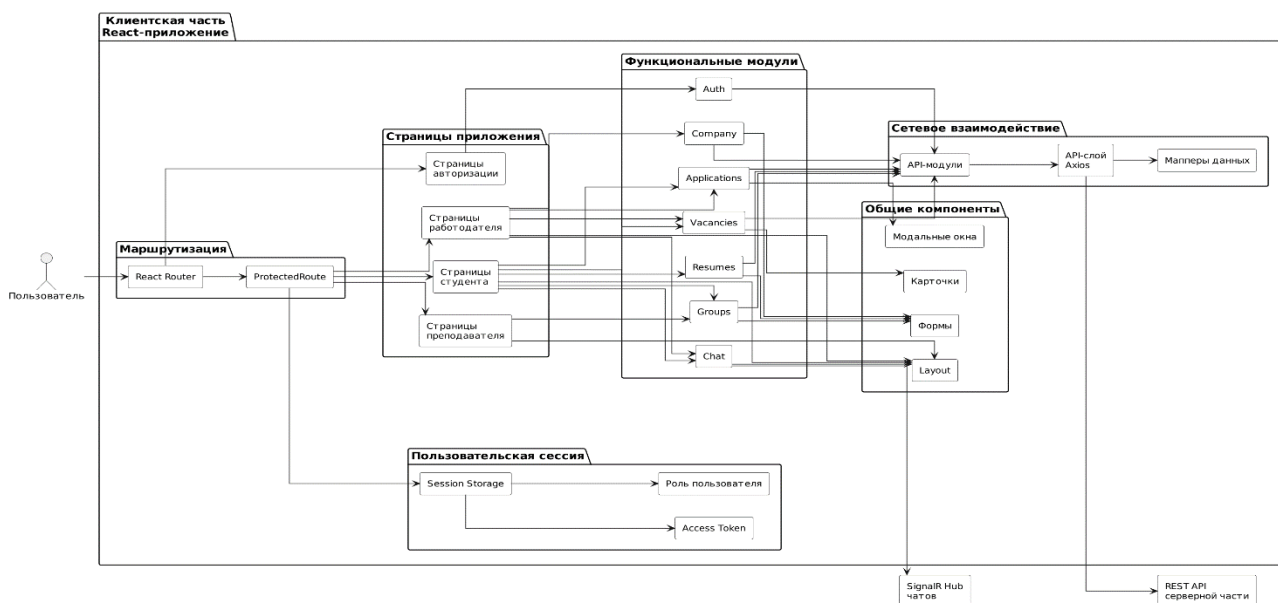


Рисунок 25 – Диаграмма компонентов клиентской части приложения

На рисунке 25 представлена диаграмма компонентов клиентской части приложения. Клиентская часть реализована в виде React-приложения и включает несколько групп компонентов: маршрутизацию, страницы приложения, функциональные модули, общие элементы интерфейса, API-слой и механизм хранения пользовательской сессии. Такое разделение соответствует модульному подходу к построению web-приложения и позволяет изолировать различные части клиентской логики.

## 4.2 Реализация функции отображения вакансий

Для отображения вакансий в клиентской части информационной платформы используется страница `StudentVacanciesPage`, предназначенная для просмотра студентом доступных предложений от работодателей. Данный компонент обеспечивает загрузку вакансий с серверной части, применение поискового запроса и фильтров, отображение рекомендованных вакансий, добавление вакансий в избранное, а также переход к детальной информации и отправку отклика.

Загрузка данных выполняется при открытии страницы. Для этого используется функция `loadData`, которая обращается к API-слою клиентского приложения. Одновременно запрашиваются список вакансий, рекомендованные вакансии, список специализаций и резюме пользователя. После получения ответа данные преобразуются к модели, удобной для отображения в интерфейсе.

В листинге 1 приведена реализация функции загрузки списка вакансий.

### Листинг 1 – Загрузка и подготовка списка вакансий

```
const loadData = async (withRefreshingState = false) => {
  try {
    if (withRefreshingState) {
      setIsRefreshing(true);
    } else {
      setIsLoading(true);
    }

    setError("");

    const [vacancyList, recommendedList, specializationList,
resumeList] =
```

```

        await Promise.all([
            getVacancies({
                search: searchValue,
                searchInTitle: true,
                searchInDescription: true,
                searchInCompanyName: true,
                isRemote:
                    filters.isRemote === ""
                    ? undefined
                    : filters.isRemote === "true",
                region: filters.region,
                salaryFrom: filters.salaryFrom || undefined,
                specializationId: filters.specializationId ||
undefined,
                pageIndex: 1,
                pageSize: 30,
            }),
            getRecommendedVacancies(1, 6),
            getSpecializations(),
            getMyResumes(),
        ]);

        setVacancies(vacancyList.map(mapVacancyToCardModel));

        setRecommendedVacancies(recommendedList.map(mapVacancyToCardModel)
);
        setSpecializations(specializationList);
        setResumes(resumeList.map(mapResumeToListModel));
    } catch (loadError) {
        setError(
            loadError?.response?.data?.message ||
            "Не удалось загрузить вакансии. Проверьте
авторизацию и попробуйте снова."
        );
    } finally {
        setIsLoading(false);
        setIsRefreshing(false);
    }
};

```

Код использует асинхронную загрузку данных. Для одновременного выполнения нескольких запросов применяется `Promise.all`, что позволяет сократить время ожидания пользователя. В функцию `getVacancies` передаются параметры поиска и фильтрации: текстовый запрос, формат работы, регион, минимальная заработная плата, специализация и параметры постраничной загрузки. После получения данных каждая вакансия преобразуется функцией `mapVacancyToCardModel`, что позволяет привести серверный ответ к единому формату, используемому компонентами интерфейса.

Для отображения одной вакансии используется отдельный компонент `VacancyCard`. Он принимает объект вакансии и функции-обработчики для просмотра подробной информации, отправки отклика и добавления вакансии в избранное. Фрагмент реализации карточки вакансии приведён в листинге 2.

## Листинг 2 – Компонент отображения карточки вакансии

```
const VacancyCard = ({
  vacancy,
  onView,
  onApply,
  onToggleFavorite,
  isFavoriteLoading = false,
}) => {
  return (
    <article className="rounded-3xl border border-slate-200
bg-white p-6 shadow-sm transition hover:-translate-y-0.5
hover:shadow-md">
      <div className="flex flex-col gap-4 lg:flex-row
lg:items-start lg:justify-between">
        <div className="min-w-0">
          <div className="flex flex-wrap items-center
gap-2">
            <span className="rounded-full bg-indigo-50
px-3 py-1 text-xs font-medium text-indigo-700">
              {vacancy.type}
            </span>
            <span className="rounded-full bg-slate-100
px-3 py-1 text-xs font-medium text-slate-600">
              {vacancy.workFormat}
            </span>
          </div>
          <h3 className="mt-4 text-xl font-semibold
text-slate-900">
            {vacancy.title}
          </h3>
          <p className="mt-2 text-sm font-medium text-
slate-600">
            {vacancy.company}
          </p>
          <p className="mt-4 text-sm leading-6 text-
slate-500">
            {vacancy.description}
          </p>
        </div>
      </div>
    </article>
  )
}
```

```

        <div className="flex shrink-0 flex-col gap-3
lg:items-end">
            <p className="text-sm font-semibold text-
slate-900">
                {vacancy.salary}
            </p>
            <p className="text-xs text-slate-500">
                {vacancy.location}
            </p>

            <button
                type="button"
                onClick={() =>
onToggleFavorite?.(vacancy) }
                disabled={!onToggleFavorite ||
isFavoriteLoading}
            >
                {vacancy.isFavorite ? "В избранном" : "В
избранное"}
            </button>

            <button type="button" onClick={() =>
onView?.(vacancy) }>
                Подробнее
            </button>

            <button type="button" onClick={() =>
onApply?.(vacancy) }>
                Откликнуться
            </button>
        </div>
    </div>
</article>
);
};

```

Компонент `VacancyCard` отвечает только за визуальное представление вакансии и вызов переданных обработчиков. В карточке отображаются название вакансии, компания, описание, формат работы, заработная плата, регион и кнопки пользовательских действий. Такой подход соответствует компонентной архитектуре React, так как логика загрузки данных находится на уровне страницы, а отображение отдельной вакансии вынесено в самостоятельный компонент [16].

Дополнительно на странице реализованы поиск и фильтрация вакансий. Пользователь может выполнить поиск по названию вакансии, стеку технологий, описанию или названию компании. Также предусмотрена фильтрация по

формату работы, региону, специализации и минимальной заработной плате. Для отображения только сохранённых предложений используется отдельный переключатель избранного.

На рисунке 26 представлено отображение списка вакансий в интерфейсе студента.

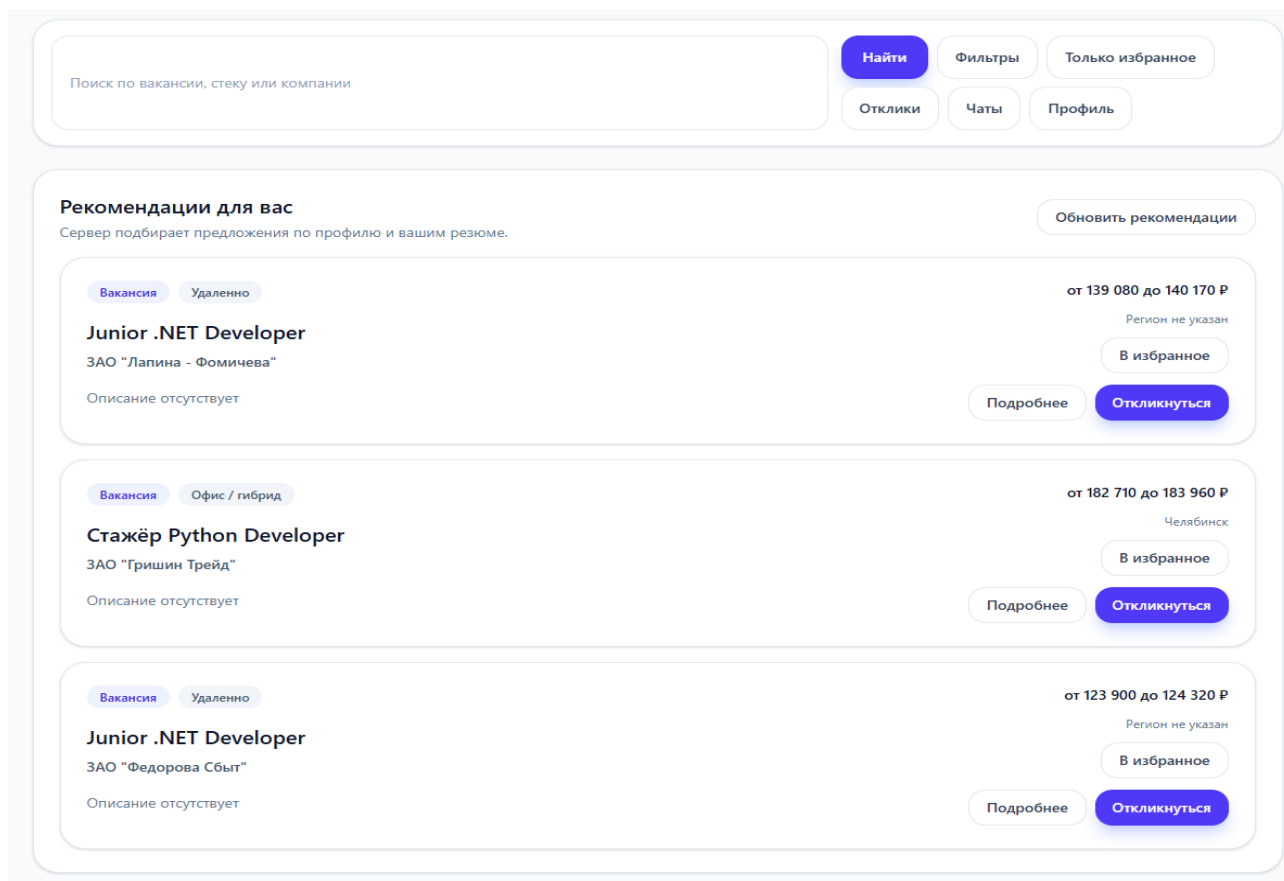


Рисунок 26 – Отображение списка вакансий

На рисунке видно, что список вакансий представлен в виде отдельных карточек. Каждая карточка содержит основную информацию о предложении и набор действий, доступных студенту. Пользователь может открыть подробное описание вакансии, добавить её в избранное или отправить отклик. В верхней части страницы расположены элементы поиска, фильтрации и перехода к связанным разделам, что обеспечивает удобную навигацию при работе с вакансиями.

