### 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 ООП |
|  |  | В.Л. Аршинский |
| подпись |  | И.О. Фамилия |

|  |
| --- |
| **Разработка приложения подготовки данных для системы** |
| **автоматизированного расчета расписания в Геологоразведочном** |
| **техникуме ИРНИТУ** |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к выпускной квалификационной работе бакалавра

Программа бакалавриата

Информационные системы и технологии

на предприятиях / в машиностроении

по направлению подготовки

09.03.02 Информационные системы и технологии

|  |  |  |
| --- | --- | --- |
| 0.942.00.00 | | ПЗ |
|  | |  |
| Разработал студент  Группы ИСМб-19-1 |  |  | |  | Ю. А. Михейко |
|  |  | подпись | |  | И.О. Фамилия |
| Руководитель |  |  | |  | Е. А. Осипова |
|  |  | подпись | |  | И.О. Фамилия |
| Консультант |  |  | |  | Е. И. Молокова |
|  |  | подпись | |  | И.О. Фамилия |
| Консультант |  |  | |  | М. Д. Каташевцев |
|  |  | подпись | |  | И.О. Фамилия |
| Нормоконтроль |  |  | |  | Е. А. Осипова |
|  |  | подпись | |  | И.О. Фамилия |

Иркутск 2023 г.

Содержание

[Введение 5](#_Toc74820768)

[1 Анализ процесса формирования списков студентов на заселение 6](#_Toc74820769)

[1.1 Процесс предоставления мест в общежитиях студгородка ИрНИТУ. 6](#_Toc74820770)

[1.2 Рейтинговые ПО других ВУЗов 7](#_Toc74820771)

[1.3 Постановка задачи 8](#_Toc74820772)

[1.4 Функциональное назначение системы 10](#_Toc74820773)

[1.5 Формирование требований к функциям пользователей 18](#_Toc74820774)

[2 Проектирование системы 19](#_Toc74820775)

[2.1 Описание интерфейса 19](#_Toc74820776)

[2.2 Обоснование использования программных средств 27](#_Toc74820777)

[2.3 Архитектура системы 28](#_Toc74820778)

[2.4 Описание сценариев использования 30](#_Toc74820779)

[2.4.1 Спецификация варианта использования «Ручное формирование» 30](#_Toc74820780)

[2.4.2 Спецификация варианта использования «Выбор способа распределения мест» 31](#_Toc74820781)

[2.5 Проектирование модели данных 31](#_Toc74820782)

[3 Реализация 35](#_Toc74820783)

[3.1 Реализация классов пользовательского интерфейса 35](#_Toc74820784)

[3.2 Реализация функции «Ввод количества мест» 36](#_Toc74820785)

[3.3 Реализация функции «Вывод списка студентов» 37](#_Toc74820786)

[3.4 Реализация функции «Ручное формирование списка» 38](#_Toc74820787)

[3.5 Реализация функции «Фильтрация списка студентов» 39](#_Toc74820788)

[3.6 Экспорт/импорт Excel-файлов 39](#_Toc74820789)

[3.7 Результаты экспорта списка студентов в Excel 41](#_Toc74820790)

[3.8 Соединение с БД 41](#_Toc74820791)

[3.9 Реализация функции «Вход в систему» 44](#_Toc74820792)

[3.10 Реализация функции «Подача заявки» 45](#_Toc74820793)

[3.11 Реализация функции «Просмотр статуса» 46](#_Toc74820794)

[4 Экономическая часть 48](#_Toc74820795)

[5 Безопасность жизнедеятельности 51](#_Toc74820796)

[5.1 Краткая характеристика объекта 51](#_Toc74820797)

[5.2 Электробезопасность 58](#_Toc74820798)

[5.3 Пожарная безопасность 59](#_Toc74820799)

[5.4 Заключение по разделу 61](#_Toc74820800)

[Заключение 62](#_Toc74820801)

[Список используемых источников 63](#_Toc74820802)

# Введение

Данная работа ориентирована на проектирование и разработку информационной системы для рейтингового заселения студентов в общежития ИРНИТУ, с помощью которой будет автоматизирован процесс формирования списков студентов на заселение. Создание данной системы увеличит скорость заселения, а также сократит «бумажную» работу сотрудников.

Задача является актуальной, так как с каждым годом растет число иногородних студентов, поступающих в высшее учебное заведение. В связи с тяжелой экономической ситуацией в нашей стране в последние годы, не каждый студент может себе позволить снять квартиру, или жить в гостинице. В общежитиях ИРНИТУ, по данным управления студенческого городка, проживает почти четверть (около 25%) из общего числа студентов. Это обуславливается довольно низкой стоимостью проживания, удобным расположением по отношению к учебным корпусам университета, и не плохими жилищными условиями [1-2].

В первую очередь ИРНИТУ заселяет студентов 1 курса и студентов, обучающихся на льготной основе. Только после этого происходит заселение студентов старших курсов, из-за чего конкуренция на место значительно возрастает. Сотрудники Центра заселения, основываясь на определенный перечень документов, принимают решение о заселении/незаселении студента только в назначенный день. В связи с этим студенты не могут знать о решении заранее [3].

Потому разработка данной системы не только ускорит работу сотрудников, но также поможет студентам своевременно узнавать о принятом решении.

Цель: ускорить процесс рассмотрения заявлений от студентов на заселение.

Для достижения цели, необходимо решить следующие задачи:

* Изучить процесс формирования списков студентов на заселение
* Выполнить анализ предметной области
* Сформировать требования к системе
* Выполнить проектирование системы
* Реализовать прототип системы

# Анализ процесса формирования списков студентов на заселение в общежития

## Процесс предоставления мест в общежитиях студгородка ИрНИТУ.

В первую очередь политех предоставляет места для студентов, обучающихся на льготной основе и первокурсников. После, оставшиеся места распределяют между студентами 2-4 курсов.

Для предоставления места в общежитии, студенту необходимо в назначенный день прийти в центр заселения и написать заявление на заселение (студент, ранее проживающий в общежитии и планирующий заселиться на следующий год, пишет заявление на заселение по окончанию учебного года, то есть по факту выселения). После этого сотрудник центра заселения проверяет студента по нескольким критериям: отсутствие задолженностей, отсутствие нарушений, активное участие в жизни ВУЗа и/или общежития, а также наличие свободных мест. Условия для предоставления места в общежитии показаны в таблице 1.1.

* + - * 1. Условия для предоставления места в общежитии

|  |  |  |  |
| --- | --- | --- | --- |
| **Наличие свободных мест** | **Наличие задолженностей** | **Наличие нарушений** | **Заселение** |
| Да | Да | Да | Студенту заселяется в последнюю очередь |
| Да | Да | Нет | Студент заселяется в последнюю очередь |
| Да | Нет | Нет | Студент будет заселен |
| Нет | Да | Нет | Студент не будет заселен |
| Нет | Нет | Да | Студент не будет заселен |
| Нет | Нет | Нет | Студент может прийти с заявлением к начальнику управления студгородка и попросить выделить для него место. Если в других общежитиях места для заселения имеются, то студента заселяют, в противном случае, студент встает в очередь на заселение |

Также во всех случаях активное участие в жизни Вуза и/или общежития дает преимущество студенту и очередь на заселение строится следующим образом:

1. Студенты, не имеющие задолженностей и нарушений и принимающие активное участие в жизни ВУЗа и/или общежития;
2. Студенты, не имеющие задолженностей и нарушений;
3. Студенты с нарушениями, но не имеющие задолженностей, и принимающие активное участие в жизни ВУЗа и/или общежития;
4. Студенты с нарушениями, но не имеющие задолженностей;
5. Студенты с задолженностями, но без нарушений, и принимающие активное участие в жизни ВУЗа и/или общежития;
6. Студенты с задолженностями, но без нарушений;
7. Студенты с задолженностями и нарушениями и принимающие активное участие в жизни ВУЗа и/или общежития;
8. Студенты с задолженностями и нарушениями.

Как можно было заметить, распределение мест между желающими заселиться не простая задача, но также есть ряд факторов, усложняющих работу:

* Все критерии для заселения проверяются сотрудником вручную из разных источников.
* Студент не может узнать заранее предоставят ли ему место в общежитии.

## Рейтинговые ПО других ВУЗов

Для анализа конкурентов мной было рассмотрено три ВУЗа, два из которых используют методы рейтингового заселения. В открытом доступе достаточно мало информации о функциональных возможностях данных систем.

**ИТМО** - Санкт-Петербургский государственный университет информационных технологий, механики и оптики.

В данном университете присутствует рейтинговая система заселения. Распределение мест осуществляется в ИСУ «Общежитие» ВУЗа и только между студентами 1-го курса и закрепляется за ними на весь период обучения. В качестве конкурсных баллов принято считать суммарное количество баллов ЕГЭ по профильным дисциплинам. Привилегии при заселении имеют социально незащищенная категория граждан (студенты, поступившие на льготных основаниях), иностранные студенты, а также студенты 1-го курса, поступившие без вступительных испытаний как победители и призеры олимпиад.

Для предоставления места в общежитии при поступлении абитуриенту необходимо пройти авторизацию в личном кабинете ВУЗа и отметить пункт «Нуждаюсь в общежитии и временной регистрации на период обучения». После выхода приказа о зачислении поступившим студентам отправляется письмо на почту с инструкцией для обязательной регистрации в информационной системе управления университета (ИСУ). После регистрации в ИСУ абитуриенту/обучающемуся открывается доступ к приложению «Общежитие». Далее в очередном порядке по сформированному списку студентам предоставляются места. После получения электронного уведомления о назначении общежития студенту в течении 48 часов необходимо выбрать дату и время заселения.

Заселение обучающихся в общежития производится по мере появления свободных мест. Заселение происходит строго в порядке очереди, сроки назначения общежития не установлены и зависят от наличия жилых помещений, доступных для заселения.

До момента заселения студенту необходимо решать вопрос временного проживания в Санкт-Петербурге самостоятельно и продолжать следить за движением очереди в Личном кабинете ИСУ, пока не появится статус «предоставлено место» [4].

**МАИ** – Московский авиационный институт.

Данное учебное заведение также поддерживает рейтинговое заселение студентов в общежития. Места предоставляются на конкурсной основе в рамках квот, выделенных по факультетам и институтам на правах факультетов.

Распределение мест в студенческом городке МАИ между факультетами и другими структурными подразделениями и порядок заселения (в том числе утверждение списка обучающихся на вселение в студенческое общежитие) определяется по согласованию с профсоюзной организацией студентов и объявляется приказом Ректора МАИ [5].

**ИГУ** – Иркутский государственный университет.

Также рассмотрим такое учебное заведение, как ИГУ. В данном ВУЗе не предусмотрена система рейтингового заселения. Общежития предоставляются студентам на весь период обучения. При поступлении студенты делятся на 3 категории, в зависимости от которых формируется порядок очередности предоставления мест в общежитиях.

Главное отличие рассмотренных ВУЗов от ИРНИТУ состоит в том, что в данных университетах места в общежитиях предоставляются студентам на весь период обучения, в то время, как политех предоставляет место в общежитии на один год обучения. Из-за этого система рейтингового заселения ИРНИТУ становится более востребованной [6].

## Постановка задачи

Необходимо реализовать прототип системы «Рейтинговое заселение» для автоматизации процесса формирования списка студентов на заселение в общежитие.

Система должна быть реализована в виде WinForms приложения.

Требуется реализовать два метода формирования списка студентов: автоматизированное и ручное. Автоматизированное должно делиться на 3 способа распределения мест между студентами: по пороговым баллам, между институтами и на все места независимо от института. Список студентов на заселение должен формироваться с учетом их достижений в учебе, а также в иных видах вне учебной деятельности на уровне университета. Данные для формирования рейтинговых баллов будут поступать непосредственно от структурных подразделений, деканатов и заведующих общежитий и загружаться в качестве файлов с разрешением .xls/.xlsx в систему. Данные для подсчета баллов по успеваемости будут поступать из БД университета. Также из БД должны поступать данные о студентах, нуждающихся в общежитии.

То есть необходимо реализовать считывание данных с файлов форматов .xls/.xlsx, а также предоставить возможность интеграции системы с БД университета. Для этого будет создан прототип базы данных с указанием дополнительных сущностей для внедрения. Файлы для считывания должны храниться на носителе в определенной структуре: «…\Год\Организация\Название\Файл.xls».

Количество выделенных мест для каждого института в выбранном общежитии будут вводиться вручную, согласно приказу о распределении мест по общежитиям [2].

В рейтинговом заселении участвуют все студенты, нуждающиеся в общежитии, независимо от рода их деятельности. Всего предусмотрено 3 критерия оценки рейтинга:

• успеваемость и промежуточная аттестация,

• соблюдение правил внутреннего распорядка проживания в студенческих общежитиях ВУЗа и участие в общественной деятельности общежития,

• наличие ходатайства от руководителя структурного подразделения.

Для различных списков студентов должна быть предусмотрена возможность фильтрации по институтам и общежитиям. По запросу пользователя списки должны выводиться на экран в виде таблицы и сохраняться в Excel-файле.

Также необходимо реализовать системы для студентов, в виде Web-приложения.

Система будет предназначена для подачи заявления на заселение в общежитие онлайн. Авторизация студента должна проходить через личный кабинет студента на сайте ВУЗа. В качестве прототипа необходимо создать авторизацию пользователя по номеру зачетной книжки. После входа в систему пользователь должен иметь возможность подать заявление на заселение в общежитие, а также возможность просматривать статус поданного заявления.

## Функциональное назначение системы

В данном разделе описан бизнес-процесс формирования списков студентов на заселение с использованием нотации IDEF0. Диаграммы построены в программе Ramus.

Описание контекста модели:

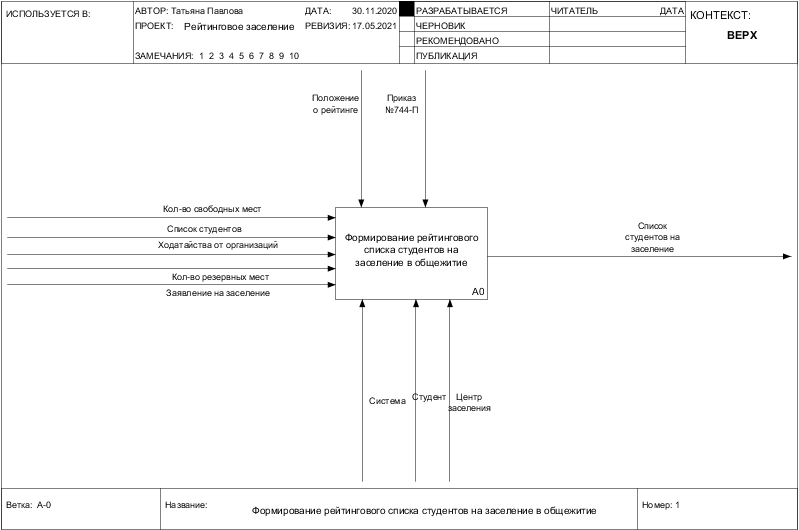
• предмет моделирования – деятельность сотрудников центра заселения.

• область моделирования – модель охватывает процессы, связанные с заселением студентов 2-4 курсов.

• точка зрения – сотрудники центра заселения.

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

На рисунке 1.1 представлена контекстная диаграмма, главный процесс – заселение студентов по рейтингу. Процесс осуществляет система и сотрудники центра заселения, используя БД. Для выполнения процесса необходимы ходатайства от организаций, список студентов, нуждающихся в общежитии и кол-во свободных мест. По итогу на выходе сотрудники центра заселения получают сформированный список студентов на заселение.



* + - 1. ­­­­­­Рисунок 1.1 – Процесс «Формирование рейтингового списка студентов»

Процесс «Формирование рейтингового списка студентов» декомпозируется на следующие этапы (см. рисунок 1.2):

1. Сбор и формирование общего рейтингового списка студентов.

На этом этапе происходит, считывание данных студентов, нуждающихся в общежитии с БД, загрузка всех необходимых документов, их обработка и формирование общего списка студентов с рейтинговыми баллами. После, этот список передается для дальнейшей обработки.

1. Сортировка списка по баллам.

Список сортируется по убыванию.

1. Выбор способа распределения мест.

Всего будет реализовано три способа распределения мест: распределение по институтам, распределение по пороговым баллам и распределение на все места независимо от института

1. Формирование списка на заселение.

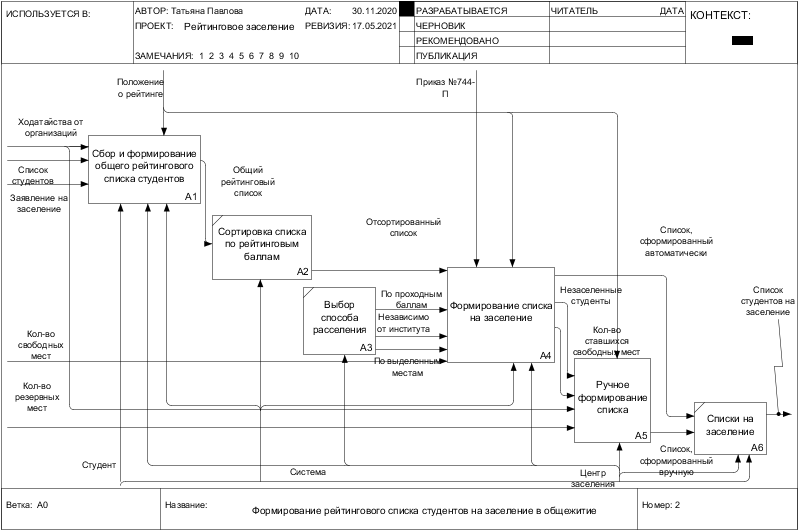
На данном этапе вводится количество мест для каждого института в определенном общежитии и происходит формирование списка студентов на заселение в общежития в зависимости от выбранного способа.

