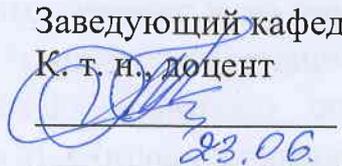


МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК
Кафедра программного обеспечения

РЕКОМЕНДОВАНО К ЗАЩИТЕ В ГЭК
Заведующий кафедрой

К. т. н., доцент



М. С. Воробьева

23.06. 2023 г.

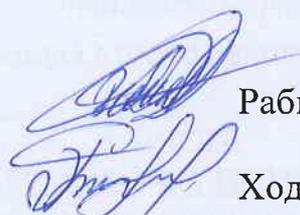
ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
магистерская диссертация

**РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ПЕРВИЧНОЙ
ОБРАБОТКИ ИНФОРМАЦИИ О ЗАРУБЕЖНЫХ СТУДЕНТАХ ТЮМГУ**

02.04.03 Математическое обеспечение и администрирование информационных систем

Магистерская программа «Разработка технологий Интернета вещей и больших данных»

Выполнили работу
(групповой проект)
студенты 2 курса
очной формы обучения



Рабиев Саймухаммад Рахматович

Ходжаев Худойназар Фазлидинович

Научный руководитель
доцент, к.ф.-м.н.



Ступников Андрей Анатольевич

Рецензент
Доцент кафедры
информационной безопасности



Паюсова Татьяна Игоревна

ОГЛАВЛЕНИЕ

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ.....	3
ВВЕДЕНИЕ.....	4
ГЛАВА 1. ПРЕДОБРАБОТКА ДАННЫХ И СОЗДАНИЕ РЕЛЯЦИОННОГО ХРАНИЛИЩА ДЛЯ ДАННЫХ.....	6
1.1. АНАЛИЗ И ВЫРАБОТКА ТРЕБОВАНИЙ К БАЗЕ ДАННЫХ	6
1.2. ВЫБОР ИНСТРУМЕНТОВ СОЗДАНИЯ БАЗЫ ДАННЫХ.....	7
1.3. СОЗДАНИЕ СТРУКТУРЫ БАЗЫ ДАННЫХ.....	10
1.4. НАПОЛНЕНИЕ БАЗЫ ДАННЫХ	12
1.5 ПРОБЛЕМЫ, СВЯЗАННЫЕ С ОСОБЕННОСТЯМИ ОРГАНИЗАЦИИ ИСХОДНЫХ ДАННЫХ.....	25
ГЛАВА 2. РАЗРАБОТКА ПРИЛОЖЕНИЯ.....	27
2.1 ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИИ	27
2.2 АРХИТЕКТУРА ПРИЛОЖЕНИЯ.....	28
2.3. ЭЛЕМЕНТЫ ПРОГРАММЫ.....	30
2.5 ИНТЕРФЕЙС ПОЛЬЗОВАТЕЛЯ	38
2.6. ВЫВОД ИНФОРМАЦИИ В ТЕКСТОВОМ И ГРАФИЧЕСКОМ ВАРИАНТАХ	41
ГЛАВА 3. ОРГАНИЗАЦИЯ ПРОВЕДЕНИЯ АНКЕТИРОВАНИЯ ИНОСТРАННЫХ СТУДЕНТОВ ТЮМГУ	47
ЗАКЛЮЧЕНИЕ	61
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	63
ПРИЛОЖЕНИЕ. ТЕКСТ КОДА КЛАССА GRAPHWINDOW (QMAINWINDOW), ПРЕДСТАВЛЯЮЩЕГО ОКНО ПРОГРАММЫ ДЛЯ ОТОБРАЖЕНИЯ ГРАФИКОВ.....	64

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

ВКР – Выпускная квалификационная работа

API (Application Programming Interface) – Описание способов которыми одна компьютерная программа может взаимодействовать с другой программой.

CSS (Cascading Style Sheets) – Формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

HTML (HyperText Markup Language) – Стандартизированный язык гипертекстовой разметки документов.

MVC (Model-View-Controller) – Схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер.

REST (Representational State Transfer) – Архитектурный стиль взаимодействия компонентов приложения в сети.

WEB – Интернет пространство.

БД – База данных.

РФ – Российская Федерация.

СУБД – Система управления базами данных.

ВВЕДЕНИЕ

Результатом политики глобализации мировых процессов в области образования является интернационализация высшего образования. Установлена прямая связь между интернационализацией и качеством образования, показатели интернационализации включены в современные системы оценки качества и конкурентоспособности вузов (доля иностранных студентов, доля иностранных научно-педагогических работников и т.д.).

При существенном росте числа иностранных студентов возрастает и количество возникающих в процессе их адаптации проблемных ситуаций [1]. Наиболее существенным в этом плане являются ситуации, связанные с возникновением и закрытием задолженностей иностранных студентов. При этом важно оценить, не связано ли наличие подобных задолженностей с другими факторами, являющимися как общими для всех студентов, так и специфическими для студентов-иностранцев.

В распоряжении сотрудников учебной части вуза имеется разнообразная информация об иностранных студентах, обучающихся в нём.

