**Содержание**

[Введение 3](#_Toc137416877)

[1 Общая часть 4](#_Toc137416878)

[1.1 Описание предметной области 4](#_Toc137416879)

[1.2 Анализ существующей ситуации 6](#_Toc137416880)

[1.3 Постановка задачи 7](#_Toc137416881)

[1.4 Анализ существующих разработок и обоснование необходимости разработки……… 8](#_Toc137416882)

[2 Специальная часть 10](#_Toc137416883)

[2.1 Выбор технологий и инструментальных средств 10](#_Toc137416884)

[2.1.1 Выбор подхода к разработке 10](#_Toc137416885)

[2.1.2 Выбор среды разработки и языка программирования 11](#_Toc137416886)

[2.2 Разработка спецификаций 12](#_Toc137416887)

[2.2.1 Разработка диаграмм вариантов использования 12](#_Toc137416888)

[2.2.2 Разработка диаграмм последовательностей системы 14](#_Toc137416889)

[2.2.3 Разработка диаграммы пакетов 15](#_Toc137416890)

[2.2.4 Разработка диаграмм классов 15](#_Toc137416891)

[2.2.5 Разработка диаграммы состояний 17](#_Toc137416892)

[2.2.6 Разработка диаграммы деятельности 18](#_Toc137416893)  
2.2.7 Разработка диаграммы размещения………………………………….18

[2.2.8 Разработка инфологической модели базы данных 19](#_Toc137416894)

[2.2.9 Разработка физической модели базы данных 20](#_Toc137416895)

[2.3 Проектирование программного модуля 20](#_Toc137416896)

[2.3.1 Разработка алгоритмов реализации основных функций программного обеспечения…………. 20](#_Toc137416897)

[2.3.2 Проектирование пользовательского интерфейса 21](#_Toc137416898)

[2.4 Реализация программного обеспечения на выбранном языке программирования и в выбранной среде разработки 23](#_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 Безопасность жизнедеятельности 32](#_Toc137416907)

[3.1 Требования к рабочим помещениям и рабочим местам оператора ПК 32](#_Toc137416908)

[3.2 Расчет искусственной освещенности рабочего места оператора ПК 33](#_Toc137416909)

[3.3 Техника безопасности при работе на ПК 35](#_Toc137416910)

[Заключение 38](#_Toc137416911)

[Библиография 39](#_Toc137416912)

# Введение

Современные организации сталкиваются с необходимостью эффективного учета работы сотрудников. Автоматизированные системы учета рабочего времени и анализа трудовой деятельности позволяют не только повысить прозрачность учета, но и оптимизировать процессы управления персоналом.

Тема разработки системы учета работы сотрудников является актуальной в настоящее время, так как многие компании стремятся к повышению эффективности работы, автоматизации процессов и сокращению издержек. Традиционные методы учета, такие как бумажные журналы или ручное внесение данных, подвержены человеческому фактору и могут приводить к неточностям. Внедрение автоматизированной системы учета работы сотрудников позволяет решить эти проблемы, обеспечивая точность данных, удобство их анализа и снижение административной нагрузки.

Разработка такой системы позволит руководителям контролировать рабочее время сотрудников, анализировать их активность, оптимизировать распределение задач и повысить общую производительность компании. Кроме того, система будет полезна как для работодателей, так и для сотрудников, обеспечивая прозрачность рабочего процесса и возможность объективной оценки труда.

В настоящее время автоматизация учета рабочего времени является одним из ключевых направлений в области управления персоналом. Многие компании внедряют подобные системы, что повышает конкурентоспособность бизнеса и улучшает внутренние процессы. Разработка и внедрение такой системы является важным шагом в повышении эффективности работы организации.

Цель данной дипломной работы – разработать систему учета работы сотрудников для АО "ФосАгро". Для ее реализации необходимо выполнить следующие задачи:

1) изучить существующие методы и системы учета рабочего времени;

2) провести анализ предметной области и выявить требования к системе;

3) разработать алгоритмы учета рабочего времени и анализа трудовой активности;

4) спроектировать программное обеспечение;

5) реализовать систему;

6) провести тестирование и апробацию.