### 4.3 Реализация управления профилем компании

Для управления профилем компании в клиентской части информационной платформы используется интерфейс работодателя. Данный раздел предназначен для просмотра и изменения сведений об организации, от имени которой работодатель размещает вакансии и взаимодействует со студентами. Наличие профиля компании позволяет студентам получать дополнительную информацию о работодателе перед отправкой отклика на вакансию.

Функция управления профилем компании включает следующие возможности:

- 
- получение текущих данных компании;
- редактирование названия компании, ИНН, ссылки на сайт и описания;
- загрузку логотипа компании;
- удаление ранее загруженного логотипа;
- отображение ошибок и сообщений об успешном сохранении данных.

Загрузка информации о компании выполняется при открытии личного кабинета работодателя. После получения данных от серверной части значения полей формы заполняются автоматически. Это позволяет работодателю редактировать уже существующую информацию без повторного ввода всех данных.

Рассмотрим реализацию загрузки данных профиля работодателя и компании, приведённую в листинге 3.

#### Листинг 3 – Загрузка данных работодателя и компании

```
const loadProfile = async () => {
  try {
    setIsLoading(true);
    setError("");

    const profile = await getEmployerProfile();

    setEmployerForm({
      email: profile.email || "",
```

```

        password: "",
        passwordConfirm: "",
    });

    setCompanyForm({
        name: profile.company?.name || "",
        inn: profile.company?.inn || "",
        link: profile.company?.link || "",
        description: profile.company?.description || "",
        logoUrl: profile.company?.logoUrl || "",
    });
} catch (loadError) {
    setError(
        loadError?.response?.data?.message ||
        "Не удалось загрузить данные профиля работодателя."
    );
} finally {
    setIsLoading(false);
}
};

```

В данном фрагменте функция `loadProfile` выполняет обращение к API-слою клиентского приложения и получает сведения о работодателе и связанной с ним компании. После успешного выполнения запроса данные сохраняются в состоянии компонента с помощью функций `setEmployerForm` и `setCompanyForm`. В случае возникновения ошибки пользователю выводится сообщение о невозможности загрузить данные профиля.

Редактирование информации о компании выполняется через отдельную форму. Пользователь изменяет значения полей, после чего данные отправляются на серверную часть для сохранения.

Фрагмент кода функции для сохранения данных компании приведён в листинге 4.

#### Листинг 4 – Сохранение изменений профиля компании

```

const handleCompanySubmit = async (event) => {
    event.preventDefault();
    try {
        setIsSavingCompany(true);
        setCompanyError("");
        setCompanySuccess("");
        await updateCompany({
            name: companyForm.name,
            inn: companyForm.inn,
            link: companyForm.link,
            description: companyForm.description,

```

```

    });
    setCompanySuccess("Данные компании успешно сохранены.");
    await loadProfile();
  } catch (saveError) {
    setCompanyError(
      saveError?.response?.data?.message ||
      "Не удалось сохранить данные компании."
    );
  } finally {
    setIsSavingCompany(false);
  }
};

```

В данном фрагменте обработчик `handleCompanySubmit` выполняет сохранение данных через функцию `updateCompany`. Перед отправкой запроса сбрасываются предыдущие сообщения об ошибках и успешных действиях. После успешного сохранения данные профиля повторно загружаются, что позволяет отобразить в интерфейсе актуальное состояние компании.

Для изменения логотипа компании используется отдельный механизм загрузки файла. Пользователь выбирает изображение, после чего файл передаётся серверной части через API-запрос.

Фрагмент реализации загрузки логотипа приведён в листинге 5.

#### Листинг 5 – Загрузка логотипа компании

```

const handleLogoUpload = async (event) => {
  const file = event.target.files?.[0];

  if (!file) {
    return;
  }

  try {
    setIsLogoUploading(true);
    setCompanyError("");

    await uploadCompanyLogo(file);
    await loadProfile();
  } catch (uploadError) {
    setCompanyError(
      uploadError?.response?.data?.message ||
      "Не удалось загрузить логотип компании."
    );
  } finally {
    setIsLogoUploading(false);
  }
};

```

Загрузка логотипа выполняется только в том случае, если пользователь выбрал файл. После успешной отправки изображения данные профиля обновляются повторно, благодаря чему новый логотип сразу отображается в интерфейсе. Такой подход позволяет избежать ручного обновления страницы и делает работу пользователя с профилем более удобной.

Удаление логотипа реализуется аналогичным образом. Пользователь нажимает кнопку удаления, после чего клиентская часть отправляет запрос на сервер и обновляет состояние профиля компании.

Фрагмент реализации удаления логотипа компании приведен в листинге 6.

#### Листинг 6 – Удаление логотипа компании

```
const handleLogoDelete = async () => {
  try {
    setIsLogoUploading(true);
    setCompanyError("");

    await deleteCompanyLogo();
    await loadProfile();
  } catch (deleteError) {
    setCompanyError(
      deleteError?.response?.data?.message ||
      "Не удалось удалить логотип компании."
    );
  } finally {
    setIsLogoUploading(false);
  }
};
```

В интерфейсе формы компании отображаются поля для ввода названия организации, ИНН, ссылки на сайт и описания

На рисунке 27 представлена форма управления профилем компании.

Рисунок 27 – Форма управления профилем компании

На рисунке видно, что форма управления профилем компании содержит основные сведения об организации и элементы управления логотипом. Работодатель может изменить информацию о компании, сохранить изменения, загрузить изображение логотипа или удалить ранее добавленный файл. Реализация данного интерфейса позволяет работодателю самостоятельно поддерживать актуальность информации, отображаемой студентам при просмотре вакансий.

#### **4.4 Реализация взаимодействия с клиентской частью**

Взаимодействие клиентской части с серверной частью является одним из основных механизмов работы информационной платформы. Клиентское приложение не хранит основные данные самостоятельно, а получает их от серверной части посредством HTTP-запросов. Такой подход позволяет отделить пользовательский интерфейс от бизнес-логики приложения и обеспечить централизованную обработку данных на стороне сервера.

В клиентской части взаимодействие с сервером используется при выполнении большинства пользовательских сценариев: регистрации и авторизации, загрузке профиля пользователя, редактировании данных студента или работодателя, просмотре вакансий, создании резюме, отправке откликов, управлении учебными группами и обмене сообщениями. Для выполнения запросов применяется библиотека Axios, а работа с серверными методами вынесена в отдельный API-слой.

Рассмотрим взаимодействие с серверной частью на примере получения списка вакансий. Список рекомендуемых вакансий отображается на странице просмотра вакансий. При открытии странице, клиентское приложение формирует и отправляет запрос на сервер, сервер формирует ответ, содержащий список вакансий и отправляет его обратно, клиентская часть принимает ответ и преобразует данные и выводит их в виде карточек.

Фрагмент кода для функции получения списка вакансий приведен в листинге 7.

### Листинг 7 – Получение списка вакансий

```
export const getVacancies = async (params) => {
  const response = await apiClient.get("/Vacancy", {
    params,
  });

  return response.data;
};
```

В данном фрагменте функция `getVacancies` выполняет GET-запрос к серверному методу `/Vacancy`. В качестве аргумента функция принимает объект `params`, содержащий параметры поиска и фильтрации. Это позволяет использовать один и тот же API-метод для получения как полного списка вакансий, так и списка, отфильтрованного по заданным пользователем условиям.

Пример использования данной функции в компоненте страницы приведён в листинге 8.

### Листинг 8 – Использование серверных данных в компоненте интерфейса

```
try {
  setIsLoading(true);
  setError("");

  const vacancyList = await getVacancies({
    search: searchValue,
    searchInTitle: true,
    searchInDescription: true,
    searchInCompanyName: true,
    region: filters.region,
    salaryFrom: filters.salaryFrom || undefined,
    specializationId: filters.specializationId ||
undefined,
  });

  setVacancies(vacancyList.map(mapVacancyToCardModel));
}
```

Полный код функции использования серверных данных можно найти в открытом репозитории [17]. В данном фрагменте компонент вызывает функцию `getVacancies` и передаёт в неё параметры, сформированные на основе состояния интерфейса. К таким параметрам относятся поисковая строка, регион, минимальная заработная плата и выбранная специализация. После получения ответа от сервера данные преобразуются функцией `mapVacancyToCardModel` и

сохраняются в состоянии компонента с помощью `setVacancies`. Это обеспечивает последующее отображение списка вакансий в пользовательском интерфейсе.

Для повышения удобства работы пользователя в компоненте также предусмотрена обработка состояний загрузки и ошибок. Перед выполнением запроса устанавливается состояние `isLoading`, которое может использоваться для отображения индикатора загрузки. В случае ошибки текст сообщения сохраняется в состоянии `error` и выводится пользователю. После завершения запроса, независимо от результата, состояние загрузки сбрасывается.

Отдельное значение имеет передача данных авторизации при обращении к защищённым серверным методам. Для этого в общем экземпляре `Axios` используется перехватчик запросов, который перед отправкой каждого запроса получает `access token` из пользовательской сессии и добавляет его в заголовок `Authorization`.

Функция для добавления токена авторизации приведена в листинге 9.

#### Листинг 9 – Добавление токена авторизации в запросы

```
apiClient.interceptors.request.use((config) => {
  const token = getAccessToken();

  if (token) {
    config.headers.Authorization = `Bearer ${token}`;
  }

  return config;
});
```

Использование данного механизма позволяет не добавлять токен вручную в каждом отдельном запросе. Все функции API-слоя используют общий экземпляр `apiClient`, поэтому авторизационные данные передаются автоматически. Это особенно важно для защищённых разделов приложения, например профиля пользователя, резюме, откликов, учебных групп и чатов.

Взаимодействие с серверной частью применяется не только для получения данных, но и для их изменения. Например, при редактировании профиля компании клиентская часть отправляет PUT-запрос с обновлёнными данными. После успешного выполнения операции интерфейс обновляет отображаемую информацию.

Реализация функции отправки измененных данных компании на сервер приведена в листинге 10.

#### Листинг 10 – Отправка изменённых данных компании на сервер

```
export const updateCompany = async (companyData) => {
  const response = await apiClient.put("/Company", companyData);

  return response.data;
};
```

В данном фрагменте функция `updateCompany` принимает объект с данными компании и отправляет его серверной части. После обработки запроса сервер возвращает результат, который используется клиентской частью для обновления состояния. При успешном выполнении операции, пользователь получает актуальные данные компании. В случае возникновения ошибки клиентская часть отображает соответствующее сообщение. Такой подход обеспечивает удобное редактирование профиля компании и поддерживает согласованность данных между интерфейсом и серверной частью.

На рисунке 28 представлена схема взаимодействия клиентской части с серверной частью.