При возникновении необходимости получить агрегирующую или выборочную информацию о студентах из зарубежных стран требуется вручную обработка данных, хранящихся в Excel – таблицах. На основе данной информации сотрудники уже получают разнообразные отчёты о конкретных студентах, о студентах, имеющих задолженности, а также о студентах из разных стран. Они хранят и анализируют эти данные с помощью таблиц Excel. Таблицы, входящие в составляемые отчёты, можно создать в Excel, но это не очень удобно. Следовательно, возникает задача создания информационной системы, которая бы аккумулировала данные из таблиц Excel и предоставляла пользователю возможность получать различные результаты, подавать запросы, сохранять и визуализировать результаты этих запросов.

Цель: Создать приложение, позволяющие специалистом университета получать информацию о зарубежных студентах, обучающихся в ТюмГУ в табличном и графическом форматах.

Задачи:

- Организовать загрузку данных из Excel - таблицы
- Провести предварительную обработку учесть;
 - ✓ - наличие пропусков;
 - ✓ - отсутствие нормализации исходных данных;
 - ✓ - выполнить выделение экземпляров сущности.
- Выбрать виды показателей для студентов и реализовать их вычисление
- Организовать возможности выбора правил фильтрации для отбора студента
- Организовать представления данных в текстовом и графически форматах
- Реализовать возможность обработки результатов анкетирования студентов с привязкой к их атрибутам.

ГЛАВА 1. ПРЕДОБРАБОТКА ДАННЫХ И СОЗДАНИЕ РЕЛЯЦИОННОГО ХРАНИЛИЩА ДЛЯ ДАННЫХ

Очевидно, что программная реализация может быть выполнена с использованием различных средств и методов разработки приложений.

Наиболее интересным с точки зрения богатого инструментария поддержки сервисных функций и средств визуализации нам представляется использование языка программирования python. Возможности данного языка позволяют наглядно и эффективно организовать как работу с данными (в том числе, представленными Excel-файлами), так и реализовать текстовую и графическую реализацию результата.

1.1. АНАЛИЗ И ВЫРАБОТКА ТРЕБОВАНИЙ К БАЗЕ ДАННЫХ

Информация об иностранных студентах и их задолженностях хранится в отдельном файле EXCEL, состоящим из двух таблиц. В первой таблице хранятся сведения об иностранных студентах ТюмГУ из стран СНГ. Во второй таблице сведения об иностранных студентов, имеющих задолженности, и сведения об этих задолженностях. В планируемой базе данных предполагается хранение данных из этих двух таблиц.

На рисунках 1.1 и 1.2 представлена таблица из EXCEL-файла со списком всех иностранных студентов стран СНГ с указанием гражданства, курса обучения, направления и уровня подготовки. Размер данной таблицы: 1170 строк. Все поля имеют строковый тип.

№п/п	ФИО	Гражданство	Курс	Специальность/направление	Уровень подготовки
1	Абдихоидов Бехруз Бахтиёр угли	УЗБЕКИСТАН	Второй	38.04.02. Менеджмент	Магистратура
2	Абдимиталипов Мустафа Тыныбесович	КИРГИЗИЯ	Первый	10.05.03. Информационная безопасность автоматизированных систем	Специалитет
4	Абдиреуков Самат Талгатулы	КАЗАХСТАН	Второй	02.03.03. Математическое обеспечение и администрирование информационных систем	Бакалавриат
5	130. Бильдин Сергей Михайлович	КАЗАХСТАН	Второй	02.03.03. Математическое обеспечение и администрирование информационных систем	Бакалавриат
6	5. Абдуалимова Билолиддин Насрулдин	ТАДЖИКИСТАН	Первый	38.05.01. Экономическая безопасность	Специалитет
7	6. Абдуганиев Мирабдоржон Бехзод угли	УЗБЕКИСТАН	Третий	40.03.01. Юриспруденция	Бакалавриат
8	7. Абдуганиев Илхомжон Холмурод угли	УЗБЕКИСТАН	Первый	09.03.02. Информационные системы и технологии	Бакалавриат
9	8. Абдуллаев Жахонгир Жалол угли	УЗБЕКИСТАН	Первый	38.03.01. Экономика	Бакалавриат
10	9. угли	УЗБЕКИСТАН	Четвертый	04.03.01. Химия	Бакалавриат
11	10. Абдуллин Амаль Фуадович	УЗБЕКИСТАН	Четвертый	05.03.06. Экология и природопользование	Прикладной бакалавриат

Рис.1.1(а). Таблица со списком всех иностранных студентов (начало).

