МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г. ШУХОВА»

(БГТУ им. В.Г. Шухова)

Институт *информационных технологий и управляющих систем*

Кафедра *программного обеспечения вычислительной техники и автоматизированных систем*

Направление подготовки *09.03.04 – Программная инженерия*

Направленность (профиль) образовательной программы *Разработка программно-информационных систем*

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему:

«Разработка front-end Web – приложения – учебной среды с чатами и AI-анализом кода лабораторных работ»

Студент: Бондаренко Сергей Владимирович

Зав. кафедрой: канд. техн. наук, доц. Поляков В.М.

Руководитель: Мельников А.Б.

К защите допустить:

Зав. кафедрой _____ /Поляков В.М./

«____» _____ 2025 г.

Белгород 2025 г.

МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г. ШУХОВА»

(БГТУ им. В.Г. Шухова)

Институт *информационных технологий и управляющих систем*

Кафедра *программного обеспечения вычислительной техники и автоматизированных систем*

Направление подготовки *09.03.04 – Программная инженерия*

Направленность (профиль) образовательной программы *Разработка программно-информационных систем*

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему:

«Разработка front-end Web – приложения – учебной среды с чатами и AI-анализом кода лабораторных работ»

Студент: Бондаренко Сергей Владимирович

Зав. кафедрой: канд. техн. наук, доц. Поляков В.М.

Руководитель: Мельников А.Б.

К защите допустить:

Зав. кафедрой _____ /Поляков В.М./

«____» _____ 2025 г.

Белгород 2025 г.

1 ОПРЕДЕЛЕНИЯ, СОКРАЩЕНИЯ И ОБОЗНАЧЕНИЯ

ИИ - искуственный интелект.

Содержание

2 Введение

Развитие цифровых технологий в сфере образования значительно меняет способы взаимодействия между преподавателями и студентами, предоставляя новые возможности для обучения и обмена информацией. В условиях дистанционного и смешанного обучения особенно важной становится необходимость создания платформ, которые бы объединяли образовательные инструменты в едином пространстве. Веб-приложения, которые решают задачи взаимодействия, позволяют сократить барьеры между преподавателями и студентами, улучшить коммуникацию и повысить качество образования. Цифровая среда должна обеспечивать не только размещение учебных материалов и заданий, но и средства для общения, автоматической оценки и анализа решений с использованием современных технологий, включая искусственный интеллект.

Актуальность темы заключается в потребности создания интегрированной образовательной платформы, которая объединяет функции чатов, проведения занятий и автоматического анализа решений, используя возможности ИИ. На данный момент отсутствует единая система, которая бы эффективно сочетала в себе эти ключевые аспекты: возможность общения через чаты, создание заданий и автоматизированную проверку решений с помощью ИИ. Современные платформы, как правило, фрагментированы — отдельные системы для чатов, другие для размещения заданий, третьи для автоматической проверки кода, что значительно усложняет организацию учебного процесса и снижает его эффективность. Разработка интегрированного решения, которое объединило бы эти элементы, позволяет улучшить качество образовательного процесса, повысив продуктивность студентов и преподавателей, а также упростив взаимодействие и автоматизировав многие рутинные задачи.

Целью данной работы является разработка клиентской части образовательной платформы, которая будет включать функции взаимодействия между преподавателями и студентами, автоматизированную проверку кода, а также возможности общения в рамках чатов. Особое внимание уделяется созданию такого интерфейса,

который позволит преподавателям и студентам взаимодействовать в едином пространстве, где будут доступны все образовательные инструменты и ресурсы.

Для достижения поставленной цели необходимо решить следующие задачи:

а) Проанализировать предметную область и существующие системы, выявив их сильные и слабые стороны.

б) Определить архитектурные и технологические решения, подходящие для реализации клиентской части платформы.

в) Спроектировать пользовательский интерфейс, обеспечивающий интуитивное и удобное взаимодействие для преподавателей и студентов.

г) Разработать компоненты для управления учебными структурами (институт, кафедра, группа), заданиями и чатами.

д) Интегрировать средства для автоматизированной проверки решений студентов с применением ИИ.

е) Реализовать тестирование бизнес-логики приложения для обеспечения её корректности и эффективности.

Структура пояснительной записки включает следующие разделы:

– В первом разделе рассматриваются особенности предметной области, проводится анализ существующих решений и обоснование выбора технологий и методов проектирования. Приводится обзор существующих образовательных платформ и их недостатков, а также объясняется необходимость разработки интегрированного решения.

– Во втором разделе описывается архитектура клиентской части приложения, структура пользовательского интерфейса, проектирование компонентов и их взаимодействие. Рассматриваются решения для реализации системы чатов, создания и проверки заданий, а также интеграции ИИ-анализа.

– В третьем разделе приводится описание реализации: структура кода, используемые технологии (Next.js, React, TypeScript, Redux, Auth.js), описание экрана и взаимодействий, примеры реализации различных компонентов системы.

– В заключении приводятся выводы по выполненной работе, оценивается эффективность разработанного интерфейса и функционала, а также определяются направления для дальнейшего развития и улучшения системы. Указываются перспективы внедрения ИИ в образовательные платформы для улучшения процессов оценки и взаимодействия.

