Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук
Кафедра информационных и управляющих систем
Направление подготовки 09.03.02 – Информационные системы и технологии
Направленность (профиль) образовательной программы Информационной системы и технологии

ДОПУСТИТЬ К ЗАЩИТЕ

Зав. кафедрой

_____ А.В. Бушманов
«_____» _____ 2025 г.

БАКАЛАВРСКАЯ РАБОТА

на тему: Разработка информационной системы для компании-разработчика бухгалтерских программ

Исполнитель студент группы 1104-об	_____ (подпись, дата)	М.А. Маканников
Руководитель канд. физ-мат. наук	_____ (подпись, дата)	А.В. Павельчук
Консультант: по безопасности и экологичности доцент, канд. техн. наук	_____ (подпись, дата)	А.Б. Булгаков
Нормоконтроль инженер кафедры	_____ (подпись, дата)	В.Н. Адаменко

Благовещенск 2025

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук
Кафедра информационных и управляющих систем
Направление подготовки 09.03.02 Информационные системы и технологии
Направленность (профиль) образовательной программы Информационные системы и технологии

УТВЕРЖДАЮ
Зав. кафедрой
_____ А. В. Бушманов
«_____» _____ 2025 г.

З А Д А Н И Е

К выпускной квалификационной работе студента Маканникова М.А.

1. Тема выпускной квалификационной работы: Разработка информационной системы для компании-разработчика бухгалтерских программ (утверждено приказом от 14.04.2025 № 980-уч)
2. Срок сдачи студентом законченной работы (проекта): 10.06.2025
3. Содержание выпускной квалификационной работы: анализ предметной области; освоение программного и технического обеспечения; разработка алгоритма решения; тестирование.
4. Перечень материалов приложения: перечень вопросов из опроса, демонстрация работы программной визуализация, техническое задание.
5. Дата выдачи задания 02.10.2024

Руководитель выпускной квалификационной работы: Павельчук Анна Владимировна, доцент, канд. техн. наук

Задание принял к исполнению: 02.10.2024 г. _____

РЕФЕРАТ

Бакалаврская работа содержит 95 страниц, 26 рисунков, 17 таблиц, 25 источников, 1 приложение

HTML, CSS, JAVASCRIPT, JQUERY, BOOTSTRAP, PHP, MODX, REDBEANPHP, PHPMYADMIN, ИНФОРМАЦИОННАЯ СИСТЕМА, АВТОМАТИЗАЦИЯ, БАЗА ДАННЫХ, ПРОЕКТИРОВАНИЕ, ТЕСТИРОВАНИЕ, БЕЗОПАСНОСТЬ, УСПРАВЛЕНИЕ ЗАДАЧАМИ.

Целью выпускной квалификационной работы стало создание информационной системы для поддержки управления проектами в компании, разрабатывающей бухгалтерское ПО. Основная задача – повышение эффективности за счёт централизации данных, улучшения взаимодействия сотрудников и автоматизации рутинных операций.

Реализация включала выбор клиент-серверной архитектуры, построение ER-диаграмм, описание сущностей и прототипов интерфейса. Для фронтенда использовались HTML5, CSS3, JavaScript и Bootstrap, для серверной части – PHP, CMS ModX, ORM RedBeanPHP и СУБД MySQL через PHPMyAdmin.

При разработке применялись MVC, модульный подход и Git для контроля версий. Обеспечена защита от SQL-инъекций и реализована система ролей (ответственный, наблюдатель, создатель) для ограничения доступа.

На завершающем этапе проведено модульное, функциональное и нагрузочное тестирование, подтвердившее надёжность и соответствие требованиям.

НОРМАТИВНЫЕ ССЫЛКИ

Приказ Министерства труда и социальной защиты Российской Федерации от 29 октября 2021 г. N 774н «Об утверждении общих требований к организации безопасного рабочего места»

СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания»

СанПиН 2.1.3684-21 «Санитарно-эпидемиологические требования к содержанию территорий городских и сельских поселений, к водным объектам, питьевой воде и питьевому водоснабжению населения, атмосферному воздуху, почвам, жилым помещениям, эксплуатации производственных, общественных помещений, организации и проведению санитарно-противоэпидемических (профилактических) мероприятий»

СТО СМК 4.2.3.21-2018 Оформление выпускных квалификационных и курсовых работ (проектов)

ГОСТ Р 50948-2001 Средства отображения информации индивидуального пользования. Общие эргономические требования и требования безопасности

ГОСТ Р 52872-2019 Интернет-ресурсы и другая информация, представленная в электронно-цифровой форме. Приложение для стационарных и мобильных устройств, иные пользовательские интерфейсы

ГОСТ Р 58751-2019 Слаботочные системы. Кабельные системы. Телекоммуникационные пространства и помещения. Рабочее место

ГОСТ Р ИСО 9241-161-2016 Эргономика взаимодействия человек-система. Часть 161. Элементы графического пользовательского интерфейса

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

HTML – язык гипертекстовой разметки, используемый для структурирования и отображения содержимого веб-страниц.

CSS – каскадные таблицы стилей, применяемые для оформления внешнего вида веб-страниц и элементов интерфейса.

JavaScript – скриптовый язык, добавляющий динамику и интерактивность на клиентской стороне сайта.

Bootstrap – популярный фронтенд-фреймворк, позволяющий быстро создавать адаптивные и кроссбраузерные интерфейсы.

PHP – серверный язык программирования, использованный для обработки данных, реализации бизнес-логики и взаимодействия с базой данных.

ModX – гибкая система управления контентом (CMS), обеспечившая удобную административную панель и управляемость ресурсами.

RedBeanPHP – простая и удобная ORM-библиотека, позволяющая работать с базой данных в объектно-ориентированном стиле.

MySQL через PHPMyAdmin – система управления реляционными базами данных с графическим интерфейсом для упрощения администрирования.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные тенденции развития цифровой экономики требуют от организаций адаптации и модернизации своих процессов, чтобы оставаться конкурентоспособными. Особенно важным становится вопрос оптимизации внутренних операций в тех сферах, где управленческий персонал сталкивается с высокой нагрузкой, а также где критически важны точность и скорость принятия решений. В таких условиях автоматизация перестаёт быть просто инструментом повышения производительности и становится ключевым элементом стратегического развития компании.

Исследование сосредоточено на проблеме управления проектами в деятельности компании, специализирующейся на разработке бухгалтерского программного обеспечения. Основным объектом анализа стал процесс координации задач между отделами, начиная от постановки целей и заканчивая финальным контролем реализации. Предметом исследования выступила разрабатываемая информационная система, предназначенная для автоматизации этих процессов, снижения административной нагрузки и предоставления аналитической базы для принятия решений.

Актуальность данной работы обусловлена рядом важных факторов. Во-первых, увеличение масштабов проектов ведет к росту объема данных, которые необходимо учитывать при управлении, что делает традиционные подходы и инструменты недостаточно эффективными. Во-вторых, существующие системы управления задачами, хотя и являются универсальными, часто не отвечают специфическим требованиям IT-компаний, особенно тех, которые функционируют в регулируемых сферах, таких как разработка программного обеспечения для бухгалтерии и финансов

Целью данного исследования стало создание информационной системы, ориентированной на специфические потребности компании, занимающейся разработкой бухгалтерских решений. Такая система направлена на повышение

скорости выполнения проектных задач, улучшение внутренней коммуникации и обеспечение более точного прогнозирования сроков реализации проектов. Для достижения поставленной цели были определены и последовательно решены следующие задачи:

– провести детальный анализ текущих управленческих процессов и выявить «узкие места»;

– изучить существующие подходы к автоматизации проектного управления и их соответствие специфике целевой организации;

– разработать архитектурную модель системы, включающую интерфейс, структуру БД и модули взаимодействия;

– реализовать функциональный прототип и протестировать его в условиях, приближенных к реальным;

– дать оценку практической ценности внедрения и предложить рекомендации по дальнейшему развитию системы.

Практическая польза заключается в том, что разработанная система может быть внедрена в штатные процессы компании, что позволит улучшить контроль над проектами, повысить прозрачность задач и снизить риск возникновения ошибок на этапах распределения и исполнения.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ОРГАНИЗАЦИОННАЯ СТРУКТУРА ПРЕДПРИЯТИЯ

1.1 Обоснование актуальности

Информационная система — взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели. Современное понимание информационной системы предполагает использование в качестве основного технического средства переработки информации персонального компьютера. В крупных организациях наряду с персональным компьютером в состав технической базы информационной системы может входить мэйнфрейм или суперЭВМ. Кроме того, техническое воплощение информационной системы само по себе ничего не будет значить, если не учтена роль человека, для которого предназначена производимая информация и без которого невозможно ее получение и представление. Под организацией будем понимать сообщество людей, объединенных общими целями и использующих общие материальные и финансовые средства для производства материальных и информационных продуктов и услуг [1].

Современные реалии развития экономики требуют от компаний постоянного совершенствования своих процессов. Особенно это касается организаций, функционирующих в условиях высокой ответственности и строгих нормативных требований, где любые неточности могут привести к серьёзным последствиям. Именно поэтому автоматизация всё чаще рассматривается не как вспомогательный инструмент, а как стратегический ресурс, способствующий устойчивому развитию предприятия. Это особенно важно для компаний, занимающихся разработкой программного обеспечения специального назначения — например, в области финансового и бухгалтерского учета, где даже незначительные ошибки могут повлечь за собой юридические и экономические риски.

Особую значимость внедрение автоматизированных решений

приобретает в сфере управления проектами в IT-компаниях. Современный руководитель здесь сталкивается с необходимостью одновременного контроля нескольких направлений деятельности, организации работы межфункциональных команд, оперативного реагирования на изменения условий и соблюдения сроков выполнения задач. При этом традиционные средства планирования — такие как Excel или популярные онлайн-платформы (Trello, Asana, Jira) — часто оказываются недостаточно гибкими, безопасными и интегрируемыми с внутренними корпоративными системами. Как результат, возникает фрагментация данных, снижается прозрачность процессов, увеличивается время на рутинные действия.

Создание специализированной информационной системы может стать ключевым шагом в оптимизации управленческих процессов. Единая цифровая платформа позволяет объединить все задачи в одном пространстве, распределять их между исполнителями, отслеживать прогресс и анализировать эффективность. Такой подход значительно снижает нагрузку на менеджеров, делает коммуникацию более понятной и устойчивой, а также повышает общую производительность. Более того, автоматизация таких рутинных функций, как выявление задержек или оценка загрузки персонала, даёт возможность сосредоточиться на стратегических задачах.

Ещё одной причиной актуальности подобных решений является необходимость интеграции новой системы с уже существующими корпоративными платформами. Современная IT-компания обычно использует множество инструментов: ERP и CRM-системы, системы документооборота, внутренние базы знаний и другие. Информационная система управления проектами должна обеспечивать обмен данными между этими платформами, создавая целостное информационное поле. Это позволяет избежать дублирования информации, минимизировать ошибки и повысить прозрачность процессов.

Не менее важным фактором остаётся обеспечение информационной

безопасности. Учитывая, что компания работает с чувствительными финансовыми данными, защита информации становится критически важной. Современные требования законодательства и рост киберугроз требуют реализации надёжных механизмов шифрования, ограничения прав доступа и детальной регистрации действий пользователей. Эти меры должны быть заложены ещё на этапе проектирования системы, а не добавлены постфактум.

Также нельзя игнорировать аналитические возможности такой системы. Одно из главных преимуществ автоматизации — сбор и анализ данных о выполнении задач, частых проблемах и типичных задержках. На основе этих данных можно формировать отчёты, выявлять «узкие места», оптимизировать процессы и принимать более обоснованные управленческие решения. Например, если на этапе тестирования регулярно возникают проблемы, это может свидетельствовать о недостаточной подготовке кода или нехватке ресурсов. Таким образом, система превращается в мощный инструмент анализа и стратегического управления.

Кроме того, внедрение автоматизированной системы положительно влияет на культуру организации. Чётко структурированная среда, где каждая задача чётко определена, сроки известны, а ответственность распределена, способствует повышению мотивации сотрудников и удовлетворённости работой. Упрощается адаптация новых сотрудников, возрастает уровень вовлечённости, снижается вероятность конфликтов. Работа команд становится более согласованной, а управленческие процессы — предсказуемыми и эффективными.

Наконец, в условиях цифровизации и глобализации рынка компании вынуждены активно использовать новые технологии, чтобы оставаться конкурентоспособными. Автоматизация управленческой деятельности перестаёт быть преимуществом и становится обязательным условием успешного функционирования бизнеса. Особенно это касается организаций, занятых разработкой программного обеспечения, где своевременность

выполнения проектов напрямую влияет на репутацию и финансовую устойчивость компании. Таким образом, разработка и внедрение информационной системы управления проектами для компании-разработчика бухгалтерских программ представляет собой важное и своевременное решение. Оно направлено на повышение эффективности управленческой деятельности, снижение ошибок, улучшение аналитики и создание целостной цифровой экосистемы внутри компании. Такая система станет основой для дальнейшего развития, позволяя организации адаптироваться к изменяющимся условиям рынка, улучшать качество выпускаемых продуктов и достигать новых уровней профессионализма.

1.2 Предметная характеристика объекта исследования

Объектом исследования в рамках данной выпускной квалификационной работы выступает система управления проектами внутри компании, специализирующейся на разработке бухгалтерского программного обеспечения. Это организация, где ключевыми элементами управленческой деятельности являются планирование задач, их распределение между исполнителями, отслеживание прогресса выполнения и анализ результатов для последующей оптимизации процессов. Компания, как правило, имеет сложную внутреннюю структуру, включающую несколько функциональных подразделений, взаимодействие которых требует высокой степени координации и синхронизации.

Предметом исследования является проектирование и реализация информационной системы, ориентированной на автоматизацию управленческих процессов в указанной сфере. Особое внимание уделяется не только технической реализации системы, но и её интеграции в существующие бизнес-процессы, а также возможности повышения эффективности работы менеджеров и руководителей проектов. Основная цель заключается в том, чтобы создать гибкий и надёжный инструмент, способный централизовать

информацию о задачах, повысить прозрачность рабочих процессов и снизить административную нагрузку на управленческий персонал.

Особенность выбранной сферы деятельности — это её высокая регулируемость, где соблюдение строгих стандартов точности и контроля становится «не просто нормой, а основой всей деятельности». Программы для бухгалтерии напрямую связаны с финансовыми операциями клиентов, налоговым законодательством и отчётностью, что делает их «крайне чувствительными к любым неточностям». Любые ошибки или задержки в разработке могут обернуться не только юридическими последствиями, но и «глубоким ущербом для репутации компании». Поэтому внедрение информационной системы, способной минимизировать риски, повысить качество продуктов и ускорить выполнение задач, «становится не просто выгодным решением, но необходимостью».

Центральным элементом разрабатываемой информационной системы является обработка данных о проектах и участниках. Эти данные формируют основу всех управленческих решений и аналитических выводов. В контексте компании, занимающейся разработкой бухгалтерских программ, информация о проекте включает в себя:

- название и описание проекта;
- ответственные сотрудники или команды;
- участники;
- текущий статус;
- документы.

Система должна обеспечивать возможность сбора, хранения, фильтрации и представления этих данных в удобном для восприятия виде. При этом важно не просто хранить информацию, но и предоставлять её в формате, который позволяет быстро принимать управленческие решения. Например, менеджер должен иметь возможность одним взглядом понять, какие задачи

находятся в зоне риска по срокам, кто из сотрудников перегружен, а кто может взять дополнительные задачи.

Кроме того, система должна обеспечивать гибкость в организации данных.

Без использования специализированной системы все процессы зачастую выполняются вручную или через разрозненные инструменты, что приводит к снижению прозрачности, увеличению времени на согласования и большему количеству ошибок. Разрабатываемая информационная система призвана объединить все эти этапы в одном цифровом пространстве, обеспечивая не только автоматизацию, но и упрощение управленческих действий.

Интеграция с уже существующими в компании системами – ещё один важный аспект исследования. Современная IT-компания, особенно если она работает в области разработки программного обеспечения, обычно использует целый набор корпоративных инструментов:

– ERP-системы для управления ресурсами;
– CRM-системы для взаимодействия с клиентами;
– системы документооборота;
– платформы для хранения знаний и внутренних документов;
– системы контроля версий (например, GitLab, GitHub);
– чат-платформы (Slack, Microsoft Teams и т. д.).

Информационная система управления задачами должна быть совместима с этими платформами, обеспечивая двусторонний обмен данными. Например, при добавлении нового проекта в систему он может автоматически создаваться в соответствующем репозитории Git. Такая интеграция позволяет избежать дублирования информации, повысить скорость реакции и улучшить общую согласованность процессов.

Учитывая, что компания занимается разработкой программного обеспечения в финансовой сфере, вопросы безопасности играют особую роль. Информационная система будет обрабатывать большое количество

конфиденциальных данных, включая личные данные сотрудников, информацию о клиентах, детали проектов, внутренние коммуникации и т. д.

Поэтому при разработке системы необходимо предусмотреть следующие меры защиты:

– шифрование передаваемых и хранимых данных;
– разделение прав доступа по ролям
– защиту от SQL-инъекций, XSS и CSRF-атак;
– резервное копирование данных.

Эти меры позволят не только защитить данные от внешних угроз, но и обеспечить доверие со стороны сотрудников и клиентов, что особенно важно при внедрении новых технологий.

Для более глубокого понимания условий внедрения системы важно рассмотреть организационную структуру компании-заказчика. Как правило, компания, занимающаяся разработкой бухгалтерских программ, имеет следующие ключевые подразделения, как показано на рисунке 1:

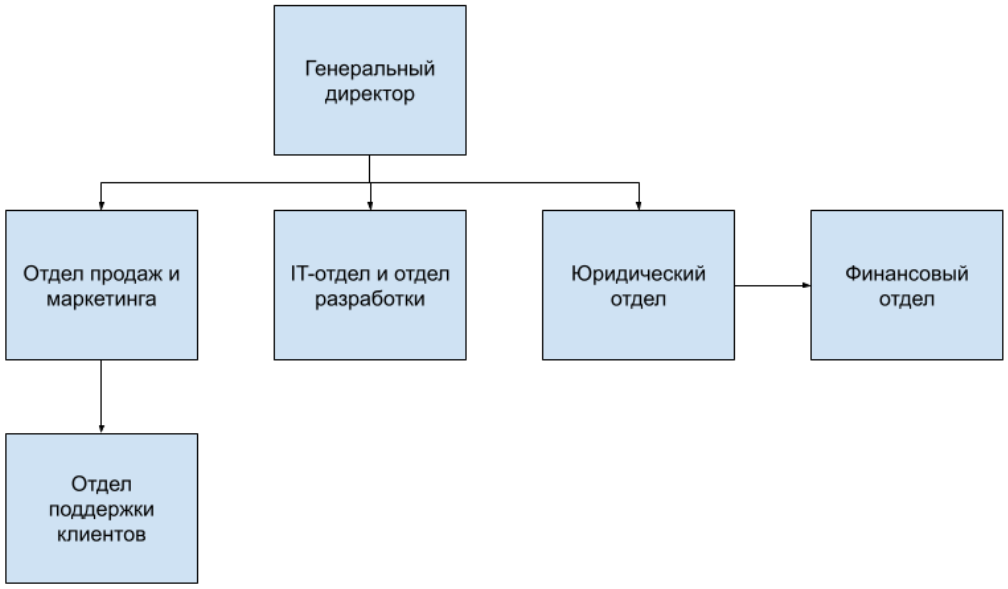


Рисунок 1 – Структура предприятия

Генеральный директор – отвечает за стратегическое развитие компании, принимает ключевые решения и контролирует работу всех отделов.

Финансовый отдел – обеспечивает устойчивость компании, ведёт бюджетирование, отчётность и взаимодействует с налоговыми органами.

Отдел продаж и маркетинга – занимается продвижением продукта, формированием спроса и привлечением клиентов.

Отдел поддержки клиентов – обеспечивает качественное обслуживание, помогает пользователям решать проблемы и собирает обратную связь.

IT-отдел и отдел разработки – основная группа, занимающаяся непосредственно созданием программного обеспечения, его тестированием и сопровождением.

Юридический отдел – обеспечивает правовое сопровождение сделок, проверяет договоры и контролирует соблюдение норм законодательства.

Именно в таком окружении будет внедряться разрабатываемая информационная система. Её успешное функционирование зависит от того, насколько хорошо она будет интегрирована в текущие процессы каждого подразделения и насколько удобно будет ею пользоваться сотрудникам разных уровней подготовки.