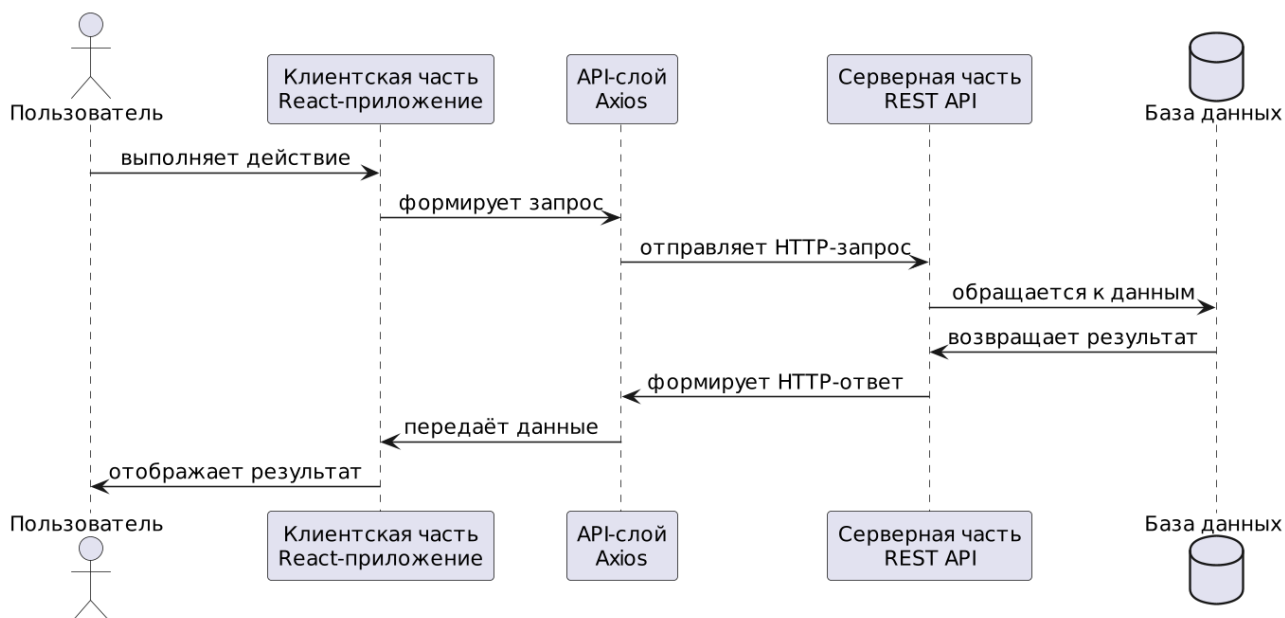


Рисунок 28 – Взаимодействие клиентской части с серверной частью

На рисунке показано, что пользователь выполняет действие в интерфейсе приложения, после чего компонент вызывает функцию API-слоя. API-слой

формирует HTTP-запрос и передаёт его серверной части. Серверная часть выполняет обработку запроса, обращается к базе данных при необходимости и возвращает ответ. После получения ответа клиентская часть обновляет состояние страницы и отображает результат пользователю.

Реализация взаимодействия с серверной частью обеспечивает получение, изменение и отображение актуальных данных информационной платформы.

#### 4.5 Реализация взаимодействия с серверной частью

Для взаимодействия клиентской части с серверной частью был реализован отдельный слой. Его основное назначение заключается в вынесении сетевых запросов из компонентов пользовательского интерфейса и сосредоточить работу с серверными методами в отдельных модулях. Такой подход позволяет снизить связанность между визуальными компонентами и логикой обмена данными, повысить читаемость кода и упростить дальнейшее сопровождение приложения.

В клиентской части API-слой используется для выполнения операций авторизации, получения и изменения профилей пользователей, работы с вакансиями, резюме, откликами, учебными группами, компаниями и чатами. Визуальные компоненты не обращаются к серверу напрямую, а вызывают специальные функции, которые формируют HTTP-запросы и возвращают результат в удобном для обработки виде.

Для выполнения HTTP-запросов используется библиотека Axios. Она позволяет централизованно настроить базовый адрес серверной части, заголовки запросов, передачу токена авторизации и обработку ошибок.

Общая настройка экземпляра Axios приведена в листинге 11.

Листинг 11 – Настройка экземпляра Axios для взаимодействия с серверной частью

```
import axios from "axios";
import { getAccessToken } from "../features/auth/session";

const apiClient = axios.create({
  baseURL: "http://localhost:5000/api",
  headers: {
```

```

        "Content-Type": "application/json",
      },
    });

apiClient.interceptors.request.use((config) => {
  const token = getAccessToken();

  if (token) {
    config.headers.Authorization = `Bearer ${token}`;
  }

  return config;
});

export default apiClient;

```

В данном фрагменте создаётся общий экземпляр `apiClient`, который используется во всех API-модулях приложения. В параметре `baseUrl` указывается адрес серверного API. Благодаря этому в отдельных функциях не требуется каждый раз прописывать полный путь к серверу. Также задаётся стандартный заголовок `Content-Type`, указывающий на передачу данных в формате JSON.

Особое значение имеет использование перехватчика запросов `interceptors.request`. Перед отправкой каждого запроса клиентское приложение получает `access token` из пользовательской сессии и, при его наличии, добавляет его в заголовок `Authorization`. Это позволяет автоматически передавать данные авторизации при обращении к защищённым методам серверной части.

API-слой разделён по функциональным областям приложения. Например, методы для работы с вакансиями вынесены в отдельный модуль. Это позволяет компонентам интерфейса использовать простые функции, не содержащие подробностей формирования HTTP-запроса.

Пример реализации функций для получения списка представлен в листинге 12.

#### Листинг 12 – API-функции для работы с вакансиями

```

export const getVacancies = async (params) => {
  const response = await apiClient.get("/Vacancy", {
    params,
  });
  return response.data;
}

```

```
};
export const getVacancyById = async (id) => {
  const response = await apiClient.get(`/Vacancy/${id}`);
  return response.data;};
```

Функция `getVacancies` выполняет запрос списка вакансий и принимает параметры поиска и фильтрации. Функция `getVacancyById` используется для получения подробной информации о выбранной вакансии. Отдельные функции предназначены для добавления вакансии в избранное и удаления её из избранного.

Компоненты пользовательского интерфейса используют данные функции без необходимости знать особенности URL-адресов и HTTP-методов. Например, страница просмотра вакансий вызывает `getVacancies`, передаёт параметры фильтрации и получает готовый набор данных для отображения в виде карточек.

Для работы с профилем компании работодателя также используется отдельный API-модуль. Он содержит функции получения информации о компании, изменения данных, загрузки логотипа и его удаления.

Пример реализации приведён в листинге 13.

#### Листинг 13 – API-функции для управления профилем компании

```
import apiClient from "../../api/apiClient";

export const updateCompany = async (companyData) => {
  const response = await apiClient.put("/Company", companyData);

  return response.data;
};

export const uploadCompanyLogo = async (file) => {
  const formData = new FormData();
  formData.append("file", file);

  const response = await apiClient.post("/Company/logo",
  formData, {
    headers: {
      "Content-Type": "multipart/form-data",
    },
  });

  return response.data;
};
```

В данном фрагменте показано, что API-слой поддерживает не только стандартную передачу JSON-данных, но и отправку файлов. Для загрузки

логотипа компании используется объект `FormData`, позволяющий передавать изображение в формате `multipart/form-data`. Это необходимо, поскольку загрузка файлов отличается от обычной отправки текстовых данных.

Аналогичным образом организованы API-модули для других разделов клиентского приложения. Каждый модуль отвечает за отдельную предметную область: авторизацию, профиль студента, профиль работодателя, резюме, отклики, учебные группы или чат. Такое разделение делает структуру клиентской части более понятной и позволяет быстро находить нужные методы при доработке приложения.

Использование API-слоя также упрощает обработку ошибок. Компонент интерфейса вызывает API-функцию внутри блока `try...catch`, после чего может вывести пользователю сообщение об ошибке или обновить состояние страницы при успешном выполнении операции.

Пример API-функции в компоненте приведён в листинге 14.

#### Листинг 14 – Использование API-функции в компоненте интерфейса

```
try {  
    setIsLoading(true);  
    setError("");  
  
    const result = await getVacancies({  
        search: searchValue,  
        region: selectedRegion,  
        salaryFrom: salaryFrom || undefined,  
    });  
  
    setVacancies(result.map(mapVacancyToCardModel));  
}
```

Полный код использования API-функции в компоненте интерфейса можно найти в открытом репозитории проекта [17]. В данном фрагменте компонент вызывает функцию `getVacancies`, передавая параметры поиска и фильтрации. После получения ответа данные преобразуются к модели отображения и сохраняются в состоянии компонента. Если запрос не выполняется по какой-либо причине, пользователь получает сообщение об ошибке с описание проблемы.

На рисунке 29 представлена схема работы API-слоя клиентского приложения.

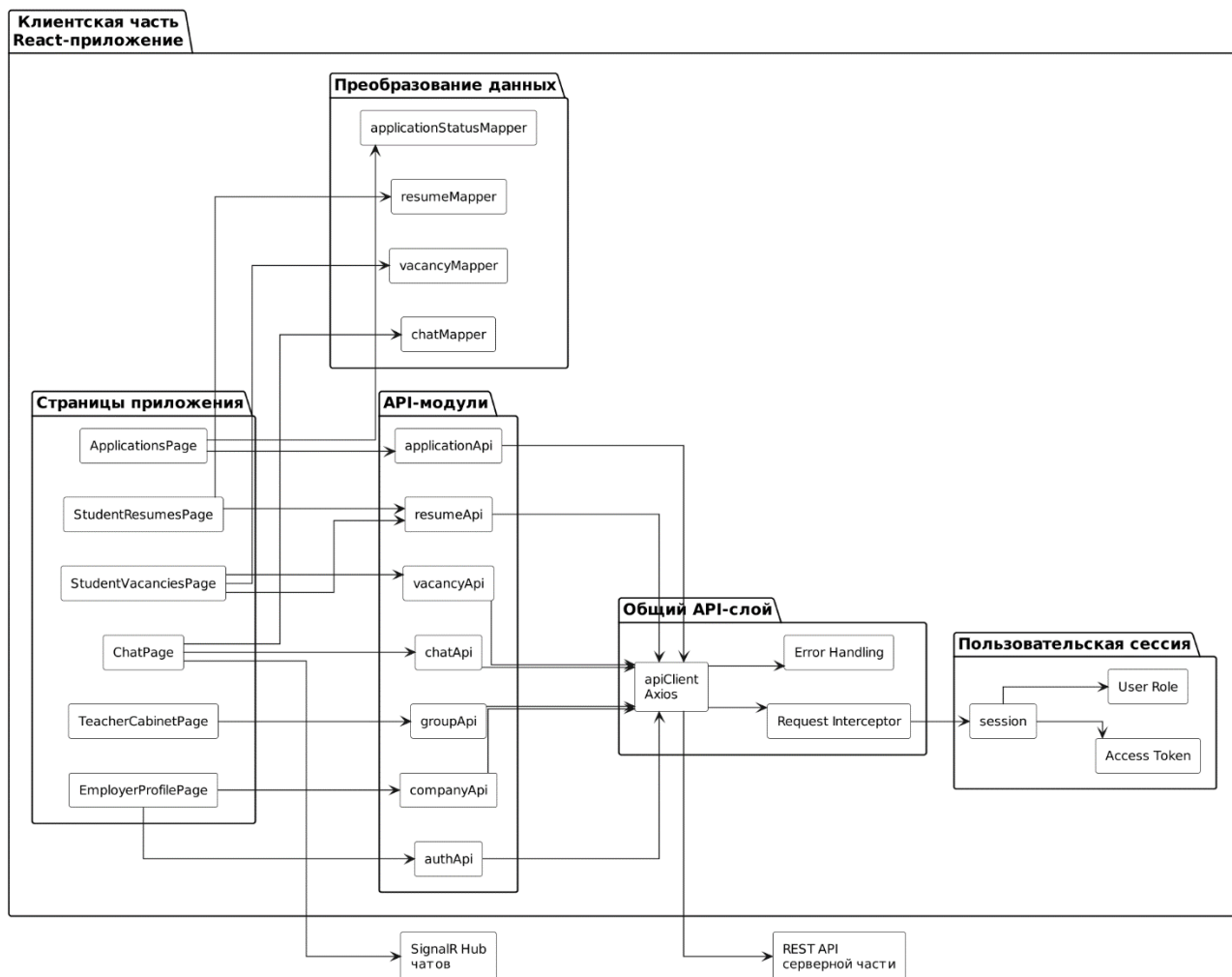


Рисунок 29 – Схема работы API-слоя клиентского приложения

На рисунке видно, что действия пользователя сначала обрабатываются компонентом интерфейса. Далее компонент вызывает функцию из соответствующего API-модуля. API-модуль использует общий экземпляр Axios, добавляет необходимые параметры и токен авторизации, после чего отправляет запрос на серверную часть. Полученный ответ возвращается в компонент, где данные отображаются пользователю или используется сообщение об ошибке.