3 ОБЗОР И ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

3.1 Введение в предметную область

Современные образовательные процессы переживают значительные изменения под воздействием цифровых технологий. В условиях быстрого роста объемов информации и перехода на дистанционное и смешанное обучение возникает потребность в создании платформ, которые объединяют различные образовательные инструменты в единую систему. Проблемы, с которыми сталкиваются преподаватели и студенты, включают фрагментацию существующих решений: чаты для общения, отдельные системы для размещения и проверки заданий, а также инструменты для анализа решений студентов.

Существующие платформы не всегда обеспечивают интеграцию всех этих функций в одном приложении, что приводит к необходимости использования множества разных сервисов для выполнения учебных задач. В рамках образовательных процессов это усложняет взаимодействие между преподавателями и студентами, увеличивает время на организацию обучения и снижает его эффективность.

Одной из важнейших задач является создание платформы, которая объединяет все эти компоненты в одном месте, обеспечивая удобный интерфейс для студентов и преподавателей. Такая система должна включать:

- возможность создания и размещения учебных заданий;
- автоматическое тестирование решений студентов с использованием ИИ для проверки правильности кода;
- чат-функциональность для общения студентов с преподавателями и внутри групп;
- централизованный доступ к учебным материалам.

Интеграция всех этих функций в одну платформу позволит значительно упростить организацию учебного процесса, улучшить взаимодействие между пре-

подавателями и студентами, а также повысить качество обучения за счет автоматизации рутинных задач.

3.2 Анализ существующих образовательных платформ

Современные образовательные платформы, такие как Moodle, Google Classroom, Microsoft Teams для образования, а также специализированные решения, предназначенные для работы с программированием, предлагают различные функциональные возможности для взаимодействия преподавателей и студентов. Однако каждая из этих платформ имеет свои ограничения и не всегда покрывает все потребности в рамках единой системы.

3.2.1 Moodle

Moodle является одной из самых популярных образовательных платформ, используемых во многих учебных заведениях. Она предоставляет инструменты для размещения учебных материалов, организации тестов и заданий, а также ведения онлайн-курсов. Однако, несмотря на свои возможности, Moodle не предоставляет встроенных решений для автоматической проверки кода студентов, а также не включает в себя продвинутые механизмы общения в реальном времени, что делает её менее эффективной для динамичного взаимодействия в процессе обучения.

3.2.2 Google Classroom

Google Classroom предлагает простоту в использовании и позволяет интегрировать различные Google сервисы. Платформа позволяет преподавателям создавать задания, прикреплять материалы и отслеживать выполнение студентами. Однако Google Classroom не предоставляет функциональности для автоматического анализа решений, особенно в контексте программирования. Это требует интеграции с внешними инструментами, что усложняет использование системы в образовательных учреждениях.

3.2.3 Microsoft Teams for Education

Microsoft Teams, в отличие от Moodle и Google Classroom, активно используется для организации видеоконференций и групповых чатов. Он позволяет преподавателям и студентам взаимодействовать в реальном времени, а также интегрирует различные сервисы Microsoft 365. Однако, как и в случае с другими платформами,

Microsoft Teams не предоставляет функционала для интегрированного анализа кода студентов с использованием искусственного интеллекта, что ограничивает его возможности в обучении программированию.

3.2.4 Платформы для анализа кода

Существуют специализированные платформы, такие как CodeSignal, Codility, LeetCode, которые позволяют преподавателям и работодателям тестировать навыки программирования студентов. Эти системы используют алгоритмы для автоматической проверки решений, однако они ограничены в функционале взаимодействия с преподавателями и студентами, а также не обеспечивают централизованный доступ к учебным материалам и заданиям.

3.3 Проблемы существующих решений

Основной проблемой существующих образовательных платформ является фрагментация функционала. На данный момент нет единой платформы, которая бы эффективно объединяла создание и проверку заданий, общение преподавателей и студентов, а также использовала бы технологии ИИ для автоматизированного анализа решений студентов. Это затрудняет образовательный процесс и снижает его эффективность, особенно в условиях быстро меняющихся требований дистанционного обучения.

Таким образом, для улучшения образовательного процесса существует необходимость в разработке единой интегрированной платформы, которая бы сочетала в себе все эти компоненты и обеспечивала бы максимально удобное взаимодействие для всех участников учебного процесса.

3.4 Потребности образовательной среды

Современные образовательные процессы предъявляют высокие требования к функциональности учебных платформ. Для эффективного взаимодействия между преподавателями и студентами необходимо создавать приложения, которые обеспечивают организационную и техническую поддержку всех ключевых элементов образовательного процесса.

3.4.1 Доступность материалов и заданий

Материалы и задания должны быть доступны студентам в любое время. Приложение должно обеспечивать размещение учебных ресурсов в различных форматах (текст, видео, презентации) и упрощать их поиск и использование. Это позволяет студентам готовиться к занятиям и выполнять задания без привязки ко времени, а преподавателям — быстро обновлять и дополнять учебные модули.

