Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) ООО ”МЦОБ. Онлайн-сервисы”

наименование предприятия, организации, учреждения

Студента 4 курса, группы ПО-11б

курса, группы

Геворкян Арнольда Юрьевича

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

директор

должность, звание, степень

Куркина А. В.

фамилия и. о.

Оценка _____

_____

подпись, дата

Руководитель практики от университета

к.т.н. доцент

должность, звание, степень

Чаплыгин А. А.

фамилия и. о.

Оценка _____

_____

подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2025 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных.

IT (ИТ) – информационные технологии.

ПО – программное обеспечение.

СУБД – система управления базами данных.

1 Анализ предметной области

1.1 Описание предметной области

В современных условиях быстро меняющегося рынка и возрастающей сложности проектов, эффективное управление проектами становится ключевым фактором успеха для многих организаций [1]. Компании постоянно ищут способы оптимизации процессов, сокращения сроков выполнения задач и повышения качества конечного продукта или услуги.

На сегодняшний день проблема неэффективного управления проектами и срыва сроков остается крайне актуальной. По данным различных исследований, значительная доля проектов не укладывается в первоначальные бюджеты, превышает запланированные сроки или не достигает поставленных целей в полном объеме [2]. В России, как и во всем мире, компании сталкиваются с потерями из-за недостаточной гибкости управления, слабой координации команд и отсутствия прозрачности в ходе выполнения работ. Неспособность быстро адаптироваться к изменениям требований, неэффективное распределение ресурсов [3] и трудности в отслеживании прогресса являются частыми причинами неудач.

Традиционные подходы к управлению проектами включают в себя несколько методологий:

1. Каскадная модель: этот подход характеризуется строгой последовательностью этапов выполнения проекта (анализ требований, проектирование, реализация, тестирование, внедрение). Планирование осуществляется детально на начальном этапе, и изменения в ходе проекта не приветствуются [2].

2. Методы, основанные на стандартах PMBOK или PRINCE2: данные подходы предлагают структурированные фреймворки, наборы процессов и областей знаний для управления проектами. Они ориентированы на формализацию и контроль всех аспектов проекта [1].

3. Управление на основе личного опыта и неформальных договоренностей: в некоторых, особенно небольших, командах или проектах управление

может осуществляться без строгой методологии, опираясь на опыт руководителя и устные договоренности.

Однако, традиционные и строго регламентированные методы управления проектами часто демонстрируют недостаточную гибкость в условиях высокой неопределенности и частых изменений требований [6], характерных для современной разработки программного обеспечения и других инновационных сфер. Они могут приводить к затягиванию сроков, увеличению бюрократии и снижению мотивации команды. Отсутствие же формализованных подходов ведет к хаосу и потере контроля над проектом. Поэтому использование гибких методологий, таких как Agile [14], и инструментов визуализации и управления потоком работ, таких как Kanban-доски [16], может значительно повысить адаптивность, прозрачность и эффективность управления проектами.

1.2 Методология Agile

Методология Agile представляет собой итеративный и гибкий подход к управлению проектами, в первую очередь к разработке программного обеспечения [11]. В отличие от традиционных каскадных моделей, где каждый этап выполняется последовательно и полностью перед началом следующего, Agile делает упор на постепенное развитие продукта через короткие циклы и постоянную обратную связь [15]. Основная цель Agile – быстро адаптироваться к изменениям требований, обеспечивать высокое качество продукта и удовлетворенность заказчика. Ключевые принципы Agile включают [43]:

1. Люди и взаимодействие важнее процессов и инструментов: подчеркивается важность командной работы, общения и сотрудничества [15].

2. Работающий продукт важнее исчерпывающей документации: акцент делается на создании функционального продукта, а не на чрезмерном документировании.

3. Сотрудничество с заказчиком важнее согласования условий контракта: подразумевается тесное взаимодействие с заказчиком на протяжении всего проекта для уточнения требований и получения обратной связи [12].

4. Готовность к изменениям важнее следования первоначальному плану: Agile-команды готовы адаптироваться к новым требованиям и изменениям в приоритетах.

Центральным элементом многих Agile-фреймворков (таких как Scrum) является понятие спринта [11]. Спринт – это короткий, ограниченный по времени период (обычно от одной до четырех недель), в течение которого команда разработчиков создает определенный, заранее согласованный объем функциональности продукта. Каждый спринт имеет четко определенную цель и набор задач, которые должны быть выполнены к его завершению.

Основные характеристики спринта:

1. Фиксированная длительность: продолжительность спринта устанавливается в начале проекта и остается неизменной для всех последующих спринтов. Это помогает команде выработать ритм и предсказуемость.

2. Цель спринта: каждый спринт направлен на достижение конкретной, измеримой цели, которая вносит вклад в общую цель продукта.

3. Бэклог спринта: в начале каждого спринта команда выбирает задачи из общего бэклога продукта – списка всех требуемых функций и улучшений – и формирует бэклог спринта. Это список задач, которые команда обязуется выполнить в течение текущего спринта.

4. Ежедневные встречи: короткие ежедневные совещания, на которых команда синхронизируется, обсуждает прогресс, выявляет препятствия и планирует работу на ближайшие 24 часа.

5. Обзор спринта: в конце спринта команда демонстрирует заказчику и другим заинтересованным сторонам созданный инкремент продукта (работающую функциональность). Цель – получить обратную связь.

6. Ретроспектива спринта: после обзора спринта команда проводит внутреннее обсуждение, чтобы проанализировать, что прошло хорошо, что можно улучшить в процессах работы, и планирует конкретные действия по улучшению для следующего спринта [13].

Спринты обеспечивают структуру для итеративной разработки [15], позволяя команде регулярно поставлять работающие части продукта, полу-

чать обратную связь и адаптироваться к изменениям, что является основой гибкости Agile-подхода.

1.3 Методология Kanban

Kanban (с японского ”сигнальная доска”или ”визуальный сигнал”) – это гибкая методология управления рабочими процессами, нацеленная на постепенное улучшение существующей системы работы и повышение ее эффективности [16]. Изначально разработанная для оптимизации производства в компании Toyota, сегодня Kanban успешно применяется в самых разных сферах, особенно в разработке программного обеспечения, IT-операциях и управлении проектами [19]. Главная цель Kanban – обеспечить плавный, предсказуемый и эффективный поток выполнения задач, минимизируя при этом потери и перегрузку команды.

Методология Kanban базируется на нескольких ключевых принципах [44]:

1. Начать с того, что есть сейчас: Kanban не требует немедленных кардинальных изменений существующих процессов, ролей или обязанностей. Он применяется ”поверх”текущей системы.

2. Стремиться к постепенным, эволюционным изменениям: Kanban поощряет небольшие, но постоянные улучшения, которые легче принимаются командой и менее рискованны.

3. Уважать текущие процессы, роли и обязанности: Kanban признает ценность существующих структур и не стремится их разрушить без необходимости.

4. Поощрять лидерство на всех уровнях: Каждый член команды может и должен вносить вклад в улучшение процессов.

Центральным инструментом и наиболее узнаваемым элементом методологии Kanban является Kanban-доска [17]. Это визуальное представление всего рабочего процесса, которое делает работу и ее статус видимыми для всей команды и заинтересованных сторон. Как устроена и работает Kanban-доска [18]:

1. Колонки: доска разделена на вертикальные колонки, каждая из которых представляет определенный этап рабочего процесса. Набор колонок адаптируется под конкретный процесс команды.

2. Карточки: каждая рабочая задача или элемент представляется отдельной карточкой. Карточка содержит информацию о задаче и перемещается по доске слева направо по мере прохождения этапов.

3. WIP-лимиты: одна из ключевых практик Kanban – это установление максимального количества задач, которые могут одновременно находиться в определенной колонке [16]. Эти лимиты явно указываются на доске. WIP-лимиты помогают предотвратить перегрузку команды, выявлять ”узкие места”и фокусироваться на завершении начатой работы, а не на старте новой. Это улучшает поток и сокращает время выполнения задач.

4. Поток: движение карточек по доске визуализирует поток работы. Цель – сделать этот поток как можно более плавным, быстрым и предсказуемым. Команда отслеживает, как задачи проходят через систему, и ищет способы устранения задержек и блокировок.

5. Явные политики: правила работы часто делаются явными и размещаются на доске или обсуждаются командой.

Преимущества использования Kanban:

1. Прозрачность: все видят текущее состояние дел, кто над чем работает и где возникают проблемы.

2. Улучшение коммуникации и сотрудничества: доска служит общим центром информации и обсуждений.

3. Выявление ”узких мест”: WIP-лимиты помогают быстро обнаружить этапы, замедляющие общий процесс.

4. Сокращение времени цикла: фокус на потоке и ограничении WIP ускоряет выполнение задач.

5. Гибкость: Kanban легко адаптируется к различным процессам и может использоваться совместно с другими методологиями.

6. Снижение стресса: ограничение многозадачности помогает команде работать более сосредоточенно.

7. Стимулирование непрерывного улучшения: визуализация и регулярный анализ потока побуждают команду постоянно искать способы оптимизации своей работы.

1.4 IT в России

Российская сфера информационных технологий прошла сложный и динамичный путь развития, отражающий как глобальные технологические тренды, так и уникальные национальные особенности. История IT-проектов в России – это повесть о переходе от централизованных государственных инициатив советской эпохи к формированию конкурентного рынка и активному стремлению к цифровому суверенитету в наши дни.