A	B	C	D	E	F
1169	Якимовский Руслан Шухратович	УЗБЕКИСТАН	Первый	40.03.01. Юриспруденция	Бакалавриат
1170	Яковлев Артур Юрьевич	КАЗАХСТАН	Второй	15.03.06. Мехатроника и робототехника	Бакалавриат
1171	Яковлева Виктория Евгеньевна	КАЗАХСТАН	Первый	05.03.06. Экология и природопользование	Бакалавриат
1172	Яковлева Екатерина Евгеньевна	КАЗАХСТАН	Третий	35.03.10. Ландшафтная архитектура	Бакалавриат
1173	Яковлева София Евгеньевна	КАЗАХСТАН	Второй	45.03.02. Лингвистика	Бакалавриат
1174	Самардинович	ТАДЖИКИСТАН	Первый	40.03.01. Юриспруденция	Бакалавриат
1175	Якубов Сухробжон Рауфжонович	ТАДЖИКИСТАН	Первый	04.04.01. Химия	Магистратура
1176	Ямалтдинова Зарина Камильевна	КАЗАХСТАН	Четвертый	05.03.06. Экология и природопользование	Прикладной бакалавриат
1177	Янгibaев Абдукодир Азимович	УЗБЕКИСТАН	Третий	04.03.01. Химия	Бакалавриат
1178	Яхьяева Наталья Александровна	УЗБЕКИСТАН	Первый	04.04.01. Химия	Магистратура
1179	Яшонкина Карина Олеговна	КАЗАХСТАН	Второй	10.03.01. Информационная безопасность	Бакалавриат
1180	Ященко Анастасия Александровна	КАЗАХСТАН	Второй	37.03.01. Психология	Бакалавриат
1181					
1182					

Рис.1.1(б). Таблица со списком всех иностранных студентов (конец).

На рисунке 1.2 изображена таблица со списком иностранных студентов стран СНГ, имеющих задолженности с теми же полями, что и в таблице 1, а также со списком задолженностей, вида контроля, с процентами пропусков и посещаемости занятий и примечанием (сдал/не сдал/-). В текущем варианте таблица содержит информацию о 135 студентах задолжниках.

A	B	C	D	E	F	G	H	I	J	K	L
№п/п	ФИО студентов по отчету должников	Гражданство	Курс	Специальность/направление	Уровень подготовки	Вид контроля	Задолженность	Дата/оценка	Процент занятий в 2022-2023 учебном году	Процент посещенных в 2022-2023 учебном году	Примечания
1	Абдихоидов Бехруз Бахтиёр угли	УЗБЕКИСТАН	Второй	38.04.02. Менеджмент	Магистратура	экзамен	1. Коммуникационная политика в маркетинге 2. Международный бизнес: стратегии развития 3. Традиционные и инновационные концепции менеджмента	13.10.2022 - неявка 05.10.2022 - неявка 04.10.2022 - неявка			
2	Абдуллаев Сардорбек Абборжон угли	УЗБЕКИСТАН	Третий	40.03.01. Юриспруденция	Бакалавриат	зачет	1. Инновационные компании - "единогори": от идеи до выхода на глобальный рынок 2. Событийный маркетинг	15.10.2022 - неявка 06.10.2022 - неявка			
3	Абдуллоев Шахзод Абдумаликович	ТАДЖИКИСТАН	Четвертый	04.03.01. Химия	Бакалавриат	отчет по практике	1. Научно-исследовательская работа 2. Практика по профилю профессиональной деятельности	19.10.2022 - неявка 20.10.2022 - неявка	38%	8%	
4	Абдулов Некруз Нуралевич	ТАДЖИКИСТАН	Второй	38.04.01. Экономика	Магистратура	экзамен	Органическая химия	14.10 - неуд 1. 14.10 - неуд; 2. 07.10 - неуд	77%	17%	
5	Абдулов Некруз Нуралевич	ТАДЖИКИСТАН	Второй	38.04.01. Экономика	Магистратура	Отчет по практике (зачет)	Научно-исследовательская работа	06.10 - не зачтено	94%	0%	
6	Абдулов Некруз Нуралевич	ТАДЖИКИСТАН	Второй	38.04.01. Экономика	Магистратура	Отчет по практике (зачет)	Ознакомительная практика	19.10.2022 - зачтено	26%	50%	Сдал
7	Абдулов Некруз Нуралевич	ТАДЖИКИСТАН	Второй	38.04.01. Экономика	Магистратура	Отчет по практике (зачет)	Теория вероятностей и математическая статистика для	12.10.2022 - неудовлетворитель			

Рис.1.2. Таблица со списком иностранных студентов стран СНГ, имеющих задолженности

1.2. ВЫБОР ИНСТРУМЕНТОВ СОЗДАНИЯ БАЗЫ ДАННЫХ

Для создания базы данных был выбран облачный блокнот для программирования на Python Google Colab. Блокнот Colab — это бесплатная интерактивная облачная среда для работы с кодом на языке Python от Google в браузере. Она позволяет одновременно с коллегами работать с данными.

Преимущество Colab в том, что он предоставляет возможность работать с кодом на языке Python через Jupyter Notebook, не устанавливая на свой компьютер дополнительных программ. В Google Colab можно применять различные библиотеки на Python, загружать и запускать файлы, анализировать данные и получать результаты в браузере.

Для хранения данных была выбрана компактная и нетребовательная к ресурсам СУБД SQLite. Данная СУБД, представленная модулем `sqlite3`, входит в стандартную библиотеку языка Python. То для работы с SQLite из Python-кода не нужно устанавливать некое клиент-серверное ПО, не нужно поддерживать работу какого-то сервиса, отвечающего за работу с СУБД. Достаточно лишь импортировать модуль `sqlite3` и приступить к его использованию в программе, получив в своё распоряжение систему управления реляционными базами данных.

При разработке приложения для хранения данных мы приняли решение использовать базу данных SQLite. В данном разделе мы обосновываем выбор SQLite и объясняем его преимущества в контексте нашего проекта.

