Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное   
учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**РЕАЛИЗАЦИЯ ПРОЕКТА   
«НАЧИСЛЕНИЕ ЗАРАБОТНОЙ ПЛАТЫ»**

Пояснительная записка к индивидуальному заданию

по дисциплине «Новые технологии в программировании»

Студент гр. 584-1

\_\_\_\_\_\_\_\_\_\_ А. А. Усольцева

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2017

Руководитель

к.н., доцент каф. КСУП

\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ А. Е. Горяинов

оценка «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2017

**Реферат**

Реферат: 17 страниц, 9 рисунков, 5 использованных источников.

MICROSOFT VISUAL STUDIO, C#, ЗАРАБОТНАЯ ПЛАТА, СОТРУДНИКИ, ПРОЕКТ, РАЗРАБОТКА

В ходе работы были изучены основные этапы разработки программного обеспечения на примере проекта «Начисление заработной платы».

В работе произведен обзор аналогов разрабатываемой программы. В соответствии с утвержденным техническим заданием были построены проект программы, включающий в себя диаграммы классов, диаграммы вариантов использования и рассмотрен макет графического интерфейса пользователя. Тестирование производилось посредством разработанных к проекту модульных тестов.

Отчет выполнен в программе Microsoft Word. Библиотека написана на языке программирования C# 5.0 для платформы .NET Framework 4.5 в IDE Visual Studio 2012 Ultimate. В разработке использовалась система контроля версий Git.

Оглавление

[1 Техническое задание 4](#_Toc502017533)

[1.1 Исходная проблема 4](#_Toc502017534)

[1.2 Цель и задачи 4](#_Toc502017535)

[1.3 Контекст использования и конечный пользователь 4](#_Toc502017536)

[1.4 Критерии качества 5](#_Toc502017537)

[1.5 Обзор аналогов 5](#_Toc502017538)

[1.6 Дополнительные требования 6](#_Toc502017539)

[1.7 Описание функциональных возможностей программы 6](#_Toc502017540)

[1.8 Прототип пользовательского интерфейса 7](#_Toc502017541)

[1.9 План разработки проекта и ресурсы 10](#_Toc502017542)

[2 Введение 11](#_Toc502017543)

[3 Описание программной системы 12](#_Toc502017544)

[3.1 UML диаграмма сценариев использования 12](#_Toc502017545)

[3.2 UML диаграмма классов 13](#_Toc502017546)

[3.3 Тестирование программы 14](#_Toc502017547)

[3.4 Версионный контроль 15](#_Toc502017548)

[4 Заключение 16](#_Toc502017549)

[Список использованных источников 17](#_Toc502017550)

# 1 Техническое задание

## 1.1 Исходная проблема

На любом предприятии, где работает больше, чем один человек, приходит время, когда следует рассчитать, сколько следует выплатить определенному сотруднику за проделанную за последний месяц работу. Расчет способа начисления заработной платы и ее размеры обговариваются при приеме на работу и утверждаются документально. Способ начисления заработной платы бывает почасовой и с окладом.

Проблемой в данном процессе является ведение кадрового учета.

## 1.2 Цель и задачи

Целью данного индивидуального задания является реализация простого проекта для расчета заработной платы сотрудников различными способами ее начисления: почасовая и с окладом.

Задачи, подлежащие решению:

– создание бизнес-логики;

– создание пользовательского интерфейса;

– юнит-тестирование;

– сборка установщика;

– написание технической документации.

## 1.3 Контекст использования и конечный пользователь

Поскольку данное работа является простым проектом и не требует особых знаний и навыков для использования, помимо знания английского языка, то ее можно отнести к продуктам широкого применения.

## 1.4 Критерии качества

В разрабатываемом программном обеспечении должны быть реализованы следующие функциональные требования:

1. Выводить диалоговое окно с отображением таблицы сотрудников, с возможностью добавления новых сотрудников, а также удаления, редактирования уже имеющихся.

2. Проводить проверку вводимых данных на корректность.

3. При верно введенных данных производить на их основе добавление в таблицу нового сотрудника.

4. В случае ввода некорректных данных выводить сообщение об ошибке.

## 1.5 Обзор аналогов

Ведение кадрового учета реализовано в подсистеме «1С: Зарплата и кадры».