Таким образом, объект и предмет исследования охватывают широкий спектр задач, связанных с созданием информационной системы, ориентированной на автоматизацию управленческих процессов в компании, занимающейся разработкой бухгалтерского программного обеспечения. Создание такой системы позволит повысить эффективность работы, снизить риск ошибок, улучшить аналитическую базу и создать целостную цифровую экосистему внутри компании. Это станет важным шагом на пути к цифровой трансформации и устойчивому развитию организации в условиях растущей конкуренции и усложняющегося рынка.

Кроме того, автоматизация управленческой деятельности положительно сказывается на взаимодействии между различными подразделениями: разработчиками, тестировщиками, аналитиками, маркетологами и технической поддержкой. Единая система позволяет избежать дублирования

усилий, минимизировать ошибки при передаче задач и повысить общую прозрачность работы. Это, в свою очередь, создаёт благоприятные условия для масштабирования бизнеса, улучшения качества выпускаемых продуктов и усиления конкурентных позиций на рынке.

Эффективно спроектированная организационная модель, дополненная внедрённой информационной системой, создаёт «прочный фундамент для устойчивого роста и развития компании». Такой подход позволяет не только успешно выполнять текущие проекты, но и «строить долгосрочные планы, основанные на точных данных, а не на предположениях».

1.3 Обзор существующих методов решения

Анализ современного рынка цифровых инструментов, предназначенных для автоматизации управленческих процессов, демонстрирует широкое разнообразие доступных решений. Каждая из представленных на рынке систем предлагает уникальный набор функций, ориентированный на определённые сценарии использования, типы компаний и уровни зрелости внутренних процессов. При выборе подходящего решения важно учитывать не только технические возможности платформы, но и её соответствие специфике деятельности организации, особенностям корпоративной культуры и требованиям к безопасности данных.

Ниже приведён обзор наиболее популярных систем управления проектами и задачами, которые могут быть потенциально применимы или послужить основой для проектирования собственной информационной системы.

YouGile представляет собой облачную платформу, ориентированную на автоматизацию процессов управления задачами и проектами. Платформа активно используется как малым, так и средним бизнесом, благодаря своей простоте освоения, удобному интерфейсу и локализации под российские реалии. Для компаний, занимающихся разработкой бухгалтерского программного обеспечения, YouGile может служить базовым инструментом в

управлении проектами, особенно на начальных этапах внедрения цифровых практик.

Основные преимущества системы:

– интуитивно понятный интерфейс, который позволяет быстро обучить сотрудников работе с платформой без значительных затрат времени;

– поддержка различных методологий управления задачами: Kanban-доски, списки задач, календари, таймлайны – всё это делает YouGile универсальным решением для разных типов проектов;

– возможность назначения задач конкретным сотрудникам или группам, установки дедлайнов, приоритетов и отслеживания прогресса выполнения;

– интеграция с популярными сервисами: Google Drive, Microsoft Office, CRM-системами, что позволяет объединять данные и минимизировать необходимость переключения между разными платформами;

– работа в режиме реального времени, что особенно важно для команд, находящихся в разных городах или работающих удалённо;

– доступность бесплатной версии и относительно невысокая стоимость подписки делают YouGile привлекательным решением для небольших и средних организаций.

Однако, несмотря на свои достоинства, YouGile имеет ряд ограничений, которые могут стать проблемой при использовании в более сложных сценариях:

– ограниченные возможности аналитики и формирования отчетов в базовых тарифах. Это может быть недостатком для компаний, которым требуются детальные метрики эффективности, прогнозирование нагрузки и выявление «узких мест» в процессах;

– относительно низкая известность по сравнению с международными аналогами, что может вызвать сомнения у руководителей, предпочитающих использовать проверенные мировые бренды;

– недостаточная гибкость в плане настройки прав доступа и ролевой модели, особенно если требуется точное соответствие внутренним политикам информационной безопасности.

Тем не менее, YouGile остаётся одним из интересных вариантов для внедрения в компанию, особенно если требуется оперативное внедрение простого, но функционального инструмента. Однако, учитывая специфику разработки бухгалтерских программ и повышенные требования к контролю, аналитике и защите данных, целесообразно рассмотреть возможность создания собственной информационной системы, которая будет максимально соответствовать потребностям компании.

Пример работы программы YouGile представлен на рисунке 1.

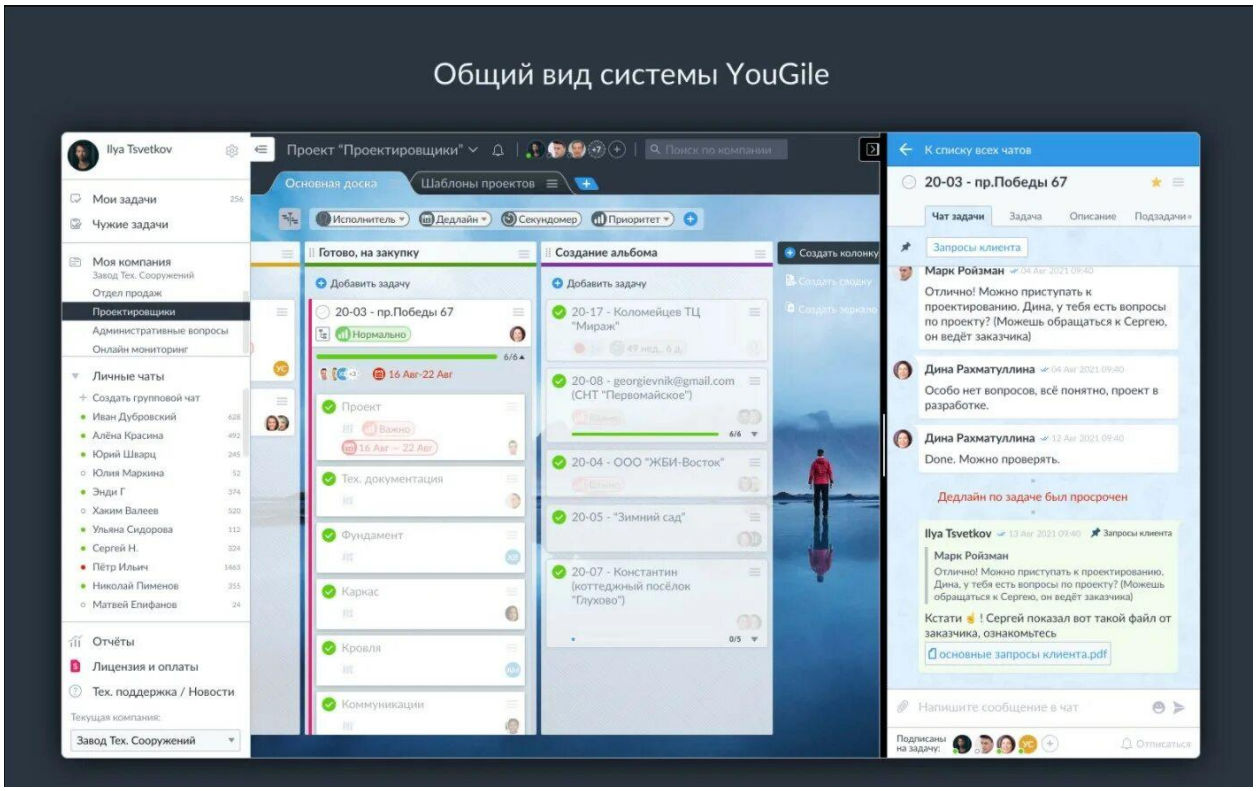


Рисунок 2 – Пример работы YouGile

Trello представляет собой одну из наиболее известных и распространённых платформ для управления проектами и задачами. Основным принципом её работы является использование Kanban-методологии, при которой информация о задачах отображается в виде

карточек, перемещаемых по колонкам в зависимости от стадии выполнения. Например, такие столбцы могут называться «Нуждается в обсуждении», «В работе», «На проверке», «Завершено» и т. д. Такая визуализация делает процесс управления задачами более понятным и наглядным, особенно для пользователей, не имеющих опыта в формальных управленческих методологиях.

Одним из ключевых преимуществ Trello является его простота использования и интуитивно понятный интерфейс, которые позволяют внедрять систему без длительного обучения сотрудников. Платформа активно используется как стартапами, так и крупными корпорациями для решения широкого круга задач – от повседневного планирования до координации масштабных проектов. Для небольших команд или отдельных отделов Trello становится удобным решением благодаря минимальному порогу входа и возможности быстрого старта

Среди основных функциональных возможностей можно выделить:

– гибкая организация задач: каждая задача может содержать описание, сроки выполнения, исполнителей, чек-листы, файлы, комментарии, метки и другие элементы;

– поддержка совместной работы: возможность привлекать к задачам несколько участников, обсуждать детали прямо в интерфейсе, делиться обновлениями в реальном времени;

– интеграция с другими сервисами: через Power-Ups доступна связь с Google Drive, Dropbox, Slack, Jira, Zapier и многими другими популярными инструментами, что позволяет использовать Trello как часть единой цифровой экосистемы;

– доступность и стоимость: бесплатная версия предоставляет достаточный набор функций для базового управления задачами, а платные тарифы предлагают расширенные возможности, включая увеличенное количество вложений, дополнительные интеграции и аналитику.

Тем не менее, несмотря на свою популярность, Trello имеет ряд ограничений, которые могут быть критичными для организаций, работающих над сложными проектами, особенно таких, как компания по разработке бухгалтерского программного обеспечения, где важны точность, контроль и аналитическая поддержка.

К числу недостатков относятся:

– ограниченная поддержка сложных проектов: если речь идёт о многоэтапных разработках, требующих точного контроля за зависимостями задач, бюджетом, ресурсами и сроками, стандартный функционал Trello может оказаться недостаточным;

– отсутствие глубокой аналитики и отчетности: в отличие от специализированных систем управления проектами, Trello не предлагает автоматической генерации аналитических отчётов, диаграмм нагрузки, прогнозирования сроков или оценки эффективности команд;

– ограниченная настройка ролевой модели и прав доступа: для компаний, где требуется строгий контроль за доступом к данным, особенно в сфере финансов, этот фактор может стать серьёзным ограничением;

– рост стоимости при использовании расширенных возможностей: хотя базовая версия бесплатна, при необходимости подключения мощных интеграций, расширения функционала или работы с большими командами суммарные затраты на использование Trello могут вырасти значительно.

Кроме того, в условиях разработки бухгалтерского программного обеспечения, где критически важно соблюдение норм безопасности и требования регуляторов, вопрос соответствия Trello современным стандартам защиты информации также требует внимательного рассмотрения.

Trello остаётся одним из самых популярных инструментов управления задачами благодаря своей простоте, гибкости и широкому спектру возможностей. Он отлично подходит для малых команд, стартапов и проектов с ограниченной степенью сложности. Однако, если речь идёт о компании,

занимающейся разработкой программного обеспечения для финансовой сферы, где необходимы детальный контроль, аналитическая поддержка и высокий уровень информационной безопасности, использование Trello в качестве основной системы управления задачами может быть недостаточно эффективным. Пример работы программы Trello представлен на рисунке 2.

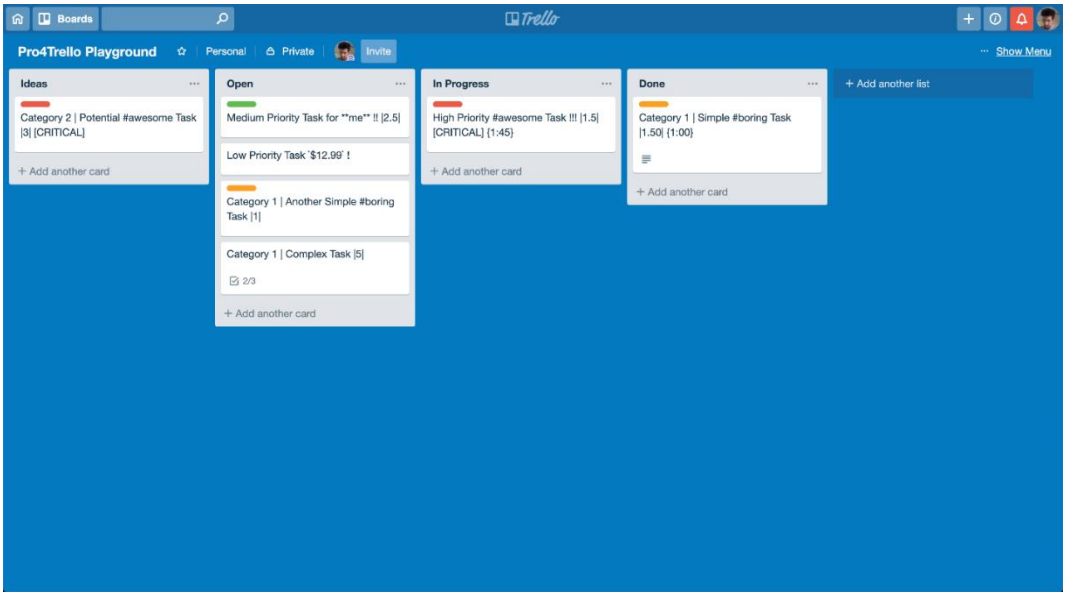


Рисунок 3 – Работа программы Trello

Jira – это одна из ведущих платформ в мире, предназначенная для управления задачами и проектами, особенно популярная среди IT-компаний, команд разработчиков программного обеспечения и организаций, работающих по гибким методологиям (Agile, Scrum, Kanban). Разработанная компанией Atlassian, Jira изначально была создана как инструмент для трекинга ошибок и дефектов в коде, но со временем превратилась в мощную систему управления проектами, поддерживающую широкий спектр сценариев использования.

Для компаний, занимающихся разработкой бухгалтерского программного обеспечения, Jira представляет особый интерес, поскольку позволяет управлять сложными процессами разработки, отслеживать задачи на разных этапах жизненного цикла продукта, а также обеспечивает высокий уровень контроля над качеством выпускаемых решений. В отличие от более простых систем, таких как Trello или YouGile, Jira предлагает гораздо более

глубокую функциональность, ориентированную на автоматизацию рутинных процессов, обеспечение прозрачности работы и повышение общей производительности

Основные преимущества Jira:

– гибкая настройка под конкретные процессы;
– управление задачами и баг-трекинг;
– поддержка Agile-подходов;
– интеграция с DevOps-инструментами;
– мощные аналитические и отчетные возможности;
– масштабируемость и безопасность;
– ограничения и потенциальные трудности при внедрении.

Несмотря на свои очевидные преимущества, использование Jira в качестве основной системы управления задачами связано с рядом сложностей, которые могут стать препятствием для некоторых организаций:

– сложность внедрения и обучения;
– высокая стоимость лицензирования;
– избыточный функционал для простых задач;
– требования к ИТ-инфраструктуре.

Jira – это мощная и гибкая система управления задачами, которая может значительно повысить эффективность работы технических команд, особенно в сферах, где требуется детальный контроль над процессами разработки. Однако для компаний, занимающихся созданием бухгалтерского программного обеспечения, она может быть, как выгодным решением, так и источником сложностей, если не учитывать специфику деятельности, размер организации и уровень подготовки персонала.

Поэтому вместо выбора между существующими системами целесообразнее рассмотреть разработку собственной информационной системы, которая объединит лучшие практики управления задачами, обеспечит полную интеграцию с корпоративными процессами и будет

соответствовать всем требованиям безопасности, аналитики и масштабируемости. Такое решение позволит создать инструмент, максимально адаптированный под нужды целевой компании, с минимальными барьерами для внедрения и максимальной пользой для бизнеса. Пример работы программы Jira представлен на рисунке 3.

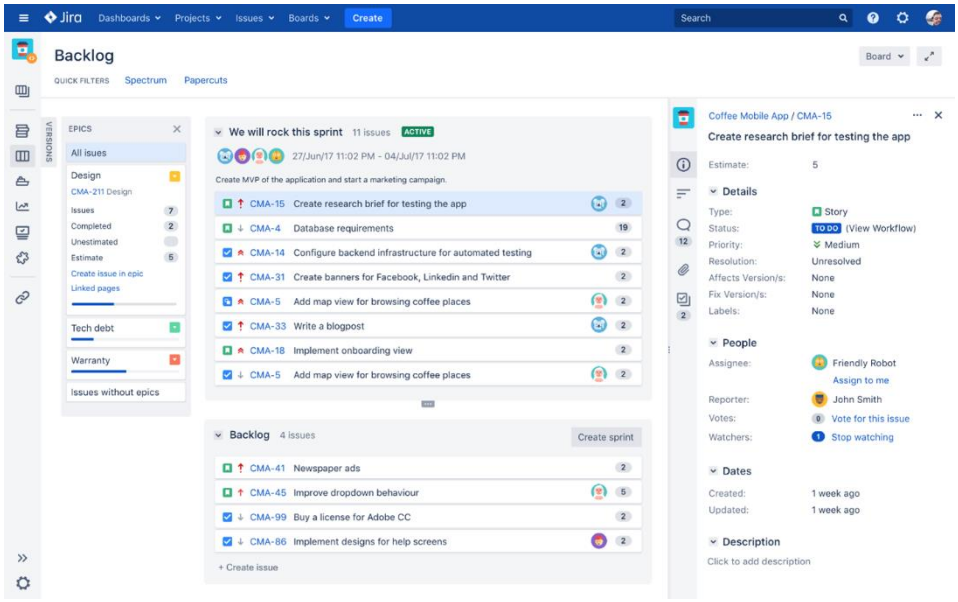


Рисунок 4 – Работа программы Jira

Таким образом, анализ существующих инструментов управления задачами показывает, что, несмотря на их доступность, удобство использования и наличие базового функционала, у большинства из них имеются существенные ограничения. Недостаточная гибкость настройки под специфические процессы, ограниченные аналитические возможности, зависимость от сторонних поставщиков и потенциальные риски в области информационной безопасности делают такие решения недостаточно эффективными для организаций с особыми требованиями к защите данных и уровню автоматизации.

Для компаний, занимающихся разработкой бухгалтерского программного обеспечения, где важны точность, контроль, соответствие нормативам и высокий уровень защиты информации, использование готовых решений может оказаться недостаточным. В таких случаях наиболее

целесообразным вариантом становится разработка собственной информационной системы, которая будет полностью соответствовать внутренним бизнес-процессам, обеспечивать необходимый уровень прозрачности и предоставлять расширенные аналитические и управленческие функции.

Создание оригинальной системы позволяет реализовать ряд ключевых преимуществ:

– полная адаптация под нужды компании, включая специфику проектов, методологии управления и особенности взаимодействия между отделами;

– контроль над данными и их безопасностью, что особенно важно при работе с конфиденциальной финансовой информацией;

– гибкая интеграция с корпоративными системами, такими как ERP, CRM и платформы документооборота, что способствует формированию единой цифровой экосистемы внутри организации;

– масштабируемость, позволяющая развивать систему по мере роста компании и изменения её потребностей.

Поэтому для компании-разработчика бухгалтерских программ внедрение собственной информационной системы управления задачами может стать стратегически правильным решением, направленным на повышение эффективности, снижение административных барьеров и укрепление конкурентных позиций на рынке.

1.4 Формулировка задач исследования и методики их решения

Определение задач и выбор подхода к разработке информационной системы требует чёткого понимания конечных целей, особенностей предметной области и этапов реализации проекта.

Для достижения этих целей предполагается выполнить ряд взаимосвязанных задач, охватывающих все этапы жизненного цикла проекта – от анализа текущих процессов до внедрения и сопровождения системы.

Первым этапом исследования выступает «комплексный аудит текущих бизнес-процессов в рамках целевой организации». Особое внимание направлено на выявление того, как в настоящий момент организовано управление проектами, «какие инструменты применяют менеджеры и с какими трудностями они сталкиваются при распределении задач и отслеживании их выполнения».

Следующая задача – определение технической базы и выбор подходящих инструментов для реализации. На этом этапе проводится сравнительный анализ возможных решений: языков программирования, фреймворков, систем управления базами данных, а также подходов к архитектуре (например, MVC, микросервисы). Учитываются требования к масштабируемости, безопасности, производительности и возможности последующей модификации.

После завершения аналитической и проектной фаз наступает стадия разработки прототипа системы. Он выступает не только как «визуальное отражение будущего продукта», но и как инструмент для проверки функциональных модулей на работоспособность. Прототип предназначен для проведения первичного тестирования и «получения обратной связи от конечных пользователей на ранних этапах разработки».

Затем следует полноценная разработка программного кода, основанная на утверждённых спецификациях. При этом применяются современные практики разработки: использование систем контроля версий, принципы SOLID, паттерны проектирования и другие подходы, направленные на обеспечение качества и надёжности кода.