Реализация API-слоя позволяет отделить сетевое взаимодействие от визуальных компонентов клиентского приложения. Это повышает модульность проекта и обеспечивает единый подход к работе с серверной частью. Благодаря централизованной настройке Axios и разделению API-функций по предметным

областям клиентская часть сохраняет понятную структуру и может быть расширена при добавлении новых возможностей платформы.

## **5 Тестирование информационной платформы**

### **5.1 Функциональное тестирование клиентской части**

Для информационной платформы трудоустройства и стажировок функциональное тестирование гарантирует, что основные операции, предоставляемые приложением, выполняются корректно и безопасно для пользователей разных ролей. В актуальной версии клиентская часть поддерживает три роли студента, работодателя и преподавателя.

Функциональное тестирование – вид тестирования, который проверяет насколько точно реализованные в приложении функции соответствуют предъявленным требованиям. Обычно данный подход называется тестированием «черным ящиком», так как оценивается внешнее поведение системы.

В рамках функционального тестирования клиентской части платформы было проведено тестирование черным ящиком формы регистрации работодателя и связанных пользовательских сценариев.

### **5.2 Функциональная диаграмм и таблица решений**

Функциональная диаграмм – визуальная диаграмма, отражающая функции системы и связи между ними. С её помощью можно увидеть, как отдельные компоненты взаимодействуют друг с другом для достижения целевого результата. В рамках тестирования была рассмотрена форма регистрации работодателя, так как она включает несколько типов входных данных: email, название компании, ИНН, пароль и подтверждение пароля.

При применении этой методики требовалось перечислить все возможные причины некорректной работы системы, зафиксировать сообщения об ошибках и на основе этого построить упрощенную схему работы формы. В результате

были проанализированы поля формы регистрации работодателя и выделены причины и следствия событий, указанные в таблице 9.

Таблица 9 – Причины и следствия событий

| Причины   |                                                         |
|-----------|---------------------------------------------------------|
| П1        | Корректный email работодателя                           |
| П2        | Заполнено название компании                             |
| П3        | Корректный ИНН компании                                 |
| П4        | Пароль содержит не менее 6 символов                     |
| П5        | Подтверждение пароля совпадает с паролем                |
| Следствия |                                                         |
| С1        | Сообщение: «Введите корректный email»                   |
| С2        | Сообщение: «Введите название компании»                  |
| С3        | Сообщение: «ИНН должен содержать 10 или 12 цифр»        |
| С4        | Сообщение: «Пароль должен содержать минимум 6 символов» |
| С5        | Сообщение: «Пароли не совпадают»                        |
| С6        | Сообщение: «Регистрация работодателя прошла успешно»    |

По данной таблице была составлена функциональная диаграмма, представленная на рисунке 30.

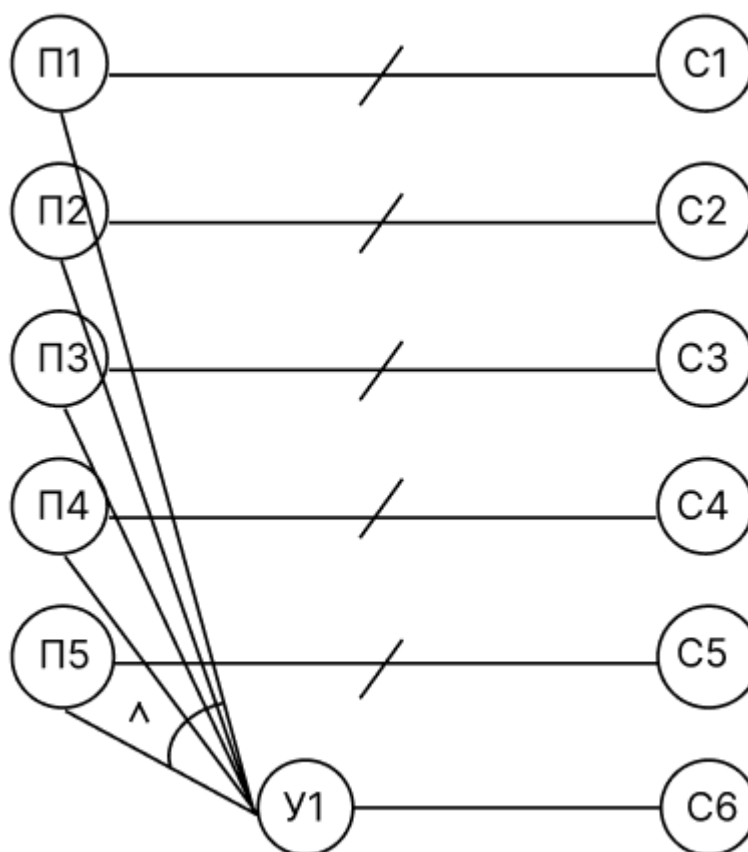


Рисунок 30 – Функциональная диаграмма регистрации работодателя

Таблицы с представлением полного перебора всех существующих ограничений составляет 32 варианта. Чтобы уменьшить число проверок без потери охвата, был применен комбинированный подход – попарное тестирование. Метод гарантирует, что для каждой возможной пары входных параметров рассматриваются все дискретные сочетания значений, тем самым ощутимо сокращая объем сценариев. Из исходных 32 вариантов были выделены 16 тест-сценариев. Окончательный перечень приведен в таблице 10.

Таблица 10 – Таблица решений

|    | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|----|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|
| П1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1  | 0  | 1  | 0  | 1  | 0  | 1  |
| П2 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0  | 1  | 1  | 0  | 0  | 1  | 1  |
| П3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0  | 0  | 0  | 1  | 1  | 1  | 1  |
| П4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| П5 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0  | 1  | 0  | 1  | 0  | 1  | 1  |
|    |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |
| С1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0  | 1  | 0  | 1  | 0  | 1  | 0  |
| С2 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1  | 0  | 0  | 1  | 1  | 0  | 0  |
| С3 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1  | 1  | 1  | 0  | 0  | 0  | 0  |
| С4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| С5 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1  | 0  | 1  | 0  | 1  | 0  | 0  |
| С6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 1  |

Результаты выполнения тестовых сценариев, определенных в таблице 9, подтвердили корректность работы формы регистрации работодателя при различных сочетаниях входных условий.

Для детального анализа была рассмотрена форма регистрации работодателя. Форма регистрации является важной частью приложения, так как без её должного функционирования пользователи не смогут получать доступ к платформе.

Результаты тестирования приведены в таблице 11.

Таблица 11 – Тест-кейсы для поля ИНН работодателя

| № | Входные данные                                                                                         | Выходные данные                   | Фактический результат                   |
|---|--------------------------------------------------------------------------------------------------------|-----------------------------------|-----------------------------------------|
| 1 | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 7707083894; Пароль: 12345678; Подтверждение: 12345678 | Успешная регистрация работодателя | Результат теста приведен на рисунке В.1 |

Продолжение таблицы 11

| №  | Входные данные                                                                                             | Выходные данные                                                                                           | Фактический результат                    |
|----|------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|------------------------------------------|
| 2  | Email: employer1@mail.ru; Компания: Компания 2; ИНН: 500100732258; Пароль: qwerty1; Подтверждение: qwerty1 | Успешная регистрация работодателя                                                                         | Результат теста приведен на рисунке В.2  |
| 3  | Email: пустое значение; Компания: ИнфоТех; ИНН: 7707083893; Пароль: 123456; Подтверждение: 123456          | Ошибка: «Введите email»                                                                                   | Результат теста приведен на рисунке В.3  |
| 4  | Email: employer; Компания: ИнфоТех; ИНН: 7707083893; Пароль: 123456; Подтверждение: 123456                 | Ошибка: «Адрес электронной почты должен содержать символ «@». В адресе «employer» отсутствует символ «@»» | Результат теста приведен на рисунке В.4  |
| 5  | Email: employer@mail.ru; Компания: пустое значение; ИНН: 7707083893; Пароль: 123456; Подтверждение: 123456 | Ошибка: «Введите название компании»                                                                       | Результат теста приведен на рисунке В.5  |
| 6  | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: пустое значение; Пароль: 123456; Подтверждение: 123456    | Ошибка: «Введите ИНН»                                                                                     | Результат теста приведен на рисунке В.6  |
| 7  | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 123456789; Пароль: 123456; Подтверждение: 123456          | Ошибка: «ИНН должен содержать 10 или 12 цифр»                                                             | Результат теста приведен на рисунке В.7  |
| 8  | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 12345678901; Пароль: 123456; Подтверждение: 123456        | Ошибка: «ИНН должен содержать 10 или 12 цифр»                                                             | Результат теста приведен на рисунке В.8  |
| 9  | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 1234567890123; Пароль: 123456; Подтверждение: 123456      | Ошибка: «ИНН должен содержать 10 или 12 цифр»                                                             | Результат теста приведен на рисунке В.9  |
| 10 | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 77070A3893; Пароль: 123456; Подтверждение: 123456         | Ошибка: «ИНН должен содержать 10 или 12 цифр»                                                             | Результат теста приведен на рисунке В.10 |
| 11 | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 770 7083893; Пароль: 123456; Подтверждение: 123456        | Ошибка: «ИНН должен содержать 10 или 12 цифр»                                                             | Результат теста приведен на рисунке В.11 |
| 12 | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 7707083893; Пароль: 12345; Подтверждение: 12345           | Ошибка: «Пароль должен содержать минимум 8 символов»                                                      | Результат теста приведен на рисунке В.12 |
| 13 | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 7707083893; Пароль: 123456; Подтверждение: 654321         | Ошибка: «Пароли не совпадают»                                                                             | Результат теста приведен на рисунке В.13 |

Продолжение таблицы 11

|    |                                                                                                                      |                                                         |                                          |
|----|----------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------|------------------------------------------|
| 14 | Email: employer@mail.com; Компания: пустое значение; ИНН: пустое значение; Пароль: 12345; Подтверждение: 123456      | Отображение сообщений об ошибках для некорректных полей | Результат теста приведен на рисунке В.14 |
| 15 | Email: employer@mail.ru; Компания: пустое значение; ИНН: 77070A3893; Пароль: 123456; Подтверждение: 654321           | Отображение сообщений об ошибках для некорректных полей | Результат теста приведен на рисунке В.15 |
| 16 | Email: employer@mail.ru; Компания: ИнфоТех; ИНН: 7707083893; Пароль: пустое значение; Подтверждение: пустое значение | Ошибка: «Введите пароль»                                | Результат теста приведен на рисунке В.16 |

Клиентская форма регистрации работодателя корректно обрабатывает как правильные, так и ошибочные данные, не отправляя на сервер заведомо некорректные значения.

### 5.3 Тестирование SignalR-взаимодействия

Одной из важных частей тестирования информационной платформы трудоустройства и стажировок является проверка корректности взаимодействия студента и работодателя в чатах. В актуальной версии клиентская часть использует библиотеку @microsoft/signalr и подключается к серверному узлу /hubs/chat. Соединение создаётся при помощи HubConnectionBuilder, передаёт access token через accessTokenFactory и поддерживает автоматическое переподключение при кратковременной потере связи.