Программа «1С:Зарплата и кадры бюджетного учреждения 8» – это мощный инструмент для комплексной автоматизации расчета заработной платы и ведения кадрового учета в государственных (муниципальных) учреждениях, состоящих на самостоятельном балансе, финансируемых из федерального, регионального (субъектов Российской Федерации) или местного бюджета, а также из бюджета государственного внебюджетного фонда, по следующим направлениям:

– расчет заработной платы с поддержкой новых систем оплаты труда работников федеральных бюджетных учреждений;

– расчет денежного довольствия военнослужащих и приравненных к ним лиц;

– расчет денежного содержания служащих на государственной гражданской службе;

– исчисление регламентированных законодательством налогов и взносов с фонда оплаты труда;

– отражение начисленной зарплаты и налогов в расходах учреждения;

– управление денежными расчетами с работниками, включая депонирование;

– учет кадров и анализ кадрового состава;

– автоматизация кадрового делопроизводства;

– набор кадров [1].

Основным недостатком данного инструмента является то, что это платный к использованию продукт.

## 1.6 Дополнительные требования

Инструменты разработки:

– язык программирования C# 5.0, .NET Framework 4.5:

– IDE Visual Studio 2012 Ultimate;

– система контроля версий git.

Содержание пояснительной записки:

– титульный лист;

– реферат;

– содержание ПЗ;

– введение;

– постановка и анализ задачи;

– описание программы для пользователя;

– тестирование программы;

– заключение;

– список литературы;

– приложения.

## 1.7 Описание функциональных возможностей программы

Разработанная программа должна представлять собой пользовательскую форму с таблицей для отражения рассчитанных параметров; кнопки позволяют управлять объектами этой системы.

Программа должна обеспечить выполнение следующих функций:

– добавление новых сотрудников с помощью формы создания нового сотрудника;

– удаление данных выбранного сотрудника;

– редактирование данных сотрудника;

– возможность загружать и сохранять данные в систему.

Входными данными являются следующие:

– имя и фамилия сотрудника;

– выбранный способ расчета заработной платы:

– почасовая: стоимость часа работы, количество отработанных часов;

– месячная: размер оклада, ставка, премия (опционально).

Выходными данными является таблица сотрудников с рассчитанной заработной платой.

## 1.8 Прототип пользовательского интерфейса

При инициализации проекта перед пользователем появляется окно, представленное на рисунке 1.1.

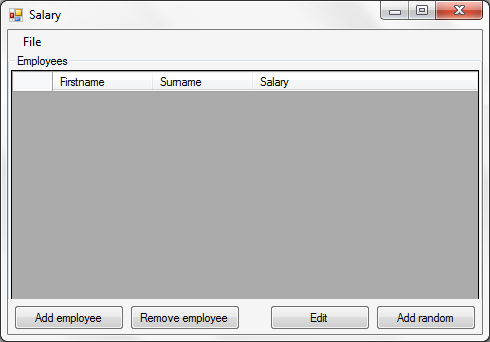


Рисунок 1.1 – Макет интерфейса пользователя

Кнопка «Add employee» открывает новое диалоговое окно, позволяющее заполнить данные для добавления нового пользователя. Окно представлено на рисунке 1.2.

Кнопка «Remove employee» позволяет удалить выбранного в таблице сотрудника.

Кнопка «Edit» позволяет редактировать выбранного в таблице сотрудника посредством вызова окна, представленного на рисунке 1.2.

Кнопка «Add random» позволяет добавить случайного сотрудника.

Диалоговое окно для добавления нового сотрудника предполагает заполнение следующих полей:

– «Firstname» – имя сотрудника;

– «Surname» – фамилия сотрудника.

Принимают строку символов. Предполагается, что пользователь может ввести один дефис в каждом поле. Введение цифр запрещено.

Выбор способа начисления заработной платы («Wage type») выбирается между почасовой оплатой «Hourly» и оплатой по окладу «Monthly». Рассмотрим каждый случай подробнее.

При выборе способа оплаты как почасовая, становятся активными соответствующие поля:

– «Price per hour» – соответствует установленной стоимости часа работы, принимает десятичное значение, которое должно быть больше либо равно 30 и меньше либо равно 1000;

– «Hours worked» – соответствует количеству отработанных сотрудником часов, принимает целочисленное значение, которое должно быть больше либо равно 80 и меньше либо равно 300.

В случае выбора способа оплаты по окладу, становятся доступными следующие поля:

– «Reward» – соответствует размеру оклада сотрудника в месяц, принимает десятичные значения, которые должны быть строго больше 7500 и меньше либо равно 150000;

– «Rate» – соответствует размеру ставки сотрудника, принимает десятичные значения, строго больше нуля и меньше либо равно 2,5;

– «Bounty» – соответствует размеру премии, принимает десятичные значения, больше либо равно 0 и меньше либо равно 50000. Данное значение выбирается опционально, по умолчанию установлено как 0.