2 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

2.1 Архитектура информационной системы

Техническая архитектура является первым уровнем архитектуры информационной системы. Она описывает все аппаратные средства, использующиеся при выполнении заявленного набора функций, а также включает средства обеспечения сетевого взаимодействия и надёжности. В технической архитектуре указываются периферийные устройства, сетевые коммутаторы и маршрутизаторы, жёсткие диски, оперативная память, процессоры, соединительные кабели, источники бесперебойного питания и т.п. Программная архитектура представляет собой совокупность компьютерных программ, предназначенных для решения конкретных задач. Данный тип архитектуры необходим для описания приложений, входящих в состав информационной системы. На данном уровне описывают программные интерфейсы, компоненты и поведение. Архитектура данных объединяет в себе как физические хранилища данных, так и средства управления данными. Кроме того, в неё входят логические хранилища данных, а при ориентированности рассматриваемой компании на работу со знаниями, может быть выделен отдельный уровень.

– архитектура знаний (Knowledge architecture). На этом уровне описываются логические и физические модели данных, определяются правила целостности, составляются ограничения для данных.

Следует особенно выделить уровень ИТ архитектуры, поскольку он является связующим. На нём формируется базовый набор сервисов, которые используются как на уровне программной архитектуры, так и на уровне архитектуры данных. Если какая–либо особенность функционирования для этих двух уровней не была предусмотрена, то сильно возрастает вероятность сбоев в работе, а, следовательно, потерь для бизнеса. В некоторых случаях невозможно разделить ИТ–архитектуру и архитектуру отдельного приложения.

Такое возможно при высокой степени интеграции приложений. Примером ИТ–архитектуры может служить SharePoint от компании Microsoft. Этот продукт предоставляет сервисы для совместной работы и хранения информации, что является очень важным аспектом функционирования любой компании. Его базовые системные модули относятся к ИТ–архитектуре, а пользовательские – к программной. Основной функцией ИТ–архитектуры является обеспечение функционирования важных бизнес–приложений для достижения обозначенных бизнес–целей. Если некоторая функция требуется сразу в нескольких приложениях, то её следует перенести на уровень ИТ–архитектуры, тем самым повысив интеграцию системы и снизить сложность архитектуры приложений. Последним в иерархии является уровень бизнес–архитектуры или архитектуры бизнес–процессов. На этом уровне определяются стратегии ведения бизнеса, способы управления, принципы организации и ключевые процессы, представляющие для бизнеса огромную важность. Архитектура информационной системы играет ключевую роль в обеспечении её надёжности, производительности и возможности дальнейшего развития. Она определяет логическую структуру приложения, взаимодействие между его компонентами и соответствие как функциональным, так и нефункциональным требованиям, предъявляемым к системе. При разработке решения для компании, специализирующейся на создании бухгалтерского программного обеспечения, особое внимание было уделено выбору архитектурного подхода, поскольку от него зависят такие важные аспекты, как масштабируемость, безопасность, простота сопровождения и возможность адаптации к изменяющимся бизнес–условиям [2].

Для реализации проекта была выбрана клиент–серверная архитектура, которая позволила распределить функционал системы на три основных уровня: клиентский интерфейс, серверную логику и уровень хранения данных. Такая организация способствует модульности, упрощает развитие системы и позволяет эффективно использовать ресурсы вычислительной среды.

Интерфейс пользователя был реализован с использованием современных технологий фронтенд–разработки – HTML5, CSS3 и JavaScript. Для создания адаптивного дизайна, корректно отображающегося на устройствах различного формата, применялся популярный CSS–фреймворк Bootstrap. Это обеспечило высокую степень кросс–платформенности и удобство использования вне зависимости от типа устройства.

Для динамического обновления содержимого страниц без необходимости полной перезагрузки был внедрён JavaScript–фреймворк jQuery. Он позволил сделать пользовательский интерфейс более отзывчивым и повысить общее качество взаимодействия с системой.

Серверная логика реализована с использованием языка программирования PHP, что обусловлено его широкой применимостью, развитой экосистемой и хорошей совместимостью с различными базами данных. В качестве платформы для построения серверной части была выбрана CMS ModX, которая отличается гибкой архитектурой, возможностью быстрой настройки административной панели и наличием мощного механизма управления контентом.

Для работы с данными использовалась объектно–реляционная модель через ORM RedBeanPHP. Этот инструмент позволил значительно упростить взаимодействие с базой данных за счёт использования объектно–ориентированного подхода и минимизировал необходимость ручного написания SQL–запросов, что положительно сказалось на скорости разработки и читаемости кода [4].

В качестве системы управления базами данных (СУБД) была выбрана PHPMyAdmin версии 5.7, установленная в окружении OpenServer. PHPMyAdmin является надёжным, высокопроизводительным решением, поддерживающим стандарты SQL и обладающим хорошими возможностями масштабирования. Благодаря своей устойчивости и совместимости с рядом

инструментов, она стала оптимальным выбором для проекта такого уровня сложности.

Таким образом, архитектура разработанной информационной системы соответствует современным требованиям к построению программных решений. Она обеспечивает устойчивость в работе, простоту сопровождения и потенциал для масштабирования.

2.2 Проектирование базы данных

Разработка информационной системы для компании, специализирующейся на создании бухгалтерских программ, требует тщательного подхода к формированию структуры базы данных. База данных выступает основой всей системы – она обеспечивает хранение, обработку и анализ информации, а также поддерживает логическую целостность и надежность функционирования приложения.

Таблица 1 – Основные сущности и типы операций

Сущность	Основные операции	Частота использования
Пользователи	Регистрация, вход, изменение данных	Высокая
Задачи	Создание, обновление, удаление, назначение	Очень высокая
Проекты	Создание, привязка задач, установка дат	Средняя
Комментарии	Добавление, редактирование	Высокая
Файлы	Загрузка, скачивание, удаление	Высокая
Сессии	Логирование действий, проверка активности	Очень высокая
Фото	Прикрепление к задачам и проектам	Средняя

Концептуальная модель представляет собой абстрактное описание структуры данных, не зависящее от конкретной СУБД. Для её реализации использовалась ER-модель (модель "сущность-связь"), которая позволила наглядно показать взаимосвязи объектов как показано в таблице 2.

Между ними установлены следующие типы связей:

– один-ко-многим (например, один проект содержит множество задач),

– многие-ко-многим (например, пользователь может быть участником нескольких проектов).

Таблица 2 – Примеры связей между сущностями

Сущность 1	Сущность 2	Тип связи	Описание
Пользователь	Проект	M:N	Пользователь может участвовать в нескольких проектах
Проект	Задача	1:N	Один проект содержит несколько задач
Задача	Комментарий	1:N	Одна задача имеет множество комментариев
Задача	Фото	1:N	К одной задаче можно прикрепить несколько фото
Пользователь	Сессия	1:N	У одного пользователя может быть много сессий

Логическая модель была построена на основе концептуальной, с учетом особенностей реляционной модели данных. Для каждой сущности были определены поля, типы данных, первичные и внешние ключи как показано в таблицах 3, 4, 5, 6, 7, 8, 9, 10, 11.

Таблица 3 – Пользователи

Поле	Тип данных	Описание
id	INT (PK)	Уникальный идентификатор
имя	VARCHAR(100)	Имя пользователя
фамилия	VARCHAR(100)	Фамилия пользователя
email	VARCHAR(100)	Электронная почта
пароль	TEXT	Хэшированный пароль
роль_id	INT (FK)	Внешний ключ на таблицу "Роли"
дата_регистрации	DATE	Дата регистрации
аватар	TEXT (URL)	Ссылка на изображение аватара

Таблица 4 – Роли

Поле	Тип данных	Описание
id	INT (PK)	Уникальный идентификатор

Продолжение таблицы 4

Поле	Тип данных	Описание
название	VARCHAR(50)	Название роли (администратор, менеджер, разработчик)
описание	TEXT	Описание прав доступа

Таблица 5 – Проекты

1	2	3
Поле	Тип данных	Описание
id	INT (PK)	Идентификатор проекта
название	VARCHAR(255)	Наименование проекта
описание	TEXT	Подробное описание проекта
старт	DATE	Дата начала
окончание	DATE	Дата завершения
статус	ENUM('активный', 'завершен')	Текущий статус проекта
менеджер_id	INT (FK)	Внешний ключ на таблицу "Пользователи"

Таблица 6 – Задачи

Поле	Тип данных	Описание
id	INT (PK)	Идентификатор задачи
заголовок	VARCHAR(255)	Название задачи
описание	TEXT	Подробное описание
статус	ENUM('в работе', 'готова', 'на паузе')	Текущий статус задачи
приоритет	ENUM('низкий', 'средний', 'высокий')	Уровень приоритета
дедлайн	DATE	Срок выполнения
проект_id	INT (FK)	Внешний ключ на таблицу "Проекты"
исполнитель_id	INT (FK)	Внешний ключ на таблицу "Пользователи"

Таблица 7 – Комментарии

Поле	Тип данных	Описание
id	INT (PK)	Идентификатор комментария
текст	TEXT	Текст сообщения
дата_публикации	TIMESTAMP	Дата и время публикации
автор_id	INT (FK)	Внешний ключ на таблицу "Пользователи"
задача_id	INT (FK)	Внешний ключ на таблицу "Задачи"

Таблица 8 – Файлы

Поле	Тип данных	Описание
id	INT (PK)	Идентификатор файла
название	VARCHAR(255)	Название файла
ссылка	TEXT (URL)	Ссылка на хранилище
тип	VARCHAR(50)	Формат файла (PDF, DOCX и др.)
проект_id	INT (FK)	Внешний ключ на таблицу "Проекты"

Таблица 9 – Сессии

Поле	Тип данных	Описание
id	INT (PK)	Идентификатор сессии
пользователь_id	INT (FK)	Внешний ключ на таблицу "Пользователи"
начало	TIMESTAMP	Дата и время начала сессии
окончание	TIMESTAMP	Дата и время окончания сессии
ip_адрес	VARCHAR(15)	IP-адрес пользователя

Таблица 10 – Фото

Поле	Тип данных	Описание
id	INT (PK)	Идентификатор фото
ссылка	TEXT (URL)	Ссылка на изображение
описание	TEXT	Подпись к фото
задача_id	INT (FK)	Внешний ключ на таблицу "Задачи"

Таблица 11 – Участники проектов (связь M:N)

Поле	Тип данных	Описание
пользователь_id	INT (FK)	Внешний ключ на таблицу "Пользователи"
проект_id	INT (FK)	Внешний ключ на таблицу "Проекты"
роль_в_проекте	VARCHAR(50)	Должность в проекте (разработчик, тестировщик и др.)

Нормализация проводилась с целью исключения дублирования информации и повышения целостности данных. Были применены следующие нормальные формы:

– 1НФ: каждое поле содержит только одно значение.

– 2НФ: исключены частичные зависимости от составного ключа.

– 3НФ: устранены транзитивные зависимости.

Пример: информация о роли пользователя вынесена в отдельную таблицу, чтобы избежать повторения названий ролей в таблице «Пользователи».

Проектирование физической модели. На данном этапе была разработана физическая модель базы данных, учитывающая специфику выбранной СУБД (PHPMyAdmin). Основные задачи этапа включали оптимизацию производительности, обеспечение безопасности данных и автоматизацию рутинных операций. Ниже приведены детализированные таблицы для каждого аспекта физического проектирования.

Для повышения эффективности хранения и доступа к данным были определены файловые группы, типы хранилищ и стратегии размещения таблиц.

Таблица 12 – Файловые группы и стратегии хранения

Файловая группа	Назначение	Тип хранилища	Особенности
PRIMARY	Хранение активных данных	SSD (быстрый доступ)	Основные таблицы: Пользователи, Задачи, Проекты

Продолжение таблицы 12

Файловая группа	Назначение	Тип хранилища	Особенности
ARCHIVE	Хранение архивных данных	HDD (низкая стоимость)	Таблицы: Сессии, старые проекты
LOGS	Хранение журналов и логов	SSD (высокая скорость записи)	Таблицы: Логирование изменений, Сессии

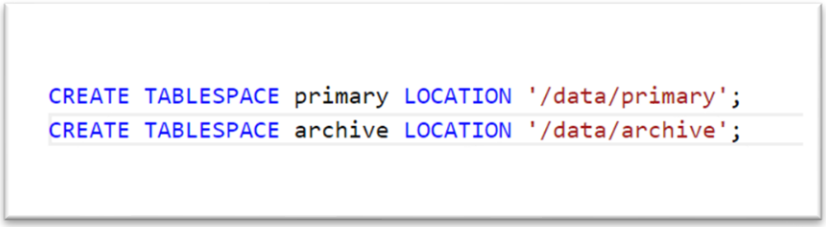


Рисунок 5 – Пример настройки файловой группы в PHPMyAdmin

Индексы были созданы для часто используемых полей, чтобы ускорить выполнение запросов и обеспечить уникальность данных.

Таблица 13 – Индексы и их назначение

Таблица	Поле	Тип индекса	Назначение
Пользователи	email	UNIQUE INDEX	Уникальность email, быстрый поиск
Задачи	дедлайн	B-TREE INDEX	Фильтрация задач по срокам
Проекты	старт	DATE INDEX	Поиск проектов по периоду
Комментарии	задача_id	HASH INDEX	Быстрый доступ к комментариям задачи
Файлы	проект_id	BTREE INDEX	Фильтрация файлов по проекту

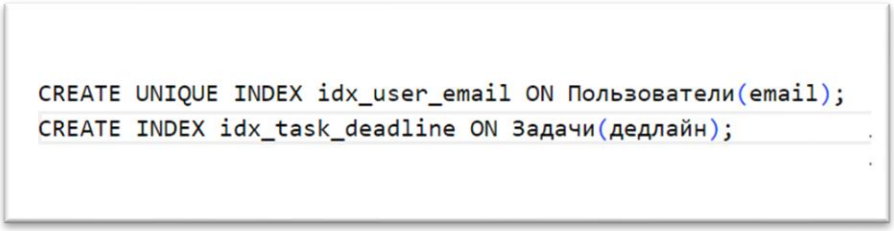


Рисунок 6 – Пример создания индекса

Для автоматизации рутинных операций были разработаны хранимые процедуры и функции.

Таблица 14 – Хранимые процедуры и их функционал

1	2	3	4
Название процедуры	Входные параметры	Выходные данные	Описание
get_project_tasks()	проект_id	Таблица задач	Возвращает все задачи проекта
create_task()	заголовок,описание,проект_id	Id новой задачи	Создает новую задачу с автоматическим присвоением исполнителя
update_task_status()	задача_id,новый_статус	Сообщение об успехе/ошибке	Обновляет статус задачи и записывает изменение в журнал
generate_report()	период_начала,период_окончания	PDF-отчет	Формирует отчет по выполненным задачам за указанный период

```
CREATE OR REPLACE FUNCTION get_project_tasks(INT)
RETURNS TABLE(id INT, заголовок VARCHAR, статус VARCHAR) AS $$
BEGIN
    RETURN QUERY
    SELECT id, заголовок, статус FROM Задачи WHERE проект_id = $1;
END;
$$ LANGUAGE plpgsql;
```

Рисунок 7 – Пример хранимой процедуры

Триггеры были реализованы для автоматизации действий при изменении данных.

Таблица 15 – Триггеры и их назначение

Таблица	Событие	Триггер	Логика работы
Задачи	AFTER UPDATE	after_task_update	Логирует изменения статуса задачи в таблицу Логи задач
Проекты	BEFORE DELETE	prevent_project_delete	Запрещает удаление проекта, если в нем есть задачи
Пользователи	AFTER INSERT	send_welcome_email	Отправляет приветственное письмо новому пользователю
Сессии	AFTER INSERT	log_user_activity	Записывает активность пользователя в журнал

```
CREATE OR REPLACE FUNCTION after_task_update()
RETURNS TRIGGER AS $$
BEGIN
    INSERT INTO Логи_задач (задача_id, старый_статус, новый_статус, дата_изменения)
    VALUES (OLD.id, OLD.статус, NEW.статус, NOW());
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER task_status_change
AFTER UPDATE ON Задачи
FOR EACH ROW
EXECUTE FUNCTION after_task_update();
```

Рисунок 8 – Пример триггера

Для защиты данных были определены регулярные процедуры резервного копирования и восстановления. Представлены в таблице 16.

Таблица 16 – Стратегии резервного копирования

Тип бэкапа	Частота	Место хранения	Особенности
Полный бэкап	Еженедельно	Внешний сервер (AWS S3)	Полное копирование всех данных
Дифференциальный	Ежедневно	Локальное хранилище	Копируются только измененные данные
Логический	Каждые 15 минут	Облачное хранилище (Google Cloud)	Копирование журналов транзакций
Архивный	Ежемесячно	Офлайн-носители (магнитные ленты)	Долгосрочное хранение

```
# Полный бэкап
pg_dump -U admin -h localhost -F c -b -v -f "backup_full.dump" accounting_system

# Восстановление
pg_restore -U admin -h localhost -d accounting_system -v "backup_full.dump"
```

Рисунок 9 – Пример резервного копирования

Физическая модель базы данных была спроектирована с учетом технических возможностей СУБД и требований к производительности. Реализованные решения:

– индексы ускоряют выполнение запросов;

– хранимые процедуры и триггеры автоматизируют бизнес-логику;

– стратегии резервного копирования обеспечивают защиту от потери данных;

– оптимизация хранения снижает нагрузку на систему и улучшает доступ к данным.

Эти меры позволили создать надежную и эффективную инфраструктуру для информационной системы компании-разработчика бухгалтерских программ.

Спроектированная база данных полностью соответствует целям и задачам информационной системы для компании-разработчика бухгалтерских программ. Она обеспечивает:

– логическую целостность данных;

– высокую степень производительности;

– гибкость в масштабировании и дальнейшей доработке;

– прозрачность структуры для администраторов и разработчиков.

Этапы проектирования позволили создать надёжную основу для последующей реализации системы, способной удовлетворить потребности

современного предприятия в автоматизации рабочих процессов и управлении проектами [5].

2.3 Проектирование пользовательского интерфейса

Проектирование интерфейса является «одним из ключевых этапов разработки информационной системы», поскольку именно через него пользователи взаимодействуют с её функциональными возможностями. Удобство, логичность и доступность интерфейса «напрямую влияют как на скорость освоения системы, так и на эффективность её применения в повседневной работе». При создании интерфейса для компании, занимающейся разработкой бухгалтерского программного обеспечения, особое внимание было уделено не только современным стандартам дизайна, но и особенностям профессиональных рабочих процессов, присущих данной отрасли.

Одним из главных критериев стало «создание интуитивно понятного взаимодействия с системой». Это предполагает, что элементы управления, навигация, формы ввода данных и отображение информации должны соответствовать «ожиданиям пользователя, сформированным его повседневным опытом», чтобы минимизировать необходимость обучения или обращения к справочным материалам. Для реализации этого принципа были проведены исследования типовых сценариев использования, выделены ключевые категории пользователей и определены их роли: ответственный, наблюдатель, создатель. Каждая роль обладает уникальным уровнем доступа и набором действий, что легло в основу построения структуры интерфейса. Система была спроектирована с учётом необходимости использования на различных устройствах – от мобильных телефонов до десктопных компьютеров.

Особое внимание уделялось организации навигационной структуры, которая должна быть простой и интуитивно понятной. В верхней части экрана расположено горизонтальное меню, обеспечивающее быстрый переход между

основными разделами: «Главная», «Проекты», «Мой аккаунт», «Настройки», «Пользователи» и другие. Такое построение интерфейса способствует повышению продуктивности за счёт минимизации времени, затрачиваемого на поиск нужного функционала.

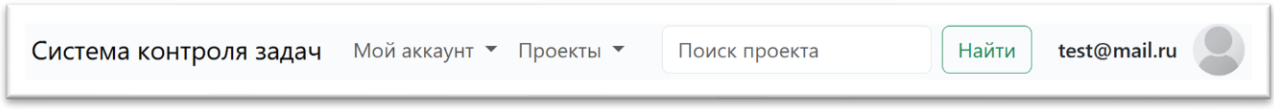


Рисунок 10 – Горизонтальное меню системы

При разработке форм ввода данных главный акцент был сделан на ясность и последовательность оформления. Все поля были структурированы логично, с учётом естественного порядка заполнения, присущего каждому типу операции. Каждое поле сопровождается краткой подсказкой, которая объясняет его назначение и допустимые значения для ввода. Если пользователь вносит некорректные данные, система выводит понятное сообщение об ошибке, указывающее, какие именно данные нужно исправить. Для ускорения работы были внедрены клиентские механизмы валидации, позволяющие проверять данные ещё до их отправки на сервер.

