Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

Допускаю к защите

|  |  |
| --- | --- |
| Руководитель |  |
|  | подпись |
|  | З.А. Бахвалова |
|  | И.О. Фамилия |

|  |
| --- |
| Проектирование и разработка прикладного ПО |
| «Онлайн-регистрация авиабилетов» |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

|  |
| --- |
| Информатика |

|  |  |  |
| --- | --- | --- |
| 1.014.00.00 – ПЗ | | |
| обозначение документа | | |
| Выполнил студент |  | ИСТб-20-3 | |  |  | |  | Е.П. Корытов |
|  |  | шифр | |  | подпись | |  | И.О. Фамилия |
| Нормоконтроль |  |  | |  |  | |  | З.А. Бахвалова |
|  |  |  | |  | подпись | |  | И.О. Фамилия |
| Курсовая работа защищена с оценкой | | | | |  | | | |

Иркутск 2023 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **По курсу** | Технологии разработки программных комплексов | | | | | | | | | | | | | | | | |
| **Студенту** | Корытов Е.П. | | | | | | | | | | | | | | | | |
|  | (фамилия, инициалы] | | | | | | | | | | | | | | | | |
| **Тема работы** | | Проектирование и разработка прикладного ПО. | | | | | | | | | | | | | | | |
| **Исходные данные** | | Онлайн-регистрации авиабилетов. Проанализировать предметную область и реализовать систему в соответствии с жизненный циклом ПО. | | | | | | | | | | | | | | | |
| **Рекомендуемая литература** | | | | | | |  | | | | | | | | | | |
| 1. СТО 005-2015 СИСТЕМА МЕНЕДЖМЕНТА КАЧЕСТВА. Учебно-методическая деятельность. Оформление курсовых проектов (работ) и выпускных квалификационных работ технических специальностей (http://www.istu.edu/structure/57/2506/) | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | |
| Дата выдачи задания | | | « | 16 | » | февраля | | | | | 2023 г. | | | | |
|  | | | | | | | | | | | | | | | | | |
| Задание получил | | | | | | | |  | | | | |  | | Е.П.Корытв | | |
|  | | | | | | | | подпись | | | | |  | | И.О. Фамилия | | |
|  | | | | | | | | |  |  | |  | |  | | |  | |
| Дата представления работы руководителю | | | | | | | | | « | 17 | | » | | мая | | | 2023г. |
|  | | | | | | | | | | | | | | | | | | |
| Руководитель курсовой работы | | | | | | | |  | | | | |  | | З.А. Бахвалова | | |
|  | | | | | | | | подпись | | | | |  | | И.О. Фамилия | | |

Оглавление

[**Введение** 4](#_Toc135148271)

[**1.** **Выработка системных требований** 5](#_Toc135148272)

[**Описание предметной области** 5](#_Toc135148273)

[**1.2 Цель проекта** 5](#_Toc135148274)

[**1.3 Описание проблемы** 5](#_Toc135148275)

[**1.4** **Словарь предметной области** 6](#_Toc135148276)

[**1.5 Функциональность ПО** 7](#_Toc135148277)

[**2. Формальное описание требований к программному продукту** 8](#_Toc135148278)

[**2.1 Определение бизнес требований** 8](#_Toc135148279)

[**2.2 Определение требований пользователя** 10](#_Toc135148280)

[**2.3 Описание поведения разрабатываемой системы** 15](#_Toc135148281)

[**3. Проектирование программного продукта** 16](#_Toc135148282)

[**Архитектура ПО** 16](#_Toc135148283)

[**3.1.1 Проектирование классов разрабатываемой системы** 16](#_Toc135148284)

[**3.1.2 Диаграмма интерфейсных классов** 20](#_Toc135148285)

[**3.1.3 Диаграмма классов данных** 20](#_Toc135148286)

[**3.1.4 Проектирование хранилища данных** 21](#_Toc135148287)

[**3.1.5 Проектирование макетов** 25](#_Toc135148288)

[**3.2 Программная реализация** 27](#_Toc135148289)

[**Заключение** 30](#_Toc135148290)

[**Список использованных источников** 31](#_Toc135148291)

**Введение**

Курсовая работа направлена на проектирование и создание многопользовательского приложения. В процессе выполнения курсовой работы студент отрабатывает технологический процесс разработки программного продукта, включающий в себя разработку технического задания, внешнее и внутреннее проектирование, кодирование, тестирование, разработку документации и развитие способности принимать обоснованные решения в ходе проектной деятельности и получать базовый опыт разработки программного обеспечения.

Для выполнения курсового проекта была выбрана тема «Учёт выполненных курсовых работ». Цель проекта заключается в разработке программного обеспечения, которое поможет вести учет оценок за курсовые работы студентов, выявлять неуспевающих и составлять отчеты по выбранным параметрам.

Программный продукт - это программа, которая может использоваться независимо от ее разработчиков для заданных целей на разных компьютерах, при условии, что они соответствуют системным требованиям.

1. **Выработка системных требований**

**Описание предметной области**

При сдаче студентом курсовой работы руководителю и выставлении оценки, руководитель добавляет в «Список выполненных курсовых работ», в котором описывается вся информация, необходимая для анализа и хранения (табельный номер студента, ФИО студента, группа, предмет, предметная область, тема, оценка и год). Этот документ является архивом информации о всех сданных курсовых работах студентами. В конце учебного года руководитель готовит отчет по сданным курсовым работам в группе. Так же есть возможность выводить список работ по определённому предмету и году. И работ по руководителям.  
 Благодаря, новой системе преподавателям не нужно будет идти в деканат и нести список, ведь они могу после сдачи курсовой работы студентом сразу же, через новое ПО проставить оценку студенту, в свою очередь благодаря БД, преподавателю даже не придётся заполнять данные о ученике.

Автоматизация задачи позволит повысить производительность работы руководителей курсовых работ, уменьшить количество ручной работы при оформлении отчетов, повысить точность и достоверность данных, уменьшить количество ошибок при подготовке документов, что будет способствовать повышению надежности и достоверности предоставляемой информации. Внедрение программы позволит сократить денежные, трудовые и временные затраты в организации процесса учета курсовых и поиска необходимой информации. Хранение информации о курсовых в базе данных и поиск по различным критериям существенно упростится. Удобный интерфейс позволит руководителю не только оперативно ввести информацию, но и узнать всю необходимую для него информацию.

**1.2 Цель проекта**

Обеспечить точность и своевременность учета выполненных работ, чтобы руководитель мог контролировать прогресс сдачи курсовых работ, управлять бюджетом и принимать информированные решения на основе актуальной информации.

**1.3 Описание проблемы**

Основной целью автоматизации данной предметной области является необходимость более удобной и быстрой обработки данных по сделанным курсовым работам. Для этого необходимо создать базу данных для хранения информации о студентах, предметах и курсовых работах на электронном носителе.

В университете учет курсовых работ, сделанных студентами, ведется посредством электронных таблиц Microsoft Excel и текстового редактора Microsoft Word. Отсутствие единой базы данных привело к усложнению учета и получения необходимой информации для преподавателей и директора. В связи с этой проблемой возникла актуальная необходимость осуществления автоматизации учета сделанных и защищенных курсовых работ студентами университета.