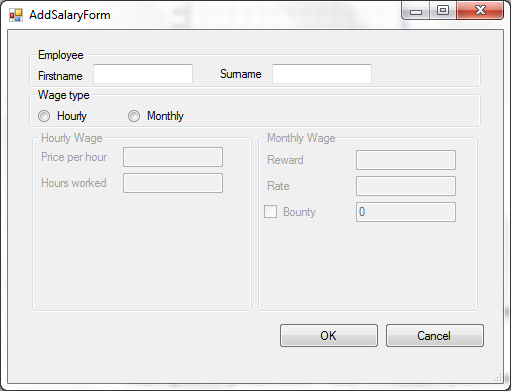


Рисунок 1.2 – Окно добавления нового сотрудника

В случае, если установленные значения удовлетворяют установленным ораничениям, при нажатии кнопки «OK», в таблицу , представленной в окне на рисунке 1.1, будет добавлен сотрудник с указанным именем и фамилией и рассчитанной заработной платой. В ином случае, пользователь получит сообщение об ошибке, где указано, какие поля заполнены неверно. Пример представлен на рисунке 1.3. При нажатии кнопки «Cancel» форма закроется

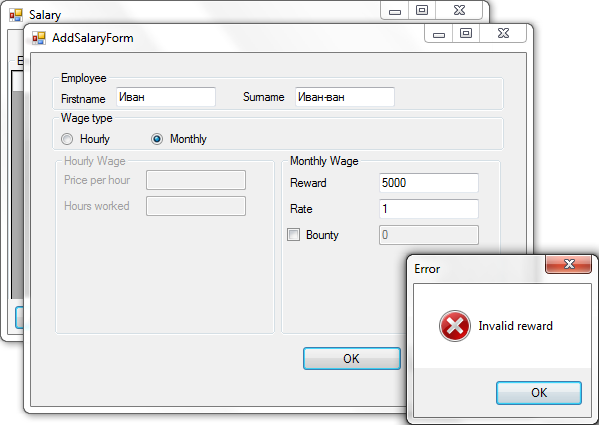


Рисунок 1.3 – Сообщение об ошибке

У пользователя имеется возможность сохранять результаты произведенных действий, либо загружать уже существующие. Выполнить это можно с помощью выпадающего меню «File», как показано на рисунке 1.4.

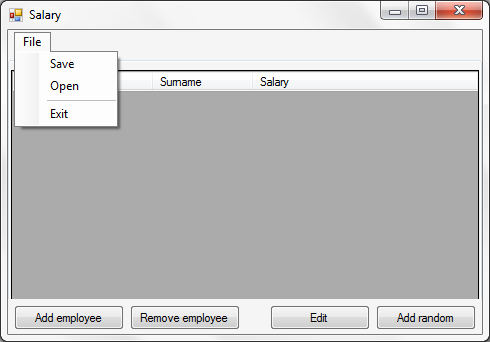


Рисунок 1.4 – Выпадающее меню

После нажатия кнопки «File» пользователь может:

– сохранить произведенные действия, выбрав пункт «Save»;

– загрузить ранее сохраненные данные, выбрав пункт «Open»;

– выйти из программы, выбрав пункт «Exit».

## 1.9 План разработки проекта и ресурсы

Разработка проекта имеет следующие стадии:

– разработка технического задания;

– реализация;

– тестирование.

На стадии разработки технического задания производится разработка, согласование и утверждение настоящего технического задания.

На стадии реализации выполняется создание эскиза пользовательского интерфейса, производится разработка архитектуры программы, ее проектирование, написание кода, отладка.

На стадии тестирования программа подвергается модульному тестированию.

# 2 Введение

Документация к проекту занимает важное место в современном понимании разработки. Наличие у проекта документации повышает скорость разработки и повышает «прозрачность» проекта, используемых в нём алгоритмов и архитектурных решений.

Одной из задач, поставленных в техническом задании и подлежащей решению, является написание технической документации.

Для описания архитектуры пользовательского сценария системы был выбран унифицированный язык моделирования (UML) [2]. На основе UML построены диаграммы классов и диаграммы классов использования.

UML (Unified Modeling Language – унифицированный язык моделирования) – язык графического описания для объектного моделирования в области разработки программного обеспечения, моделирования бизнес-процессов, системного проектирования и отображения организационных структур [3].

UML является языком широкого профиля, это – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем.