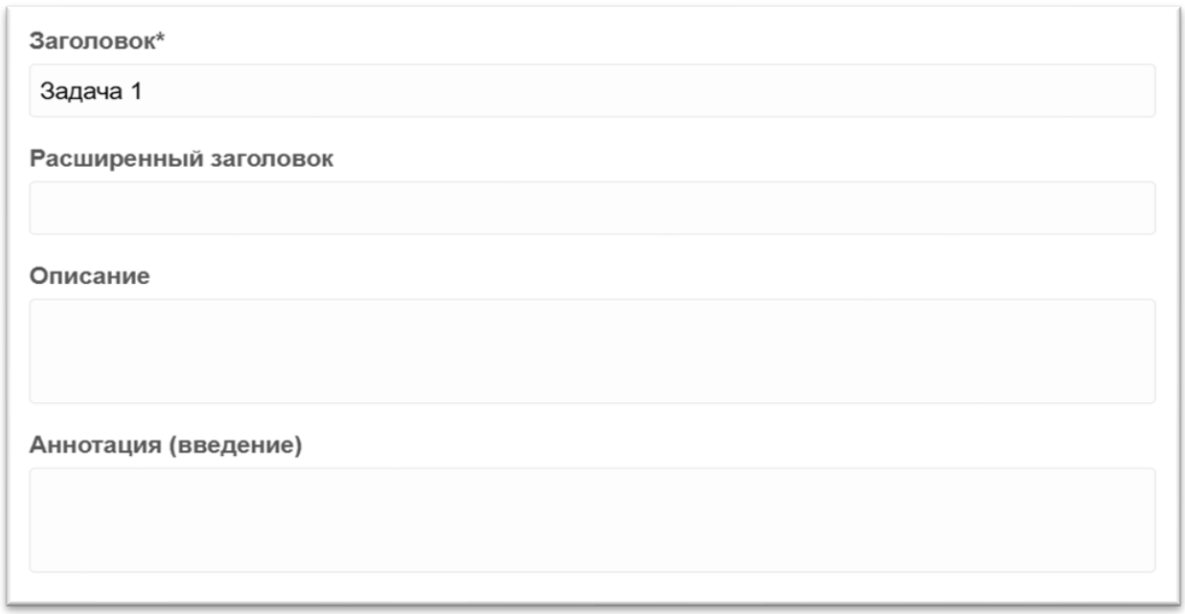


Рисунок 11 – Пример формы создания задачи

Для представления информации использовались различные методы: таблицы, карточки, списковые и табличные формы. Например, список проектов отображается в виде плиток, где каждая содержит название, статус, срок выполнения и состав команды. При клике на плитку открывается подробная информация о проекте, включающая описание, задачи, комментарии и связанные документы. Такой подход позволяет пользователю получать полное представление о текущем состоянии проекта без необходимости многократных переходов между разделами.

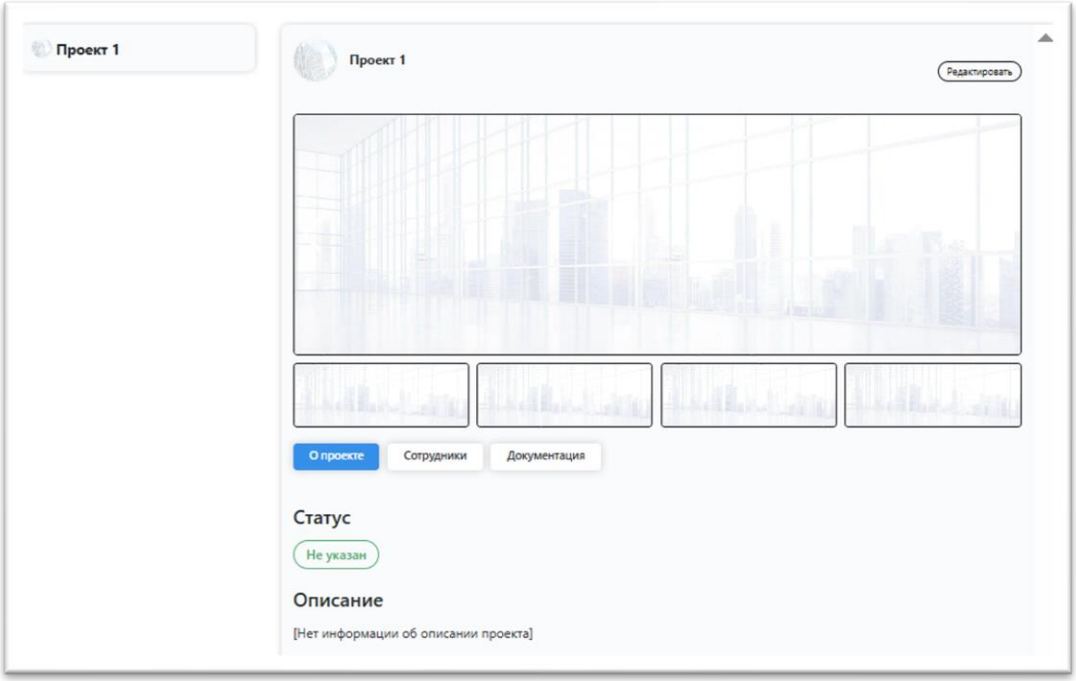


Рисунок 12 – Плитки проектов на главной странице

На этапе проектирования были созданы прототипы всех ключевых экранов. Для этого использовались инструменты вроде Figma и Adobe XD , что позволило визуализировать структуру и взаимодействие элементов до начала программной реализации.

Таким образом, разработка пользовательского интерфейса проводилась с учётом современных требований к юзабилити, доступности и эргономики. Итоговый интерфейс отличается логичной структурой, четкой навигацией, адаптивностью и высокой степенью удобства использования. Он создаёт

комфортные условия для работы сотрудников компании-разработчика бухгалтерских программ и способствует повышению общей производительности и качества управления проектами.

2.4 Подход к реализации и этапы работ

Термин HTML (HyperText Markup Language) означает «язык разметки гипертекстов», с помощью которого верстаются Web–страницы. В самом начале развития WWW регулировать стандарты разработки была призвана международная общественная организация, включающая в себя представителей фирм–разработчиков и исследовательских институтов — W3C (World Wide Web Consortium). Характеристики HTML:

- разработан специально для Web;
- открытый стандарт;
- в HTML включен гипертекст;
- в HTML включена поддержка мультимедиа.

Все что необходимо, чтобы прочитать HTML–документ – это Web–браузер, который интерпретирует элементы HTML и воспроизводит на экране документ в виде, который ему придает автор. Основное преимущество HTML заключается в том, что документ может быть просмотрен на Web–браузерах различных типов и на различных платформах.

HTML–документы могут быть созданы при помощи любого текстового редактора или специализированных HTML–редакторов и конвертеров. С другой стороны можно использовать специальные программы – редакторы HTML текстов [3].

JavaScript – это интерпретируемый язык программирования с объектно–ориентированными возможностями. С точки зрения синтаксиса базовый язык Java–Script напоминает C, C++ и Java такими программными конструкциями, как инструкция if, цикл while и оператор &&. Однако это подобие ограничивается синтаксической схожестью. JavaScript – это нетипизированный язык, т. е. в нем не требуется определять типы переменных.

Объекты в JavaScript отображают имена свойств на произвольные значения. Этим они больше напоминают ассоциативные массивы Perl, чем структуры C или объекты C++ или Java. Механизм объектно–ориентированного наследования JavaScript скорее похож на механизм прототипов в таких малоизвестных языках, как Self, и сильно отличается от механизма наследования в C++ и Java. Как и Perl, JavaScript – это интерпретируемый язык, и некоторые его инструменты, например, регулярные выражения и средства работы с массивами, реализованы по образу и подобию языка Perl.

Ядро языка JavaScript поддерживает работу с такими простыми типами данных, как числа, строки и булевы значения. Помимо, этого он обладает встроенной поддержкой массивов, дат и объектов регулярных выражений.

Обычно JavaScript применяется в вебброузерах, а расширение его возможностей за счет введения объектов позволяет организовать взаимодействие с пользователем, управлять вебброузером и изменять содержимое документа, отображаемое в пределах окна вебброузера. Эта встроенная версия JavaScript запускает сценарии, внедренные в HTML–код веб–страниц. Как правило, эта версия называется клиентским языком JavaScript, чтобы подчеркнуть, что сценарий исполняется на клиентском компьютере, а не на веб сервере. В основе языка JavaScript и поддерживаемых им типов данных лежат международные стандарты, благодаря чему обеспечивается прекрасная совместимость между реализациями. Некоторые части клиентского JavaScript формально стандартизированы, другие части стали стандартом дефакто, но есть части, которые являются специфическими расширениями конкретной версии броузера. Совместимость реализаций JavaScript в разных броузерах зачастую приносит немало беспокойств программистам, использующим клиентский язык JavaScript [4].

Создание информационной системы для компании, занимающейся разработкой бухгалтерского программного обеспечения, требует подхода, основанного на чётком планировании и учёте как технических деталей, так и

особенностей управленческих процессов целевой организации. В рамках проекта был разработан поэтапный план реализации, направленный на организацию эффективной работы, снижение рисков и обеспечение соответствия финального продукта бизнес-требованиям.

Выбранная методология объединяет принципы традиционного системного проектирования с практиками гибких подходов к разработке программного обеспечения. Это даёт возможность сохранить структурированность и прозрачность процесса с одной стороны, а с другой — быть восприимчивыми к изменениям и оперативно адаптироваться к новым обстоятельствам. Такой интегрированный подход имеет особое значение при создании информационной системы, которая должна быть не только функциональной, но и готовой к масштабированию, с высоким уровнем защищённости. Планируемая система строится на основе клиент-серверной архитектуры, обеспечивающей разделение уровней представления, логики и данных. Такой подход даёт возможность:

- разграничивать ответственность между слоями;
- повышать читаемость и поддерживаемость кода;
- упрощать расширение и модификацию системы в будущем.

Фронтенд будет отвечать за отображение информации и взаимодействие с пользователем. Серверная часть займётся обработкой запросов, проверкой прав доступа и выполнением бизнес-логики. Хранение данных будет осуществляться в реляционной базе данных, структура которой уже была предварительно спроектирована.

На данном этапе также была намечена файловая структура проекта, включающая следующие каталоги:

- templates – шаблоны страниц;
- css – таблицы стилей;
- js – клиентские скрипты;
- images – графические элементы;

– php – серверные обработчики;

– config – конфигурационные файлы;

– ajax – скрипты для динамической загрузки контента.

Были определены ключевые страницы приложения, такие как:

– main.php – главная страница с общим обзором активных задач и проектов;

– reg.php, auth.php – регистрация и авторизация;

– project.php – детальная информация по проекту;

– settings_project.php – настройки проекта;

– logout.php – выход из системы.

Кроме того, предусмотрено использование динамических AJAX-страниц, которые будут загружать данные без перезагрузки всей страницы. Примеры таких страниц:

– content_modal_window_create_project.php – модальное окно для создания нового проекта;

– dynamics_detail_project_right_menu.php – динамическое меню проекта;

– apply_edit_project_observers.php – обработчик изменения списка наблюдателей;

– upload_file.php – загрузка файлов на сервер.

Такая организация позволит сделать интерфейс более отзывчивым и повысит скорость работы с системой.

Важным этапом стало проектирование структуры базы данных, соответствующей бизнес-процессам компании

На этапе проектирования был создан прототип пользовательского интерфейса, который стал основой для дальнейшей верстки. Интерфейс разрабатывался с учётом принципов юзабилити, адаптивности и простоты навигации.

Основные элементы интерфейса:

– горизонтальное меню в верхней части экрана с переходом к основным разделам;
– панель с часто используемыми действиями;
– карточки проектов с возможностью просмотра и редактирования;
– формы ввода данных с поддержкой валидации;
– модальные окна для добавления новых объектов;
– динамические блоки, обновляемые через AJAX.

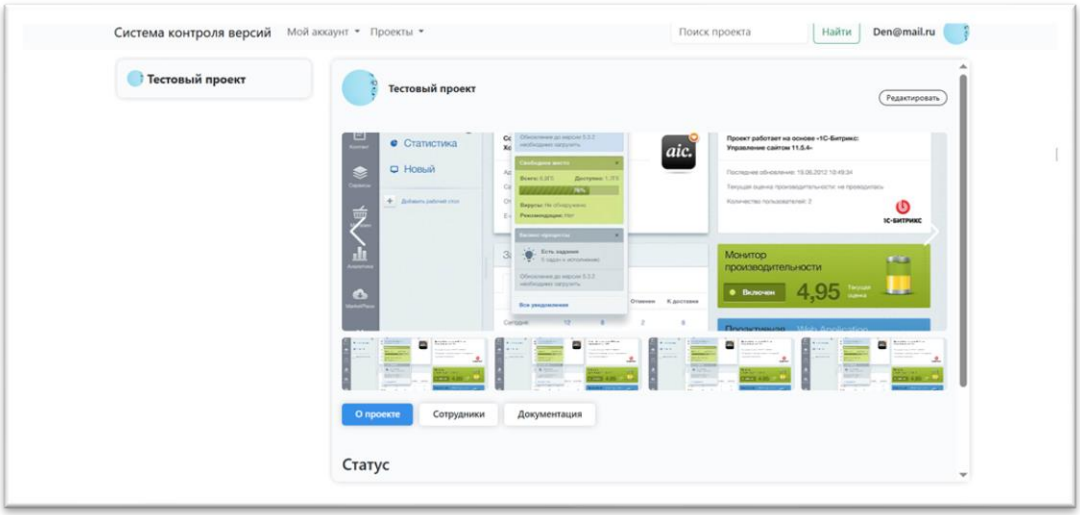


Рисунок 13 – общий вид интерфейса

Ещё на этапе проектирования были определены ключевые меры по обеспечению информационной безопасности:

– защита от SQL-инъекций через использование ORM;
– очистка входящих данных и экранирование выводимого контента;
– двухфакторная проверка данных на стороне клиента и сервера;
– шифрование паролей с помощью password_hash.

Эти меры будут реализованы на этапе разработки, но уже сейчас они являются частью технического задания и учитываются при проектировании.

На основе проделанного проектирования можно выделить следующие этапы предстоящей реализации:

– настройка рабочей среды – установка OpenServer, настройка Apache, PHP, PHPMyAdmin, подготовка файловой структуры;

– верстка интерфейса – реализация HTML-шаблонов, подключение Bootstrap и JavaScript-библиотек;

– интеграция с базой данных – подключение RedBeanPHP, создание моделей, настройка подключения к БД;

– разработка функциональных модулей – реализация модулей управления проектами, задачами, уведомлениями и другими элементами;

– реализация динамических страниц – разработка AJAX-скриптов для подгрузки данных без перезагрузки страницы;

– тестирование системы – проведение юнит-, функционального и нагрузочного тестирования.

Такой пошаговый подход позволяет заранее планировать каждую фазу разработки, учитывать возможные риски и обеспечивать высокое качество конечного продукта.

Таким образом, на этапе проектирования были определены основные направления реализации информационной системы, выбраны ключевые технологии, спроектирована архитектура и структура данных, а также подготовлены макеты пользовательского интерфейса. Был составлен подробный план этапов разработки, учитывающий, как технические, так и организационные аспекты внедрения системы в реальную рабочую среду.

3 ОПИСАНИЕ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1 Подход к реализации

Создание информационной системы для компании, специализирующейся на разработке бухгалтерского программного обеспечения, осуществлялось с использованием современных подходов к проектированию и разработке программных решений. Основной задачей стало формирование устойчивой, масштабируемой и безопасной платформы, которая будет соответствовать как текущим, так и перспективным потребностям организации.

Для достижения поставленных целей был выбран комбинированный подход, объединяющий элементы гибкой методологии (Agile) и традиционного каскадного цикла разработки. Такая стратегия позволила:

– сформировать чёткое представление о системе ещё на начальном этапе;

– последовательно внедрять функциональные модули;

– оперативно получать обратную связь от потенциальных пользователей;

– корректировать функционал без значительного влияния на уже готовые части проекта.

3.2 Верстка и дизайн интерфейса

Одним из важных аспектов разработки информационной системы стало создание удобного, интуитивно понятного и адаптивного пользовательского интерфейса. Интерфейс выступает основным каналом взаимодействия пользователя с системой, поэтому его оформление, навигация и поведение элементов были тщательно спроектированы и реализованы с учётом современных требований к юзабилити и веб-доступности.

Использование Bootstrap оказалось особенно эффективным, поскольку он предоставляет готовые компоненты (например, кнопки, формы, модальные

окна, меню), которые можно быстро внедрять и кастомизировать под нужды проекта. Это позволило значительно ускорить процесс верстки и сосредоточиться на логике работы, а не на деталях стилизации.

Система была спроектирована таким образом, чтобы одинаково хорошо работать как на десктопах, так и на мобильных устройствах. Для этого применялись принципы адаптивного дизайна:

– использование сеточной системы Bootstrap;

– медиазапросы CSS для корректного отображения на разных разрешениях;

– оптимизация размеров элементов под сенсорное управление;

– скрытие или сворачивание сложных блоков при переходе на маленькие экраны.

Благодаря этим мерам интерфейс сохраняет свою функциональность и удобство использования вне зависимости от типа устройства. Это особенно важно для сотрудников, которые могут работать с системой как в офисе, так и удалённо, используя смартфоны или планшеты.

Навигация в системе строилась на основе двухуровневой модели:

Горизонтальное меню вверху – содержит доступ ко всем основным разделам: «Главная», «Проекты», «Задачи», «Пользователи», «Документы», «Профиль».

Боковая панель (sidebar) – содержит часто используемые действия: «Создать проект», «Добавить задачу», «Мои задачи», «Уведомления» и другие.

Каждый элемент навигации был логически связан с соответствующими маршрутами в приложении. Например:

– index.php – главная страница с общим обзором активных задач;

– projects.php – список всех проектов;

– project_view.php?id=123 – карточка конкретного проекта;

– tasks.php – центр управления задачами;

– profile.php – редактирование персональных данных.

Такая организация URL-адресов сделала систему более предсказуемой и упростила её освоение.

Информация на страницах организована в виде структурированных блоков: таблиц, плиток, карточек и модальных окон. Такое построение интерфейса позволяет пользователю легко воспринимать данные и оперативно выполнять необходимые действия.

Для повышения отзывчивости интерфейса и снижения нагрузки на сервер была реализована система частичного обновления контента с помощью AJAX-запросов. Вместо полной перезагрузки страницы обновляются только те блоки, которые затрагиваются текущим действием пользователя.

Например, при добавлении новой задачи через модальное окно:

– пользователь заполняет форму и нажимает «Сохранить»;

– JavaScript отправляет данные на сервер без перезагрузки страницы;

– сервер обрабатывает запрос и возвращает JSON-ответ;

– клиентский скрипт добавляет новую задачу в список и закрывает окно.

Этот подход сделал систему более живой и интерактивной, исключив задержки, связанные с загрузкой всей страницы. Также это положительно сказалось на опыте работы с формами, где ошибки ввода отображаются мгновенно, без необходимости повторной отправки данных.

Модальные окна стали важным элементом интерфейса, поскольку позволяют выполнять действия над данными без покидания текущей страницы. Они используются в таких сценариях, как:

– создание нового проекта;

– редактирование задачи;

– загрузка файла;

– просмотр деталей задачи;

– управление участниками проекта.

Такой способ взаимодействия упрощает навигацию и делает систему более целостной, так как пользователь остаётся в рамках одного контекста, выполняя любые операции.

Формы ввода данных были разработаны с учетом принципов безопасности и удобства использования:

– поля снабжены понятными метками и подсказками;

– обязательные поля выделены визуально;

– при возникновении ошибок выводятся информативные сообщения;

– все данные проверяются как на клиентской, так и на серверной стороне;

– пароли хранятся в зашифрованном виде;

– файлы проверяются на соответствие допустимым типам и размеру.

Формы также поддерживают динамическую подгрузку данных, например, при выборе исполнителя для задачи выпадающий список заполняется через AJAX-запрос, что гарантирует актуальность списка пользователей.

Для улучшения пользовательского опыта были внедрены легкие анимационные эффекты, такие как:

– плавное появление/исчезновение элементов;

– анимация прогресса загрузки файла;

– выделение изменений в списке задач;

– визуальная обратная связь при успешном выполнении операции.

Эти элементы не только делают систему более живой и интерактивной, но и помогают пользователю лучше ориентироваться в происходящем, снижая когнитивную нагрузку.

Цветовая палитра была подобрана с учётом факторов зрительной нагрузки и восприятия информации:

– основные действия выделены яркими цветами (синий, зелёный);

– предупреждающие действия (удаление, отмена) окрашены в красные тона;

– фоновые элементы имеют светлую или нейтральную расцветку;

Шрифты подбирались с акцентом на читаемость и универсальность. Было решено использовать sans-serif начертания, которые лучше воспринимаются на экранах.

