**Содержание**

[Введение 4](#_Toc137416877)

[1 Общая часть 6](#_Toc137416878)

[1.1 Описание предметной области 6](#_Toc137416879)

[1.2 Анализ существующей ситуации 8](#_Toc137416880)

[1.3 Постановка задачи 9](#_Toc137416881)

[1.4 Анализ существующих разработок и обоснование необходимости разработки……… 9](#_Toc137416882)

[2 Специальная часть 11](#_Toc137416883)

[2.1 Выбор технологий и инструментальных средств 11](#_Toc137416884)

[2.1.1 Выбор подхода к разработке 11](#_Toc137416885)

[2.1.2 Выбор среды разработки и языка программирования 13](#_Toc137416886)

[2.2 Разработка спецификаций 14](#_Toc137416887)

[2.2.1 Разработка диаграмм вариантов использования 14](#_Toc137416888)

[2.2.2 Разработка диаграмм последовательностей системы 15](#_Toc137416889)

[2.2.3 Разработка диаграммы пакетов 15](#_Toc137416890)

[2.2.4 Разработка диаграмм классов 15](#_Toc137416891)

[2.2.5 Разработка диаграммы состояний 16](#_Toc137416892)

[2.2.6 Разработка диаграммы деятельности 18](#_Toc137416893)

[2.2.8 Разработка инфологической модели базы данных 18](#_Toc137416894)

[2.2.9 Разработка физической модели базы данных 18](#_Toc137416895)

[2.3 Проектирование программного модуля 18](#_Toc137416896)

[2.3.1 Разработка алгоритмов реализации основных функций программного обеспечения…………. 18](#_Toc137416897)

[2.3.2 Проектирование пользовательского интерфейса 20](#_Toc137416898)

[2.4 Реализация программного обеспечения на выбранном языке программирования и в выбранной среде разработки 22](#_Toc137416899)

[2.5 Выбор стратегии тестирования, разработка тестов, тестирование и отладка программного обеспечения 25](#_Toc137416900)

[2.6 Разработка эксплуатационной документации 30](#_Toc137416901)

[2.6.1 Разработка руководства системного программиста 30](#_Toc137416902)

[2.6.2 Разработка руководства пользователя 31](#_Toc137416903)

[3 Экономика производства 32](#_Toc137416904)

[3.1 Расчет затрат на разработку и внедрение программного обеспечения 32](#_Toc137416905)

[3.2. Оценка эффективности внедрения программного обеспечения……………………. 32](#_Toc137416906)

[4 Безопасность жизнедеятельности 33](#_Toc137416907)

[4.1 Требования к рабочим помещениям и рабочим местам оператора ПК 33](#_Toc137416908)

[4.2 Расчет искусственной освещенности рабочего места оператора ПК 34](#_Toc137416909)

[4.3 Техника безопасности при работе на ПК 36](#_Toc137416910)

[Заключение 39](#_Toc137416911)

[Библиография 41](#_Toc137416912)

Приложение А (Обязательное) [Графическая часть 44](#_Toc137416913)

Приложение Б (Обязательное) [Техническое задание 45](#_Toc137416914)

Приложение В (Обязательное) [Руководство системного администратора 46](#_Toc137416915)

Приложение Г (Обязательное) [Руководство пользователя 47](#_Toc137416916)

Приложение Д (Обязательное) [Фрагмент листинга программы 48](#_Toc137416917)

# Введение

В современных условиях ведения бизнеса компании сталкиваются с необходимостью эффективного управления проектами, что становится особенно актуальным в условиях постоянных изменений и жесткой конкуренции. Неэффективное распределение задач и отсутствие инструмента для отслеживания статуса выполнения могут привести к задержкам, перерасходу ресурсов и снижению качества конечного продукта. Это создает дополнительные проблемы как для организаций, так и для их клиентов.

Разработка системы управления проектами с функцией распределения задач и отслеживания статуса выполнения представляет собой важное решение для повышения эффективности работы в компаниях. Эта система позволит оптимизировать процессы взаимодействия внутри команды, облегчить контроль за выполнением задач и упростить коммуникацию между членами проекта. В условиях, когда время имеет решающее значение, такой инструмент становится незаменимым.

На сегодняшний день вопросы управления проектами занимают центральное место в деятельности многих крупных предприятий, включая АО "Апатит". Внедрение автоматизированной системы управления проектами создаст новые возможности для оптимизации процессов и улучшения конечных результатов. Успешные примеры использования подобных систем в других компаниях показывают, что это не только повышает эффективность работы, но и способствует улучшению атмосферы внутри коллектива.

Цель данной дипломной работы заключается в разработке системы управления проектами для АО "Апатит", которая будет включать в себя функции распределения задач и отслеживания статуса выполнения. Для достижения этой цели необходимо реализовать следующие задачи:

• провести анализ существующих решений в области управления проектами;

• изучить потребности и требования сотрудников АО "Апатит";

• спроектировать архитектуру и функциональные возможности системы;

• разработать программное обеспечение и провести тестирование;

• внедрить систему в работу компании.

Внедрение системы управления проектами позволит АО "Апатит" повысить уровень организации рабочих процессов, улучшить координацию между командами и обеспечить выполнение проектов в установленные сроки. Таким образом, создание данной системы станет важным шагом к повышению конкурентоспособности и эффективному управлению проектами в компании.