На ранних этапах, во времена СССР, IT-проекты были преимущественно сосредоточены в оборонной промышленности, науке и государственном управлении. Разработка автоматизированных систем управления (АСУ) для предприятий, создание вычислительных центров и специализированного программного обеспечения велись в рамках плановой экономики. Эти проекты отличались масштабностью, но зачастую инертностью внедрения и ограниченной гибкостью [3]. Недостаток конкуренции и изолированность от мирового IT-рынка сдерживали темпы инноваций в гражданском секторе.

Переход к рыночной экономике в 1990-е годы открыл новую страницу. Появились первые частные IT-компании, ориентированные на коммерческие заказы. Началась активная компьютеризация предприятий, банковского сектора и торговли. В этот период IT-проекты часто были связаны с внедрением зарубежных программных и аппаратных решений, адаптацией их к российским реалиям. Это было время накопления опыта, формирования кадрового потенциала и становления основ отечественного IT-рынка. Возникли компании, ставшие впоследствии лидерами индустрии, например, в области разработки антивирусного ПО, поисковых систем и системной интеграции.

Начало 2000-х годов ознаменовалось ростом российской экономики и увеличением инвестиций в информационные технологии. Государство стало проявлять все больший интерес к цифровизации, инициируя крупные наци-

ональные проекты. Одним из знаковых направлений стало создание ”Электронного правительства нацеленного на повышение доступности и качества государственных услуг для граждан и бизнеса. Развивались системы межведомственного электронного взаимодействия, порталы госуслуг, электронный документооборот. Параллельно активно росли коммерческие IT-проекты, особенно в сферах телекоммуникаций, интернет-сервисов, электронной коммерции и разработки программного обеспечения на заказ [5]. Российские разработчики завоевали признание на международном уровне в таких областях, как разработка игр, офшорное программирование и наукоемкие программные решения.

В последние годы IT-проекты в России развиваются под сильным влиянием глобальных тенденций, таких как распространение облачных технологий [10], больших данных, искусственного интеллекта, мобильных приложений и интернета вещей. Однако на первый план все активнее выходит задача обеспечения цифрового суверенитета и импортозамещения. В ответ на внешние вызовы государство и бизнес наращивают усилия по созданию отечественных программных и аппаратных платформ, операционных систем, систем управления базами данных и бизнес-приложений. Яркими примерами таких проектов являются национальная платежная система ”Мир развитие отечественных операционных систем на базе Linux, а также государственные инициативы по поддержке разработки российского ПО и микроэлектроники.

Особое внимание уделяется проектам в области кибербезопасности, что обусловлено как ростом глобальных киберугроз, так и стремлением защитить критическую информационную инфраструктуру страны. Активно развиваются проекты, связанные с цифровизацией промышленности, сельского хозяйства, здравоохранения и образования. Внедрение технологий искусственного интеллекта становится приоритетным направлением во многих отраслях, от финансового сектора до государственного управления.

Современные IT-проекты в России характеризуются растущей сложностью, необходимостью интеграции разнообразных технологий и высоким уровнем конкуренции [2]. Несмотря на существующие вызовы, такие как кад-

ровый голод в отдельных сегментах и необходимость адаптации к меняющимся экономическим условиям, российская IT-отрасль демонстрирует значительный потенциал для дальнейшего роста и инноваций, играя все более важную роль в развитии страны. Фокус на собственных разработках и стремление к технологической независимости определяют ключевые векторы развития IT-проектов в России на ближайшие годы.

1.5 Динамика и перспективы развития IT-сферы

Российская IT-сфера, по состоянию на май 2025 года, переживает период активной трансформации, где эффективное управление проектами становится залогом успеха в условиях курса на цифровой суверенитет и адаптации к новым экономическим реалиям [1]. Динамика отрасли характеризуется как масштабными государственными инициативами, так и гибкостью коммерческого сектора.

В управлении IT-проектами наблюдается сосуществование различных подходов. Крупные государственные проекты, особенно в сфере импортозамещения и развития критической инфраструктуры, по-прежнему требуют структурированного планирования и контроля, часто опираясь на каскадные или гибридные модели [2]. Одновременно частный бизнес и IT-компании активно применяют гибкие методологии, такие как Agile, для быстрой разработки и вывода продуктов на рынок [14]. Растет значение гибридных подходов, сочетающих дисциплину традиционного управления с адаптивностью гибких практик, что требует от менеджеров проектов высокой квалификации и умения подбирать оптимальные инструменты [8]. Основными вызовами остаются обеспечение качества в сжатые сроки [4], управление ресурсами и адаптация к высокой степени неопределенности [3]. Государственная поддержка IT стимулирует проектную деятельность, но и повышает требования к прозрачности и эффективности управления.

Перспективы российской IT-отрасли тесно связаны с развитием отечественных программных продуктов, аппаратных решений и цифровых платформ. Это формирует устойчивый спрос на IT-проекты в таких областях, как

разработка системного и прикладного ПО [41], кибербезопасность и облачные сервисы [10]. Роль управления проектами в этих условиях только усиливается: востребованы специалисты, способные вести сложные комплексные проекты, управлять портфелями и обеспечивать эффективное взаимодействие команд. Ожидается дальнейшая цифровизация самого процесса управления проектами за счет внедрения специализированных ИС и аналитических инструментов [42]. Ключевыми компетенциями становятся управление рисками и глубокие знания в предметных областях реализуемых проектов [6].

Несмотря на существующие вызовы, российская IT-сфера демонстрирует потенциал к развитию, где профессиональное управление проектами играет критически важную роль. По мере дальнейшего технологического прогресса, включая интеграцию решений на базе искусственного интеллекта и нейронных сетей, будут появляться новые типы IT-проектов, ставящие перед менеджерами еще более сложные и интересные задачи.

2 Техническое задание

2.1 Основание для разработки

Полное наименование системы: «Программная реализация системы управления проектами». Основанием для разработки программы является приказ ректора ЮЗГУ от «17» апреля 2025 г. № 1828-с приказа «О направлении (допуске) на практику».

2.2 Цель и назначение разработки

Система управления проектами предназначена для эффективной организации, отслеживания и координации рабочих процессов команд, использующих гибкие методологии Agile и, в частности, Kanban-подход, с целью повышения прозрачности, предсказуемости и продуктивности выполнения проектов. Пользователи (включая менеджеров проектов, владельцев продуктов, разработчиков и других членов команды) должны иметь возможность визуализировать этапы работы с помощью настраиваемых Kanban-досок, управлять потоком задач от создания до завершения, отслеживать прогресс в реальном времени и анализировать эффективность процессов для непрерывного улучшения. Система должна предоставлять интуитивно понятный интерфейс для совместной работы и обеспечивать легкий доступ ко всей необходимой проектной информации. Задачами данной разработки являются:

1. Создание основной архитектуры и серверной логики для управления проектами, Kanban-досками, задачами (карточками), пользователями и их ролями.

2. Разработка веб-интерфейса пользователя с интерактивной Kanban-доской, поддерживающей создание и настройку колонок, перемещение задач, установку WIP-лимитов и визуализацию статусов задач.

3. Реализация функционала управления задачами, включая их создание, редактирование, назначение ответственных, установку приоритетов, сроков выполнения, добавление описаний, чек-листов, меток (тегов) и прикрепление файлов.

4. Интеграция инструментов для поддержки Agile-практик, таких как управление бэклогом продукта, планирование итераций, а также базовые метрики для отслеживания производительности.

5. Разработка системы уведомлений и средств для совместной работы, включая комментирование задач, упоминания пользователей и отслеживание истории изменений.

6. Обеспечение базовых функций администрирования системы, включая управление пользователями, настройку прав доступа и управление проектными пространствами.

2.3 Актуальность темы разработки

В условиях современной цифровой экономики, характеризующейся высокой скоростью изменений, глобализацией рынков и постоянно растущей конкуренцией, способность компаний быстро и эффективно реализовывать проекты становится критически важным фактором успеха. Особенно остро это проявляется в сфере информационных технологий и разработки программного обеспечения, где жизненные циклы продуктов сокращаются, а требования заказчиков и пользователей эволюционируют с беспрецедентной скоростью. По состоянию на 2025 год, организации всех размеров активно ищут инструменты и подходы, позволяющие им не только управлять проектами, но и гибко адаптироваться к непредвиденным обстоятельствам, сохраняя при этом фокус на создании ценности.

Актуальность разработки такой системы обусловлена, прежде всего, повышенным спросом на гибкость и адаптивность, поскольку современные проекты требуют от команд способности быстро перестраиваться, изменять приоритеты и инкорпорировать новые требования без значительных потерь времени и ресурсов, чему идеально отвечает комбинированный подход Agile и Kanban. Эта потребность в адаптивных инструментах становится особенно острой в условиях роста популярности удаленных и гибридных моделей работы, что порождает необходимость в эффективных цифровых решениях, способных обеспечить прозрачность, синхронизацию и эффективную колла-

борацию всех участников проекта независимо от их местоположения; здесь программно реализованная Kanban-доска выступает как единый источник правды для команды. Кроме того, интегрированная система напрямую способствует стремлению к повышению производительности и сокращению потерь, позволяя не только визуализировать задачи, но и отслеживать метрики производительности, такие как время цикла и пропускная способность, а также применять WIP-лимиты для предотвращения перегрузок и фокусировки на завершении начатой работы, что ведет к повышению общей эффективности.