Основные преимущества SQLite:

1. Простота использования: SQLite предоставляет простой и интуитивно понятный интерфейс для работы с базой данных. Его команды и синтаксис легко осваиваются даже для разработчиков с небольшим опытом работы с базами данных.
2. Легковесность: SQLite является легкой и компактной базой данных, которая не требует значительных системных ресурсов для своей работы. Она может быть интегрирована непосредственно в приложение, что позволяет избежать необходимости установки и настройки отдельного сервера базы данных.
3. Встроенное хранение данных: SQLite позволяет хранить базу данных в виде единого файла на диске. Это упрощает процесс развертывания и переноса приложения между различными средами, так как нет необходимости в установке и настройке сервера базы данных.

4. Поддержка стандартного SQL: SQLite поддерживает стандартный язык структурированных запросов SQL, что делает его совместимым с другими системами управления базами данных. Разработчики могут использовать знания и навыки SQL для работы с SQLite.
5. Надежность: SQLite обеспечивает надежное хранение данных и обработку транзакций. Он предоставляет механизмы для поддержки целостности данных и восстановления после сбоев.

В контексте нашего проекта выбор SQLite был обоснован следующими факторами:

1. Простота использования: SQLite предоставляет удобный интерфейс и простой синтаксис, что упрощает разработку и поддержку нашего приложения.
2. Легковесность: Наше приложение не требует больших объемов данных или высоких нагрузок. SQLite обеспечивает достаточную производительность для наших потребностей без необходимости в установке и настройке сложного сервера базы данных.
3. Встроенное хранение данных: SQLite позволяет нам хранить базу данных в виде файла, что облегчает процесс развертывания и переноса приложения на разные серверы или хостинг-платформы.
4. Кросс-платформенность: SQLite поддерживается на различных операционных системах, включая Windows, macOS и Linux. Это позволяет нам создавать приложение, которое будет работать на разных платформах без дополнительной настройки.

Выбор SQLite в качестве базы данных для нашего приложения обеспечивает нам простоту использования, легковесность и возможность встроенного хранения данных. Это помогает упростить разработку, улучшить производительность и обеспечить надежность хранения данных в нашем приложении.

1.3. СОЗДАНИЕ СТРУКТУРЫ БАЗЫ ДАННЫХ

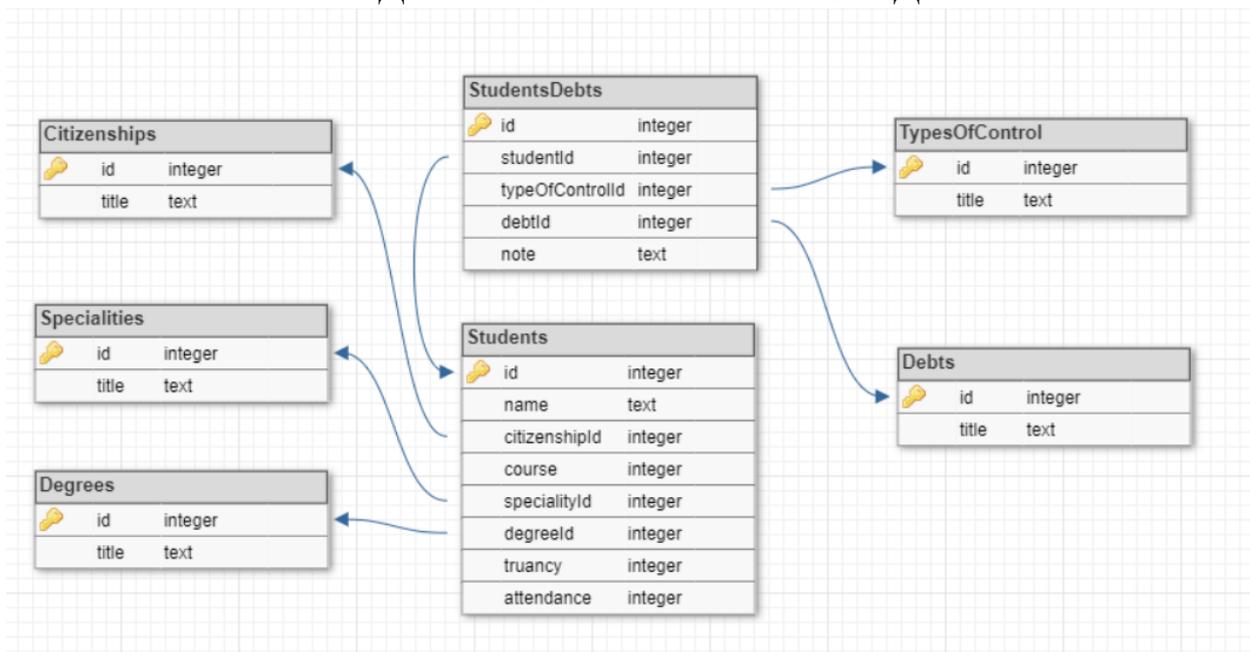


Рис.1.3. Схема базы данных

На рис. 1.3 изображена итоговая схема базы данных. Она содержит 2 таблицы и 5 справочников.