В современном бизнесе компании нуждаются в эффективных инструментах управления проектами для повышения продуктивности и организации рабочих процессов. Разработка системы управления проектами с функцией распределения задач и отслеживания статуса выполнения станет важным решением для АО "Апатит", позволяя оптимизировать работу и улучшить координацию в команде.

Данная система будет полезна всем сотрудникам, обеспечивая четкое распределение задач и контроль за их выполнением, что поможет оперативно выявлять проблемы и повышать общую эффективность проектов.

АО "Апатит" заинтересовано в создании такой системы, стремясь улучшить внутренние процессы и повысить конкурентоспособность. Эта дипломная работа направлена на разработку системы, которая обеспечит компании новое конкурентное преимущество и укрепит её позиции на рынке, предлагая более эффективные решения для управления проектами.

# Общая часть

## Описание предметной области

Предметной областью данной дипломной работы является управление проектами, а именно функции распределения задач и отслеживания статуса выполнения в АО "Апатит". Управление проектами включает в себя планирование, координацию и контроль различных задач, сформированных в рамках проектов, и направлено на достижение поставленных целей в установленные сроки и с оптимальным использованием ресурсов.

В организации проектов важным аспектом является распределение задач между участниками команды, что требует учета их компетенций и загрузки. Эффективное распределение задач способствует повышению производительности и снижению рисков задержек в выполнении работ.

Отслеживание статуса выполнения задач — это процесс мониторинга прогресса, позволяющий своевременно выявлять проблемы и корректировать планы. Это важный элемент управления проектом, который обеспечивает прозрачность и ответственность на каждом этапе его реализации.

Системы управления проектами помогают наладить коммуникацию внутри команды, оптимизировать процессы, предоставить актуальную информацию о ходе выполнения работ и повысить общее качество управляемых проектов. Неэффективность в управлении может привести к перерасходу бюджета, превышению сроков выполнения и неудаче в достижении целей проекта.

Таким образом, предметная область данной работы охватывает технологии и методы, связанные с управлением проектами, их планированием, распределением задач и отслеживанием статуса выполнения. Разработка автоматизированной системы управления проектами в АО "Апатит" позволит значительно улучшить качество управления и повысить эффективность работы команды, а также оптимизировать ресурсы и снизить риски отрицательных последствий.

## Анализ существующей ситуации

Существующая ситуация в управлении проектами в АО "Апатит" демонстрирует ряд значительных проблем, которые можно эффективно решить с помощью автоматизированной системы управления проектами с функциями распределения задач и отслеживания статуса выполнения:

1) Неоптимальное распределение задач. В современных условиях недостаточное внимание к компетенциям сотрудников при назначении задач приводит к перегрузке некоторых участников и недозагрузке других, что сказывается на производительности.

2) Трудности в отслеживании статуса выполнения. В настоящее время в компании отсутствуют централизованные инструменты для мониторинга прогресса выполнения задач, что затрудняет контроль за ходом выполнения проектов. Это может приводить к замешательству и задержкам, когда информация о текущем статусе проекта оказывается недоступной для заинтересованных сторон.

3) Недостаточная коммуникация в команде. Отсутствие прозрачности в процессах управления проектами затрудняет взаимодействие между членами команды, что приводит к трудностям в согласовании действий и устранении возникших проблем. Это может способствовать возникновению конфликтов и замедлению рабочего процесса.

4) Общая неэффективность управления проектами. Отсутствие системного подхода к управлению проектами, а также несогласованность действий участников приводят к перерасходу ресурсов, увеличению сроков выполнения задач и потере качества конечного продукта.

В целом, внедрение системы управления проектами с функцией распределения задач и отслеживания статуса выполнения может улучшить качество управления, повысить эффективность работы команды и обеспечить более прозрачный процесс, что в итоге приведет к достижению стратегических целей компании.

## Постановка задачи

Целью данной дипломной работы является разработка системы управления проектами для АО "Апатит" с функциями распределения задач и отслеживания статуса выполнения. Основное требование — обеспечение прозрачности процессов управления и повышение эффективности выполнения проектов.

Для достижения этой цели необходимо провести анализ текущих методов управления проектами, выявить существующие проблемы и определить наилучшие практики для внедрения в систему. На основе анализа будет разработана архитектура программного обеспечения, обеспечивающая интеграцию с существующими системами компании и возможность масштабирования.

Система должна быть интуитивно понятной для пользователей, предоставляя актуальную информацию о статусе задач и позволяя генерировать отчеты по проектам. В процессе разработки будут применяться современные подходы к созданию ПО, что обеспечит успешное внедрение системы в рабочие процессы АО "Апатит" и поможет достичь поставленных целей.

1.4 Анализ существующих разработок и обоснование необходимости разработки

На рынке уже существует несколько решений для управления проектами, однако многие из них не отвечают специфическим требованиям АО "Апатит" или не обеспечивают полноценного контроля за процессами распределения задач и отслеживания их выполнения.

Существующие системы управления проектами, такие как Jira и Trello, предоставляют различные подходы к организации работы команд и ведению проектов, однако каждая из них имеет свои особенности и ограничения.

Jira — это мощная система, разработанная компанией Atlassian, которая подходит для управления проектами в IT-сфере, особенно в Agile-методологиях. Она предлагает широкий набор возможностей для отслеживания задач, отчетности и интеграции с другими разработками Atlassian, такими как Confluence и Bitbucket. Однако, она может быть излишне сложной для небольших команд и требует значительного времени на обучение и настройку под конкретные процессы, что может быть нецелесообразно для организаций, не занимающихся программной разработкой.

