МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки)

Разработка программного обеспечения
(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

На тему Разработка информационной системы для планирования и учёта рабочего времени сотрудников

Обучающийся А.В. Фролова
(И.О. Фамилия) (личная подпись)

Руководитель к. э. н., доцент, Т.А. Раченко
(ученая степень, звание, И.О. Фамилия)

Консультант к. ф. н., доцент, М.В. Дайнеко
(ученая степень, звание, И.О. Фамилия)

Тольятти 2025

Аннотация

Тема бакалаврской работы «Разработка информационной системы для планирования и учёта рабочего времени сотрудников».

Объектом исследования является – рабочее время сотрудников предприятия.

Предметом выпускной квалифицированной работы является процесс учёта рабочего времени сотрудников.

Цель выпускной квалифицированной работы – разработка и внедрение информационной системы, позволяющей планировать рабочее время сотрудников.

Структура работы состоит из введения, трех глав, заключения и списка используемой литературы и источников.

В работе представлено поэтапное проектирование и реализация информационной системы учета рабочего времени. В первой главе рассматривается предметная область, обосновывается необходимость автоматизации, проводится анализ существующих решений. Во второй главе описаны архитектура системы, структура базы данных, реализация бизнес-логики и интерфейса. В третьей главе проводится тестирование, анализируются результаты и эффективность внедрения.

Результатом работы стало полностью функциональное программное решение, автоматизирующее учет рабочего времени и повышающее эффективность управления персоналом.

Бакалаврская работа состоит из 59 страниц текста, 32 рисунка, 9 таблиц и 33 источника.

Abstract

The title of the bachelor's thesis is "Development of an Information System for Planning and Recording Employee Working Time."

The object of the research is the working time of enterprise employees.

The subject of the bachelor's thesis is the process of recording employee working time.

The aim of the bachelor's thesis is to develop and implement an information system that enables planning and tracking of employee working hours.

The structure of the thesis includes an introduction, three chapters, a conclusion, and a list of references.

The thesis presents a step-by-step design and implementation of an information system for working time tracking. The first chapter examines the subject area, justifies the need for automation, and analyzes existing solutions. The second chapter describes the system architecture, database structure, business logic, and user interface implementation. The third chapter covers system testing, analysis of results, and evaluation of implementation effectiveness.

The result of the work is a fully functional software solution that automates working time tracking and improves the efficiency of personnel management.

The bachelor's thesis consists of 59 pages of text, 32 figures, 9 tables, and 33 references.

Оглавление

Введение 5
Глава 1 Постановка задачи на разработку информационной системы для предприятия 8
1.1 Значение программного обеспечения для информационной системы учета рабочего времени 8
1.2 Обзор существующих решений и их анализ 13
1.3 Требования к функционалу и интерфейсу приложения 16
Глава 2 Процесс разработки информационной системы 22
2.1 Проектирование программного обеспечения 22
2.2 Выбор технологий и инструментов для разработки 35
2.3 Реализация функциональных требований 37
Глава 3 Оценка эффективности разработанного программного обеспечения 46
3.1 Тестирование и отладка приложения 46
3.2 Оценка удобства и функциональности системы 50
Заключение 54
Список используемой литературы и источников 56

Введение

В современных условиях ведения бизнеса эффективное управление рабочим временем сотрудников является одним из ключевых факторов повышения производительности труда и успешной реализации проектов [6], [10]. Однако во многих компаниях, включая ООО «ВорлдИнтерТех Рус», процесс учёта рабочего времени до сих пор осуществляется вручную или с использованием устаревших инструментов [11], что приводит к значительным временным потерям, ошибкам в отчётности и неэффективному распределению ресурсов [4].

Основная проблема заключается в отсутствии автоматизированной системы, способной обеспечить точный контроль рабочего времени, оперативное планирование задач и формирование аналитических отчётов. Это снижает прозрачность управления, увеличивает административную нагрузку на персонал и затрудняет оценку реальной загруженности сотрудников. Поэтому разработка информационной системы, способствующей планированию и учету рабочего времени сотрудников, становится необходимой на любом уровне управления.

Актуальность темы выпускной квалификационной работы обусловлена необходимостью внедрения эффективного средства автоматизации процессов учета и планирования рабочего времени сотрудников на предприятии. Применение специализированной информационной системы позволит сократить издержки, связанные с ручным учетом, минимизировать ошибки и повысить прозрачность трудовой дисциплины.

На сегодняшний день существует множество аналогичных решений от простых десктопных программ до облачных систем учета рабочего времени (например, Bitrix24, Yaware, TimeDoctor и др.). Тем не менее, большинство из них либо ориентированы на крупные предприятия с избыточным функционалом, либо не позволяют гибко адаптироваться под специфику работы конкретной компании. Это подчеркивает целесообразность разработки

собственного программного обеспечения, адаптированного под конкретные требования и задачи предприятия.

Объектом исследования является – рабочее время сотрудников предприятия.

Предметом выпускной квалифицированной работы является процесс учета рабочего времени сотрудников.

Целью настоящей работы является разработка информационной системы для планирования и учета рабочего времени сотрудников, отвечающей современным требованиям к удобству, функциональности и эффективности применения в реальных условиях предприятия.

В соответствии с целью были поставлены следующие задачи:

1. Проанализировать существующие решения для учета рабочего времени и обосновать необходимость разработки собственной системы.

2. Определить требования к функционалу и интерфейсу информационной системы учета рабочего времени.

3. Разработать архитектуру программного обеспечения и выбрать технологии реализации.

4. Реализовать основные функции системы учета рабочего времени сотрудников.

5. Провести тестирование программного обеспечения и оценить его удобство и эффективность.

В работе применяются такие методы исследования, как анализ предметной области, моделирование бизнес-процессов, объектно-ориентированное проектирование [12], использование современных технологий программирования и проведение функционального тестирования.

Новизна работы заключается в создании специализированного программного продукта, адаптированного под нужды конкретного предприятия, без избыточного функционала, с акцентом на простоту и практическую применимость.

Практическая значимость состоит в возможности внедрения

разработанной информационной системы в рабочие процессы предприятия ООО «ВорлдИнтерТех Рус» с целью повышения точности учета, прозрачности данных и эффективности внутреннего контроля.

Работа состоит из введения, трех глав, заключения и списка использованной литературы. В первой главе рассматриваются особенности предметной области, проводится анализ существующих решений по учёту рабочего времени, а также формулируются функциональные и нефункциональные требования к разрабатываемой информационной системе. Это позволяет обосновать необходимость создания собственного программного продукта, адаптированного под нужды предприятия.

Во второй главе описан процесс проектирования и разработки программного обеспечения. Рассматриваются архитектура системы, структура базы данных, выбор технологий, а также реализация основных модулей: учёта времени, работы с задачами и формирования отчётности. Также представлены диаграммы, иллюстрирующие бизнес-логику и взаимодействие компонентов системы.

Третья глава посвящена оценке эффективности разработанной системы. В ней описываются этапы тестирования, анализируются результаты опытной эксплуатации, а также проводится оценка удобства интерфейса и соответствия функционала требованиям пользователей. На основании полученных данных делается вывод о готовности системы к внедрению.

Глава 1 Постановка задачи на разработку информационной системы для предприятия

1.1 Значение программного обеспечения для информационной системы учета рабочего времени

В современном цифровом обществе информационные системы стали ключевым инструментом обеспечения стабильной и эффективной деятельности организаций. Они играют фундаментальную роль в управлении, координации и оптимизации бизнес-процессов. Информационные системы представляют собой совокупность программных, технических и организационных средств [1], [15], предназначенных для сбора, хранения, обработки и анализа данных с целью поддержки управленческих решений. На примере ООО «ВорлдИнтерТех Рус» важным направлением внедрения ИС является автоматизация учёта рабочего времени сотрудников, поскольку текущие методы, основанные на ручном вводе данных, сопровождаются высокой вероятностью ошибок, дублированием информации и недостаточной прозрачностью. Эти проблемы приводят к снижению дисциплины труда, неэффективному использованию ресурсов и увеличению затрат на управление персоналом. Внедрение ИС позволяет устранить данные недостатки за счёт автоматизированного сбора информации, аналитики и формирования отчётности. Так, согласно исследованию, проведённому в хлебопекарной отрасли, внедрение ERP-систем позволило повысить производительность труда [4] на 12,5% и сократить издержки на 8,2%.

ООО «ВорлдИнтерТех Рус» – это компания, предоставляющая услуги по разработке программного обеспечения, включая создание веб-приложений, мобильных приложений и систем управления. Основной целью компании ООО «ВорлдИнтерТех Рус» является предоставление высококачественных услуг в области разработки ПО, способствующих достижению успеха ее клиентов и удовлетворению их потребностей. Компания стремится стать

лидером на рынке IT-услуг, обеспечивая инновационные решения и оптимизацию бизнес-процессов.

В соответствии с Общероссийским классификатором видов экономической деятельности (ОКВЭД), предприятие ООО «ВорлдИнтерТех Рус» осуществляет следующие виды деятельности:

- 62.01 – Разработка программного обеспечения для ЭВМ;
- 62.02 – Консультирование по вопросам компьютерных технологий;
- 72.20 – Исследование и разработки в области общественных и гуманитарных наук;
- и другие связанные секторы, что подтверждает широкую направленность в области информационных технологий и услуг.

Для достижения своей цели компания ставит перед собой следующие задачи:

1. Разработка инновационного ПО:

- создание и внедрение современных программных решений, которые отвечают актуальным требованиям бизнеса;
- использование передовых технологий и подходов в разработке, таких как Agile и DevOps.

2. Увеличение клиентской базы:

- привлечение новых клиентов и удержание существующих через эффективные стратегии маркетинга и продаж;
- предоставление качественного сервиса и поддержки для обеспечения удовлетворенности клиентов.

3. Постоянное обучение и развитие сотрудников:

- внедрение программ обучения и повышения квалификации для команды, чтобы обеспечить высокий уровень профессионализма и компетенции;
- создание внутренней культуры, способствующей обмену знаниями и опытом.

4. Оптимизация бизнес-процессов:

- внедрение автоматизированных систем для улучшения внутренней эффективности, включая учет рабочего времени и управление проектами;
- модернизация процессов компании для сокращения временных и финансовых затрат.

Одной из ключевых проблем, с которой столкнулось ООО «ВорлдИнтерТех Рус», стала неэффективность планирования и учета рабочего времени сотрудников. Долгое время этот процесс осуществлялся вручную с использованием табелей, что приводило к значительным административным затратам, высокой вероятности ошибок и отсутствию оперативного доступа к аналитической информации. Такой подход оказывал негативное влияние как на эффективность управления персоналом, так и на качество проектного планирования.

Причинно-следственный анализ показал, что неавтоматизированная система учета рабочего времени способствует росту управленческих рисков и снижению производительности [6]. Ручной учет требует значительных ресурсов, не обеспечивает актуальности данных и делает невозможным анализ загруженности персонала в реальном времени. На рисунке 1 представлены основные последствия использования устаревших подходов к учету рабочего времени.

Рисунок 1 – Диаграмма «Причина-следствие» неэффективности ручного учета

Разработка информационной системы позволяет устранить вышеперечисленные проблемы. Программное обеспечение автоматизирует сбор и обработку данных, исключает человеческие ошибки, предоставляет руководству инструменты для анализа и контроля, а также упрощает формирование отчётности.

На рисунке 2 представлено сравнение ключевых показателей до и после внедрения информационной системы.

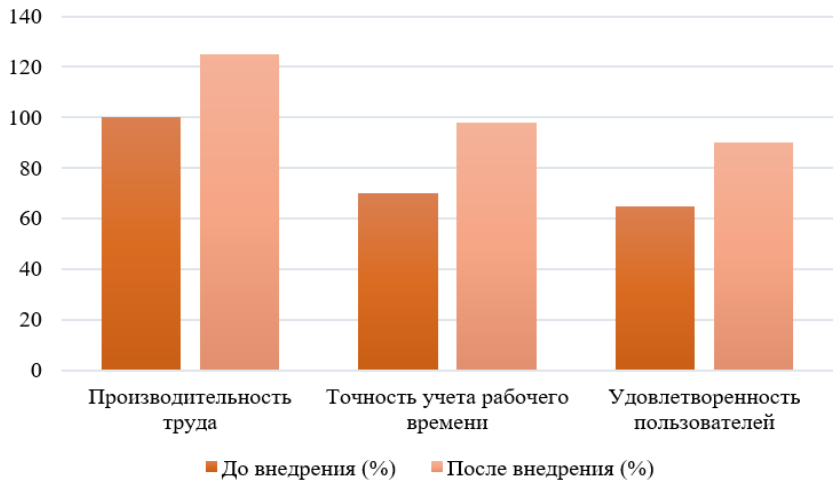

Рисунок 2 – Улучшение ключевых показателей после внедрения ИС

Как видно из диаграммы, производительность труда увеличилась на 25%, точность учета – на 28%, а удовлетворенность пользователей выросла на 25 процентных пунктов. Это подтверждает эффективность разработанного программного обеспечения.

Сегодня технологии быстро развиваются, и это сильно влияет на способы учёта рабочего времени. Например, во многих компаниях начали использовать биометрические системы – такие как отпечатки пальцев или распознавание лиц – чтобы точно фиксировать, когда сотрудник приходит и уходит с работы.

Также всё чаще применяются облачные технологии. Это значит, что данные хранятся не на одном компьютере, а в интернете, и к ним можно получить доступ из любой точки мира. Это удобно и надёжно.

Информационные системы учета времени всё чаще связывают с другими программами компании – например, с бухгалтерией или системой управления проектами. Это помогает работать быстрее и удобнее, так как все данные находятся в одном месте.

Дополнительно начинают использоваться технологии искусственного интеллекта. Они помогают анализировать, как работает персонал, и находить возможности для улучшения. А с помощью «умных» датчиков и устройств (это называется Интернет вещей) можно автоматически фиксировать присутствие сотрудников без их участия (рисунок 3).