- Students – хранятся сведения об иностранных студентах стран СНГ (таблица 1.1). В таблице хранится основная информация о каждом их иностранных студентов.

Таблица 1.1

Описание таблицы Students из базы данных

Название поля	Тип поля	Описание поля
id	integer	Поле идентификатора
name	text	ФИО студента
citizenshipId	integer	Идентификатор из справочника с гражданствами
course	integer	Курс обучения
specialityId	integer	Идентификатор из справочника со специальностями
degreeId	integer	Идентификатор из справочника с уровнями подготовки

Продолжение таблицы 1.1

Название поля	Тип поля	Описание поля
truancy	text	Хранит процент пропусков занятий
attendance	text	Хранит процент посещаемости занятий

- StudentsDebts – хранятся сведения о студентах с задолженностями (таблица 1.2)

Таблица 1.2

Описание таблицы StudentsDebts из базы данных

Название поля	Тип поля	Описание поля
id	integer	Поле идентификатора
studentId	integer	Ссылка на студента из таблицы Students
typeOfControlId	integer	Идентификатор из справочника с видами контроля
debtId	integer	Идентификатор из справочника с задолженностями
note	text	Примечание (сдал/не сдал)

Для всех справочников Citizenships, Specialities, Deegres, TypesOfControl, Debts структура таблицы одинаковая. Она приведена ниже в таблице 1.3

- Citizenships – справочник с гражданствами
- Specialities – справочник со специальностями
- Degrees – справочник с уровнями подготовки
- TypesOfControl – справочник с видами контроля
- Debts – справочник с дисциплинами

Описание таблицы для всех справочников

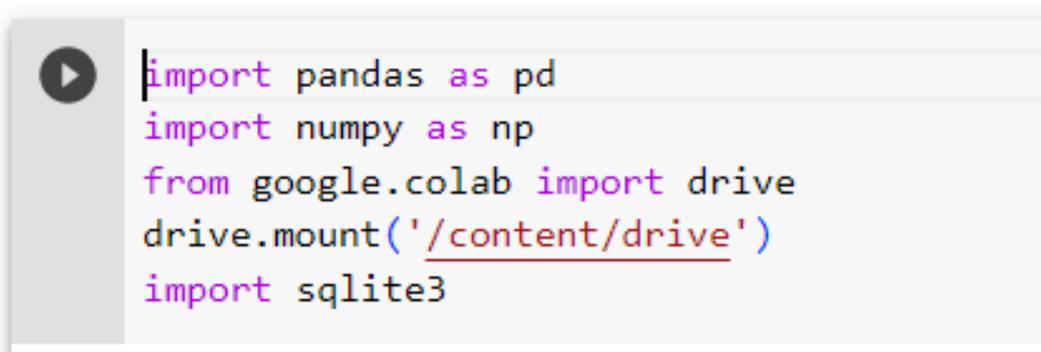
Название поля	Тип поля	Описание поля
id	integer	Поле идентификатора
title	integer	Наименование

1.4. НАПОЛНЕНИЕ БАЗЫ ДАННЫХ

Для создания базы данных и наполнения ее данными и связями, был написан программный код на языке Python в облачном блокноте Google Colab.

За работу с SQLite в Python отвечает стандартная библиотека sqlite3. Работа с любыми данными требует анализа и подготовки: необходимо удалить или заполнить пропуски, отфильтровать, отсортировать или каким-то образом изменить данные. Pandas в Python позволяет быстро выполнить все эти действия, а в большинстве случаев ещё и автоматизировать их. Поэтому для работы с данными используется библиотека pandas [5].

Python, запускаемый в среде Google Colab позволяет работать и с условно «локальной» базой данных, которая будет храниться на облачном диске Google Drive. Для подключения и разрешения доступа Google Colab к диску Google Drive можно использовать код из рис.1.4.



```

import pandas as pd
import numpy as np
from google.colab import drive
drive.mount('/content/drive')
import sqlite3

```

Рис.1.4. Код для подключения и разрешения доступа Google Colab

Создаем на диске Google каталог databases. Указываем путь к будущей базе данных. Подключаемся к базе mydatabase.db. Так как этого файла нет в каталоге, то он будет создан (рис. 1.5).

```
[ ] path = "./drive/My Drive/databases/mydatabase.db"
    conn = sqlite3.connect(path)
```

Рис.1.5. Код для создания каталог

Pandas позволяет импортировать данные разными способами. Например, прочесть их из словаря, списка или кортежа. Самый популярный способ — это работа с файлами .csv, которые часто применяются в анализе данных, именно это способ и будем использовать для считывания имеющихся данных.