3.4.2 Автоматизация проверок и оценки

Автоматизированная проверка заданий существенно ускоряет процесс получения обратной связи. Использование искусственного интеллекта для анализа кода позволяет выявлять ошибки, давать подсказки и оценивать работы без участия преподавателя. Это освобождает ресурсы преподавателя для индивидуальной поддержки студентов и более сложной экспертной оценки.

3.4.3 Удобная система заданий и общения

Приложение должно включать удобную систему создания и отслеживания заданий. Важно, чтобы преподаватели могли формулировать задания, прикреплять к ним материалы и получать результаты выполнения. Неотъемлемой частью также является возможность общения между участниками процесса — как в групповых, так и личных чатах, для обмена мнениями и получения поддержки.

3.4.4 Интеграция всех процессов в одну систему

Отдельные решения для чатов, размещения заданий и анализа кода создают фрагментированную среду. Необходима единая платформа, объединяющая все эти компоненты. Это упрощает взаимодействие, повышает удобство и эффективность обучения, а также снижает затраты на сопровождение и обучение работе с системой.

3.4.5 Вывод

Таким образом, при проектировании образовательной платформы следует учитывать потребности в постоянном доступе к материалам, автоматической проверке решений, поддержке взаимодействия и целостности функционала в рамках одного интерфейса.

3.5 Технологии разработки клиентской части приложения

Для реализации клиентской части платформы выбраны современные инструменты, обеспечивающие модульность, производительность, типизацию и масштабируемость интерфейса.

3.5.1 TypeScript

JavaScript является одним из самых популярных языков программирования для веб-разработки. Он широко используется для создания динамичных веб-страниц и приложений, поскольку позволяет работать с элементами DOM, асинхронно загружать данные и обеспечивать интерактивность пользовательских интерфейсов. Однако JavaScript имеет важный недостаток — отсутствие статической типизации. Это означает, что переменные и функции не привязываются к определённым типам данных, что может привести к ошибкам на этапе выполнения, которые трудно обнаружить в процессе разработки. Особенно это может быть проблемой в крупных приложениях, где сложно отслеживать все возможные типы данных и их изменения.

Для устранения этих проблем был разработан язык TypeScript, являющийся надмножеством JavaScript. TypeScript добавляет в JavaScript статическую типизацию, что позволяет разработчикам явно указывать типы данных для переменных и функций. Это значительно снижает вероятность ошибок и улучшает поддержку кода в будущем. Благодаря строгой типизации TypeScript помогает предотвращать баги, связанные с динамическими типами в JavaScript, и улучшает автозаполнение в редакторах кода. TypeScript распространяется как библиотека, которую можно интегрировать в проекты на JavaScript, обеспечивая совместимость с существующим кодом и позволяя постепенно внедрять типизацию без необходимости переписывать весь проект. Это особенно важно в крупных и масштабируемых приложениях, где несколько разработчиков работают с общими компонентами, и типизация помогает поддерживать консистентность кода на протяжении всего проекта.

3.5.2 React

React — библиотека для построения пользовательских интерфейсов, разработанная Facebook. Она широко используется в веб-разработке благодаря своей простоте, гибкости и высокой производительности. React обеспечивает декларативный стиль программирования, при котором разработчик описывает, как должен вы-

глядеть интерфейс при заданном состоянии, а библиотека самостоятельно обновляет DOM при изменениях. Это упрощает разработку сложных и динамичных интерфейсов.

Ключевые особенности:

- **Компонентный подход**: Приложение разбивается на переиспользуемые и изолированные компоненты
- **JSX-синтаксис**: Комбинация JavaScript и разметки, упрощающая написание UI
- **Virtual DOM**: Эффективное обновление только изменённых элементов страницы
- **Hooks API**: Современный способ управления состоянием и побочными эффектами

Преимущества для образовательных платформ:

- Быстрая разработка за счёт декларативности и компонентного подхода
- Большое сообщество и развитая экосистема (Next.js, Redux, React Query и др.)
- Поддержка SSR и SSG при использовании Next.js
- Простая интеграция с библиотеками и сторонними сервисами

Ограничения:

- Отсутствие встроенной архитектуры — требует выбора и настройки дополнительных инструментов
- Более низкий порог входа может привести к «разнообразию» архитектурных подходов в команде
- Без Next.js не включает такие возможности как маршрутизация, SSR и API

3.5.3 Angular

Angular — полноценный MVC-фреймворк, предоставляющий комплексное решение для разработки enterprise-приложений. В отличие от React, Angular накладывает строгую архитектурную модель, что обеспечивает единообразие кодовой базы в крупных проектах.

Ключевые особенности:

– **Двустороннее связывание данных**: Автоматическая синхронизация между моделью и представлением
– **Инъекция зависимостей**: Встроенный механизм для управления сервисами и их зависимостями
– **CLI-инструменты**: Генерация компонентов, сервисов и модулей через командную строку
– **TypeScript-first**: Полная поддержка статической типизации ”из коробки”

Преимущества для образовательных платформ:

– Строгая структура проекта для командной разработки
– Встроенная поддержка форм с валидацией
– Готовые решения для маршрутизации и HTTP-клиента

Ограничения:

– Высокий порог входа из-за сложной терминологии (декораторы, зоны, сервисы)
– Большой размер бандла (до 500КБ в минимальной сборке)
– Жёсткие требования к архитектуре

3.5.4 Vue.js

Vue.js — прогрессивный фреймворк, сочетающий подходы React и Angular. Особенно эффективен для быстрого прототипирования и проектов средней сложности.