* 1. **Словарь предметной области**

Таблица 1 – Словарь терминов предметной области

|  |  |  |
| --- | --- | --- |
| № | Термин | Назначение |
| 1. | Студент | Человек, который обучается в университете |
| 2. | Сотрудник деканата | Сотрудник, который работает в деканате и ведёт работу с ведомостями |
| 3. | Ведомость | Документ, в котором выставляются оценки за зачет или экзамен всем студентам группы |
| 4. | Преподаватель | Работник университета, преподающий определенную дисциплину |
| 5. | Дисциплина | Учебный предмет в университете |
| 6. | Параметр оценки | Отметка, показывающая качество освоения дисциплины учащимся, значения: «2», «3», «4», «5», «нз» - не зачёт |

Таблица 2 – Словарь предметной области по методу Аббота

|  |  |  |
| --- | --- | --- |
| № | Существительное | Глагол |
| 1. | Студент | - |
| 2. | Сотрудник деканата | - |
| 3. | Ведомость | - |
| 4. | Преподаватель | - |
| 5. | Дисциплина | - |
| 6. | Параметр оценки | Распознать и вывести |

Таблица 3 – Объектно-ориентированный словарь

|  |  |  |  |
| --- | --- | --- | --- |
| № | Класс (Сущность/Актер) | Свойство (Состояние) | Метод (Функция/Поведение) |
| 1. | Студент | ФИО, группа | - |
| 2. | Сотрудник деканата | ФИО | - |
| 3. | Ведомость | Преподаватель, дисциплина, параметр оценки, список ФИО студентов, подпись преподавателя | - |
| 4. | Преподаватель | Фио | - |
| 5. | Дисциплина | Наименование, ЗЕТ, количество часов, критерий типа контроля | - |
| 6. | Параметр оценки | «2», «3», «4», «5», «нз» - не зачёт, | Запись, редактирование |

**1.5 Функциональность ПО**

**Формирование отобржения информации о выполненных курсовых работах**

Модуль должен, вывести на экран информацию о предметах, студентах, темой и оценкой за курсовую работу. Так же должен предоставлять пользователю возможность, двигать ползунок вверх и вниз, тем самым давать всю возможную информацию.

**Формирование отчётных документов**

Модуль должен давать возможность, формировать отчётные документы, для руководства, деканата и т.д. В данных документах будет выполенная курсовая работа с оценкой, так же эти документы можно будет распечатать или переслать почтой или прочими мессенджерами.

**Изменения итоговой оценки для задолжников**

Модуль должен давать возможность пользователю изменять информацию о выбранном предмете.

**Фильтрация списка предметов**

Модуль должен давать возможность пользователю отфильтровать список предметов по заданным критериям: по студентам, руководству, специальности и группе.

При фильтрации по студенту, необходимо ввести ФИО студента.

При фильтрации по специальности, необходимо узнать о существовании данной специальности в учебном заведении.

При фильтрации по группе, необходимо указать данную группу.

При фильтрации по руководству, необходимо указать год сдачи или тему или оценку.

**2. Формальное описание требований к программному продукту**

**2.1 Определение бизнес требований**

В данном разделе описан бизнес-процесс контроль успеваемости студентов.

Описание контекста модели:

Предмет моделирования – процесс автоматизации учёта выставления оценок за курсовую работу у студентов.

Область моделирования – модель охватывает процессы, связанные с записью информации об оценках за курсовую работу студентов в базу данных.

Цель моделирования – детальное представление работы системы

Точка зрения – Стдуент

Диаграмма (А-0) является вершиной структуры и представляет собой общее описание системы и её взаимодействия с внешней средой (Рис. 1).

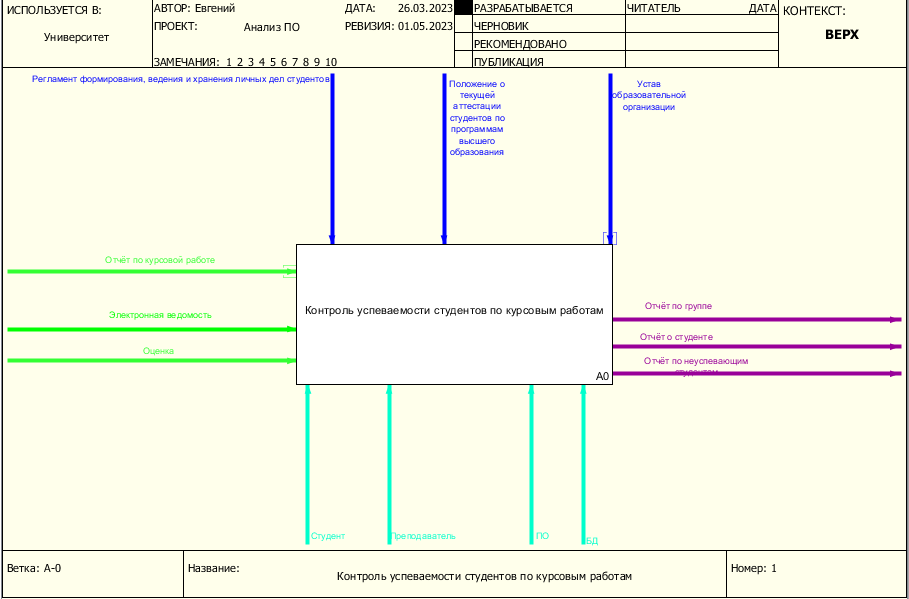


Рисунок 1 – Процесс «Контроль успеваемости студентов по курсовым работам»

Процесс «Контроль успеваемости по курсовым работам» (см. рисунок 2):

Основные бизнес-функции процесса:

1. Загрузка электронной ведомости в программу
2. Контроль успеваемости студентов
3. Запись данных в БД
4. Отчёты

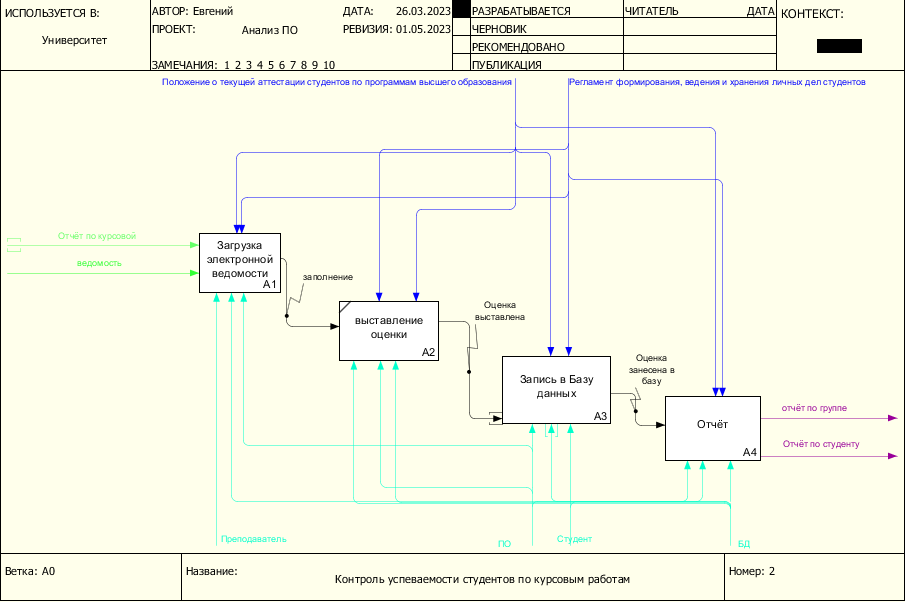


Рисунок 2 – Декомпозиция обобщенного процесса «Контроль успеваемости студентов»

