Министерство образования и науки РФ

ФГАОУ ВО «Уральский федеральный университет   
имени первого Президента России Б.Н. Ельцина»

Институт новых материалов и металлургии

Кафедра «Теплофизика и информатика в металлургии»

Оценка работы: \_\_\_\_\_\_\_\_\_\_\_\_\_

Члены комиссии:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись расшифровка подписи

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись расшифровка подписи

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Приложение для расчета зарплат сотрудников ресторана

**«Введение в технологию разработки программного обеспечения»**

Направление 09.03.02 «Информационные системы и технологии»   
(уровень бакалавриата)

Образовательная программа 09.03.02/03.01   
«Информационные системы и технологии в металлургии»

Руководитель

ст. преподаватель, к.т.н. А.С. Истомин

должность, звание подпись расшифровка подписи

Нормоконтролер

ст. преподаватель, к.т.н. А.С. Истомин

должность, звание подпись расшифровка подписи

Студенты:

НМТ-223901 В.C. Папшев

номер группы подпись расшифровка подписи

Т.A. Озорнин

подпись расшифровка подписи

М.И. Платонова

подпись расшифровка подписи

Екатеринбург

2023

**РЕФЕРАТ**

Пояснительная записка изложена на … листах и содержит … таблицу, … рисунков и … приложение.

СВЕРХУРОЧНЫЕ, ПРОГРАММИРОВАНИЕ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, АРХИТЕКТУРА, ИНТЕРФЕЙС, VISUAL STUDIO C#, ИНСТАЛЛЯЦИЯ.

Проект по модулю посвящен разработке программного обеспечения на платформе Visual Studio с использованием языка C#.

Отражены основные этапы разработки программного обеспечения: постановка задачи, реализация тестового варианта расчета в электронных таблицах Microsoft Excel; проектирование и реализация программного средства – математической библиотеки и пользовательского интерфейса; разработка системы автоматизированного тестирования математической библиотеки; создание справочной документации; подготовка дистрибутива. Размещение исходного программного кода выполнено в системе удаленного контроля версий Bitbucket.

Основными функциями программного обеспечения является расчет зарплат сотрудников ресторана с учетом премий, штрафов, сверхурочных и т д.

Основные конечные пользователи программного обеспечения – работники ресторанов, администраторы, бухгалтеры.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc167796626)

[1 ПОСТАНОВКА ЗАДАЧИ, ПРОВЕРКА КОРРЕКТНОСТИ АЛГОРИТМА РАСЧЕТА И ОРГАНИЗАЦИЯ РАБОТЫ НАД ПРОЕКТОМ В СИСТЕМЕ YOUTRACK 5](#_Toc167796627)

[1.1 Физическая постановка задачи 5](#_Toc167796628)

[1.2 Математическая модель задачи 5](#_Toc167796629)

[1.3 Создание тестового варианта расчета в электронных таблицах Microsoft Excel 5](#_Toc167796630)

[2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 7](#_Toc167796631)

[2.1 Разработка архитектуры системы 7](#_Toc167796632)

[2.2 Создание программного обеспечения в системе управления версиями GitHub 8](#_Toc167796633)

[2.3 Разработка математической библиотеки 16](#_Toc167796634)

[2.4 Реализация пользовательского интерфейса 18](#_Toc167796635)

[2.5 Обработка исключительных ситуаций 23](#_Toc167796636)

[2.6 Создание справочной помощи 23](#_Toc167796637)

[2.7 Создание дистрибутива 25](#_Toc167796638)

[3 РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ МАТЕМАТИЧЕСКОЙ БИБЛИОТЕКИ 26](#_Toc167796639)

[3.1 Реализация системы 26](#_Toc167796640)

[4 ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 28](#_Toc167796641)

[4.1 Установка и настройка программного средства 28](#_Toc167796642)

[4.2 Функциональные возможности программного продукта 28](#_Toc167796643)

[4.3 Технология выполнения расчетов 28](#_Toc167796644)

[ЗАКЛЮЧЕНИЕ 29](#_Toc167796645)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 30](#_Toc167796646)

[ПРИЛОЖЕНИЕ 31](#_Toc167796647)

# ВВЕДЕНИЕ

На сегодняшний день большинство процессов управления коммерческих предприятий производится на автоматизированных системах расчета данных. В частности, в среде управления ресторанным бизнесом как никогда актуальными становятся различные приложения для расчета основных показателей бух учета, в особенности зарплат сотрудников. Данная альтернатива позволяет существенно упростить и ускорить процессы расчета с большим количеством данных, экономя людские и финансовые ресурсы предприятия.