Основные характеристики:

– **Реактивная система**: Автоматическое отслеживание зависимостей данных
– **Гибкая интеграция**: Возможность использования как через CDN, так и в составе сложных SPA
– **Single-File Components**: Объединение шаблона, логики и стилей в одном .vue-файле
– **Переходные анимации**: Встроенная поддержка анимации состояний

Сильные стороны для учебных проектов:

- Понятная документация с интерактивными примерами
- Мягкая кривая обучения для начинающих
- Компактный размер ядра (24КБ в gzip)

Ограничения:

- Относительно небольшое сообщество по сравнению с React/Angular
- Ограниченные возможности SSR без Nuxt.js
- Меньший выбор готовых UI-библиотек

3.5.5 Сравнительный анализ фреймворков

Таблица 1 – Сравнение характеристик фреймворков

Параметр	React + Next.js	Angular	Vue.js
Кривая обучения	Средняя	Высокая	Средняя
Сообщество	Крупное	Крупное	Растущее
Производительность	Высокая	Средняя	Высокая
Гибкость архитектуры	Высокая	Минимальная	Средняя
Поддержка SSR	Встроенная (Next.js)	Встроенная	Nuxt.js
Поддержка TypeScript	Да	Да	Да
Готовая маршрутизация	Да (Next.js)	Да	Да (Vue Router)

Ключевые выводы:

- **React + Next.js**: Предпочтителен для современных, SEO-оптимизированных приложений с гибкой архитектурой и возможностью инкрементального масштабирования
- **Angular**: Подходит для крупных enterprise-систем с чёткими архитектурными требованиями и строгой типизацией
- **Vue.js**: Оптимален для быстрого старта, MVP и небольших команд с ограниченным опытом

Для образовательной платформы выбран стек **React + Next.js**, поскольку он:

– Обеспечивает SSR и SSG «из коробки», что критично для SEO
– Позволяет гибко комбинировать клиентскую и серверную логику
– Имеет развитую экосистему и отличную интеграцию с библиотеками (Auth.js, Redux, Socket.IO)
– Упрощает масштабирование и поддержку проекта в долгосрочной перспективе

3.5.6 Next.js

Next.js — это популярный фреймворк для React, который значительно расширяет его возможности, предоставляя разработчикам мощные инструменты для создания высокопроизводительных веб-приложений. Одной из ключевых особенностей Next.js является поддержка рендеринга на сервере (SSR, Server-Side Rendering) и статической генерации контента (SSG, Static Site Generation). Эти подходы позволяют улучшить производительность приложений, поскольку они обеспечивают быструю загрузку страниц, оптимизированную для поисковых систем и пользователей.

С серверным рендерингом Next.js позволяет генерировать HTML на сервере для каждой страницы перед её отправкой клиенту, что обеспечивает быстрое отображение контента. Это особенно полезно для SEO, поскольку поисковые системы могут индексировать контент сразу после его загрузки. Такой подход значительно улучшает видимость веб-приложений в поисковых системах и способствует их более высокому ранжированию.

Одним из ключевых преимуществ Next.js является автоматическая разбивка кода (code splitting). Это означает, что Next.js разделяет приложение на небольшие части, которые загружаются только по мере необходимости, что помогает сократить время загрузки страниц и улучшить пользовательский опыт. Таким образом, браузер загружает только тот код, который необходим для отображения текущей страницы, а не весь код приложения.

Кроме того, Next.js поддерживает гибкие методы рендеринга, что дает разработчикам возможность выбирать наиболее подходящий способ для каждой страницы. Статическая генерация (SSG) идеально подходит для страниц, которые не изменяются часто и могут быть сгенерированы заранее, например, блоговые записи или страницы с информацией о компании. В то время как для динамических стра-

ниц, которые требуют актуализации данных на сервере при каждом запросе, можно использовать серверный рендеринг.

Next.js также упрощает настройку маршрутизации и управление данными. Встроенная система маршрутизации автоматически генерирует страницы на основе файловой структуры, что делает создание новых страниц и маршрутов простым и интуитивно понятным. Кроме того, Next.js предоставляет инструменты для работы с API, что позволяет без труда интегрировать серверную логику в приложение.

Еще одной значимой особенностью является поддержка типизации с помощью TypeScript, что делает разработку в Next.js ещё более удобной и безопасной. Комбинация TypeScript и Next.js позволяет создавать стабильные и хорошо структурированные приложения, минимизируя количество ошибок на этапе разработки.

Важно отметить, что реализацию приложения можно было бы построить и на чистом React, однако в этом случае значительная часть функциональности, такой как маршрутизация, SSR, SSG и работа с API, потребовала бы ручной настройки и подключения дополнительных библиотек. Использование Next.js избавляет от необходимости собирать всё вручную и предоставляет готовую, хорошо спроектированную архитектуру. Таким образом, Next.js становится де-факто стандартом разработки современных React-приложений. Это не просто библиотека, а фреймворк — а значит, он предлагает определённую «протоптанную дорожку», следование которой позволяет создавать более надёжные и поддерживаемые решения.