Trello — это более простая и визуально ориентированная система, которая основана на концепции канбан-доски. Trello позволяет пользователям легко добавлять и перемещать карточки задач между колонками, что делает его интуитивно понятным и удобным для визуализации рабочего процесса. Тем не менее, его функционал ограничен по сравнению с более сложными инструментами, что может стать препятствием для более серьезного проектного управления, поскольку система не обеспечивает достаточных возможностей по аналитике и отчетности.

Существуют также специализированные системы, такие как Asana и Basecamp, которые предлагают различные функции для управления проектами, такие как управление задачами, календарем и коммуникацией внутри команды. Эти решения могут быть более подходящими для определенных типов проектов, но также могут не удовлетворять специфические нужды АО "Апатит".

Каждая из представленных систем имеет свои преимущества и недостатки, однако ни одна из них не предоставляет полностью интегрированного решения, которое могло бы быть адаптировано под уникальные процессы и требования АО "Апатит". Это создает конкуренцию для разработки собственной системы управления проектами, способной более точно отвечать специфическим запросам компании и улучшить эффективность ее работы.

# Специальная часть

## Выбор технологий и инструментальных средств

### 2.1.1 Выбор подхода к разработке

Для разработки системы более подходящим будет использовать объектно-ориентированный подход.

Этот подход обеспечит четкое разделение ответственности между компонентами системы, такими как управление задачами, интерфейсы пользователя и базы данных. Важно, что система сможет легко адаптироваться к изменениям и расширяться за счет добавления новых функций или интеграции с другими системами.

При использовании объектно-ориентированного подхода можно применять классы и объекты для определения компонентов системы, их свойств и методов, а также для управления взаимодействием между ними. Это позволяет создать структурированную систему, которая будет удобной для поддержки и масштабирования, а также легко приспособится к изменяющимся бизнес-требованиям.

В отличие от объектно-ориентированного, структурный подход может оказаться ограничивающим для управления сложными взаимодействиями и обеспечивать недостаточную гибкость для переиспользования компонентов в других проектах. Он также может привести к проблемам с повторением кода и конфликтами между различными модулями, что затруднит поддержку системы.

Таким образом, объектно-ориентированный подход является предпочтительным для разработки системы управления проектами, так как он предоставляет необходимую гибкость, масштабируемость и облегчает процесс тестирования и отладки, что важно для долгосрочной работоспособности и успешного обслуживания системы.

### 2.1.2 Выбор среды разработки и языка программирования

Язык C# и Visual Studio Community 2022 являются, как по мне, отличным выбором для разработки системы управления проектами с функциями распределения задач и отслеживания статуса выполнения.

С# — это объектно-ориентированный язык программирования, который имеет сильную типизацию и поддержку сборки мусора. Он разработан Microsoft, и обладает большим набором возможностей по разработке приложений для Windows. С# также поддерживает парадигму функционального программирования, что может быть полезным в разработке более сложных частей системы.

Visual Studio Community 2022 — это интегрированная среда разработки (IDE) от Microsoft, которая обладает множеством функций и инструментов, которые значительно упрощают процесс разработки ПО. Visual Studio Community 2022 предоставляет различные инструменты разработки, включая мощный отладчик, поддержку Git, автоматические рефакторинги и другие.

Среди прочих преимуществ использования C# и Visual Studio Community 2022 для разработки системы управления проектами с функциями распределения задач и отслеживания статуса выполнения можно отметить:

1. Высокая скорость разработки. Большое количество доступных функций и удобство Visual Studio Community 2022 позволяет быстро создавать и отлаживать код.
2. Широкий спектр библиотек и фреймворков. C# имеет большое количество полезных библиотек и фреймворков для работы с различными частями системы, включая работу с базами данных, сетевыми протоколами и т.д.
3. Большое количество сообщества. C# имеет широкое сообщество разработчиков, которые готовы помочь локализовать проблемы и поделиться знаниями и опытом.

В целом, использование C# и Visual Studio Community 2022 позволит создать быструю, функциональную и надежную автоматизированную систему, которая будет работать на высоком уровне.

## Разработка спецификаций

### 2.2.1 Разработка диаграмм вариантов использования

Диаграмма представляет собой процесс работы программного обеспечения, представлена ниже на рисунке 1.

Изображение выглядит как текст, круг, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1 – Диаграмма вариантов использования

На диаграмме представлены следующие варианты использования для системы управления проектами:

«Регистрация/вход в систему» – процесс создания аккаунта или авторизации пользователя.

«Создание проекта» – инициация нового проекта с указанием названия, описания и сроков выполнения.

«Назначение задач» – распределение задач между членами команды в рамках созданного проекта.

«Обновление статуса задачи» – обновление текущего статуса задач для отслеживания прогресса выполнения.

«Просмотр списка задач» – возможность видеть все задачи проекта с их статусами, исполнителями и сроками.

«Просмотр отчетов о выполнении» – возможность видеть сводные данные о прогрессе выполнения задач и проекта.

«Редактирование задач» – изменение деталей назначенных задач в любое время после их назначения.

### 2.2.2 Разработка диаграмм последовательностей системы

### 2.2.3 Разработка диаграммы пакетов

Для данной разработки построена диаграмма пакетов, которая показана ниже на рисунке 3. Она показывает зависимости между частями.