Внедрение данной системы обеспечит АО "ФосАгро" дополнительное конкурентное преимущество, повысит эффективность управления персоналом и создаст удобные механизмы контроля рабочего времени сотрудников. Кроме того, разработка подобного решения соответствует современным тенденциям цифровизации и автоматизации бизнес-процессов, что делает его востребованным на рынке информационных технологий.

# Общая часть

## Описание предметной области

Предметной областью данного программного обеспечения является учет рабочего времени и трудовой активности сотрудников в организациях. Система учета работы сотрудников охватывает процессы фиксации начала и окончания рабочего дня, учета времени выполнения задач, анализа загруженности и эффективности персонала.

В условиях современной корпоративной среды важно обеспечить прозрачность и точность учета рабочего времени. Это необходимо как для соблюдения трудового законодательства, так и для повышения внутренней дисциплины, улучшения планирования и управления человеческими ресурсами.

Организация учета работы сотрудников включает в себя разработку регламентов и правил фиксации рабочего времени, определение критериев продуктивности, внедрение технических решений для автоматического сбора и анализа данных, а также обеспечение информационной безопасности.

Одной из ключевых проблем в данной предметной области является недостаточная точность и достоверность учета рабочего времени при использовании ручных или устаревших методов. Это может приводить к потере производительности, снижению мотивации сотрудников и трудностям в управлении персоналом.

Разработка автоматизированной системы учета работы сотрудников позволяет решить эту проблему. Такая система обеспечивает точный сбор данных, минимизирует влияние человеческого фактора и предоставляет руководству инструменты для анализа и принятия управленческих решений на основе объективной информации.

Еще одной задачей, которую решает данное ПО, является повышение эффективности распределения задач и контроль за их выполнением. Благодаря интеграции с внутренними сервисами компании, система может автоматически фиксировать рабочую активность и своевременно информировать руководство о потенциальных перегрузках или недозагрузке отдельных сотрудников.

Также подобные системы актуальны для крупных промышленных предприятий, таких как АО «ФосАгро», где большое количество сотрудников работает в сменном графике или на удаленных производственных объектах. Автоматизация учета рабочего времени помогает улучшить планирование смен, учет выработки и контроль соблюдения графиков.

В целом, предметная область данного ПО — это управление трудовыми ресурсами, учет и анализ рабочего времени, а также автоматизация процессов, связанных с контролем и оценкой работы сотрудников. Система будет активно использоваться в крупных компаниях, стремящихся к цифровизации внутренней отчетности и повышению эффективности трудовой деятельности.

## Анализ существующей ситуации

Существующая ситуация в области учета рабочего времени и деятельности сотрудников значительно варьируется в зависимости от размера организации, уровня цифровизации и отраслевых особенностей. Тем не менее, можно выделить ряд общих проблем, которые могут быть эффективно решены с помощью автоматизированной системы учета работы сотрудников:

Недостаточная точность учета рабочего времени. Во многих организациях до сих пор используются устаревшие методы учета, такие как бумажные журналы или ручной ввод данных. Это делает систему уязвимой к ошибкам и злоупотреблениям, снижает достоверность информации и затрудняет анализ.

Человеческий фактор. При отсутствии автоматизации сотрудники могут забывать вовремя отмечать приход или уход, а руководители — своевременно проверять и корректировать данные. Это приводит к несоответствию отчетности фактической ситуации и создает дополнительные трудности при расчете заработной платы или премий.

Сложности в управлении рабочими графиками. В больших компаниях с посменным графиком или гибким режимом работы сложно эффективно координировать и контролировать занятость персонала. Возникают трудности с заменами, переработками и соблюдением трудового законодательства.

Низкая прозрачность. Отсутствие единого цифрового инструмента для отслеживания активности сотрудников затрудняет объективную оценку производительности, выявление проблемных зон и принятие управленческих решений.

Высокая административная нагрузка. Ручной учет требует участия большого количества сотрудников отдела кадров и бухгалтерии, что увеличивает издержки и снижает оперативность обработки информации.

В целом, автоматизированная система учета работы сотрудников способна значительно улучшить ситуацию. Она обеспечит точность и своевременность учета, снизит влияние человеческого фактора, оптимизирует управление графиками и сделает процессы более прозрачными. Это особенно актуально для крупных предприятий, таких как АО «ФосАгро», где эффективность управления персоналом напрямую влияет на производственные показатели и конкурентоспособность компании .