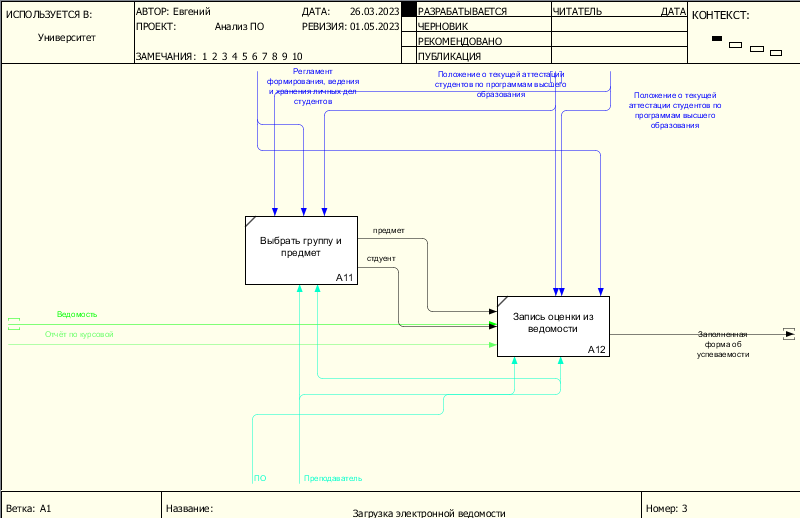


Рисунок 3 – Детализация бизнес-функции процесса «Загрузка электронной ведомости в программу»

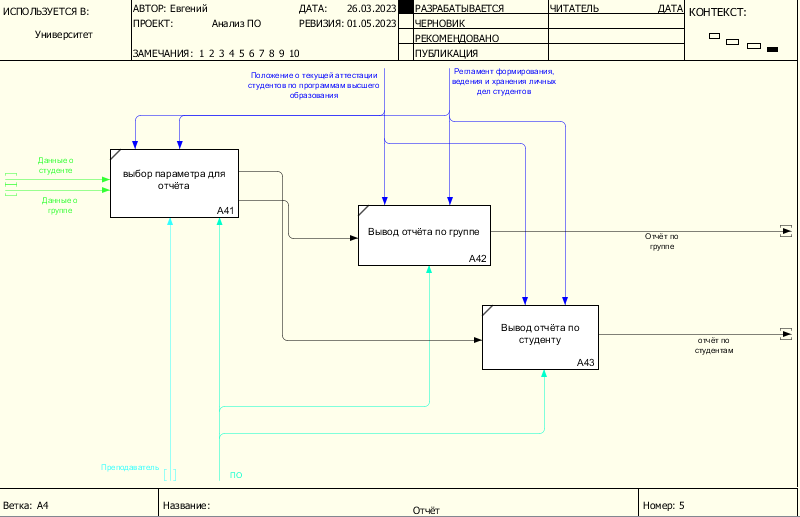
****

Рисунок 4 – Детализация бизнес-функции процесса «Отчёты»

**2.2 Определение требований пользователя**

Опишем функциональное назначение системы с помощью диаграммы вариантов использования, таким образом, показав, что будет делать разрабатываемся система.

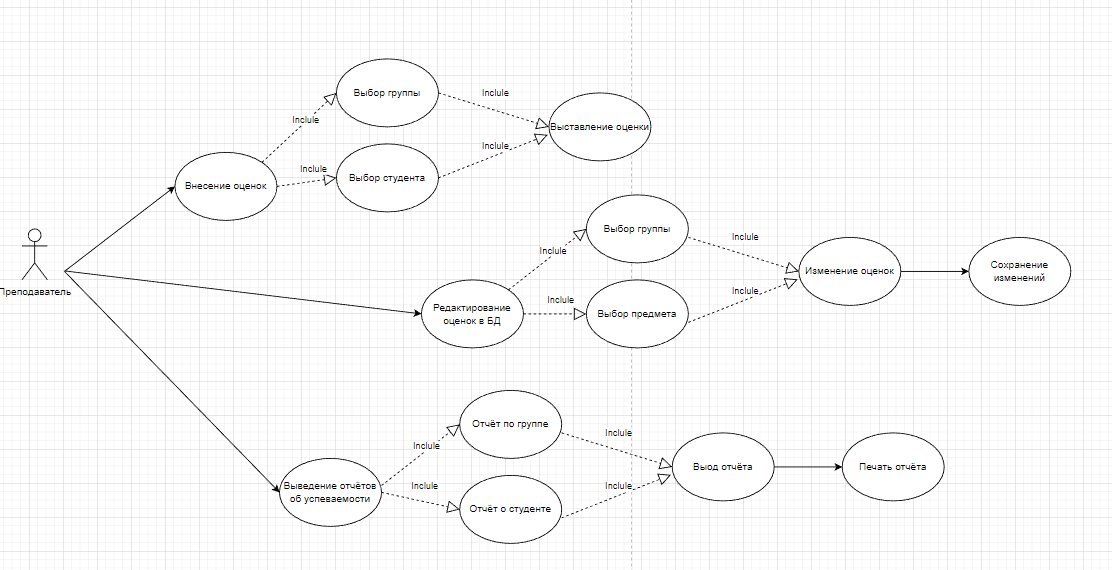


Рисунок 5 – Диаграмма функциональности разрабатываемого приложения

**Спецификации варианта использования метода «Загрузка списка групп»**

**Цель:** Загрузить список групп

**Активные субъекты:** Сотрудник деканата.

**Предусловие:** При запуске программы открывается главная форма с пунктами меню: Внесение оценок, редактирование оценок в БД и выведение отчётов об успеваемости. Затем при нажатии на кнопку внесение оценок формируется вторая форма «Внесение оценок» с пунктами: выбор группы, выбор предмета и кнопкой занесение оценок в БД. Список групп для выбора подгружается при переходе с главной формы на форму «Внесение оценок» через кнопку внесение оценок

**Основной сценарий:**

1. Нажатие кнопки внесения оценок
2. Запрос списка групп
3. Получение данных
4. Отображение списка групп в выпадающем списке
5. Передача управления

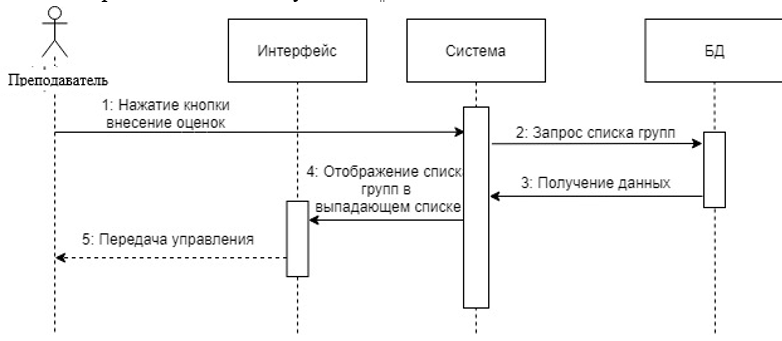


Рисунок 6 – Описание функции «Загрузка списка групп»

**Спецификации варианта использования метода «Выбор группы»**

**Цель:** Выбрать группу ивывести соответствующий список студентов и список предметов для выбранной группы.

**Активные субъекты:** Сотрудник деканата.

**Предусловие:** Перед выбором группы уже загружен список групп.

**Основной сценарий:**

1. Выбор нужной группы из списка
2. Запрос по студентам выбранной группы
3. Запрос по изучаемым предметам выбранной группы
4. Получение списка студентов группы
5. Получение списка изучаемых предметов группы
6. Отображение списка студентов группы
7. Отображение изучаемых предметов в выпадающем списке
8. Передача управления



Рисунок 7 – Описание функции «Выбор группы»

**Спецификации варианта использования метода «Выбор предмета»**

**Цель:** Выбрать предмет.

**Активные субъекты:** Сотрудник деканата.

**Предусловие:** Перед этим загружена группа, список студентов и список предметов. Только после выбора группы становится доступен список предметов.