Несмотря на наличие на рынке множества систем управления проектами, сохраняется выраженный запрос на инструменты, которые можно легко адаптировать под специфические процессы конкретной команды или организации, и которые обладают интуитивно понятным интерфейсом и низким порогом вхождения для пользователей. В конечном итоге, предлагаемая программная система поддерживает культуру непрерывного улучшения, поощряемую как Agile, так и Kanban, предоставляя ценные данные для проведения ретроспектив и анализа эффективности потока работ.

2.4 Этапы разработки

Для реализации программной системы предполагается выполнение следующих этапов:

1. Анализ предметной области и определение ключевых требований к функционалу Kanban-доски и интеграции Agile-методик.
2. Проектирование архитектуры приложения.
3. Разработка структуры базы данных для хранения информации об основных сущностях системы, их состояниях и прочей технической информации.
4. Реализация серверной логики и API для управления основными сущностями системы.
5. Создание пользовательского интерфейса для удобного пользования приложением.

6. Проведение функционального тестирования для выявления критичных несоответствий.

2.5 Требования к программной системе

2.5.1 Требования к данным

Входными данными для системы являются:

– учетные данные пользователя;

– параметры задач, проектов, досок;

– комментарии к задачам.

Выходными данными для системы являются:

– полная и детализированная информация о пользователях, задачах, проектах, досках;

– системные уведомления и информационные сообщения.

2.5.2 Функциональные требования

В разрабатываемой системе управления проектами должно быть предусмотрено наличие трех основных ролей пользователей с различными наборами прав и функциональных возможностей в рамках каждого проекта: «Участник проекта», «Владелец проекта» и «Администратор».

Пользователю с ролью «Участник проекта» должны быть доступны следующие функции:

– регистрация и аутентификация;

– просмотр и изменение персональной информации;

– удаление профиля;

– просмотр проектов, в которых он состоит, и существующих в них досок;

– просмотр, создание, редактирование, перемещение всех задач;

– удаление собственных задач.

Пользователю с ролью «Владелец проекта» должны быть доступны следующие функции:

– регистрация и аутентификация;

– просмотр и изменение персональной информации;

– удаление профиля;

– просмотр, создание, редактирование, удаление собственных проектов и досок;

– просмотр, создание, редактирование, перемещение, удаление любых задач в собственных проектах независимо от автора;

– управление участниками собственных проектов.

Пользователю с ролью «Администратор» должны быть доступны следующие функции:

– регистрация и аутентификация;

– просмотр и изменение персональной информации;

– удаление профиля;

– просмотр, создание, редактирование, удаление всех существующих проектов, досок и задач (а также их перемещение);

– управление участниками всех проектов.

На рисунке 2.1 представлена диаграмма прецедентов.

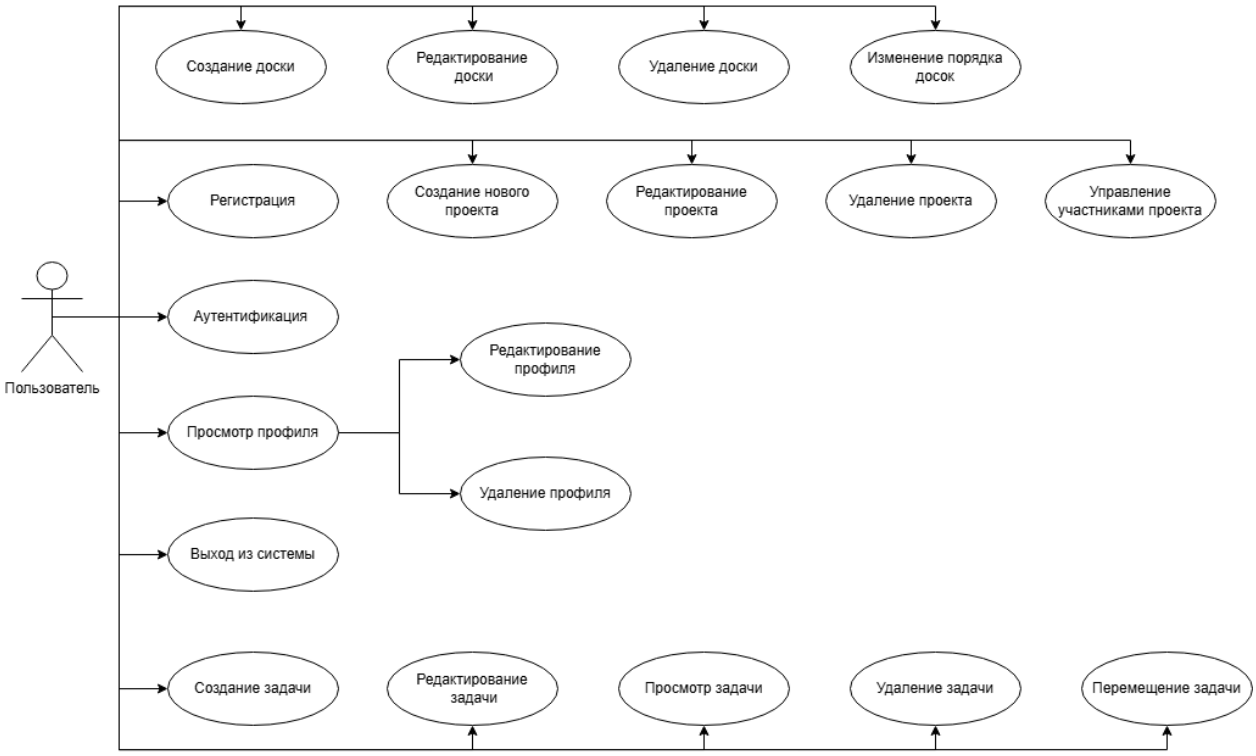

Рисунок 2.1 – Диаграмма прецедентов

2.5.3 Сценарий прецедента «Регистрация»

Основной успешный сценарий:

1. Пользователь переходит на страницу аутентификации web-приложения.
2. Пользователь нажимает на ссылку/кнопку «Создать».
3. Пользователь вводит свое имя в поле «Имя».
4. Пользователь вводит свою фамилию в поле «Фамилия».
5. Пользователь вводит свой адрес электронной почты в поле «Адрес электронной почты».
6. Пользователь вводит свой пароль в поле «Пароль», соблюдая указанные критерии.
7. Пользователь подтверждает свой пароль в поле «Подтвердите пароль».
8. Пользователь нажимает кнопку «Создать аккаунт».
9. Система проверяет корректность введенных данных и уникальность email.

10. Система регистрирует нового пользователя.

11. Система автоматически аутентифицирует пользователя, сохраняет токен и перенаправляет его на основной интерфейс приложения (доску Kanban).

2.5.4 Сценарий прецедента «Аутентификация»

Основной успешный сценарий:

1. Пользователь переходит на страницу аутентификации web-приложения.

2. Пользователь вводит свой адрес электронной почты в поле «Адрес электронной почты».

3. Пользователь вводит свой пароль в поле «Пароль».

4. Пользователь нажимает кнопку «Войти».

5. Система проверяет корректность введенных учетных данных.

6. Система аутентифицирует пользователя, сохраняет токен и перенаправляет его на основной интерфейс приложения (доску Kanban).

2.5.5 Сценарий прецедента «Выход из системы»

Основной успешный сценарий:

1. Аутентифицированный пользователь нажимает на иконку «Выйти» на панели настроек.

2. Система отображает модальное окно с запросом подтверждения выхода.

3. Пользователь нажимает кнопку «Выйти» в модальном окне.

4. Система завершает сеанс пользователя (удаляет токен).

5. Система перенаправляет пользователя на страницу аутентификации.

2.5.6 Сценарий прецедента «Просмотр профиля»

Основной успешный сценарий:

1. Аутентифицированный пользователь нажимает на иконку «Профиль пользователя» на панели настроек.

2. Система отображает страницу профиля пользователя с его данными (ФИО, email, дата регистрации, аватар).

2.5.7 Сценарий прецедента «Редактирование профиля»

Основной успешный сценарий:

1. Пользователь находится на странице своего профиля.
2. Пользователь нажимает кнопку «Редактировать профиль».
3. Система отображает модальное окно редактирования профиля с текущими данными.
4. Пользователь изменяет необходимые поля (имя, фамилия, отчество, URL аватара).
5. Пользователь нажимает кнопку «Сохранить изменения».
6. Система проверяет корректность введенных данных.
7. Система обновляет данные профиля пользователя в базе данных.
8. Система обновляет данные профиля в интерфейсе и закрывает модальное окно.

2.5.8 Сценарий прецедента «Удаление профиля»

Основной успешный сценарий:

1. Пользователь находится на странице своего профиля.
2. Пользователь нажимает кнопку «Удалить профиль».
3. Система отображает модальное окно с запросом подтверждения удаления профиля.
4. Пользователь нажимает кнопку «Да, удалить мой профиль».
5. Система удаляет профиль пользователя из базы данных.
6. Система завершает сеанс пользователя и перенаправляет на страницу аутентификации.