Импортируем датасет с информацией об иностранных студентах, обучающихся в ТюмГУ и выводим на экран, получившийся датафрейм. (рис.1.6)

```
[ ] df = pd.read_csv('/content/drive/MyDrive/data/students_list.csv', sep=";")
df
```

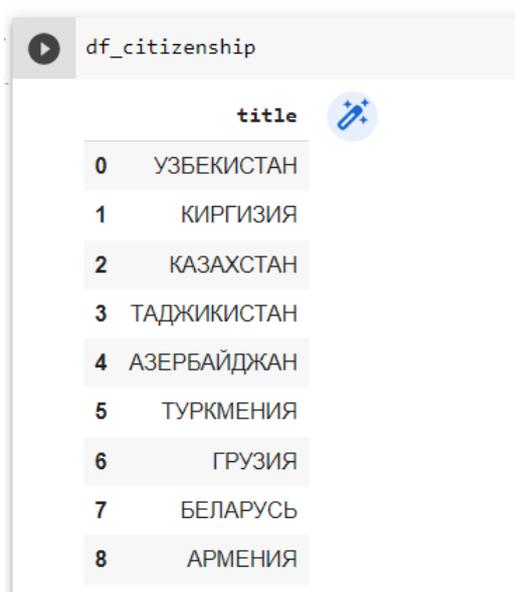
	Ип/п	ФИО	Гражданство	Курс	Специальность/направление	Уровень подготовки
0	1.	Абдивохилов Бехруз Бахтиёр угли	УЗБЕКИСТАН	Второй	38.04.02. Менеджмент	Магистратура
1	2.	Абдимиталипов Мустафа Тыныбекович	КИРГИЗИЯ	Первый	10.05.03. Информационная безопасность автомати...	Специалитет
2	4.	Абдиреуков Самат Талгатұлы	КАЗАХСТАН	Второй	02.03.03. Математическое обеспечение и админис...	Бакалавриат
3	130.	Бильдин Сергей Михайлович	КАЗАХСТАН	Второй	02.03.03. Математическое обеспечение и админис...	Бакалавриат
4	5.	Абдуалимзода Билолиддин Насрулдин	ТАДЖИКИСТАН	Первый	38.05.01. Экономическая безопасность	Специалитет
...
1174	1 175.	Ямалтдинова Зарина Камилъевна	КАЗАХСТАН	Четвертый	05.03.06. Экология и природопользование	Прикладной бакалавриат
1175	1 176.	Янгибаев Абдукодир Азимович	УЗБЕКИСТАН	Третий	04.03.01. Химия	Бакалавриат
1176	1 177.	Яхьяева Наталья Александровна	УЗБЕКИСТАН	Первый	04.04.01. Химия	Магистратура
1177	1 178.	Яшонкина Карина Олеговна	КАЗАХСТАН	Второй	10.03.01. Информационная безопасность	Бакалавриат
1178	1 179.	Ященко Анастасия Александровна	КАЗАХСТАН	Второй	37.03.01. Психология	Бакалавриат

Рис. 1.6. Датасет с информацией об иностранных студентах

Далее из столбцов Гражданство, Специальность/направления, Уровень подготовки сформируем таблицы из уникальных значений для наполнения справочников (рис. 1.7).

```
temp_lst=df['Гражданство'].unique().tolist()
df_citizenship=pd.DataFrame(temp_lst,columns=["title"])
temp_lst=df['Специальность/направление'].unique().tolist()
df_speciality=pd.DataFrame(temp_lst,columns=["title"])
temp_lst=df['Уровень подготовки'].unique().tolist()
df_degree=pd.DataFrame(temp_lst,columns=["title"])
```

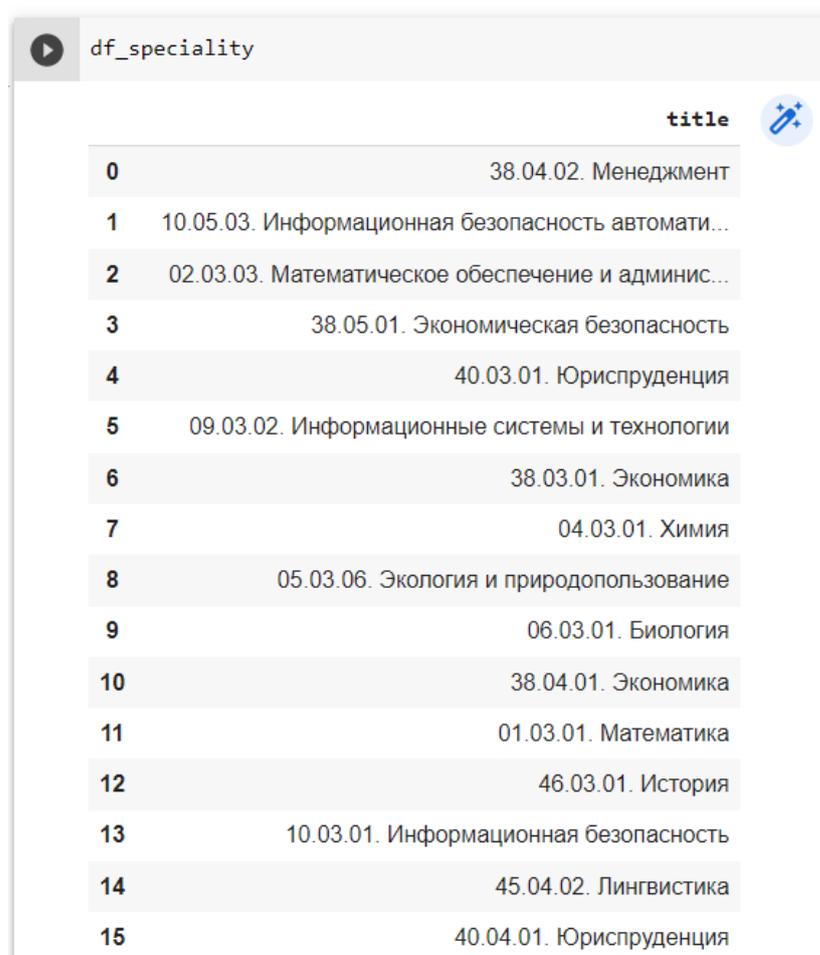
Рис. 1.7. Код из уникальных значений для наполнения справочников