**Основной сценарий:**

1. Выбор нужного предмета
2. Обработка
3. Отображение предмета
4. Передача управления

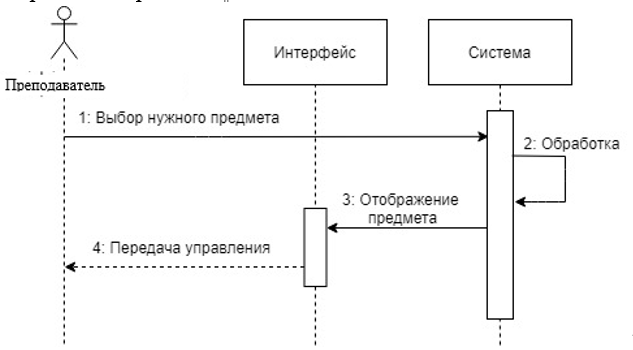


Рисунок 8 – Описание функции «Выбор предмета»

**Спецификации варианта использования метода «Загрузка файла ведомости»**

**Цель:** Открыть файл ведомости

**Активные субъекты:** Сотрудник деканата.

**Предусловие:** Перед этим выбрана группа и предмет.

**Основной сценарий:**

1. Загрузка файла
2. Загрузка
3. Отображение оценок студентов
4. Передача управления



Рисунок 9 – Описание функции «Загрузка файла ведомости»

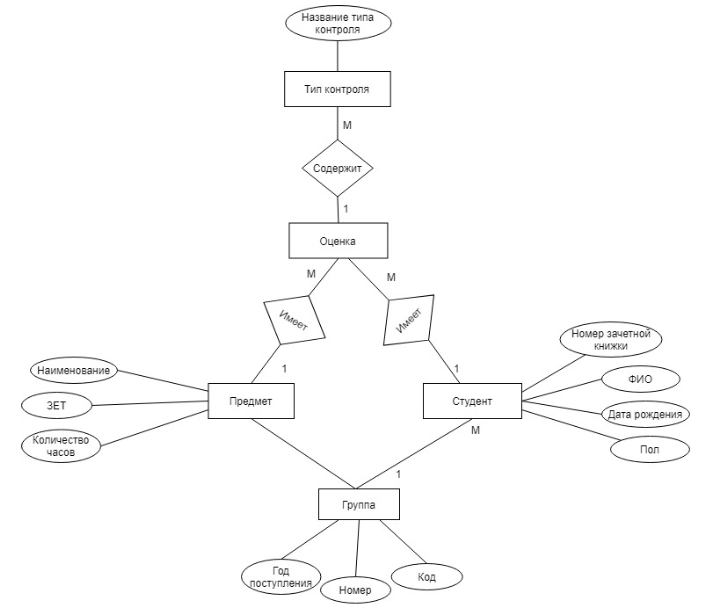


Рисунок 10 – Диаграмма сущностей-связей по Чену

**2.3 Описание поведения разрабатываемой системы**

Составим диаграмму активностей, которая будет описывать процесс редактирования оценок.

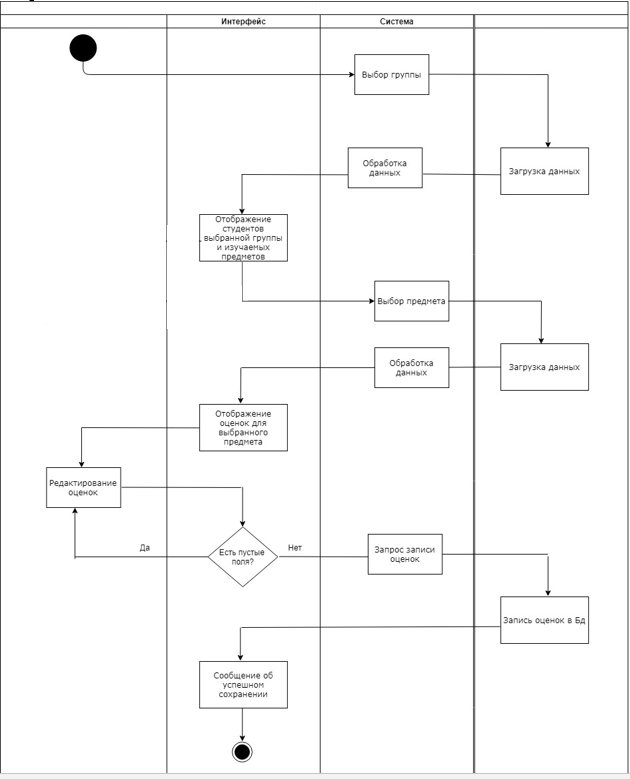


Рисунок 11 – Диаграмма активностей процессов «Редактирование оценок в БД»

**3. Проектирование программного продукта**

**Архитектура ПО**

**3.1.1 Проектирование классов разрабатываемой системы**

Для разработки программы будет использоваться язык программирования C# в среде Visual Studio. Visual Studio - это интегрированная среда разработки (IDE), которая предоставляет возможности для написания, отладки, сборки и публикации приложений. Она включает в себя не только стандартные редакторы и отладчики, но также компиляторы, средства автозавершения кода, графические конструкторы и другие функции, упрощающие процесс разработки. Использование C# обеспечивает удобное взаимодействие с базой данных, особенно с MS SQL Server, так как C# уже имеет встроенные функции для работы с ней.

База данных будет создана с использованием MS SQL Server, который является реляционной системой управления базами данных (СУБД) от Microsoft. Он обладает широким спектром функций, включая связанные запросы, связь с внешними таблицами и базами данных. SQL Server также поддерживает язык XML, протокол HTTP и предоставляет средства для повышения производительности и доступности данных, позволяющие распределить нагрузку и обеспечить непрерывную работу. Сложный доступ к данным используется для обеспечения надежного хранения, а SQL Server позволяет выполнять резервное копирование базы данных в любое время без отключения пользователей. Кроме того, при работе с большими объемами данных SQL Server является предпочтительным выбором для обеспечения функциональности.

Для связи программы с базой данных будет использован ADO.NET, который представляет набор классов для доступа к данным в реляционном формате, включая реляционные базы данных, такие как Microsoft SQL Server. ADO.NET является основной моделью доступа к данным для приложений на C# и предоставляет средства для работы с данными.

Архитектура, описанная в тексте, является клиент-серверной архитектурой. В данном случае программа, написанная на языке C# и разработанная в среде Visual Studio, выступает в роли клиента, который взаимодействует с базой данных, созданной в MS SQL Server.

ADO.NET используется для связи между клиентской программой (C#) и сервером базы данных (MS SQL Server). ADO.NET предоставляет набор классов и компонентов, которые позволяют программе установить соединение с базой данных, выполнить запросы и получить результаты.

Таким образом, клиентская программа, написанная на C# с использованием Visual Studio, взаимодействует с сервером базы данных MS SQL Server с помощью ADO.NET, что позволяет выполнять операции чтения, записи и обработки данных в базе данных.