Рисунок 3 – Технологические тренды в сфере учета времени

Программное обеспечение сегодня – это важная часть любой современной компании. Оно помогает быстрее решать задачи, снижает ошибки и упрощает управление. Для ООО «ВорлдИнтерТех Рус» создание системы учета рабочего времени особенно важно, так как это позволит уйти от ручного ввода данных, сделать работу более точной и эффективной, а также сэкономить время и ресурсы.

1.2 Обзор существующих решений и их анализ

На сегодняшний день на рынке представлены разнообразные решения, от простых таймеров до комплексных платформ, интегрирующих функции управления проектами, отчетности и анализа. Каждое из этих решений имеет свои особенности, преимущества и недостатки, которые стоит учитывать при выборе оптимальной системы для конкретной организации. Рассмотрим некоторые из популярных продуктов, их функциональные возможности, а также подходы, которые позволят развить более эффективные практики работы с учетом рабочего времени сотрудников.

Time Doctor. Идеальное решение для компаний и фрилансеров, нуждающихся в детальном анализе использования рабочего времени. Предоставляет подробную информацию о продуктивности сотрудников, особенно ценную для удалённых команд. Фокус – на аналитике и отслеживании активности [31].

Harvest. Подходит для фрилансеров и небольших организаций. Отличается интегрированными функциями выставления счетов, учёта расходов и управления командой, что делает его удобным инструментом для комплексного управления проектами и финансами [26].

Toggl Track. Универсальное облачное решение для индивидуальных пользователей и команд. Простой и эффективный инструмент для отслеживания рабочих часов, сфокусированный на удобстве использования и понятном интерфейсе [32].

Bitcop. Представляет собой более продвинутую систему, которая охватывает не только учет рабочего времени, но и отслеживание активности сотрудников на компьютере. Она идеально подходит для компаний, требующих более тщательного контроля [23].

CrocoTime. В свою очередь, также автоматизирует процесс учета рабочего времени и обеспечивает мониторинг компьютерной активности сотрудников. Ориентировано на автоматизацию и предотвращение нецелевого использования рабочего времени [25].

Сравнительный анализ представлен в Таблице 1.

Таблица 1 – Сравнительный анализ аналогов программного обеспечения

Критерий	Time Doctor	Harvest	Toggl Track	Bitcop	CrocoTime
Фиксация времени	Да (авто и ручная)	Да (ручная)	Да (ручная)	Да (автоматическая)	Да (авто и ручная)
Мониторинг активности	Да (скриншоты, сайты, приложения)	Нет	Нет	Да (детальный контроль)	Да (мониторинг приложений)
Учет проектов и задач	Частично	Да (проекты и бюджеты)	Да	Нет	Частично
Отчётность и аналитика	Подробные отчеты, экспорт	Интуитивные отчеты	Визуальная аналитика	Отчеты о времени и бездействии	Визуальные графики, отчеты
Интеграции с другими системами	Slack, Asana, Trello и др.	Более 50 интеграций	100+ интеграций	Нет	Outlook, Excel, Active Directory
Поддержка многопользовательского режима	Да	Да	Да	Да	Да
Тип доступа	Облако	Облако	Облако	Установка на ПК	Локальная установка
Платформы	Windows, macOS, Linux, Android, iOS	Windows, macOS, Android, iOS	Windows, macOS, Android, iOS	Windows	Windows

Продолжение таблицы 1

Безопасность данных	Шифрование, резервное копирование	Стандартное шифрование	Стандартное шифрование	Повышенный контроль, доступ по ролям	Локальное хранение, резервные копии
Стоимость	Средняя (от $10/польз. в мес.)	Средняя (от $12/мес.)	Бесплатно + платные тарифы	Средняя	По запросу (лицензии от 10 пользователей)

Анализ существующих решений показал, что ни одно из них не удовлетворяет требованиям ООО «ВорлдИнтерТех Рус» полностью. Готовые продукты либо содержат избыточный функционал, не нужный компании (например, постоянный мониторинг экрана), либо не поддерживают нужные функции, такие как интеграция с внутренними базами, учет задач по проектам и гибкая ролевая модель. Кроме того, использование стороннего ПО связано с рисками безопасности и высокой стоимостью долгосрочного использования.

Собственная разработка позволит адаптировать систему под внутренние процессы, обеспечить безопасность данных, интеграцию с корпоративной инфраструктурой и избежать постоянных лицензионных затрат. Это решение отвечает как текущим потребностям предприятия, так и перспективам масштабирования.

Для наглядной иллюстрации взаимодействия пользователя с разрабатываемой системой учета рабочего времени, ниже представлена диаграмма взаимодействия (рисунок 4).

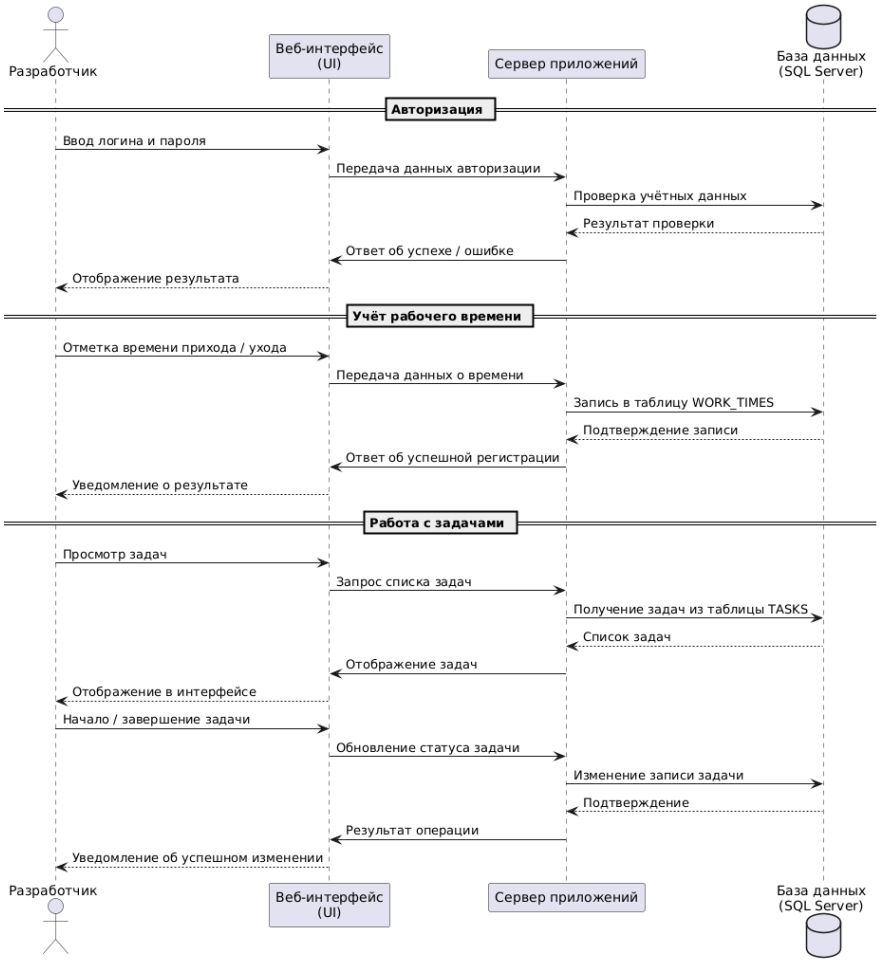

Рисунок 4 – Диаграмма взаимодействия разработчика с системой

Данная диаграмма демонстрирует, как разработчик осуществляет основные действия: авторизацию, фиксацию рабочего времени, а также работу с задачами через веб-интерфейс, при участии серверной части и базы данных.

1.3 Требования к функционалу и интерфейсу системы

Для успешной разработки и внедрения информационной системы учёта рабочего времени сотрудников в ООО «ВорлдИнтерТех Рус» необходимо чётко определить требования к её функционалу и интерфейсу. Эти требования делятся на функциональные и нефункциональные, и их выполнение обеспечит соответствие приложения ожиданиям пользователей, его надёжность и

эффективность.

В таблице 2 представлены ключевые функции разрабатываемой системы с кратким пояснением их назначения.

Таблица 2 – Основные функции информационной системы

Функция	Краткое описание
Регистрация пользователей	Создание учетных записей с привязкой к ролям и отделам
Авторизация	Безопасный вход с проверкой прав доступа
Учёт рабочего времени	Фиксация времени прихода и ухода сотрудников
Управление задачами	Назначение, просмотр и редактирование задач
Формирование отчётов	Генерация отчетов по сотрудникам и отделам в различных форматах
Управление ролями и правами	Назначение ролей (сотрудник, руководитель, администратор)
Интеграция с Outlook/Excel	Экспорт данных и отчётов в офисные приложения
Аналитика	Статистика по времени, выполненным задачам и загрузке сотрудников

Разработка этих функций учитывает реальные задачи предприятия и призвана заменить разрозненные ручные процессы единым автоматизированным решением. Далее представлены диаграммы, иллюстрирующие ключевые процессы, важные как для технической реализации, так и для проверки качества системы.

Одним из ключевых процессов является вход в систему. На рисунке 5 показан пошаговый процесс создания и проверки учетной записи: от ввода данных до их валидации, записи в базу данных и выдачи токена/сессии.

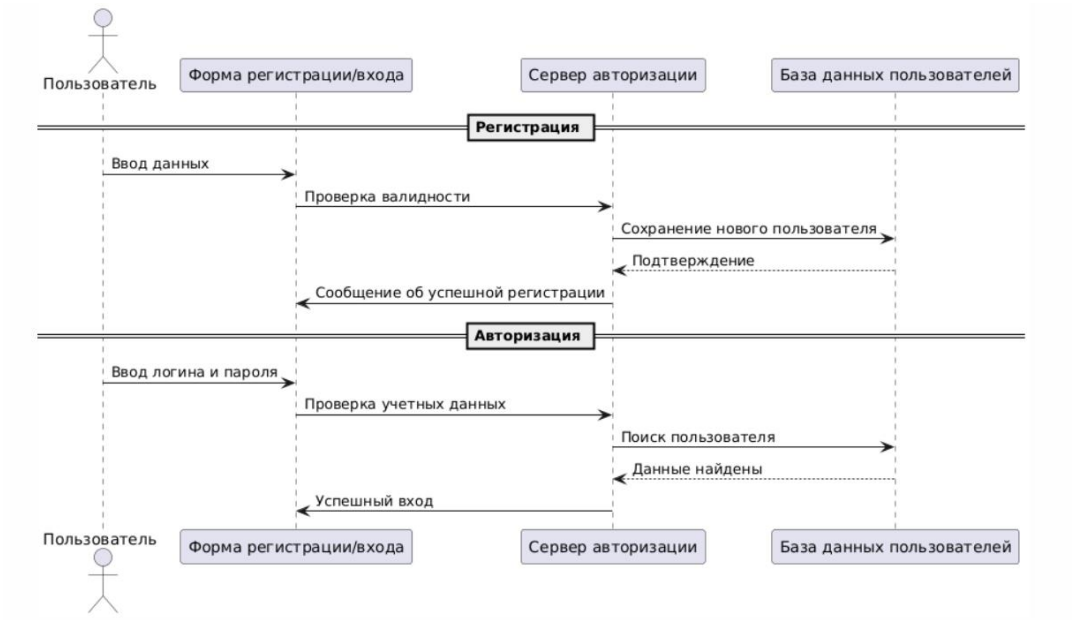

Рисунок 5 – Схема регистрации и авторизации пользователя

Такой подход обеспечивает безопасность, корректную идентификацию пользователей и беспрепятственный доступ к функционалу после успешного входа.

Чтобы дополнительно повысить качество интерфейса, важно проводить тестирование удобства использования, что и продемонстрировано на рисунке 6.

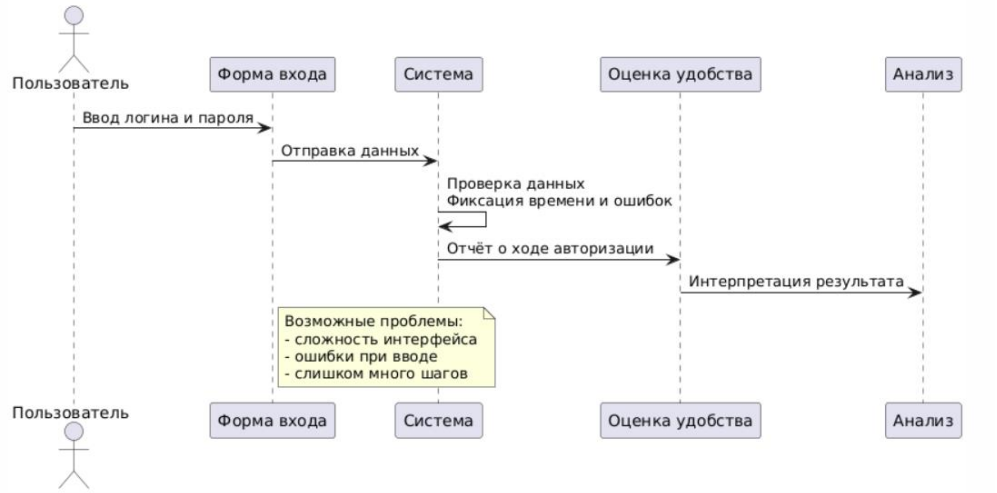

Рисунок 6 – Схема процесса тестирования удобства

Диаграмма демонстрирует сценарий юзабилити-тестирования на примере авторизации. Пользователь вводит данные в форму входа, система фиксирует ошибки и время выполнения. Эти данные поступают на этап оценки удобства, где проводится анализ поведения пользователя.

После описания функциональных требований, перейдём к нефункциональным, которые характеризуют качество и поведение системы в целом. Эти требования задают рамки для архитектуры, производительности, безопасности и удобства использования приложения. К ним относятся:

- архитектура: клиент-серверная модель:
- режим функционирования: круглосуточная работа (24/7);
- минимальное количество одновременных пользователей: не менее 5;
- масштабируемость: возможность добавления новых функций и расширения числа пользователей;
- уровень безопасности: реализация механизмов аутентификации, авторизации и защиты данных;
- удобство интерфейса: интуитивная структура, понятная навигация и адаптивный дизайн.