Целью данной курсовой работы является изучение возможностей среды Visual Studio и создание системы расчета данных по зарплатам сотрудников.

Задачами данной работы являются: организация работы в системе YouTrack, создание математической библиотеки для расчета показателей, создание графического интерфейса приложения для удобного взаимодействия с функционалом и базой данных и программные тесты для подтверждения устойчивой работоспособности.

# 1 ПОСТАНОВКА ЗАДАЧИ, ПРОВЕРКА КОРРЕКТНОСТИ АЛГОРИТМА РАСЧЕТА И ОРГАНИЗАЦИЯ РАБОТЫ НАД ПРОЕКТОМ В СИСТЕМЕ YOUTRACK

## 1.1 Физическая постановка задачи

Разработать систему расчета зарплат сотрудников, состоящую из графического интерфейса, базы данных, математических библиотек и программных тестов.

## 1.2 Математическая модель задачи

Расчет итоговой зарплаты для каждого сотрудника осуществляется следующим образом: к базовой зарплате добавляется надбавка по сверхурочным и различные премии, затем из полученной суммы вычитается сумма штрафов, полученных сотрудником за месяц, а также вычитается сумма разности потерь и МРОТ по больничному периоду.

Базовая зарплата в свою очередь считается как произведение зарплаты за час на количество рабочих часов в день и на количество рабочих дней в неделе, после чего данная сумма умножается на 4 для получения помесячной зарплаты

## 1.3 Создание тестового варианта расчета в электронных таблицах Microsoft Excel

Тестовый вариант, созданный в приложении Excel имеет 5 таблиц:

1 таблица – Сотрудники, где представлена информация обо всех сотрудниках ресторана, включая ФИО, дату рождения и основную информацию о работе (время начала работы, оклад, количество рабочих дней, премии). (см рисунок 1.3.1)

Изображение выглядит как текст, снимок экрана, линия, число

Автоматически созданное описание

Рисунок 1.3.1 – Таблица «Сотрудники»

2 таблица – Премии. Здесь представлена информация обо всех готовых к получению премиях и их владельцах (см рисунок 1.3.2)

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 1.3.2 – Таблица «Премии»

3 таблица – Штрафы. Здесь представлена информация обо всех штрафах, имеющихся на данный момент и их владельцах. (см рисунок 1.3.3)

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

Рисунок 1.3.3 – Таблица «Штрафы»

4 таблица – Больничные, где представлена информация обо всех больничных периодах вместе с дополнительной информацией, представляющей данные для расчета выплат (конец, начало периода, вычитаемая сумма, количество пропущенных часов). (см рисунок 1.3.4)

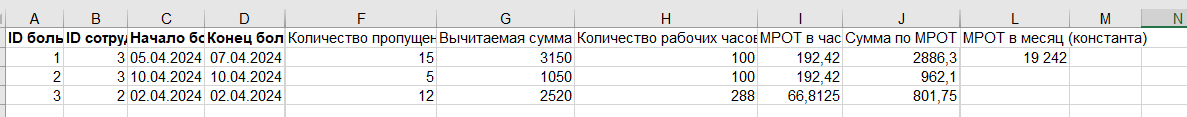


Рисунок 1.3.4 – Таблица «Больничные»

5 таблица – Сверхурочные часы, где представлена информация о количестве выработанных сверхурочных часов вместе с количеством надбавки за каждый час. (см рисунок 1.3.5)

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 1.3.5 – Таблица «Сверхурочные часы»

# 2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## 2.1 Разработка архитектуры системы

Архитектура данной системы содержит несколько ключевых элементов. 1 – приложение, которое в свою очередь содержит 3 компонента: пользовательский интерфейс (обеспечивает диалог между системой расчета и пользователем), математическую библиотеку (содержащую основные методы для обеспечения расчетов) и блок тестов (для проверки работоспособности системы в различных ситуациях), 2 – база данных SQL Server, где хранятся все основные данные, используемые приложением для расчета.