1. Ручное формирование списка.

При ручном формирование списка выбирается студент и назначается общежитие.

1. Списки на заселение.

На данном этапе список полностью формируется и готов к выводу.



* + - 1. ­­­­­­Рисунок 1.2 – Декомпозиция процесса «Формирование рейтингового списка студентов»

Процесс «Сбор и формирование общего рейтингового списка студентов» декомпозируется на следующие этапы (см. рисунок 1.3):

1. Загрузка ходатайств от организаций с последующим сохранением в БД.

На данном этапе пользователь выбирает общежитие или структурное подразделение и загружает соответствующий документ.

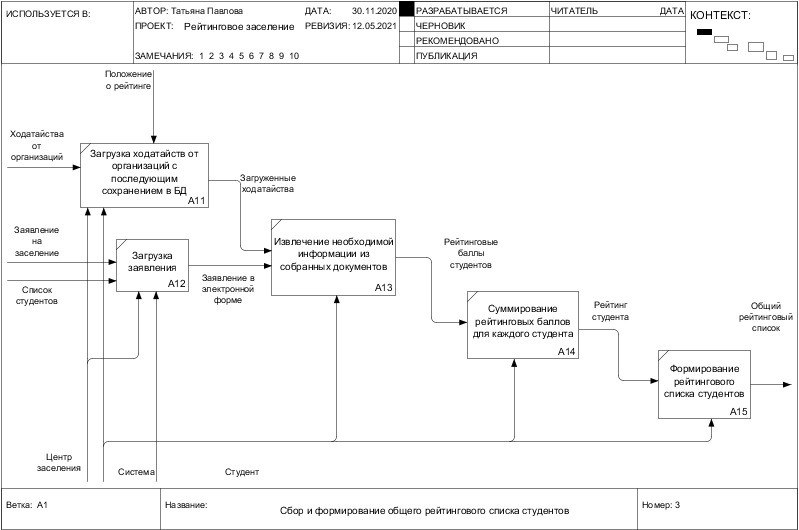
1. Подача заявки.

Студент подает заявку на заселение в общежитие в web-приложении. После подачи заявки студент считается нуждающимся в общежитии.

1. Извлечение необходимой информации из собранных документов.

На данном этапе данные из документов обрабатываются и сохраняются в БД.

1. Суммирование рейтинговых баллов для каждого студента
2. Формирование рейтингового списка студентов.



* + - 1. ­­­­­­Рисунок 1.3 – Декомпозиция процесса «Сбор и формирование общего рейтингового списка студентов»

Процесс «Формирование списка на заселение» декомпозируется на следующие этапы (см. рисунок 1.4):

1. Ввод количества мест для каждого института.
2. Распределение по проходным баллам.

Для распределения по проходным баллам, в первую очередь, рассчитывается проходной балл, после чего список фильтруется по проходному баллу.

Далее данные переходят в следующий процесс. Таким образом удалось избежать повторения процессов.

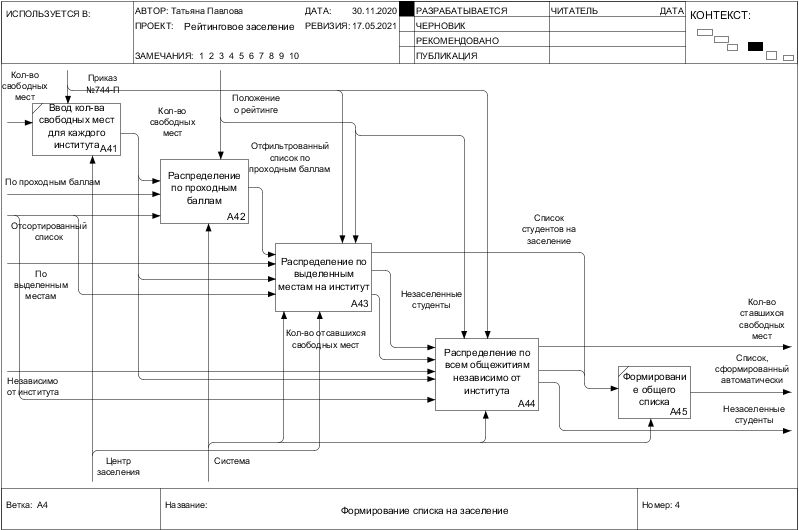
1. Распределение по выделенным местам на институт.

Для распределения по выделенным местам на институт необходимо отфильтровать список по институтам, выбрать соответствующее общежитие, сформировать список на заселение. Далее данные переходят в следующий процесс.

1. Распределение по всем общежитиям независимо от института.

У данного процесса есть два варианта развития событий. Первый, когда данное распределение выбирают изначально, и второй, когда данный процесс является частью одного из двух предыдущих распределений. На данном этапе формируется общий список студентов, отсортированный по баллам, также формируется общее количество мест со всех общежитий. Далее на все места распределяются студенты из списка. Формируется список студентов на заселение.

1. Формирование общего списка.



* + - 1. ­­­­­­Рисунок 1.4 – Декомпозиция процесса «Формирование списка на заселение»

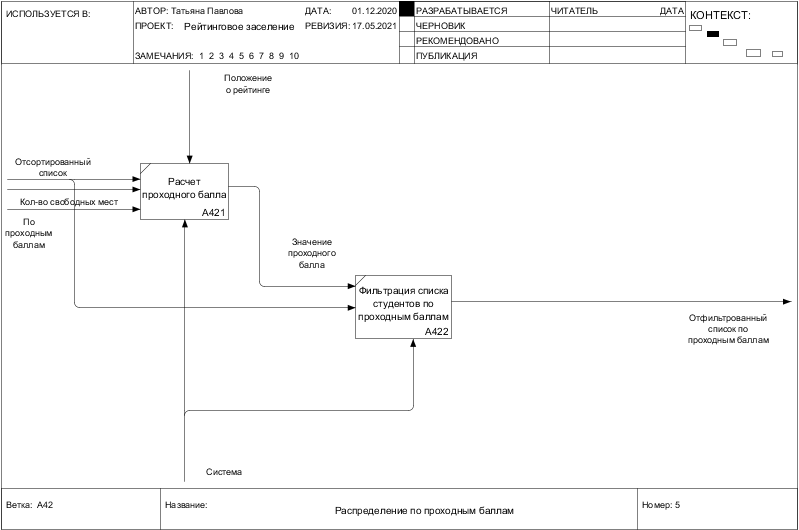
Процесс «Распределение по проходным баллам» декомпозируется на следующие этапы (см. рисунок 1.5):

1. Расчет проходного балла

Проходной балл будет рассчитываться следующим образом: рассчитывается общее количество свободных мест во всех общежитиях, после этого в отсортированном по баллам списке считывается количество баллов студента, чей порядковый номер в списке равен суммарному количеству мест. Данное количество баллов принимается за проходной.

1. Фильтрация списка по проходным баллам

После расчет проходного балла, список фильтруется.



* + - 1. ­­­­­­Рисунок 1.5 – Декомпозиция процесса «Распределение по проходным баллам»

Процесс «Распределение по выделенным местам на институт» декомпозируется на следующие этапы (см. рисунок 1.6):

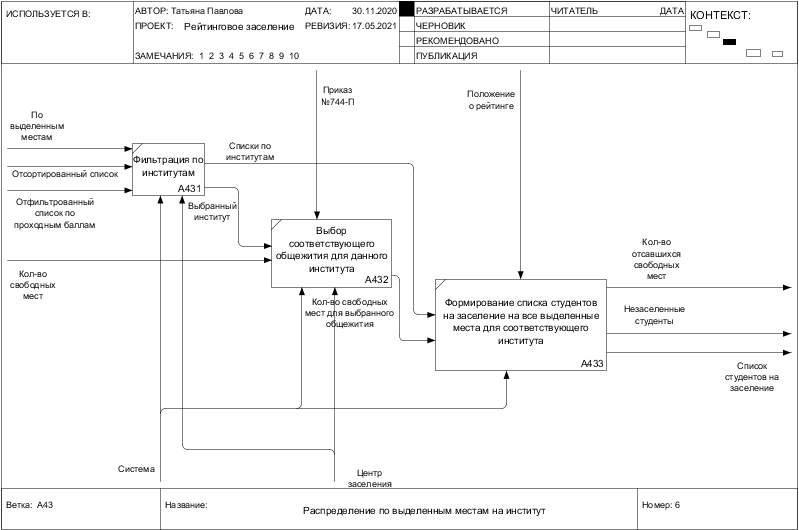
1. Фильтрация по институтам.

Отсортированный список фильтруется по институтам.

1. Выбор соответствующего общежития для данного института.

После этого выбирается общежитие для заселения студентов.

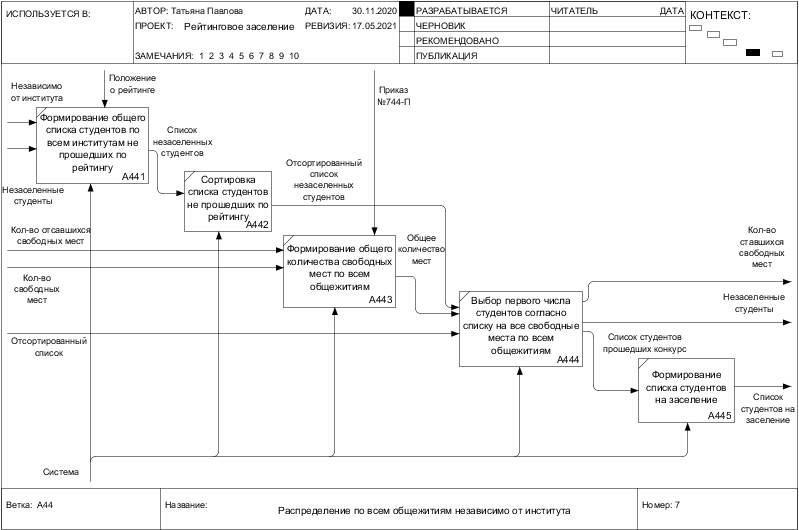
1. Формирование списка студентов на заселение на все выделенные места для соответствующего института.



* + - 1. ­­­­­­Рисунок 1.6 – Декомпозиция процесса «Распределение по выделенным местам на институт»

Процесс «Распределение по всем общежитиям независимо от института» декомпозируется на следующие этапы (см. рисунок 1.7):

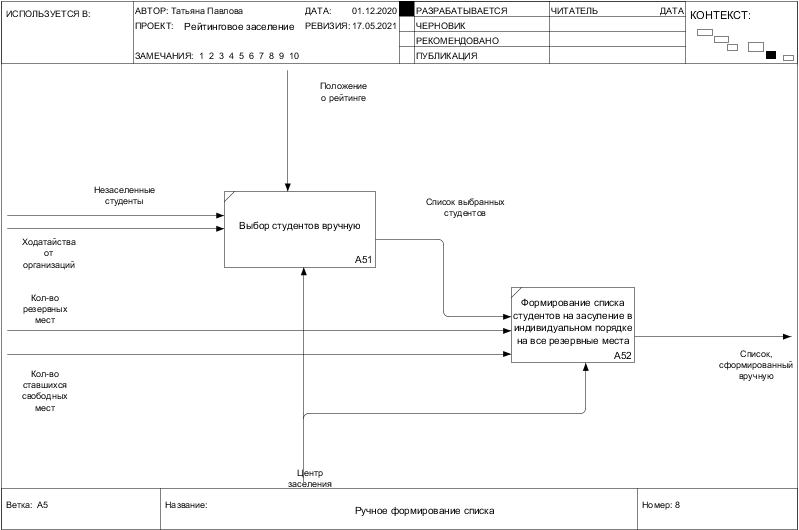
1. Формирование общего списка студентов по всем институтам не прошедших по рейтингу.
2. Сортировка списка студентов, не прошедших по рейтингу.
3. Формирование общего количества свободных мест по всем общежитиям.
4. Выбор первого числа студентов согласно списку, на все свободные места по всем общежитиям.
5. Формирование списка студентов на заселение.



* + - 1. ­­­­­­Рисунок 1.7 – Декомпозиция процесса «Распределение по всем общежитиям независимо от института»

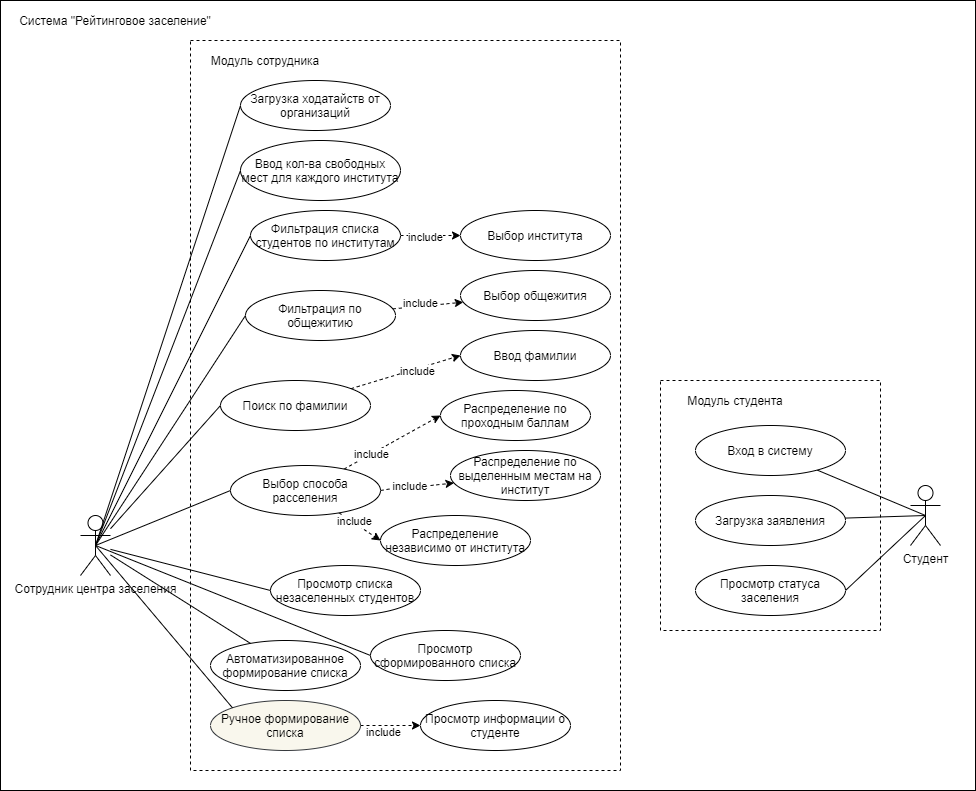
Процесс «Ручное формирование списка» декомпозируется на следующие этапы (см. рисунок 1.8):

1. Выбор студентов вручную.
2. Выбор общежития.
3. Формирование списка студентов на заселение в индивидуальном порядке.