Верстка проводилась с учетом стандартов доступности (WCAG), чтобы система могла быть использована даже людьми с ограниченными возможностями. Для этого были реализованы следующие меры:

– обеспечение контрастности между текстом и фоном;

– использование ARIA-атрибутов для скринридеров;

– возможность работы с клавиатуры;

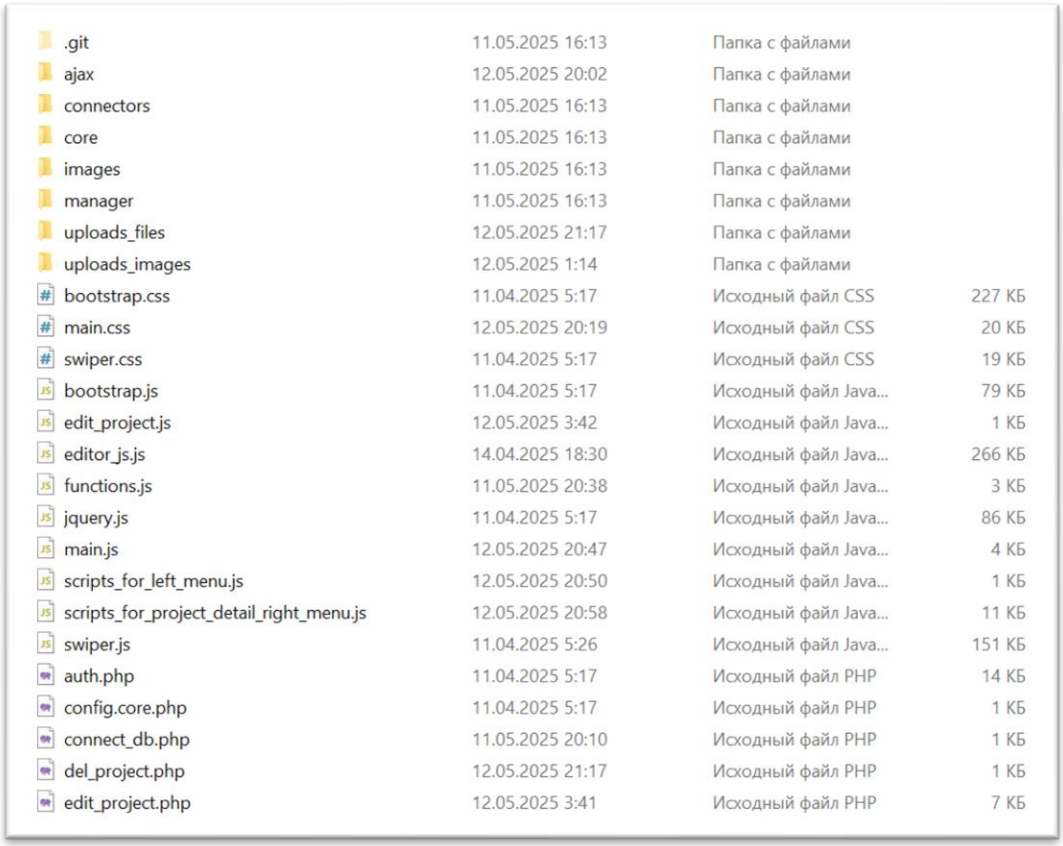
– масштабируемость интерфейса;

– наличие всплывающих подсказок (tooltips) для непонятных элементов.

Эти подходы обеспечивают равный доступ к функционалу системы и делают её более универсальной в использовании.

3.3 Создание структуры сайта

Для реализации навигационной системы использовался маршрутизатор (Vue Router / React Router), который позволил организовать переходы между разделами без перезагрузки страницы. Была определена иерархия маршрутов:



.git	11.05.2025 16:13	Папка с файлами	
ajax	12.05.2025 20:02	Папка с файлами	
connectors	11.05.2025 16:13	Папка с файлами	
core	11.05.2025 16:13	Папка с файлами	
images	11.05.2025 16:13	Папка с файлами	
manager	11.05.2025 16:13	Папка с файлами	
uploads_files	12.05.2025 21:17	Папка с файлами	
uploads_images	12.05.2025 1:14	Папка с файлами	
bootstrap.css	11.04.2025 5:17	Исходный файл CSS	227 КБ
main.css	12.05.2025 20:19	Исходный файл CSS	20 КБ
swiper.css	11.04.2025 5:17	Исходный файл CSS	19 КБ
bootstrap.js	11.04.2025 5:17	Исходный файл Java...	79 КБ
edit_project.js	12.05.2025 3:42	Исходный файл Java...	1 КБ
editor_js.js	14.04.2025 18:30	Исходный файл Java...	266 КБ
functions.js	11.05.2025 20:38	Исходный файл Java...	3 КБ
jquery.js	11.04.2025 5:17	Исходный файл Java...	86 КБ
main.js	12.05.2025 20:47	Исходный файл Java...	4 КБ
scripts_for_left_menu.js	12.05.2025 20:50	Исходный файл Java...	1 КБ
scripts_for_project_detail_right_menu.js	12.05.2025 20:58	Исходный файл Java...	11 КБ
swiper.js	11.04.2025 5:26	Исходный файл Java...	151 КБ
auth.php	11.04.2025 5:17	Исходный файл PHP	14 КБ
config.core.php	11.04.2025 5:17	Исходный файл PHP	1 КБ
connect_db.php	11.05.2025 20:10	Исходный файл PHP	1 КБ
del_project.php	12.05.2025 21:17	Исходный файл PHP	1 КБ
edit_project.php	12.05.2025 3:41	Исходный файл PHP	7 КБ

Рисунок 14 – Структура компонента формы

Каждый маршрут привязан к соответствующему компоненту, что упрощает поддержку и расширяемость. Для часто используемых действий (например, создание проекта) реализованы динамические ссылки, генерируемые на основе прав доступа пользователя.

Для работы с формами применялись библиотеки управления состоянием (Vuelidate / Formik), что позволило централизовать валидацию данных. Каждое поле формы связано с моделью данных, где заданы правила валидации (например, обязательное поле, формат даты). Ошибки ввода отображаются в реальном времени с помощью UI-компонентов (<ValidationMessage>).

Для повышения производительности реализована отложенная валидация: проверка данных запускается только при потере фокуса с поля или попытке отправки формы. Это снижает нагрузку на клиентский JS-движок.

```php
try {
    // Получаем данные из POST-запроса
    $login = $_POST['login'] ?? '';
    $email = $_POST['email'] ?? '';
    $password = $_POST['password'] ?? '';
    $password_second = $_POST['password_second'] ?? '';
    // Проверяем, что все поля заполнены
    if (empty($login) || empty($email) || empty($password) || empty($password_second)) {
        echo json_encode(['success' => false, 'message' => 'Все поля обязательны для заполнения.']);
        exit;
    }
    // Проверяем длину логина (например, минимум 3 символа)
    if (strlen($login) < 3) {
        echo json_encode(['success' => false, 'message' => 'Логин должен содержать минимум 3 символа.']);
        exit;
    }
    // Проверяем, что логин содержит только буквы, цифры и нижнее подчеркивание
    if (!preg_match('/^[a-zA-Z0-9_]+$/', $login)) {
        echo json_encode(['success' => false, 'message' => 'Логин может содержать только буквы, цифры и нижнее подчеркивание.']);
        exit;
    }
    // Проверяем валидность email
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo json_encode(['success' => false, 'message' => 'Некорректный email.']);
        exit;
    }
    // Проверяем длину пароля (например, от 8 до 50 символов)
    if (strlen($password) < 8 || strlen($password) > 50) {
        echo json_encode(['success' => false, 'message' => 'Пароль должен быть от 8 до 50 символов.']);
        exit;
    }
```

Рисунок 15 – Структура компонента формы

Для представления информации использовались подходы, ориентированные на производительность:

– ленивая загрузка данных (lazy loading): плитки проектов загружаются порциями по 10 элементов, что сокращает время первого рендера;

– автоматизация сжатия загружаемых изображений.

Структура сайта была реализована с использованием компонентного подхода. Основные модули:

– header – шапка с навигацией;

– projectGrid – список проектов;

– taskForm – форма задачи;

– modal – модальное окно для действий.

Каждый компонент изолирован, имеет свою логику и стили, что упрощает тестирование и переиспользование. Для стилизации применялись CSS-модули, исключающие конфликты классов.

Для сборки сайта использовался Webpack, который:

– объединял JavaScript-файлы в чанки;

– оптимизировал изображения с помощью image-webpack-loader;

– внедрял хэши в названия файлов для борьбы с кэшированием.

Деплой осуществлялся через GitHub Pages с автоматизацией через GitHub Actions. Для повышения скорости загрузки страницы подключались шрифты через font-display: swap, а критические CSS-стили встраивались в HTML.

3.4 Реализация пользовательских форм

Одним из ключевых этапов создания информационной системы стало проектирование и реализация пользовательских форм – основного инструмента взаимодействия между сотрудниками и системой. Формы обеспечивают выполнение базовых операций: регистрация и авторизация, создание и редактирование проектов, и другие. Их разработка велась с учётом требований к безопасности, удобству использования и соответствию современным стандартам веб-интерфейсов.

При разработке форм использовались принципы юзабилити и доступности, что позволило сделать интерфейс понятным и удобным даже для новых пользователей. Каждая форма была спроектирована таким образом, чтобы:

– минимизировать количество действий для выполнения задачи;

– обеспечить логическую последовательность заполнения полей;

– предоставлять понятные подсказки и пояснения;

– давать обратную связь при возникновении ошибок;

– корректно отображаться как на десктопных устройствах, так и на мобильных.

Формы были условно разделены на две группы: формы аутентификации (регистрация, вход) и рабочие формы, предназначенные для управления проектами, задачами и другими элементами системы. При этом общий стиль

оформления и поведение форм оставались едиными, что способствовало целостности интерфейса.

С использованием классов Bootstrap, таких как form-control, form-label, btn, card, была достигнута визуальная согласованность всех форм. Это также упростило дальнейшее сопровождение и модификацию интерфейса. В ряде случаев формы открывались в модальных окнах, что позволило избежать лишних переходов между страницами и сделало работу с системой более плавной.

Форма регистрации предоставляет возможность новым пользователям создавать учётную запись в системе. Она включает поля для ввода имени, электронной почты, пароля и его повторного ввода. Проверка данных осуществляется как на стороне клиента (через JavaScript), так и на сервере (в PHP-обработчике). Пароль перед сохранением хэшируется функцией password_hash(), что обеспечивает защиту конфиденциальной информации.

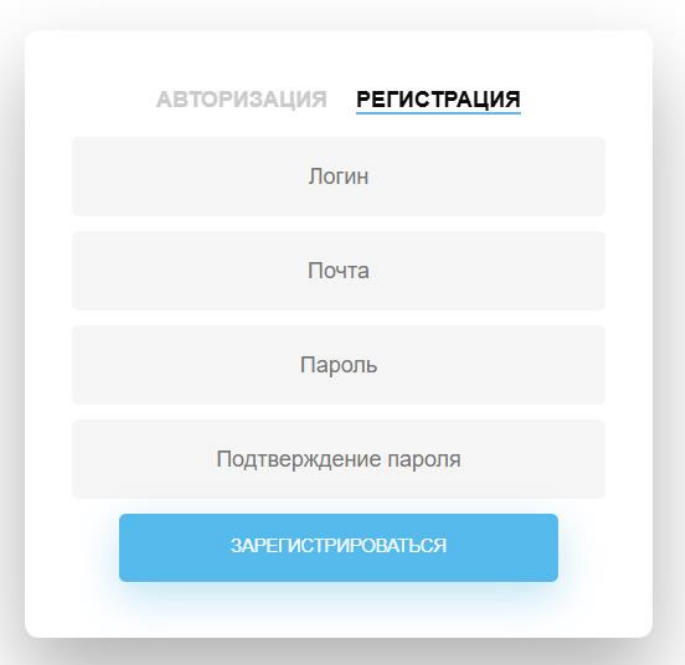


Рисунок 16 – Форма регистрации

Форма входа позволяет зарегистрированным пользователям получить доступ к системе через указание email и пароля. При успешной проверке данных создаются сессионные переменные, и пользователь перенаправляется

на главную страницу. В случае неверного ввода или блокировки аккаунта выводится понятное сообщение об ошибке, позволяющее быстро исправить ситуацию.

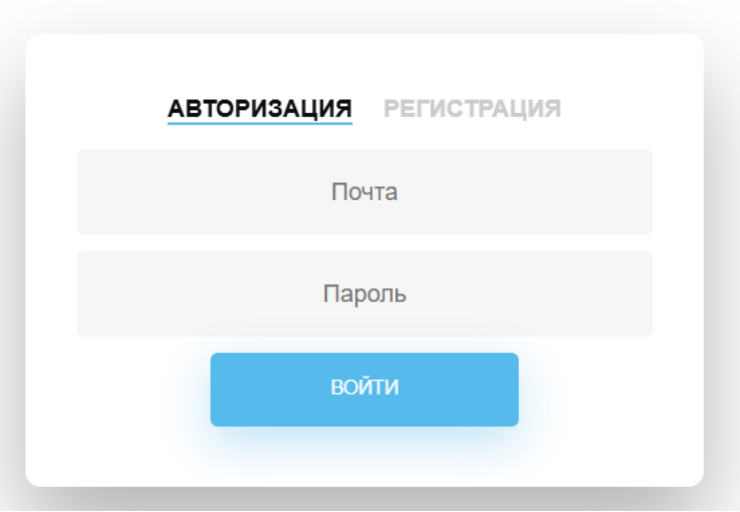


Рисунок 17 – Форма авторизации

Создание проекта

Форма добавления нового проекта содержит такие поля, как:

– название;

– логотип проекта.

Выбор сотрудников осуществляется через выпадающий список, данные для которого загружаются динамически с помощью AJAX-запроса. Это гарантирует актуальность списка и упрощает процесс назначения команды.

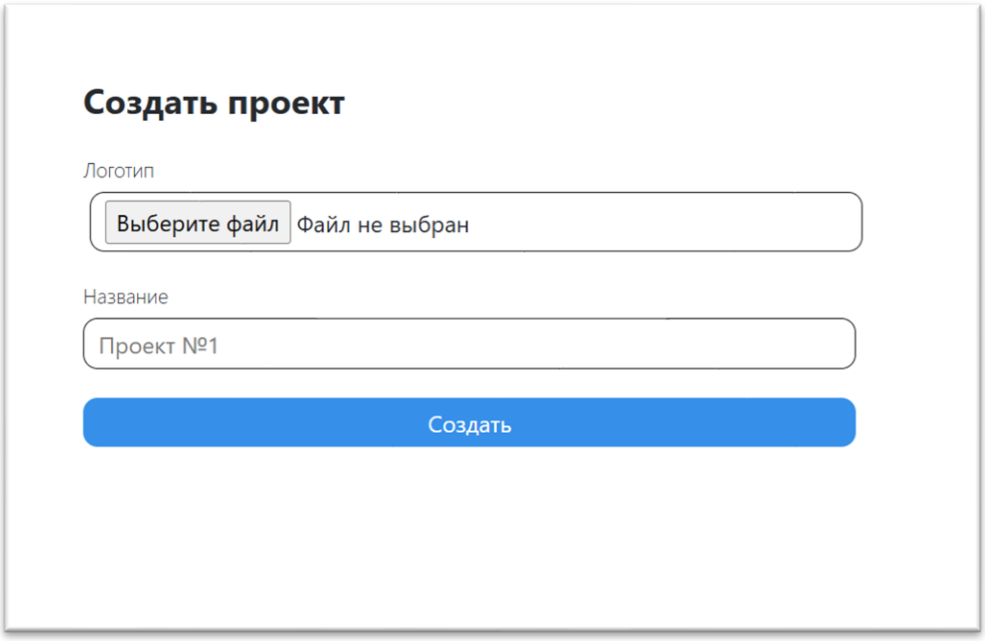


Рисунок 18 – Форма создания проекта

Форма редактирования представляет собой аналог формы создания, но предварительно заполняется данными из базы. Такой подход позволяет менять параметры проекта без необходимости повторного ввода всей информации, что значительно ускоряет рабочий процесс.

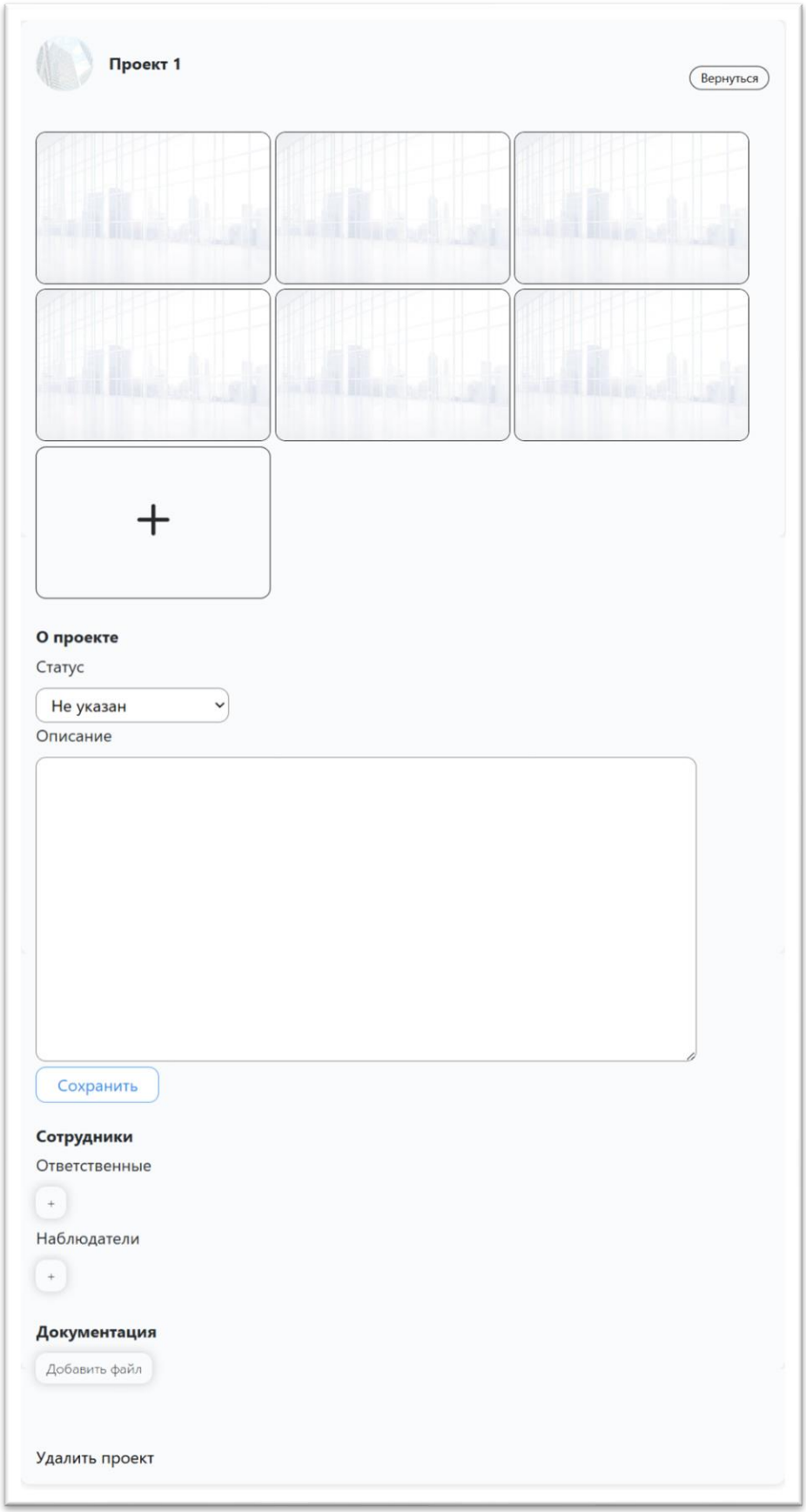


Рисунок 19 – Форма редактирования проекта

Форма создания задачи включает:

– заголовок;
– описание;
– уровень приоритета (низкий, средний, высокий);
– исполнителя (выбор из списка пользователей);
– срок выполнения;
– статус задачи (по умолчанию – «в очереди»).

Использование календаря для установки даты и выпадающих списков для выбора статуса, и исполнителя делает работу с формой удобной и интуитивно понятной. Также предусмотрена возможность изменения статуса задачи прямо на карточке, без необходимости открытия формы редактирования.

Настройка профиля пользователя. Форма редактирования профиля включает поля для изменения:

– имени и фамилии;
– должности;
– даты рождения;
– контактной информации;
– аватара.

Загрузка аватара реализована через поле типа photo. После загрузки файл сохраняется в директорию /assets/images/avatars/ с уникальным именем, чтобы избежать перезаписи. Изображение сразу отображается в профиле, а данные о нём сохраняются в базе данных.