Приложение связано с базой данных с помощью утилиты EntityFramework, благодаря которой программа взаимодействует с классами объектов строк из базы с использованием математических библиотек.

Также в приложении может осуществляться функция выгрузки результатов в readonly форматы.

Схема архитектуры системы представлена на рисунке 2.1.1.

Изображение выглядит как текст, снимок экрана, диаграмма, Прямоугольник

Автоматически созданное описание

Рисунок 2.1.1 – Схема архитектуры системы

## 2.2 Создание программного обеспечения в системе управления версиями GitHub

Для организации процесса разработки в ходе работы использовалась система контроля версий под названием GitHub. В процессе работы с программой в нее добавлялись различные коммиты в соответствии с планом разработки. Для данного проекта существует единственная ветка мастер, внутри которой последовательно идет цепочка коммитов. Примеры коммитов данной ветки можно видеть на рисунке 2.2.1.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.2.1 – Ветка мастер

Сперва было создано новое решение в программе Visual Studio, в котором были созданы следующие проекты на языке C# с использованием .NET Core 6: настольное приложение с технологией Windows Forms, математическая библиотека и проект с тестами (см рисунок 2.2.2). Были установлены все нужные модули, такие как EntityFramework.

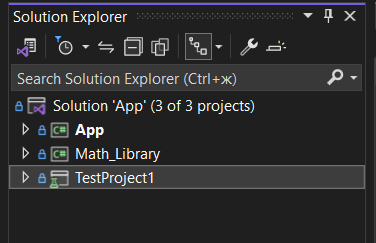


Рисунок 2.2.2 – Проекты в решении

Была настроена связь приложения с базой данных SQL Server с помощью встроенных инструментов EntityFramework (см рисунок 2.2.3).

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.2.3 – Файлы связи приложения с таблицами базы данных

Была создана форма аутентификации пользователя для входа в приложение (см рисунок 2.2.4).

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Автоматически созданное описание

Рисунок 2.2.4 – Форма входа

Для более безопасного использования приложения в базе данных хранятся хеши паролей администраторов вместо простого текста, поэтому был написан класс для хеширования и верификации паролей. Метод для верификации пароля можно видеть на рисунке 2.2.5. Данный метод используется в форме входа для входа в главное меню либо показа сообщении о неправильном пароле (см рисунок 2.2.6).

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 2.2.5 – Верификация пароля

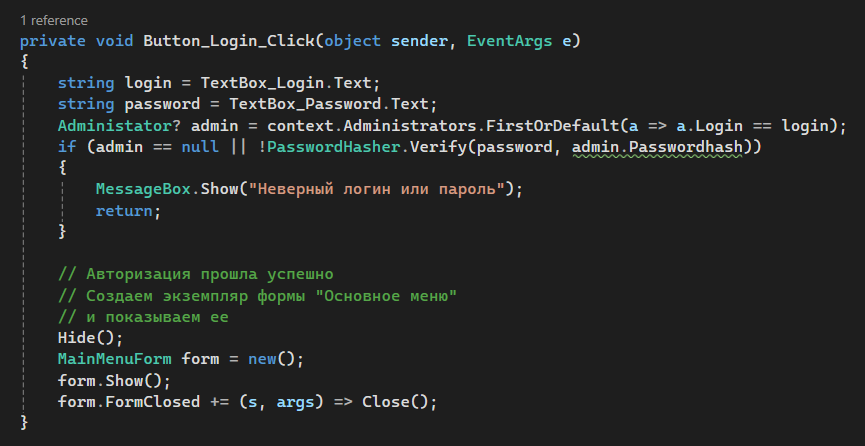


Рисунок 2.2.6 – Верификация пароля на форме

Была создана форма главного меню (см рисунок 2.2.7)

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.2.7 – Форма главного меню

Для вывода данных из базы данных в таблицу был создан BindingSource, который заполняется сразу на этапе инициализации формы (см рисунок 2.2.8).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 2.2.8 – Инициализация главного меню

Фильтрация работает так, что если текстовое поле содержит введённый пользователем текст, то для каждого сотрудника в базе данных проверяется, что текст в текстовом поле фильтрации содержится в соответствующем столбце таблицы базы данных в ячейке сотрудника, при этом регистр текста не учитывается. Для дат, если галочка рядом была поставлена, то она сравнивается с датой в базе данных через оператор «==» (см рисунок 2.2.9).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.2.9 – Фильтрация данных

Если пользователь изменил какую-либо информацию в таблице, то чтобы изменения сохранились, он должен нажать на кнопку «Сохранить изменения», которая вызывает метод context.SaveChanges() и показывает окно с сообщением «Изменения были успешно сохранены».