UML позволяет разработчикам программного обеспечения достигнуть соглашения в графических обозначениях для представления общих понятий и больше сконцентрироваться на проектировании и архитектуре.

# 3 Описание программной системы

Одним из важных моментов при проектировании программы является внешний вид программы. Внешний вид программы должен максимально отражать функциональные особенности программы, чтобы пользователь мог за минимальное время освоить программу и воспользоваться ею.

Для построения UML диаграмм была использована программа Enterprise Architect.

## 3.1 UML диаграмма сценариев использования

Диаграмма сценариев использования (Use case diagram, диаграмма прецедентов) – диаграмма, на которой отражены отношения, существующие между актёрами и вариантами использования [4].

Основное назначение диаграммы – описание функциональности и поведения, позволяющее заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему.

Работа над диаграммой может начаться с текстового описания, полученного при работе с заказчиком. При этом нефункциональные требования (например, конкретный язык или система программирования) при составлении модели прецедентов опускаются

UML диаграмма сценариев использования была разбита на две части: первая в полной мере отражает добавление сотрудника с почасовой оплатой труда, что представлено на рисунке 3.1, вторая – добавление сотрудника с фиксированным окладом, на рисунке 3.2.

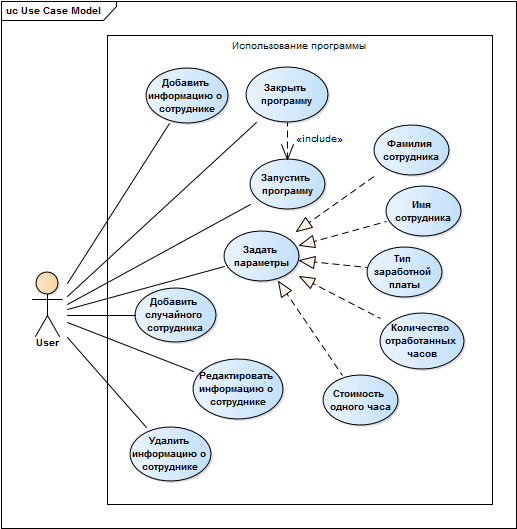


Рисунок 3.1 – Добавление сотрудника с почасовой оплатой

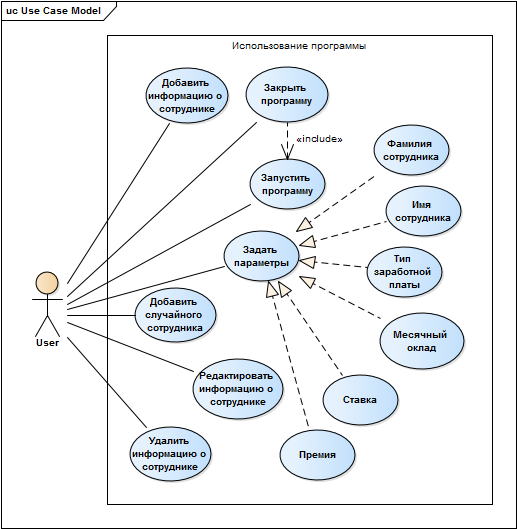


Рисунок 3.2 – Добавление сотрудника с фиксированным окладом

## 3.2 UML диаграмма классов

Диаграмма классов – диаграмма, демонстрирующая классы системы, их атрибуты, методы и взаимосвязи вежду ними [4]. Диаграмма классов является ключевым элементом в объектно-ориентированном моделировании. На диаграмме классы представлены в рамках, содержащих три компонента:

В верхней части написано имя класса. Имя класса выравнивается по центру и пишется полужирным шрифтом. Имена классов начинаются с заглавной буквы. Если класс абстрактный – то его имя пишется полужирным курсивом.

Посередине располагаются поля (атрибуты) класса. Они выровнены по левому краю и начинаются с маленькой буквы.

Нижняя часть содержит методы класса. Они также выровнены по левому краю и пишутся с маленькой буквы.

Статическое описание программной системы с помощью классов представлено на рисунке 3.3.