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2 – Диаграмма пакетов

### 2.2.4 Разработка диаграмм классов

Для разработки пакета для взаимодействия создана диаграмма классов, показанная ниже на рисунке 3.

☹

Рисунок 3 – Диаграмма классов

Диаграмма классов описывает типы объектов системы и различного рода статические отношения, которые существуют между ними.

### 2.2.5 Разработка диаграммы состояний

Изображение выглядит как снимок экрана, текст, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4 – Диаграмма состояний

Диаграмма состоит из трех состояний:

1) Ожидание ввода данных - начальное состояние, в котором система ожидает ввода пользователем информации о проекте, задачах и участниках.

2) Обработка данных - состояние, в котором система обрабатывает введенные данные: распределяет задачи между участниками, устанавливает начальные статусы и генерирует необходимые отчеты.

3) Отображение статуса задач - конечное состояние, в котором система отображает пользователю актуальные статусы выполнения задач и общую информацию о проекте.

Переход между состояниями происходит следующим образом:

- Пользователь вводит данные: Система переходит из состояния "Ожидания ввода данных" в состояние "Обработка данных".

- Система завершает обработку и отображает результаты: Система переходит в состояние "Отображение статуса задач".

- Пользователь решает ввести новые данные: Система возвращается в состояние "Ожидание ввода данных".

Данная диаграмма позволяет лучше понимать работу системы и последовательность её действий в зависимости от введенных пользователем данных.

### 2.2.6 Разработка диаграммы деятельности

Изображение выглядит как текст, снимок экрана, диаграмма, зарисовка

Контент, сгенерированный ИИ, может содержать ошибки.

Данная диаграмма деятельности описывает последовательность действий, необходимых для создания системы, которая позволяет пользователю такие как создание проектов и задач, их распределение и изменение статусов.

### 2.2.8 Разработка инфологической модели базы данных

### 2.2.9 Разработка физической модели базы данных

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Пункт 2.2.8

## Проектирование программного модуля

### 2.3.1 Разработка алгоритмов реализации основных функций программного обеспечения

Алгоритм начинается с начинается с запуска ПО.

Далее запускается система и просит пользователя ввести данные для входа.

Следующим этапом является создание и просмотр проектов, назначение статуса задач и установление сроков.

Дальше на выходе система предоставляет актуальные статусы задач, распределение исполнителей, уведомления и отчёты по проектам.

Изображение выглядит как текст, чек, диаграмма, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5 – Алгоритм получения результатов

### 2.3.2 Проектирование пользовательского интерфейса

Процедурно-ориентированные интерфейсы используют традиционную модель взаимодействия с пользователем, основанную на понятиях «процедура» и «операция». В рамках этой модели программное обеспечение предоставляет пользователю возможность выполнения некоторых действий, для которых пользователь определяет соответствующие данные и следствием выполнения которых является получение желаемых результатов.

Объектно-ориентированные интерфейсы используют несколько иную модель взаимодействия с пользователем, ориентированную на манипулирование объектами предметной области. В рамках этой модели пользователю предоставляется возможность напрямую взаимодействовать с каждым объектом и инициировать выполнение операций, в процессе которых взаимодействуют несколько объектов.

Для того, чтобы понять какой интерфейс использовать для разработки проведена небольшая сравнительная характеристика ниже в таблице 1.

Таблица 1 – Сравнительная характеристика интерфейсов

|  |  |  |
| --- | --- | --- |
| № | **Процедурно-ориентированные пользовательские интерфейсы** | **Объектно-ориентированные пользовательские интерфейсы** |
|  | | |
| 1 | Обеспечивают пользователей функциями, необходимыми для выполнения задач | Обеспечивают пользователям возможность взаимодействия с объектами |
| 2 | Акцент делается на задачи | Акцент делается на входные данные и результаты |
| 3 | Пиктограммы представляют приложения, окна или операции | Пиктограммы представляют объекты |
| 4 | Содержание папок и справочников отображается с помощью таблиц и списков | Папки и справочники являются визуальными контейнерами объектов |

У данных интерфейсов также есть типы.

Процедурно-ориентированные и объектно-ориентированные интерфейсы делятся на 4 типа: «примитивные», меню и со свободной навигацией, а также прямого манипулирования. Их сравнение приведено ниже.

Примитивным называют интерфейс, который организует взаимодействие с пользователем в консольном режиме. Обычно такой интерфейс реализует конкретный сценарий работы программного обеспечения.

Интерфейс-меню в отличие от примитивного интерфейса позволяет пользователю выбирать необходимые операции из специального списка, выводимого ему программой. Эти интерфейсы предполагают реализацию множества сценариев работы, последовательность действий в которых определяется пользователем.

Интерфейсы со свободной навигацией также называют графическими пользовательскими интерфейсами - что пользователь видит на экране, то он и получит при печати. Эти названия подчеркивают, что интерфейсы данного типа ориентированы на использование экрана в графическом режиме с высокой разрешающей способностью.

Объектно-ориентированные интерфейсы пока представлены одним видом: интерфейс прямого манипулирования. Этот тип интерфейса предполагает, что взаимодействие пользователя с программным обеспечением осуществляется посредством выбора и перемещения пиктограмм, соответствующих объектам предметной области.

Для данного ПО был выбран интерфейс-меню, так как в отличие от других интерфейсов он позволяет пользователю выбирать необходимые операции из специального списка, выводимого ему программой.