Для наглядности классификации требований применим модель FURPS+ [2], которая объединяет основные категории требований, влияющих на качество ПО. В таблице 3 приведены основные категории и описание их содержания.

Таблица 3 – Классификация требований к системе по модели FURPS+

Категория	Описание
Functionality	Учёт рабочего времени, отчётность, управление доступом
Usability	Простой и понятный интерфейс, минимум кликов для задач
Reliability	Надёжность хранения данных, отсутствие сбоев
Performance	Быстрый отклик интерфейса, обработка запросов менее 2 сек
Supportability	Возможность обновлений, сопровождения, логирование ошибок
+Security, Scalability	Шифрование данных, возможность увеличения нагрузки и добавления модулей

Чтобы наглядно показать, кто и как будет пользоваться разрабатываемой системой, была создана диаграмма вариантов использования. Она помогает понять, какие действия может выполнять каждый тип пользователя и как эти действия связаны между собой. Такая диаграмма особенно полезна на этапе проектирования, чтобы заранее продумать, что именно должна уметь система.

На диаграмме показаны три роли: разработчик, руководитель и администратор. У каждого из них есть свои функции, которые он может выполнять в системе. Также некоторые действия зависят друг от друга – это тоже отражено на рисунке 7.

Рисунок 7 – Диаграмма вариантов использования

Разработчик – это сотрудник отдела разработки. Он может отмечать время прихода и ухода с работы, просматривать список задач, а также просматривать свою статистику по отработанному времени. Когда разработчик просматривает задачи, ему становятся доступны дополнительные действия: отметка о начале работы над задачей, отметка о завершении работы,

а также создание личных задач. Все эти действия напрямую связаны с просмотром задач, так как начать или завершить задачу, а также добавить новую можно только после того, как разработчик ознакомится со списком.

Руководитель может назначать задачи своим подчинённым и формировать отчёты по их работе. Чтобы составить отчёт, ему нужно просматривать статистику работы сотрудников – это тоже указано на диаграмме.

Администратор управляет данными о сотрудниках и отделах. Он может добавлять, редактировать и удалять информацию о работниках, а также назначать роли пользователям. Все эти действия объединены в общее – «управление сотрудниками и отделами», потому что они связаны между собой.

В первой главе выпускной квалификационной работы подробно рассмотрены особенности предметной области, связанной с учетом рабочего времени сотрудников. Для подтверждения актуальности задачи проведен сравнительный анализ популярных программных решений, который показал, что они либо избыточны, либо недостаточно гибкие для адаптации под специфику предприятия. На основе изучения предметной области и анализа аналогов сформулированы функциональные и нефункциональные требования к разрабатываемой системе. К функциональным относятся: регистрация и авторизация пользователей, фиксация времени прихода и ухода, формирование отчетов, управление задачами, а также разграничение прав доступа. Нефункциональные требования включают надежность, производительность, безопасность, масштабируемость и удобство интерфейса. Для систематизации требований к разрабатываемой системе была использована модель FURPS+. Разработка собственного программного обеспечения признана целесообразной, учитывая уникальные потребности организации и стремление повысить эффективность управления персоналом.

Глава 2 Процесс разработки информационной системы

2.1 Проектирование программного обеспечения

При изучении архитектуры автоматизированных информационных систем (АИС) можно выделить ряд основных типов [3], каждый из которых обладает своими уникальными характеристиками и областями применения:

Файл-серверная архитектура – данный тип подразумевает наличие одного компьютера, который отвечает за обработку информации и взаимодействие с пользователями [1]. Все данные хранятся на этом сервере, а клиенты обращаются к нему для выполнения операций. Данный подход обеспечивает простоту реализации, но может ограничивать производительность и масштабируемость системы, особенно при увеличении числа пользователей.

Клиент-серверная архитектура представляет собой модель, в которой компоненты приложения разделены между клиентскими устройствами и сервером [1]. При этом сервер баз данных работает независимо от клиентских компьютеров, что позволяет более эффективно организовать ресурсы и улучшить процесс взаимодействия. Данная схема способствует увеличению производительности и рациональному распределению нагрузки по всей системе.

Технологии интернет/интранет – данная архитектура используется для веб-приложений и позволяет пользователям получать доступ к системе через интернет-браузеры, что обеспечивает высокую степень доступности и гибкости.

В контексте текущей работы выбрана клиент-серверная архитектура (рисунок 8). Данная архитектура часто используется при создании информационных систем, однако, как и любая другая архитектурная модель, она имеет некоторые незначительные недостатки [22]. Во-первых, с увеличением числа пользователей могут возникнуть сложности с обновлением

данных, что может привести к их несогласованности. Во-вторых, каждый клиент имеет возможность устанавливать соединение с системой управления базами данных (СУБД) без учета действий других пользователей, что может вызвать проблемы с одновременным доступом к данным.

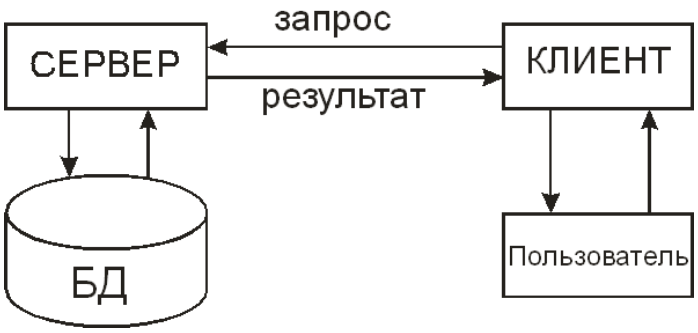

Рисунок 8 – Архитектура клиент-сервер

Для более наглядного понимания архитектуры клиент-серверной системы и взаимодействия между модулями, на рисунке 9 представлена диаграмма компонентов, демонстрирующая основные части приложения и связи между ними.

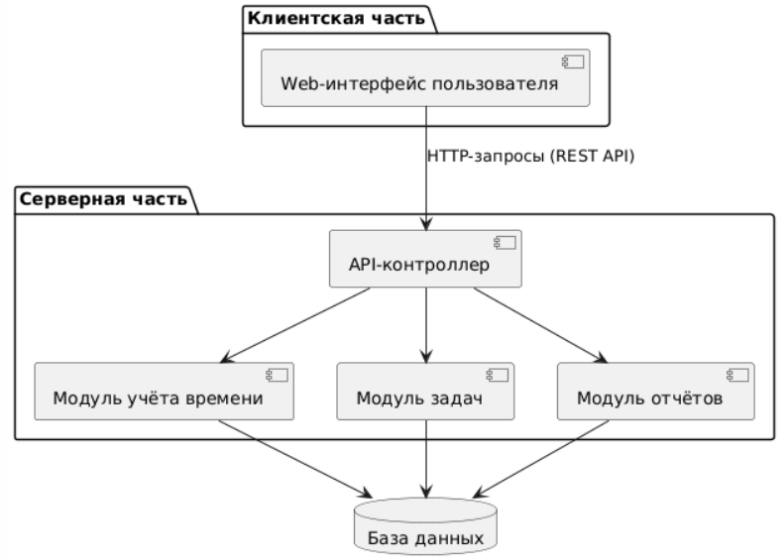

Рисунок 9 – Диаграмма компонентов

На диаграмме показана структура взаимодействия компонентов в рамках клиент-серверной архитектуры. Пользовательский интерфейс взаимодействует с сервером через HTTP-запросы, используя REST API. API-контроллер выполняет роль промежуточного слоя, распределяя входящие запросы к соответствующим модулям. Каждый серверный модуль взаимодействует с общей базой данных, выполняя операции чтения и записи информации, связанной со своей областью ответственности. Подобные подходы к описанию и структурированию компонентов системы часто реализуются в методологии IDEF, которая широко применяется для моделирования архитектурных и функциональных моделей [24].

Для наглядного отображения того, как устроена разработанная информационная система на физическом уровне, была создана диаграмма развертывания (рисунок 10). Она показывает, где именно размещены основные части системы и как они взаимодействуют между собой.

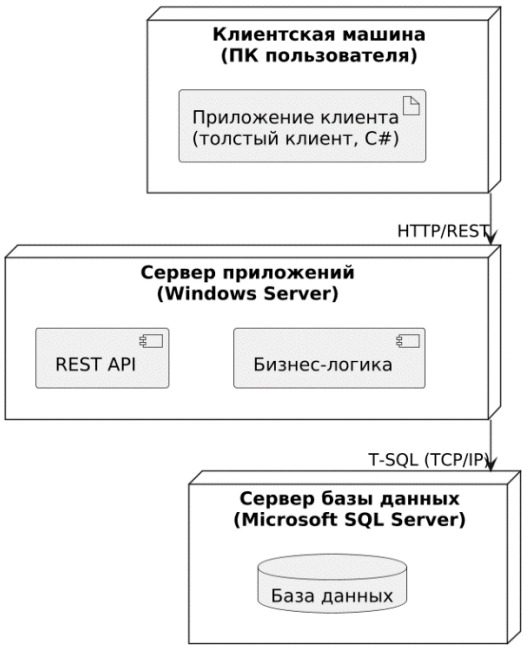

Рисунок 10 – Диаграмма развертывания компонентов информационной системы

В системе участвуют три основные части. Первая – это компьютер

сотрудника, на котором установлено клиентское приложение. Пользователь запускает это приложение и через него фиксирует время прихода, ухода, просматривает задачи и формирует отчёты. Приложение написано на языке C# и устанавливается как обычная программа.

Вторая часть – это сервер приложений. Он находится в сети предприятия и отвечает за обработку всех запросов от клиентского приложения. Именно здесь работает вся логика – проверка данных, обработка времени, работа с задачами и отчётами [19]. Клиент подключается к этому серверу через локальную сеть, а запросы передаются по протоколу HTTP с использованием REST API.

Третья часть – это сервер базы данных. Он хранит всю информацию: данные о сотрудниках, задачах, отметках времени, отделах и так далее. Сервер приложений подключается к базе данных и получает оттуда нужную информацию или записывает новую.

Для создания структуры базы данных была применена концепция «Сущность-связь», представляющая собой методику, которая обеспечивает четкое и наглядное отображение основных объектов (сущностей), их характеристик (атрибутов) и взаимных отношений. Это существенно упрощает как первоначальное проектирование, так и последующую модернизацию базы данных.

В реляционной модели часто используется связь один-ко-многим, при которой один экземпляр родительской сущности может быть связан с несколькими экземплярами дочерней сущности. Например, связь между сотрудниками и задачами иллюстрирует ситуацию, когда один сотрудник может выполнять множество задач (рисунок 11).

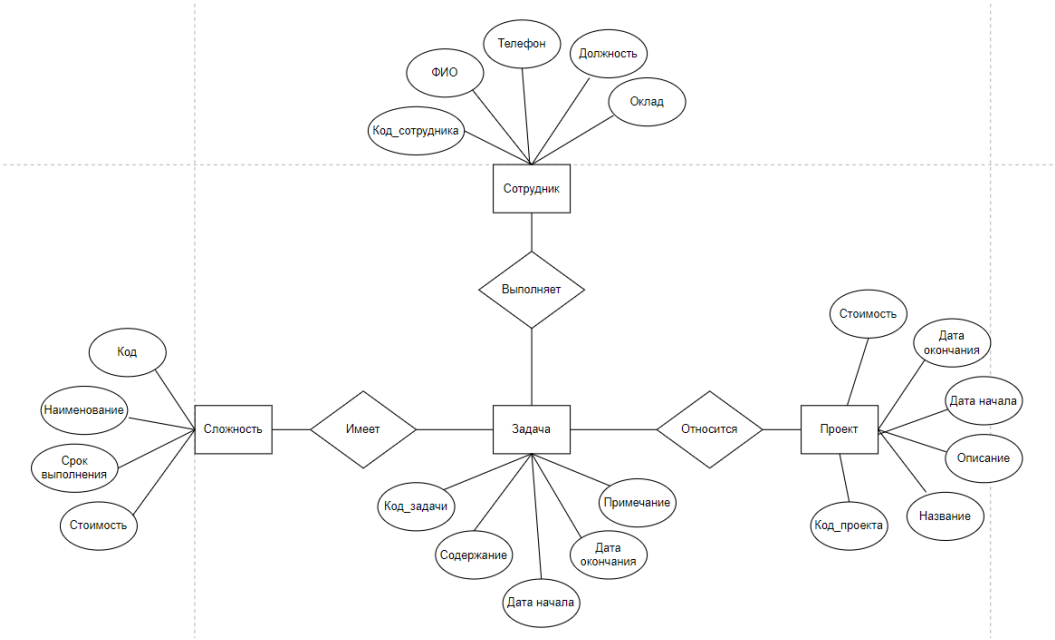

Рисунок 11 – ER-диаграмма

Структурные связи, которые описывают взаимодействие между сущностями на диаграмме представлены следующим образом:

«Сотрудник» – «Задача» – связь типа один-ко-многим (1:M). Это означает, что один сотрудник может быть задействован в выполнении нескольких различных задач.

«Сложность» – «Задача» – также связь один-ко-многим (1:M). Несколько задач могут иметь один и тот же уровень сложности.

«Проект» – «Задача» – связь один-ко-многим (1:M). В рамках одного проекта может быть реализовано множество задач.

Проектирование структуры базы данных осуществлялось на основе анализа функциональных требований системы. В результате проектирования были определены следующие таблицы: DEPARTMENTS – для хранения сведений об отделах организации, EMPLOYEES – для информации о сотрудниках, MAKES – для классификации задач по типам, TASKS – для хранения задач, назначенных сотрудникам, и WORK_TIMES – для учёта времени, затраченного на выполнение работы [16].

База данных для информационной системы была создана с помощью Microsoft SQL Server 2019. Для работы с ней использовалась среда Microsoft SQL Server Management Studio (SSMS), версия 18.12.1. Эта программа позволяет удобно управлять таблицами, выполнять SQL-запросы и контролировать работу базы данных [17]. Физическая структура базы данных представлена на рисунке 12.