3.5.7 Socket.IO

Socket.IO — это JavaScript-библиотека с открытым исходным кодом, предназначенная для реализации двустороннего взаимодействия между клиентом и сервером в режиме реального времени. Основной особенностью данной технологии является использование собственного протокола поверх WebSocket с возможностью автоматического переключения на альтернативные методы связи (long polling и др.) при отсутствии поддержки WebSocket.

Преимущества использования Socket.IO:

- **Гибкость**: Разработчики имеют полный контроль над архитектурой соединения, включая маршрутизацию сообщений, обработку событий, систему комнат (rooms) и пространств имён (namespaces).
- **Масштабируемость**: Поддержка кластеризации и горизонтального мас-

штабирования при помощи Redis-адаптеров.

- **Совместимость с Node.js**: Socket.IO органично интегрируется в стек на основе Node.js, что упрощает реализацию единой инфраструктуры.
- **Производительность**: Низкая задержка при передаче сообщений благодаря постоянному соединению между клиентом и сервером.
- **Интеграция с Python**: Для серверной части на Python существует аналогичная библиотека *python-socketio*, которая позволяет использовать те же возможности для реализации чатов и двустороннего общения между клиентом и сервером. Это делает возможным использование Socket.IO как на front-end (с помощью Node.js), так и на back-end (с помощью Python), обеспечивая совместимость и синхронизацию данных между клиентом и сервером.

Недостатки:

- Необходимость разработки и поддержки собственной серверной инфраструктуры.
- Повышенная сложность при масштабировании без использования внешних инструментов (Redis, Kubernetes).
- Отсутствие встроенной панели мониторинга или аналитики соединений.

Socket.IO предоставляет высокий уровень кастомизации и гибкости, что делает его предпочтительным выбором в проектах, где важна точная настройка логики взаимодействия в реальном времени.

3.5.8 Pusher

Pusher — это облачная платформа, предоставляющая API и SDK для реализации push-уведомлений и двусторонней передачи данных в реальном времени. В отличие от Socket.IO, Pusher представляет собой managed-сервис, абстрагирующий низкоуровневые детали инфраструктуры.

Преимущества использования Pusher:

- **Упрощённая интеграция**: SDK и готовые клиентские библиотеки позволяют быстро настроить соединение и передавать события.

– **Масштабируемость**: Обеспечивается на уровне платформы без участия разработчика.
– **Надёжность**: Pusher использует устойчивую облачную инфраструктуру с балансировкой нагрузки.
– **Аналитика и мониторинг**: Панель управления предоставляет данные о соединениях, событиях и каналах.

Ограничения:

– **Платная модель**: Бесплатный тариф ограничен по числу соединений и событий, что делает использование невыгодным при росте нагрузки.
– **Зависимость от стороннего сервиса**: Потенциальные риски, связанные с отказоустойчивостью внешнего провайдера.
– **Ограниченная кастомизация**: Структура событий и поведения определяется особенностями платформы.

Pusher является удобным решением для проектов с ограниченными временными ресурсами и отсутствием внутренней серверной инфраструктуры, однако его применимость в образовательных продуктах с ограниченным бюджетом вызывает сомнения.

3.5.9 WebSocket в JavaScript

JavaScript предоставляет встроенный класс WebSocket для организации двустороннего обмена данными между клиентом и сервером. Этот класс реализует базовую функциональность протокола WebSocket, предоставляя разработчикам простой способ обмена данными в реальном времени без необходимости использовать сторонние библиотеки. Однако, несмотря на свою доступность, использование WebSocket требует значительных усилий для реализации различных важных аспектов взаимодействия.

К примеру, при использовании WebSocket разработчик должен самостоятельно реализовать:

– переподключение сокета при его падении,
– буферизацию сообщений в случае разрыва соединения,
– обработку таймаутов и ошибок,

– поддержку различных сред (например, Node.js и браузер),

– масштабирование на сервере.

Таким образом, несмотря на его доступность и гибкость, использование WebSocket требует написания значительного объема дополнительной логики. Это делает его менее удобным для быстрого внедрения в проект, особенно в случае масштабируемых приложений.

С учетом всех этих факторов, было принято решение отказаться от использования низкоуровневого WebSocket в пользу более высокоуровневых решений, таких как Socket.IO или Pusher, которые предоставляют необходимую функциональность и обрабатывают множество нюансов «из коробки», позволяя сосредоточиться на бизнес-логике приложения.