## 2.4 Реализация программного обеспечения на выбранном языке программирования и в выбранной среде разработки

Для реализации программного обеспечения на языке C# для разработки системы управления проектами с функцией распределения задач и отслеживания статуса выполнения можно использовать различные среды разработки, такие как Microsoft Visual Studio, JetBrains Rider, SharpDevelop.

Разработка программы в среде Visual Studio будет представлять собой следующие шаги:

1. Создание проекта.

В меню выбирается пункт "Создать проект", а затем - тип проекта, например, "Windows Forms App".

1. Добавление нужных файлов.

В проект необходимо добавить файлы, отвечающие за обработку видео, выделение областей и прочие нужные для реализации функции.

1. Написание кода.

Код программы написанный на языке C# реализует основные функции, описанные выше. В коде используются методы классы для проектов и пользователей и многое другое.

Пример некоторых функций, которые могут быть реализованы:

// Распределение задач

public void AssignTask(TaskItem task, string assignee)

{

task.Assignee = assignee;

}

// Отслеживание статуса

public void UpdateTaskStatus(TaskItem task, string status)

{

task.Status = status;

}

// Добавление новой задачи

private void AddTask(TaskItem task)

{

taskList.Add(task);

}

4. Тестирование и отладка.

Система проходит тестирование с использованием готовых и своих сценариев для проверки корректности распределения задач и обновления статусов.

5. Компиляция и сборка.

После успешных тестов проект компилируется в исполняемый файл ".exe" для удобного запуска пользователями.

6. Развертывание.

Программа устанавливается на ПК сотрудников, серверы или предоставляется через интернет для доступа и работы с системой.

Таким образом, разработка системы управления проектами с функцией распределения задач и отслеживания статуса требует поэтапной реализации, тщательного тестирования, качественной сборки и грамотного развертывания, что гарантирует надежность и эффективность решения в рамках конкретных бизнес-задач.

## 2.5 Выбор стратегии тестирования, разработка тестов, тестирование и отладка программного обеспечения

Для выбора наиболее подходящего метода и инструмента тестирования необходимо учитывать ряд критериев, таких как освоение, эффективность, распространенность и принцип использования.

В табличной форме приведена сравнительная характеристика методов тестирования и инструментов для автоматизации тестирования, соответственно. Так, в таблице 1 представлено сравнение черного, белого и серого ящиков, включая преимущества и недостатки каждого метода. В таблице 2 отражены основные характеристики инструментов Selenium, Selenium IDE и PHPUnit, таких как функциональность, сложность использования и распространенность среди разработчиков.

Однако стоит учитывать, что выбор метода и инструмента тестирования зависит от конкретного проекта и его специфики. Поэтому рекомендуется проводить тщательный анализ требований проекта и оценивать эффективность тестирования на основе выбранного метода и инструмента.

| **Характеристика** | **Черный ящик** | **Белый ящик** | **Серый ящик** |
| --- | --- | --- | --- |
| Описание | В этом методе тестирующий работает со стороны пользователя, не зная деталей внутренней работы программы. | В этом методе тестирующий имеет доступ к исходному коду программы и может проводить тестирование на уровне алгоритмов и структур данных. | Это смешанный метод, который сочетает некоторые элементы "черного ящика" и "белого ящика". Тестирующий имеет ограниченный доступ к исходному коду программы, что позволяет ему проводить более детальное тестирование, однако он все еще не имеет полного доступа ко всем деталям реализации. |
| Объект тестирования | Объектом тестирования является готовое приложение или система в целом. | Объектом тестирования является исходный код и отдельные компоненты программы. | Объектом тестирования является как готовое приложение, так и частично доступный для анализа исходный код. |
| Сложность тестирования | Метод "черного ящика" обычно проще в использовании, так как не требует знания языка программирования или особой экспертизы в данной области. | Метод "белого ящика" может быть более сложным, так как требуется знание языка программирования и понимание внутренней работы программы. | Метод "серого ящика" находится между "черным" и "белым" ящиками по сложности. Тестирующий должен иметь некоторые знания в области программирования, чтобы анализировать доступный ему код и проводить дополнительные тесты. |
| Уровень детализации тестирования | Метод "черного ящика" позволяет проводить тестирование на уровне функциональности приложения, однако не обеспечивает глубокого понимания внутренней работы программы. | Метод "белого ящика" позволяет проводить тестирование на уровне отдельных модулей программы и алгоритмов. | Метод "серого ящика" обеспечивает более детальное тестирование, чем "черный ящик", но не такое глубокое и полное, как "белый ящик". |
| Возможность автоматизации тестирования | Метод "черного ящика" легко поддается автоматизации с помощью специализированных инструментов для автоматического тестирования. | Метод "белого ящика" также может быть автоматизирован, однако требует более тщательной настройки и ручной работы. | Метод "серого ящика" может быть автоматизирован в определенной степени. |