* + - 1. ­­­­­­Рисунок 1.8 – Декомпозиция процесса «Ручное формирование списка»

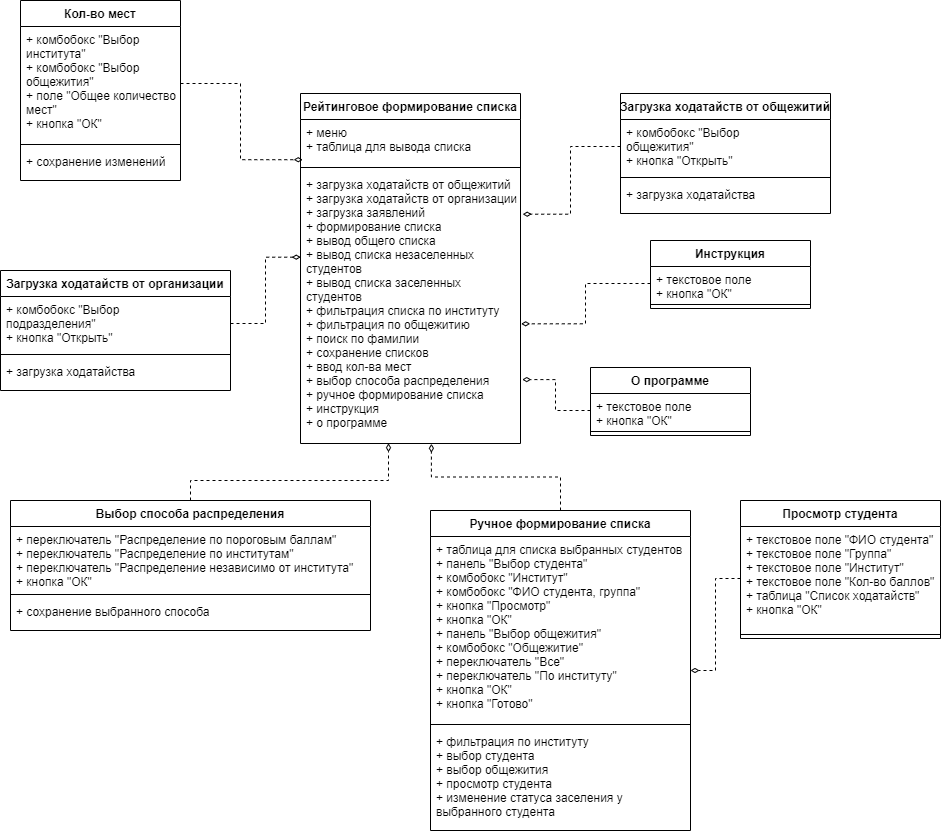
## Формирование требований к функциям пользователей



* + - 1. ­­­­­­Рисунок 1.9 – Диаграмма функций сотрудника Центра заселения и студента

# Проектирование системы

## Описание интерфейса



­­­­­Рисунок 2.1 – Диаграмма интерфейсных классов для WinForms модуля

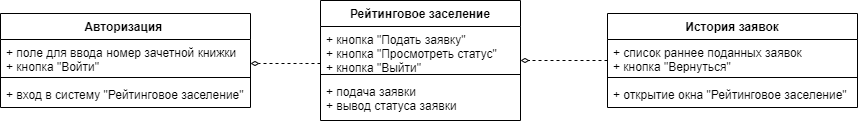
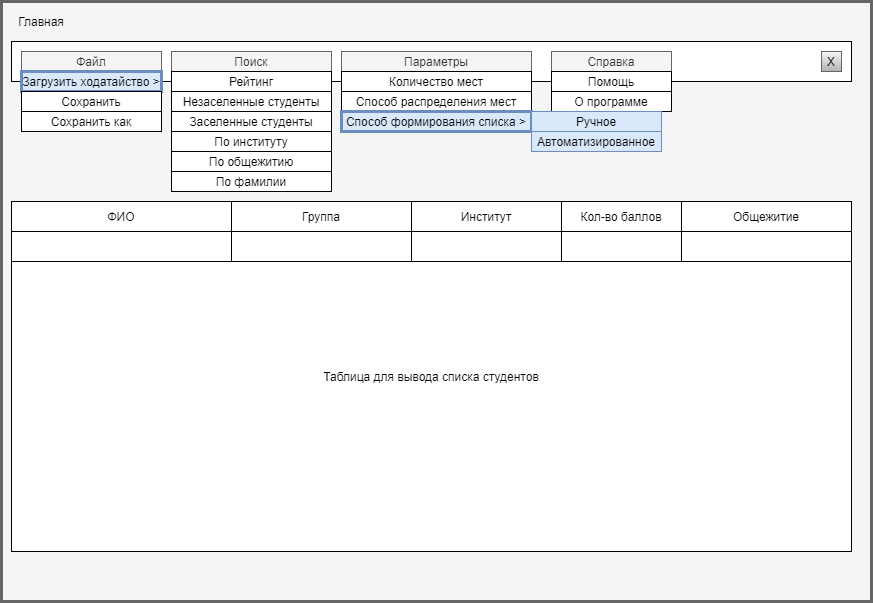
****

Рисунок 2.2 – Диаграмма интерфейсных классов для Web-модуля

При запуске приложения для сотрудника открывается главная форма программы (Рис. 2.3).

**2.1.1 Описание макетов модуля сотрудника**



* + - 1. Рисунок 2.3 – Главная форма системы

Главная форма содержит таблицу для вывода списков студентов со столбцами: «ФИО», «Группа», «Институт», «Кол-во баллов», «Общежитие». Выбор списка осуществляется во вкладке меню «Поиск», некоторые пункты которого недоступны до совершения определенных действий («Рейтинг», «Заселенные студенты», «По общежитию»). После загрузки ходатайств от общежитий и организаций (рис. 2.4) открывается доступ к пункту «Поиск/Рейтинг».

После ввода количества мест (рис. 2.5) и выбора способа распределения (рис. 2.6) активируются два оставшихся пункта «Поиска», а также автоматизированное формирование списка в пункте «Параметры/Формирование списка».

Пустые поля в таблице указывают на недостаточность введенных данных (кроме поля «Общежитие» в списке «Незаселенные студенты»).

При наведении на неактивные пункты меню всплывают подсказки для их активации.

Рассмотрим каждый из пунктов:

* Файл/Загрузить ходатайство

При переходе по данному пункту открывается окно (рис. 2.4), с помощью которого можно выбрать общежитие или организацию, для которого необходимо загрузить соответствующее ходатайство.

* Файл/Сохранить как;

Сохраняет данные из таблицы в файл определенного формата, который выберет пользователь (.xls/.csv). Пункт активируется только тогда, когда таблица не является пустой.

* Поиск/Рейтинг;

Выводит общий рейтинговый список студентов в таблицу. Активируется тогда, когда загружены все ходатайства.

* Поиск/Незаселенные студенты;

Выводит список незаселенных студентов в таблицу на главной форме для просмотра.

* Поиск/Заселенные студенты;

Выводит список заселенных студентов на главную форму для просмотра. Данный пункт активируется тогда, когда был сформирован список студентов на заселение: ручным или автоматизированным способом.

* Поиск/По институту;

Открывает диалоговое окно для выбора института. Выводит отфильтрованный список по институту в таблицу.

* Поиск/По общежитию;

Открывает диалоговое окно для выбора общежития. Выводит отфильтрованный список по общежитию в таблицу.

* Поиск/По фамилии;

Открывает диалоговое окно для ввода фамилии студента. Выводит отфильтрованный список по фамилии.

* Параметры/Количество мест;

Открывает окно с выбором института и общежития для просмотра доступных свободных мест и возможностью редактирования (рис. 2.5);

* Параметры/Способ распределения мест;

Открывает окно с выбором способа распределения. По умолчанию выбрано значение «Распределение независимо от института» (рис. 2.6);

* Параметры/Способ формирования списка/Ручное;

Открывает форму «Ручное формирование списка» для редактирования списка незаселенных студентов с возможностью поиска (рис. 2.7);

* Параметры/Способ формирования списка/Автоматизированное;

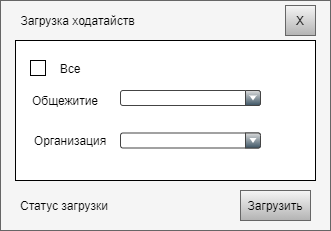
При нажатии на данную ссылку формируется список студентов на заселение. Список выгружается на главную форму (рис. 2.3).

* Справка/Помощь;

Открывает окно с инструкцией по использованию программы;

* Справка/О программе;

Открывает окно с информацией о назначении системы.



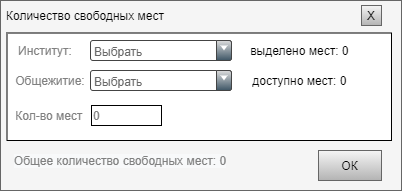
* + - 1. Рисунок 2.4 – Форма «Загрузка ходатайств от общежитий»

Данная форма содержит выплывающие списки с общежитиями и организациями, а также возможность выбора загрузки сразу всех документов.

Поле «Статус загрузки» показывает было загружено/не загружено выбранное ходатайство.

Кнопка «Загрузить» - производит загрузку документа в программу.

После этого статус загрузки меняется, и пользователь может выбрать следующее общежитие/организацию, либо закрыть форму.



* + - 1. Рисунок 2.5 – Форма «Кол-во свободных мест»

При первом открытии данной формы пользователь должен выбрать институт и общежитие, в которое могут быть заселены студенты выбранного института. После чего в поле для ввода ввести необходимое количество мест, которое должно быть меньше или равно доступному количеству мест в выбранном общежитии (отображается справа от выбранного общежития).

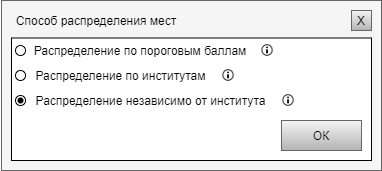
Данная форма содержит два комбобокса с выбором института и общежития.

При выборе института в текстовом поле справа отображается количество выделенных мест для выбранного института, данное число обновляется при каждом вводе, либо изменении количества мест.

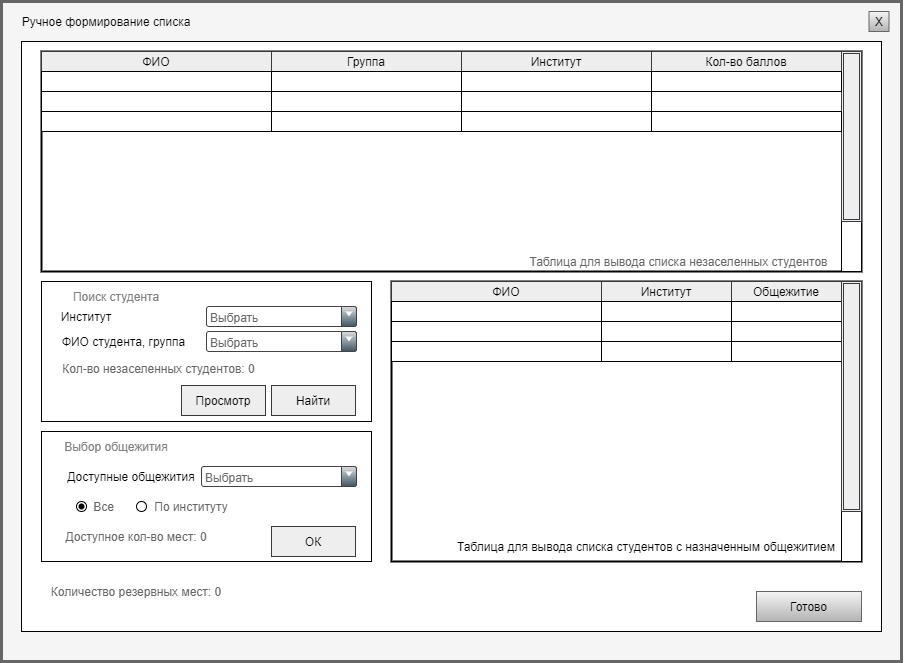
При выборе общежития в текстовом поле справа отображаются доступное количество мест, которые можно распределить между институтами, данное число обновляется при каждом вводе, либо изменении количества мест.

На форме присутствует поле для ввода кол-ва мест для выбранного института в выбранном общежитии.

Поле «Общее количество свободных мест» показывает общее количество свободных мест во всех общежитиях.



* + - 1. Рисунок 2.6 – Форма «Выбор способа распределения»



* + - 1. Рисунок 2.7 – Форма «Ручное формирование списка»

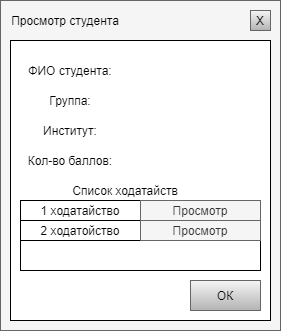
Данная форма предназначена для ручного формирования списка студентов на заселение.

На форме присутствуют две таблицы: таблица для вывода списка незаселенных студентов и таблица для вывода списка заселенных студентов. Для назначения общежития студенту необходимо выбрать студента из таблицы незаселенных студентов. Одно общежитие можно назначать сразу нескольким, либо всем студентам, главное условие при этом, количество выбранных студентов должно быть меньше, либо равно количеству свободных мест в выбранном общежитии. Выбрать общежитие можно в группбоксе «Выбор общежития» с помощью комбобокса «Общежитие». Список общежитий можно отфильтровать по выбранному институту.

Выбрать всех студентов в таблице можно поставив флажок около надписи: «Все».

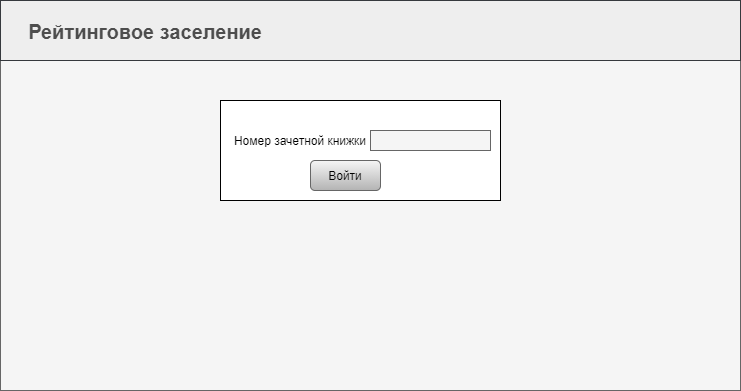
Текстовое поле «Кол-во незаселенных студентов» показывает сколько студентов нуждаются в общежитии в выбранном институте.

Текстовое поле «Доступное кол-во мест» показывает сколько мест свободно в выбранном общежитии.



* + - 1. Рисунок 2.8 – Форма «Просмотр студента»

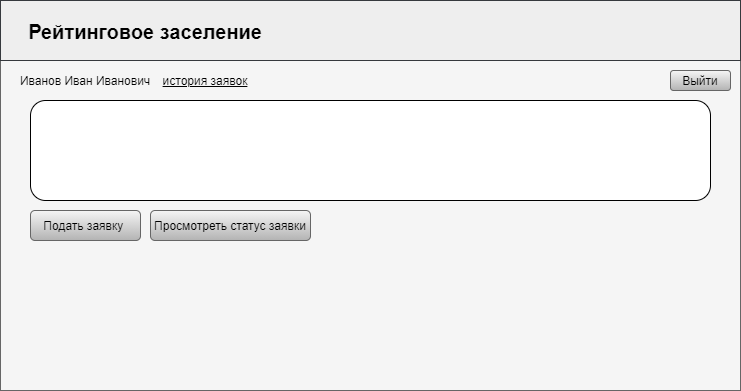
**2.1.2 Описание макетов модуля студента**



* + - 1. Рисунок 2.9 – Окно авторизации

Данная страница предназначена для авторизации студента в системе.

Авторизация будет происходить по номеру зачетной книжки.

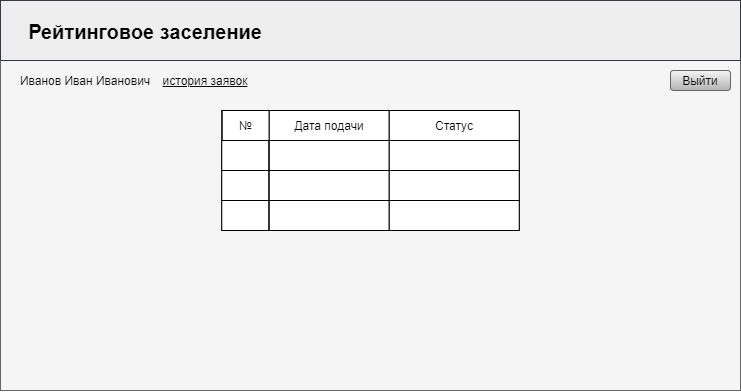


* + - 1. Рисунок 2.10 – Главное окно web-приложения

После авторизации пользователя открывается страница с прототипом личного кабинета студента. На данной странице студент может подать заявку на общежитие, а также просмотреть статус ранее поданной заявки.

Статус заявки будет отображаться в диалоговом окне данной страницы.

Также пользователь может перейти по ссылке «история заявок» для просмотра поданных в течении учебного периода заявок и их статусов (рис. 2.11).



* + - 1. Рисунок 2.11 – Окно "История заявок"

## Обоснование использования программных средств

**Ramus** — это программа, при помощи которой можно создавать визуальные диаграммы, используемые для наглядного отображения различных бизнес процессов. Программа предлагает пользователю удобный встроенный графический редактор для работы над диаграммами. В Ramus реализована методология IDEF0 – методология функционального моделирования для описания бизнес-процессов в виде иерархической системы взаимосвязанных функций [7].

**Draw.io** – это бесплатное онлайн-приложение для создания диаграмм для рабочих процессов, BPM, организационных, сетевых диаграмм. Позволяет сохранять проекты в форматах JPG, PNG, SVG, PDF, HTML, XML. Присутствует возможность интеграции с Google Документами, Dropbox, OneDrive, JIRA, Confluence, Chrome и GitHub [8].

С помощью данного приложения будут построены следующие диаграммы: диаграмма вариантов использования, диаграмма последовательности, диаграмма активности, архитектура системы, инфологическая и логическая модели данных.

**DBDesigner** – это инструмент для визуального проектирования баз данных, позволяющий видеть и управлять всеми связями между таблицами. Это не простой графический редактор для иллюстрации взаимосвязей, это полноценный инструмент для управления базами в визуальном стиле, поддерживающий MySQL, SQLite, Oracle и MSSQL [9].

Программа DBDesigner будет использована для построения физической модели БД.

**Microsoft SQL Server** — система управления реляционными базами данных (СУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL [10].

С помощью СУБД Microsoft SQL Server будет создана БД.

**.Net** – это бесплатная платформа разработки с открытым исходным кодом для создания различных типов приложений, таких как Веб-приложения, веб-API и микрослужбы, облачные приложения, мобильные приложения, классические приложения и тд [11].

**Microsoft Visual Studio** — это версия Visual Studio и .NET Framework, которая поддерживает новые и улучшенные объекты, включает среду разработки с обновленным интерфейсом и отличается интегрированной поддержкой Microsoft SQL Server, позволяя создавать и развертывать проекты с применением сервера баз данных [12].