3.5.10 Сравнение и выбор технологии для чатов

При сравнении библиотек Socket.IO и Pusher необходимо учитывать как технические, так и организационные аспекты. Таблица 2 представляет краткое сопоставление ключевых параметров:

Таблица 2 – Сравнение технологий для реализации чатов в реальном времени

Критерий	**Socket.IO**	**Pusher**
Тип решения	Open-source библиотека	Облачный managed-сервис
Контроль над архитектурой	Полный	Ограниченный
Поддержка масштабирования	Через Redis и кластеризацию	Встроенная на уровне платформы
Простота настройки	Средняя (требует сервера)	Высокая (SDK)
Затраты на использование	Бесплатно	Платная модель
Интеграция с Node.js	Нативная	Через SDK
Надёжность соединения	Высокая	Высокая
Кастомизация протокола	Да	Нет

С учётом специфики проекта — ограниченного бюджета, необходимости полной кастомизации и тесной интеграции с Node.js-сервером — наилучшим выбором является использование **Socket.IO**. Данная библиотека предоставляет все необходимые механизмы для реализации масштабируемой и надёжной чат-системы, при этом позволяя оптимизировать производительность без привлечения сторонних сервисов.

Более того, благодаря открытому коду, Socket.IO не ограничивает разработчика в выборе архитектурных решений, а также обеспечивает

возможность расширения функционала в будущем. В условиях ограниченных ресурсов образовательной платформы такой подход оказывается наиболее целесообразным.

3.5.11 Auth.js

Auth.js — это библиотека для реализации аутентификации и авторизации в веб-приложениях. Она является официальным решением, рекомендуемым и поддерживаемым фреймворком Next.js, что гарантирует хорошую интеграцию и поддержку всех необходимых функций. Библиотека позволяет легко подключать сторонние провайдеры аутентификации, такие как Google, Facebook и другие, а также реализовывать собственную аутентификацию с использованием базы данных. Auth.js обеспечивает надежную защиту пользовательских данных, управление сессиями, работу с токенами и предоставляет удобные API для быстрой настройки. Это решение упрощает реализацию всех ключевых механизмов безопасности, освобождая разработчиков от необходимости погружаться в тонкости реализации.

3.5.12 Redux

Redux — это библиотека для управления состоянием в приложениях, основанных на React. Она используется для централизованного хранения состояния приложения, что облегчает обмен данными между компонентами и упрощает их взаимодействие. Redux помогает избежать ”проблемы пропс-дерева”в больших приложениях, когда передача данных через множество вложенных компонентов становится сложной. Хотя Redux часто используется в более сложных приложениях, в данном проекте его роль заключается в том, чтобы сделать взаимодействие между компонентами более организованным и улучшить предсказуемость состояния приложения.

3.6 Требования к функциональности приложения

Разрабатываемое приложение представляет собой образовательную платформу, ориентированную на университетскую среду. Основная цель — предоставить единое пространство для организации учебного процесса, взаимодействия между преподавателями и студентами, а также управления учебными структурами.

3.6.1 Регистрация и структура университетов

Каждый университет имеет возможность зарегистрироваться на платформе и получить доступ к собственной административной панели. Через неё администраторы могут создавать внутреннюю структуру: институты, кафедры и учебные группы. Эти сущности используются как основа для управления доступом, назначения преподавателей и приглашения студентов.

3.6.2 Панель преподавателя

Преподаватели, закреплённые за кафедрами, получают доступ ко всем учебным группам, относящимся к соответствующей кафедре. Через панель преподавателя доступен следующий функционал:

- создание групповых чатов для любой группы своей кафедры;
- размещение учебных материалов — как в групповых чатах, так и в личных сообщениях;
- формирование и отправка заданий для студентов;
- просмотр и анализ результатов выполнения заданий;
- предоставление обратной связи студентам.

3.6.3 Панель студента

Студенты, присоединенные к определенным группам, имеют доступ к:

- групповым чатам своей учебной группы;
- личной переписке с преподавателями;
- материалам, отправленным преподавателями;
- заданиям, опубликованным в рамках их группы;
- форме отправки решений и получению обратной связи.

3.7 Диаграмма вариантов использования

На рисунке представлена диаграмма вариантов использования, демонстрирующая взаимодействие пользователей с системой. Каждый пользователь (актёр) обладает определённым набором действий, доступных в рамках его роли:

– **Преподаватель** — создание и управление группами и заданиями, взаимодействие со студентами в чатах, а также анализ кода лабораторных работ с использованием AI;
– **Студент** — участие в групповых и личных чатах, выполнение заданий и просмотр результатов;
– **Администратор** — управление институтами, кафедрами, группами, студентами и преподавателями, а также создание приглашений в систему.