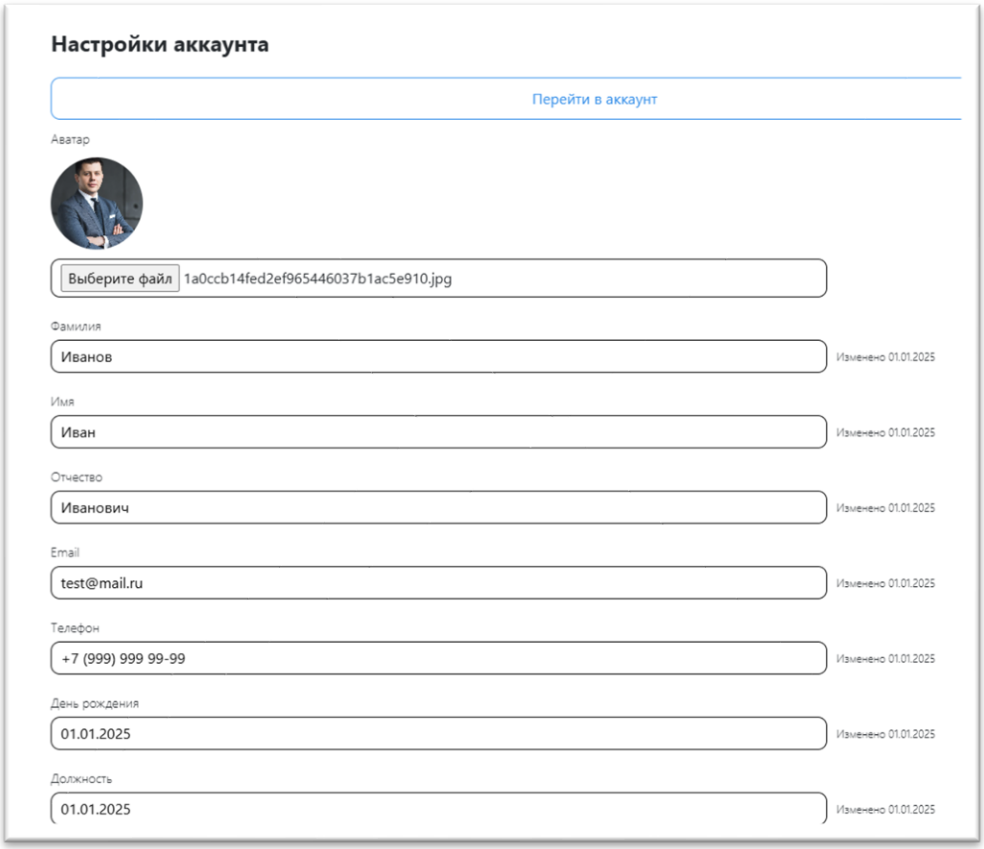


Рисунок 20 – Форма редактирования профиля

Обеспечение корректности вводимых данных являлось важной частью реализации. Проверка проводилась на двух уровнях:

– клиентский уровень – с использованием JavaScript и jQuery. Здесь проверялись обязательные поля, формат email, совпадение паролей, минимальная длина и другие параметры;

– серверный уровень – реализован с помощью PHP. Все данные дополнительно очищались и проверялись на соответствие формату. Например, email проверялся через filter_var(), а пароль – на сложность и длину.

Такой многоуровневый подход позволил исключить попадание некорректных данных в систему и повысить её надёжность.

Безопасность форм стала одним из центральных аспектов их реализации. Были внедрены следующие меры:

– очистка данных – использование функций htmlspecialchars() и trim() для защиты от XSS-атак;

– проверка email – с использованием встроенного фильтра FILTER_VALIDATE_EMAIL;

– хранение паролей – с применением безопасного алгоритма password_hash();

– защита от CSRF-атак – реализация токенов, препятствующих выполнению запросов от лица пользователя без его ведома;

– ограничение попыток входа – система блокировала учетную запись после нескольких неудачных попыток авторизации, что снижало риск подбора пароля.

Эти меры обеспечили высокий уровень защищённости и доверие со стороны пользователей.

Для улучшения пользовательского опыта были реализованы модальные окна, в которых открывались формы добавления и редактирования. Это позволило избежать лишних переходов между страницами и сделает работу с системой более плавной.

Кроме того, для некоторых форм (например, добавление задачи или комментария) был внедрен механизм частичного обновления контента с помощью AJAX. После отправки данных информация отображается мгновенно, без необходимости перезагрузки страницы. Такой подход заметно повысил скорость работы и улучшил восприятие интерфейса [7].

Учитывая, что система строится на основе CMS ModX, формы были интегрированы в существующую архитектуру платформы. Для этого использовались:

– сниппеты – для получения данных из формы и выполнения бизнес-логики;

– плагины – для дополнительной обработки событий;

– TV-параметры – для передачи значений в шаблоны.

Такое построение позволило гибко управлять формами и легко интегрировать их в уже созданную структуру сайта.

После завершения разработки все формы были протестированы на соответствие требованиям. Основные направления тестирования включали:

- корректность ввода и отображения данных;
- работу валидации;
- защиту от некорректных и злонамеренных запросов;
- отправку данных на сервер и обработку;
- отображение результатов действия.

В ходе тестирования были выявлены и исправлены возможные проблемы:

- ошибки валидации;
- сбои в отображении сообщений;
- трудности с обработкой загружаемых файлов;
- потенциальные уязвимости при передаче данных.

После доработки формы стали работать стабильно и соответствовали всем заявленным техническим и функциональным требованиям.

Таким образом, разработка пользовательских форм сыграла ключевую роль в организации взаимодействия с системой. Применение современных технологий и соблюдение принципов юзабилити позволило создать удобный и интуитивно понятный интерфейс.

Формы обеспечивают:

- точный ввод данных;
- защиту информации от внешних угроз;
- корректную обработку и отображение информации;
- отзывчивость и адаптивность на различных устройствах.

Все формы были протестированы и успешно прошли проверку на функциональность, безопасность и удобство использования. Они стали важной частью интерфейса, обеспечивая прозрачность и простоту взаимодействия с системой.

3.5 Интеграция RedBeanPHP

Одним из ключевых аспектов реализации информационной системы стало обеспечение эффективного взаимодействия с базой данных. Для упрощения работы с реляционной моделью данных была внедрена объектно-реляционная модель (ORM) RedBeanPHP, которая позволила сократить объём SQL-запросов, повысить читаемость кода и уменьшить вероятность ошибок при обработке информации.

При разработке систем, ориентированных на управление проектами и задачами, особенно важна простота и надёжность работы с данными. Вместо использования нативных SQL-запросов, которые могут быть трудозатратными и склонными к ошибкам, было принято решение использовать ORM. Выбор пал на RedBeanPHP по следующим причинам:

– простота установки и интеграции – библиотека легко подключается и не требует сложной настройки;

– лёгкость и минимальные требования – RedBeanPHP является легковесным решением, подходящим как для прототипирования, так и для полноценных проектов;

– поддержка ООП – работа с данными в виде объектов делает логику приложения более понятной и структурированной;

– автоматическое создание таблиц – при обращении к несуществующей модели ORM сама создаёт необходимые таблицы и поля, что ускоряет разработку;

– удобство работы с отношениями – поддерживаются связи один-ко-многим и многие-ко-многим без необходимости ручного написания JOIN-запросов;

– открытый исходный код и сообщество – наличие активного сообщества и документации обеспечивает доступность решения и возможность его гибкой настройки.

Эти особенности делают RedBeanPHP удобным инструментом для создания информационной системы, особенно на этапах проектирования и тестирования, когда важна скорость реализации и простота модификации.

Интеграция ORM в проект происходила поэтапно, с соблюдением принципов устойчивости и масштабируемости.

Библиотека была загружена с официального сайта и добавлена в структуру проекта. Файл rb.php был помещён в директорию /core/, а его подключение осуществлялось в конфигурационном файле, чтобы обеспечить доступ к функционалу ORM во всех частях приложения. Это дало возможность работать с данными в объектно-ориентированном стиле, не используя сложные SQL-конструкции.

На основе заранее спроектированной ER-диаграммы были созданы соответствующие классы, отражающие ключевые сущности системы:

- user – пользователь системы;
- project – проект;
- document – связанные файлы.

Каждая из этих сущностей имеет свои атрибуты, напрямую соответствующие столбцам в таблицах базы данных. Например, экземпляр класса Task содержит такие свойства, как title, description, status, executor_id, project_id, что позволяет легко манипулировать данными в привычном формате.

С помощью RedBeanPHP реализация операций создания, чтения, обновления и удаления данных стала значительно проще. Все действия выполняются через вызовы методов, что упрощает разработку и делает код более читаемым. Это позволило сосредоточиться на бизнес-логике, минимизируя работу с техническими деталями SQL.

Особенностью RedBeanPHP является способность автоматически создавать таблицы и поля при первом обращении к ещё не определённым сущностям. Например, если вызвать R::dispense('document'), а таблица

document отсутствует, ORM сама создаст её с необходимыми полями, основываясь на типах переданных данных. Это позволило избежать необходимости ручного написания DDL-скриптов и ускорило процесс разработки.

ORM обеспечивает защиту от SQL-инъекций за счёт использования подготовленных запросов. Это исключает возможность внедрения вредоносного кода через параметры, полученные из форм. Дополнительно все входящие данные очищались и проверялись на уровне PHP с использованием таких функций, как htmlspecialchars(), trim() и filter_input(). Это усилило уровень защиты всей системы и снизило риск возникновения уязвимостей.

Хотя использование ORM добавляет уровень абстракции, влияющий на производительность, RedBeanPHP показала себя как достаточно быстрое решение для текущего масштаба системы. Для повышения скорости работы с данными применялись следующие подходы:

– кэширование часто используемых данных – снижение нагрузки на СУБД;

– использование индексов – ускорение выборки данных;

– фильтрация и ограничение запросов – оптимизация объема извлекаемых данных.

Поскольку система строилась на основе CMS ModX, ORM была корректно интегрирована в существующую архитектуру платформы. Для этого были созданы сниппеты, которые вызывают методы RedBeanPHP и возвращают результат в шаблон. Такой подход сохранил гибкость системы и позволил совмещать преимущества CMS и ORM.

Однако, как и у любого решения, у ORM есть и некоторые ограничения:

– необходимость строгого соответствия имён сущностей и таблиц;

– ограниченная поддержка сложных запросов – в некоторых случаях приходилось использовать нативный SQL;

– контроль автоматического создания таблиц, чтобы избежать случайного изменения структуры БД.

Выбор RedBeanPHP оказался обоснованным в условиях ограниченного времени и необходимости быстрого прототипирования. Полученная система показала хорошие результаты по производительности и стабильности, что делает её готовой к дальнейшему развитию и расширению функционала.

4 ТЕСТИРОВАНИЕ

4.1 Подход к тестированию системы

Тестирование является неотъемлемой частью разработки любого программного обеспечения, поскольку позволяет выявить ошибки, оценить производительность, проверить удобство использования и убедиться в надёжности системы. При создании информационной системы для компании, специализирующейся на разработке бухгалтерского программного обеспечения, был применён комплексный подход к тестированию, включающий модульное, функциональное, нагрузочное и тестирование безопасности. Такая стратегия позволила всесторонне оценить качество разработанного решения и подготовить его к дальнейшему внедрению.

Основными задачами этапа тестирования стали:

– проверка корректности работы всех компонентов системы;

– выявление и устранение технических и логических ошибок;

– оценка производительности при различных сценариях использования;

– обеспечение соответствия интерфейса принципам юзабилити;

– проверка уровня защищенности данных и механизмов доступа.

Эти цели определили выбор методов и видов тестирования, направленных как на техническую надёжность, так и на пользовательский опыт.

Модульное тестирование проводилось с целью проверки отдельных элементов системы на соответствие заданному поведению. Для этого использовались простые PHP-скрипты, вызывающие отдельные функции и проверяющие их реакцию на различные входные данные. Тестировались такие операции, как:

– создание пользователя;

– добавление проекта;

– назначение исполнителя;
– обновление статуса задачи;
– отправка уведомлений.

```php
<?
$task = R::dispense('_tasks');
$task->title = 'Новая задача';
$task->executor_id = 5;
$id = R::store($task);
assert(R::load('task', $id) !== null);
?>
```

Рисунок 21 – Пример кода для проверки создания задачи

Такие тесты помогли заранее выявить и устранить потенциальные проблемы на уровне отдельных модулей, прежде чем они могли повлиять на работу всей системы.

Функциональное тестирование направлено на проверку выполнения системой своих ключевых задач. Были протестированы следующие сценарии:

– регистрация и авторизация;
– создание, редактирование и удаление проектов;
– работа с задачами (добавление, изменение, фильтрация);
– взаимодействие нескольких участников в рамках одного проекта;
– отправка и получение уведомлений;
– загрузка, хранение и просмотр документов.

Учитывая, что система работает с чувствительными данными, особое внимание было уделено тестированию на безопасность. Проверялись следующие аспекты:

– защита от XSS-атак – все данные, выводимые в интерфейс, прошли очистку через htmlspecialchars() и другие механизмы защиты;

– предотвращение SQL-инъекций – благодаря использованию ORM RedBeanPHP все запросы выполнялись через подготовленные выражения;

– контроль прав доступа – проверялась корректная работа ролевой модели, чтобы пользователи не могли выполнять действия вне своей компетенции;

– валидация форм – поля, принимающие пароли, email и даты, тщательно проверялись на стороне клиента и сервера;

– безопасность сессий – тестировалось время жизни сессии, защита от перехвата и других угроз.

В результате тестирования были выявлены и устранены несколько потенциальных уязвимостей, таких как утечка сессионных данных и возможность множественной отправки формы. Это повысило уровень защищённости информации и доверие со стороны будущих пользователей.

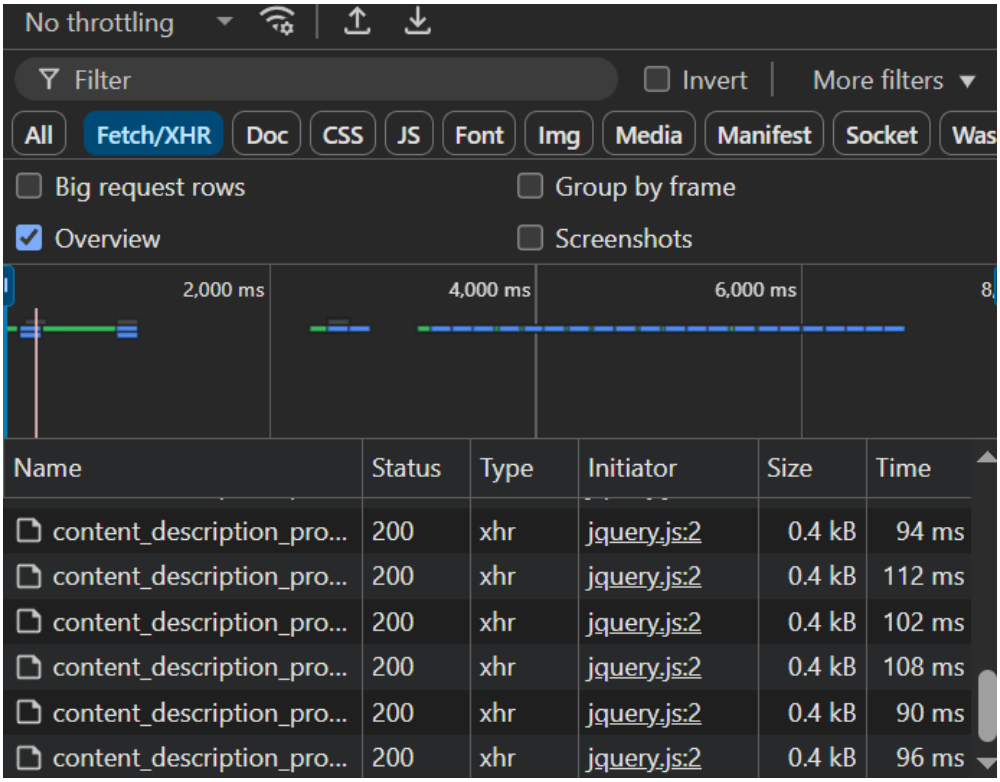


Рисунок 22 – Ошибка в виде множества отправляющихся форм

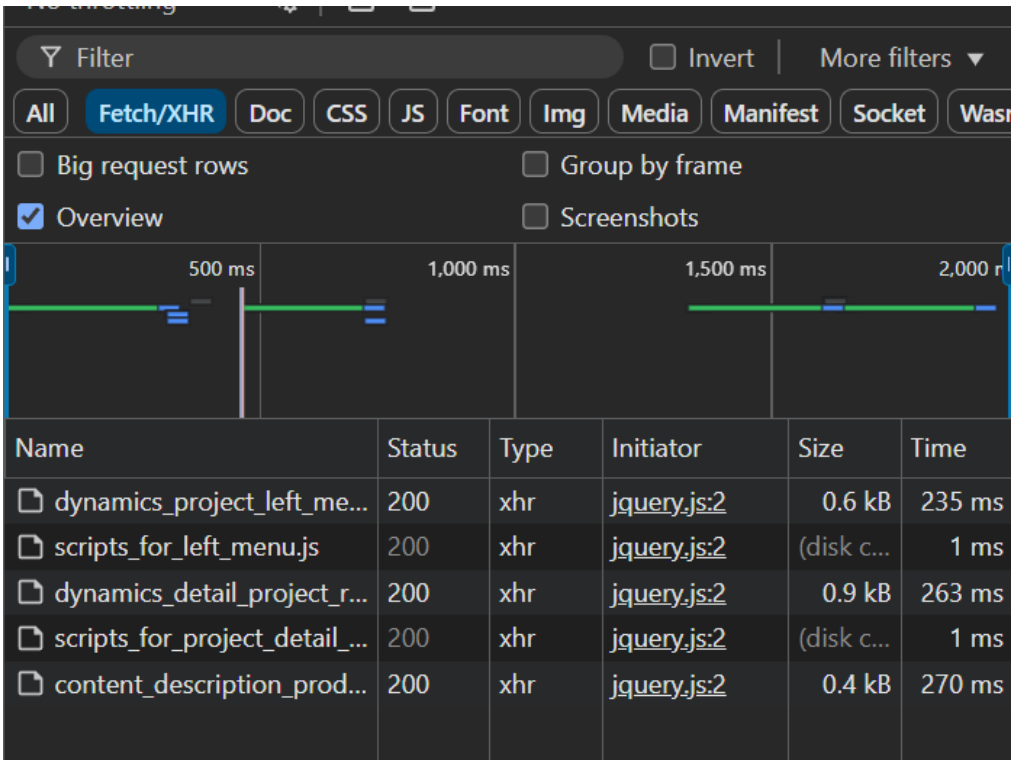


Рисунок 23 – Ошибка в виде множества отправляющихся форм после исправления

Нагрузочное тестирование проводилось с целью оценки производительности системы при увеличении объёма данных и количества одновременных пользователей. Для имитации нагрузки использовались такие инструменты, как Apache Benchmark и Postman, которые позволяют генерировать одновременные запросы и анализировать отклик сервера.

Проверялись следующие параметры:

– время ответа при 20, 40 и 50 одновременных запросах;

– загрузка процессора и использование памяти;

– скорость выполнения CRUD-операций;

– время отображения страниц с большим объёмом данных.

Результаты показали следующие показатели:

20 одновременных запросов – 0.21 секунд;

40 одновременных запросов – 0.3 секунд;

50 одновременных запросов – 0.34 секунд.

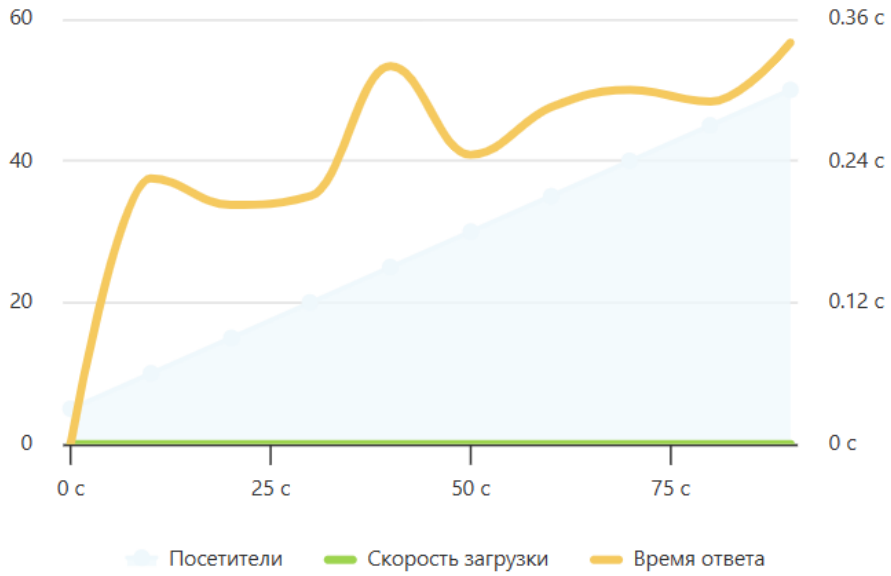


Рисунок 24 – Результаты нагрузочного тестирования

Система показала хорошую устойчивость к нагрузке, что говорит о возможности масштабирования на большее количество пользователей. При необходимости можно будет перенести систему на облачную платформу или использовать распределённое хранение данных.