Для проверки такого взаимодействия использовались интерфейс приложения и инструменты разработчика браузера. В ходе тестирования проверялись установка соединения с серверным хабом, получение списка чатов, выбор конкретного диалога, загрузка истории сообщений, отправка нового сообщения, получение входящего сообщения и обновление признака прочтения.

При открытии диалога клиентское приложение сначала получает информацию о доступных чатах и истории сообщений через REST API. После этого устанавливается соединение с SignalR-хабом, который выполняет проверку прав доступа пользователя и подключает его к группе выбранного чата.

Такой механизм необходим для того, чтобы сообщения получали только участники соответствующего диалога.

При отправке сообщения данные передаются из клиентской части в серверную часть через SignalR-хаб. Далее сообщение сохраняется в базе данных и рассылается участникам диалога. Благодаря этому новые сообщения отображаются у пользователей в режиме реального времени без ручного обновления страницы и без повторного запроса данных к серверу.

Такой подход исключает необходимость постоянного опроса сервера, снижает задержки при доставке сообщений и уменьшает нагрузку на сеть. В результате обеспечивается более плавная и отзывчивая работа интерфейса чата. Дополнительно проверялось поведение соединения при временной потере связи: механизм автоматического переподключения позволяет восстановить обмен сообщениями без повторного входа пользователя в систему.

Тестирование отправки и получения сообщений выполнялось с использованием двух пользовательских сессий, открытых от имени разных участников одного диалога. После подключения обоих пользователей к SignalR-хабу отправлялось сообщение от имени первого пользователя, после чего проверялось его отображение как у отправителя, так и у получателя. Успешным результатом считалось отправка сообщения.

Тестирование взаимодействия клиентской и серверной частей подтвердило корректность работы авторизации, основных REST-запросов и SignalR-чата, используемого для общения студента и работодателя.

#### **5.4 Модульное тестирование клиентской части**

Модульное тестирование — это автоматизированный тест, предназначенный для проверки корректности работы отдельного модуля программы в изоляции от других компонентов. Он позволяет убедиться, что конкретная часть кода функционирует согласно заданным требованиям, независимо от внешних зависимостей [18].

Для реализации модульного тестирования клиентской части информационной платформы была использована библиотека Vitest. Она обеспечивает высокую скорость выполнения тестов благодаря использованию многопоточности и предоставляет поддержку таких функций, как мокирование, покрытие кода и снапшоты, что делает её удобным инструментом для современных веб-приложений [19].

Рассмотрим работу модульных тестов на примере проверки компонента валидации профиля студента, данный тест представлен в листинге 15.

#### Листинг 15 – Тест валидации профиля студента

```
it("validates email, password, password confirmation and
graduation year", () => {
  assert.deepEqual(
    validateStudentProfileForm({
      email: "invalid-email",
      name: "Ivan",
      surname: "Ivanov",
      password: "123",
      passwordConfirm: "321",
      graduationYear: "20A4",
    }),
    {
      email: "Введите корректный email",
      password: "Пароль должен содержать минимум 6 символов",
      passwordConfirm: "Пароли не совпадают",
      graduationYear: "Укажите год в формате YYYY",
    }
  );
});
```

По данному тесту можно заметить, что функция `validateStudentProfile` вызывается в тестовом окружении с заведомо некорректными данными. Вместо реального пользовательского ввода здесь используется прямая передача объекта, а эмуляция событий не требуется, так как тестируется чистая логика валидации. Тест проверяет, что функция возвращает объект с ожидаемыми сообщениями об ошибках для каждого поля: email, пароль, подтверждение пароля и год окончания. Если возвращенные сообщения совпадают с ожидаемыми, тест считается пройденным успешно.

На рисунке 31 представлен результат выполнения этого теста в терминале.

```

▶ validateStudentProfileForm
  ✓ returns required field errors (3.3994ms)
  ✓ validates email, password, password confirmation and graduation year (0.2853ms)
  ✓ returns no errors for valid data (1.112ms)
  ✓ validateStudentProfileForm (5.9844ms)

```

Рисунок 31 – Выполнения теста проверки валидации

Аналогичный подход был применен для написания модульных тестов, покрывающих ключевые компоненты клиентской части, в том числе обработка статусов пользователя, обработка данных профиля, авторизация и регистрация, и другие ключевые сценарии приложения. Общий результат выполнения всех тестов приведен на рисунке 32.

```

✓ src/features/auth/api/auth.api.test.js (6 tests) 7ms
✓ src/features/student/utls/studentProfileValidation.test.js (3 tests) 4ms
✓ src/features/employer/utls/employerProfileUtils.test.js (3 tests) 5ms
✓ src/features/student/utls/studentProfileMappers.test.js (2 tests) 4ms
✓ src/features/employer/utls/companyValidation.test.js (3 tests) 4ms
✓ src/features/applications/utls/applicationStatus.test.js (2 tests) 3ms

Test Files  10 passed (10)
Tests      35 passed (35)
Start at   14:41:25
Duration   811ms (transform 299ms, setup 0ms, import 700ms, tests 66ms, environment 1ms)

```

Рисунок 32 – Результат выполнения всех модульных тестов

На рисунке показано, что все тестовые наборы завершились успешно – ни одного падения или ошибки. Это подтверждает корректность работы клиентской части приложения. Приложение Vitest для модульного тестирования позволило добиться высокой надежности кода, своевременно выявить дефекты и обеспечить стабильность пользовательского интерфейса. Исходный код приведен в приложении Б.

## 5.5 Оценка качества программного кода

Для проведения оценки качества программного кода были использованы метрики программного кода. Их использование позволяет определить готовность программы к дальнейшему развитию, расширяемости и надежности кода.

Метрики программного кода обеспечивают возможность измерения сложности разрабатываемой системы, что позволяет сделать вывод о её поддерживаемости и удобстве внесения изменений [20].

Для оценки качества кода разрабатываемой информационной платформы были применены широко распространённые метрики – метрика Холстеда и метрика Чепина.

Метрика Чепина служит инструментом для измерения структурной сложности программных модулей на основе анализа их внутренней логики. В рамках данного подхода производится классификация и количественный подсчет переменных, задействованных внутри модуля, что в итоге позволяет оценить его сложность и повысить читаемость исходного кода [21].

## 5.6 Метрики Холстеда

С целью количественного анализа сложности программного кода информационной платформы в качестве исследуемого объекта была выбрана функция `validateCompanyForm`, реализованная в модуле `companyValidate.js`.

В рамках метрики Холстеда исходный код рассматривается как совокупность лексем двух типов – операторов и операндов. Для вычисления данной метрики необходимо зафиксировать следующие параметры:

- количество уникальных операторов ( $n_1$ );
- общее число всех операторов ( $N_1$ );
- количество уникальных операндов ( $n_2$ );
- общее число всех операндов ( $N_2$ ).

Указанные четыре базовые характеристики выступают в качестве исходных данных для расчетных формул Холстеда. На их основе представляется возможность вычислить интегральные показатели сложности и трудоемкости анализируемой программы.

Перечень операторов, встречающихся в функции `validateCompanyForm`, а также частота их использования приведены в таблице 12.

Таблица 12 – Операторы функции validateCompanyForm

| №       | Операторы    | Число вхождений |
|---------|--------------|-----------------|
| 1       | export const | 1               |
| 2       | =            | 4               |
| 3       | =>           | 1               |
| 4       | Const        | 1               |
| 5       | {}           | 1               |
| 6       | if           | 3               |
| 7       | !            | 2               |
| 8       | .            | 7               |
| 9       | trim         | 3               |
| 10      | &&           | 2               |
| 11      | test         | 2               |
| 12      | return       | 1               |
| 13      | ;            | 3               |
| n1 = 13 | N1 = 31      |                 |

Таблица 13 – Операнды функции validateCompanyForm

| №       | Операнды                                                    | Число вхождений |
|---------|-------------------------------------------------------------|-----------------|
| 1       | validateCompanyForm                                         | 1               |
| 2       | formData                                                    | 7               |
| 3       | errors                                                      | 5               |
| 4       | Name                                                        | 2               |
| 5       | inn                                                         | 3               |
| 6       | link                                                        | 3               |
| 7       | «Введите название компании»                                 | 1               |
| 8       | /^\d{10}(\d{2})?\$/                                         | 1               |
| 9       | «ИНН должен содержать 10 или 12 цифр»                       | 1               |
| 10      | /^https?:\\S+\$ /i                                          | 1               |
| 11      | «Введите корректную ссылку, начиная с http:// или https://» | 1               |
| n2 = 11 | N2 = 26                                                     |                 |

Результат лексического разбора функции validateCompanyForm, представленные в таблицах 12 (операторы) и 13 (операнды), послужили основой для расчёта всех метрических показателей согласно методике Холстеда. Полученные значения сведены в таблицу 14.

Таблица 14 – Результаты вычисления по метрике Холстеда

| Значение                   | Результат |
|----------------------------|-----------|
| Длина реализации программы | 57        |
| Словарь программы          | 24        |
| Расчетная длина программы  | 86.16     |
| Объем программы            | 261.34    |
| Сложность программы        | 15.36     |
| Работа программирования    | 4015.18   |

Рассчитанные по методике Холстеда показатели дали возможность количественно охарактеризовать сложность реализации функции `validateCompanyForm`. Полученные числовые значения свидетельствуют о том, что данная функция обладает небольшим лексическим объемом и логически прозрачной структурой, поскольку выполняет строго ограниченный набор валидационных проверок и возвращает объект с сообщениями об ошибках, не производя побочных эффектов в системе.

## 5.7 Метрика Чепина

Для характеристики структурной сложности была произведена оценка по метрике Чепина для функции `validateStudentProfileForm` из модуля `studentProfileValidation.js`. Все идентификаторы функции проанализированы и распределены на 4 категории:

- параметры, поступающие на вход и участвующие в формировании результата (P);
- переменные, чье значение изменяется в ходе работы (M);
- элементы, которые влияют на выбор ветви исполнения (C);
- неиспользуемые идентификаторы (T).

Так как один идентификатор может выполнять несколько ролей, при классификации в функции он учитывался во всех категориях. Для каждой группы методика Чепина приписывает собственный коэффициент, отражающий её вклад в общую сложность программного блока: для входных и выходных переменных коэффициент равен 1, для изменяемых – удвоенный (2), для управляющих – тройной (3), для неиспользуемых – половинный (0,5).

В результате анализа в программном блоке валидации профиля студента были обнаружены 1 входная, 1 изменяющаяся, 6 управляющих и 0 неиспользуемых переменных. Итоговый показатель вычисляется как количество переменных каждой категории, умноженное на её коэффициент, затем полученные произведения складываются.

Полученное значение метрики Чепина  $Q = 21$  говорит о том, что функция `validateStudentProfileForm` имеет умеренную сложность: она содержит несколько независимых проверок пользовательских данных, но остается компактной и хорошо читаемой.

В результате оценки качества кода можно сделать вывод, что ключевые функции клиентской части обладают приемлемой сложностью и хорошо подходят для дальнейшего сопровождения.

## Заключение

В рамках выполнения выпускной квалификационной работы была разработана клиентская часть информационной платформы трудоустройства и стажировок студентов в ИТ-компаниях. Разработанное web-приложение предназначено для организации удобного взаимодействия между студентами, работодателями и преподавателями, а также обеспечивает доступ пользователей к основным функциям платформы через единый пользовательский интерфейс.

На первом этапе работы был выполнен анализ предметной области и существующих программных аналогов. Были рассмотрены общие платформы поиска работы, специализированные сервисы для ИТ-сферы и университетские информационные системы. По результатам анализа было установлено, что существующие решения решают отдельные задачи трудоустройства и сопровождения студентов, однако не обеспечивают комплексного взаимодействия между студентами, работодателями и преподавателями. На основании этого были сформулированы функциональные и нефункциональные требования к разрабатываемой клиентской части.