Данная IDE будет использована для написания web и windows forms приложений, а также для работы с БД с помощью дополнения SQL Server Data Tools.

**C#** – это язык программирования, созданный компанией Microsoft специально ориентированным .NET. Преимуществом C# является то, что это объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения [13].

Язык программирования С# будет использован для написания web и windows forms приложений.

**Windows Forms** — это технология пользовательского интерфейса для .NET, представляющая собой набор управляемых библиотек. Она обеспечивает один из самых эффективных способов создания классических приложений с помощью визуального конструктора в Visual Studio. Такие функции, как размещение визуальных элементов управления путем перетаскивания, упрощают создание классических приложений [14].

С помощью данной технологии будет реализован интерфейс для приложения windows forms.

**LINQ** (Language-Integrated Query) представляет простой и удобный язык запросов к источнику данных. В качестве источника данных может выступать объект, реализующий интерфейс IEnumerable (например, стандартные коллекции, массивы), набор данных DataSet, документ XML. Но вне зависимости от типа источника LINQ позволяет применить ко всем один и тот же подход для выборки данных [15].

Технология LINQ будет использована для доступа к данным MS SQL Server.

**EPPlus** — это библиотека .NET Framework/.NET Core для управления электронными таблицами Office Open XML, распространяемой через Nuget .

Библиотека будет использована для экспортирования и импортирования списков студентов в формат .xlsx [16].

## Архитектура системы

На рисунке 2.12 представлена архитектура разрабатываемой системы.

Для реализации Windows Forms приложения для сотрудников центра заселения будет использован паттерн MVP (Model, View, Presenter).

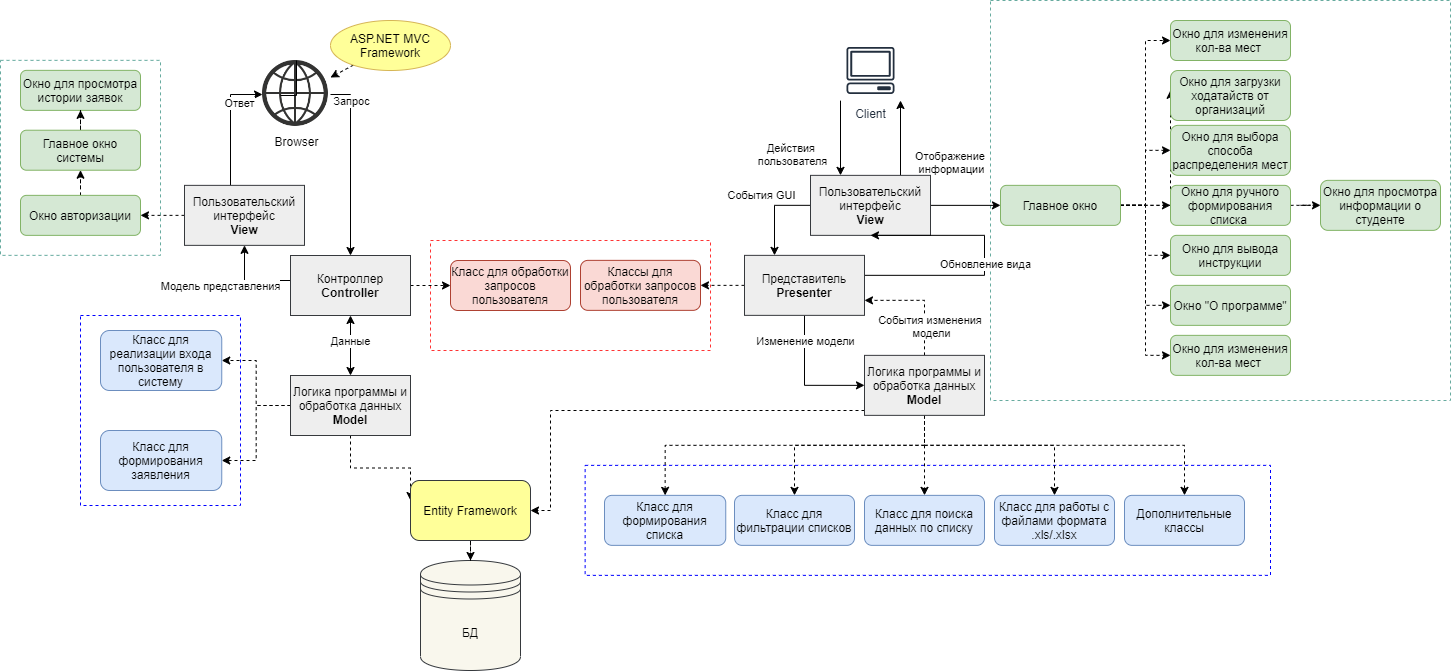
Реализация Web-приложения будет проходить с использованием фреймворка ASP.NET core MVC

Зеленые блоки – окна пользовательского интерфейса (view);

синие блоки – классы для реализации логической части системы (model);

красные блоки – классы для обработки пользовательских запросов (controller/presenter).

|  |
| --- |
| Рисунок 2.12 – Архитектура системы |



## Описание сценариев использования

Ниже представлены текстовые описания сценариев, а также диаграммы функции «ручное формирование» и процесса «выбор способа распределения мест».

### Спецификация варианта использования «Ручное формирование»

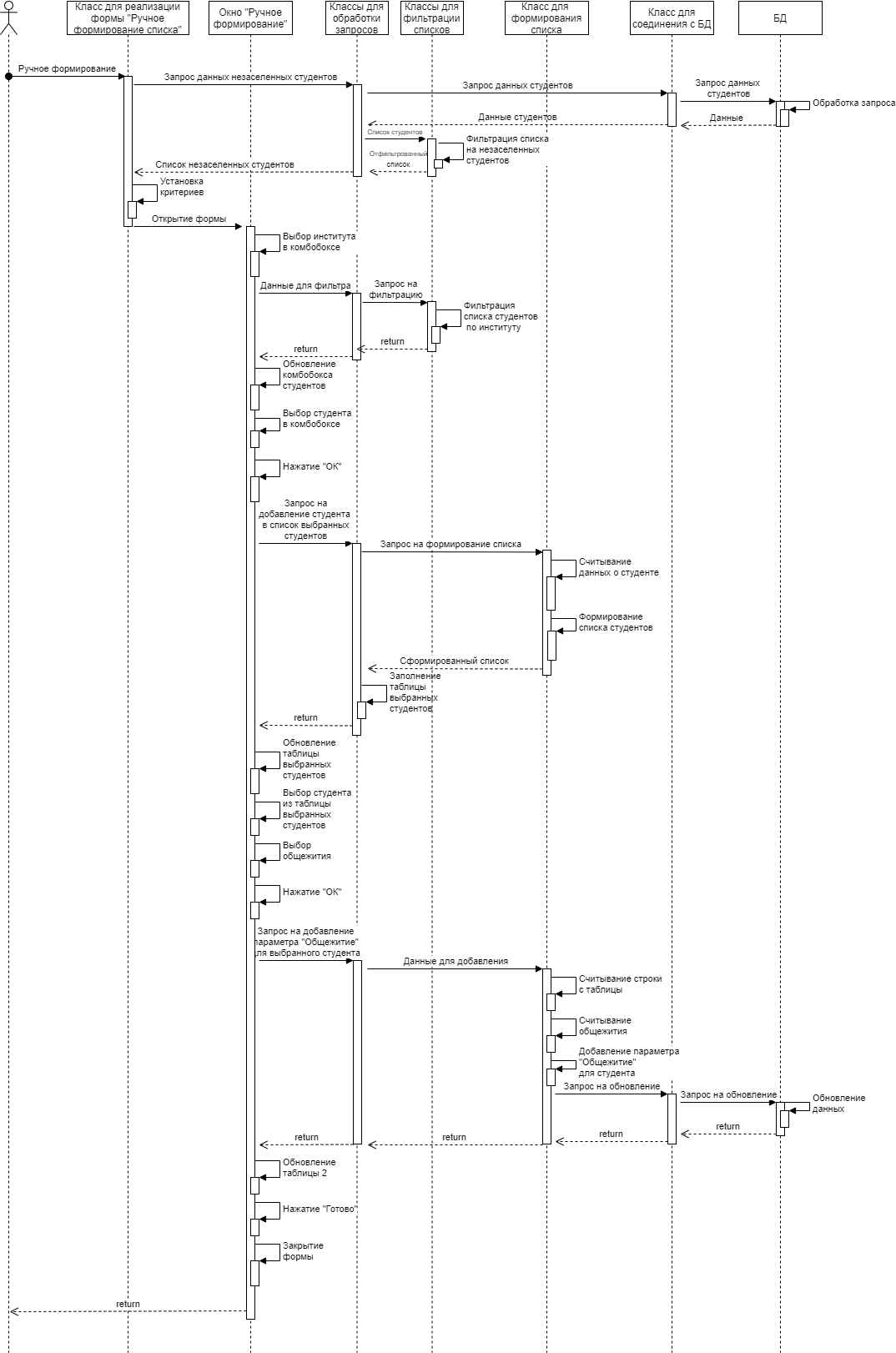
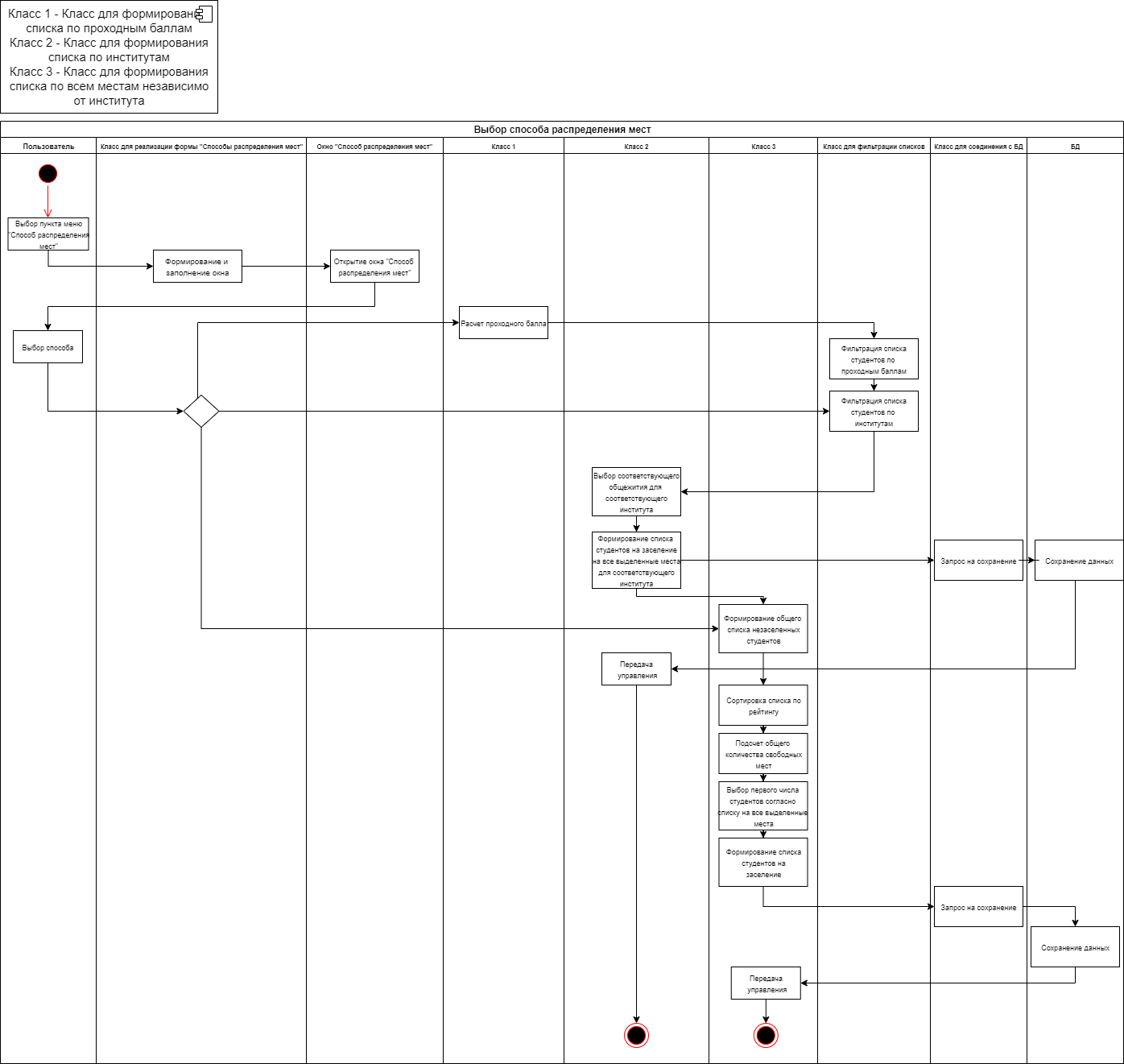


Рисунок 2.13 – Ручное формирование списка

### Спецификация варианта использования «Выбор способа распределения мест»



* + - 1. Рисунок 2.14 – Выбор способа распределения мест

## Проектирование модели данных

Для реализации данного проекта создадим простейшую базу данных, которая должна будет хранить информации о студентах, нуждающихся в общежитии.

На рисунке 2.15 представлена инфологическая модель предметной области, описанная с помощью диаграммы «сущность-связь» по нотации Чена.

Для формирования списка студентов на заселение в базе должны храниться следующие сведения: ФИО студента, курс, институт, количество баллов, номер общежития, а также количество свободных мест в общежитии и количество выделенных мест на институт.

Выделим соответствующие сущности:

- Студент (номер зачетной книжки, ФИО, группа);

- Институт (наименование);

- Общежитие (номер общежития, количество свободных мест).

- Институт\_Общежитие (количество мест на институт)

- Список

- Рейтинговые баллы (успеваемость, общ. деятельность, культурно-массовая деятельность, спортивная деятельность, учебная деятельность, научная деятельность, проживание в общежитии)

- Дата (дата)

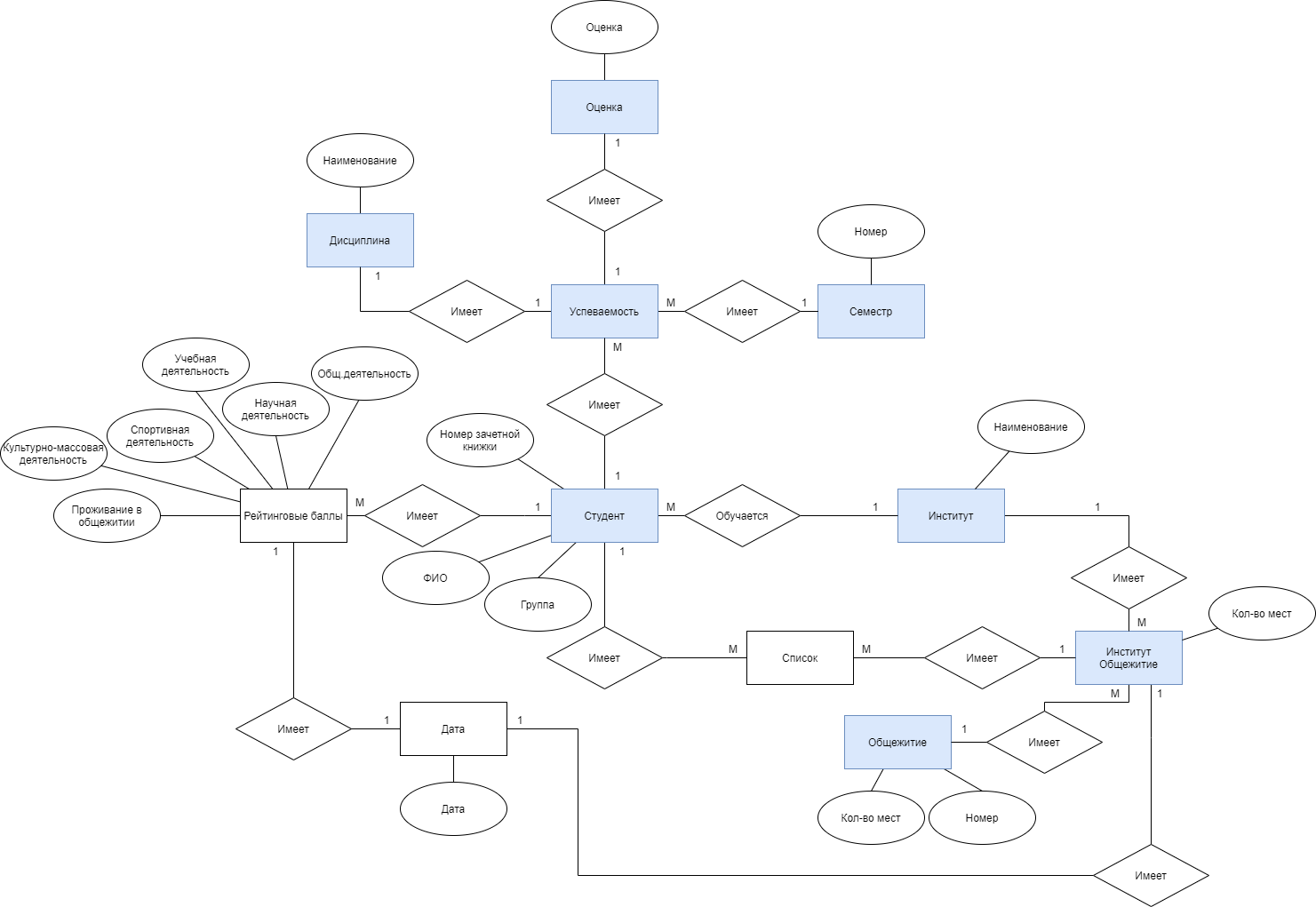
- Успеваемость

- Дисциплина (наименование)

- Оценка (оценка)

- Семестр (номер)

Структура, модели: инфологическая, логическая, физическая



* + - 1. Рисунок 2.15 – Инфологическая модель данных

Синими блоками отмечены сущности централизованной базы данных ИРНИТУ, белыми блоками –дополнительные сущности для успешной работы проектируемой системы.