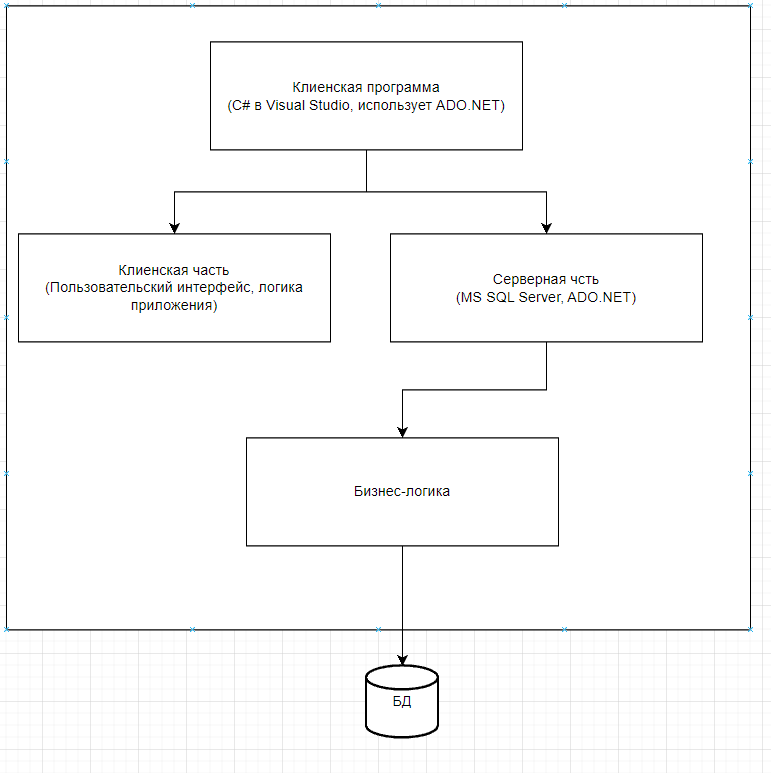


Рисунок 12 – Архитектура ИС

1. Клиентская программа:

* Включает пользовательский интерфейс (UI) и логику приложения.
* Написана на языке C# в среде Visual Studio.
* Использует ADO.NET для взаимодействия с серверной частью.

2. Клиентская часть:

* Содержит компоненты, связанные с пользовательским интерфейсом и логикой приложения.

3. Бизнес-логика:

* Отвечает за обработку и управление данными, выполняет операции бизнес-логики приложения.

4. Серверная часть:

* Содержит MS SQL Server и использует ADO.NET для связи с базой данных.

5. База данных:

* Хранит данные и предоставляет функциональность для их сохранения, извлечения и изменения.

Такая схема позволяет понять, что клиентская программа взаимодействует с серверной частью через ADO.NET, где на сервере расположена база данных. Бизнес-логика обеспечивает обработку данных и выполнение операций, необходимых для функционирования приложения.

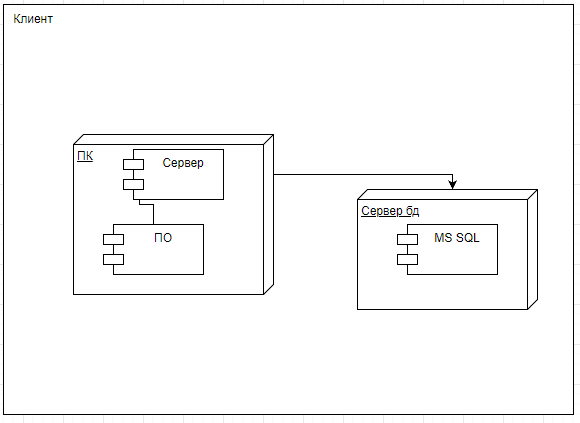
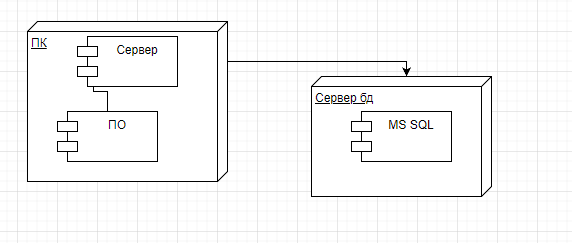


Рисунок 13 – диаграмма разверки

На диаграмме развертывания представлена архитектура ПО. Клиент (браузер) используя протокол http получает доступ к приложению, загружая файлы для отображения интерфейса. Клиент также может посылать запросы к API, с помощью которого происходит манипуляция с данными и работой с СУБД. FLASK API – сервер принимающий запросы от клиента. Его основная задача выдача генерация файла и его выдача и манипуляция с данными БД.



**3.1.2 Диаграмма интерфейсных классов**

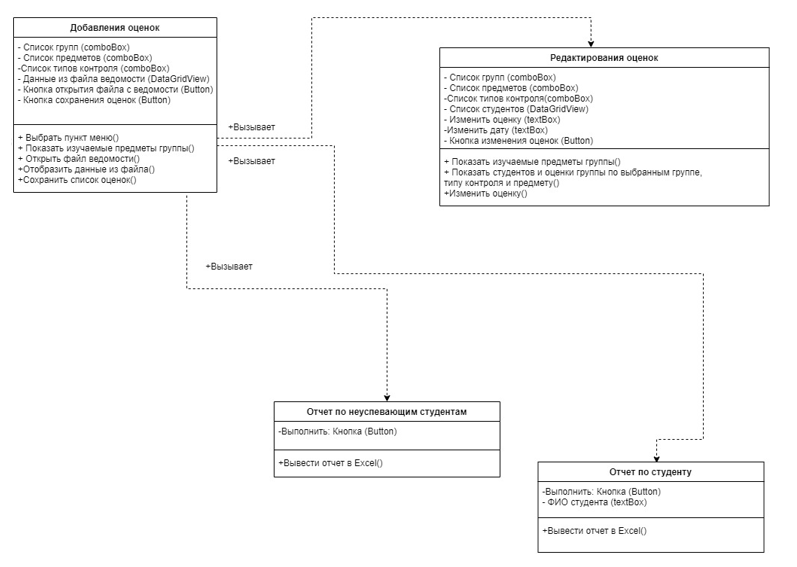


Рисунок 13 – Интерфейсные классы разрабатываемой системы

После входа в программу пользователю будет показано окно "Добавить оценки". В этом окне пользователь сможет выбрать группу из выпадающего списка и тип контроля. Затем в выпадающем списке будут отображены предметы для выбранной группы и типа контроля. После этого пользователю необходимо нажать на кнопку "Открыть файл" и выбрать файл с ведомостью с компьютера. После выбора файла, студенты, оценки, дата сдачи и тип контроля из файла будут добавлены. Затем пользователь должен сверить загруженные оценки с бумажной ведомостью и нажать кнопку "Сохранить оценки". Если все поля заполнены, оценки будут сохранены в базе данных. В случае, если какое-то поле с оценкой не заполнено, пользователь получит сообщение об ошибке.

При выборе пункта "Отчёты" пользователю будет доступна работа с отчётами, включающая формы отчёта по группам и типу контроля, по неуспевающим студентам и по конкретному студенту. Пользователь сможет выбрать нужный отчёт, нажав соответствующую кнопку, после чего откроется соответствующая форма в формате Excel.

**3.1.3 Диаграмма классов данных**

На рисунке 15 предоставлена диаграмма классов. Диаграмма классов необходима для описания функциональности и поведения программы.