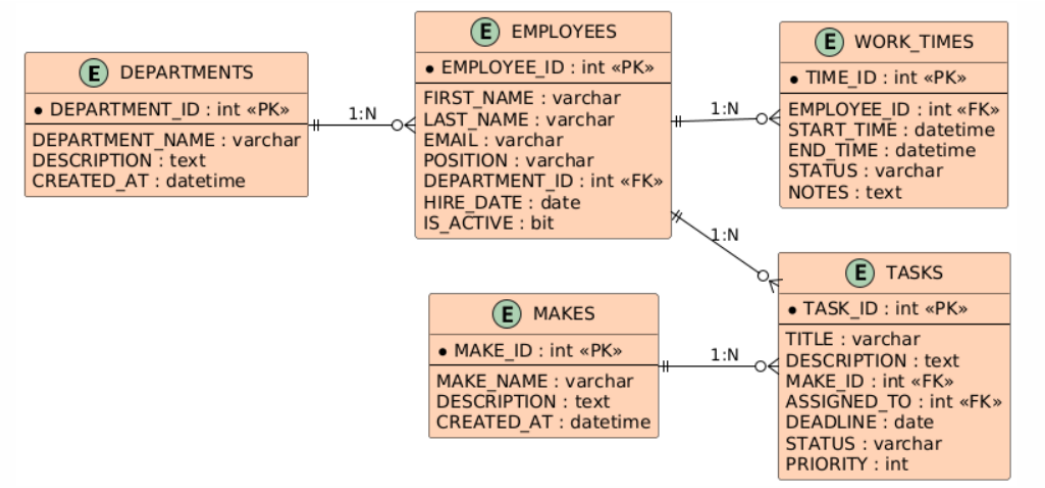

Рисунок 12 – Физическая модель базы данных

При создании физической модели использовалась стандартная реляционная структура [18]. Каждая из таблиц содержит первичный ключ, обеспечивающий уникальность записей, а также атрибуты, соответствующие требованиям информационной системы. Таблица EMPLOYEES, например, включает в себя идентификатор сотрудника, имя, фамилию, адрес электронной почты, должность, ссылку на отдел, дату приёма на работу и флаг активности. Таблица TASKS содержит такие поля, как заголовок задачи, описание, категория, исполнитель, крайний срок, статус и приоритет. В таблице WORK_TIMES регистрируются идентификатор сотрудника, время начала и окончания работы, статус и дополнительные заметки.

Связи между таблицами реализованы посредством внешних ключей. Так, каждый сотрудник связан с определённым отделом через внешний ключ

DEPARTMENT_ID. Один отдел может включать множество сотрудников, что реализует связь «один ко многим». Аналогичным образом, один сотрудник может быть связан с множеством записей о времени работы и с множеством задач, назначенных ему. Каждая задача также относится к определённой категории, хранящейся в таблице MAKES.

Важной особенностью является то, что структура базы данных продумана таким образом, чтобы в будущем её можно было расширять — добавлять новые таблицы, поля и связи без нарушения уже работающей системы. Это делает базу надёжной основой для развития информационной системы.

Бизнес-логика является центральным элементом информационной системы, обеспечивающим реализацию основных функций и правил, необходимых для автоматизации учета рабочего времени. На этапе проектирования бизнес-логики формируются четкие алгоритмы обработки данных и взаимодействия компонентов системы, которые позволяют корректно и эффективно решать поставленные задачи.

В рамках данной работы ключевыми бизнес-процессами являются учет времени прихода и ухода сотрудников, регистрация и контроль выполнения задач, а также формирование отчетов для руководства. Правильное моделирование этих процессов обеспечивает надежность, прозрачность и удобство использования системы [33].

Для визуализации и детального анализа бизнес-процессов применяются диаграммы UML [21], в частности диаграммы последовательности и диаграммы активности, которые позволяют наглядно представить потоки данных и взаимодействия между элементами системы [12].

Процесс регистрации времени прихода сотрудника представлен на рисунке 13 в виде диаграммы последовательности, отображающей взаимодействие между пользователем, интерфейсом, системой и базой данных.

Рисунок 13 – Диаграмма последовательности «Регистрация времени прихода сотрудника»

Диаграмма отображает взаимодействие между сотрудником, интерфейсом, системой и базой данных при регистрации времени прихода. Она иллюстрирует проверки на наличие уже зарегистрированного времени и обработку возможных ошибок.

Логика обработки этого процесса в виде потока действий представлена на рисунке 14, где показана диаграмма активности учета времени прихода сотрудника.

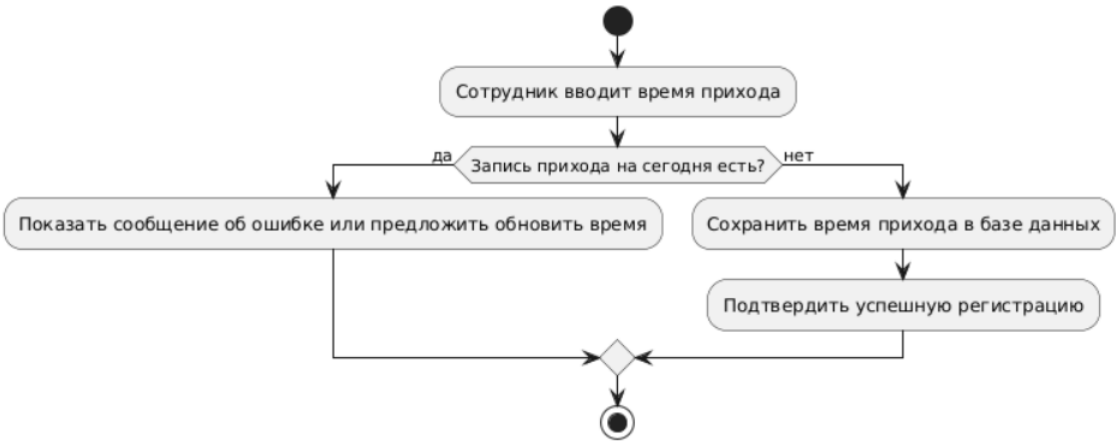

Рисунок 14 – Диаграмма активности «Учет времени прихода сотрудника»

На диаграмме показано, что система выполняет проверку на наличие уже зарегистрированного времени прихода за текущий день. Это критический этап, предотвращающий дублирование записей и возможные ошибки учета, которые могут привести к искажению статистики рабочего времени.

Если время прихода уже зарегистрировано, система уведомляет пользователя об ошибке или предлагает обновить существующую запись, что гарантирует гибкость и возможность исправления данных. В случае отсутствия записи время успешно сохраняется, и сотрудник получает подтверждение.

Рассмотрим следующий бизнес-процесс, который важен для контроля и анализа эффективности работы персонала и принятия управленческих решений.

Руководитель системы может формировать итоговые отчеты по работе сотрудников и в разрезе отделов. Для этого он выбирает необходимые параметры отчета (период, отдел, сотрудник и др.), после чего система собирает данные об отработанном времени и выполненных задачах, агрегирует информацию и отображает готовый отчет.

Процесс формирования отчета руководителем представлен на рисунке 15 в виде диаграммы последовательности, иллюстрирующей этапы от задания параметров до получения готового отчета.

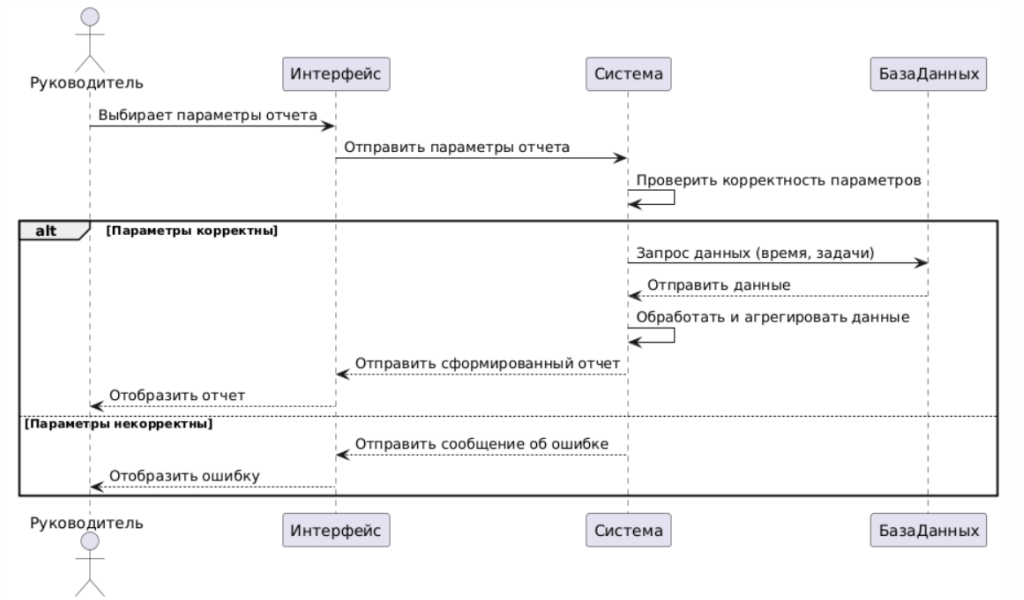

Рисунок 15 – Диаграмма последовательности «Формирование отчета руководителем»

Данная диаграмма позволяет детально проследить, как осуществляется формирование отчета руководителем – начиная с выбора параметров и заканчивая получением готового результата.

Подробный поток действий в рамках этого процесса показан на рисунке 16, где представлена диаграмма активности.

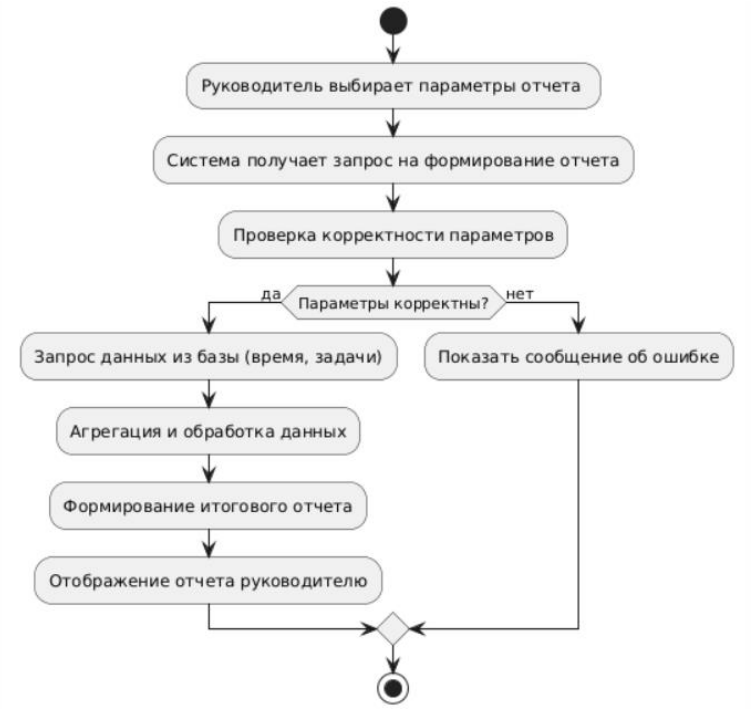

Рисунок 16 – Диаграмма активности «Формирование отчета руководителем»

Диаграмма активности процесса «Формирование отчета руководителем» показывает последовательность действий, выполняемых при генерации отчета. Процесс начинается с выбора руководителем параметров отчета, после чего система проверяет их корректность, запрашивает и обрабатывает необходимые данные из базы, а затем формирует и отображает итоговый отчет. В случае обнаружения ошибок пользователь получает уведомление с возможностью их исправления. Диаграмма наглядно иллюстрирует поток управления в процессе и демонстрирует логику работы системы на уровне бизнес-процесса.

На этапе проектирования взаимодействия определяются способы, с помощью которых компоненты информационной системы обмениваются данными между собой, а также с внешними сервисами (при наличии таковых). Это позволяет обеспечить чёткую архитектурную структуру, согласованность компонентов и масштабируемость решения.

В рамках проектирования взаимодействия выполняются задачи по

определению API и протоколов обмена информацией между модулями, разработке схем взаимодействия и передачи данных, а также построению моделей, отражающих логические связи и процессы в системе.

Диаграмма взаимодействия отображает, как отдельные объекты или компоненты системы взаимодействуют друг с другом при выполнении определённого бизнес-процесса. В отличие от диаграммы последовательности, здесь акцент сделан не на времени, а на структуре связей и маршрутах обмена сообщениями.

На рисунке 17 представлена диаграмма взаимодействия, отображающая, как отдельные объекты или компоненты системы обмениваются сообщениями при выполнении бизнес-процесса формирования отчета руководителем.

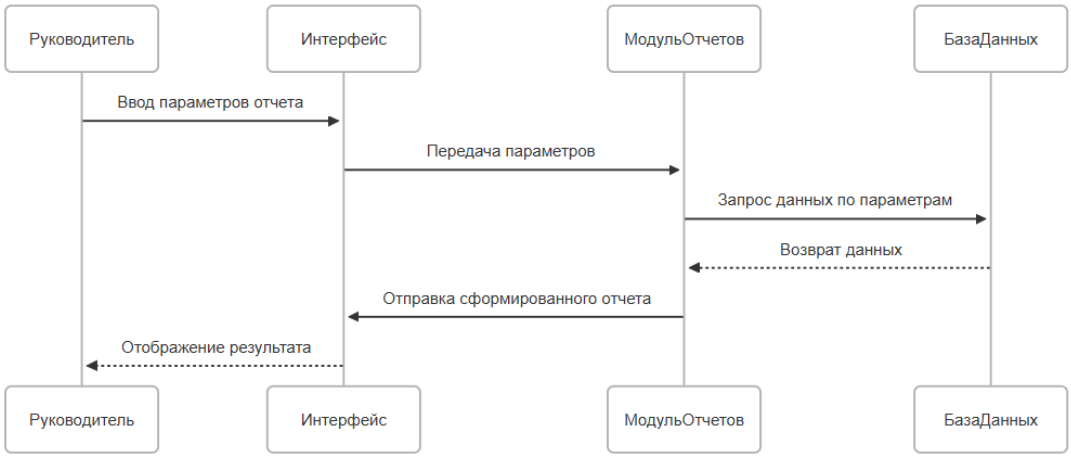

Рисунок 17 – Диаграмма взаимодействия

На данной диаграмме, отражающей процесс формирования отчета руководителем, представлены четыре участника: руководитель, интерфейс пользователя, отчетный модуль (логика формирования) и база данных.