После завершения всех этапов тестирования была проведена оценка эффективности системы – то есть того, насколько она соответствует заявленным целям и какие преимущества может принести целевой организации.

Основные метрики эффективности:

– снижение времени на управление задачами – сравнение временных затрат на планирование и контроль задач до и после внедрения системы;

– уменьшение количества ошибок – оценка снижения числа просроченных задач и неточностей в распределении обязанностей;

– удобство интерфейса – оценка пользователей по шкале от 1 до 5;

– скорость выполнения операций – замеры времени отклика системы при выполнении основных действий.

Автоматизация управления задачами позволила минимизировать риск человеческой ошибки. Благодаря наглядному представлению задач и проектов стало возможным быстрое выявление проблем и их оперативное устранение. Также была внедрена система уведомлений, которая информирует пользователей о критических событиях – истечении дедлайнов, изменениях в задачах и других важных моментах.

Система полностью соответствует современным требованиям к интерфейсу и может быть внедрена без длительного обучения сотрудников.

Все базовые операции, такие как создание задачи, переход между проектами, обновление данных, выполняются менее чем за одну секунду. Это говорит о высокой отзывчивости системы и её готовности к регулярному использованию в условиях бизнеса.

Следующим этапом тестирования информационной системы стал анализ её функциональности. Оценка проводилась на основе тестирования и практического использования в условиях, приближенных к реальным.

Рассмотрим работу программы с точки зрения двух типов пользователей: постановщика задачи и исполнителя.

– с точки зрения создателя проекта – после авторизации менеджер может создать новый проект, указать сроки, описание и назначить исполнителей. Задачи добавляются по мере необходимости, с указанием приоритета и дедлайна. Интерфейс позволяет фильтровать задачи и отслеживать прогресс проекта в режиме реального времени. Пример представлен на рисунке 25.

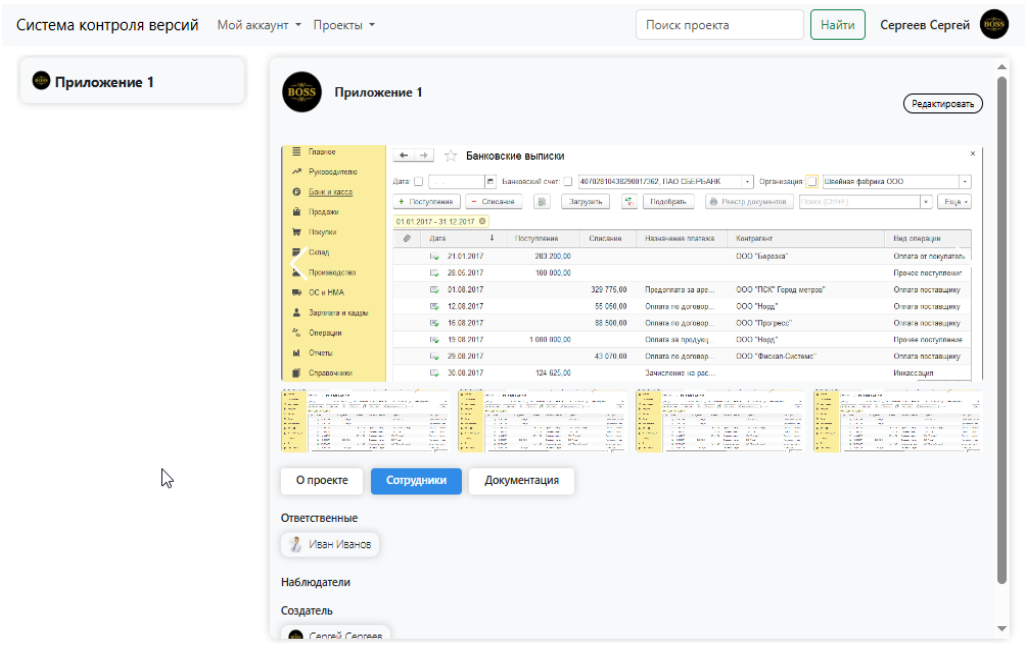


Рисунок 25 - Пример экрана с точки зрения постановщика задачи

– с точки зрения исполнителя – после назначения задачи, исполнитель видит задание в своем личном кабинете. Он может менять статус задачи, прикреплять файлы и следить за обновлениями. Это делает процесс прозрачным и упрощает контроль над поручениями. Пример на рисунке 26.

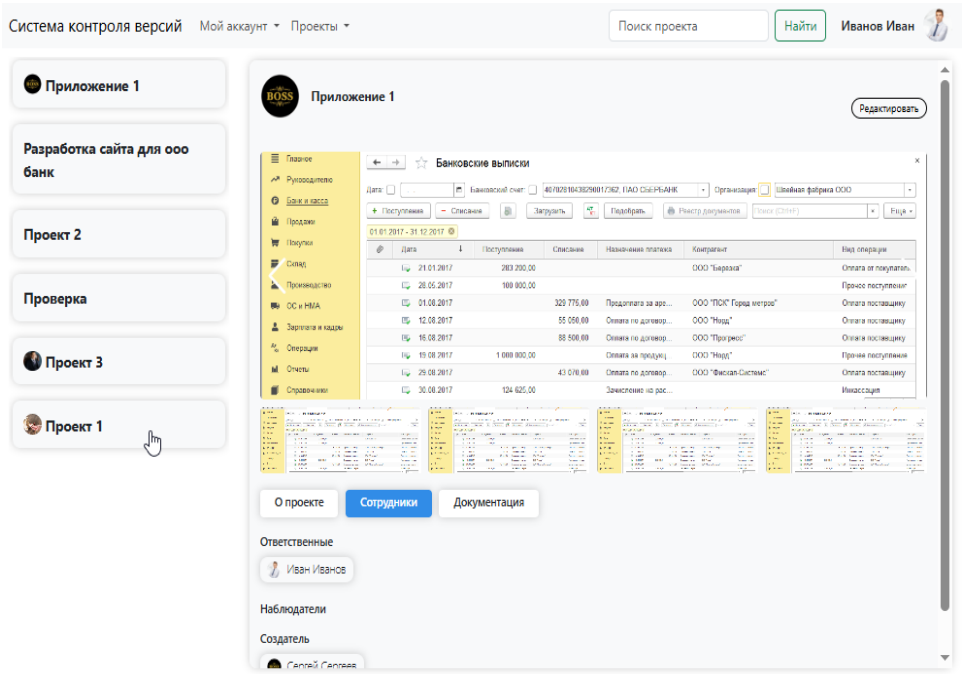


Рисунок 26 - Пример экрана с точки зрения исполнителя

Таким образом, тестирование системы показало её высокую работоспособность, надёжность и безопасность. Все выявленные ошибки были своевременно исправлены, а пользовательский опыт улучшен. Система продемонстрировала отличные показатели производительности, соответствует современным стандартам безопасности и удобства использования.

Полученные результаты говорят о том, что разработанное решение готово к полноценному внедрению и способно стать эффективным инструментом управления проектами в компании, занимающейся разработкой бухгалтерского программного обеспечения.

4.2 Анализ полученных результатов

Финальная стадия создания информационной системы — анализ полученных результатов — является ключевым этапом, позволяющим оценить её эффективность, функциональное соответствие требованиям и степень достижения поставленных целей. В рамках проведённого исследования была выполнена комплексная оценка разработанного программного продукта на основе данных, собранных как при тестировании, так и при использовании в условиях, приближенных к практическим. При этом особое внимание уделялось следующим параметрам: функциональность, производительность, удобство интерфейса, уровень информационной безопасности и потенциал масштабируемости и развития.

Созданная система полностью реализует необходимый набор функций для эффективного управления проектами в организации, специализирующейся на разработке бухгалтерского программного обеспечения:

– возможность создания и редактирования проектов с распределением ответственности и контролем сроков;

– управление задачами — от их добавления до изменения статуса выполнения;

– гибкая система ролей, обеспечивающая контроль доступа;

– поддержка обмена документами — загрузка, хранение и привязка файлов к соответствующим задачам и проектам;

– адаптивный пользовательский интерфейс, поддерживающий корректное отображение на различных устройствах.

Все модули системы взаимодействуют между собой согласованно, что позволяет использовать платформу как единое информационное пространство. При этом были учтены специфические требования компании, связанной с высокой точностью и строгим контролем за исполнением задач. Интеграция с текущими бизнес-процессами позволила достичь высокой степени адаптации решения под конкретные потребности предприятия.

Особую ценность представляет интуитивно понятный интерфейс, который стал одним из главных преимуществ системы. Пользователи могут освоить базовые функции без необходимости проходить длительное обучение или изучать сложные руководства. Это стало возможным благодаря продуманному UX/UI дизайну, акценту на минималистичном подходе и логичной навигации. Такой подход снижает порог входа для новых сотрудников и способствует более высокому уровню вовлечённости пользователей.

Не менее важным фактором, влияющим на выбор данной системы, является её бесплатная доступность. В отличие от коммерческих аналогов, таких как Jira, Asana или Trello, внедрение которых связано с оплатой подписки или лицензий, предложенное решение может быть развернуто без затрат на приобретение программного обеспечения. Это делает его особенно привлекательным для малых и средних предприятий, которые стремятся к цифровизации управленческих процессов с минимальными финансовыми вложениями.

Для наглядности ниже представлена таблица 17, сравнивающая разработанную систему с популярными аналогами.

Таблица 17 – Сравнение системы с аналогами

Критерий	Разработанная система	Jira	Trello	Asana
Интуитивный интерфейс	Присутствует	Отсутствует	Присутствует	Присутствует
Бесплатность	Да	Нет	Частично	Нет
Гибкая настройка под бизнес-процессы	Присутствует	Присутствует	Отсутствует	Присутствует
Система ролей и прав	Присутствует	Присутствует	Отсутствует	Присутствует
Аналитика и отчетность	Присутствует	Присутствует	Отсутствует	Присутствует

Как видно из таблицы, разработанная система выгодно выделяется сочетанием интуитивного интерфейса и бесплатности, что делает её уникальным предложением на рынке решений для управления проектами в нише бухгалтерского ПО.

Таким образом, все поставленные задачи успешно решены, а выбранные технологии и подходы к разработке оказались корректными и соответствующими условиям реализации проекта. Анализ показал, что система обладает высокой степенью функциональности, надёжности и масштабируемости. Она готова к внедрению в штатную работу компании и способна стать эффективным инструментом управления проектами, направленным на повышение прозрачности, снижение административной нагрузки и автоматизацию ключевых процессов.

5 БЕЗОПАСНОСТЬ И ЭКОЛОГИЧНОСТЬ

5.1 Безопасность

Безопасность при разработке информационной системы для ООО «Эмвиджер» в первую очередь направлена на защиту персональных данных хранящихся в базе данных.

Безопасность информационных систем, содержащих персональные данные, критически важная задача для компаний, стремящихся защитить клиентов, сотрудников и репутацию. Персональные данные, такие как имена, адреса, номера телефонов, паспортные данные или банковские карты, являются мишенью для злоумышленников. Фишинг – одна из самых распространённых угроз, но не менее опасны вредоносное ПО, социальная инженерия и другие методы атак.

Фишинг – это мошенническая схема, направленная на кражу конфиденциальных данных (логины, пароли, банковские данные) через поддельные письма, сообщения или сайты, имитирующие официальные ресурсы. Например, сотрудник может получить письмо, якобы от IT–отдела, с просьбой перейти по ссылке для смены пароля, ведущей на фальшивый сайт.

Фишинг опасен, так как эксплуатирует человеческий фактор, обходя даже надёжные системы. Последствия включают утечку данных, финансовые потери, репутационный ущерб и юридические санкции за нарушение законов, таких как ФЗ–152.

Помимо фишинга, существуют другие методы, угрожающие безопасности информационных систем. Они также часто используют человеческий фактор или технические уязвимости.

Вредоносное ПО – это программы, предназначенные для нанесения вреда системе или кражи данных. К ним относятся вирусы, трояны, программы–вымогатели (ransomware) и шпионское ПО. Например, троян

может маскироваться под легитимное приложение, а после установки собирать данные или предоставлять хакерам удалённый доступ к системе. Малварь часто распространяется через фишинговые письма с заражёнными вложениями или ссылками.

Социальная инженерия – это манипуляция людьми для получения доступа к данным или системам. В отличие от фишинга, она может включать не только цифровые, но и физические методы. Например, злоумышленник может представиться сотрудником компании, чтобы выведать пароль по телефону, или проникнуть в офис под видом курьера. Цель – заставить жертву добровольно раскрыть информацию или выполнить действия, угрожающие безопасности.

Смишинг – это фишинг через SMS, где жертва получает сообщение с просьбой перейти по ссылке или сообщить данные. Вишинг использует телефонные звонки: мошенник, представляясь банком или службой поддержки, выманивает конфиденциальную информацию. Эти методы опасны, так как пользователи часто доверяют SMS или звонкам больше, чем письмам.

Для обеспечения информационной безопасности необходимо соблюдать следующие меры защиты:

– обучение сотрудников, регулярное обучение помогает распознавать не только фишинг, но и другие угрозы, сотрудники должны проверять адрес отправителя в письмах и SMS (например, настоящий домен не содержит ошибок), обращать внимание на орфографию, странные формулировки или необычные запросы, избегать перехода по ссылкам, скачивания вложений или предоставления данных по телефону;

– антифишинговые и антивирусные фильтры, антивирусное ПО с функцией защиты от подозрительных ссылок должно быть установлено на всех устройствах и регулярное обновление антивирусов и фильтров обязательно;

– обновление ПО и защита DNS, регулярные обновления устраняют уязвимости. Настройка защищённых DNS–серверов (например, с фильтрацией вредоносных доменов);

– шифрование данных, использование AES–256 для хранения и передачи данных защищает от перехвата;

– резервное копирование, защищает от программ–вымогателей, которые могут быть установлены через вредоносное ПО.

Защита информационных систем, хранящих персональные данные, требует комплексного подхода: обучение, технологии и строгие политики. Фишинг, вредоносное ПО, социальная инженерия, смишинг, вишинг и фарминг представляют серьёзные угрозы, эксплуатирующие человеческий фактор и технические уязвимости. Регулярное обучение, внедрение современных инструментов и бдительность сотрудников минимизируют риски и обеспечивают безопасность данных.

Безопасные условия труда начинаются с соответствия рабочей среды установленным нормам. Это включает контроль температуры, освещения и уровня шума. Например, температура в помещении должна быть в пределах 20–25 °C, чтобы избежать перегрева или переохлаждения. Для предотвращения профессиональных заболеваний применяются меры по снижению физической нагрузки. Например, работникам, чья деятельность связана с длительным пребыванием в одной позе, предоставляются перерывы каждые 45–60 минут для разминки. Также используются средства индивидуальной защиты (СИЗ), если работа связана с потенциально опасными факторами, такими как химические вещества или шумное оборудование.

Эргономичная организация рабочего места играет ключевую роль в предотвращении травм и повышении производительности. Рабочее кресло должно быть регулируемым по высоте, с поддержкой поясницы и удобной спинкой. Стол располагается так, чтобы монитор находился на расстоянии

50–70 см от глаз, а его верхняя граница была на уровне глаз или чуть ниже. Это помогает снизить нагрузку на шею и позвоночник.

Клавиатура и мышь должны быть расположены так, чтобы запястья оставались в нейтральном положении, без изгиба. Для этого можно использовать подставки под запястья или эргономичные устройства ввода. Пространство вокруг рабочего места должно быть свободным от лишних предметов, чтобы исключить риск спотыкания или случайного повреждения оборудования. Рабочее место включает стол и стул, а также монитор, расположенный на столе, рабочее место согласно ГОСТ 12.2.032–78 должно соответствовать следующим размерам:

– конструкция должна обеспечивать размещение на рабочей поверхности необходимого комплекта оборудования и документов с учётом характера выполняемой работы;

– регулируемая высота рабочей поверхности стола должна изменяться в пределах от 680 до 800 мм;

– высота рабочей поверхности стола при нерегулируемой высоте должна составлять 725 мм;

– размеры рабочей поверхности стола: глубина — не менее 600 (800) мм, ширина — не менее 1200 (1600) мм;

– рабочая поверхность стола не должна иметь острых углов и краёв.– монитор, ГОСТ 12.2.032–78 требует, чтобы верхний край экрана находился на уровне глаз или чуть ниже, а расстояние от глаз до экрана составляло 50–70 см;

– угол наклона экрана: На схеме угол наклона монитора не указан, но стандарт рекомендует угол наклона 10–20° для уменьшения бликов и нагрузки на шею.

– высота сиденья, 400–500 мм, в зависимости от роста пользователя, это обеспечивает правильное положение ног (бедра параллельны полу, угол в коленях около 90°);

– глубина сиденья, (150 мм от нижней части спинки до края сиденья) меньше рекомендованных значений (обычно 380–420 мм), что может быть некомфортно для длительного сидения, так как бедра не будут полностью поддерживаться.

Важным аспектом является организация хранения документов и инструментов. Все необходимое должно быть под рукой, но не загромождать рабочую поверхность. Например, использование вертикальных органайзеров или ящиков помогает поддерживать порядок и снижает вероятность отвлечения из-за поиска нужных предметов.

Длительная работа за компьютером связана с повышенными рисками профессиональных заболеваний, таких как синдром запястного канала, шейный остеохондроз и зрительные расстройства. Для их предотвращения в ООО «Эмвиджер» реализованы следующие меры:

Эргономичное оснащение рабочих мест. Сотрудники работают за столами с регулируемой высотой (65–80 см) и креслами с ортопедической поддержкой поясницы и шейного отдела. Мониторы с диагональю 21–27 дюймов оснащены антибликовым покрытием и установлены на расстоянии 50–70 см от глаз, что соответствует ГОСТ Р ИСО 1503–2014. Клавиатуры и мыши имеют эргономичный дизайн для снижения нагрузки на запястья.

Контроль освещения и микроклимата. Освещенность рабочих зон поддерживается на уровне 300–500 лк в соответствии с СНиП 23–05–95, с использованием светодиодных ламп для минимизации мерцания. Системы вентиляции обеспечивают приток свежего воздуха (30 м³/ч на человека) и поддерживают температуру 20–24 °C и влажность 40–60 % (СанПиН 2.1.3684–21).

Профилактика зрительного утомления. Мониторы имеют частоту обновления не менее 75 Гц и фильтры синего света, что снижает риск развития компьютерного зрительного синдрома. Сотрудникам предоставляются рекомендации по выполнению гимнастики для глаз каждые

45–60 минут работы.

Работа с компьютерной техникой сопряжена с рисками поражения электрическим током, коротких замыканий и повреждения оборудования. Для их минимизации:

Технические меры. Все электроустановки соответствуют ГОСТ Р 50571.3–2009, а системы заземления и устройства защитного отключения (УЗО) проходят ежеквартальные проверки. Используются кабели с двойной изоляцией и блоки питания с защитой от перегрузок.

Обучение персонала. Сотрудники проходят ежегодный инструктаж по электробезопасности, включающий правила работы с электрооборудованием, действия при обнаружении неисправностей и использование защитных средств, таких как диэлектрические перчатки и коврики.

Высокая концентрация электроники в рабочих помещениях повышает риск возгораний. ООО «Эмвиджер» применяет многоуровневый подход к пожарной безопасности:

Техническое оснащение. Установлены автоматические системы пожарной сигнализации с датчиками дыма (чувствительность 0,2 дБ/м) и тепла, соответствующие СП 5.13130.2009. В каждом помещении имеются углекислотные огнетушители (ОУ–5) и противопожарные покрывала.

Организационные меры. Проводятся ежеквартальные учения по эвакуации с моделированием различных сценариев, таких как задымление или отключение электроэнергии. Планы эвакуации размещены на видных местах, а аварийные выходы оснащены световыми указателями.

Профилактика. Используются огнестойкие материалы для отделки помещений, а серверные стойки оснащены системами автоматического отключения питания при перегреве.

Интенсивная работа с компьютерами и высокая ответственность за выполнение задач могут вызывать стресс и профессиональное выгорание.