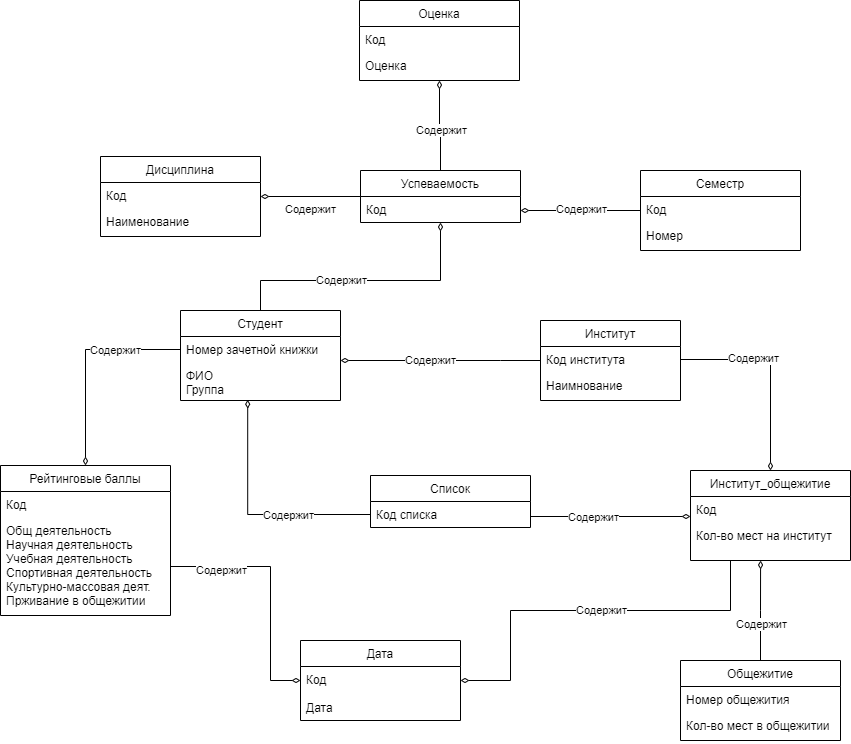
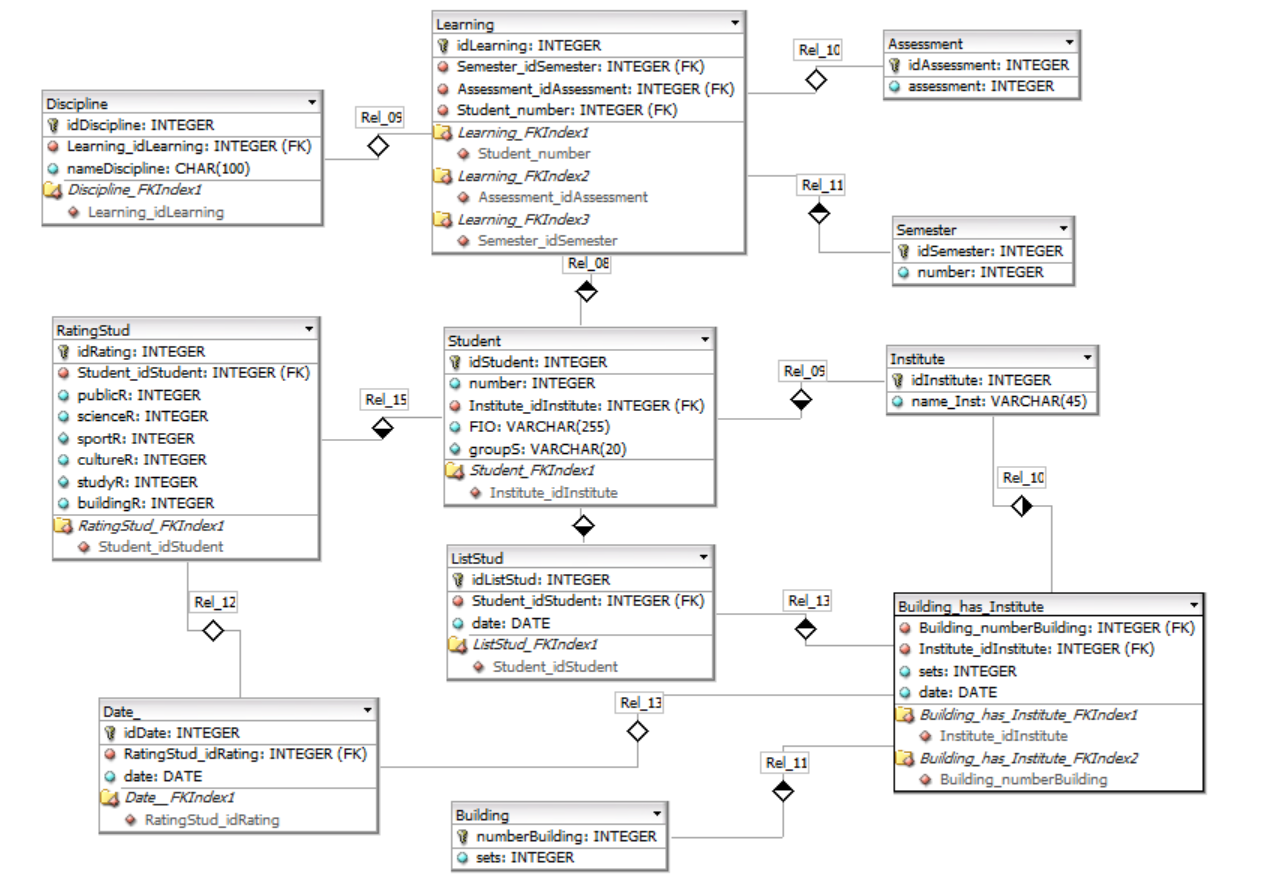


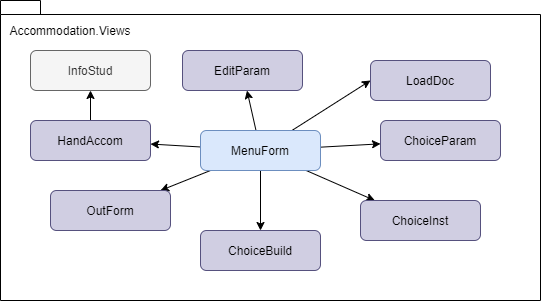
Рисунок 2.16 – Логическая модель данных



* + - 1. Рисунок 2.17 – Физическая модель БД

# Реализация

## Реализация классов пользовательского интерфейса



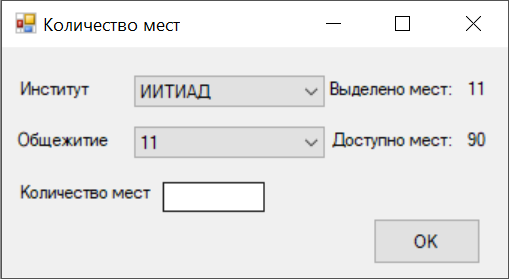
* + - 1. Рисунок 3.1 – Диаграмма классов пользовательского интерфейса

Назначение классов описано в таблице 3.1.

* + - * 1. Назначение классов пакета Views

| **Класс** | **Назначение** |
| --- | --- |
| MenuForm | Главное окно приложения. Содержит меню с функциями системы. |
| EditParam | Окно для редактирования/ввода количество мест , занимаемое институтом в конкретном общежитии |
| ChoiceInst | Диалоговое окно выбора института для фильтрации |
| ChoiceBuild | Диалоговое окно выбора общежития для фильтрации |
| ChoiceParam | Окно выбора способа для автоматизированного распределения мест |
| HandAccom | Окно для ручного формирования списка студентов на заселение в общежития |
| InfoStud | Форма для отображения информации о студенте. |
| LoadDoc | Окно для загрузки ходатайств от организаций и общежитий. |
| OutForm | Диалоговое окно вывода сообщений |

## Реализация функции «Ввод количества мест»



* + - 1. Рисунок 3.2 – Ввод количества мест

Данная форма предназначена для распределения мест между институтами в общежитиях

Реализуется с помощью класса EditParam.

**Часть листинга класса EditParam для заполнения формы**

private void upLbl()

{

int idBuild = (int)comboBox2.SelectedValue;

var result = db.SelSets(idBuild);

if (result != null) lblAc.Text = result.ToString();

else lblAc.Text = "0";

int idInst = (int)comboBox1.SelectedValue;

var result1 = db.setLblSel(idInst);

if (result1 != null) lblSel.Text = result1.ToString();

else lblSel.Text = "0";

}

private void upCmb()

{

var q = from i in dataEntities.Institutes

select new { Name = i.name\_Inst, ID = i.idInstitute };

comboBox1.DataSource = q.ToList();

comboBox1.DisplayMember = "Name";

comboBox1.ValueMember = "ID";

fillCmb();

upLbl();

}

private void fillCmb()

{

var q2 = from b in dataEntities.Building

join ib in dataEntities.Institutes\_has\_Building on b.idBuilding equals ib.Building\_idBuilding

where ib.Institutes\_idInstitute == (int)comboBox1.SelectedValue

select new { Name = b.numberBuilding, ID = b.idBuilding };

comboBox2.DataSource = q2.ToList();

comboBox2.DisplayMember = "Name";

comboBox2.ValueMember = "ID";

upLbl();

}

## Реализация функции «Вывод списка студентов»

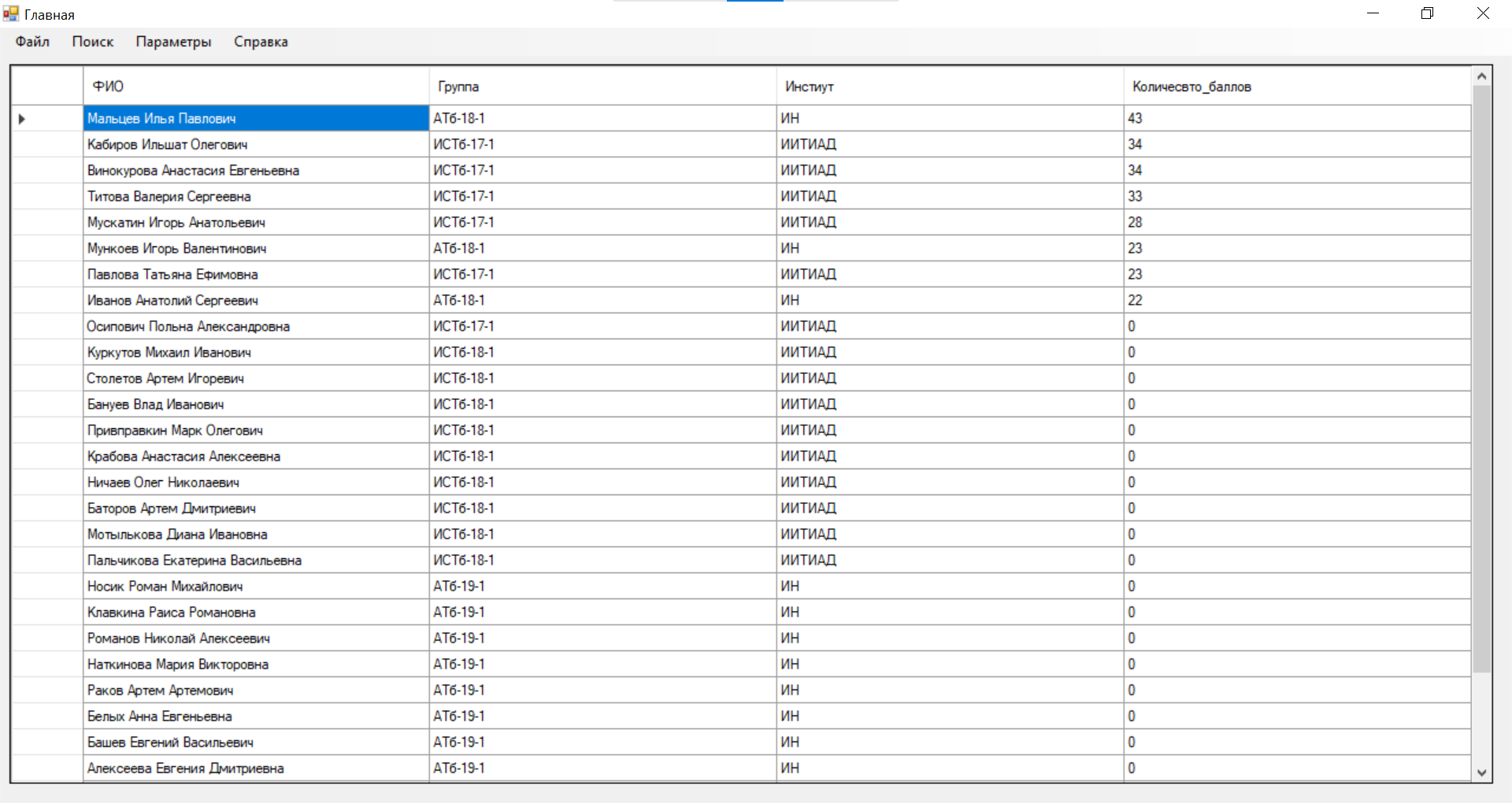


Рисунок 3.3 – Вывод списка студентов

На данной форме реализована функция вывода рейтингового списка.

Также в системе предусмотрен вывод списка незаселенных студентов и вывод сформированного списка.

Ниже представлен листинг метода, реализующего вывод рейтингового списка.

**Листинг метода fillTable**

private void fillTable()

{

var query = from s in dataEntities.Students

join i in dataEntities.Institutes

on s.Institutes\_idInstitute equals i.idInstitute

orderby s.countRate descending

where s.statement == true

select new { ФИО = s.FIO, Группа = s.groupS, Инстиут = i.name\_Inst, Количесвто\_баллов = s.countRate };

dataGridView1.DataSource = query.ToList();

}

## Реализация функции «Ручное формирование списка»

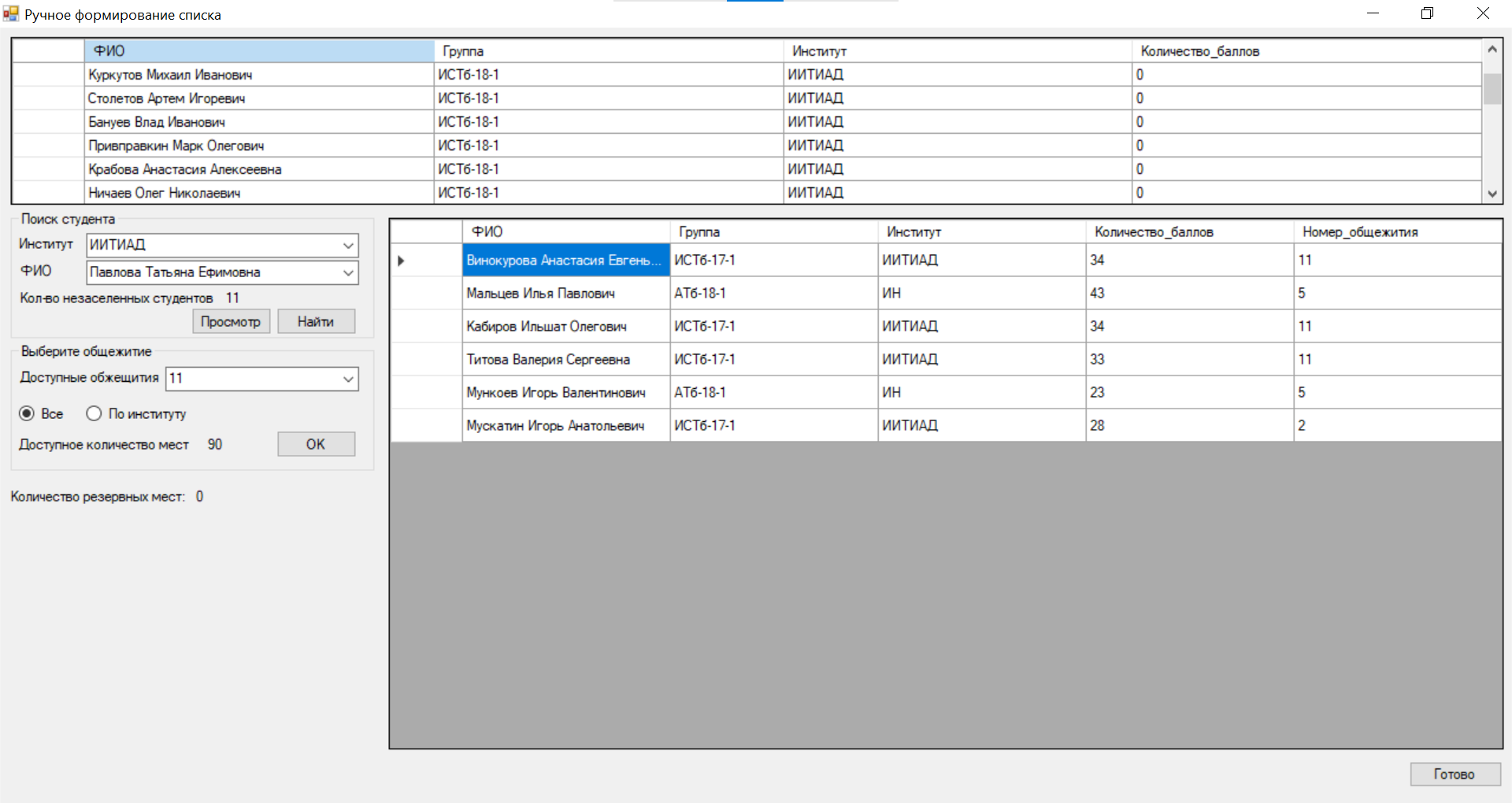


Рисунок 3.4 – Ручное формирование списка

**Листинг части класса, реализующего ручное формирование списка**

private void updateCmb3()