Процесс включает следующие шаги. Руководитель вводит параметры для формирования отчета, после чего интерфейс передает эти параметры в отчетный модуль. Отчетный модуль обращается к базе данных для получения необходимой информации. После обработки данных результат возвращается

через интерфейс руководителю.

Следующим этапом была разработана диаграмма классов (рисунок 18). Она позволяет сформировать логическую модель предметной области и определить структуру данных, лежащую в основе программной реализации.

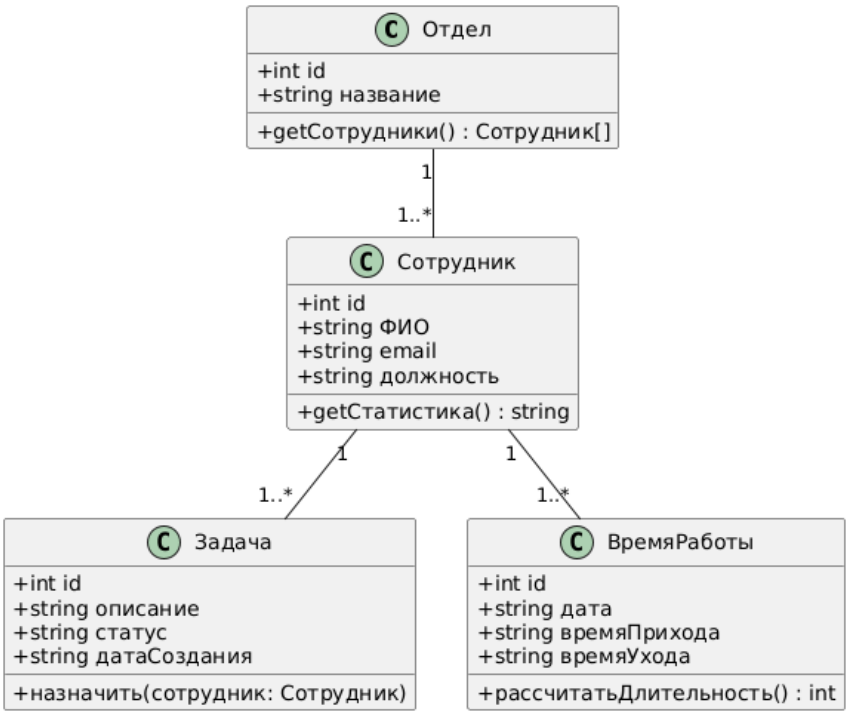

Рисунок 18 – Диаграмма классов

В представленной диаграмме выделены ключевые классы: «Сотрудник», который содержит личные данные и связан с задачами и временем работы; «Отдел», объединяющий сотрудников по организационной структуре; «Задача», представляющая назначенные задания, которые могут быть привязаны к сотруднику и «Время Работы», содержащая информацию о приходе и уходе сотрудника, а также длительности работы.

Связи между классами показывают, что один сотрудник может быть связан с несколькими задачами и записями учета времени, а один отдел включает нескольких сотрудников.

2.2 Выбор технологий и инструментов для разработки

При создании информационной системы для управления рабочим временем сотрудников особое внимание было уделено выбору инструментальных средств разработки. Эффективность, производительность, удобство использования и возможности расширения системы во многом зависят от выбранных технологий.

При выборе языка программирования в первую очередь учитывались такие критерии, как производительность, удобство разработки, поддержка сообщества и экосистемы, возможность интеграции с корпоративными решениями, строгость типизации и масштабируемость приложений.

В таблице 4 показано сравнительное соответствие языков программирования ключевым требованиям:

Таблица 4 – Сравнение языков программирования

Критерии оценки	C#	Java	Python
Производительность	+	+	−
Удобство разработки	+	±	+
Поддержка и экосистема	+	+	+
Интеграция с MS стеком	+	−	−
Строгая типизация	+	+	−
Масштабируемость	+	+	±

C# выделяется высокой производительностью [9] благодаря компиляции в промежуточный язык IL (Intermediate Language) с последующей оптимизацией JIT-компилятором .NET CLR. Это обеспечивает баланс между скоростью выполнения и гибкостью. Строгая типизация способствует снижению числа ошибок и упрощает сопровождение кода в больших командах.

Кроме того, C# имеет богатый синтаксис, строгую типизацию и развитую экосистему библиотек для корпоративной разработки [14], что упрощает реализацию сложной бизнес-логики. Интеграция с Microsoft Visual

Studio 2022 и экосистемой .NET дает доступ к мощным инструментам отладки, профилирования и тестирования [8].

Java также обладает высокой производительностью и масштабируемостью, широко используется в корпоративной среде, но уступает по интеграции с Microsoft технологиями, что важно для нашего проекта. Python удобен для быстрой разработки и прототипирования, однако ограничены в плане производительности и строгой типизации, что может привести к ошибкам в сложных системах.

На основе анализа преимуществ и недостатков выбран язык программирования C#, который оптимально удовлетворяет требованиям корпоративной среды, высокой производительности и поддерживаемой архитектуры.

Данные в информационной системе – это ключевой ресурс, поэтому выбор СУБД является стратегическим решением, влияющим на надежность, безопасность, масштабируемость и удобство эксплуатации системы. Для данного проекта рассматриваются следующие СУБД: Microsoft SQL Server, PostgreSQL, MySQL и SQLite.

Таблица 5 отражает сравнительный анализ СУБД по указанным параметрам.

Таблица 5 – Сравнение систем управления базами данных

Критерии оценки	Microsoft SQL Server	PostgreSQL	SQLite
Производительность	+	+	–
Надежность	+	+	–
Безопасность	+	+	–
Интеграция с MS стеком	+	–	–
Стоимость	–	+	+
Масштабируемость	+	+	–

Microsoft SQL Server обеспечивает продвинутые функции управления транзакциями, репликации и резервного копирования, что критически важно для бизнес-приложений, работающих с чувствительными данными [29].

Поддержка T-SQL позволяет создавать сложные хранимые процедуры и триггеры для автоматизации обработки данных [5].

Его тесная интеграция с .NET Framework и Visual Studio значительно упрощает разработку, а средства мониторинга и диагностики помогают поддерживать высокую производительность и надежность. Основным недостатком является стоимость лицензирования, однако для корпоративных решений преимущества перевешивают этот фактор.

PostgreSQL и MySQL – мощные и надежные СУБД с открытым исходным кодом, широко используемые в различных сферах, однако их интеграция с Microsoft экосистемой ограничена, что усложняет разработку и сопровождение единой системы [13].

Выбор Microsoft SQL Server обусловлен необходимостью обеспечить высокую производительность, безопасность, отказоустойчивость и легкую интеграцию с другими компонентами системы.

2.3 Реализация функциональных требований

Архитектура программной системы построена по принципу клиент-серверной модели, которая является одной из самых распространённых архитектурных схем в современных информационных системах. Основная идея данной архитектуры заключается в разделении приложения на две независимые, но взаимодействующие части: клиентскую и серверную. Такое разделение обеспечивает высокую гибкость, масштабируемость и надёжность при разработке и эксплуатации программного продукта. На рисунке 19 представлена архитектурная схема компонентов разрабатываемой информационной системы учета рабочего времени с указанием взаимодействий и технологий.

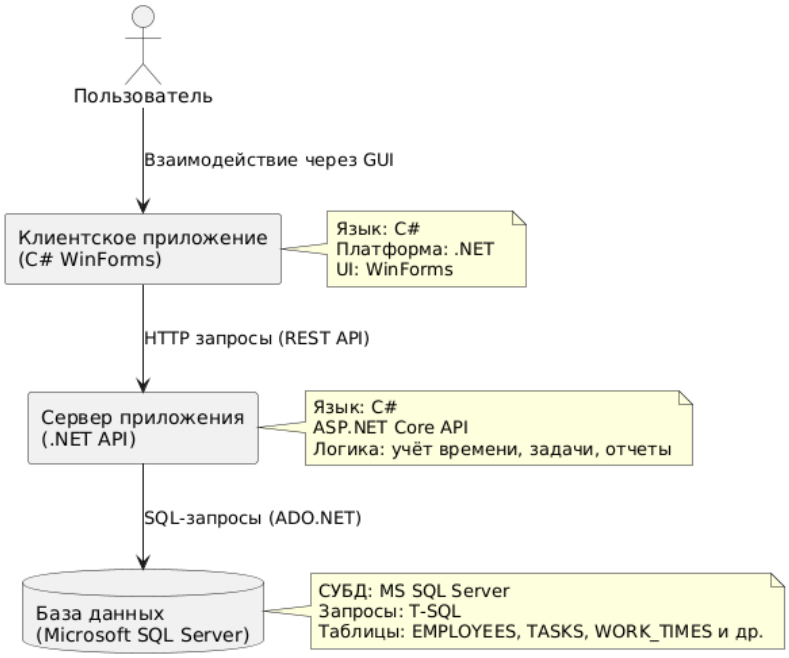

Рисунок 19 – Архитектурная схема компонентов информационной системы

Клиентская часть представляет собой пользовательский интерфейс, она отвечает за взаимодействие с пользователем, отправку запросов на сервер и отображение полученных данных в удобной форме. Серверная часть содержит всю бизнес-логику, обрабатывает запросы от клиента, выполняет операции с базой данных и формирует ответы.

Сервер реализует набор функциональных модулей, каждый из которых решает строго определённую задачу в рамках общей бизнес-логики. Модуль учёта времени обеспечивает регистрацию приходов и уходов сотрудников, хранение временных меток и контроль корректности данных. Модуль задач позволяет сотрудникам регистрировать личные задачи, а руководителям – назначать служебные поручения. Модуль отчётов отвечает за агрегирование информации, её анализ и формирование отчетной документации по отделам и отдельным сотрудникам. Все модули работают с общей базой данных, что обеспечивает консистентность и целостность информации.

После запуска приложения пользователь попадает на главную страницу,

где отображается форма для авторизации. Чтобы начать использовать приложение, необходимо пройти процедуру аутентификации. После успешного входа пользователю откроется доступ к функционалу, соответствующему его роли в системе. Система автоматически распознает роль пользователя при входе, поэтому указание роли не требуется. Роль пользователя назначается администратором в момент создания учетной записи. Форма назначения ролей пользователю представлена на рисунке 21.

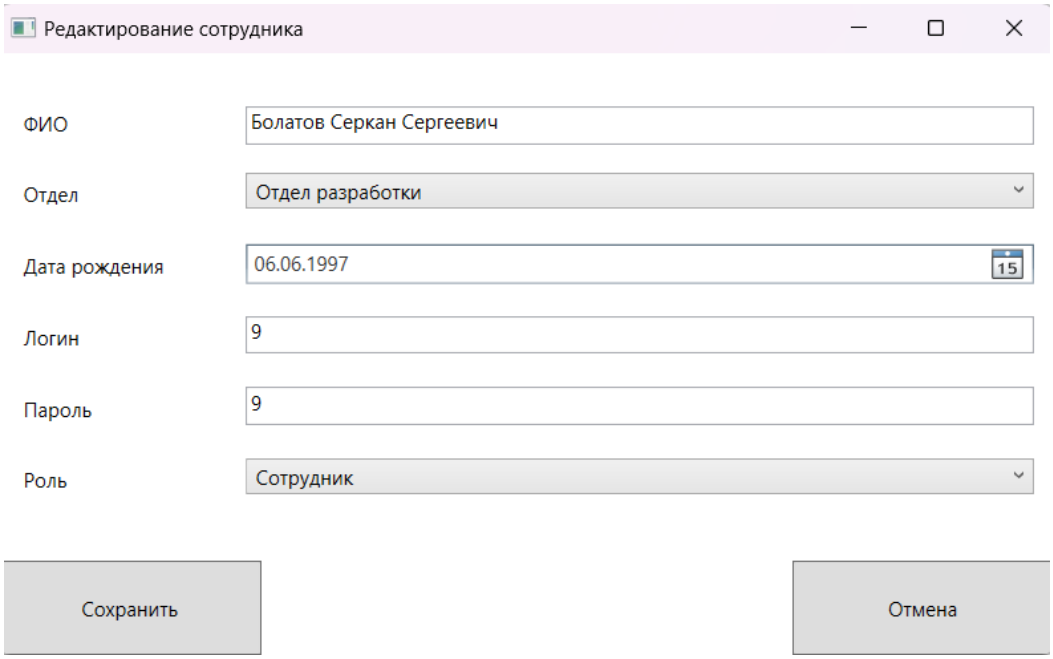

Рисунок 21 – Форма назначения ролей пользователю

На рисунке 22 представлена форма авторизации, реализованная в приложении. Пользователь вводит логин и пароль, после чего может выполнить вход или завершить работу программы.

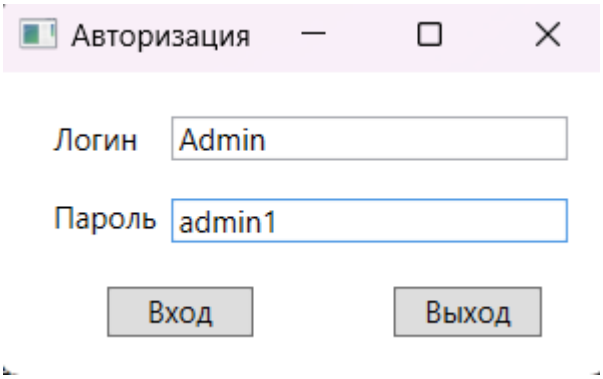

Рисунок 22 – Форма авторизации пользователя

После успешного входа в систему пользователя перенаправят на главную страницу, которая будет адаптирована в зависимости от его роли в этой системе. Изучим интерфейс для пользователей, имеющих роль «Руководитель» (рисунок 23).