На этапе проектирования была определена архитектура информационной платформы и место клиентского приложения в общей структуре системы. Клиентская часть была спроектирована как модульное React-приложение, включающее интерфейсы для ролей студента, работодателя и преподавателя. Были разработаны решения по организации маршрутизации, разграничению доступа, построению пользовательских интерфейсов, взаимодействию с серверной частью и отображению данных, получаемых через API.

В ходе реализации клиентской части были созданы основные пользовательские интерфейсы и функциональные модули приложения. Для студента реализованы возможности регистрации, авторизации, работы с профилем, просмотра и фильтрации вакансий, создания и редактирования резюме, отправки откликов и обмена сообщениями с работодателями. Для работодателя реализованы функции управления профилем компании, размещения вакансий, просмотра откликов и взаимодействия с кандидатами. Для

преподавателя предусмотрены интерфейсы работы с учебными группами и заявками студентов.

В рамках тестирования были проверены основные пользовательские сценарии клиентской части. Тестирование SignalR-взаимодействия клиентской и серверной частей, модульное тестирование отдельных функций и компонентов, а также оценка качества программного кода с использованием метрик Холстеда и Чепина. Полученные результаты подтвердили корректность работы реализованных механизмов и соответствие приложения поставленным требованиям.

В ходе выполнения выпускной квалификационной работы была создана клиентская часть информационной платформы, обеспечивающая выполнение основных сценариев. Разработанное приложение соответствует поставленным функциональным и нефункциональным требованиям.

### Список использованных источников

1. HRtech-рынок в России вырос на 38% [Электронный ресурс]. URL: <https://smartranking.ru/ru/analytics/hrtech/hrtech-rynok-v-rossii-vyros-na-38/> (дата обращения: 01.05.2026).
2. Рынок труда в России (ИТ и телеком) [Электронный ресурс]. URL: [https://www.tadviser.ru/index.php/Статья:Рынок\\_труда\\_в\\_России\\_%28ИТ\\_и\\_телеком%29?](https://www.tadviser.ru/index.php/Статья:Рынок_труда_в_России_%28ИТ_и_телеком%29?) (дата обращения: 01.05.2026).
3. Образование в цифрах: 2024 [Электронный ресурс]. URL: <https://issek.hse.ru/news/969714262.html> (дата обращения: 01.05.2026).
4. HeadHunter [Электронный ресурс]. URL: <https://penza.hh.ru/> (дата обращения: 01.05.2026).
5. SuperJob [Электронный ресурс]. URL: <https://penza.superjob.ru/> (дата обращения: 01.05.2026).
6. Хабр Карьера [Электронный ресурс]. URL: <https://career.habr.com/> (дата обращения: 01.05.2026).
7. Работа России [Электронный ресурс]. URL: <https://trudvsem.ru/> (дата обращения: 01.05.2026).
8. Модуль ВУЗ+РАБОТОДАТЕЛЬ [Электронный ресурс]. URL: [https://lider.pnzgu.ru/vyz\\_rabotodatel](https://lider.pnzgu.ru/vyz_rabotodatel) (дата обращения: 01.05.2026).
9. Функциональные и нефункциональные требования к ПО: что нужно знать перед запуском проекта [Электронный ресурс]. URL: <https://blog.skillfactory.ru/funktsionalnye-i-nefunktsionalnye-trebovaniya-k-po/> (дата обращения: 01.05.2026).
10. React [Электронный ресурс]. URL: <https://react.dev/> (дата обращения: 01.05.2026).
11. The Build Tool for the Web [Электронный ресурс]. URL: <https://vite.dev/> (дата обращения: 01.05.2026).
12. React Router Official Documentation [Электронный ресурс]. URL: <https://reactrouter.com/> (дата обращения: 01.05.2026).

13. First steps | axios | Promise based HTTP client [Электронный ресурс]. URL: <https://axios.rest/pages/getting-started/first-steps> (дата обращения: 01.05.2026).
14. Tailwind CSS [Электронный ресурс]. URL: <https://tailwindcss.com/> (дата обращения: 01.05.2026).
15. ASP.NET Core [Электронный ресурс]. URL: <https://dotnet.microsoft.com/ru-ru/apps/aspnet> (дата обращения: 01.05.2026).
16. Понимание компонентной архитектуры React: почему это важно [Электронный ресурс]. URL: <https://ru.linkedin.com/pulse/understanding-reacts-component-based-architecture-why-agnihotri-mf7rf?tl=ru> (дата обращения: 01.05.2026).
17. InternshipPlatform – клиентская часть информационной платформы трудоустройства и стажировки студентов в ИТ-компаниях // GitHub [Электронный ресурс]. URL: <https://github.com/p4shok/InternshipPlatform>
18. Ошеров, Р. Искусство автономного тестирования с примерами на C# / Р. Ошеров. – 2-е изд. – М.: ДМК Пресс, 2016. – 360 с.
19. Vitest. Руководство пользователя [Электронный ресурс] / Vitest. – Режим доступа: <https://www.getbook.com/ru/book/vitest-0/guide/features> (дата обращения: 01.05.2026).
20. Программный код и его метрики // Тематический блог «Хабр». [Электронный ресурс]. URL: <https://habr.com/ru/companies/intel/articles/106082/> (дата обращения 01.05.2026).
21. Ледовских, И. Метрики сложности кода // Технический отчет 2012-2 И. Ледовских. [Электронный ресурс]. URL: [https://se.math.spbu.ru/SE/preprints/prep\\_25\\_2013.pdf](https://se.math.spbu.ru/SE/preprints/prep_25_2013.pdf) (дата обращения: 01.05.2026).