К столбцам были добавлены кнопки действий с сотрудником в базе данных, такие как: просмотр премий, сверхурочных часов, штрафов, больничных, подсчёт зарплаты и удаление из базы данных. Для того, чтобы эти кнопки работали, был создан новый метод, который подключается к событию DataGridView.CellContentClick и в котором находится сотрудник по ID и открывается нужная форма (см рисунок 2.2.10).

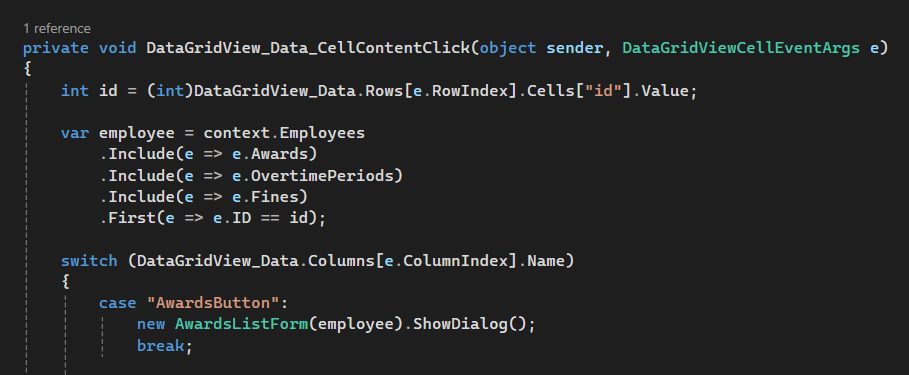


Рисунок 2.2.10 – Пример открытия формы премий из главного меню

Была создана форма для добавления сотрудников в базу данных (см рисунок 2.2.11). Для каждого текстового поля идёт проверка, чтобы оно содержало текст, в противном случае появляется окно с сообщением «Поле <название поля> не может быть пустым». Для числовых полей идёт проверка на содержание текста и успешную конвертацию этого текста в число (см рисунок 2.2.12). После успешной проверки всех данных создаётся новая ячейка в базе данных и изменения тут же сохраняются.

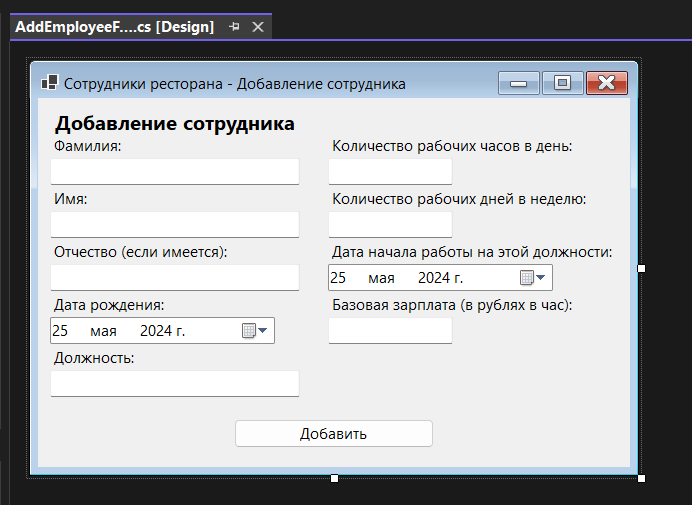


Рисунок 2.2.11 – Форма добавления сотрудника в базу данных

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 2.2.12 – Методы для проверки правильности написания данных

Была создана форма для просмотра премий сотрудника (см рисунок 2.2.13). Для того, чтобы инициализировать данную форму, нужно в конструктор класса подать объект сотрудника, из которого будет извлечены Ф.И.О. и все премии для таблицы (см рисунок 2.2.14).