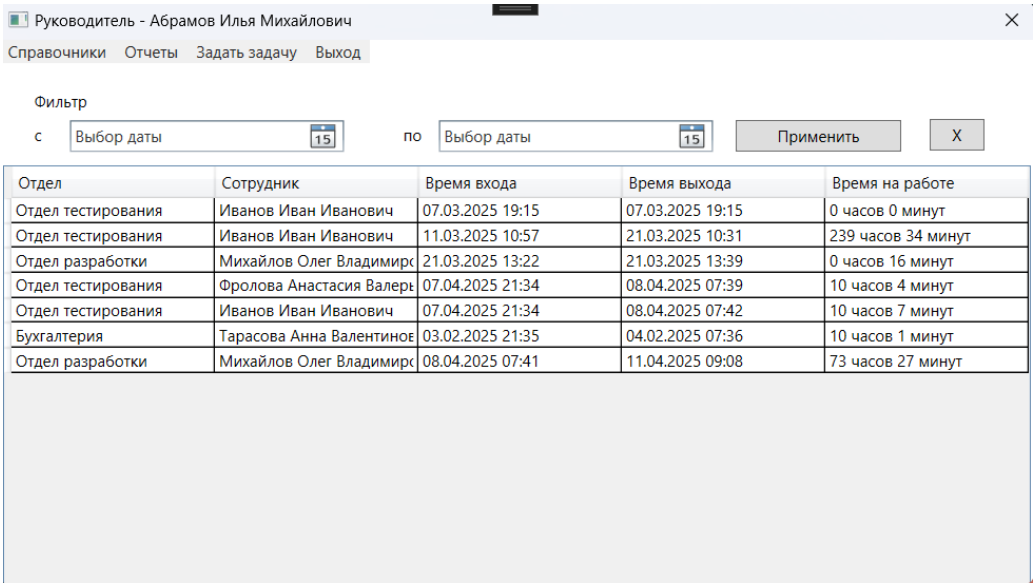

Рисунок 23 – Форма электронного отчета рабочего времени сотрудников

После успешной авторизации они будут перенаправлены на главную страницу, которая представляет собой форму для электронного отчета о рабочем времени сотрудников. На данной странице руководителю доступна

сводная информация по каждому сотруднику.

С помощью формы «Обработка данных о задаче» руководитель может делегировать задачи конкретным сотрудникам и установить сроки их выполнения (рисунок 24).

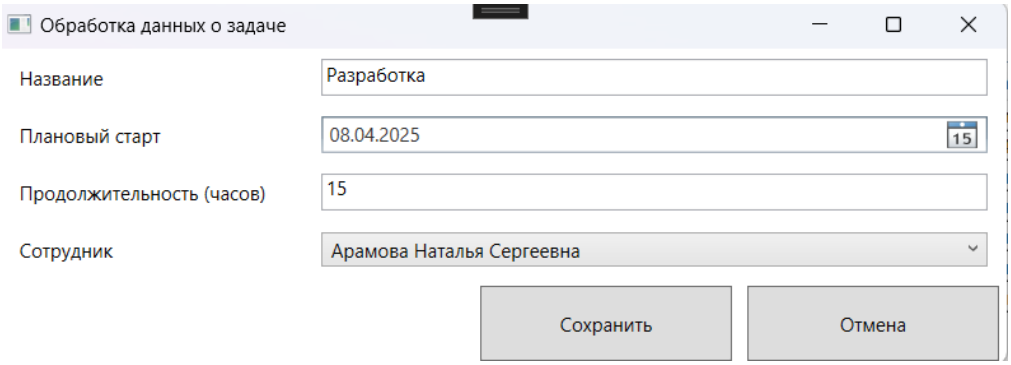

Рисунок 24 – Форма создания задачи

Все назначенные задания автоматически появляются в личном кабинете сотрудника, что обеспечивает мгновенное информирование и исключает необходимость использования дополнительных средств коммуникации. Это существенно повышает оперативность управления и снижает вероятность организационных сбоев.

Кроме того, руководитель имеет доступ к детализированной статистике по отделам, что позволяет отслеживать общую выработку сотрудников, анализировать распределение нагрузки и выявлять слабые звенья в производственном процессе. Для удобства хранения, обмена и дальнейшего анализа данные могут быть экспортированы в формате XML, что обеспечивает совместимость с внешними системами отчетности и аналитики (см. рисунок 25 и рисунок 26).

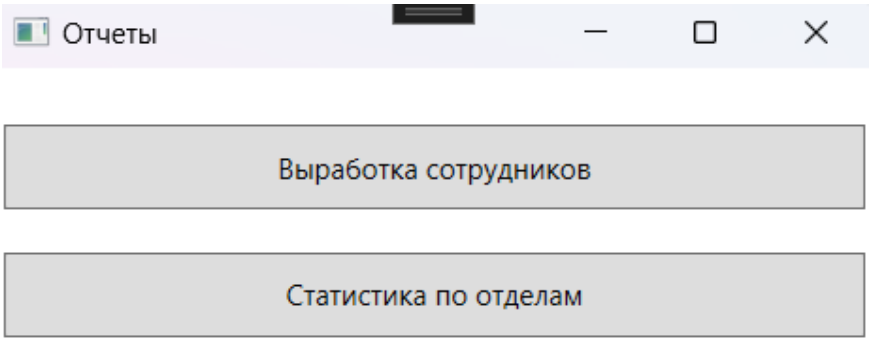

Рисунок 25 – Форма «Отчеты»

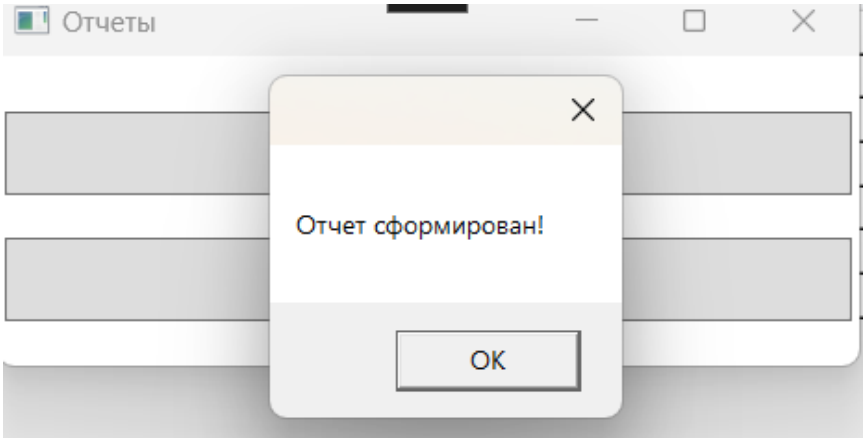

Рисунок 26 – Окно уведомления об успешном формировании отчета

На рисунке 27 представлен интерфейс главной страницы пользователя с ролью «Сотрудник».

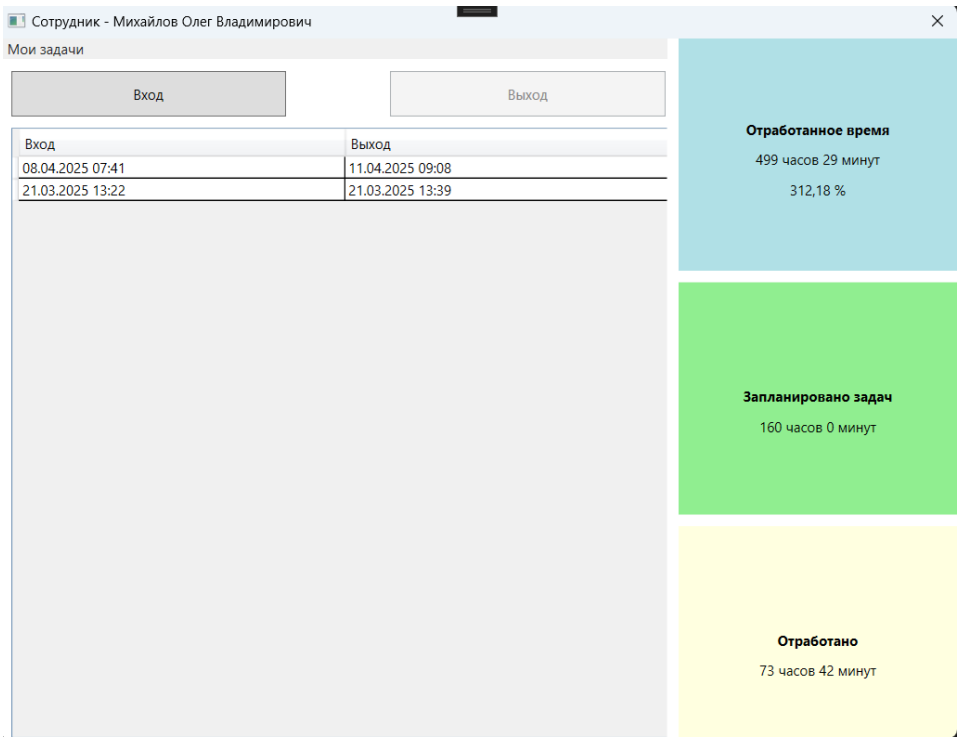

Рисунок 27 – Главная страница пользователя «Сотрудник»

В левой части окна отображается таблица с фиксацией времени входа и выхода, что позволяет отслеживать посещаемость и длительность рабочего времени. В правом блоке отображаются сводные данные: общее количество отработанного времени, суммарное запланированное время задач и фактически отработанное время. Отработанное время показывает, сколько часов сотрудник находился на рабочем месте. Запланировано задач – это сумма часов, предусмотренных для выполнения поставленных задач. Показатель «отработано» отражает, сколько времени сотрудник реально затратил на выполнение задач. Цветовая индикация блоков облегчает восприятие информации. Интерфейс интуитивно понятен и предназначен для повседневного использования без необходимости.

На рисунке 28 представлен фрагмент программного кода, реализующий фиксацию времени прихода и ухода сотрудника в системе. Данный код написан на языке C# с использованием библиотеки System.Data.SqlClient для работы с базой данных Microsoft SQL Server.

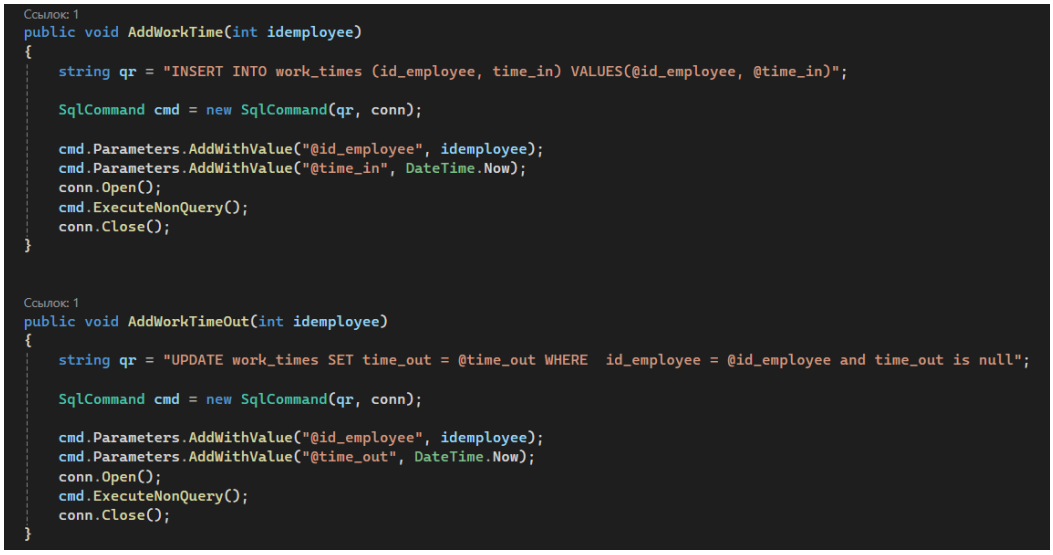

Рисунок 28 – Код фиксации времени прихода/ухода сотрудника

Метод AddWorkTime вызывается в момент, когда сотрудник отмечает начало рабочего дня. Он формирует SQL-запрос на добавление новой записи в таблицу work_times [28], указывая идентификатор сотрудника и текущее время (DateTime.Now) как значение поля time_in.

Метод AddWorkTimeOut, в свою очередь, вызывается при завершении рабочего дня. Он обновляет существующую запись, соответствующую сотруднику, и устанавливает время ухода (time_out), также используя текущее время. При этом условием обновления является наличие записи с time_out IS NULL, чтобы избежать перезаписи ранее зафиксированных данных.

В обоих методах используется параметризация запросов через SqlCommand и Parameters.AddWithValue, что снижает риск SQL-инъекций и повышает безопасность взаимодействия с базой данных [30]. Также соблюдается порядок открытия и закрытия соединения с СУБД.

Следующим элементом интерфейса системы является форма отображения задач, представленная на рисунке 29, которая предназначена для повседневной работы сотрудника.