Управление Проектами

2.5.9 Сценарий прецедента «Создание нового проекта»

Основной успешный сценарий:

1. Аутентифицированный пользователь нажимает на иконку «Добавить новый проект» на панели настроек.
2. Система отображает модальное окно для добавления проекта.
3. Пользователь вводит название проекта в поле «Название проекта».
4. Пользователь опционально вводит описание проекта в поле «Описание».
5. Пользователь нажимает кнопку «Создать проект».
6. Система создает новый проект в базе данных, назначая текущего пользователя владельцем.
7. Система добавляет владельца в участники проекта.
8. Система обновляет список проектов в интерфейсе, новый проект становится текущим (если это первый проект или по логике приложения).
9. Модальное окно закрывается.

2.5.10 Сценарий прецедента «Редактирование проекта»

Условия: Пользователь является владельцем проекта или администратором. Основной успешный сценарий:

1. Пользователь (владелец/админ) нажимает на иконку «Редактировать текущий проект» на панели настроек (кнопка активна, если проект выбран).
2. Система отображает модальное окно редактирования текущего проекта с его данными.
3. Пользователь изменяет название и/или описание проекта.
4. Пользователь нажимает кнопку «Сохранить изменения».
5. Система обновляет данные проекта в базе данных.
6. Система обновляет название/описание проекта в интерфейсе. Модальное окно может закрыться или показать сообщение об успехе.

2.5.11 Сценарий прецедента «Удаление проекта»

Условия: Пользователь является владельцем проекта или администратором. Основной успешный сценарий:

1. Пользователь (владелец/админ) открывает модальное окно редактирования проекта.
2. Пользователь нажимает кнопку «Удалить проект».
3. Система отображает модальное окно подтверждения удаления проекта.
4. Пользователь нажимает кнопку «Да, удалить проект».
5. Система удаляет проект, все связанные с ним доски и задачи из базы данных.
6. Система обновляет список проектов в интерфейсе. Если удален текущий проект, выбирается другой или отображается состояние ”нет проектов”.
7. Модальные окна закрываются.

2.5.12 Сценарий прецедента «Управление участниками проекта»

Условия: Пользователь является владельцем проекта или администратором. Основной успешный сценарий (Просмотр):

1. Пользователь (владелец/админ) нажимает на иконку «Управление участниками проекта» на панели настроек (кнопка активна, если проект выбран).
2. Система отображает модальное окно со списком текущих участников проекта.

Основной успешный сценарий (Добавление участника):

1. В модальном окне управления участниками пользователь вводит email существующего пользователя системы в поле «Введите email пользователя для добавления».
2. Пользователь нажимает кнопку «Добавить».

3. Система проверяет, существует ли пользователь с таким email и не является ли он уже участником.

4. Система добавляет пользователя в участники проекта.

5. Список участников в модальном окне обновляется.

Основной успешный сценарий (Удаление участника):

1. В модальном окне управления участниками пользователь нажимает иконку удаления напротив участника (не владельца).

2. Система удаляет пользователя из участников проекта.

3. Список участников в модальном окне обновляется.

2.5.13 Сценарий прецедента «Создание доски»

Условия: Пользователь является владельцем текущего проекта или администратором. Проект выбран. Основной успешный сценарий:

1. Пользователь (владелец/админ) нажимает на иконку «Добавить новую доску» на панели настроек.

2. Система отображает модальное окно для добавления доски.

3. Пользователь вводит название доски в поле «Заголовок доски».

4. Пользователь опционально вводит описание доски.

5. Пользователь нажимает кнопку «Добавить доску».

6. Система создает новую доску в текущем проекте.

7. Система обновляет интерфейс, отображая новую доску в конце списка досок текущего проекта.

8. Модальное окно закрывается.

2.5.14 Сценарий прецедента «Редактирование доски»

Условия: Пользователь является владельцем проекта, к которому принадлежит доска, или администратором. Основной успешный сценарий:

1. Пользователь нажимает на иконку редактирования на заголовке доски.

2. Система отображает модальное окно редактирования доски с ее текущими данными.

3. Пользователь изменяет название и/или описание доски.

4. Пользователь нажимает кнопку «Сохранить».

5. Система обновляет данные доски в базе данных.

6. Система обновляет отображение доски в интерфейсе. Модальное окно закрывается.

2.5.15 Сценарий прецедента «Удаление доски»

Условия: Пользователь является владельцем проекта, к которому принадлежит доска, или администратором. Основной успешный сценарий:

1. Пользователь открывает модальное окно редактирования доски.

2. Пользователь нажимает кнопку «Удалить доску».

3. Система (опционально) запрашивает подтверждение.

4. Система удаляет доску и все связанные с ней задачи.

5. Система обновляет интерфейс, убирая доску. Модальное окно закрывается.

2.5.16 Сценарий прецедента «Изменение порядка досок»

Основной успешный сценарий:

1. Аутентифицированный пользователь наводит курсор на заголовок доски.

2. Пользователь зажимает левую кнопку мыши на заголовке доски и перетаскивает доску на новую позицию среди других досок текущего проекта.

3. Пользователь отпускает кнопку мыши.

4. Система визуально обновляет порядок досок на экране.

5. Система сохраняет новый порядок отображения досок для текущего проекта в базе данных.

2.5.17 Сценарий прецедента «Создание задачи»

Условия: Пользователь аутентифицирован, проект выбран, и пользователь является участником проекта (или владельцем/администратором). Основной успешный сценарий:

1. Пользователь нажимает на иконку «Добавить новую задачу» на панели настроек.
2. Система отображает модальное окно для добавления задачи.
3. Пользователь выбирает доску из выпадающего списка досок текущего проекта.
4. Пользователь вводит заголовок задачи.
5. Пользователь опционально вводит описание задачи.
6. Пользователь выбирает срок выполнения, приоритет и тип задачи.
7. Пользователь нажимает кнопку «Добавить задачу».
8. Система создает новую задачу и связывает ее с выбранной доской и текущим проектом. Текущий пользователь назначается автором.
9. Система обновляет интерфейс, отображая новую задачу на соответствующей доске.
10. Модальное окно закрывается.

2.5.18 Сценарий прецедента «Просмотр задачи»

Основной успешный сценарий:

1. Аутентифицированный пользователь нажимает на карточку задачи на одной из досок.
2. Система переключает вид основной области контента на страницу просмотра деталей задачи.
3. На странице отображается полная информация о задаче: заголовок, описание, проект, доска, срок выполнения, приоритет, тип, автор, разработчик (если назначен), QA (если назначен).

2.5.19 Сценарий прецедента «Редактирование задачи»

Условия: Пользователь является автором задачи или администратором. Основной успешный сценарий:

1. Пользователь находится на странице просмотра деталей задачи.
2. Пользователь нажимает кнопку «Редактировать задачу».
3. Система отображает модальное окно редактирования задачи с ее текущими данными (заголовок, описание, доска, приоритет; срок выполнения и тип НЕ редактируются в этой модали).
4. Пользователь изменяет необходимые поля.
5. Пользователь нажимает кнопку «Сохранить изменения».
6. Система обновляет данные задачи в базе данных.
7. Система обновляет отображение деталей задачи на странице и на карточке (если она видна). Модальное окно закрывается.

2.5.20 Сценарий прецедента «Удаление задачи»

Условия: Пользователь является автором задачи или администратором. Основной успешный сценарий:

1. Пользователь находится на странице просмотра деталей задачи.
2. Пользователь нажимает кнопку «Удалить задачу».
3. Система отображает модальное окно подтверждения удаления.
4. Пользователь нажимает кнопку «Да, удалить».
5. Система удаляет задачу из базы данных.
6. Система перенаправляет пользователя на доску Kanban и обновляет интерфейс.

2.5.21 Сценарий прецедента «Перемещение задачи»

Основной успешный сценарий:

1. Аутентифицированный пользователь наводит курсор на карточку задачи.

2. Пользователь зажимает левую кнопку мыши на карточке задачи и перетаскивает ее на другую доску в рамках текущего проекта.

3. Пользователь отпускает кнопку мыши над целевой доской.

4. Система визуально перемещает карточку задачи на новую доску и обновляет ее позицию в списке задач новой доски.

5. Система обновляет board_id и display_order задачи в базе данных.

2.6 Требования пользователя к интерфейсу web-интерфейса

Интерфейс должен быть интуитивно понятным, чтобы пользователи могли легко взаимодействовать с системой. На рисунке 2.2 представлен макет страницы.