Для их предотвращения:

Программы поддержки. Проводятся тренинги по управлению стрессом, основанные на методиках когнитивно–поведенческой терапии. Сотрудники обучаются техникам релаксации, таким как прогрессивная мышечная релаксация и дыхательные упражнения.

Гибкость рабочего процесса. Компания предоставляет возможность гибкого графика и удаленной работы, что позволяет сотрудникам адаптировать рабочий процесс под свои биоритмы. Это особенно важно для программистов и системных администраторов, чья работа требует высокой концентрации.

Создание комфортной среды. Офисы оформлены с учетом биофилического дизайна: используются комнатные растения, панорамные окна и зоны отдыха с мягкой мебелью. Исследования показывают, что такие элементы снижают уровень кортизола на 10–15 %.

Психологическая поддержка. Сотрудникам доступна анонимная консультационная линия с психологом для обсуждения рабочих и личных проблем.

5.2 Экологичность

Экологическая политика ООО «Эмвиджер» направлена на минимизацию воздействия на окружающую среду в соответствии с принципами устойчивого развития и стандартом ГОСТ Р ИСО 14001–2016. Работа с компьютерной техникой связана с такими вызовами, как утилизация электронных отходов, высокое энергопотребление и использование невозобновляемых ресурсов. Компания применяет комплексный подход, включающий технические, организационные и образовательные меры.

Электронные отходы содержат токсичные вещества, такие как ртуть, свинец и бромированные соединения, которые могут загрязнять почву, воду и воздух. Для их безопасной утилизации:

Сотрудничество с переработчиками. Компания передает отработанную

технику (компьютеры, мониторы, периферийные устройства) лицензированным организациям, соответствующим Федеральному закону № 89–ФЗ «Об отходах производства и потребления». Это позволяет извлекать ценные материалы (медь, алюминий, редкоземельные металлы) и предотвращать экологический ущерб.

Энергопотребление компьютерной техники и серверных систем составляет значительную часть экологического следа компании.

Энергоэффективное оборудование. Используются компьютеры и серверы, сертифицированные по стандарту ГОСТ Р 51387–99, что снижает энергопотребление на 25–35 % по сравнению с несертифицированными аналогами.

Интеллектуальные системы управления. Внедрены системы автоматического перехода техники в спящий режим и отключения питания в нерабочее время. Это позволило сократить энергопотребление на 18% за 2024 год, согласно внутренним замерам.

Мониторинг энергопотребления. Используются системы учета электроэнергии с аналитикой в реальном времени, что позволяет выявлять пиковые нагрузки и оптимизировать энергозатраты.

Вышедшая из строя оргтехника собирается в специально оборудованных помещениях, соответствующих СанПиН 2.1.7.1322–03. Отходы сортируются и хранятся в герметичных контейнерах не более 11 месяцев.

Используются картриджи с возможностью многократной заправки, а отработанные передаются организациям, специализирующимся на их рециклинге. Это снижает объем пластиковых отходов и потребность в производстве новых картриджей.

Использованные люминесцентные лампы, содержащие ртуть, собираются в герметичные металлические контейнеры, исключающие испарение ртути. Они передаются специализированным компаниям для

демеркуризации, что соответствует Федеральному закону № 7–ФЗ «Об охране окружающей среды».

Цифровизация процессов является ключевым инструментом экологической политики. При необходимости печати используются принтеры с функцией двусторонней печати и картриджи с возможностью многократной заправки. Проводятся регулярные тренинги по минимизации бумажного документооборота, включая рекомендации по использованию электронных подписей и облачных хранилищ.

ООО «Эмвиджер» активно внедряет инновационные решения для повышения экологической эффективности:

Модульная техника. Используются модульные компьютеры и серверы, позволяющие заменять отдельные компоненты (например, процессоры или накопители) вместо полной утилизации устройства. Это продлевает жизненный цикл техники на 2–3 года. Для оптимизации процессов тестирования и ремонта внедряются технологии цифровых двойников, что сокращает необходимость в физических прототипах и снижает отходы на 10–15 %.

5.3 Чрезвычайные ситуации

Современные предприятия, работающие в сфере информационных технологий, сталкиваются с широким спектром потенциальных чрезвычайных ситуаций, включая пожары, сбои электроснабжения, кибератаки и природные катастрофы. Для ООО «Эмвиджер», специализирующегося на разработке, обслуживании и эксплуатации компьютерных систем, обеспечение готовности к таким ситуациям является неотъемлемой частью корпоративной политики. Управление чрезвычайными ситуациями в компании строится на основе строгого соблюдения нормативных требований, закрепленных в ГОСТ Р 22.0.02–2016 «Безопасность в чрезвычайных ситуациях», Федеральном законе № 68–ФЗ «О защите населения и территорий от чрезвычайных ситуаций природного и

техногенного характера», а также других стандартах, регулирующих безопасность в технологическом секторе. Такой подход позволяет минимизировать риски для сотрудников, оборудования и данных, а также обеспечить непрерывность бизнес–процессов.

Основой управления чрезвычайными ситуациями в ООО «Эмвиджер» является систематическая идентификация и оценка потенциальных угроз, которые могут повлиять на деятельность компании. Учитывая специфику работы с компьютерной техникой, особое внимание уделяется техногенным рискам, таким как возгорания, вызванные неисправностью электрооборудования, и киберугрозам, связанным с утечкой данных или нарушением работы серверов. Для анализа рисков компания применяет методику FMEA (Failure Modes and Effects Analysis), рекомендованную ГОСТ Р ИСО 31010–2011, что позволяет выявить наиболее вероятные сценарии ЧС и определить их потенциальные последствия. Например, высокая концентрация электроники в серверных помещениях увеличивает вероятность пожаров, а зависимость от стабильного электроснабжения делает компанию уязвимой к отключениям электроэнергии.

Для обеспечения пожарной безопасности в офисе ООО «Эмвиджер» применяются всесторонние меры, направленные на предотвращение возгораний и снижение возможных рисков. Электрические сети регулярно проходят проверку на соответствие стандартам и отсутствие дефектов, чтобы исключить вероятность короткого замыкания. Всё используемое электрооборудование, включая рабочие компьютеры и серверы, оснащено системой заземления. Для защиты от сбоев в подаче электроэнергии установлены устройства бесперебойного питания, которые позволяют оборудованию работать автономно до 30 минут.

В помещениях офиса размещены автоматические датчики пожарной сигнализации, реагирующие на дым и повышение температуры. Также функционирует система звукового оповещения, которая в случае опасности

информирует сотрудников о необходимости эвакуации. На каждые 100 м² офисной площади предусмотрено минимум два огнетушителя — порошковые и углекислотные, подходящие для ликвидации возгораний электрооборудования. В серверной комнате используется автоматическая система газового пожаротушения, безопасная как для техники, так и для людей.

На видных местах в офисе вывешены схемы эвакуационных путей. Сотрудники ежегодно проходят обучение и участвуют в тренировочных эвакуациях, чтобы быть готовыми к действиям в экстренных ситуациях. Горючие материалы, такие как бумага или картон, хранятся в специально отведённых местах, вдали от источников тепла. Люминесцентные лампы, содержащие ртуть, помещаются в герметичные контейнеры и передаются на утилизацию в специализированные компании, что минимизирует риск пожара и экологического вреда.

При возникновении пожара приоритет отдаётся безопасности людей. Сотрудники должны немедленно покинуть здание, следуя указанным эвакуационным маршрутам и ориентируясь на сигналы системы оповещения, после чего собраться в заранее определённой безопасной зоне. Если очаг возгорания небольшой и не представляет серьёзной угрозы, специально обученный сотрудник может использовать огнетушитель, направляя струю на основание пламени и избегая вдыхания токсичного дыма.

Профилактика чрезвычайных ситуаций занимает центральное место в стратегии компании. Для предотвращения техногенных аварий серверные помещения оснащаются системами бесперебойного питания (ИБП) с автономностью до двух часов, а также резервными дизель–генераторами, способными поддерживать работу критически важных систем в течение длительных перебоев с электроснабжением. Системы охлаждения серверов имеют резервирование по схеме N+1, что исключает риск перегрева оборудования, согласно требованиям ГОСТ Р 58065–2018. Для защиты от

киберугроз применяются межсетевые экраны, системы обнаружения вторжений (IDS) и антивирусное программное обеспечение с регулярными обновлениями. Компания проводит стресс–тестирование своих систем на устойчивость к DDoS–атакам, что соответствует рекомендациям ГОСТ Р 57580.1–2017 по обеспечению безопасности информационных систем. Персонал проходит регулярное обучение по действиям в условиях ЧС, включая использование средств пожаротушения, эвакуацию и восстановление данных после сбоев. Эти тренинги проводятся в соответствии с ГОСТ Р 22.3.15–2018, который регламентирует подготовку сотрудников к чрезвычайным ситуациям.

В случае возникновения чрезвычайной ситуации ООО «Эмвиджер» обеспечивает оперативное реагирование, основанное на заранее разработанных планах действий. Для каждого типа ЧС, будь то пожар, сбой электроснабжения, созданы детализированные алгоритмы, включающие координацию с экстренными службами, эвакуацию персонала и защиту данных. Например, при обнаружении пожара автоматические системы сигнализации, соответствующие СП 5.13130.2009, активируют датчики дыма с чувствительностью 0,2 дБ/м, а сотрудники следуют четким инструкциям по эвакуации, размещенным на видных местах. Аварийные выходы оснащены световыми указателями, а помещения — углекислотными огнетушителями (ОУ–5), что обеспечивает быстрое подавление очагов возгорания. Для управления кризисом формируется оперативный штаб, состоящий из руководителей отделов ИТ, безопасности и логистики, который координирует действия в реальном времени. Важным элементом реагирования является защита данных: ежедневное резервное копирование в двух независимых дата–центрах позволяет восстановить информацию в течение четырех часов после сбоя, что минимизирует потери.

Восстановление после чрезвычайных ситуаций направлено на скорейшее возобновление бизнес–процессов и предотвращение повторных

инцидентов. Например, если сбой был вызван неисправностью оборудования, компания обновляет процедуры профилактического обслуживания, включая проверку блоков питания и кабелей. Для восстановления инфраструктуры используются заранее подготовленные резервные серверы и комплектующие, что позволяет сократить время простоя до нескольких часов.

Сотрудничество с органами государственной власти и экстренными службами является важной частью управления ЧС. ООО «Эмвиджер» регулярно взаимодействует с региональными подразделениями МЧС, участвуя в совместных учениях по ликвидации пожаров и других аварий. Такие учения проводятся не реже одного раза в год и включают моделирование сценариев, таких как эвакуация при задымлении или восстановление после кибератаки. Компания также консультируется с МЧС по вопросам обновления планов реагирования, чтобы обеспечить их соответствие актуальным требованиям законодательства.

Таким образом, управление чрезвычайными ситуациями в ООО «Эмвиджер» представляет собой комплексный процесс, охватывающий профилактику, реагирование и восстановление. Опираясь на ГОСТ Р 22.0.02–2016 и другие нормативные документы, компания обеспечивает высокий уровень готовности к потенциальным угрозам, минимизируя риски для сотрудников, оборудования и данных. Этот подход не только гарантирует непрерывность бизнес–процессов, но и укрепляет репутацию компании как надежного и ответственного участника ИТ–рынка, способного эффективно справляться с вызовами в условиях неопределенности.

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы была разработана информационная система для автоматизации управления проектами в компании, занимающейся бухгалтерским ПО. Цель – повысить эффективность, сократить рутину и улучшить прозрачность задач.

На основе анализа выявленных проблем (отсутствие единого хранилища, слабая автоматизация, плохой контроль сроков) была спроектирована система, адаптированная под нужды компании. Были определены ключевые сущности и их связи, разработан удобный интерфейс с акцентом на юзабилити.

Для реализации использовались современные технологии: HTML5, CSS3, JavaScript, Bootstrap (фронтенд), PHP + ModX (бэкенд), RedBeanPHP (ORM), MySQL (база данных). Это обеспечило высокую производительность, безопасность и простоту поддержки.

Были реализованы меры защиты от SQL-инъекций, XSS и CSRF-атак, шифрование паролей, контроль прав доступа и проверка загрузок. Система прошла полное тестирование – модульное, функциональное и нагрузочное, показав отличные результаты.

Решение полностью соответствует поставленным целям, превосходя аналоги по гибкости, возможности локального запуска и отсутствию лицензионных ограничений. Оно подходит для малого и среднего бизнеса и легко масштабируется.

Эта работа стала примером успешного применения современных технологий для решения реальных бизнес-задач. Результат – практичная, надёжная и безопасная система, способная повысить эффективность управленческой деятельности.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1 Гаспариан М. С. Информационные системы, 2002. 4 с.

2 Логинов А. А. Основы проектирования информационных систем. СПб.: Питер, 2019. 66 с.

3 Зудилова Т. В. Буркова М. Л. Web–программирование HTML Учебное пособие. М.: Инфра–Инженерия, 2012. 5 с.

4 Дэвид Ф. JavaScript. Подробное руководство, пятое издание. 20 с.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Кренке, Д. В. Информационные системы. Теория и практика / Д. В. Кренке. – М.: Финансы и статистика, 2020. – 352 с.

2 Логинов, А. А. Основы проектирования информационных систем / А. А. Логинов. – СПб.: Питер, 2015. – 416 с.

3 Бендер, Р. А. Системный анализ и проектирование информационных систем / Р. А. Бендер. – М.: Академия, 2018. – 288 с.

4 Касперович, А. В. Проектирование информационных систем: учебное пособие / А. В. Касперович. – Минск: Вышэйшая школа, 2021. – 240 с.

5 Фленов, М. Е. PHP: Полный справочник / М. Е. Фленов. – СПб.: Питер, 2021.

6 Карпова, Т. С. Базы данных: модели, разработка, администрирование / Т. С. Карпова. – М.: Академия, 2018.

7 Введение в MySQL и SQL / [Электронный ресурс]. – Режим доступа: https://www.mysql.com /. – 15.10.2023.

8 Фролов, А. В. JavaScript. Современный учебный курс / А. В. Фролов. – М.: ДМК Пресс, 2020.

9 Дунаев, В. В. Веб–программирование для начинающих / В. В. Дунаев. – СПб.: БХВ–Петербург, 2021.

10 Фленов, М. Е. JavaScript и jQuery: справочник разработчика / М. Е. Фленов. – СПб.: БХВ–Петербург, 2020. – 512 с.

11 ГОСТ 19.101–90 Единая система программной документации. Виды программ и программных документов. – М.: Издательство стандартов, 1990.

12 ГОСТ Р 51514–2000 Информационная технология. Комплекс стандартов на автоматизированные системы. – М.: Госстандарт России, 2000.

13 Круп, Д. JavaScript–приложения в архитектуре MVC / Д. Круп. – М.: Вильямс, 2019. – 432 с.

14 Севи, Э. PHP 8. На примерах / Э. Севи. – М.: ДМК Пресс, 2021. – 368 с.

15 Круг, С. Не заставляйте думать! Правила удобного веб–дизайна / С. Круг. – М.: Вильямс, 2020. – 256 с.

16 phpMyAdmin: официальный сайт / [Электронный ресурс]. – Режим доступа: https://www.phpmyadmin.net /. – 15.10.2024.

17 Проектное управление: методические рекомендации / [Электронный ресурс]. – Режим доступа: https://pmi.org /. – 15.10.2024.

18 Афанасьев, В. В. Управление проектами информационных систем / В. В. Афанасьев. – М.: Альпина Паблишер, 2021. – 304 с.

19 Холл, М. Информационные технологии управления проектами / М. Холл. – М.: Академия, 2019. – 272 с.

21 Орлов, С. А. Технологии разработки программного обеспечения / С. А. Орлов. – СПб.: Питер, 2020. – 448 с

22 Шестаков, Р. И. Безопасность информационных систем: подходы и практики / Р. И. Шестаков. – М.: Техносфера, 2025. – 740 с.

23 ГОСТ Р ИСО 1503–2014. Эргономика. Требования к пространственной ориентации и направлениям движения органов управления.

24 СанПиН 2.1.3684–21 "Санитарно–эпидемиологические требования к содержанию территорий городских и сельских поселений, к водным объектам, питьевой воде и питьевому водоснабжению населения, атмосферному воздуху, почвам, жилым помещениям, эксплуатации производственных, общественных помещений, организации и проведению санитарно–противоэпидемических (профилактических) мероприятий".

25 ГОСТ Р 50949–2001. Средства отображения информации индивидуального пользования. Методы измерений и оценки эргономических параметров и параметров без опасности.

ПРИЛОЖЕНИЕ А

Техническое задание

1 ОБЩИЕ СВЕДЕНИЯ

1.1 Полное наименование системы

Информационная система управления проектами для компании по разработке бухгалтерского программного обеспечения.

1.2 Плановые сроки:

– начало работ: 01.10.2024;

– окончание работ: 30.05.2025.

2 НАЗНАЧЕНИЕ И ЦЕЛИ СИСТЕМЫ

2.1 Назначение системы

Система предназначена для автоматизации и централизации процессов управления проектами в компании, занимающейся разработкой бухгалтерского программного обеспечения. Предоставляет инструменты для планирования задач, распределения нагрузки, отслеживания прогресса, коммуникации между сотрудниками и контроля сроков выполнения проектов.

2.2 Цели создания системы

– повысить эффективность работы команд за счёт централизованного хранения данных;

– улучшить коммуникацию между сотрудниками и отделами;

– автоматизировать рутинные процессы управления проектами (отчеты, назначение задач, напоминания);

– сократить количество ошибок при управлении задачами.

Критерии оценки успеха:

– сокращение времени на планирование проектов на 25 %;

– повышение прозрачности проектов на 40 %;

– снижение количества просроченных задач на 30 %.

3 ТРЕБОВАНИЯ К СИСТЕМЕ

3.1 Требования к системе в целом

Структура и функционирование: модуль управления проектами, модуль управления задачами, модуль пользователей, модуль отчетов.

Численность и квалификация персонала: администратор и сотрудники с базовыми навыками работы с компьютером. Обучение проводится автором разработки.

Надежность: система должна корректно работать при одновременном доступе до 20 пользователей.

Безопасность: защита от SQL-инъекций, ограничение прав доступа, шифрование паролей.

Эргономика: удобный интерфейс, поддержка мобильных устройств, минимальное время освоения.

3.2 Требования к функциям системы:

– создание, редактирование и удаление проектов;

– добавление, назначение, изменение статуса задач;

– реализация системы ролей (ответственный, наблюдатель, создатель);

3.3 Требования к видам обеспечения:

– математическое обеспечение: алгоритмы расчета сроков выполнения задач и загрузки сотрудников;

– информационное обеспечение: использование базы данных MySQL через PHPMyAdmin;

– программное обеспечение: совместимость с Windows, macOS, Linux, iOS, Android;

– техническое обеспечение: сервер с поддержкой PHP 8+, клиентские устройства с поддержкой современных браузеров

Продолжение ПРИЛОЖЕНИЯ А

4 СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ СИСТЕМЫ

4.1 Этапы работ:

– анализ предметной области и требований заказчика;

– проектирование архитектуры системы и ER-диаграмм;

– разработка прототипа интерфейса;

– реализация клиентской и серверной частей;

– тестирование (модульное, функциональное, нагрузочное);

5 ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ СИСТЕМЫ

5.1 Контроль качества:

– проверка соответствия требованиям технического задания;

– тестирование производительности, безопасности и совместимости;

– проверка корректности работы всех модулей.

5.2 Приемка системы

Система считается принятой после успешного прохождения всех этапов тестирования и подписания акта приемки.

6 ТРЕБОВАНИЯ К ПОДГОТОВКЕ ОБЪЕКТА АВТОМАТИЗАЦИИ

6.1 Подготовка персонала

– проведение обучения сотрудников работе с системой;

– подготовка материалов по основам использования системы.

6.2 Техническая подготовка

– установка необходимого оборудования и программного обеспечения;

– настройка сервера и клиентских рабочих мест.

7 ТРЕБОВАНИЯ К ДОКУМЕНТИРОВАНИЮ

7.1 Перечень документов:

– техническое задание;

– руководство пользователя;

– руководство администратора;

– описание архитектуры системы;

– код программы;

– результаты тестирования;

– пояснительная записка выпускной квалификационной работы.

8 ИСТОЧНИКИ РАЗРАБОТКИ

8.1 Нормативные документы:

– ГОСТ 34.602-89 «Техническое задание на создание автоматизированной системы»;

– ГОСТ 34.601-90 «Автоматизированные системы. Стадии создания».

8.2 Литература:

– материалы по HTML5, CSS3, JavaScript, Bootstrap;

– источники по использованию PHP, CMS ModX, ORM RedBeanPHP;

– исследования по методологиям управления проектами;

– документация по Git и MVC-паттерну.