{

if (rdbAll.Checked)

{

var query2 = from b in dataEntities.Building

select new { Name = b.numberBuilding, ID = b.idBuilding };

comboBox3.DataSource = query2.ToList();

comboBox3.DisplayMember = "Name";

comboBox3.ValueMember = "ID";

}

if (rdbInst.Checked)

{

var query2 = from b in dataEntities.Building

join ib in dataEntities.Institutes\_has\_Building

on b.idBuilding equals ib.Building\_idBuilding

where ib.Institutes\_idInstitute == (int)comboBox1.SelectedValue

select new { Name = b.numberBuilding, ID = b.idBuilding };

comboBox3.DataSource = query2.ToList();

comboBox3.DisplayMember = "Name";

comboBox3.ValueMember = "ID";

}

}

private void upLabel()

{

int idInst = (int)comboBox1.SelectedValue;

object result = db.countSt(idInst);

if(result != null) lblS.Text = result.ToString();

else lblS.Text = "0";

}

public void addStud(int idStud, int idBuild)

{

string sqlExpression = "AddStudList";

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

command.CommandType = CommandType.StoredProcedure;

SqlParameter idBuildParam = new SqlParameter

{

ParameterName = "@idBuild",

Value = idBuild

};

command.Parameters.Add(idBuildParam);

SqlParameter idStudParam = new SqlParameter

{

ParameterName = "@idStud",

Value = idStud

};

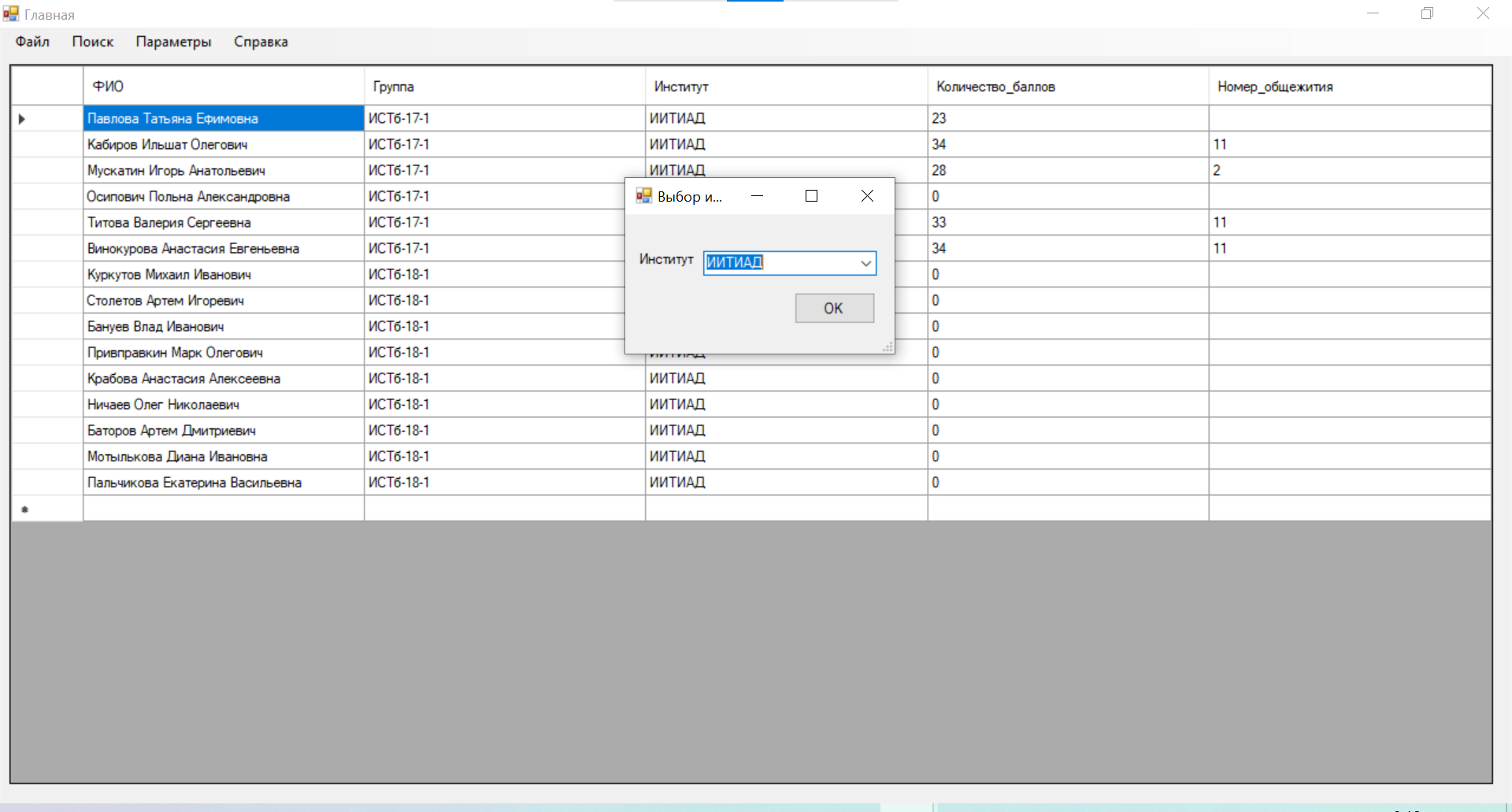
command.Parameters.Add(idStudParam);

var result = command.ExecuteNonQuery();

}

}

## Реализация функции «Фильтрация списка студентов»



* + - 1. Рисунок 3.5 – Фильтрация списка студентов

**Листинг метода filterTable**

public DataTable filterTable(string param, string a)

{

DataTable table = db.loadTable();

table.DefaultView.RowFilter = $"{param} like '%{a}%'";

return table;

}

## Экспорт/импорт Excel-файлов

class ExcelClass

{

public void writeExcel(DataTable table)

{

Excel.Application exApp = new Excel.Application();

exApp.Workbooks.Add();

Excel.Worksheet wsh = (Excel.Worksheet)exApp.ActiveSheet;

int i, j;

for (i = 0; i < table.Columns.Count; i++)

{

wsh.Cells[1, i + 1] = table.Columns[i];

}

var a = table.Columns[0];

for (i = 1; i <= table.Rows.Count; i++)

{

for (j = 0; j <= table.Columns.Count - 1; j++)

{

if (table.Rows[i - 1].Cells[j].Value != null) wsh.Cells[i + 1, j + 1] = table.Rows[i - 1].Cells[j].Value.ToString();

else wsh.Cells[i + 1, j + 1] = "Отсутсвует";

}

}

exApp.Visible = true;

exApp.UserControl = true;

}

public List<List<string>> readExcel(string fileName)

{

Excel.Application excelApp = new Excel.Application();

excelApp.Visible = true;

excelApp.Workbooks.Open(fileName);

int row = 1;

List<List<string>> maping = new List<List<string>>();

Excel.Worksheet currentSheet = (Excel.Worksheet)excelApp.Workbooks[1].Worksheets[1];

while (currentSheet.get\_Range("A" + row).Value2 != null)

{

List<string> tempList = new List<string>();

for (char column = 'A'; column < 'J'; column++)

{

Excel.Range cell = currentSheet.get\_Range(column + row);

tempList.Add (cell != null ? cell.Value2.ToString() : "");

}

maping.Add(tempList);

row++;

}

excelApp.Quit();

return maping;

}

}

## Результаты экспорта списка студентов в Excel

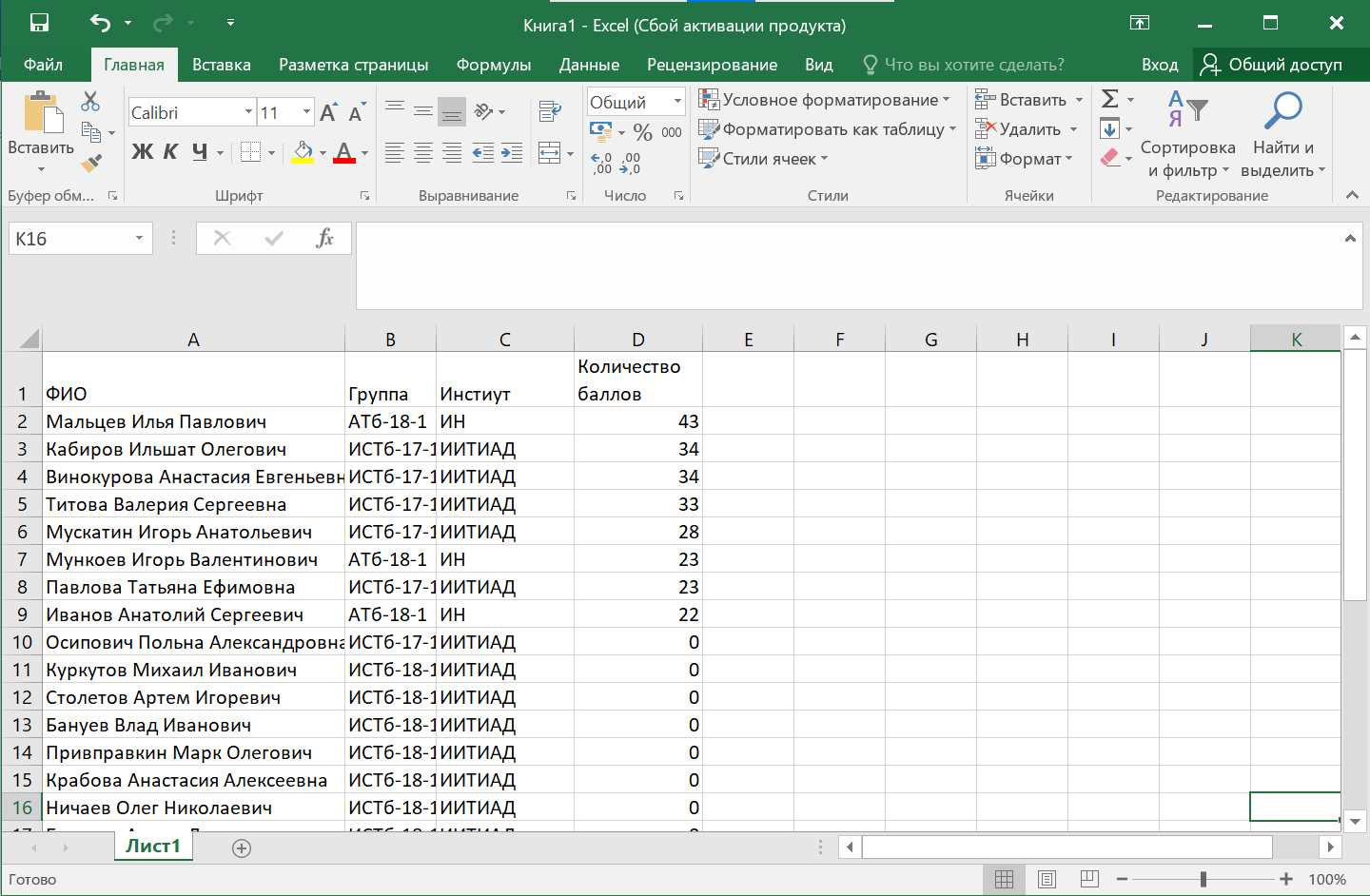


Рисунок 3.6 – Результат экспорта в Excel

## Соединение с БД

Соединение с базой данных было осуществлено с помощью DataEntity Framework.

**Листинг класса ConnectionDB**

class ConnectionDB

{

AccommodationDBEntities2 dataEntities = new AccommodationDBEntities2();

string connectionString = "Server=(localdb)\\MSSQLLocalDB;Database=AccommodationDB;Trusted\_Connection=True;";

public void addStud(int idStud, int idBuild)

{

string sqlExpression = "AddStudList";

using (SqlConnection connection = new SqlConnection(connectionString))

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

// указываем, что команда представляет хранимую процедуру

command.CommandType = CommandType.StoredProcedure;

SqlParameter idBuildParam = new SqlParameter

{

ParameterName = "@idBuild",

Value = idBuild

};

command.Parameters.Add(idBuildParam);

SqlParameter idStudParam = new SqlParameter

{

ParameterName = "@idStud",

Value = idStud

};

command.Parameters.Add(idStudParam);

var result = command.ExecuteNonQuery();

}

}

}

public object countSt(int idInst)

{

object result;

string sqlExpression = "CountSt";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

command.CommandType = CommandType.StoredProcedure;

SqlParameter idInstParam = new SqlParameter

{

ParameterName = "@idInst",

Value = idInst

};

command.Parameters.Add(idInstParam);

result = command.ExecuteScalar();

}

return result;

}

public void createForInst()

{

var q = from s in dataEntities.Students

orderby s.countRate descending

where s.statement == true

where !dataEntities.ListStud.Any(ls => (ls.Students\_idStudent == s.idStudent))

select s;

var arrS = q.ToArray();

foreach (var a in arrS)

{

var q2 = from ib in dataEntities.Institutes\_has\_Building

where ib.Institutes\_idInstitute == a.Institutes\_idInstitute

select ib;

var arrB = q2.ToArray();

foreach (var ib in arrB)

{

if (ib.sets > 0)

{

addStud(a.idStudent, ib.Building\_idBuilding);

break;

}

}

}

}

public DataTable loadTable()

{

var query = dataEntities.View1;

List<View1> list = new List<View1>();

list = query.ToList();

DataTable table = new DataTable();

using (var reader = ObjectReader.Create(list))

{

table.Load(reader);

}

return table;

}

public List<View3\_1> loadTableUnBuild()

{

var query = dataEntities.View3\_1;

return query.ToList();

}

public List<View2> loadTableBuild()

{

var query = dataEntities.View2;

return query.ToList();

}

public object SelSets(int idBuild)

{

object result;

string sqlExpression = "SelSets";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

// указываем, что команда представляет хранимую процедуру

command.CommandType = CommandType.StoredProcedure;

SqlParameter idBuildParam = new SqlParameter

{

ParameterName = "@idBuild",

Value = idBuild

};

command.Parameters.Add(idBuildParam);

result = command.ExecuteScalar();

}

return result;

}

public object setLblSel(int idInst)

{

object result;

string sqlE = "SelSetsInst";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlE, connection);

// указываем, что команда представляет хранимую процедуру

command.CommandType = CommandType.StoredProcedure;

SqlParameter idInstParam = new SqlParameter

{

ParameterName = "@idInst",

Value = idInst

};

command.Parameters.Add(idInstParam);

result = command.ExecuteScalar();

}

return result;

}

public void addSets(int idInst, int idBuild, int set)

{

string sqlE = "AddSetsInst";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlE, connection);

// указываем, что команда представляет хранимую процедуру

command.CommandType = CommandType.StoredProcedure;

SqlParameter idInstParam = new SqlParameter

{

ParameterName = "@idInst",

Value = idInst

};

command.Parameters.Add(idInstParam);

SqlParameter idBuildParam = new SqlParameter

{

ParameterName = "@idBuild",

Value = idBuild

};

command.Parameters.Add(idBuildParam);

SqlParameter idSetParam = new SqlParameter

{

ParameterName = "@set",

Value = set

};

command.Parameters.Add(idSetParam);

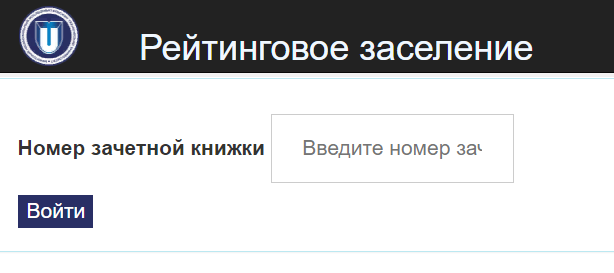
var result = command.ExecuteNonQuery();

}

}

}

## Реализация функции «Вход в систему»



* + - 1. Рисунок 3.7 – Авторизация пользователя «Студент»