Таблица 2 – Инструменты для автоматизации тестирования

| **Инструмент** | **Описание** | **Функциональность** | **Сложность использования** | **Распространенность** |
| --- | --- | --- | --- | --- |
| Selenium | Фреймворк для автоматизированного тестирования веб-приложений. | Позволяет записывать и воспроизводить действия пользователя на веб-страницах, проверять содержимое элементов интерфейса, реализовать тестовые сценарии. | Высокая сложность использования, так как требует знания языка программирования и разработки автоматических скриптов. | Один из самых популярных инструментов для автоматизации тестирования веб-приложений. |
| Appium | Фреймворк для автоматизированного тестирования мобильных приложений. | Позволяет записывать и воспроизводить действия пользователя на мобильных устройствах, проверять содержимое элементов интерфейса, реализовать тестовые сценарии. | Высокая сложность использования, так как требует знания языка программирования и разработки автоматических скриптов. | Широко используется для автоматизации тестирования мобильных приложений на платформах Android и iOS. |
| JMeter | Инструмент для функционального и нагрузочного тестирования веб-приложений. | Позволяет создавать и запускать тестовые сценарии, генерировать нагрузку на серверы, анализировать результаты. | Умеренная сложность использования, так как требует знания основ работы с веб-приложениями и настройки нагрузочных тестов. | Широко используется для функционального и нагрузочного тестирования веб-приложений. |
| TestComplete | Коммерческий инструмент для автоматизированного тестирования различных типов приложений. | Позволяет записывать и воспроизводить действия пользователя на экране, проверять содержимое элементов интерфейса, реализовать тестовые сценарии. | Относительно низкая сложность использования, так как имеет графический интерфейс и не требует знания языка программирования. | Широко используется для автоматизации тестирования различных типов приложений. |
| Postman | Инструмент для тестирования API. | Позволяет отправлять запросы к API и анализировать ответы, создавать тестовые сценарии для проверки работоспособности API. | Относительно низкая сложность использования, так как имеет графический интерфейс и не требует знания языка программирования. | Широко используется для тестирования API различных веб-сервисов. |

Исходя из таблицы выше и разрабатываемого ПО, необходимо выбрать инструмент для тестирования приложения, которое использует нейросеть для разработки системы управления проектами с функциями распределения задач и отслеживания статуса выполнения. Учитывая, что приложение написано на языке C#, можно выделить два наиболее подходящих инструмента:

1 Selenium - этот инструмент позволяет проводить автоматизированное тестирование веб-приложений, в том числе приложений, написанных на языке C#. Он поддерживает множество языков программирования, включая C#, и может использоваться для тестирования в разных браузерах.

2 NUnit — это популярный фреймворк для автоматизированного тестирования на языке C#. Он имеет обширную документацию и сообщество разработчиков, что делает его удобным инструментом для работы. Также он поддерживает множество функций и интеграцию с другими инструментами. Кроме того, NUnit позволяет проводить модульное тестирование, что может быть полезным для отдельных компонентов. Он может быть использован для тестирования, написанных на языке C#.

Таким образом, для тестирования приложения, наиболее подходящим инструментом будет NUnit. Он предоставляет полный набор функций для тестирования приложений на языке C# и позволяет проводить проверку отдельных компонентов системы.

## 2.6 Разработка эксплуатационной документации

### 2.6.1 Разработка руководства системного программиста

По ГОСТ 19.503-79 руководство системного программиста должно содержать всю информацию, необходимую для установки программного обеспечения, его настройки и проверки работоспособности. Кроме того, как указывалось выше, в него часто включают и описание необходимого обслуживания, которое раньше приводилось в руководстве оператора (ГОСТ 19.505-79) и/или руководстве по техническому обслуживанию (ГОСТ 19.508-79). В настоящее время данную схему используют для составления руководства системному администратору.

Руководство системного программиста должно содержать следующие разделы:

* 1. общие сведения о программном продукте;
  2. структура;
  3. настройка;
  4. проверка;
  5. дополнительные возможности;
  6. сообщения системному программисту.

Руководство системного программиста представлено в приложении В.

### 2.6.2 Разработка руководства пользователя

Руководство пользователя согласно ГОСТ РД 50-34.698-90- документ, в котором описывается, как пользоваться продуктом, включая установку на компьютер и работу с ним. В него также может быть включена информация об ограничениях на использование продукта и обратная связь с поддержкой.

Руководство пользователя представлено в приложении Г.

# Экономика производства

## 3.1 Расчет затрат на разработку и внедрение программного обеспечения

Затраты на разработку программного продукта определяются по формуле:

*Зрп = Зот + Змв + Зэл + 3п* , *(1)*

где *Зот* – затраты на оплату труда и страховые взносы, руб(172842,84);

*Змв* - затраты на оплату машинного времени, руб(5136,52);

*Зэл* - Затраты на электроэнергию, руб(4188,35);

*Зп*- Прочие затраты, руб(9108,39).

*Зрп = 172842,84+5136,52+4188,35+9108,39*

*Зрп =191276,10 руб*

## 3.2. Оценка эффективности внедрения программного обеспечения

# 4 Безопасность жизнедеятельности

## 4.1 Требования к рабочим помещениям и рабочим местам оператора ПК

Рабочие помещения и рабочие места оператора ПК должны соответствовать определенным требованиям, которые обеспечат комфортную и безопасную работу. К ним относятся:

1. Размеры помещения не менее 6 квадратных метров на одно рабочее место.
2. Наличие естественного и искусственного освещения, соответствующего СанПиН 2.2.2./2.4.1340-03 «Гигиенические требования к условиям труда при работе на персональных компьютерах».
3. Температура воздуха в помещении должна находиться в пределах 18-24 градусов Цельсия, влажность - 40-60%.
4. Отсутствие шума и других нежелательных звуковых эффектов.
5. Наличие пожарных средств и путей эвакуации.
6. Наличие мебели, соответствующей правилам эргономики (регулируемая высота, регулируемый наклон стола, спинка и сиденье стула, наличие подставки для ног и т.д.).
7. Наличие специального оборудования для защиты здоровья (экраны, фильтры, подставки для документов и т.д.).
8. Соответствие параметров компьютера установленным нормам.
9. Работа оператора ПК должна проходить в условиях микроклимата, обеспечивающего наиболее комфортные условия для зрения (уровень освещенности должен быть не менее 300 лк).
10. Обеспечение необходимой пространственной ориентации с помощью различных методов размещения монитора и клавиатуры.