## Постановка задачи

Целью данной дипломной работы является разработка программного обеспечения (ПО) для учета работы сотрудников на предприятии АО «ФосАгро». Разрабатываемая система должна обеспечивать автоматизированный учет рабочего времени, регистрацию начала и окончания рабочего дня, а также возможность анализа трудовой активности сотрудников.

Ключевыми требованиями к программному обеспечению являются:

1. удобный пользовательский интерфейс для взаимодействия сотрудников и администраторов;
2. надежная регистрация событий (приход, уход, перерывы и т.п.);
3. хранение и обработка данных в структурированном формате;
4. возможность формирования отчетов по сотрудникам за выбранный период времени;
5. поддержка прав доступа с разграничением ролей пользователей (например, сотрудник, руководитель, администратор).

Разработка будет вестись с использованием языка программирования C# и технологии Windows Forms или WPF, что обеспечит удобство в реализации пользовательского интерфейса и взаимодействия с системой. В процессе работы необходимо спроектировать архитектуру приложения, реализовать все необходимые функции учета и анализа данных, а также провести тестирование системы на предмет корректности и устойчивости работы.

Создание подобной системы позволит повысить точность учета рабочего времени, снизить административную нагрузку и обеспечить прозрачность трудовой деятельности сотрудников.

1.4 Анализ существующих разработок и обоснование необходимости разработки

На рынке существует множество программных решений для учета рабочего времени и анализа трудовой деятельности сотрудников. Некоторые из них предоставляют базовые функции регистрации прихода и ухода сотрудников, в то время как другие включают более сложные возможности: ведение статистики, интеграцию с системой расчета заработной платы, контроль выполнения задач и отчетность.

Примерами таких решений являются:

1С:Управление персоналом — система, включающая модули для учета рабочего времени, ведения кадрового учета и расчета заработной платы. Однако её внедрение и настройка требуют значительных ресурсов и часто избыточны для задач локального контроля трудовой активности.

ТаймДоктор (Time Doctor) — облачное решение, ориентированное в первую очередь на компании с удаленными сотрудниками. Оно предлагает функционал мониторинга активности, но его использование требует постоянного интернет-соединения и может вызывать негативную реакцию со стороны персонала из-за высокого уровня контроля.

StaffCop — система мониторинга сотрудников с возможностью учета времени, но она ориентирована на корпоративный сектор с усиленными требованиями к безопасности и аудиту действий пользователей.

Kaspersky Security for Business — включает функции аудита времени и активности, но основной упор делается на информационную безопасность, а не на управление персоналом.

Несмотря на наличие таких решений, необходимость в разработке собственного программного обеспечения для АО «ФосАгро» обусловлена рядом факторов:

1. потребностью в адаптации под конкретные внутренние процессы и структуру организации;
2. желанием избежать лишних затрат на лицензии и обслуживание сторонних систем;
3. отсутствием избыточного функционала, не востребованного в рамках текущих задач;
4. необходимостью интеграции с уже существующими внутренними сервисами и учетными системами.

Таким образом, разработка собственной системы учета работы сотрудников позволит обеспечить гибкость, соответствие внутренним требованиям предприятия, снижение издержек и удобство использования. Это решение также будет способствовать повышению прозрачности учета и эффективности управления персоналом в АО «ФосАгро».

# Специальная часть

## Выбор технологий и инструментальных средств

### 2.1.1 Выбор подхода к разработке

Более подходящим вариантом для разработки автоматизированной системы учета и анализа работы сотрудников является объектно-ориентированный подход.

При создании такого программного обеспечения необходимо обеспечить четкое разделение ответственности между различными компонентами системы: модулем авторизации пользователей, базой данных с записями о трудовой активности, интерфейсом администратора, системой учета задач и временем их выполнения, а также механизмами формирования отчетов. Важно, чтобы архитектура позволяла при необходимости легко добавлять новые функции — например, поддержку дополнительных форм аналитики, интеграцию с кадровыми или бухгалтерскими системами предприятия.

Объектно-ориентированный подход позволяет выделить ключевые сущности (например, Сотрудник, Задача, Рабочий день, Отчет) и описать их в виде классов с четко определенными свойствами и методами. Это обеспечивает структурированность кода, упрощает повторное использование компонентов, повышает читаемость и сопровождаемость программного продукта.