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 2.2.13 – Форма просмотра премий сотрудника

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.2.14 – Инициализация просмотра премий сотрудника

Отличий в процессе фильтрации, добавления элементов, сохранения изменений между данным меню и главным меню нет.

Были созданы формы по типу формы просмотра премий сотрудника для просмотра сверхурочных часов, штрафов, больничных (см рисунок 2.2.15).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.2.15 – Формы для просмотра и изменения данных в базе данных о сотруднике

Была создана форма подсчёта зарплаты (см рисунок 2.2.16), в которой пользователю нужно выбрать месяц и год, для которого нужно сделать подсчёт зарплаты сотрудника, и которая обращается к математической библиотеке (см раздел 2.3).

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.2.16 – Форма добавления сотрудника в базу данных

Кнопки «Просмотр премий» и др. работают так, что при нажатии на них открывается соответствующая форма, которая использует специальный конструктор, фильтрующий данные по периоду с начала месяца по конец месяца. Пример данного конструктора для просмотра премий сотрудника можно видеть на рисунке 2.2.17.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 2.2.17 – Конструктор для фильтрации данных

## 2.3 Разработка математической библиотеки

Для обеспечения в приложении функционала расчета зарплат сотрудников была создана математическая библиотека под названием Math\_Library. В ней реализовано 3 класса: Formulas (см Рисунок 2.3.1), Employee (см Рисунок 2.3.3), SickPeriod (см Рисунок 2.3.3).

Класс Formulas содержит все формулы математической библиотеки и представляет собой набор основных методов, в которых соответственно реализован подсчет базовой зарплаты, расчет доплат по больничному периоду, расчет доплат за премии и итоговый расчет зарплат сотрудника.

Класс Employee содержит информацию о сотруднике, необходимую для расчётов, и представляет собой 3 поля, в которых соответственно хранится информация о зарплате за час, рабочих часах в день, рабочих дней в неделю.

Класс SickPeriod содержит информацию о больничном периоде, необходимую для расчётов, и представляет собой 2 поля, в которых хранится информация о дате начала и конца больничного периода. Конструктор класса соответственно принимает данные значения и присваивает их в эти переменные.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.3.1 – Метод из класса Formulas для подсчёта зарплаты

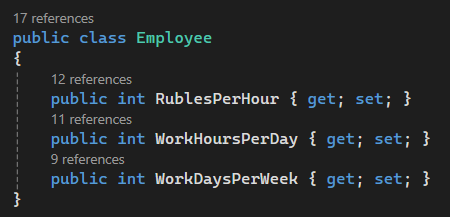


Рисунок 2.3.2 – Код класса Employee

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.3.3 – Код класса SickPeriod

## 2.4 Реализация пользовательского интерфейса

При открытии приложения пользователя встречает окно входа (см рисунок 2.4.1)

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 2.4.1 – Окно входа в приложение

После входа в аккаунт администратора пользователю открывается главное меню приложения (см рисунки 2.4.2, 2.4.3, 2.4.4)

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 2.4.2 – Главное меню приложения

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 2.4.3 – Просмотр премий, сверхурочных часов, штрафов сотрудников в главном меню

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 2.4.4 – Просмотр больничных периодов, подсчёт зарплаты и удаление сотрудника

В главном меню приложения можно изменять данные в базе данных (см рисунок 2.4.5), а также предусмотрена фильтрация и поиск данных (см рисунок 2.4.6).

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 2.4.5 – Изменение данных в приложении

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 2.4.6 – Фильтрация данных

В главном меню приложения можно добавлять информацию о новых сотрудниках ресторана (см рисунок 2.4.7).

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 2.4.7 – Добавление новых сотрудников в базу данных

Описанный функционал главного меню (изменение, фильтрация, добавление и удаление данных) доступен во всех меню приложения: просмотр премий (см рисунок 2.4.8), сверхурочных часов (см рисунок 2.4.9), штрафов (см рисунок 2.4.10) и больничных периодов сотрудника (см рисунок 2.4.11).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 2.4.8 – Просмотр премий сотрудника

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 2.4.9 – Просмотр сверхурочных часов сотрудника

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 2.4.10 – Просмотр штрафов сотрудника

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.4.11 – Просмотр больничных периодов сотрудника

Предусмотрен функционал для подсчёта зарплаты выбранного сотрудника за выбранный месяц (см рисунок 2.4.12). При нажатии на кнопку «Просмотр ...» после подсчёта зарплаты можно увидеть информацию о премиях, сверхурочных часах, штрафах и больничных за этот месяц (см рисунок 2.4.13).