Общие требования при использовании компьютерной техники:

1. Защита зрения.
2. Защита органов слуха.
3. Защита органов пищеварения.
4. Защита кожных покровов.
5. Установка расписания работы и время отдыха в течение дня.

## 4.2 Расчет искусственной освещенности рабочего места оператора ПК

Расчет искусственной освещенности рабочего места оператора ПК является необходимым для обеспечения комфортных условий работы с учетом световых параметров, необходимых для нормального функционирования зрительного аппарата человека.

Согласно требованиям СанПиН 2.2.2./2.4.1340-03, искусственное освещение должно обеспечивать не менее 300 лк на поверхности стола и не менее 500 лк на поверхности клавиатуры и экрана монитора.

Для расчета искусственной освещенности рабочего места оператора ПК необходимо знать:

1. Площадь рабочего помещения (S).
2. Коэффициент использования светового потока (K) - показатель, показывающий, какая часть из общей мощности светильника будет использоваться для освещения рабочей поверхности.
3. Световой поток, выделяемый одним светильником (F).
4. Коэффициент освещенности помещения, отношение светового потока к площади помещения (E=F/S).

Искусственная освещенность оператора ПК рассчитывается по формуле:

*E x K x η x T = L, (3)*

гдеE – коэффициент освещенности помещения, лк/м²;

K – коэффициент использования светового потока, принимается примерно равным 0,5;

η - коэффициент светопропускания светопрозрачных элементов в помещении (окна и т.п.), принимается в пределах 0,7-0,9;

T- коэффициент светового рассеивания стен белого цвета, имеющих светлую поверхность или покрытие, принимается равным 0,7 - 0,9;

L – необходимая освещенность поверхности рабочего места оператора ПК, лк.

Согласно стандартам, коэффициент освещенности для помещения размером 15 квадратных метров должен составлять не менее 300 лк/м². Для достижения этого значения и энергоэффективной работы светильника, мы можем рассчитать оптимальную световую мощность и эффективность светильника.

Подставляем значения в формулу:

*300 лк/м² × 15 м² = 4500 лк*

Для достижения необходимой световой мощности в 4500 лм при коэффициенте использования светового потока K=0,5, мы можем пересчитать эффективность светильника по формуле:

*P = (F × K) / η*

*η = (F × K) / P*

*η = (4500 лм × 0,5) / 50 Вт = 45*

Таким образом, чтобы обеспечить достаточный уровень освещенности помещения 15 кв. м с общим световым потоком 4500 лм, достаточно использовать светильники с эффективностью 45 лм/Вт и мощностью 50 Вт. Кроме того, для получения энергоэффективного решения, можно использовать светодиодные светильники, которые потребляют меньше энергии при той же световой мощности.

## 4.3. Техника безопасности при работе на ПК

Современная жизнь невозможна без использования компьютеров и интернета. Тем не менее, работа на ПК может представлять опасность для здоровья человека, если не соблюдать правила безопасности. Такие заболевания, как синдром карпального канала, синдром "сухого глаза", головные боли и шум в ушах, связаны с длительным пребыванием за компьютером. В связи с этим, необходимо соблюдать следующие правила техники безопасности при работе на ПК:

1. Регулярно делать перерывы. При работе за компьютером необходимо делать перерыв каждые 45-60 минут. Во время перерывов рекомендуется выполнять упражнения для глаз и рук.
2. Правильно настроить рабочее место. Рабочее место должно быть правильно настроено, чтобы минимизировать риск различных заболеваний. Клавиатура должна быть расположена на уровне локтя, а экран - на уровне глаз.
3. Использовать эргономическую мебель. Для работы за компьютером рекомендуется использовать эргономические кресла и столы, которые помогают поддерживать правильную позу.
4. Соблюдать правильный режим работы. Необходимо соблюдать правильный режим работы и сна, чтобы избежать усталости и стресса, которые могут привести к заболеваниям.
5. Использовать программы для защиты глаз. Существует множество программ, которые помогают защитить глаза при работе за компьютером. Они уменьшают яркость экрана, фильтруют синий свет и т.д.
6. Использовать антивирусное программное обеспечение. Антивирусное программное обеспечение помогает защитить компьютер от вирусов и злонамеренных программ, которые могут повредить систему или украсть личную информацию.
7. Избегать монотонной нагрузки. Повторяющиеся действия могут привести к снижению производительности и здоровью. Для этого рекомендуется использовать различные программы и ресурсы, менять виды деятельности, чтобы предотвратить нагрузку на конкретную группу мышц.
8. Использовать безопасные пароли. Для защиты личной информации и конфиденциальных данных следует использовать сложные пароли, которые не легко угадать или взломать. Рекомендуется использовать комбинации букв, цифр и символов.
9. Создавать резервные копии данных. Для сохранения важной информации необходимо регулярно создавать резервные копии данных, чтобы предотвратить потерю или повреждение важных файлов.
10. Обновлять программное обеспечение. Регулярное обновление программного обеспечения помогает устранять уязвимости и предотвращать атаки злонамеренных программ. Рекомендуется устанавливать обновления операционной системы, браузера и антивирусного ПО.
11. Использовать безопасную сеть. При работе на публичных сетях необходимо использовать VPN для защиты информации, передаваемой между компьютером и сервером или другим устройством.
12. Избегать использования нелицензионного программного обеспечения. Использование нелицензионного программного обеспечения может привести к нарушению законодательства и повышенному риску заражения вирусами и злонамеренными программами.