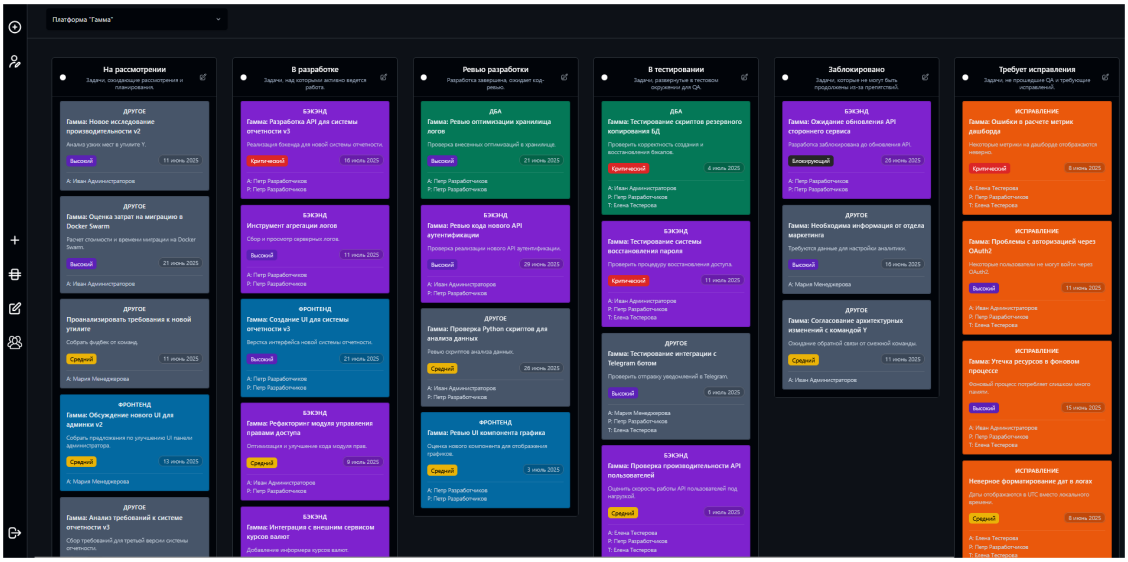

Рисунок 2.2 – Макет страницы

2.7 Нефункциональные требования

2.7.1 Требования к программному обеспечению

Для разработки и полноценной работы системы управления проектами необходимы следующие программные компоненты:

1. Среда выполнения JavaScript - Node.js.
2. Инструмент для сборки и разработки на основе Node.js - Vite.
3. Библиотека React для построения пользовательских интерфейсов.

4. Библиотека Zustand для управления состоянием приложения.

5. Библиотека react-beautiful-dnd для реализации функциональности перетаскивания.

6. CSS-фреймворк Tailwind CSS для стилизации интерфейса.

7. Библиотека для проверки типов свойств React-компонентов - prop-types.

2.8 Ограничения

Необходимо учитывать следующие ограничения:

1. Для доступа ко всем функциям системы, включая совместную работу и синхронизацию данных в реальном времени, необходимо постоянное и стабильное подключение к сети Интернет.

2. Система ориентирована на управление проектами с использованием Kanban-методологии и может не покрывать все потребности проектов, требующих других подходов или расширенного инструментария (например, диаграмм Ганта или сложного финансового учета).

3. Точность и эффективность специфических аналитических функций напрямую зависят от полноты, актуальности и качества данных, вводимых пользователями в систему.

4. Производительность системы при работе с очень большим количеством одновременных пользователей или значительными объемами данных (множество проектов, досок, задач) будет зависеть от характеристик и масштабируемости серверной инфраструктуры.

2.8.1 Требования к аппаратному обеспечению

Для работы приложения требуется дисковое пространство не менее 5 Гб, минимум 16 Гб оперативной памяти и подключение к Интернету. Рекомендуется использовать процессор с 6 или более ядрами и частотой 2 ГГц или выше.

2.9 Требования к оформлению документации

Разработка программной документации и программного изделия должна производиться согласно ГОСТ 19.102-77 и ГОСТ 34.601-90. Единая система программной документации.

3 Технический проект

3.1 Общая характеристика организации решения задачи

Задача заключается в разработке и программной реализации веб-приложения для управления проектами, основанного на использовании Kanban-доски и принципов Agile-методологий. Приложение предназначено для командной работы и поможет руководителям проектов, а также членам команд эффективно планировать спринты и текущую деятельность, визуализировать рабочие процессы, отслеживать выполнение задач и улучшать взаимодействие, что способствует повышению общей продуктивности и успешности проектов.

Приложение будет представлять собой многопользовательский веб-сервис, доступный через сеть Интернет. Основными элементами интерфейса будут являться интерактивная Kanban-доска для визуального управления задачами, страницы для настройки проектов и досок, а также инструменты для поддержки Agile-практик, такие как управление бэклогом и отслеживание прогресса по итерациям.

3.2 Обоснование выбора технологии проектирования

Для создания приложения выбраны технологии, которые обеспечивают высокую производительность и удобство для пользователей.

3.2.1 Язык программирования JavaScript

JavaScript — это высокоуровневый, мультипарадигменный язык программирования, являющийся ключевой технологией для создания интерактивных веб-сайтов и приложений. Он выполняется непосредственно в браузере пользователя, обеспечивая динамическое обновление контента и взаимодействие без перезагрузки страницы, а также может использоваться на серверной стороне благодаря среде Node.js [21].

3.2.2 Среда выполнения Node.js

Node.js — это кроссплатформенная среда выполнения JavaScript, построенная на движке V8 от Google Chrome. Она позволяет исполнять JavaScript-код на стороне сервера, что открывает возможности для создания быстрых и масштабируемых сетевых приложений [31].

3.2.3 Библиотека React

React — это популярная JavaScript-библиотека для создания пользовательских интерфейсов, разработанная и поддерживаемая Facebook (ныне Meta). В основе React лежит концепция компонентного подхода, позволяющая разбивать сложный интерфейс на независимые, переиспользуемые части — компоненты, каждый из которых управляет собственным состоянием [26].

3.2.4 PostgreSQL

PostgreSQL — это объектно-реляционная система управления базами данных, известная своей надежностью, гибкостью и расширяемостью. Она поддерживает широкий спектр функциональности, включая сложные запросы, транзакции, уровни изоляции и возможность создания пользовательских типов данных [39].

3.3 Диаграмма компонентов

На рисунке 3.1 изображена диаграмма компонентов системы.

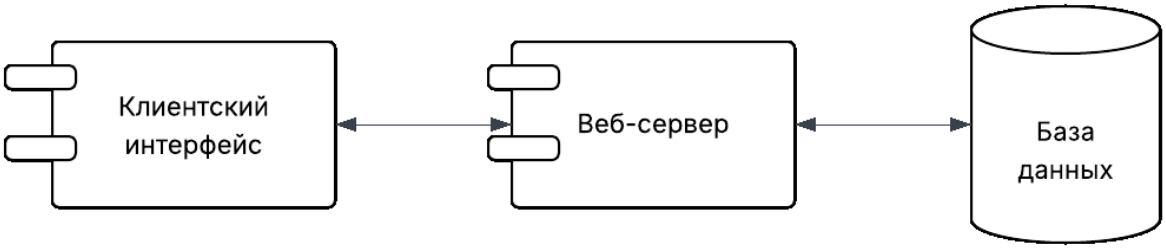

Рисунок 3.1 – Диаграмма компонентов

На представленной диаграмме показана архитектура веб-приложения для управления проектами, основанного на Kanban-доске и Agile-методиках. Схема включает три основных компонента: «Клиентский интерфейс», «Веб-сервер» и «База данных». Эти компоненты соединены стрелками, которые указывают направление потоков данных и взаимодействия между ними. Каждая часть системы выполняет свои специфические функции, обеспечивая совместную и эффективную работу всего приложения.

«Клиентский интерфейс» — это веб-приложение, которое запускается в браузере пользователя и служит основной точкой взаимодействия с системой. Через него пользователи могут регистрироваться и аутентифицироваться, создавать новые проекты, настраивать Kanban-доски, добавлять, редактировать и перемещать задачи между колонками. На страницах интерфейса в реальном времени отображается актуальное состояние проектов, Kanban-доски, списки задач, бэклоги и другая информация, необходимая для организации работы по Agile-принципам.

«Веб-сервер» функционирует как центральный узел архитектуры, обрабатывающий всю бизнес-логику системы. Он получает HTTP-запросы от «Клиентского интерфейса», выполняет соответствующие операции, взаимодействует с «Базой данных» для сохранения или извлечения необходимой информации, и отправляет ответы «Клиентскому интерфейсу» для обновления отображаемых данных или подтверждения выполненных действий.

«База данных» отвечает за надежное и долговременное хранение всей информации, используемой в системе управления проектами. В ней содержатся структурированные данные о зарегистрированных пользователях, созданных проектах, деталях задач, конфигурациях Kanban-досок и другие сведения, необходимые для функционирования Agile-процессов. «Веб-сервер» постоянно обращается к «Базе данных» для выполнения операций чтения и записи данных при каждом значимом взаимодействии пользователя с системой.

3.4 Структура программы

Система управления проектами представляет собой веб-приложение, чья структура организована в виде набора модулей, каждый из которых выполняет определённые функции.

3.4.1 Клиентская часть

Клиентская часть приложения разработана с использованием библиотеки React и отвечает за пользовательский интерфейс и взаимодействие с пользователем. Она включает в себя набор компонентов, каждый из которых выполняет определенную функцию, начиная от отображения основных элементов управления и заканчивая сложными модальными окнами для управления данными:

1. Модуль App.jsx. Главный компонент React-приложения. Управляет состоянием аутентификации пользователя, маршрутизацией между основными представлениями, отображением глобальных модальных окон. Инициализирует загрузку основных данных при входе пользователя и при их обновлении.

2. Модуль Layout.jsx. Компонент, определяющий основной макет интерфейса для аутентифицированных пользователей. Включает боковую панель настроек, верхнюю панель со списком проектов и поиском, а также основную область контента, где динамически отображаются либо доски Kanban с задачами, либо страница детального просмотра задачи. Реализует логику перетаскивания для досок и задач.

3. Модуль AuthPage.jsx. Компонент страницы аутентификации. Предоставляет пользователю интерфейс для регистрации новой учетной записи и входа в существующую. Включает валидацию вводимых данных и отображение требований к паролю.