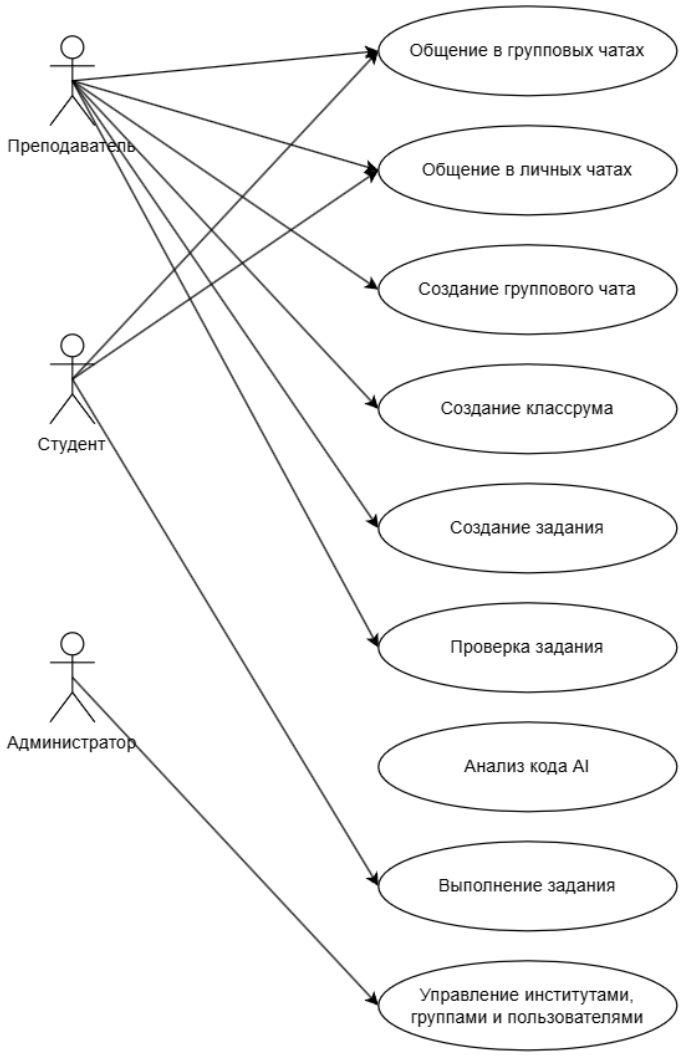

Рисунок 1 – Диаграмма вариантов использования системы для различных ролей пользователей.

3.8 Коммуникация и взаимодействие

Ключевым элементом платформы является система обмена сообщениями, включающая:

- **Групповые чаты**:
  - Общение участников учебной группы в режиме реального времени
  - Передача файлов (форматы: .jpg, .png, .mp4, .pdf, .zip и др.)
  - Привязка к группам
- **Личная переписка**:
  - Обмен сообщениями один на один (студент-преподаватель или студент-студент)
  - Передача файлов тех же форматов, что и в групповых чатах

Обе системы поддерживают базовые функции:

- Отображение истории сообщений
- Индикаторы прочтения сообщений
- Поиск по тексту переписки
- Уведомления о новых сообщениях

Таким образом, платформа предоставляет функциональность, охватывающую весь цикл учебной коммуникации — от административного управления структурами до взаимодействия по заданиям и материалам между преподавателями и студентами.

3.9 Безопасность данных

3.9.1 Авторизация с использованием JWT и Auth.js

Механизм авторизации в приложении реализован на основе JSON Web Token (JWT) и библиотеки Auth.js, которая обеспечивает безопасную и гибкую аутентификацию пользователей на стороне front-end. Процесс состоит из следующих этапов:

а) **Аутентификация пользователя**:

- Пользователь выполняет вход с помощью логина/пароля или через одного из OAuth-провайдеров (например, Google).
- Auth.js инициирует процесс аутентификации и, при успешной проверке, получает JWT.

б) **Работа с токеном**:

- Полученный JWT содержит минимальный необходимый payload (например, идентификатор пользователя, роль и срок действия).
- Токен сохраняется в HTTP-only cookie с флагами Secure и SameSite=Strict, что предотвращает XSS- и CSRF-атаки.

в) **Доступ к защищённым ресурсам**:

- При обращении к API, front-end автоматически прикрепляет токен к запросу.
- При недействительном или истёкшем токене, Auth.js может автоматически обновить его через refresh-токен, если он присутствует.

3.9.2 Роль Auth.js

Библиотека Auth.js упрощает реализацию безопасной авторизации за счёт следующих возможностей:

а) **Инкапсуляция логики**:

- Обрабатывает все основные сценарии авторизации (вход, выход, обновление токена).
- Управляет хранением и безопасной передачей токенов.

б) **Поддержка современных стандартов**:

- Генерирует CSRF-токены.
- Поддерживает стратегию Single Sign-On (SSO).

3.9.3 Меры защиты

Для повышения безопасности механизма авторизации реализованы дополнительные меры:

а) **JWT**:

- Access-токен действует ограниченное время (например, 5 минут), а refresh-токен — 30 дней.

б) **Auth.js**:

- Валидирует параметры входа, включая redirect URI.

Таким образом, связка JWT и Auth.js позволяет реализовать надёжный и гибкий механизм авторизации на стороне front-end с минимальной утечкой чувствительных данных.

3.10 Система создания заданий с AI-анализом кода

3.10.1 Общее описание модуля

Разработанная система позволяет преподавателям создавать виртуальные классрумы, выдавать задания и автоматически анализировать решения студентов с использованием AI-инструментов. Это аналог образовательной платформы (например, Google Classroom), ориентированный на технические дисциплины с программированием.

Основные функции:

- Создание виртуального класса преподавателем
- Назначение заданий с параметрами оценки
- Загрузка решений студентами
- Получение отчётов об автоматическом AI-анализе кода
- Просмотр статистики и аналитики преподавателем

3.10.2 Функциональные возможности

Для преподавателей:

- Создание и управление классами
- Назначение заданий
- Просмотр AI-отчётов по каждому студенту
- Сводная статистика по группе

Для студентов:

- Просмотр активных заданий и дедлайнов
- Загрузка решения

3.10.3 AI-анализ кода

После загрузки решения студентом система автоматически выполняет его анализ с использованием инструментов искусственного интеллекта. Проверка охватывает корректность, читаемость, соответствие заданию и уровень оригинальности кода.