**Приложение А**  
**(рекомендуемое)**  
**Диаграммы вариантов использования**

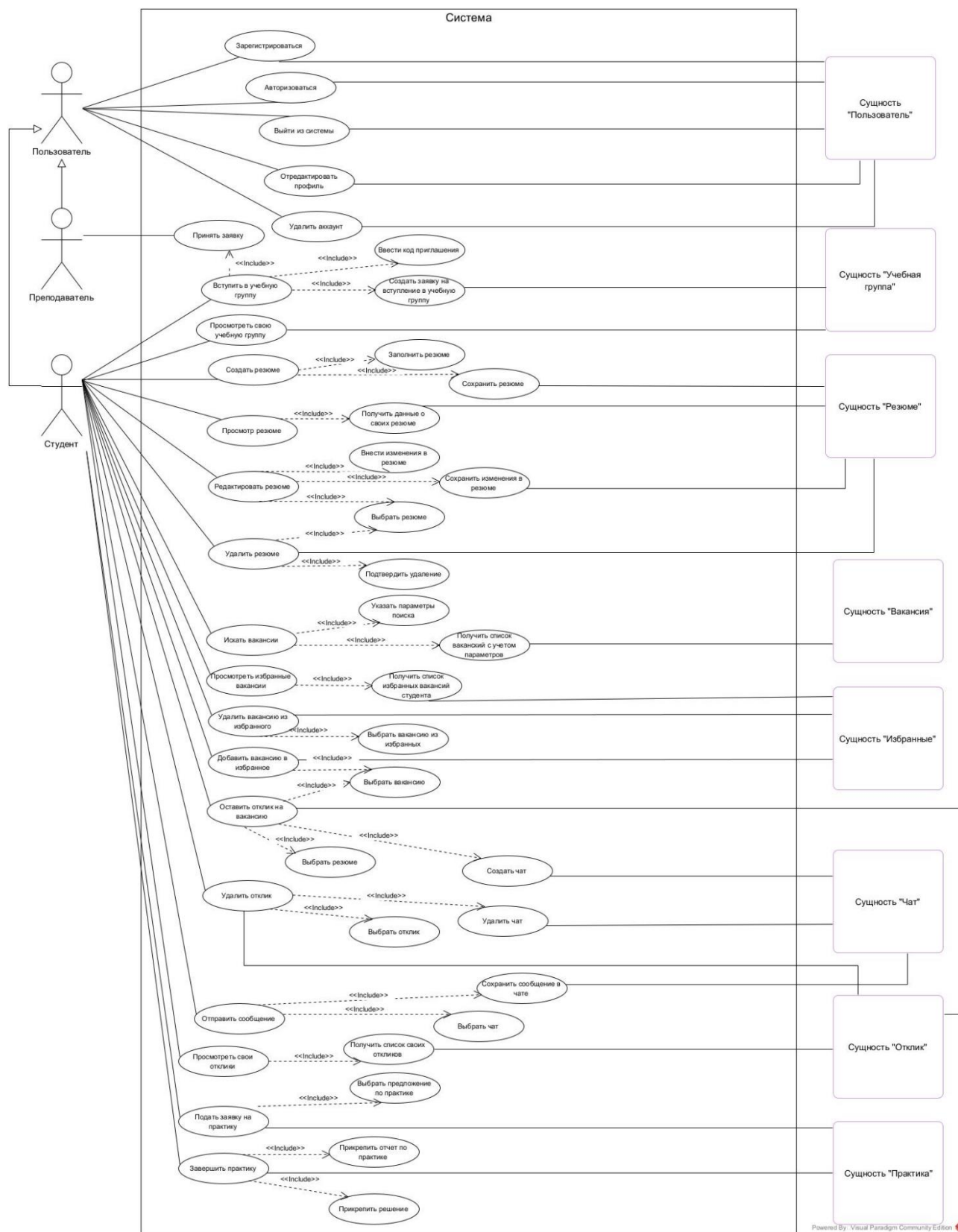


Рисунок А.1 – Диаграмма вариантов использования для роли студента

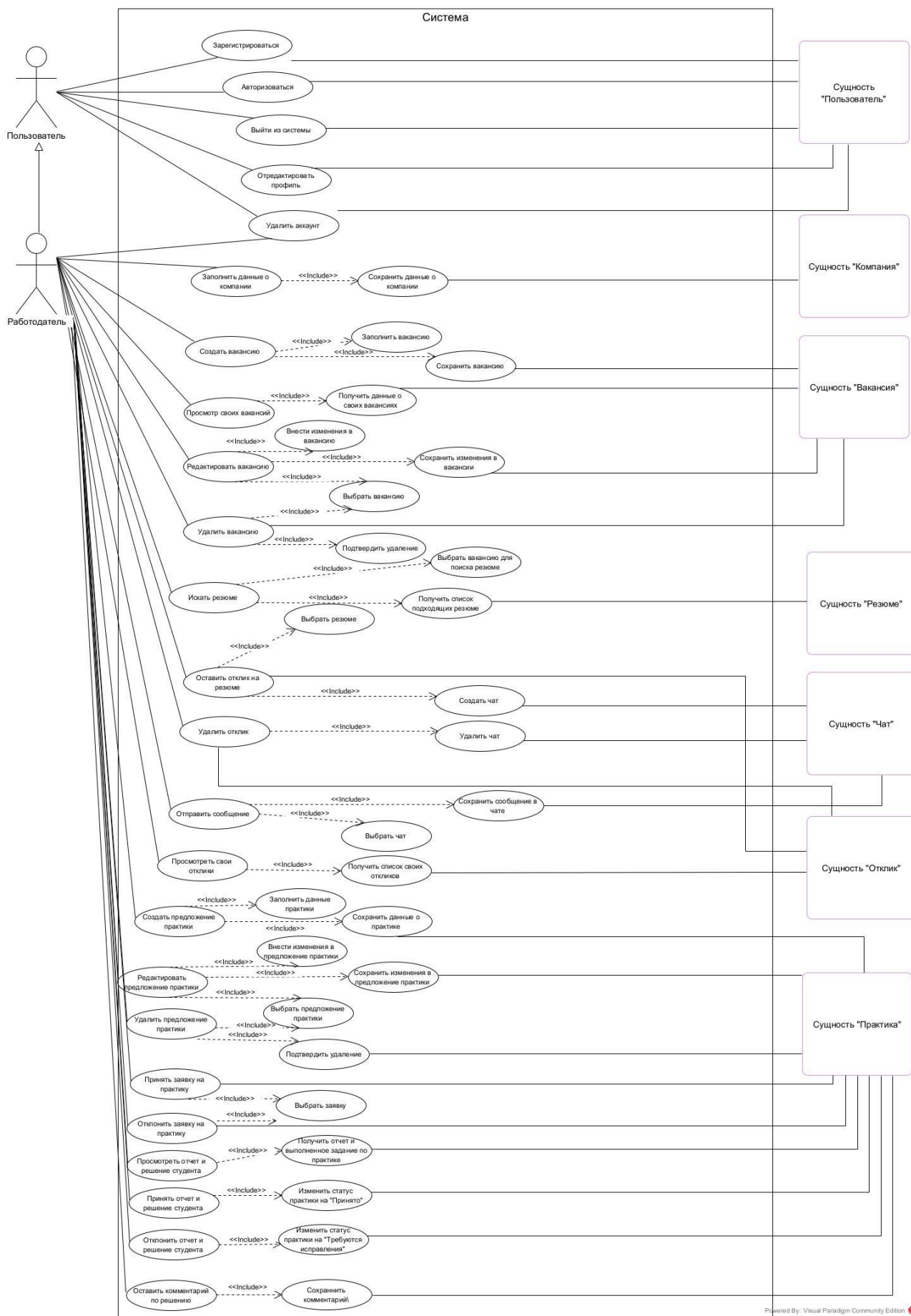


Рисунок А.2 – Диаграмма вариантов использования для роли работодателя

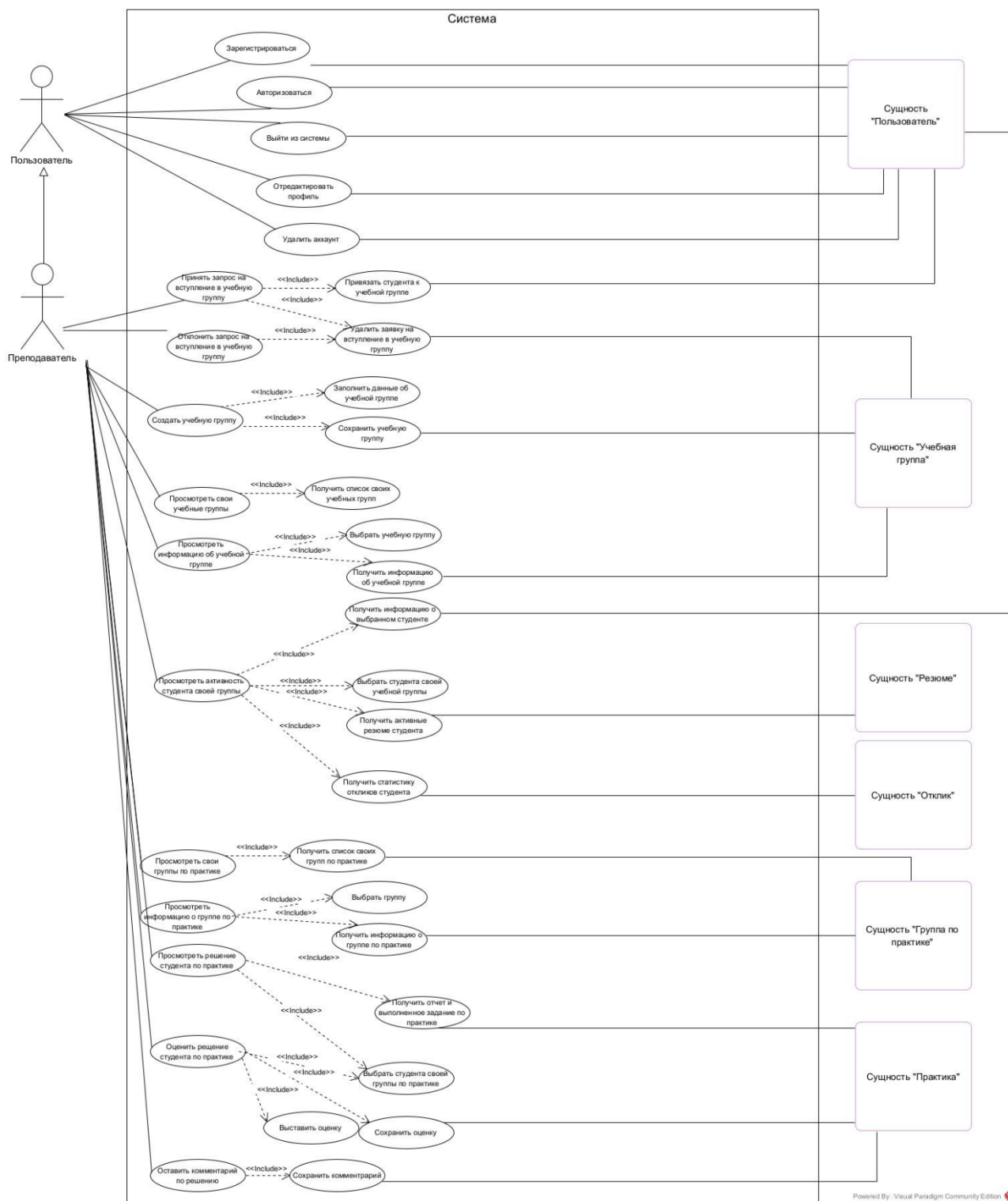


Рисунок А.3 – Диаграмма вариантов использования для роли преподавателя

## **Приложение Б**

**(обязательное)**

**Код клиентской части приложения**

### Листинг Б.1 – main.jsx

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import App from './app/App.jsx'
import './styles/index.css';

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```

### Листинг Б.2 – App.jsx

```
import React from "react";
import { BrowserRouter } from "react-router-dom";
import AppRoutes from "../routes/AppRoutes";

const App = () => {
  return (
    <BrowserRouter>
      <AppRoutes />
    </BrowserRouter>
  );
};

export default App;
```

### Листинг Б.3 – axios.js

```
import axios from "axios";
import { getAccessToken } from
"../features/auth/utils/session.js";

const api = axios.create({
  baseURL: import.meta.env?.VITE_API_BASE_URL || "",
  withCredentials: true,
  headers: {
    "Content-Type": "application/json",
  },
});

api.interceptors.request.use((config) => {
  const accessToken = getAccessToken();

  if (accessToken) {
    config.headers.Authorization = `Bearer ${accessToken}`;
  }

  return config;
});
```

```
export default api;
```

### Листинг Б.4 – AuthInput.jsx

```
import React from "react";

const AuthInput = ({
  label,
  type = "text",
  name,
  value,
  onChange,
  placeholder,
  error,
  autoComplete,
  ...props
}) => {
  return (
    <div className="space-y-2">
      <label
        htmlFor={name}
        className="block text-sm font-medium text-slate-700"
      >
        {label}
      </label>

      <input
        id={name}
        name={name}
        type={type}
        value={value}
        onChange={onChange}
        placeholder={placeholder}
        autoComplete={autoComplete}
        {...props}
        className={`w-full rounded-2xl border bg-white px-4 py-3
text-sm text-slate-900 shadow-sm outline-none transition-all
duration-200 placeholder:text-slate-400
      ${
        error
          ? "border-red-400 ring-4 ring-red-100 focus:border-red-500"
          : "border-slate-200 focus:border-indigo-500
focus:ring-4 focus:ring-indigo-100"
      }`}
      />

      {error && <p className="text-sm text-red-500">{error}</p>}
    </div>
  );
};
```

```
export default AuthInput;
```

### Листинг Б.5 – PasswordInput.jsx

```
import React, { useState } from "react";

const PasswordInput = ({
  label,
  name,
  value,
  onChange,
  placeholder,
  error,
  autoComplete,
}) => {
  const [showPassword, setShowPassword] = useState(false);

  return (
    <div className="space-y-2">
      <label
        htmlFor={name}
        className="block text-sm font-medium text-slate-700"
      >
        {label}
      </label>

      <div className="relative">
        <input
          id={name}
          name={name}
          type={showPassword ? "text" : "password"}
          value={value}
          onChange={onChange}
          placeholder={placeholder}
          autoComplete={autoComplete}
          className={`w-full rounded-2xl border bg-white px-4 py-3
pr-14 text-sm text-slate-900 shadow-sm outline-none transition-all
duration-200 placeholder:text-slate-400
          ${
            error
              ? "border-red-400 ring-4 ring-red-100
focus:border-red-500"
              : "border-slate-200 focus:border-indigo-500
focus:ring-4 focus:ring-indigo-100"
          }`}
        />

        <button
          type="button"
          onClick={() => setShowPassword((prev) => !prev)}
          className="absolute right-3 top-1/2 -translate-y-1/2
rounded-xl px-3 py-1 text-xs font-medium text-slate-600 transition
hover:bg-slate-100 hover:text-slate-900"
```

```

        >
        {showPassword ? "Скрыть" : "Показать"}
      </button>
    </div>

    {error && <p className="text-sm text-red-500">{error}</p>}
  </div>
);
};

export default PasswordInput;

```

### Листинг Б.6 – chatHub.js

```

import * as signalR from "@microsoft/signalr";
import { CHAT_HUB_ENDPOINT } from "../../../../../api/endpoints";
import { getAccessToken } from "../../auth/utils/session";

const baseUrl = import.meta.env.VITE_API_BASE_URL || "";
const appBaseUrl = baseUrl.replace(/\\/api\\/?$/, "");
const hubUrl = `${appBaseUrl}${CHAT_HUB_ENDPOINT}`;

let connectionPromise = null;

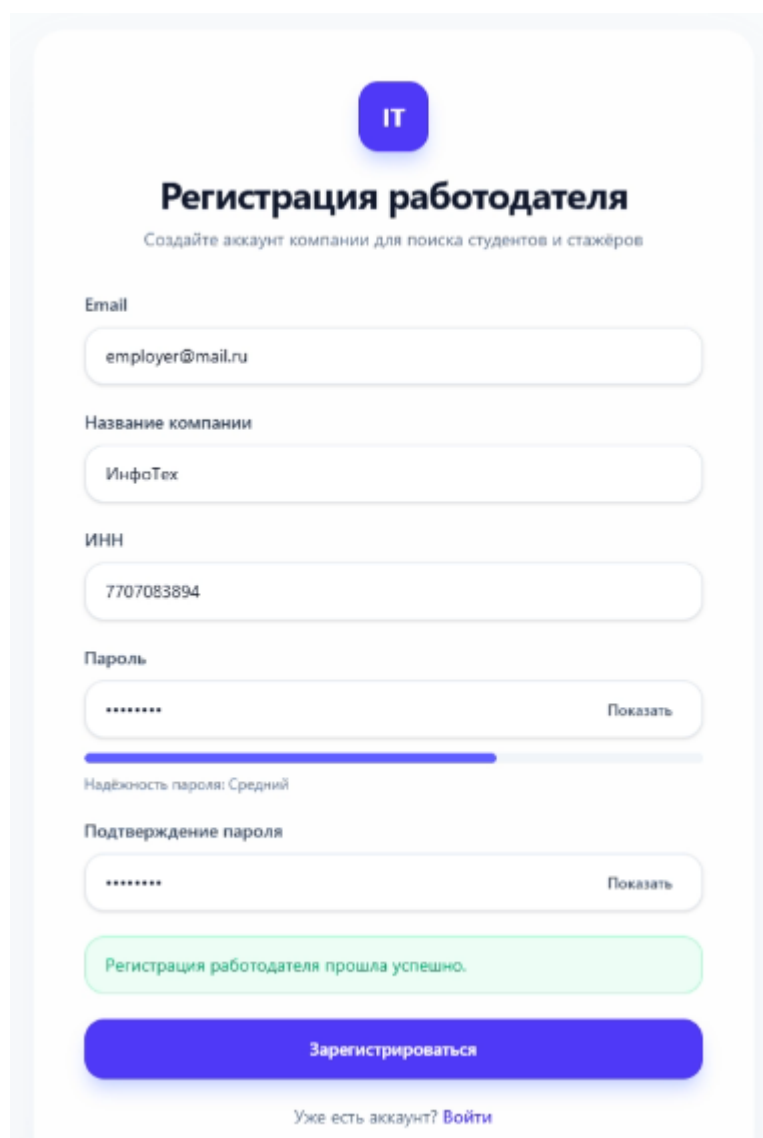
const createConnection = () =>
  new signalR.HubConnectionBuilder()
    .withUrl(hubUrl, {
      accessTokenFactory: () => getAccessToken(),
      withCredentials: true,
    })
    .withAutomaticReconnect()
    .build();

export const getChatConnection = async () => {
  if (!connectionPromise) {
    const connection = createConnection();
    connectionPromise = connection.start().then(() =>
connection).catch((error) => {
    connectionPromise = null;
    throw error;
  });
  }

  return connectionPromise;
};

```

**Приложение В**  
**(рекомендуемое)**  
**Тестирование**



The image shows a web registration form for an employer. At the top, there is a blue square icon with the letters 'IT'. Below it, the title 'Регистрация работодателя' (Employer Registration) is displayed in bold, followed by the subtitle 'Создайте аккаунт компании для поиска студентов и стажёров' (Create a company account for finding students and interns). The form contains several input fields: 'Email' with the value 'employer@mail.ru', 'Название компании' (Company Name) with 'ИнфоТех', and 'ИНН' (TIN) with '7707083894'. There are two password fields, both masked with dots, each with a 'Показать' (Show) link. A progress bar is visible below the first password field, and a message 'Надёжность пароля: Средний' (Password strength: Average) is shown. A green message box at the bottom states 'Регистрация работодателя прошла успешно.' (Employer registration was successful.). A large blue button labeled 'Зарегистрироваться' (Register) is positioned below the message. At the very bottom, a link says 'Уже есть аккаунт? Войти' (Already have an account? Log in).