4. Модуль UserProfilePage.jsx. Компонент страницы профиля пользователя. Отображает информацию о текущем аутентифицированном пользо-

вателе и предоставляет функционал для редактирования и удаления профиля через соответствующие модальные окна.

5. Модуль TaskViewPage.jsx. Компонент страницы детального просмотра задачи. Отображает всю информацию о выбранной задаче, включая ее заголовок, описание, проект, доску, срок выполнения, приоритет, тип, автора, а также назначенных разработчика и QA-специалиста. Предоставляет пользователю возможность назначить себя на роль разработчика или QA, если эти роли свободны. Также содержит кнопки для редактирования и удаления задачи.

6. Модуль Board.jsx. Компонент, представляющий одну доску Kanban в рамках проекта. Отображает заголовок и описание доски. Содержит список задач, принадлежащих этой доске, и обеспечивает возможность перетаскивания задач. Предоставляет кнопку для редактирования/удаления доски.

7. Модуль Task.jsx. Компонент, представляющий карточку отдельной задачи на доске. Отображает заголовок задачи, ее тип, приоритет, срок выполнения, а также краткую информацию об авторе, разработчике и QA-специалисте. Позволяет пользователю кликнуть по карточке для перехода на страницу детального просмотра задачи.

8. Модуль SettingsPanel.jsx. Компонент боковой панели навигации и действий. Содержит кнопки для добавления новой задачи, перехода к профилю пользователя, добавления нового проекта, добавления новой доски, редактирования текущего проекта, управления участниками текущего проекта и выхода из системы. Видимость некоторых кнопок зависит от роли пользователя и выбранного проекта.

9. Модуль ProjectListBar.jsx. Компонент, отображаемый в верхней части интерфейса. Содержит выпадающий список для выбора текущего проекта из числа тех, в которых пользователь состоит.

10. Модуль Tooltip.jsx. Вспомогательный компонент для отображения всплывающих текстовых подсказок при наведении курсора на элементы интерфейса.

11. Модуль Modal.jsx. Компонент модального окна для создания новой задачи. Позволяет выбрать доску, ввести заголовок, описание, срок выполнения, а также выбрать приоритет и тип задачи.

12. Модуль ModalAddBoard.jsx. Компонент модального окна для добавления новой доски в выбранный проект. Позволяет ввести заголовок и описание доски.

13. Модуль ModalAddProject.jsx. Компонент модального окна для создания нового проекта. Позволяет ввести название и описание проекта.

14. Модуль ModalAlert.jsx. Общий компонент для отображения простых информационных сообщений или оповещений об ошибках с одной кнопкой «OK».

15. Модуль ModalConfirm.jsx. Общий компонент для запроса подтверждения у пользователя перед выполнением деструктивных или важных действий. Предоставляет кнопки «Подтвердить» и «Отмена».

16. Модуль ModalEditBoard.jsx. Компонент модального окна для редактирования существующей доски. Позволяет изменить заголовок и описание доски, а также удалить доску.

17. Модуль ModalEditProfile.jsx. Компонент модального окна для редактирования персональной информации пользователя.

18. Модуль ModalEditProject.jsx. Компонент модального окна для редактирования существующего проекта. Позволяет изменить название и описание проекта, а также удалить проект.

19. Модуль ModalEditTask.jsx. Компонент модального окна для редактирования существующей задачи. Позволяет изменить заголовок, описание, доску и приоритет задачи.

20. Модуль ModalManageProjectMembers.jsx. Компонент модального окна для управления участниками выбранного проекта. Позволяет просматривать список участников, добавлять новых участников по email и удалять существующих.

3.4.2 Серверная часть

Серверная часть приложения, построенная на Node.js с использованием фреймворка Express.js, обеспечивает обработку всех API-запросов, управление бизнес-логикой, взаимодействие с базой данных и реализацию механизмов безопасности, таких как аутентификация и авторизация. Списко связанных модулей:

1. Модуль server.js. Основной файл серверной части приложения, реализованный на Node.js с использованием фреймворка Express.js. Отвечает за обработку всех API-запросов от клиентской части. Реализует эндпоинты для:
   - регистрации и аутентификации пользователей;
   - управления профилями пользователей;
   - получения списка всех пользователей системы;
   - управления проектами;
   - управления участниками проектов;
   - управления досками;
   - управления задачами.

Взаимодействует с базой данных PostgreSQL для хранения и извлечения данных. Реализует логику авторизации, проверяя права пользователя (например, владелец проекта, администратор, участник проекта) перед выполнением защищенных операций.

3.4.3 Клиентские сервисы

Клиентские сервисы представляют собой вспомогательные JavaScript-модули, которые инкапсулируют специфическую логику, используемую различными компонентами клиентской части. Они помогают упростить взаимодействие с API и управление состоянием аутентификации. Список связанных модулей:

1. Модуль apiService.js. Модуль на стороне клиента, предоставляющий функцию authenticatedFetch для упрощения выполнения аутентифицированных HTTP-запросов к API серверной части. Автоматически добавляет JWT

токен аутентификации в заголовки запросов. Содержит логику для обработки специфических ошибок от сервера и вызова глобального обработчика для принудительного выхода пользователя из системы при проблемах с сессией или токеном.

2. Модуль tokenService.js. Модуль на стороне клиента, предоставляющий утилиты для работы с JWT токеном. Включает функцию isTokenActive для проверки наличия токена в localStorage и валидации его срока действия путем декодирования с использованием jwt-decode.

3.4.4 Управление состоянием и стилизация

Для эффективного управления состоянием приложения на стороне клиента используется библиотека Zustand, обеспечивающая централизованное и реактивное хранилище данных. Визуальное оформление реализуется с помощью Tailwind CSS и кастомных CSS-правил для создания консистентного и адаптивного пользовательского интерфейса. Список связанных модулей:

1. Модуль store.js. Файл, определяющий глобальное хранилище состояния приложения с использованием библиотеки Zustand. Содержит состояние для проектов, досок, задач, информации о текущем пользователе, списке всех пользователей, текущем выбранном проекте и задаче, виде отображения приложения, а также различные флаги для модальных окон и оповещений. Реализует действия для изменения этого состояния. Использует middleware persist для сохранения части состояния в localStorage браузера.

2. Модуль index.css. Основной файл стилей приложения. Включает базовые стили и утилиты Tailwind CSS, а также кастомные CSS-правила для оформления полос прокрутки и других элементов интерфейса для обеспечения единого визуального стиля.

3.4.5 Точка входа и конфигурация БД

Этот раздел описывает основные файлы, отвечающие за запуск клиентского приложения, а также скрипт для инициализации и настройки структу-

ры базы данных PostgreSQL, включая создание таблиц и заполнение начальными данными. Список связанных элементов:

1. Модуль main.jsx. Точка входа клиентского React-приложения. Инициализирует корневой компонент App и рендерит его в DOM-элемент с id root. Подключает основные стили из index.css.

2. db_schema.sql. SQL-скрипт, предназначенный для инициализации структуры базы данных PostgreSQL. Содержит команды создания для всех необходимых сущностей, определяет первичные и внешние ключи, ограничения, значения по умолчанию и триггеры для автоматического обновления временных меток. Также включает секцию для заполнения базы данных начальными тестовыми данными для демонстрации и тестирования функционала.

3.5 Структура базы данных

Сущности и отношения между ними отображены на ER-диаграмме.