В целом, соблюдение правил техники безопасности при работе на ПК имеет важное значение для сохранения здоровья и безопасности человека. Рекомендуется следовать простым правилам, чтобы избежать проблем, связанных с длительным пребыванием за компьютером.

# Заключение

В заключение дипломной работы по разработке системы управления проектами с функцией распределения задач и отслеживания статуса выполнения для АО "Апатит" можно отметить, что создание данной системы представляет собой значительный шаг в направлении повышения эффективности управления проектами в организации. Разработка системы позволяет оптимизировать процессы планирования и контроля выполнения задач, а также значительно улучшить взаимодействие между участниками проектов.

Система обеспечивает автоматизированный подход к распределению задач, что позволяет не только упростить управление проектами, но и повысить прозрачность статусов выполнения делегированных задач. Благодаря интуитивно понятному интерфейсу и функционалу, пользователи смогут легко адаптироваться к использованию системы, что, в свою очередь, способствует сокращению временных затрат на обучение и внедрение.

Для достижения поставленных целей были выполнены следующие задачи:

* проведено детальное описание предметной области и потребностей организации.
* выполнен анализ существующих решений в сфере управления проектами.
* обоснована необходимость разработки собственной системы для оптимизации текущих процессов.
* выбраны подходящие технологии и инструменты для реализации.
* разработана спецификация проектируемой системы.
* проведено проектирование программного обеспечения.
* реализовано и протестировано программное обеспечение.
* составлены эксплуатационная документация и руководство для пользователей.

Таким образом, разработанная система является актуальной и полезной для повышения общей продуктивности и эффективности работы команды. Внедрение данной системы позволит организации более успешно справляться с текущими вызовами, оптимизировать внутренние процессы и улучшить качество выпускаемой продукции

# Библиография

1. Бек К. Экстремальное программирование: Встречайте перемены. — М.: Вильямс, 2004.
2. Белломо С., Бианкофиоре В. Основы управления проектами. — СПб.: Питер, 2018.
3. Вайсоки Р. К. Эффективное управление проектами: традиционное, Agile, экстремальное. — М.: Вильямс, 2013.
4. Гандомани Т. Дж., Зульзалил Х., Гани А. А. А., Султан А. Б. М. Погрешности управления agile-проектами: концептуальное исследование // International Journal of Managing Projects in Business. — 2013. — Т.6, №1. — С.156–189.
5. ГОСТ Р ИСО 9001-2015 Качество. Управление качеством. Требования [Электронный ресурс]. URL: https://docs.cntd.ru/document/1200134310 (дата обращения: 13.05.2024).
6. Джеффри Рихтер. «Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд».
7. Дингсьёр Т., Нерур С., Балиджепалли В., Мое Н. Б. Десятилетие гибких методологий: объяснение agile-разработки // Journal of Systems and Software. — 2012. — Т.85, №6. — С.1213–1221.
8. Иванов В. П., Петров С. А. Автоматизация процессов управления проектами в среде C#. — М.: Наука, 2020.
9. Кеннет Р., Тоска Р. Управление задачами и контроль выполнения в IT-проектах. — СПб.: Питер, 2019.
10. Кёртис Б., Краснер Х., Иско Н. Исследование процесса проектирования ПО для крупных систем // Communications of the ACM. — 1988. — Т.31, №11. — С.1268–1287.
11. Керцнер Х. Управление проектами: системный подход к планированию, расписанию и контролю. — СПб.: Питер, 2017.
12. Кокберн А. Гибкая разработка программного обеспечения. — М.: Питер, 2006.
13. Леффингвелл Д. Требования в Agile: практики для команд и предприятий. — М.: Вильямс, 2011.
14. Листер Т., ДиМарко Т. Люди и проекты: продуктивные команды. — М.: Вильямс, 2006.
15. Марк Прайс. «C# 10 и .NET 6. Современная кроссплатформенная разработка».
16. Поппендик М., Поппендик Т. Бережливая разработка ПО: инструментарий Agile. — М.: Вильямс, 2003.
17. Рис Э. Бережливый стартап: как современные предприниматели используют непрерывные инновации. — М.: Олимп-Бизнес, 2011.
18. Рубин К. S. Важнейшие практики SCRUM — М.: Вильямс, 2013.
19. Соммервилл И. Инженерия программного обеспечения. — М.: Питер, 2015.
20. Спинеллис Д. Качество кода: взгляд открытого сообщества. — М.: Вильямс, 2012.
21. Хайсмит Дж. Управление проектами в Agile: создание инновационных продуктов. — М.: Вильямс, 2004.
22. Швабер К., Сазерленд Дж. Руководство Scrum. — 2020. — URL: https://scrumguides.org/

**Приложение А**

**(обязательное)**

# Графическая часть

**Приложение Б**

**(Обязательное)**

# Техническое задание

**Приложение В**

**(Обязательное)**

# Руководство системного администратора

**Приложение Г**

**(обязательное)**

# Руководство пользователя

**Приложение Д**

**(обязательное)**

# Фрагмент листинга программы