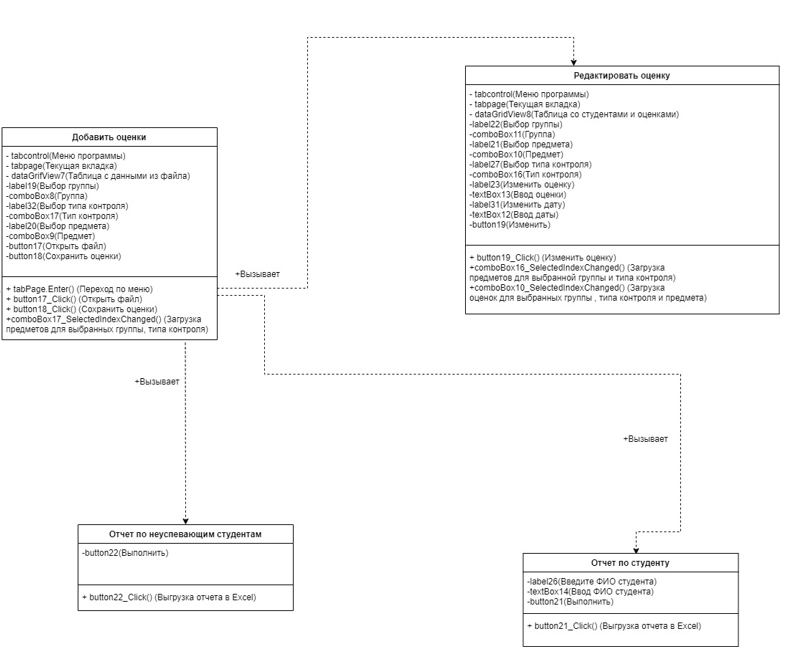


Рисунок 14 – Диаграмма классов данных

Доступны следующие функциональные классы: "Добавить оценки", "Редактировать оценку", "Отчет по группе и типу контроля", "Отчет по студенту" и "Отчет по неуспевающим студентам". Для реализации функциональности добавления оценок был создан класс "tabPage1", а для редактирования оценок - класс "tabPage2". Класс "tabPage3" отвечает за формирование отчета по группе и типу контроля. Для формирования отчета по конкретному студенту используется класс "tabPage4". И, наконец, класс "tabPage5" служит для формирования отчета по неуспевающим студентам.