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2.4.12 – Подсчёт зарплаты сотрудника

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 2.4.12 – Пример просмотра больничных периодов сотрудника за месяц

## 2.5 Обработка исключительных ситуаций

В каждом методе класса математической библиотеки реализован выброс исключений, связанный с входными данными. Полученные функцией параметры тщательно проверяются на не отрицательность. В противном случае пишется текст исключения вместе с информацией о типе и месте возникновения исключительной ситуации.

В приложении при вводе текстовых данных идёт проверка, чтобы в поле обязательно был текст. При вводе числовых данных помимо проверки содержания текст идёт проверка на успешное преобразование данных в числа.

## 2.6 Создание справочной помощи

Начнём со справочной помощи для IT отдела ресторана.

Для того, чтобы изменять и, при желании, дорабатывать приложение, нужны следующие компоненты: Microsoft Visual Studio, .NET Core 6, Microsoft SQL Server.

В SQL Server нужно подключиться к серверу ресторана (см рисунок 2.6.1), после чего восстановить базу данных из файла (см рисунок 2.6.2).

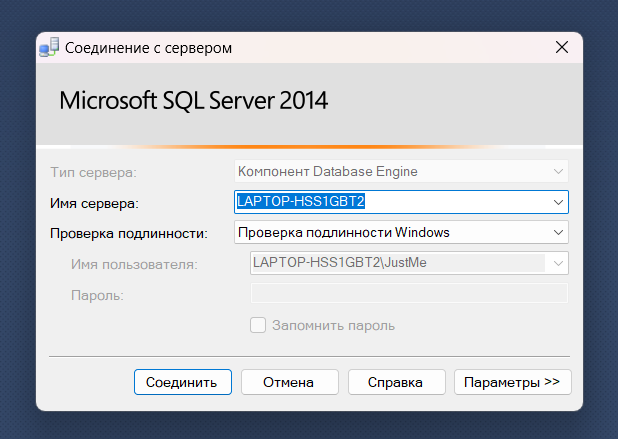


Рисунок 2.6.1 – Авторизация в SQL Server

Изображение выглядит как текст, электроника, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.6.2 – Восстановление базы данных в SQL Server

В файле “Data/RestaurantContext.cs” в методе “OnConfiguring” потребуется поменять строку подключения к базе данных[[1]](#footnote-1) (см рисунок 2.6.3). После компиляции приложения (см рисунок 2.6.4) оно будет готово к использованию персоналом ресторана.

Изображение выглядит как текст, линия, снимок экрана

Автоматически созданное описание

Рисунок 2.6.3 – Строка подключения к базе данных

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.6.4 – Компиляция приложения

Таблицы в базе данных связаны с приложением посредством моделей в Папке “Data”, и подключение к базе данных осуществляется с помощью контекста “RestaurantContext”, в котором хранятся ссылки на объекты типа “DbSet”, которые предоставляют доступ к базам данных.

Чтобы начать работу с базой данных, в каждой форме при инициализации создаётся объект context типа “RestaurantContext”, и когда нужно сохранить изменения, вызывается метод context.SaveChanges().

Каждая форма по работе с базой данных должна содержать следующий функционал: вывод данных в виде таблицы с возможностью изменения содержания, фильтрация и поиск данных по самым важным полям таблицы (по возможности, всем полям), кнопка добавления элементов, открывающая новую форму с вводом данных, кнопки «Удалить» как отдельный столбец таблицы вывода данных и кнопка «Сохранить изменения». Это нужно для того, чтобы обеспечить удобство пользования приложением сотрудникам ресторана и постоянство интерфейса.

Для аккаунтов администраторов используется хеширование паролей. Чтобы получить хеш пароля, можно использовать метод PasswordHasher.Hash, после чего вставить его в таблицу администраторов базы данных, если понадобится добавить новых администраторов.