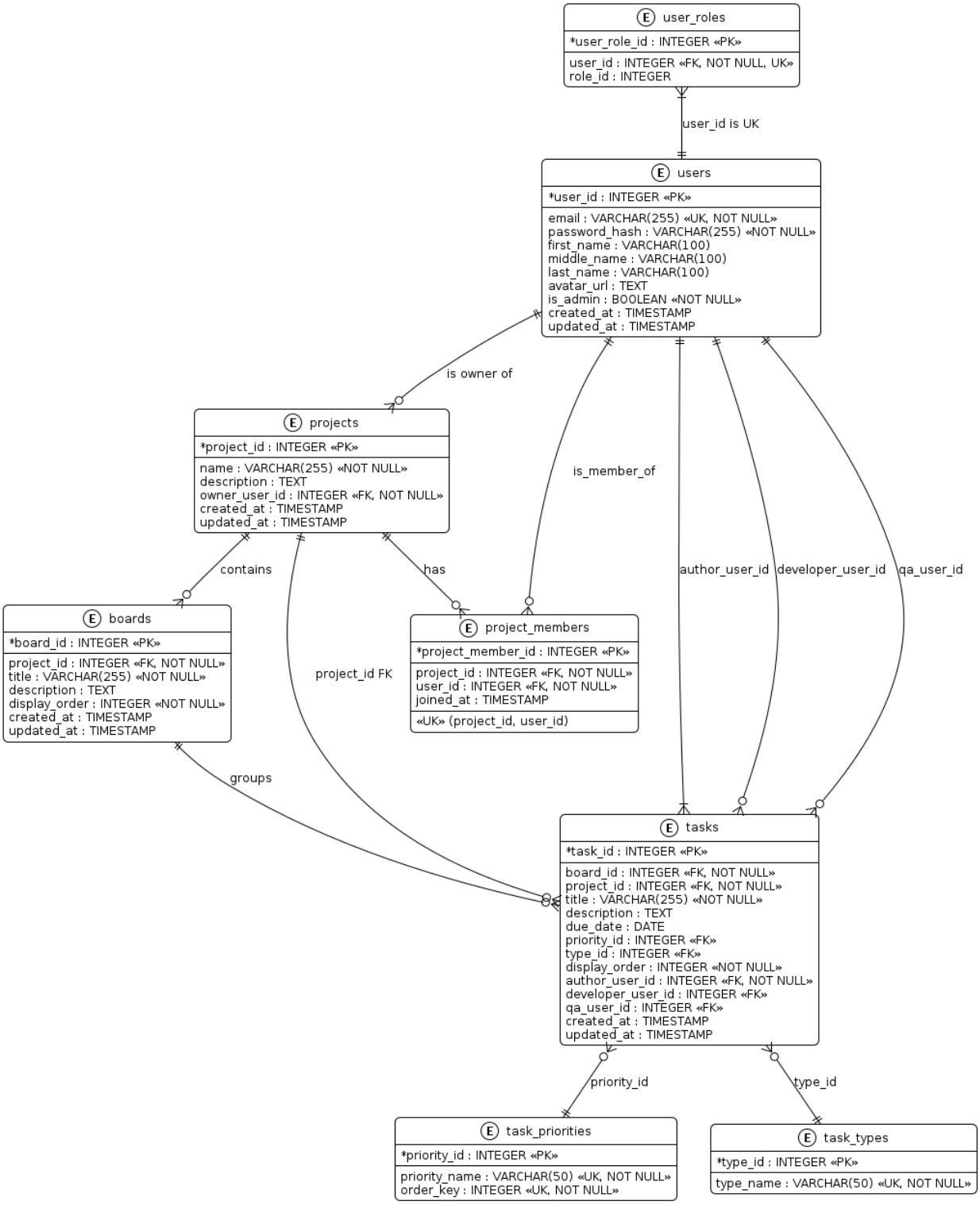

Рисунок 3.2 – ER-диаграмма

Таблица 3.1 – Атрибуты сущности «Users»

Поле	Тип	Обязательное	Описание
1	2	3	4
user_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор пользователя.
email	VARCHAR(255), UNIQUE, NOT NULL	да	Адрес электронной почты пользователя, уникальный.
password_hash	VARCHAR(255), NOT NULL	да	Хеш пароля пользователя.
first_name	VARCHAR(100)	нет	Имя пользователя.
middle_name	VARCHAR(100)	нет	Отчество пользователя.
last_name	VARCHAR(100)	нет	Фамилия пользователя.
avatar_url	TEXT	нет	URL-адрес аватара пользователя.
is_admin	BOOLEAN, DEFAULT FALSE, NOT NULL	да	Флаг, указывающий, является ли пользователь администратором.
created_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_TIMESTAMP	да	Дата и время создания записи.
updated_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_TIMESTAMP	да	Дата и время последнего обновления записи.

Таблица 3.2 – Атрибуты сущности «Projects»

Поле	Тип	Обязательное	Описание
1	2	3	4
project_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор проекта.
name	VARCHAR(255), NOT NULL	да	Название проекта.
description	TEXT	нет	Описание проекта.
owner_ user_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на users(user_id). Идентификатор пользователя-владельца проекта.
created_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время создания записи.
updated_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время последнего обновления записи.

Таблица 3.3 – Атрибуты сущности «Project Members»

Поле	Тип	Обязательное	Описание
1	2	3	4
project_ member_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор записи об участии.
project_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на projects(project_id). Идентификатор проекта.
user_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на users(user_id). Идентификатор пользователя-участника.
joined_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время присоединения пользователя к проекту.

Таблица 3.4 – Атрибуты сущности «Boards»

Поле	Тип	Обязательное	Описание
1	2	3	4
board_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор доски.
project_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на projects(project_id). Идентификатор проекта, к которому принадлежит доска.

Продолжение таблицы 3.4

1	2	3	4
title	VARCHAR(255), NOT NULL	да	Название доски.
description	TEXT	нет	Описание доски.
display_order	INTEGER, NOT NULL, DEFAULT 0	да	Порядок отображения доски в проекте.
created_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время создания записи.
updated_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время последнего обновления записи.

Таблица 3.5 – Атрибуты сущности «Task Priorities»

Поле	Тип	Обязательное	Описание
1	2	3	4
priority_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор приоритета.
priority_ name	VARCHAR(50), UNIQUE, NOT NULL	да	Название приоритета (например, ’Low’, ’Medium’).

Продолжение таблицы 3.5

1	2	3	4
order_key	INTEGER, UNIQUE, NOT NULL	да	Ключ для сортировки приоритетов.

Таблица 3.6 – Атрибуты сущности «Task Types»

Поле	Тип	Обязательное	Описание
1	2	3	4
type_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор типа задачи.
type_name	VARCHAR(50), UNIQUE, NOT NULL	да	Название типа задачи (например, ’FRONTEND’, ’BUGFIX’).

Таблица 3.7 – Атрибуты сущности «Tasks»

Поле	Тип	Обязательное	Описание
1	2	3	4
task_id	SERIAL, PK	да	Первичный ключ, уникальный идентификатор задачи.
board_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на boards(board_id). Идентификатор доски, на которой находится задача.

Продолжение таблицы 3.7

1	2	3	4
project_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на projects(project_id). Идентификатор проекта, к которому принадлежит задача.
title	VARCHAR(255), NOT NULL	да	Заголовок задачи.
description	TEXT	нет	Описание задачи.
due_date	DATE	нет	Срок выполнения задачи.
priority_id	INTEGER, FK	нет	Внешний ключ, ссылается на task_priorities(priority_id). Идентификатор приоритета задачи.
type_id	INTEGER, FK	нет	Внешний ключ, ссылается на task_types(type_id). Идентификатор типа задачи.
display_ order	INTEGER, NOT NULL, DEFAULT 0	да	Порядок отображения задачи на доске.
author_ user_id	INTEGER, FK, NOT NULL	да	Внешний ключ, ссылается на users(user_id). Идентификатор пользователя-автора задачи.
developer_ user_id	INTEGER, FK	нет	Внешний ключ, ссылается на users(user_id). Идентификатор пользователя-разработчика.
qa_user_id	INTEGER, FK	нет	Внешний ключ, ссылается на users(user_id). Идентификатор пользователя-тестировщика.

Продолжение таблицы 3.7

1	2	3	4
created_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время создания записи.
updated_at	TIMESTAMP WITH TIME ZONE, DEFAULT CURRENT_ TIMESTAMP	да	Дата и время последнего обновления записи.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Орлов, С. А. Основы управления ИТ-проектами / С. А. Орлов – Москва : ДМК Пресс, 2023. - 320 с. – ISBN 978-5-9706-0881-4. – Текст : непосредственный.

2. Семенов, М. И. Управление программными проектами: стандарты, инструменты, практика / М. И. Семенов, И. В. Волков – Санкт-Петербург : БХВ-Петербург, 2024. - 450 с. – ISBN 978-5-9775-4321-7. – Текст : непосредственный.

3. Ковалев, А. П. Риск-менеджмент в проектах разработки программного обеспечения / А. П. Ковалев – Москва : Инфра-М, 2022. - 280 с. – ISBN 978-5-16-017854-9. – Текст : непосредственный.

4. Липаев, В. В. Качество программного обеспечения: принципы и методы оценки / В. В. Липаев – Москва : Янус-К, 2023. - 512 с. – ISBN 978-5-8037-0912-5. – Текст : непосредственный.

5. Петров, К. Д. Эффективные коммуникации в ИТ-командах / К. Д. Петров – Москва : Альпина Паблишер, 2024. - 198 с. – ISBN 978-5-9614-8234-1. – Текст : непосредственный.

6. Вигерс, К. Разработка требований к программному обеспечению. 3-е издание / К. Вигерс, Дж. Битти – Москва : БИНОМ. Лаборатория знаний, 2023. - 736 с. – ISBN 978-5-9909700-3-8. – Текст : непосредственный.

7. Группа Акселос. ITIL® 4: Основы / Группа Акселос – Москва : VAN HAREN PUBLISHING, 2024. - 320 с. – ISBN 978-9-40180-888-0. – Текст : непосредственный.

8. Лебедев, А. Н. Лидерство в технологических командах: создание культуры инноваций и сотрудничества / А. Н. Лебедев – Санкт-Петербург : Питер, 2023. - 288 с. – ISBN 978-5-4461-2345-6. – Текст : непосредственный.

9. Ладушкин, В. С. Управление данными и Data Governance: стратегии и внедрение / В. С. Ладушкин – Москва : ДМК Пресс, 2024. - 410 с. – ISBN 978-5-9706-1234-0. – Текст : непосредственный.

10. Ершов, М. П. Основы облачных вычислений: технологии, модели и сервисы / М. П. Ершов – Москва : Техносфера, 2023. - 350 с. – ISBN 978-5-94836-987-0. – Текст : непосредственный.

11. Кон, М. Agile: оценка и планирование проектов / М. Кон – Москва : Вильямс, 2022. - 450 с. – ISBN 978-5-6045556-8-7. – Текст : непосредственный.

12. Вольф, К. С. Пользовательские истории: искусство гибкой разработки ПО / К. С. Вольф, Л. Гофман – Москва : ДМК Пресс, 2023. - 304 с. – ISBN 978-5-9706-0901-9. – Текст : непосредственный.