df_citizenship

	title
0	УЗБЕКИСТАН
1	КИРГИЗИЯ
2	КАЗАХСТАН
3	ТАДЖИКИСТАН
4	АЗЕРБАЙДЖАН
5	ТУРКМЕНИЯ
6	ГРУЗИЯ
7	БЕЛАРУСЬ
8	АРМЕНИЯ

Рис.1.8. Получившиеся датафрейм (гражданство)



df_speciality

	title
0	38.04.02. Менеджмент
1	10.05.03. Информационная безопасность автомати...
2	02.03.03. Математическое обеспечение и админис...
3	38.05.01. Экономическая безопасность
4	40.03.01. Юриспруденция
5	09.03.02. Информационные системы и технологии
6	38.03.01. Экономика
7	04.03.01. Химия
8	05.03.06. Экология и природопользование
9	06.03.01. Биология
10	38.04.01. Экономика
11	01.03.01. Математика
12	46.03.01. История
13	10.03.01. Информационная безопасность
14	45.04.02. Лингвистика
15	40.04.01. Юриспруденция

Рис.1.9. Получившиеся датафрейм (специальность)

df_degree

	title
0	Магистратура
1	Специалитет
2	Бакалавриат
3	Прикладной бакалавриат
4	Академический бакалавриат
5	Прикладная магистратура
6	Академическая магистратура

Рис. 1.10. Получившиеся датафрейм (уровень подготовки)

Теперь данные в столбцах, которые содержат значения из справочников, нужно заменить на идентификаторы из соответствующего справочника. Для замены используем метод `replace()`, а также создаем словарь из справочников для замены по ключу. Получившийся датафрейм изображен на рис. 1.11.

```
df_new = df.replace({"Гражданство": {v:k for k, v in df_citizenship['title'].to_dict().items()}})
df_new = df_new.replace({"Специальность/направление": {v:k for k, v in df_speciality['title'].to_dict().items()}})
df_new = df_new.replace({"Уровень подготовки": {v:k for k, v in df_degree['title'].to_dict().items()}})
df_new
```

	№п/п	ФИО	Гражданство	Курс	Специальность/направление	Уровень подготовки
0	1.	Абдивоихидов Бехруз Бахтиёр угли	0	Второй	0	0
1	2.	Абдимиталипов Мустафа Тыныбекович	1	Первый	1	1
2	4.	Абдиреуков Самат Талгатулы	2	Второй	2	2
3	130.	Бильдин Сергей Михайлович	2	Второй	2	2
4	5.	Абдуалимзода Билолиддин Насрулдин	3	Первый	3	1
...
1174	1 175.	Ямалтдинова Зарина Камильевна	2	Четвертый	8	3
1175	1 176.	Янгибаев Абдукодир Азимович	0	Третий	7	2
1176	1 177.	Яхьяева Наталья Александровна	0	Первый	51	0
1177	1 178.	Яшонкина Карина Олеговна	2	Второй	13	2
1178	1 179.	Яценко Анастасия Александровна	2	Второй	27	2

1179 rows × 6 columns

Рис. 1.11. Получившийся датафрейм

Для замены строкового значения в столбце «Курс», также используем словарь и создаем дополнительный столбец с полем идентификатора id (рис. 1.12).

```
dict_course = {'Первый' : '1', 'Второй' : '2', 'Третий' : '3', 'Четвертый' : '4', 'Пятый': '5'}
df_new ['id'] = range(1, len(df_new ) + 1)
df_new ['Курс'] = df_new ['Курс'].map(dict_course)
df_new
```

	№п/п	ФИО	Гражданство	Курс	Специальность/направление	Уровень подготовки	id
0	1.	Абдихоидов Бехруз Бахтиёр угли		0		0	1
1	2.	Абдимиталипов Мустафа Тыныбекович		1		1	2
2	4.	Абдиреуков Самат Талгатулы		2		2	3
3	130.	Бильдин Сергей Михайлович		2		2	4
4	5.	Абдуалимзода Билолиддин Насруддин		3		3	5
...
1174	1 175.	Ямалтдинова Зарина Камильевна		2		8	1175
1175	1 176.	Янгибаев Абдукодир Азимович		0		7	1176
1176	1 177.	Яхьяева Наталья Александровна		0		51	1177
1177	1 178.	Яшонкина Карина Олеговна		2		13	1178
1178	1 179.	Ященко Анастасия Александровна		2		27	1179

1179 rows x 7 columns

Рис. 1.12. Код для замены строкового значения в столбце «Курс»

На рис. 1.13 считываем второй датасет в формате csv.

```
df_debts = pd.read_csv('/content/drive/MyDrive/data/debts.csv', sep=";")
df_debts
```