Структурный подход, напротив, может привести к дублированию логики, усложнению взаимодействия между модулями и снижению гибкости при доработке. При изменении бизнес-требований или масштабировании системы на другие подразделения предприятия поддержка такой архитектуры может стать затруднительной.

Объектно-ориентированное проектирование облегчает модульное тестирование, что особенно важно при проверке корректности учета рабочего времени и формирования отчетных документов. Кроме того, такой подход способствует созданию масштабируемой и адаптивной системы, которая способна развиваться вместе с потребностями предприятия.

Таким образом, объектно-ориентированный подход является наиболее рациональным решением для разработки системы учета и анализа работы сотрудников в АО «ФосАгро», обеспечивая гибкость, расширяемость и надежность разрабатываемого программного обеспечения.

### 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 учета работы сотрудников.

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 учета работы сотрудников можно отметить:

Высокая скорость разработки. Большое количество доступных функций и удобство Visual Studio Community 2022 позволяет быстро создавать и отлаживать код.

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.

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 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.

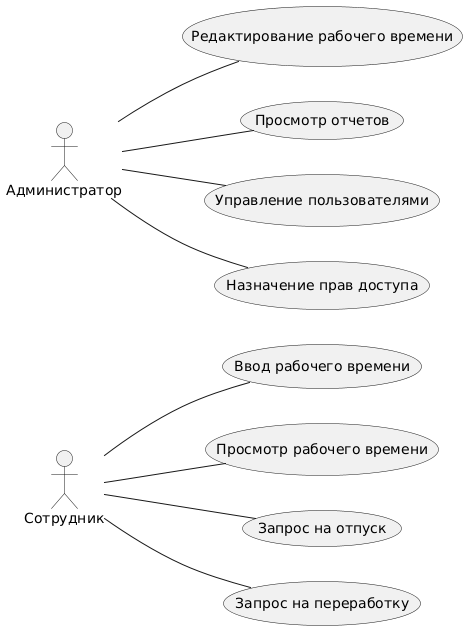


Рисунок 1 – Диаграмма вариантов использования

На диаграмме представлены следующие варианты использования:

«Ввод рабочего времени» – сотрудник вводит данные о своем рабочем времени, включая часы начала и окончания работы.

«Просмотр рабочего времени» – сотрудник может просматривать свои рабочие часы за определенный период.

«Запрос на отпуск» – сотрудник подает запрос на отпуск, который затем отправляется на одобрение.

«Запрос на переработку» – сотрудник подает запрос на переработку рабочего времени, который также требует одобрения.

«Редактирование рабочего времени» – администратор может редактировать рабочее время сотрудников в случае ошибок или изменений.

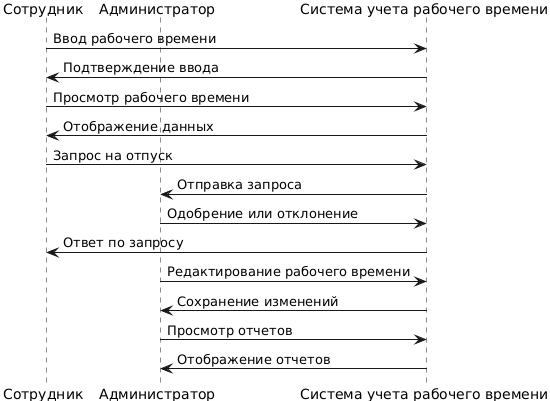
«Просмотр отчетов» – администратор просматривает отчеты о рабочем времени сотрудников, включая переработки и отпуска.

«Управление пользователями» – администратор добавляет, редактирует или удаляет пользователей системы, в том числе сотрудников.

«Назначение прав доступа» – администратор назначает права доступа для различных пользователей, включая настройку прав на редактирование или просмотр данных..

### 2.2.2 Разработка диаграмм последовательностей системы

Для данной разработки построена диаграмма пакетов, которая показана ниже на рисунке 2. Она показывает зависимости между частями.



### 2.2.3 Разработка диаграммы пакетов

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



Рисунок 3 – Диаграмма пакетов

### 2.2.4 Разработка диаграмм классов

Для разработки пакета для взаимодействия создана диаграмма классов, показанная ниже на рисунке 4.