13. Дерби, Э. Agile-ретроспектива: как превратить хорошую команду в великую / Э. Дерби, Д. Ларсен – Москва : Эксмо, 2024. - 224 с. – ISBN 978-5-04-178834-3. – Текст : непосредственный.

14. Пуппол, М. Принципы Agile. Гибкое управление проектами для начинающих / М. Пуппол – Москва : АСТ, 2023. - 128 с. – ISBN 978-5-17-154321-0. – Текст : непосредственный.

15. Сидоренко, В. П. Коучинг Agile-команд: практическое руководство / В. П. Сидоренко – Москва : Доброе слово, 2024. - 280 с. – ISBN 978-5-9909876-5-4. – Текст : непосредственный.

16. Андерсон, Д. Канбан: альтернативный путь в Agile / Д. Андерсон – Москва : Символ-Плюс, 2023. - 352 с. – ISBN 978-5-93286-234-5. – Текст : непосредственный.

17. Бергманн, Б. Kanban изнутри: два года успешного применения / Б. Бергманн, П. Ахмар – Санкт-Петербург : Питер, 2024. - 240 с. – ISBN 978-5-496-03456-7. – Текст : непосредственный.

18. Леопольд, К. Практический Kanban: от идеи до поставки / К. Леопольд, З. Маурус – Москва : ДМК Пресс, 2022. - 288 с. – ISBN 978-5-9706-0777-0. – Текст : непосредственный.

19. Зайцев, М. В. Kanban для команд разработки ПО: внедрение и оптимизация / М. В. Зайцев – Москва : Техносфера, 2023. - 192 с. – ISBN 978-5-94836-789-0. – Текст : непосредственный.

20. Бенсон, Дж. Персональный Kanban: карта вашей работы / Дж. Бенсон, Т. де Мариа Бенсон – Москва : Эксмо, 2024. - 208 с. – ISBN 978-5-04-165432-1. – Текст : непосредственный.

21. Флэнаган, Д. JavaScript: подробное руководство. 7-е издание / Д. Флэнаган – Санкт-Петербург : Символ-Плюс, 2022. - 720 с. – ISBN 978-5-93286-228-4. – Текст : непосредственный.

22. Хавербеке, М. Выразительный JavaScript. 3-е издание / М. Хавербеке – Санкт-Петербург : Питер, 2023. - 480 с. – ISBN 978-5-4461-1700-0. – Текст : непосредственный.

23. Стефанов, С. JavaScript. Шаблоны / С. Стефанов – Москва : Вильямс, 2024. - 272 с. – ISBN 978-5-8459-2299-9. – Текст : непосредственный.

24. Сойер, К. Современный JavaScript для нетерпеливых / К. Сойер – Санкт-Петербург : БХВ-Петербург, 2024. - 512 с. – ISBN 978-5-9775-1234-5. – Текст : непосредственный.

25. Макфарланд, Д. Новая большая книга CSS / Д. Макфарланд – Москва : Эксмо, 2023. - 640 с. – ISBN 978-5-04-112345-6. – Текст : непосредственный.

26. Бэнкс, А. React и Redux: функциональная веб-разработка / А. Бэнкс, Е. Порселло – Санкт-Петербург : Питер, 2023. - 336 с. – ISBN 978-5-4461-1987-5. – Текст : непосредственный.

27. Фримен, А. React для профессионалов / А. Фримен – Москва : ДМК Пресс, 2024. - 800 с. – ISBN 978-5-9706-0999-6. – Текст : непосредственный.

28. Васильев, Д. А. React. Быстрый старт с хуками и TypeScript / Д. А. Васильев – Москва : Солон-Пресс, 2023. - 352 с. – ISBN 978-5-91359-555-1. – Текст : непосредственный.

29. Шварцмюллер, М. React – полное руководство (включая Hooks, React Router, Redux) / М. Шварцмюллер – Москва : Эксмо, 2025. - 688 с. – ISBN 978-5-04-188765-4. – Текст : непосредственный.

30. Громов, И. П. Сборка современных веб-приложений с Vite: React, Vue, Svelte / И. П. Громов – Санкт-Петербург : Наука и Техника, 2024. - 280 с. – ISBN 978-5-94387-888-9. – Текст : непосредственный.

31. Кантелон, М. Node.js. Разработка серверных приложений / М. Кантелон, Т. Хартер, Н. Раджмохан – Санкт-Петербург : Питер, 2023. - 576 с. – ISBN 978-5-4461-1654-0. – Текст : непосредственный.

32. Чен, Ф. Разработка веб-приложений с помощью Node.js и Express / Ф. Чен – Москва : Вильямс, 2024. - 432 с. – ISBN 978-5-907114-77-7. – Текст : непосредственный.

33. Тейшейра, П. Профессиональный Node.js: создание масштабируемых приложений / П. Тейшейра – Москва : ДМК Пресс, 2022. - 624 с. – ISBN 978-5-9706-0811-1. – Текст : непосредственный.

34. Белов, А. В. Node.js: шаблоны проектирования и лучшие практики / А. В. Белов – Москва : Бином. Лаборатория знаний, 2023. - 368 с. – ISBN 978-5-9963-6789-0. – Текст : непосредственный.

35. Голдберг, А. REST API на Node.js и MongoDB / А. Голдберг – Санкт-Петербург : БХВ-Петербург, 2024. - 312 с. – ISBN 978-5-9775-3210-5. – Текст : непосредственный.

36. Карвин, Б. SQL. Антипаттерны: как избежать ошибок при проектировании баз данных / Б. Карвин – Санкт-Петербург : Питер, 2023. - 320 с. – ISBN 978-5-4461-1543-3. – Текст : непосредственный.

37. Петров, А. И. SQL для анализа данных: практическое руководство / А. И. Петров – Москва : ДМК Пресс, 2024. - 384 с. – ISBN 978-5-9706-1001-5. – Текст : непосредственный.

38. Ульман, Дж. Д. Введение в системы баз данных / Дж. Д. Ульман, Дж. Уидом – Москва : Вильямс, 2021. - 1072 с. – ISBN 978-5-8459-2153-3. – Текст : непосредственный.

39. Дуглас, К. PostgreSQL для начинающих. От основ к профессиональной разработке / К. Дуглас – Москва : Лори, 2024. - 480 с. – ISBN 978-5-85582-456-7. – Текст : непосредственный.

40. Васкес, Л. PostgreSQL. Администрирование баз данных для профессионалов / Л. Васкес – Москва : Символ-Плюс, 2022. - 704 с. – ISBN 978-5-93286-301-4. – Текст : непосредственный.

41. Мартин, Р. Чистая архитектура: Искусство разработки программного обеспечения / Р. Мартин – Санкт-Петербург : Питер, 2023. - 352 с. – ISBN 978-5-4461-0772-8. – Текст : непосредственный.

42. Ким, Дж. Проект «Феникс». Роман о том, как DevOps меняет бизнес к лучшему / Дж. Ким, К. Бер, Дж. Спэффорд – Москва : Манн, Иванов и Фербер, 2022. - 480 с. – ISBN 978-5-00169-567-4. – Текст : непосредственный.

43. Манифест гибкой разработки программного обеспечения : сайт / AgileManifesto.org. – [Б. м.] : AgileManifesto.org, 2001 – . – URL: https://agilemanifesto.org/iso/ru/manifesto.html (дата обращения: 17.04.2025). – Текст: электронный.

44. Kanban University : сайт / Kanban University. – [Б. м.] : Kanban University, 2010 – . – URL: https://kanban.university/ (дата обращения: 19.04.2025). – Текст: электронный.

45. React – официальная документация : сайт / React team. – [Б. м.] : Meta Platforms, Inc., 2013 – . – URL: https://react.dev/ (дата обращения: 19.04.2025). – Текст: электронный.

46. Vite – официальная документация : сайт / Vite team. – [Б. м.] : Vite team, 2020 – . – URL: https://vitejs.dev/guide/ (дата обращения: 19.04.2025). – Текст: электронный.

47. Node.js – официальная документация : сайт / OpenJS Foundation. – [Б. м.] : OpenJS Foundation, 2009 – . – URL: https://nodejs.org/ru/docs/ (дата обращения: 19.04.2025). – Текст: электронный.

48. PostgreSQL – официальная документация : сайт / The PostgreSQL Global Development Group. – [Б. м.] : The PostgreSQL Global Development Group, 1996 – . – URL: https://www.postgresql.org/docs/ (дата обращения: 23.04.2025). – Текст: электронный.

49. JavaScript | MDN : сайт / Mozilla Developer Network. – [Б. м.] : Mozilla, 2005 – . – URL: https://developer.mozilla.org/ru/docs/Web/JavaScript (дата обращения: 18.04.2025). – Текст: электронный.

50. Канбан-метод в разработке: от теории к практике : сайт / Habr. – Москва : Хабр, 2023 – . – URL: https://habr.com/ru/companies/otus/articles/780000/ (дата обращения: 17.04.2025). – Текст: электронный.

51. Эффективное управление состоянием в React приложениях : сайт / Smashing Magazine. – Freiburg : Smashing Media AG, 2024 – . – URL: https://www.smashingmagazine.com/tag/react/ (дата обращения: 19.04.2025). – Текст: электронный.