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

employer@mail.ru

Название компании

ИнфоТех

ИНН

7707083894

Пароль

Показать

Надёжность пароля: Средний

Подтверждение пароля

Показать

Регистрация работодателя прошла успешно.

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.1 – Успешная регистрация нового работодателя

The image shows a web form titled "Регистрация работодателя" (Employer Registration) with a sub-header "Создайте аккаунт компании для поиска студентов и стажёров" (Create a company account for finding students and interns). The form includes fields for Email (employer1@mail.ru), Название компании (Компания 2), ИНН (500100732258), and Password (masked with asterisks). A password strength indicator shows "Надёжность пароля: Средний" (Password reliability: Average) with a blue progress bar. A confirmation field for the password is also present. A green message box states "Регистрация работодателя прошла успешно." (Employer registration was successful). A large blue button at the bottom says "Зарегистрироваться" (Register). At the very bottom, there is a link: "Уже есть аккаунт? Войти" (Already have an account? Log in).

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

employer1@mail.ru

Название компании

Компания 2

ИНН

500100732258

Пароль

Показать

Надёжность пароля: Средний

Подтверждение пароля

Показать

Регистрация работодателя прошла успешно.

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.2 – Успешная регистрация нового работодателя

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

company@example.com

Введите email

Название компании

Компания без почты

ИНН

7707083893

Пароль

.....

Показать

Надёжность пароля: Средний

Подтверждение пароля

.....

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.3 – Пустой ввод email

The image shows a web registration form for an employer. At the top, there is a blue square icon with the letters 'IT'. Below it, the title 'Регистрация работодателя' (Employer Registration) is displayed in a large, bold, black font. Under the title, a subtitle reads 'Создайте аккаунт компании для поиска студентов и стажёров' (Create a company account for finding students and interns). The form contains several input fields: 'Email' with the value 'employer', 'Компания без почты' (Company without email), 'ИНН' (TIN) with the value '7707083893', 'Пароль' (Password) with masked characters and a 'Показать' (Show) link, and 'Подтверждение пароля' (Confirm password) also with masked characters and a 'Показать' link. A blue progress bar is positioned below the password field, showing approximately 50% completion. A red error message box is overlaid on the form, stating: 'Адрес электронной почты должен содержать символ "@". В адресе "employer" отсутствует символ "@".' (Email address must contain the symbol '@'. The address 'employer' does not contain the symbol '@'). At the bottom of the form is a large blue button labeled 'Зарегистрироваться' (Register). Below the button, there is a link that says 'Уже есть аккаунт? Войти' (Already have an account? Log in).

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

employer

Адрес электронной почты должен содержать символ "@". В адресе "employer" отсутствует символ "@".

Компания без почты

ИНН

7707083893

Пароль

..... Показать

Надёжность пароля: Средний

Подтверждение пароля

..... Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.4 – Некорректный ввод email

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

Введите название компании

Введите название компании

ИНН

7707083893

Пароль

.....

Показать

Надёжность пароля: Средний

Подтверждение пароля

.....

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.5 – Пустое значение поля названия компании

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

Название компании

ИНН

Введите ИНН

Пароль

Показать

Надёжность пароля: Средний

Подтверждение пароля

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.6 – Пустое значение ИНН

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

Название

ИНН

123456789

ИНН должен содержать 10 или 12 цифр

Пароль

.....

Показать

Надёжность пароля: Средний

Подтверждение пароля

.....

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.7 – Ввод меньше 10 символов в поле ИНН

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

Название компании

ИНН

ИНН должен содержать 10 или 12 цифр

Пароль

Показать

Надёжность пароля: Средний

Подтверждение пароля

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.8 – Ввод 11 символов в поле ИНН

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

Название

ИНН

1234567890123

ИНН должен содержать 10 или 12 цифр

Пароль

.....

Показать

Надёжность пароля: Средний

Подтверждение пароля

.....

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.9 – Ввод 13 символов в поле ИНН

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

Название компании

ИНН

ИНН должен содержать 10 или 12 цифр

Пароль

Показать

Надёжность пароля: Средний

Подтверждение пароля

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.10 – Ввод ИНН с недопустимыми символами

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

Название

ИНН

770 7083893

ИНН должен содержать 10 или 12 цифр

Пароль

.....

Показать

Надёжность пароля: Средний

Подтверждение пароля

.....

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.11 – Ввод ИНН с пробелом

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

Название компании

ИНН

Пароль

Показать

Пароль должен содержать минимум 6 символов

Надёжность пароля: Слабый

Подтверждение пароля

Показать

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.12 – Ввод пятизначного пароля

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

Название компании

ИНН

Пароль

Показать

Надёжность пароля: Средний

Подтверждение пароля

Показать

Пароли не совпадают

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.13 – Ввод разных паролей

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

Введите название компании

Введите название компании

ИНН

Введите ИНН

Введите ИНН

Пароль

12345678

Скрыть

Надёжность пароля: Средний

Подтверждение пароля

12345678

Скрыть

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.14 – Ввод пустых полей для названия компании и ИНН

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

Введите название компании

Введите название компании

ИНН

77070A3893

ИНН должен содержать 10 или 12 цифр

Пароль

12345678

Скрыть

Надёжность пароля: Средний

Подтверждение пароля

123456

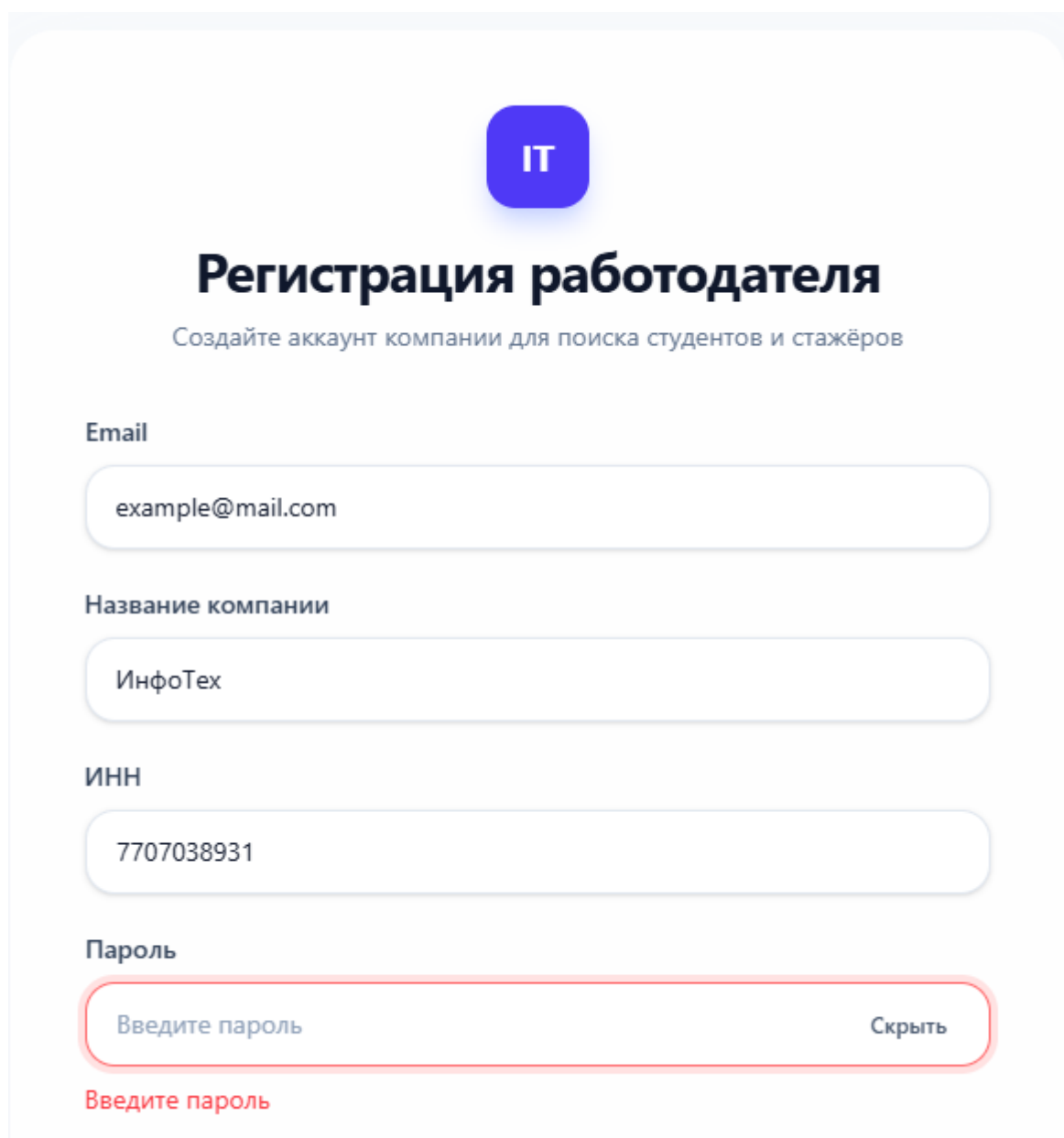
Скрыть

Пароли не совпадают

Зарегистрироваться

Уже есть аккаунт? [Войти](#)

Рисунок В.15 – Ввод некорректных данных регистрации



The image shows a registration form for an employer. At the top, there is a blue square icon with the letters 'IT' in white. Below the icon, the title 'Регистрация работодателя' (Employer Registration) is displayed in a large, bold, black font. Underneath the title, a subtitle in a smaller, gray font reads 'Создайте аккаунт компании для поиска студентов и стажёров' (Create a company account for finding students and interns). The form consists of four input fields, each with a label above it: 'Email' (containing 'example@mail.com'), 'Название компании' (containing 'ИнфоТех'), 'ИНН' (containing '7707038931'), and 'Пароль' (containing 'Введите пароль'). The password field is highlighted with a red border, and a 'Скрыть' (Hide) button is located to its right. Below the password field, the text 'Введите пароль' (Enter password) is written in red.

IT

## Регистрация работодателя

Создайте аккаунт компании для поиска студентов и стажёров

Email

example@mail.com

Название компании

ИнфоТех

ИНН

7707038931

Пароль

Введите пароль

Скрыть

Введите пароль

Рисунок В.16 – Пустое значение пароля