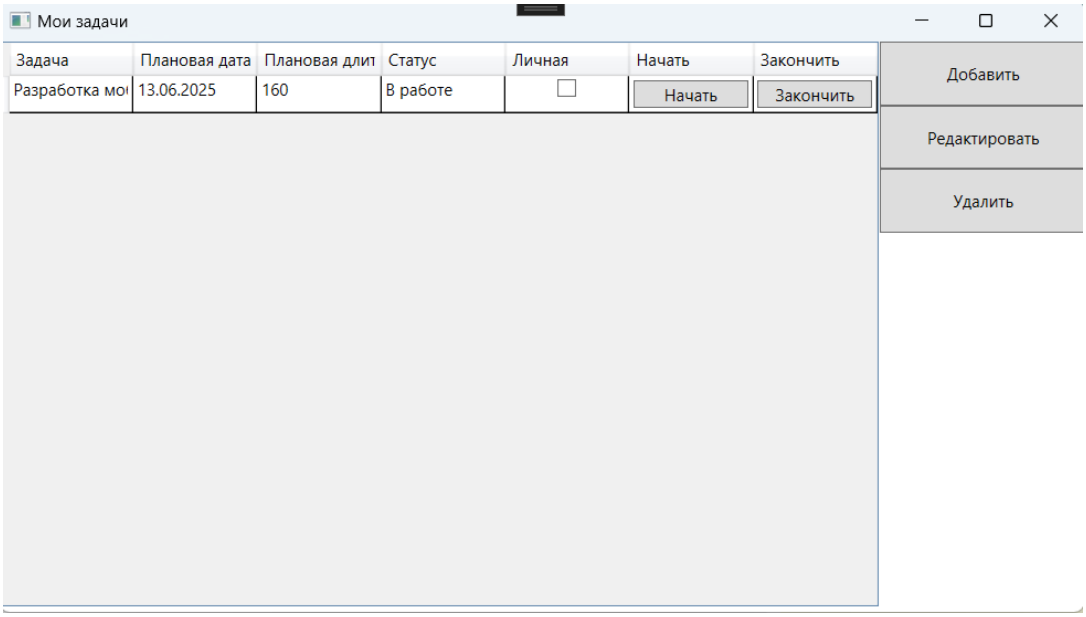

Рисунок 29 – Форма «Мои задачи»

Сотрудник имеет возможность просматривать перечень задач, доступных для его выполнения, в удобной табличной форме. Интерфейс обеспечивает наглядное отображение ключевых параметров задач, таких как название, продолжительность и статус выполнения. В дополнение к просмотру списка заданий, реализованы функция добавления личных задач. Такая функциональность позволяет пользователю гибко управлять своей рабочей нагрузкой, оперативно реагировать на изменения в производственных процессах и повышать личную эффективность.

Во второй главе был детально описан процесс проектирования и реализации информационной системы. Определена архитектура системы, спроектированы физическая модель базы данных, разработана бизнес-логика и реализован пользовательский интерфейс. Обоснован выбор технологий: язык программирования C# и СУБД Microsoft SQL Server. Все модули приложения успешно интегрированы в единую систему с учетом требований надёжности, масштабируемости и удобства.

Глава 3 Оценка эффективности разработанного программного обеспечения

3.1 Тестирование и отладка приложения

Тестирование и отладка являются важнейшими этапами жизненного цикла разработки программного обеспечения. Они направлены на обеспечение корректной работы всех компонентов, повышение надёжности и качества разработанного приложения.

После реализации каждого модуля системы необходимо проводить его тестирование, чтобы убедиться в правильности функционирования и отсутствии критических ошибок.

Перед началом тестирования был составлен тест-план, в котором указаны основные цели, методы проверки и ответственные за каждый этап. Это помогло организовать процесс и убедиться, что все важные части программы были проверены. В таблице 6 приведён общий план тестирования, включающий описание этапов и распределение ролей.

Таблица 6 – Тест-план

№	Цель тестирования	Метод тестирования	Ответственный исполнитель
1	Проверить работу отдельных частей программы	Юнит-тестирование	Разработчик
2	Убедиться, что функции работают правильно	Функциональное тестирование	Разработчик / Тестировщик
3	Оценить работу при большой нагрузке	Нагрузочное тестирование	Системный администратор
4	Проверить, удобно ли пользоваться программой	Тестирование удобства (юзабилити)	Руководитель проекта
5	Оценить работу в реальных условиях	Бета-тестирование	Заказчик / Руководитель проекта

На раннем этапе разработки проводилось юнит-тестирование, сразу после реализации отдельных модулей. Основной целью этого этапа являлась

проверка корректности внутренней логики компонентов системы, таких как механизмы учёта рабочего времени, фильтрации сотрудников и авторизации пользователей.

Для выполнения тестирования использовались встроенные средства среды разработки Visual Studio и фреймворк MSTest [7], [27]. Тесты автоматически запускались при каждом коммите в систему контроля версий, что позволяло своевременно обнаруживать и устранять логические ошибки в коде, обеспечивая стабильность и надёжность функционирования каждого элемента приложения.

На рисунке 30 представлена схема процесса юнит-тестирования.

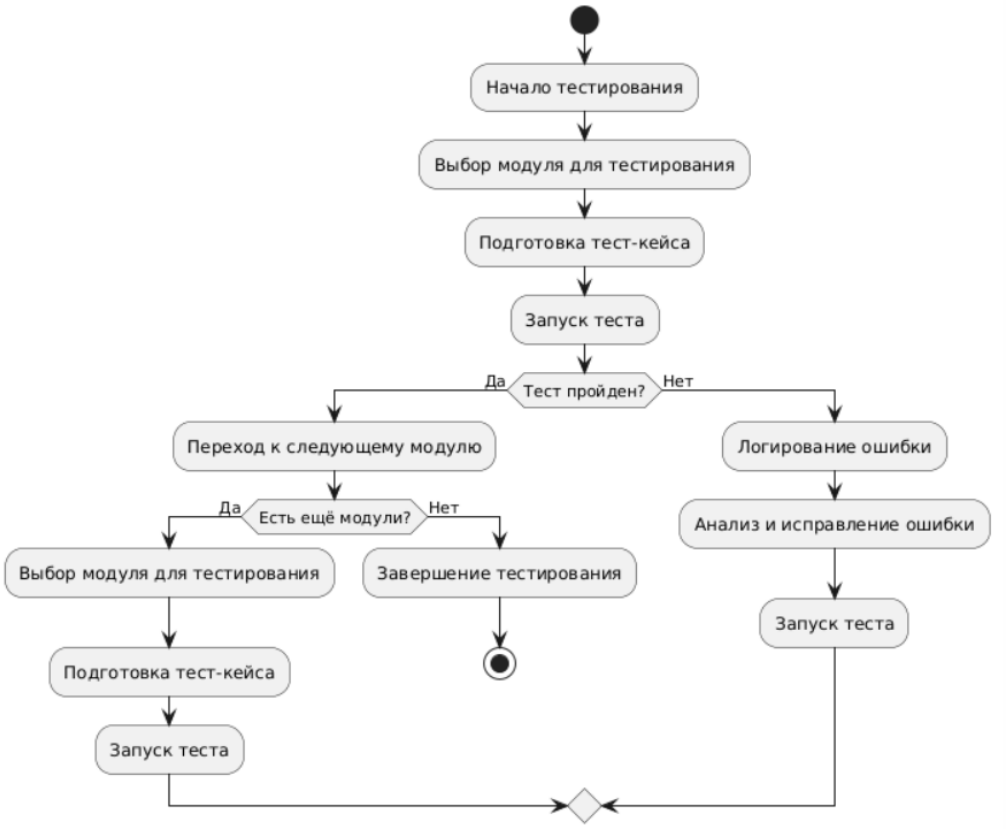

Рисунок 30 – Схема процесса юнит-тестирования

Данная диаграмма отражает этапы проведения юнит-тестирования, начиная с выбора модуля до возможной повторной отладки и перезапуска

тестов. Такой итеративный подход обеспечивает своевременное выявление ошибок в логике работы отдельных компонентов системы и минимизирует риск критических сбоев при интеграции. На рисунке 31 представлены результаты юнит-тестирования.

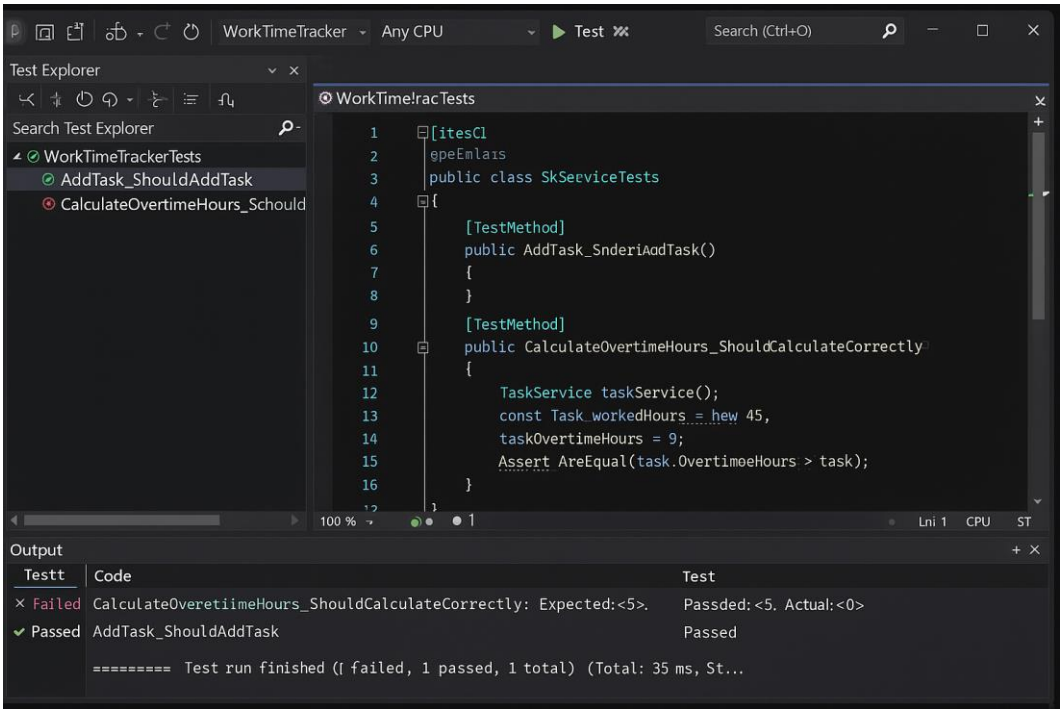

Рисунок 31 – Результаты юнит-тестов

После успешного прохождения юнит-тестов было проведено функциональное тестирование.

Функциональное тестирование позволило оценить, насколько поведение приложения соответствует его спецификации. Это наиболее прикладной уровень тестирования, так как он ориентирован на поведение системы в глазах конечного пользователя.

Были протестированы основные функции: авторизация, регистрация времени прихода и ухода, создание и редактирование задач, формирование отчётов. Для каждого теста определялись входные данные, ожидаемый результат и фактическое поведение системы, представленные в таблице 7.

Таблица 7 – Тестируемые функции и критерии успешности

№	Функция	Критерий успешности
1	Авторизация	Успешный вход при корректных данных
2	Учёт рабочего времени	Сохраняется корректное время, отображается в отчётах
3	Создание задач	Задачи присваиваются выбранному сотруднику
4	Фильтрация задач	Отображаются только задачи, удовлетворяющие фильтру
5	Генерация отчётов	Формируется PDF/Excel с нужными параметрами

Следующим этапом было нагрузочное тестирование, оно проводилось с целью проверки производительности приложения при одновременной работе большого количества пользователей. Моделировалась ситуация, при которой более 50 сотрудников одновременно осуществляют вход, регистрируют задачи и формируют отчёты. В ходе тестирования были определены ключевые метрики: среднее время отклика, частота ошибок и использование ресурсов сервера.

На основе собранных данных была построена диаграмма зависимости среднего времени отклика от количества пользователей (рисунок 32). При увеличении количества одновременных сессий до 150 наблюдался лишь незначительный рост времени отклика – с 120 до 170 миллисекунд, что указывает на стабильную работу приложения. Однако при дальнейшем увеличении нагрузки (175 и более пользователей) фиксировался резкий скачок времени отклика: до 300 мс при 175 и до 600 мс при 200 пользователях. Это свидетельствует о наступлении порога пропускной способности системы и начале деградации производительности.

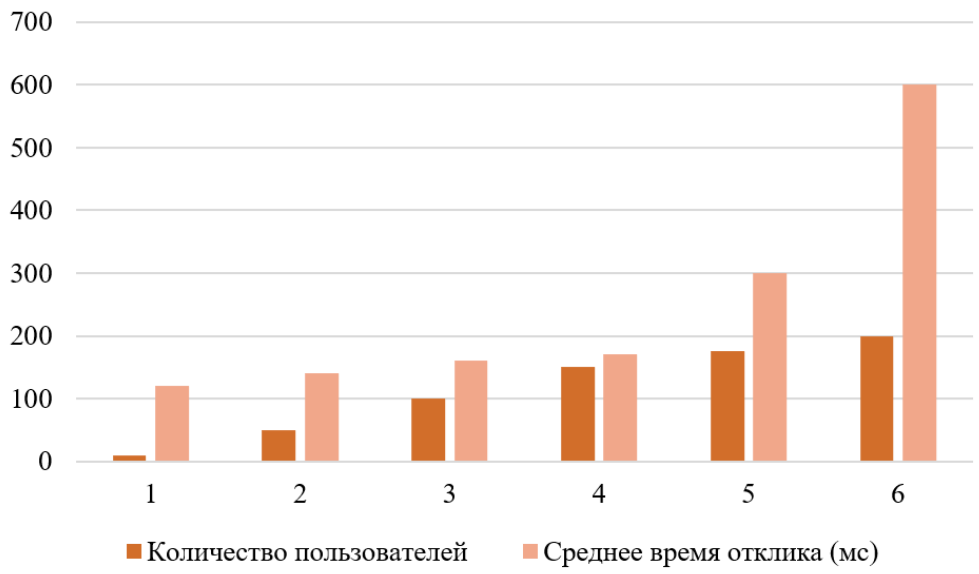

Рисунок 32 – Диаграмма зависимости среднего времени отклика от количества пользователей

Исходя из результатов, нагрузочное тестирование подтвердило, что информационная система функционирует стабильно при одновременной работе до 150 пользователей. При превышении этого значения отмечается рост задержек и снижение эффективности. Данный результат позволяет сделать вывод, что при текущей архитектуре и серверной конфигурации система готова к эксплуатации в заявленных условиях, однако при необходимости масштабирования потребуется оптимизация или модернизация инфраструктуры.