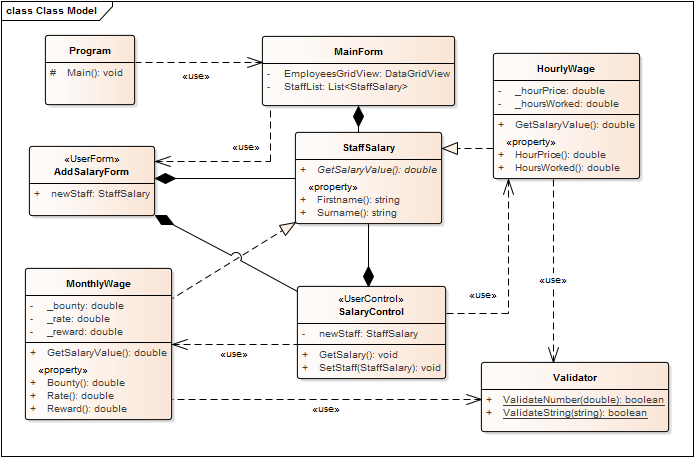


Рисунок 3.3 – Диаграмма классов

## 3.3 Тестирование программы

В модульном тестировании применяется подход к написанию тестов для каждой нетривиальной функции или метода, что позволяет достаточно быстро проверять, не привело ли очередное изменение кода к появлению ошибок в уже оттестированных местах программы, а также облегчает обнаружение и устранение таких ошибок.

Для проверки открытых членов, полей и методов классов было произведено модульное тестирование. Результаты выполнения тестовых классов HourlyWageTests и MonthlyWageTests представлены на рисунках 3.4 и 3.5 соответственно.

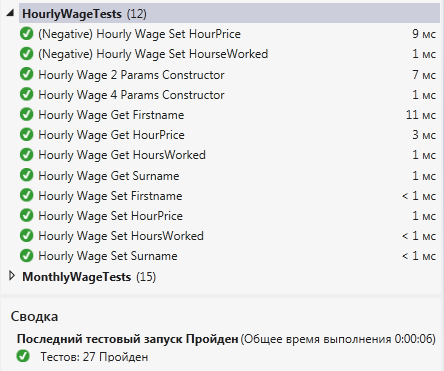


Рисунок 3.4 – Результаты тестирования

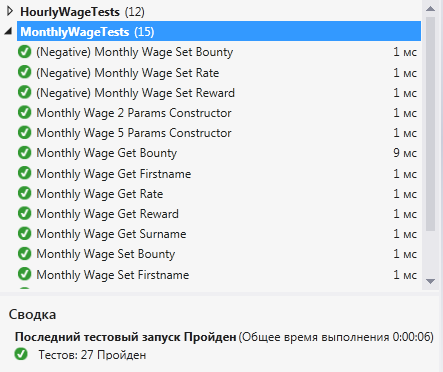


Рисунок 3.5 – Результаты тестирования

## 3.4 Версионный контроль

Работа над данным проектом проводилась с помощью использования системы управления версиями – Git.

В качестве хостинга для репозитория выступил GitHub – крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки [5]. Данный сервис был выбран по причине возможности его бесплатного использования для проектов с открытым исходным кодом, которым и является разрабатываемый проект.

# 4 Заключение

В ходе выполнения данного индивидуального задания были изучены основные этапы разработки программного обеспечения на примере проекта «Начисление заработной платы». Пройдены значимые этапы разработки, начиная от постановки технического задания и заканчивая написанием технической документации.

В разработанном приложении предусмотрены возможности добавления, удаления, редактирования данных о сотрудниках, а также сохранение проделанной работы и возможность ее загрузки. При заполнении данных о сотруднике, предусмотрена проверка полей на вводимые параметры, а также вывод сообщений об ошибке, если данные были заполнены неверно.

Разработанные в проекте модульные тесты позволяют своевременно отслеживать ошибки, которые могли появиться при добавлении нового кода либо изменении уже существующего.

В разработке применялась каскадная модель управления проектами, также известная как «Водопад» («Waterfall»). Данная модель подразумевает последовательное прохождение стадий разработки, каждая из которых начинается после завершения предыдущей. В ней четко определены стоимость работы и сроки разработки. Недостаток данной модели управления проектами заключается в следующем: она дает хорошие результаты в проектах с четко и заранее определенными требованиями и способами их реализации.

В процессе разработки использовалась распределенная система управления версиями – Git.

# Список использованных источников

1 1С: Зарплата и кадры бюджетного учреждения 8 [Электронный ресурс]. – Режим доступа: <http://v8.1c.ru/budghrm/> (дата обращения: 25.12.2017)

2 UML. Основы / Фаулер, М. – 3-е изд., пер. с англ. – СПб: Символ-Плюс, 2004. – 192 с.

3 UML [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/UML> (дата обращения: 25.12.2017)

4 Диаграмма вариантов использования [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Диаграмма_прецедентов> (дата обращения: 25.12.2017)

5 GitHub [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/GitHub> (дата обращения: 25.12.2017)