**3.1.4 Проектирование хранилища данных**

На рисунке 16 представлена логическая модель представления данных

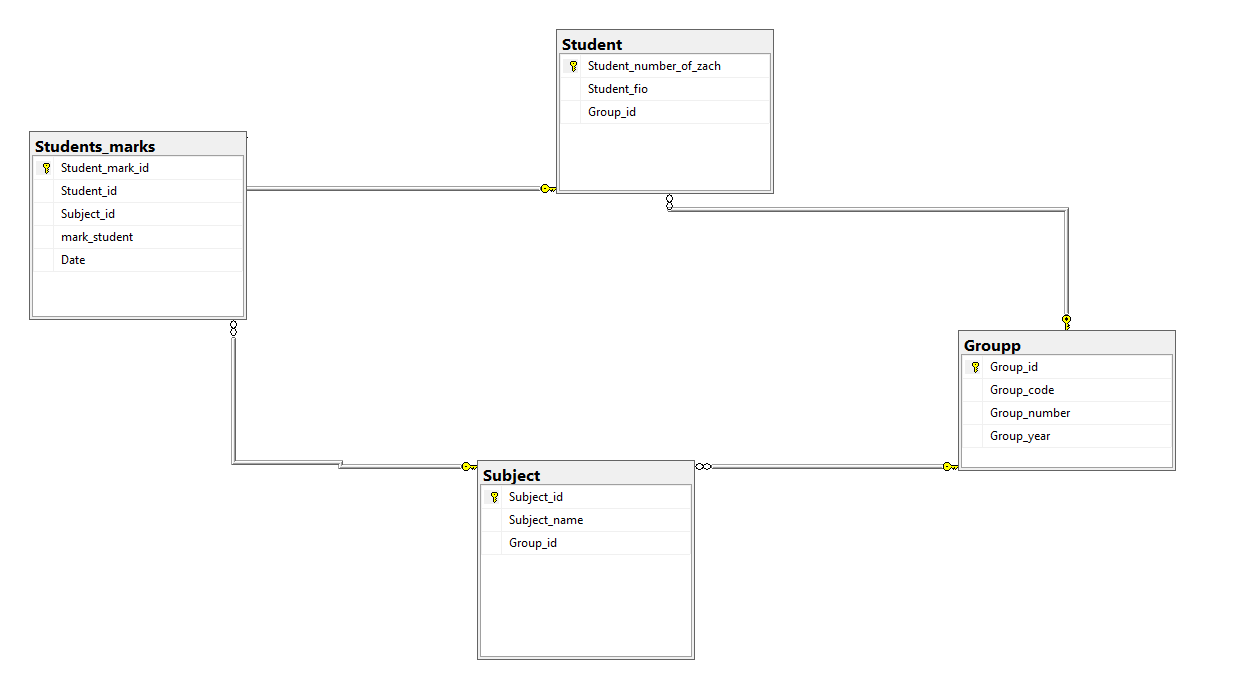


Рисунок 16 - Логическая модель представления данных

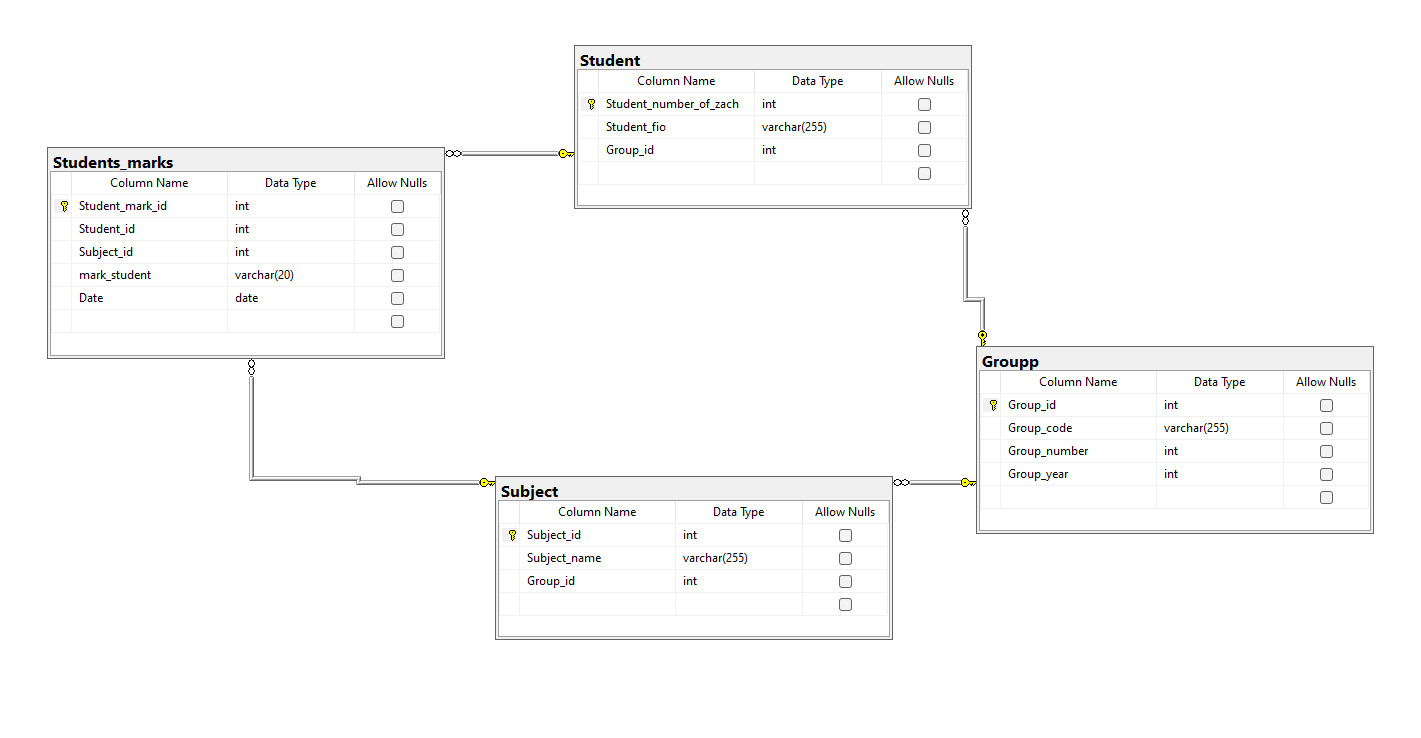


Рисунок 17 – Физическая схема БД

Соответствие между объектами логического и физического уровней

Таблица 3 – Соответствие между объектами логического и физического уровней

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Сущность | Описание | Атрибут | Поле | Тип |
| Оценки студента | Содержит данные об оценках студента | id Оценки | Student\_mark\_id | integer |
| id Студента | Student\_id | integer |
| id Предмета | Subject\_id | integer |
| Оценка студента | mark\_student | varchar |
| Дата сдачи | Date | date |
| Студент | Содержит данные о студенте | Номер зачетной книжки | Student\_number\_of\_zach | integer |
| Фио студента | Student\_fio | Varchar |
| Id Группы | Group\_id | integer |
| Предмет | Содержит данные о предмете | Id Предмета | Subject\_id | integer |
| Название предмета | Subject\_name | Varchar |
| Тип контроля | Type\_control | integer |
| Группа | Содержит данные о группе | Id Группы | Group\_id | integer |
| Код группы | Group\_code | varchar |
| Номер группы | Group\_number | integer |
| Год поступления | Group\_year | integer |

Физическое проектирование при помощи MS SQL Server:

USE [master]

GO

/\*\*\*\*\*\* Object: Database [kursach] Script Date: 15.05.2023 19:42:11 \*\*\*\*\*\*/

CREATE DATABASE [kursach]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'kursach', FILENAME = N'D:\SQL\MSSQL16.MSSQLSERVER\MSSQL\DATA\kursach.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'kursach\_log', FILENAME = N'D:\SQL\MSSQL16.MSSQLSERVER\MSSQL\DATA\kursach\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

WITH CATALOG\_COLLATION = DATABASE\_DEFAULT, LEDGER = OFF

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [kursach].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [kursach] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [kursach] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [kursach] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [kursach] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [kursach] SET ARITHABORT OFF

GO

ALTER DATABASE [kursach] SET AUTO\_CLOSE OFF

GO

ALTER DATABASE [kursach] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [kursach] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [kursach] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [kursach] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [kursach] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [kursach] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [kursach] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [kursach] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [kursach] SET DISABLE\_BROKER

GO

ALTER DATABASE [kursach] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [kursach] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [kursach] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [kursach] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [kursach] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [kursach] SET READ\_COMMITTED\_SNAPSHOT OFF

GO

ALTER DATABASE [kursach] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [kursach] SET RECOVERY FULL

GO

ALTER DATABASE [kursach] SET MULTI\_USER

GO

ALTER DATABASE [kursach] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [kursach] SET DB\_CHAINING OFF

GO

ALTER DATABASE [kursach] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [kursach] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [kursach] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [kursach] SET ACCELERATED\_DATABASE\_RECOVERY = OFF

GO

ALTER DATABASE [kursach] SET QUERY\_STORE = ON

GO

ALTER DATABASE [kursach] SET QUERY\_STORE (OPERATION\_MODE = READ\_WRITE, CLEANUP\_POLICY = (STALE\_QUERY\_THRESHOLD\_DAYS = 30), DATA\_FLUSH\_INTERVAL\_SECONDS = 900, INTERVAL\_LENGTH\_MINUTES = 60, MAX\_STORAGE\_SIZE\_MB = 1000, QUERY\_CAPTURE\_MODE = AUTO, SIZE\_BASED\_CLEANUP\_MODE = AUTO, MAX\_PLANS\_PER\_QUERY = 200, WAIT\_STATS\_CAPTURE\_MODE = ON)

GO

ALTER DATABASE [kursach] SET READ\_WRITE

GO

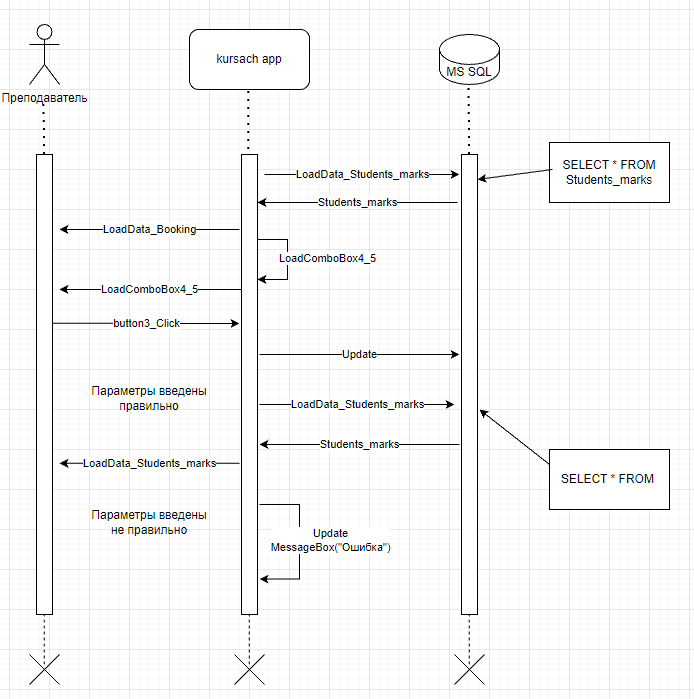
H

Рисунок 18 -Проектирование Use Case редактирование оценки

**3.1.5 Проектирование макетов**

Для создания графического интерфейса приложения будем использовать WindowsForms.

На рисунке 18 изображён макет окна добавления оценок (tabPage1). Именно это окно видит пользователь, когда запускает приложение.

Окно содержит следующие основные компоненты: меню (tabControl), выпадающий список групп (comboBox), выпадающий список предметов (comboBox), кнопки (button) для выбора файла ведомости с компьютера и сохранения оценок в базу данных, а также таблицу (dataGridView), которая отображает данные, загруженные из файла ведомости.

Пользователь использует меню для выбора функции программы. В таблице отображаются студенты выбранной группы, их оценки, дата сдачи и тип контроля, которые были загружены из файла ведомости.

Чтобы добавить оценку, необходимо выбрать группу и предмет из соответствующих выпадающих списков. Затем нужно загрузить файл ведомости с оценками, нажав кнопку "открыть файл". Данные из выбранного файла будут загружены в таблицу. После этого необходимо сопоставить оценки и дату сдачи с бумажной ведомостью и нажать кнопку "сохранить оценки".