На основе заданных преподавателем критериев формируется интерактивный отчёт, включающий:

- Комментарии и замечания по структуре и стилю кода
- Оценку соответствия решению поставленным требованиям

3.11 Выводы

В результате проведённого анализа можно сделать вывод о наличии устойчивого запроса на интегрированное образовательное приложение, способное решать сразу несколько ключевых задач. Современные платформы зачастую фокусируются либо на предоставлении учебных материалов, либо на коммуникации, либо на автоматизации проверки знаний, при этом разрозненность этих функций создаёт неудобства для всех участников образовательного процесса.

Потребности преподавателей включают в себя удобное управление группами и заданиями, возможность оперативной обратной связи, загрузку и распространение материалов. Студентам, в свою очередь, важно иметь

стабильный и понятный доступ к заданиям, личным сообщениям и учебным ресурсам, а также возможность взаимодействовать с преподавателями и одногруппниками в привычном цифровом формате.

Предлагаемое приложение должно закрыть этот разрыв, обеспечив единую среду, в которой объединены функции управления учебным процессом, общения, публикации и проверки заданий. Такой подход позволит повысить качество образовательного взаимодействия, сократить технические барьеры и обеспечить более высокую степень вовлечённости пользователей.

Таким образом, на основании проведённого анализа подтверждается необходимость разработки новой системы, в которой ключевые элементы образовательной среды будут интегрированы в одно приложение, удовлетворяющее современным требованиям пользователей.

4 ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ ПРОЕКТА

4.1 Архитектура клиентской части системы

В качестве архитектурного подхода к проектированию клиентской части приложения выбран **Feature-Sliced Design (FSD)** — современный метод модульной архитектуры, ориентированный на frontend-приложения, использующие стек React, Redux, TypeScript, Next.js и др.

4.1.1 Общее описание Feature-Sliced Design

Feature-Sliced Design (FSD) — это архитектурный подход, в котором структура проекта строится вокруг **бизнес-логики и функциональных сценариев**, а не вокруг технических понятий вроде компонентов или страниц. Вместо вертикального деления на слои, FSD предлагает горизонтальное деление на сегменты (срезы), соответствующие смысловым блокам приложения.

Целью FSD является обеспечение **масштабируемости, читаемости и модульности** клиентской части. Он упрощает командную разработку, делает код более устойчивым к изменениям и облегчает тестирование.

4.1.2 Слои архитектуры FSD

В FSD определены следующие ключевые слои, каждый из которых имеет своё назначение и строгие границы ответственности:

Таблица 3 – Слои архитектуры FSD

Слой	**Назначение**
app	Точка входа в приложение, глобальные конфигурации, роутинг, провайдеры
processes	Бизнес-процессы, объединяющие несколько фич в сценарии (опционально)
pages	Отдельные страницы приложения, собирают из виджетов и фич
widgets	Крупные интерфейсные блоки, объединяющие фичи и сущности
features	Отдельные бизнес-функции: авторизация, отправка сообщений, регистрация и т.д.
entities	Базовые предметные сущности: пользователь, задание, сообщение и т.д.
shared	Универсальные модули: UI-компоненты, утилиты, темы, типы и пр.

4.1.3 Описание слоёв

Слой app. Содержит глобальную инициализацию: конфигурацию провайдеров (AuthProvider, StoreProvider), роутинг, глобальные стили и интеграцию с внешними сервисами.

Слой shared. Хранилище нейтральных, переиспользуемых компонентов и утилит, не зависящих от предметной области. Примеры: Button, Input, formatDate(), UserRole.

Слой entities. Описывает предметные сущности и их внутреннюю логику. Каждая сущность содержит типизацию, модели, UI-компоненты (например, UserCard), работу с хранилищем данных.

Слой features. Бизнес-функциональность, реализующая действия пользователя. Примеры: sendMessage, submitTask, loginUser. Каждая фича использует одну или несколько сущностей.

Слой widgets. Крупные интерфейсные блоки, собирающие интерфейс из сущностей и фич. Например: ChatWindow, TaskList, Sidebar.

Слой pages. Реализует страницы приложения. Каждая страница включает в себя нужные виджеты и фичи, но не содержит логики.

Слой processes. (Опционально) Используется для объединения фич и сущностей в единые бизнес-сценарии, например: процесс регистрации нового пользователя.

4.2 Пример структуры проекта

4.2.1 Преимущества применения FSD

- Чёткое разделение ответственности и слабая связность модулей;
- Масштабируемость проекта без деградации архитектуры;
- Упрощённое тестирование и переиспользование;
- Быстрое включение новых разработчиков в проект;
- Возможность гибкой миграции между проектами с похожей архитектурой.

Применение FSD позволило реализовать гибкую и масштабируемую архитектуру клиентской части, обеспечив стабильность, понятность и надёжность структуры при росте количества модулей и функционала.

5 Список литературы

6 Приложения