**Листинг класса Login**

public partial class Login : System.Web.UI.Page

{

string connectionString = "Server=(localdb)\\MSSQLLocalDB;Database=AccommodationDB;Trusted\_Connection=True;";

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button2\_Click(object sender, EventArgs e)

{

try

{

int number = int.Parse(TextBox2.Text);

string sqlExpression = "SearchNumber";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

command.CommandType = System.Data.CommandType.StoredProcedure;

SqlParameter numberParam = new SqlParameter

{

ParameterName = "@number",

Value = number

};

command.Parameters.Add(numberParam);

var result = command.ExecuteScalar();

if (Request.Cookies["idStud"] != null)

{

Response.Cookies["idStud"].Expires = DateTime.Now.AddDays(-1);

}

Response.Cookies.Add(new HttpCookie("idStud", result.ToString()));

}

Response.Redirect("MainForm.aspx");

}

catch (Exception ex)

{

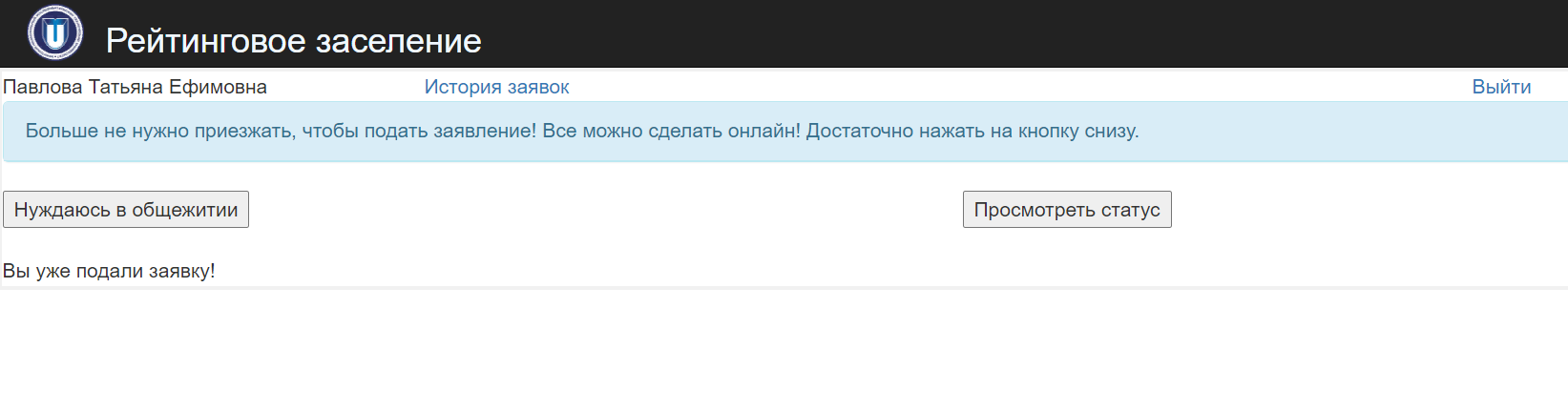
Label1.Text = ex.Message;

}

}

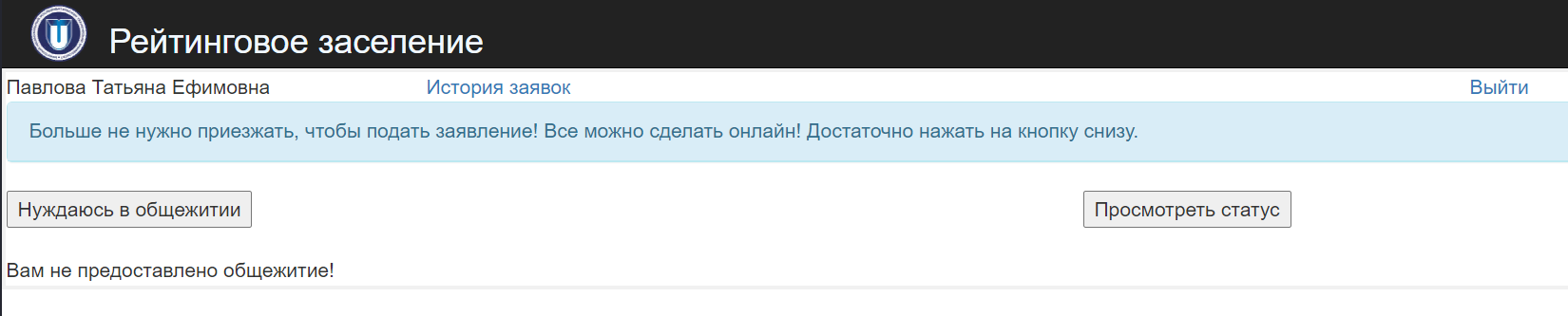
}

## Реализация функции «Подача заявки»



* + - 1. Рисунок 3.8 – Подача заявки

## Реализация функции «Просмотр статуса»



* + - 1. Рисунок 3.9 – Просмотр статуса заявки

Функции подача и просмотр статуса заявки осуществляются с помощью класса MainForm.cs.

**Листинг класса MainForm**

public partial class MainForm : System.Web.UI.Page

{

string connectionString = "Server=(localdb)\\MSSQLLocalDB;Database=AccommodationDB;Trusted\_Connection=True;";

protected void Page\_Load(object sender, EventArgs e)

{

getFIO();

}

private int getIdSrud()

{

HttpCookieCollection myId = Request.Cookies;

String[] arr = myId.AllKeys;

return int.Parse(myId[arr[0]].Value);

}

protected void btn1\_Click(object sender, EventArgs e)

{

try

{

int id = getIdSrud();

string sqlExpression = "State";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

command.CommandType = System.Data.CommandType.StoredProcedure;

SqlParameter idParam = new SqlParameter

{

ParameterName = "@idStud",

Value = id

};

command.Parameters.Add(idParam);

var result = command.ExecuteNonQuery();

Label2.Text = "Заявка отправлена!";

}

}

catch (Exception ex)

{

Label2.Text = ex.Message;

}

}

private void getFIO()

{

int id = getIdSrud();

string sqlExpression = "getFio";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

command.CommandType = System.Data.CommandType.StoredProcedure;

// параметр для ввода имени

SqlParameter idParam = new SqlParameter

{

ParameterName = "@idStud",

Value = id

};

// добавляем параметр

command.Parameters.Add(idParam);

var result = command.ExecuteScalar();

Label1.Text = result.ToString();

}

}

protected void btn2\_Click(object sender, EventArgs e)

{

try

{

int id = getIdSrud();

string sqlExpression = "SearchStudent";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

command.CommandType = System.Data.CommandType.StoredProcedure;

SqlParameter idParam = new SqlParameter

{

ParameterName = "@id",

Value = id

};

command.Parameters.Add(idParam);

var result = command.ExecuteNonQuery();

}

}

catch (Exception ex)

{

Label2.Text = ex.Message;

}

}

}

}

# Экономическое обоснование

**Расчёт единовременных затрат разработчика**

К единовременным затратам относятся затраты процесса проектирования программного обеспечения. В нашем случае существует 5 стадий проектирования.

* + - * 1. Содержание стадий разработки проекта ИС

|  |  |  |
| --- | --- | --- |
| **Этап** | **Трудоемкость, дн** | **Трудоемкость, %** |
| Предварительный этап | 3 | 7 |
| Сбор и анализ требований | 5 | 12 |
| Проектирование | 13 | 31 |
| Реализация | 18 | 43 |
| Тестирование | 3 | 7 |
| **Итого** | **42** | **100** |

К единовременным затратам разработчика отнесём затраты на разработку программного обеспечения для выпускной квалификационной работы (ВКР).

К затратам по экономическим элементам на создание информационной системы относятся:

* материальные затраты
* затраты на электроэнергию.

**Материальные затраты**

Под материальными затратами понимают отчисления на материалы, использующиеся в процессе разработки и внедрении программного продукта (в т.ч. стоимость бумаги, картриджей для принтера, дискет, дисков и т.д.) по действующим ценам.

В процессе работы использовались следующие материалы и принадлежности, представленные в смете.

* + - * 1. Материальные затраты

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование** | **Цена** | **Количество** | **Стоимость** |
| Ручки, карандаши и тд. | 5 руб | 2 | 10 |
| Печать (1 лист) | 2 руб | 80 | 160 |
| Интернет (1 гб) | 50 руб | 5 | 250 |
| **Итого** | | | **400** |

**Затраты на электроэнергию**

В процессе разработки проекта информационной системы для работы использовался ноутбук Acer Swift SF114-32. Потребление электроэнергии ноутбуком – 0,12 кВт/ч/

Стоимость 1кВт электроэнергии составляет 1,11 руб.

Стоимость машинного часа работы равна:

Время работы ЭВМ, :

Себестоимость электроэнергии рассчитывается по формуле:

Используя формулу (4), получаем:

По данным вышеприведённых расчётов составляется смета затрат на программное обеспечение.

* + - * 1. Смета затрат

|  |  |
| --- | --- |
| **Элемент затрат** | **Сметная стоимость, руб** |
| Материальные затраты | 400 руб |
| Затраты на электроэнергию | 27,972 руб |
| **Итого** | **427,972 руб** |

Затраты на ВКР равны:

**Расчет экономической выгоды по времени**

Для расчета временной выгоды необходимо узнать, какое количество времени тратится сотрудником центра заселения для принятия решения о заселении или незаселении студента.

При заселении студентов сотрудник центра заселения опирается, в первую очередь, на успеваемость студентов, иными словами, на отсутствие задолженностей. То есть студент 2-4 курса, нуждающийся в общежитии и не имея задолженностей будет заселен. У сотрудника на таких студентов уходит чуть менее 5 минуты: узнать данные студента, убедиться в отсутствие задолженностей и приступить к оформлению. Для студентов с задолженностями тратится немного больше времени. Сотруднику необходимо проверить данные студента по нескольким критериям, помимо его успеваемости: активное участие в какой-либо организации ИРНИТУ, а также отсутствие нарушений правил проживания в общежитии за предыдущий год. На таких студентов у сотрудника тратится около 7 минут.

Общежития ИРНИТУ вмещают в себя, в общей сложности, 4000 студентов (по данным приказа «О порядке заселения студентов в общежития ИРНИТУ» за 2020 г. [2]). Вычитаем из этого количества места, выделенные для международных студентов, аспирантов и студентов геологоразведочного техникума, получаем 3376 мест. Предположим, что около 50% мест занимают студенты 1 курса, тогда для 2-4 курсов остается 1688 мест. Также около 30% студентов не имеют задолженностей.

Рассчитаем время, затрачиваемое одним сотрудником для рассмотрения заявлением от студентов без задолженностей ():

Рассчитаем время, затрачиваемой одним сотрудником для рассмотрения заявлений от студентов, имеющих задолженности ():

Также учтем, что около 4-5% студентов от числа заселенных получают отказ в заселении (). На их рассмотрение тратится также около 7 минут:

Общее время:

По данным предыдущего года, в центре заселения одновременно работают около 20 человек, соответственно, время (В), затрачиваемое на рассмотрение заявок, будет равно:

С помощью разрабатываемой системы, в свою очередь, на формирование списка сотрудник потратит приблизительно 30 мин., с учетом загрузки необходимых документов в систему.

Следовательно, временная выгода () от внедрения системы составляет:

# Безопасность жизнедеятельности

Безопасность жизнедеятельности – это наука, изучающая безопасное взаимодействие человека с техносферой. Является областью знаний, изучающей опасности, с которыми может столкнуться человек в рабочем пространстве, а также способа защиты человека от этих опасностей.

Большинство людей на работе сталкиваются с опасными и вредными факторами, которые так или иначе влияют на здоровье и работоспособность. Существуют различные способы защититься от их воздействия.

Целью данного раздела является анализ и изучение рабочего пространства. В данном случае рассматривается кабинет (отдел по работе со студентами и выпускниками) Е323.

Задачи раздела:

1. Привести краткую характеристику рассматриваемого объекта;
2. Провести анализ на уровень вредных и опасных факторов в рабочем пространстве;
3. Рассчитать освещение в помещении Центра заселения ИРНИТУ
4. Провести анализ электробезопасности;
5. Провести анализ пожаробезопасности.

## Краткая характеристика объекта

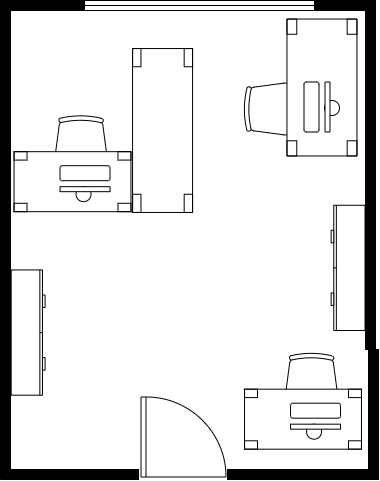
Рассматриваемое рабочее пространство представляет собой кабинет (отдел по работе со студентами и выпускниками) с идентификационным номером Е323. Помещение является частью университета. Е – это корпус, 323 – порядковый номер помещения.

Часть Иркутской области вблизи озера Байкал (Прибайкалье) является одним из сейсмических районов России. Здание университета построено с учетом сейсмической активности региона, и отвечает всем требованиям СНиП II-7-81[17]. В здании предусмотрены запасные выходы, сводные лестничные проходы, вентиляция, равномерное освещение, противопожарные датчики и тд.

Университет расположен на достаточно близком расстоянии к реке. Но благодаря возвышенной территории наводнение ему не грозит. Организована система канализаций, с помощью которых отходы выводятся в стоки, мусор вывозят специализированной техникой, университет не производит вредных выбросов в окружающую среду.

Кабинет Е323 находится на первом этаже университета, коридоры корпуса имеют равномерное и регулируемое освещение от искусственных источников. В помещениях корпуса присутствует интернет, освещение, локальная компьютерная сеть, вентиляция, кондиционеры и отопительные батареи.

Рассматриваемое помещение (Е323) представлено на рисунке 5.1



* + - 1. Рисунок 5.1 – Схема рабочего помещения, кабинет Е323

Кабинет содержит 3 рабочих стола 2 книжных шкафа и выход в коридор. В кабинете присутствует офисная мебель, вентиляция, персональный компьютер и две отопительные батареи.

Длина кабинета составляет 6,5 м, ширина 4 м, общая площадь кабинета составляет 26 м2. В помещении присутствует одно окно длиной 2м, а высотой 3м, площадь окна составляет 6м2, высота помещения 3,5м.

По стандартам СП 2.2.3670-20 [18] регламентированный размер рабочего пространства офисного работника составляет не менее 4,5м2, умножая это число на 3 рабочих места минимальная площадь помещения должна быть не менее 13,5 м2, а фактическая площадь составляет 26 м2, что соответствует требуемым стандартам.

В помещении имеется искусственное освещение, представленное лампами, установленными по периметру потолка и на рабочих столах, а также естественное освещение, которое представляет собой окно 2х3м, площадью 6м2.

Влажность воздуха в помещении составляет 55%, а также при условии, что окно закрыто скорость воздуха не превышает 0,1 м/c, что соответствует нормам, прописанным в СП 2.2.3670-20 [18].

**5.2 Характеристика вредных и опасных факторов**

Человек во время любой трудовой деятельности может оказаться под влиянием вредных и опасных факторов, которые отрицательно влияют на его здоровье. При воздействии вредных факторов человек теряет работоспособность, а воздействие опасных факторов способно привести к резкому ухудшению здоровья.

Вредные и опасные факторы делятся на несколько видов: физические, химические, биологические, психофизиологические.

В соответствии с руководящими документами Р 2.2.2006-05 [19] и ГОСТ 12.0.003-2015[20] перечень и характеристика вредных и опасных факторов, которые воздействуют на работника приведены в таблице 5.1.