3.2 Оценка удобства и функциональности системы

Одним из ключевых критериев успешности разработанного программного обеспечения является его удобство использования (usability) и полнота реализации функциональных требований. Оценка удобства пользовательского интерфейса должна проводиться с учётом сценариев взаимодействия, времени выполнения операций и количества совершаемых ошибок [6]. В рамках данного этапа была проведена оценка пользовательского опыта и функциональности системы в реальных условиях эксплуатации.

Для оценки удобства интерфейса проводилось наблюдение за действиями тестовых пользователей, которым были выданы роли «Сотрудник» и «Руководитель». Испытуемым предлагалось выполнить ряд типовых действий: авторизоваться, зафиксировать время прихода, создать задачу, отфильтровать данные и сгенерировать отчет.

Результаты показали, что:

- интерфейс оказался интуитивно понятным для большинства пользователей без необходимости предварительного обучения;
- среднее время выполнения базового сценария (авторизация + учёт прихода) составило менее 20 секунд;
- пользователи положительно оценили структуру главного меню, логичную навигацию и быструю реакцию системы;
- для руководителей особенно удобными оказались формы создания задач и отчётности с возможностью экспорта в Excel.

Таблица 8 – Критерии оценки удобства и функциональности

№	Критерий	Описание	Весовой коэффициент
1	Интуитивность интерфейса	Насколько легко пользователю понять, как использовать систему	0.25
2	Время выполнения задачи	Среднее время выполнения типового сценария	0.20
3	Число ошибок	Количество ошибок, допущенных пользователем в процессе взаимодействия	0.15
4	Удовлетворенность пользователя	Оценка субъективного восприятия интерфейса	0.20
5	Логичность навигации	Насколько удобно и понятно устроено меню и переходы	0.20

Полученные данные в результате наблюдения за действиями пользователей были систематизированы и представлены в таблице 9. Оценка проводилась по пяти критериям, перечисленным в таблице 8: интуитивность интерфейса, среднее время выполнения типовых сценариев, количество

ошибок при взаимодействии, субъективная удовлетворённость интерфейсом и логичность навигации. Для каждого критерия был установлен весовой коэффициент, отражающий его значимость при общей оценке.

Таблица 9 – Результаты юзабилити-тестирования

Задача	Среднее время (сек)	Количество ошибок	Оценка удобства (1–5)
Авторизация	6	0	4.8
Учёт времени прихода	10	1	4.7
Создание задачи	15	1	4.6
Фильтрация данных	12	2	4.5
Генерация отчёта	17	0	4.9

На основании анализа поведения пользователей можно сделать вывод о высокой степени эргономичности системы и её соответствии требованиям в части usability. Интерфейс удовлетворяет основным критериям: простота, логичность, минимальное количество действий для выполнения задачи.

3.3 Анализ результатов использования программного обеспечения

На заключительном этапе была проведена опытная эксплуатация информационной системы в условиях ООО «ВорлдИнтерТех Рус». Основной целью данного этапа являлось определение фактической эффективности внедрения программного обеспечения, а также выявление возможных направлений его доработки.

По итогам использования в течение 14 рабочих дней были получены следующие результаты:

- снижение времени на ведение табелей: вручную табель в среднем заполнялся за 2–3 часа в неделю, с новой системой этот процесс автоматизирован и требует менее 20 минут;
- повышение прозрачности контроля: руководитель получает доступ к актуальной статистике в реальном времени, без необходимости

запроса отчётов у сотрудников;

- минимизация ошибок: исключены типичные проблемы ручного учёта пропуски, дублирование данных, некорректные даты;
- рост дисциплины сотрудников: благодаря регистрации времени входа и выхода повысилась точность соблюдения графика. Подобные эффекты особенно важны при удалённой занятости, где контроль и самодисциплина затруднены [20].

Следовательно, можно сделать вывод, что система успешно справляется с поставленными задачами, обладает высокой стабильностью, функциональностью и адаптирована под реальные условия эксплуатации предприятия.

В третьей главе проведено тестирование и анализ эффективности разработанного программного обеспечения. Система показала высокую стабильность при работе с большим количеством пользователей и удобство для конечных пользователей. Было установлено, что программное обеспечение значительно упрощает процесс учета рабочего времени, снижает количество ошибок и экономит время сотрудников. Также проведен анализ опыта использования системы на предприятии, который подтвердил её практическую пользу и готовность к реальному внедрению.

Заключение

Выпускная квалификационная работа была посвящена разработке информационной системы для учета и планирования рабочего времени сотрудников». В результате данной работы достигнута поставленная цель – создан программный продукт, автоматизирующий процесс учета рабочего времени и обеспечивающий руководство инструментами для контроля загруженности персонала.

В ходе выполнения работы решены следующие задачи:

- выполнен анализ существующих решений и обоснована необходимость разработки собственной системы;
- сформулированы функциональные и нефункциональные требования к программному обеспечению;
- спроектирована архитектура системы и структура базы данных;
- реализованы ключевые модули – учета времени, задач и отчетов;
- проведено тестирование и оценена эффективность внедрения.

С теоретической точки зрения работа обладает значимостью благодаря применению современных подходов к разработке информационных систем. В процессе реализации использовались принципы объектно-ориентированного программирования, методы построения ER-моделей и диаграмм UML, технологии клиент-серверной архитектуры и современные средства тестирования. Результаты работы могут служить примером практического применения базовых и профильных дисциплин, изученных в рамках подготовки бакалавров по направлению «Прикладная информатика».

Практическая значимость проекта подтверждена результатами опытной эксплуатации. Система показала стабильность работы при одновременном доступе до 150 пользователей, обеспечила сокращение времени на ведение табельного учета, повысила прозрачность контроля и дисциплину персонала. Данные результаты позволяют рекомендовать разработанное решение к внедрению в аналогичных организациях.

Информационная система имеет потенциал для дальнейшего совершенствования. Перспективными направлениями развития являются:

- расширение функционала для ведения учета удалённой работы;
- реализация уведомлений и напоминаний через почту или мессенджеры;
- разработка модуля аналитики и прогнозирования с использованием методов машинного обучения;
- внедрение системы мотивации на основе автоматизированной оценки продуктивности.

В результате проведённой работы удалось подтвердить, что выбранные подходы к созданию информационной системы являются эффективными как с теоретической, так и с практической точки зрения. Разработанная система соответствует задачам предприятия, удобна в использовании и может быть легко доработана или расширена в будущем при необходимости.

Список используемой литературы и источников

1. Баллод, Б. А. Проектирование информационных систем: технология автоматизированного проектирования. Лабораторный практикум / Б. А. Баллод, Т. В. Гвоздева. – 2-е изд., стер. – Санкт-Петербург: Лань, 2020. – 156 с.

2. Баранов, А. А. Основы проектирования информационных систем / А. А. Баранов. – Москва: Издательство «Кемерово», 2022. – 210 с.

3. Брусникин, Г. Н. Разработка UML-моделей при проектировании информационных систем: учебное пособие / Г. Н. Брусникин, Н. Ю. Соколова. — Москва: МИЭТ, 2023. – 52 с. – ISBN 978-5-7256-1016-1. – URL: https://e.lanbook.com/book/461570 (дата обращения: 17.05.2025).

4. Гильванов, Р. Г. Программирование на языке C#: учебное пособие / Р. Г. Гильванов, Л. М. Божко, А. Д. Хомоненко, И. Д. Липанов. – Санкт-Петербург: ПГУПС, 2024. – 89 с. – ISBN 978-5-7641-1977-9. – URL: https://e.lanbook.com/book/439523 (дата обращения: 25.05.2025).

5. Грекул, В. И. Проектирование информационных систем: учебник и практикум для среднего профессионального образования / В. И. Грекул, Н. Л. Коровкина, Г. А. Левочкина. – Москва: Юрайт, 2021. – 385 с. – ISBN 978-5-534-12104-9. – URL: https://urait.ru/bcode/476534 (дата обращения: 02.04.2025).

6. Гречко, Н. В. Автоматизация учета рабочего времени: тенденции и перспективы / Н. В. Гречко // Журнал информационных технологий. – 2023. – Т. 15, № 3. – С. 23–31.

7. Документация по интегрированной среде разработки Visual Studio // Microsoft. – URL: https://learn.microsoft.com/ru-ru/visualstudio/ide/?view=vs-2022 (дата обращения: 02.04.2025).

8. Заборовский, Г. А. Программирование на языке C#: учебно-методическое пособие / Г. А. Заборовский, В. В. Сидорик. — Минск: БНТУ, 2020. – 84 с. – ISBN 978-985-583-074-1. – URL: https://e.lanbook.com/book/248405 (дата обращения: 04.04.2025).

9. Зиновьев, П. А. Практическое применение Agile и Scrum в управлении проектами / П. А. Зиновьев // Журнал Agile и цифровизация. – 2023. – Т. 15, № 4. – С. 53–60.

10. Кузнецов, В. Е. Автоматизация учета рабочего времени сотрудников компании / В. Е. Кузнецов, И. Г. Смирнова, Н. Ю. Брызгалова // Universum: технические науки. – 2021. – № 6(87). – URL: https://7universum.com/ru/tech/archive/item/11891 (дата обращения: 05.05.2025).

11. Остроух, А. В. Проектирование информационных систем: монография / А. В. Остроух, Н. Е. Суркова. – 2-е изд., стер. – Санкт-Петербург: Лань, 2021. – 164 с. – ISBN 978-5-8114-8377-8. – URL: https://e.lanbook.com/book/175513 (дата обращения: 24.04.2025).

12. Официальный сайт компании ООО «ВорлдИнтерТех Рус» [Электронный ресурс]. – URL: http://www.worldintertech.ru (дата обращения: 01.11.2024).

13. Прайс, М. C# 10 и .NET 6. Современная кроссплатформенная разработка / М. Прайс. – 6-е изд. – Москва: ДМК Пресс, 2023. – 1019 с.

14. Приймак, К. С. Исследование и разработка информационной системы учета работы сотрудников / К. С. Приймак, И. А. Королькова, Е. Е. Сурина // Вестник науки. – 2024. – № 12 (81). – URL: https://cyberleninka.ru/article/n/issledovanie-i-razrabotka-informatsionnoy-sistemy-ucheta-raboty-sotrudnikov (дата обращения: 11.04.2025).

15. Проектирование информационных систем: практикум / под ред. П. В. Павлова. – Москва: НИУ ВШЭ, 2022. – 312 с.

16. Стружкин, Н. П. Базы данных: проектирование. Практикум: учебное пособие для вузов / Н. П. Стружкин, В. В. Годин. – Москва: Юрайт, 2021. – 291 с. – ISBN 978-5-534-00739-8. – URL: https://urait.ru/bcode/470023 (дата обращения: 05.05.2025).

17. Тихомиров, А. В. Проектирование баз данных для информационных систем / А. В. Тихомиров. – Москва: Бином, 2022. – 245 с.

18. Тюкачев, Н. А. C#. Основы программирования: учебное пособие для

вузов / Н. А. Тюкачев, В. Г. Хлебостроев. – 4-е изд., стер. – Санкт-Петербург: Лань, 2021. – 272 с. – ISBN 978-5-8114-7266-6. – URL: https://e.lanbook.com/book/158960 (дата обращения: 26.05.2025).

19. Шарп, Д. C# для профессионалов / Д. Шарп. – Москва: Бином, 2022. – 480 с.

20. Шевнина, Ю. С. Автоматизация учета рабочего времени сотрудников предприятия в условиях удаленной работы / Ю. С. Шевнина, А. Н. Буравов // Программные продукты и системы. – 2022. – № 1. – URL: https://cyberleninka.ru/article/n/avtomatizatsiya-ucheta-rabochego-vremeni-sotrudnikov-predpriyatiya-v-usloviyah-udalennoy-raboty (дата обращения: 24.05.2025).

21. Arlow, J. Generative Analysis: The Power of Generative AI for Object-Oriented Software Engineering with UML / J. Arlow, I. Neustadt. – Boston: Addison-Wesley Professional, 2024. – 672 p.

22. Bell, M. Software Architect / M. Bell. – Hoboken: Wiley, 2023. – 432 p.

23. Bitcop [Электронный ресурс]. – URL: https://bitcop.com (дата обращения: 13.04.2025).

24. Blockdick, G. IDEF / G. Blockdick. – 3rd ed. – 5STARCooks, 2022. – 299 p.

25. CrocoTime [Электронный ресурс]. – URL: https://crocotime.com (дата обращения: 13.04.2025).

26. Harvest [Электронный ресурс]. – URL: https://getharvest.com (дата обращения: 13.04.2025).

27. Hevo. C# SQL Server Connection: 3 Easy Methods. – 2023. – URL: https://hevodata.com/learn/c-sql-server/ (дата обращения: 25.05.2025).

28. Kaplan, V. Using Scripting for Working with SQL Server in C# / V. Kaplan // CODE Magazine. – 2021. – May/June. – URL: https://www.codemag.com/Article/2105071/ (дата обращения: 25.05.2025).

29. Microsoft. Open sourcing the .NET 5 C# Language Extension for SQL Server // Microsoft SQL Server Blog. – 2021. – URL:

https://www.microsoft.com/en-us/sql-server/blog/2021/09/08/open-sourcing-the-net-5-c-language-extension-for-sql-server/ (дата обращения: 25.05.2025).

30. Tarashchuk, S. Understanding ADO.NET with SQL Server in C# / S. Tarashchuk // Medium. – 2025. – May. – URL: https://medium.com/@sasa_de/understanding-ado-net-with-sql-server-in-c-7e00ca74cc6d (дата обращения: 25.05.2025).

31. Time Doctor [Электронный ресурс]. – URL: https://www.timedoctor.com (дата обращения: 13.04.2025).

32. Toggl Track [Электронный ресурс]. – URL: https://toggl.com (дата обращения: 13.04.2025).

33. Yang, H. UML Tutorials – Herong's Tutorial Examples / H. Yang. – 2024. – 72 p.