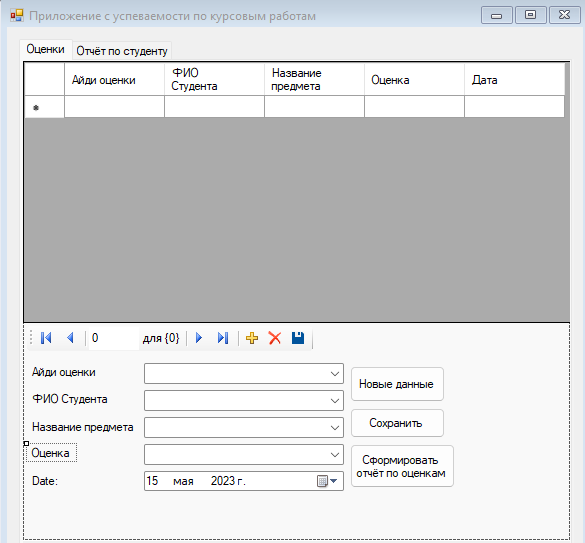


Рисунок 18 – Макет главного окна, функция «Добавить оценки»

На рисунке 19 изображен макет формы «Отчет по студенту» (tabPage4). Данная форма активируется при нажатии на соответствующую вкладку в меню программы.

Окно содержит следующие основные компоненты: меню (tabControl), текстовое поле для ввода ФИО студента (textBox) и кнопку "Выполнить" (button).

Для того чтобы вывести отчет, необходимо ввести ФИО студента в текстовое поле и нажать кнопку "Выполнить". После этого отчет будет сгенерирован и выведен в формате Excel файл.

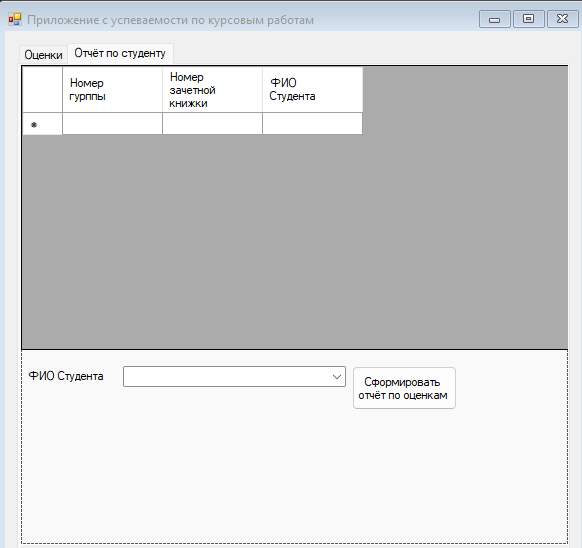


Рисунок 21 – Макет окна «Отчет по студенту»

**3.2 Программная реализация**

В результате проведенной исследовательской работы, был разработан совершенно уникальный и привлекательный графический интерфейс, который не только эффективно взаимодействует с пользователем, но и предоставляет широкий спектр возможностей, реализованных в разделе 1 нашей курсовой работы.

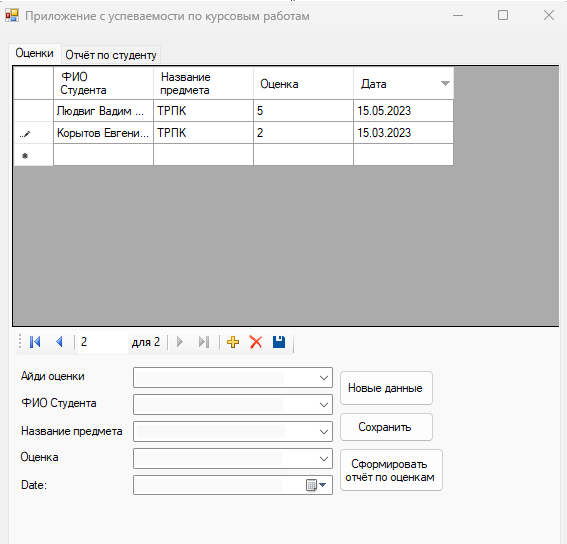


Рисунок 22 – Главное окно программы «Редактор оценок»

Для редактирования или добавления оценок, достаточно зайти в программу, с уже загруженной базой, благодаря (combobox), мы можем найти интересующего нас ученика и выставить ему оценку, при этом, выбрав ее в combobox, а так же указать дату выставления, если оценка уже стоит, для редактирования будет достаточно, нажать на студента и в combobox, она отразится, нам останется лишь выбрать другую нас интересующую.

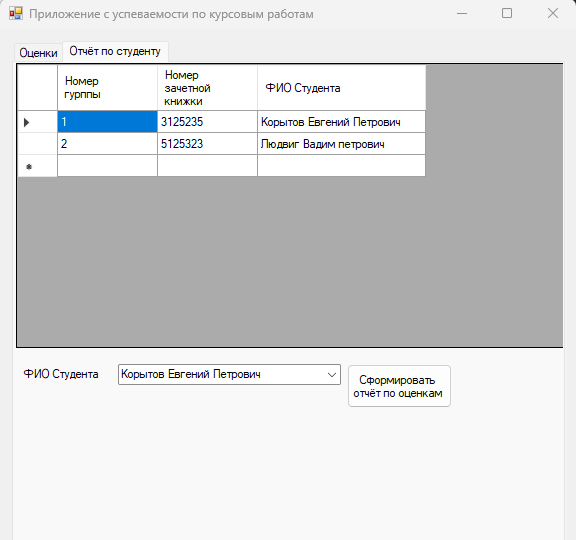


Рисунок 23 – Окно вывода отчёта по студенту

Для выведения отчета необходимо ввести ФИО студента и нажать кнопку выполнить. Отчет будет выведен в excel файл.

Весь исходный код выложен в репозитории онлайн-сервиса Github <https://github.com/LetfdGod/Kursovaia>

**Заключение**

В рамках нашей курсовой работы мы разработали инновационное приложение на языке C#, которое полностью автоматизирует процессы учета выполненных курсовых работ. Наше приложение представляет собой мощный инструмент, позволяющий эффективно управлять данными о выполненных работах и оценках студентов.

В процессе работы над проектом мы получили ценные навыки в формализации системных требований к программному продукту и в архитектурном и интерфейсном проектировании. Мы освоили передовые методологии, позволяющие построить модели бизнес-процессов и основных компонентов программного продукта, что обеспечило нам прочную основу для разработки приложения.

Особое внимание мы уделили работе с базами данных. Мы приобрели навыки создания и нормализации таблиц, а также связывания таблиц по ключу. Также мы освоили навыки формирования SQL-запросов и связывания базы данных с нашим приложением с использованием JDBC-драйвера. Эти навыки позволили нам создать эффективную систему учета, которая легко взаимодействует с базой данных и обеспечивает быстрый доступ к нужной информации.

На текущем этапе разработки мы успешно формализовали системные требования и спроектировали всю систему. Мы создали привлекательный графический интерфейс приложения, который позволяет легко добавлять и редактировать оценки, а также генерировать отчеты. Наше приложение предлагает интуитивно понятный пользовательский опыт, который значительно сокращает временные затраты на решение задачи учета успеваемости студентов и уменьшает трудовые затраты для сотрудников.

Дальнейшая доработка и использование нашей системы в реальных условиях обещает значительные выгоды. Она позволит организациям сократить время, затрачиваемое на учет успеваемости курсовых работ, и повысить эффективность работы персонала. Наше приложение будет надежным инструментом для автоматизации процессов и улучшения управления данными о курсовых работах и оценках студентов.

**Список использованных источников**

1. Пример использования диаграмм UML для описание предметной области (документ Microsoft word). Составители: Бахвалова З.А.
2. Анализ ПО\_25.11.16 (презентация). Составители: Бахвалова З.А.
3. Курсовая работа «Разработка прикладного программного обеспечения» Руководство и методические указания для студентов направления: 09.03.02 «Информационные системы и технологии». Составитель Бахвалова З.А. – Иркутск, Изд-во ИРНИТУ, 2018, 61 с.