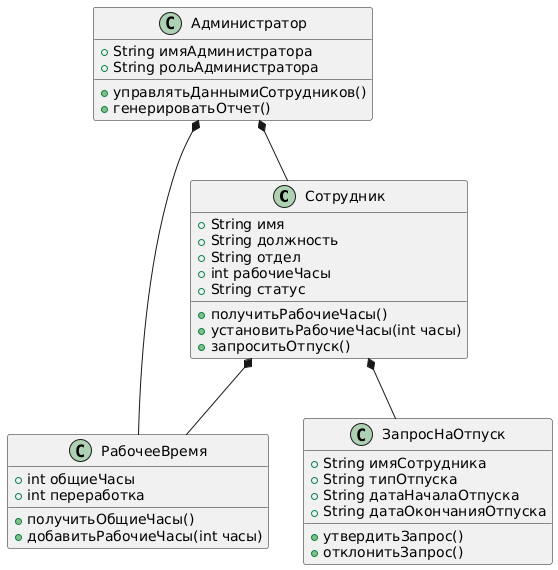


Рисунок 4 – Диаграмма классов

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

### 2.2.5 Разработка диаграммы состояний

### 

Рисунок 5 – Диаграмма состояний

Диаграмма состоит из трех состояний:

Ожидание начала рабочего дня — начальное состояние, в котором система ожидает, когда сотрудник отметит начало работы.

Рабочий день — активное состояние, в котором система регистрирует выполнение сотрудником своих обязанностей.

Завершение рабочего дня — конечное состояние, когда сотрудник завершает свой рабочий день и система регистрирует его завершение.

Переход между состояниями происходит следующим образом:

Сотрудник начинает рабочий день — система переходит из состояния «Ожидание начала рабочего дня» в состояние «Рабочий день».

В конце дня сотрудник завершает работу, и система переходит в состояние «Завершение рабочего дня».

Данная диаграмма позволяет наглядно представить жизненный цикл одного рабочего дня сотрудника и дает общее представление о логике учета рабочего времени в системе..

### 2.2.6 Разработка диаграммы деятельности

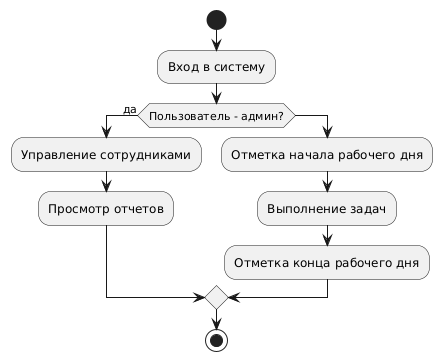


Рисунок 6 – Диаграмма деятельности.

Данная диаграмма деятельности описывает последовательность действий пользователей в системе учета работы сотрудников. Пользователь входит в систему, после чего в зависимости от своих прав выполняет определенные действия. Администратор управляет данными сотрудников и просматривает отчеты. Сотрудник отмечает начало и конец рабочего дня, а также фиксирует выполнение задач. Диаграмма отражает базовую логику поведения пользователей внутри системы и помогает наглядно представить процессы учета рабочего времени.

### 2.2.7 Разработка диаграммы размещения

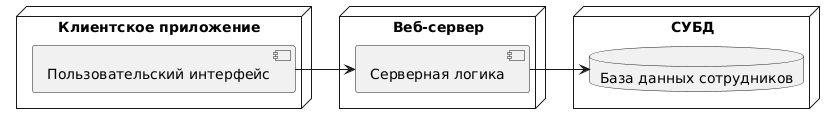


Рисунок 7 – Диаграмма размещения.

### 2.2.8 Разработка инфологической модели б азы данных

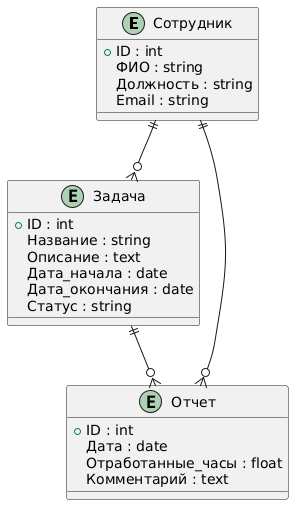


Рисунок 8 – Инфологическая модель базы данных.