* + - * 1. Характеристика вредных и опасных факторов

| **Опасные и вредные факторы** | **Источники опасных и вредных факторов** | **Нормируемые параметры** | **Основные средства защиты** |
| --- | --- | --- | --- |
| **Вредные факторы** | | | |
| Повышенная или пониженная освещенность | Неисправные осветительные приборы, загряз-ненность окон | ен =1,275 %  Ен =300 лк СанПиН 1.2.3685-21 [21]  СанПиН 52.13330.2016 [22] | Заменить осве-тительные при-боры и очистить окна от загрязнений |
| Повышенный уровень интенсивности звука | Кулер, оргтехника, электромагнитный шум. Внешний шум L>80-120 дБ | L = 50 дБ СанПиН 1.2.3685-21 [21] | Монтаж звукоизоляции, замена обору-дования на менее шумное |
| Повышенная  или пониженная интенсивность электромагнитного излучения | Компьютер, оргтехника | В диапазоне 5Гц – 2кГц: E=25, В/м; В=250нТл, Н= 2,5 В/м  СанПиН 1.2.3685-21 [21] | Регламентиро-ванный перерыв, изоляционные экраны |

Продолжение таблицы 5.1 – Характеристика вредных и опасных факторов

|  |  |  |  |
| --- | --- | --- | --- |
| **Опасные и вредные факторы** | **Источники опасных и вредных факторов** | **Нормируемые параметры** | **Основные средства защиты** |
| **Вредные факторы** | | | |
| Отклонение  параметра  микроклимата  от нормируемых  значений | Отсутствие вентиляции, систем кондиционирования, наличие щелей в окнах, неисправ-ность отопления | Категории работ 1b Холодный период: 𝑡=21−23℃ 𝜑=60−40%  𝜗=0,1 м/с  𝑡 поверхностей = 20-24℃.  Тёплый период: 𝑡=22−24℃ 𝜑=60−40%  𝜗=0,1 м/с 𝑡 поверхностей = 21-25℃.  СанПиН 1.2.3685-21 [21] | Установка дополнитель-ных обогрева-телей и кондиционеров, вентиляторов |
| Повышенный показатель напряженности трудового процесса | Плотность сигналов (световых, звуковых) и сообщений в сред-нем за 1 час работы | допустимо 76 – 175  Р 2.2.2006 – 05 [19] | Использовать средства инди-видуальной за-щиты: специ-альные очки для работы за компьютером |
| Стереотипные рабочие движения (количество за смену) При локальной нагрузке (с участием мышц кистей и пальцев рук) | Оптимально до 20000 однотипных движений за смену  Р 2.2.2006 – 05 [19] | Смена позы, ре-гламентирован-ный перерыв |

Продолжение таблицы 5.1 – Характеристика вредных и опасных факторов

|  |  |  |  |
| --- | --- | --- | --- |
| **Опасные и вредные факторы** | **Источники опасных и вредных факторов** | **Нормируемые параметры** | **Основные средства защиты** |
| **Вредные факторы** | | | |
| Повышенный показатель напряженности трудового процесса | Рабочая поза | Свободная, удобная поза, возможность смены рабочего положения тела (сидя, стоя). Нахождение в позе стоя до 40 % времени смены  Р 2.2.2006 – 05 [19] | Смена позы и регламентиро-ванный перерыв |
| Сенсорные нагрузки: плотность сигналов (световых, звуковых) и сообщений в сред-нем за 1 час работы | допустимо 76 – 175 оптимально до 75  Р 2.2.2006 – 05 [19] | Использовать средства инди-видуальной за-щиты: специ-альные очки для работы за компьютером |
| Эмоциональные нагрузки: Степень ответственности за результат собственной деятельности. Значимость ошибки | Несет ответственность за выполнение отдельных элементов заданий. Влечет за собой дополнительные усилия в работе со стороны работника  Р 2.2.2006 – 05 [19] | Регламентиро-ванный перерыв |
| Режим работы: Фактическая продолжительность рабочего дня | Оптимально 6-7 часов  Р 2.2.2006 – 05 [19] |  |

Продолжение таблицы 5.1 – Характеристика вредных и опасных факторов

|  |  |  |  |
| --- | --- | --- | --- |
| **Опасные и вредные факторы** | **Источники опасных и вредных факторов** | **Нормируемые параметры** | **Основные средства защиты** |
| **Вредные факторы** | | | |
| Повышенный показатель напряженности трудового процесса | Режим работы: Наличие регламентированных перерывов и их продолжительность | Перерывы регла-ментированы, достаточной про-должительности: 7% и более рабочего времени Р 2.2.2006 – 05 [19] |  |
| **Опасные факторы** | | | |
| Опасность поражения элек-трическим током | Оголенные провода, неисправности электропроводки и оборудования | U=220B I=0,6-1,5мА (ток самоосвобождение)  ГОСТ 12.1.019-2017 [23] | Наличие изоляций токопроводящих поверхностей. Наличие защитного заземления или зануления |
| Опасность возникновения пожаров и взрывов | Горючие материалы, короткое замыкание, Несоблюдение техники безопасности | Категория по пожарной опасности: В1-В4 Классов горючих веществ и материалов: Класс А Класс Е  123-ФЗ [24] | Использование предупреди-тельной сигнализации, автоматическое пожаротуше-ние, условия эвакуации людей |

Проанализировав таблицу 5.1, можно значительно увеличить безопасность труда. Достаточно лишь придерживаться рекомендациям и не выходит за границы нормируемых значений.

**5.3 Расчет освещения в помещении Центра заселения ИРНИТУ**

Необходимо рассчитать освещенность аудитории Е323 методом светового потока. Расчет происходит по следующему алгоритму:

1. Тип помещения – Аудитория
2. Площадь помещения – 26 м2
3. Длина (A) – 6,5 м, ширина (B) – 4 м, высота – 3,5 м
4. Люминесцентная лампа ЛТБ20, диаметр 40 мм, длина 590 мм.
5. Нормируемое значение освещенности Е(лк) = 400 по СанПиН 2.2.1/2.1.1.1278-03 [25]
6. Расчет высоты подвеса светильника вычисляется по формуле
7. Определение расстояния между рядами светильников L, принимается в зависимости от λ. Для общественных зданий λ принимается по усредненному значению =1,5.
8. – расстояние от крайних светильников или рядов светильников до стены, м (принимается (0,3-0,5)L в зависимости от наличия вблизи стен рабочих мест)
9. Число рядов светильников определяется по выражению:
10. А число светильников в ряду из соотношения:
11. Полученные результаты округляются до ближайшего целого числа, после чего пересчитываются реальные расстояния:
12. Между рядами светильников:
13. Между центрами светильников в ряду:
14. Определение индекса помещения:
15. Принимаем приблизительные значения коэффициентов отражения от стен и потолка по Кнорринг Г.М. Справочная книга для проектирования электрического освещения [26].ρст = 70%, ρпот = 50%, ρпола = 30%
16. Коэффициент использования светового потока равен 49%.
17. Определение требуемого светового потока лампы:
18. Выбираем на один светильник – 6 светодиодных ламп мощностью 25 Вт со световым потоком 2100 лм.
19. Фном отличается от Ф на 4 %, что находится в допустимых пределах  
    (–10%; +20%).

Расчетный и световой поток отличаются на 4%, из этого можно сделать вывод, что искусственное освещение находится на достаточном уровне и количество ламп изменять не стоит.

## Электробезопасность

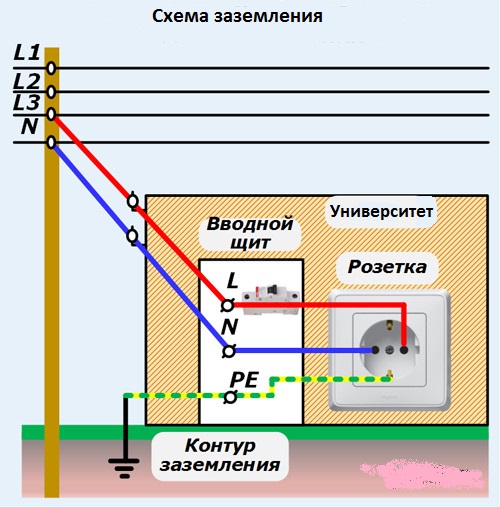
Под электробезопасностью понимается система мероприятий, направленных на предотвращение вредного или опасного поражения работников электрическим током, электрической дугой и т.п.

В помещении Е323 отсутствует повышенная опасность, так как кабинет имеет класс опасности 1, сеть имеет стандартное напряжение в 220В, присутствует полная изоляция проводки, отсутствуют оголенные провода или поврежденные розетки, все используемые удлинители имеют встроенный предохранитель. Обесточивание сети происходит при какой-либо серьезной аварии.

При неисправности какого-либо оборудования необходимо обесточить устройство и дождаться специалиста, иначе неизбежно поражение электрическим током.

Для того чтобы защитить работника от поражения электрическим током устанавливают заземление. Защитное заземление – это особое, преднамеренное соединение с землей металлических нетоковедущих частей для защиты человека от поражения электричеством.

Схема типового заземления, установленного в университете согласно ГОСТ 12.1.019-2017 [23] представлена на рисунке 5.2



* + - 1. Рисунок 5.2 – Схема заземления

Рабочее помещение соответствует стандартам, прописанным в ГОСТ 12.1.019-2017 [23], что позволяет работникам продолжать трудовую деятельность без угрозы для здоровья.

## Пожарная безопасность

Пожарная безопасность регламентируется Закон РФ от 22 июля 2008 г. N 123-ФЗ [24]. Пожар – неконтролируемое горение, которое опасно для всех работников в помещении. Причинами пожара могут послужить: неисправность проводки, неисправность электронагревательных устройств, халатное отношение работников к пожарной безопасности, умышленный поджог.

В рассматриваемом помещении для защиты от пожара установлены датчики дыма, представленные на рисунке 5.3. Принцип работы датчиков дыма заключается в том, что посылаемый луч в приборе при наличии дыма рассеивается.



* + - 1. Рисунок 5.3 – Датчик дыма «ИПД-3.2»

В коридорах университета расположены пожарные шиты, предусмотренные на случай возникновения возгорания, в корпусе на каждом этаже таких щитов по три, в каждом находится огнетушитель и шланг, рисунок 5.4 иллюстрирует один из установленных щитов.



* + - 1. Рисунок 5.4 – Пожарный щиток

На рисунке 5.5(а) изображен огнетушитель ОУ-2, а на рисунке 5.5(б) изображен пожарный рукав, которые устанавливаются в выше описанные пожарные щитки в коридорах.

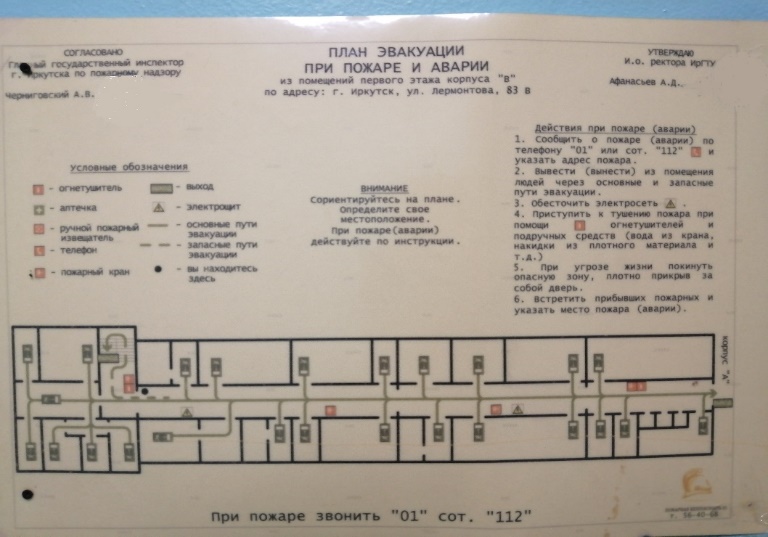


а) б)

* + - 1. Рисунок 5.5 – Огнетушитель ОУ-2 и пожарный рукав

Внутри огнетушителя ОУ – 2 находится углекислота и углекислый газ в жидкой форме под давлением от 5,7 до 15 Мпа, вырываясь наружу мгновенно сбивает пламя, лишая его кислорода, вследствие этого огонь не может далее гореть. Пожарный рукав имеет стандартную длину 20м.

В каждом здании должна присутствовать схема эвакуации, описывающая принцип эвакуации при возникновении чрезвычайных ситуаций. На рисунке 5.6 представлена схема эвакуации, которая установлена в коридорах университета в двух местах.



* + - 1. Рисунок 5.6 – Схема эвакуации

Здание соответствует всем необходимым стандартам для проведения учебных занятий.

## Заключение по разделу

Из проведенного анализа в рамках данного раздела, можно сделать вывод о соответствии рабочего пространства кабинета Е323 всем необходимым требованиям для проведения трудовой деятельности. Даны рекомендации по снижению напряженности трудового процесса, а именно снизить наблюдение за экраном видеотерминала (п 2.6), планировать содержание работы (п 1.1) и характер выполняемой работы (п 1.4) для менее напряженного их выполнения, а также влиянию вредных и опасных факторов.

# Заключение

В ходе выполнения дипломной работы был проведен анализ проблем формирования списка студентов на заселение в предыдущие года, выделены основные цели и задачи.

Разрабатываемый программный продукт адаптирован под специфику ВУЗа, благодаря чему гарантирован ожидаемы результат, созданный согласно требованиям заказчика.

В результате дипломной работы были решены следующие задачи:

* Изучен процесс формирования списков студентов на заселение
* Выполнен анализ предметной области
* Сформированы требования к системе
* Выполнено проектирование системы
* Реализован прототип системы, состоящий из двух модулей: Web и WinForms.

Первый модуль служит для подачи заявки на заселение в онлайн формате, а также для просмотра статуса поданной заявки.

Второй – для формирования списка студентов на заселение. Модуль имеет ручной способ формирования списка и три способа автоматизированного формирования.

Также, автору удалось рассчитать временную выгоду после внедрения системы, которая составит около 9 часов. Следовательно, процесс ускорен, цель достигнута.

# Список используемых источников

1. ИрНИТУ. [Электронный ресурс] URL: <https://www.istu.edu/>. (дата обращения 19.10.20)
2. Приказ №744-П «О порядке заселения студентов в общежития студгородка на 2020/21 учебный год» / ИрНИТУ – г. Иркутск, 2020. – 11 стр.
3. Положение о студенческом городке ФГБОУ ВО «ИРНИТУ» от 2021 г. / ИрНИТУ – г. Иркутск, 2021 – 15 стр.
4. Университет ИТМО. [Электронный ресурс] URL: <https://www.itmo.ru/>. (дата обращения 20.11.20)
5. МАИ. [Электронный ресурс] URL: <https://www.mai.ru/>. (дата обращения 20.11.20)
6. Иркутский государственный университет. [Электронный ресурс] URL: <https://www.isu.ru/>. (дата обращения 20.11.20)
7. Ramus. URL: <http://ramussoftware.com/>. (Дата обращения: 21.03.2021)
8. draw.io – Diagrams for Confluence and Jira. URL: <https://drawio-app.com/>. (Дата обращения: 21.03.2021)
9. Инструменты проектирования диаграмм базы данных. [Электронный ресурс] // CoderLessons. URL: <https://coderlessons.com/tutorials/bazy-dannykh/osnovy-subd/22-instrumenty-proektirovaniia-diagramm-bazy-dannykh>. (Дата обращения: 21.03.2021).
10. SQL Server Data Tools для Visual Studio. [Электронный ресурс] URL: <https://visualstudio.microsoft.com/ru/vs/features/ssdt/>. (Дата обращения: 21.04.2021)
11. .NET Framework. [Электронный ресурс] // Википедия. URL: <https://ru.wikipedia.org/wiki/.NET_Framework>. (Дата обращения: 21.04.2021)
12. Visual Studio. [Электронный ресурс] URL: <https://visualstudio.microsoft.com/ru/> (Дата обращения: 21.04.2021)
13. Документация по C# [Электронный ресурс]//Microsoft URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/> (Дата обращения: 21.05.2021)
14. Создание приложения Windows Forms на C# в Visual Studio [Электронный ресурс]//Microsoft URL: <https://docs.microsoft.com/ru-ru/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2019> (Дата обращения: 21.05.2021)
15. Основы LINQ [Электронный ресурс] URL: <https://metanit.com/sharp/tutorial/15.1.php>. (Дата обращения: 21.04.2021)
16. EPPlus Software [Электронный ресурс] URL: <https://epplussoftware.com/> (Дата обращения: 21.04.2021)
17. СанПиН II-7-81. Строительство в сейсмических районах. [Электронный ресурс] URL: <https://docs.cntd.ru/document/1200000291>. (дата обращения 25.05.21)
18. СанПиН 2.2.3670-20. Санитарно-эпидемиологические требования к условиям труда. [Электронный ресурс] URL: <https://docs.cntd.ru/document/573230583>. (дата обращения 25.05.21)
19. Р 2.2.2006-05. Руководство, по гигиенической оценке, факторов рабочей среды и трудового процесса. Критерии и классификация условий труда. [Электронный ресурс] URL: <https://docs.cntd.ru/document/1200040973>. (дата обращения 26.05.21)
20. ГОСТ 12.0.003-2015. Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классификация. [Электронный ресурс] URL: <https://docs.cntd.ru/document/1200136071>. (дата обращения 25.05.21)
21. СанПиН 1.2.3685-21. Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания. [Электронный ресурс] URL: <https://docs.cntd.ru/document/573500115>. (дата обращения 25.05.21)
22. СанПиН 52.13330.2016. Естественное и искусственное освещение. [Электронный ресурс] URL: <https://docs.cntd.ru/document/456054197>. (дата обращения 26.05.21)
23. ГОСТ 12.1.019-2017. Система стандартов безопасности труда. Электробезопасность. Общие требования и номенклатура видов защиты. [Электронный ресурс] URL: <https://docs.cntd.ru/document/1200161238>. (дата обращения 25.05.21)
24. Федеральный закон "Технический регламент о требованиях пожарной безопасности" от 22.07.2008 N 123-ФЗ. [Электронный ресурс] URL: <https://docs.cntd.ru/document/902111644>. (дата обращения 26.05.21)
25. СанПиН 2.2.1/2.1.1.1278-03. Естественное и искусственное освещение.
26. Кнорринг Г.М. Справочная книга для проектирования электрического освещения / Кнорринг Г.М., 1976. – 384 c.
27. Положение о системе рейтинга обучающихся, проживающих в студенческих общежитиях ФГБОУ ВО «ИРНИТУ» от 2020 г. / ИрНИТУ – г. Иркутск, 2020 – 10 стр.