Приступим к справочной информации для администраторов ресторана.

Работа интерфейса описана в разделе 2.4 пояснительной записки.

При входе в приложение в появившемся окне входа нужно ввести данные от аккаунта администратора. После входа в аккаунт администратора открывается главное меню приложения с информацией обо всех сотрудниках ресторана. Чтобы увидеть действия, которые можно произвести с информацией о конкретном сотруднике, можно пролистать таблицу с выводом вправо.

## 2.7 Создание дистрибутива

Для успешной работы приложения на компьютерах администраторов нужны лишь следующие элементы: Microsoft SQL Server с подключением к базе данных ресторана, справочная информация и само приложение.

Чтобы создать дистрибутив, IT отдел ресторана должен сначала скомпилировать исходный код приложения. После этого нужно создать архив с этими файлами, справочной информацией и, при необходимости, установочный файл SQL Server, если он находится не на всех компьютерах администраторов. Данный архив является готовым дистрибутивом, который можно использовать для начала работы с приложением.

# 3 РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ МАТЕМАТИЧЕСКОЙ БИБЛИОТЕКИ

## 3.1 Реализация системы

Для обеспечения тестирования корректности работы функций для каждого компонента математической библиотеки был создан свой индивидуальный тест, выполняемый в отдельном проекте.

Метод Salary, выполняющий итоговый подсчет зарплаты сотрудника с учетом всех факторов, проходит через ряд тестов в которых используются заведомо недействительные данные, включающие отрицательные значения для каждого компонента входных данных (даты конца и начала больничного периода, зарплата за час, количество рабочих часов в неделю, сверхурочные и т д), либо же происходит проверка на попадание в значений в необходимый диапазон. Пример теста формул для расчёта представлен на рисунке 3.1.1.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 3.1.1 - Пример теста формул для расчета.

Также в этом блоке происходит тестирование класса SickPeriod для больничных периодов путем сопоставления результата с заранее определенным диапазоном (см рисунок 3.1.2). Были написаны тесты для проверки преобразования информации о больничных периодах, которые хранятся в базе данных, в информацию о больничных периодах, которые используются в математической библиотеке (см рисунок 3.1.3).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 3.1.2 – Пример теста для больничных периодов

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 3.1.3 – Пример теста для больничных периодов математической библиотеки

# 4 ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## 4.1 Установка и настройка программного средства

Для работы приложения на компьютере администратора нужны только 2 элемента: Microsoft SQL Server с подключением к базе данных ресторана и само приложение.

После установки SQL Server нужно создать новое подключение и успешно подключиться к базе данных ресторана (см рисунок 2.6.1).

После успешного подключения к базе данных через SQL Server можно открыть приложение и начать им пользоваться.

## 4.2 Функциональные возможности программного продукта

Данная система может рассчитывать показатели зарплаты сотрудников в любом месяце и годе, учитывая все следующие элементы: премии, сверхурочные часы, штрафы, больничные дни – а также создавать подробный отчёт обо всех зарплатах сотрудников в формате таблицы Microsoft Excel.

## 4.3 Технология выполнения расчетов

# ЗАКЛЮЧЕНИЕ

В ходе данной работы мы создали автоматизированную систему расчета зарплат сотрудников ресторана, позволяющую быстро высчитать необходимые показатели и способную работать с большими объемами исходных данных, хранящихся в базе. Кроме того, мы научились работать с Visual Studio 2019 и системой контроля версий GitHub.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Марк Прайс. C# 10 и .NET 6. Современная кроссплатформенная разработка / Прайс Марк - «Издательский дом «Питер»», 2022
2. Эндрю Стиллмен. Изучаем C# / Эндрю Стиллмен, Дженнифер Грин - «Издательский дом «Питер»», 2021
3. Смит Джон П. Entity Framework Core в действии / Смит Джон П. 2023
4. Мурадханов, С. Э. Разработка на языке C# приложений с графическим интерфейсом (использование Windows Forms) : учебник / С. Э. Мурадханов. — Москва : МИСИС, 2019. — 396 с.
5. Бондарь А.Г. Microsoft SQL Server 2014 / А.Г. Бондарь — БХВ-Петербург, 2015
6. Лавров В.В. Методические указания к выполнению курсовой работы для студентов бакалавриата, обучающихся по направлению «Информационные системы и технологии» и студентов магистратуры, обучающихся по программе «Информационные системы в металлургии» / В.В. Лавров, К.А. Щипанов, А.А. Бурыкин – Екатеринбург: УрФУ, 2014. – 49 с.
7. Лошкарев Н.Б. Указания к оформлению дипломных и курсовых проектов и работ: методические указания / Н.Б. Лошкарев, А.Н. Лошкарев, Л.А. Зайнуллин. – Екатеринбург: УГТУ–УПИ, 2007. – 49 с.