### 2.2.9 Разработка физической модели базы данных

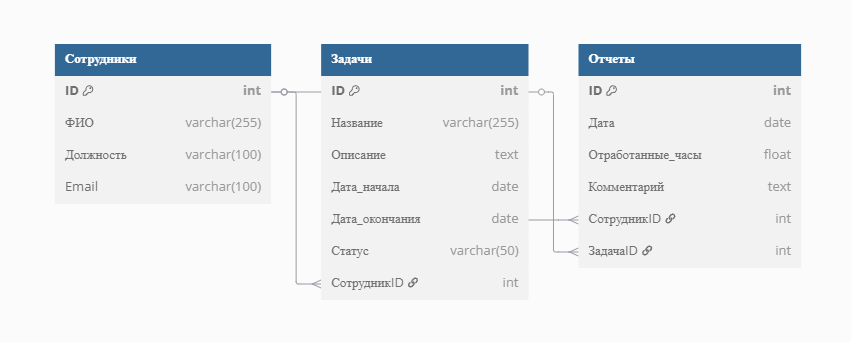


Рисунок 9 – Физическая модель базы данных.

## Проектирование программного модуля

### 2.3.1 Разработка алгоритмов реализации основных функций программного обеспечения

Алгоритм начинается с запуска программного обеспечения.

После запуска пользователь проходит процесс аутентификации в системе (администратор или сотрудник).

В зависимости от типа пользователя отображается соответствующий интерфейс.

Если вошел администратор, он получает доступ к функциям добавления, редактирования и удаления информации о сотрудниках, а также к назначению задач и просмотру отчетности.

При выборе функции назначения задачи — система предлагает выбрать сотрудника, ввести описание задачи, сроки выполнения и сохранить данные в базу.

При просмотре отчетов администратор получает информацию о выполненных задачах и отработанном времени каждого сотрудника.

Если вошел сотрудник, он может просматривать назначенные ему задачи, добавлять отчеты о проделанной работе (указывая дату, часы и комментарий), а также редактировать свои отчеты при необходимости.

По завершении всех действий данные сохраняются в базе данных, и система возвращается к начальному состоянию — ожиданию следующего входа пользователя.

### 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 автоматизированной системы учета работы сотрудников можно использовать различные среды разработки, такие как Microsoft Visual Studio, JetBrains Rider или SharpDevelop.

Разработка программы в среде Visual Studio включает следующие основные шаги:

1. Создание проекта.

В меню выбирается пункт "Создать проект", затем выбирается тип проекта, например, "Windows Forms App" или "ASP.NET Web Application", в зависимости от того, требуется ли настольное или веб-приложение.

2. Добавление необходимых файлов.

В проект добавляются файлы, отвечающие за авторизацию пользователей, отображение интерфейса администратора и сотрудников, взаимодействие с базой данных, хранение информации о задачах и отчетах по работе.

3. Написание кода.

Код программы на языке C# реализует следующие функции: регистрация и вход пользователей, назначение задач, ведение отчетности, а также управление списком сотрудников и анализ отработанного времени.

Примеры возможных функций:

// Добавление нового сотрудника

private void AddEmployee(string name, string position)

{

// добавление сотрудника в базу данных

}

// Назначение задачи сотруднику

private void AssignTask(int employeeId, string taskDescription, DateTime dueDate)

{

// сохранение задачи в базе данных

}

// Внесение отчета о проделанной работе

private void SubmitReport(int employeeId, DateTime date, double hoursWorked, string comment)

{

// добавление записи в таблицу отчетов

}

4. Тестирование и отладка.

После написания кода необходимо провести тестирование работы программы: проверка авторизации, назначение задач, добавление и редактирование отчетов, работа с базой данных. Для этого можно использовать тестовые данные.

5. Компиляция и сборка.

При успешном прохождении всех этапов тестирования, приложение компилируется. В случае Windows-приложения — получается исполняемый файл .exe.

6. Развертывание.

Если система предназначена для корпоративного использования, приложение можно развернуть на локальных компьютерах сотрудников или на сервере. В случае веб-приложения — требуется размещение на веб-сервере и настройка доступа пользователей.