# ПРИЛОЖЕНИЕ

Фрагменты листинга кода программного обеспечения

Листинг кода главного меню приложения:

using App.Data;

using Microsoft.EntityFrameworkCore;

using System.Data;

namespace App

{

public partial class MainMenuForm : Form

{

RestaurantContext context = new();

public MainMenuForm()

{

InitializeComponent();

ReloadData();

}

private void ReloadData()

{

employeeBindingSource.DataSource = context.Employees.ToList();

}

private bool FilterData(Employee employee)

{

// Проверка на имя, фамилию и отчество

if (employee.LastName == null ||

!employee.LastName.Contains(TextBox\_FilterLastName.Text, StringComparison.OrdinalIgnoreCase))

return false;

if (employee.FirstName == null ||

!employee.FirstName.Contains(TextBox\_FilterFirstName.Text, StringComparison.OrdinalIgnoreCase))

return false;

if (employee.MiddleName == null ||

!employee.MiddleName.Contains(TextBox\_FilterMiddleName.Text, StringComparison.OrdinalIgnoreCase))

return false;

// Проверка на роль

if (employee.Role == null ||

!employee.Role.Contains(TextBox\_FilterRole.Text, StringComparison.OrdinalIgnoreCase))

return false;

// Проверка на дату рождения, если галочка была поставлена

if (CheckBox\_FilterBirthday.Checked && employee.Birthday != DateTimePicker\_FilterBirthday.Value)

return false;

return true;

}

private void Button\_Filter\_Click(object sender, EventArgs e)

{

employeeBindingSource.DataSource = context.Employees.Where(FilterData).ToList();

}

private void Button\_Save\_Click(object sender, EventArgs e)

{

context.SaveChanges();

MessageBox.Show("Изменения были успешно сохранены.");

}

private void Button\_AddEmployee\_Click(object sender, EventArgs e)

{

new AddEmployeeForm().ShowDialog();

ReloadData();

}

private void DataGridView\_Data\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int id = (int)DataGridView\_Data.Rows[e.RowIndex].Cells["id"].Value;

var employee = context.Employees

.Include(e => e.Awards)

.Include(e => e.OvertimePeriods)

.Include(e => e.Fines)

.First(e => e.ID == id);

switch (DataGridView\_Data.Columns[e.ColumnIndex].Name)

{

case "AwardsButton":

new AwardsListForm(employee).ShowDialog();

break;

case "OvertimePeriodsButton":

new OvertimePeriodsListForm(employee).ShowDialog();

break;

case "FinesButton":

new FinesListForm(employee).ShowDialog();

break;

case "SickPeriodsButton":

new SickPeriodsListForm(employee).ShowDialog();

break;

case "CalculateSalaryButton":

new CalculateSalaryForm(employee).ShowDialog();

break;

case "DeleteButton":

var result = MessageBox.Show(

$"Вы уверены, что вы хотите удалить сотрудника \"{employee.GetFullName()}\" из базы данных?",

"Требуется подтверждение",

MessageBoxButtons.YesNoCancel);

if (result == DialogResult.Yes)

{

employee.Delete();

context.SaveChanges();

MessageBox.Show($"Сотрудник \"{employee.GetFullName()}\" был успешно удалён из базы данных.");

ReloadData();

}

break;

}

}

private void Button\_Report\_Click(object sender, EventArgs e)

=> new PrepareReportForm().ShowDialog();

}

}

1. https://metanit.com/sharp/entityframework/3.5.php?ysclid=lwp6f5vnf3915597625 [↑](#footnote-ref-1)