Таким образом, реализация программного обеспечения на языке C# для системы учета работы сотрудников включает в себя весь цикл — от проектирования интерфейса до развертывания и поддержки, обеспечивая стабильную работу и удобство использования.

## 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 системы учета работы сотрудников. 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. Он может быть использован для тестирования приложений, написанных на языке C#.

Таким образом, для тестирования данного 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.

### 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.

# 3 Безопасность жизнедеятельности

## 3.1 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.

## 3.2 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.

## 3.3. 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

Разработка автоматизированной системы для учета работы сотрудников в АО «Апатит» позволила эффективно решить задачи, связанные с оптимизацией рабочего процесса и улучшением учета рабочего времени. В рамках дипломной работы была создана система, которая позволяет администрировать рабочие часы сотрудников, контролировать их занятость и повышать общую продуктивность через автоматизацию процесса учета.

Анализ существующих решений показал, что многие из них не учитывают специфику бизнес-процессов компании. Разработанная система решает этот вопрос за счет гибкости настройки и интеграции с внутренними системами. Использование объектно-ориентированного подхода и возможности масштабирования делают систему легко адаптируемой под любые изменения в бизнес-логике.

Перспективы развития проекта включают в себя интеграцию с другими корпоративными системами для более глубокого анализа данных и автоматизации отчетности, а также добавление функционала для учета дополнительных факторов, таких как производственные задачи и временные приоритеты.

Таким образом, успешная реализация системы для учета работы сотрудников подтверждает её высокую эффективность и потенциал для дальнейшего развития в условиях динамично меняющихся требований в сфере управления персоналом.

# Библиография

1. Агальцов, В. П. Базы данных. Microsoft SQL Server 2016. Учебный курс / В. П. Агальцов. — М.: ДМК Пресс, 2018. — 460 с.
2. Андреев, А. В. Проектирование пользовательских интерфейсов: учебное пособие / А. В. Андреев. — СПб.: БХВ-Петербург, 2021. — 320 с.
3. ГОСТ 2.105-95 ЕСКД. Общие требования к текстовым документам. — Введ. 1996-07-01. — М.: Стандартинформ, 2020. — 41 с.
4. ГОСТ Р 7.0.100-2018 Библиографическая запись. Библиографическое описание. — М.: Стандартинформ, 2018. — 52 с.
5. ГОСТ Р 50922-2006 Защита информации. Основные термины и определения. — М.: Стандартинформ, 2006. — 12 с.
6. Дейтел, П. Дж. C# для профессионалов: тонкости программирования / П. Дж. Дейтел, Х. М. Дейтел. — М.: Вильямс, 2021. — 1328 с.
7. Иванова, Г. С. Технологии программирования: учебник / Г. С. Иванова. — М.: КноРус, 2020. — 336 с.
8. Казаков, С. И. Разработка клиент-серверных приложений на C# / С. И. Казаков. — СПб.: Питер, 2019. — 288 с.
9. Климов, А. П. Автоматизация управления персоналом: современные решения / А. П. Климов // Управление корпоративной культурой. — 2022. — № 3. — С. 45-52.
10. Код безопасности: СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к ПЭВМ» [Электронный ресурс]. URL: http://docs.cntd.ru/document/901898464 (дата обращения: 13.04.2025).
11. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. — СПб.: Питер, 2022. — 352 с.
12. Официальная документация по .NET Framework [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/dotnet/ (дата обращения: 13.04.2025).
13. Петров, В. Н. UML-проектирование: от анализа к реализации / В. Н. Петров. — М.: ДМК Пресс, 2020. — 214 с.
14. Робачевский, А. М. SQL и реляционные базы данных: разработка запросов / А. М. Робачевский. — СПб.: БХВ-Петербург, 2021. — 416 с.
15. Скиена, С. Алгоритмы. Руководство по разработке / С. Скиена. — М.: Вильямс, 2019. — 720 с.
16. Троелсен, Э. Язык программирования C# 9.0 и платформа .NET 5 / Э. Троелсен. — М.: Вильямс, 2021. — 1328 с.
17. Фаулер, М. Рефакторинг: улучшение существующего кода / М. Фаулер. — СПб.: Питер, 2020. — 448 с.
18. Шилдт, Г. C# 10. Полное руководство / Г. Шилдт. — М.: Диалектика, 2022. — 800 с.