	№п/п	ФИО студентов по отчету должников	Гражданство	Курс	Специальность/направление	Уровень подготовки	Вид контроля	Задолженность	Дата/ оценка ППА	Пропуски занятий в 2022-2023 учебном году	Процент посещения в 2022-2023 учебном году	Примечание
0	1.0	Абдихоидов Бехруз Бахтиёр угли	УЗБЕКИСТАН	Второй	38 04 02 Менеджмент	Магистратура	экзамен	1. Коммуникационная политика в маркетинге/2. ...	13 10 2022 - неявка/10 05 10 2022 - неявка/10 10 2022 - неявка/10 10 2022 - неявка	38%	8%	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN	зачет	1. Инновационные компании - "единогори": от ид...	15 10 2022 - неявка/10 05 10 2022 - неявка	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	отчет по практике	1. Научно-исследовательская работа/2. Практик...	19 10 2022 - неявка/20 10 2022 - неявка/10 10 2022 - неявка	NaN	NaN	NaN
3	2.0	Абдуалимов Мирабдоржон Бехзод угли	УЗБЕКИСТАН	Третий	40 03 01. Юриспруденция	Бакалавриат	зачет	Уголовное право	15 10 - не зачтено	24%	47%	NaN
4	3.0	Абдуллаев Сардорбек Аброржон угли	УЗБЕКИСТАН	Четвертый	04 03 01. Химия	Бакалавриат	экзамен	Органическая химия	14 10 - неуд	77%	17%	NaN

Рис. 1.13. Считывание второй датасет в формате csv

Есть ненужные столбцы, удаляем их, а также заменяем столбец ФИО студентов по отчету должников на id из первой таблицы (рис. 1.13). Как видно из рисунка, в таблице содержатся пустые значения. Они получились из объединенных строчек в csv файле.

```
df_debts = df_debts.drop(columns=['Гражданство', 'Курс', 'Специальность/направление', 'Уровень подготовки', 'Дата/ оценка ППА', '№п/п'])
#Создаем словарь из ФИО и id студентов из первой таблицы для замены ФИО во второй
id_dict = dict(zip(df_new['ФИО'], df_new['id']))
df_debts ['ФИО студентов по отчету должников'] = df_debts ['ФИО студентов по отчету должников'].map(id_dict)
df_debts
```

	ФИО студентов по отчету должников	Вид контроля	Задолженность	Пропуски занятий в 2022-2023 учебном году	Процент посещения в 2022-2023 учебном году	Примечание
0	1.0	экзамен	1. Коммуникационная политика в маркетинге in2...	38%	8%	NaN
1	NaN	зачет	1. Инновационные компании - "единороги" от ид...	NaN	NaN	NaN
2	NaN	отчет по практике	1. Научно-исследовательская работа in2. Практик...	NaN	NaN	NaN
3	6.0	зачет	Уголовное право	24%	47%	NaN
4	9.0	экзамен	Органическая химия	77%	17%	NaN
...

Рис. 1.14. Замена столбец ФИО студентов по отчету должников на id

Перед тем как избавиться от пропусков перенесем столбцы Процент посещаемости и Пропуски занятий в первую таблицу (рис. 1.15). Как можем увидеть, в датафрейме данная информация отобразилась у студентов, имеющих какие-либо задолженности.

```
temp_df=df_debts[df_debts['ФИО студентов по отчету должников'].notna()]
truancy_dict = dict(zip(temp_df['ФИО студентов по отчету должников'], temp_df['Пропуски занятий в 2022-2023 учебном году']))
attendance_dict=dict(zip(temp_df['ФИО студентов по отчету должников'], temp_df['Процент посещения в 2022-2023 учебном году']))
df_new ['truancy'] = df_new['id'].map(truancy_dict)
df_new ['attendance'] = df_new['id'].map(attendance_dict)
df_new
```

	№п/п	ФИО	Гражданство	Курс	Специальность/направление	Уровень подготовки	id	truancy	attendance	
0	1.	Абдихидов Бехруз Бахтиёр угли		0	2	0	0	1	38%	8%
1	2.	Абдимиталипов Мустафа Тыныбекович		1	1	1	1	2	NaN	NaN
2	4.	Абдиреуков Самат Талгатулы		2	2	2	2	3	NaN	NaN
3	130.	Бильдин Сергей Михайлович		2	2	2	2	4	NaN	NaN
4	5.	Абдуалимзода Билолиддин Насрулдин		3	1	3	1	5	NaN	NaN
...
1174	1 175.	Ямалтдинова Зарина Камильевна		2	4	8	3	1175	NaN	NaN
1175	1 176.	Янгибаев Абдукодир Азимович		0	3	7	2	1176	27%	41%
1176	1 177.	Яхьяева Наталья Александровна		0	1	51	0	1177	NaN	NaN
1177	1 178.	Яшонкина Карина Олеговна		2	2	13	2	1178	NaN	NaN
1178	1 179.	Ященко Анастасия Александровна		2	2	27	2	1179	NaN	NaN

1179 rows x 9 columns

Рис. 1.15. Перенесение столбцы Процент посещаемости и Пропуски занятий

теперь удалим эти столбцы из второй таблицы (рис. 1.16).

```
[ ] df_debts = df_debts.drop(columns=['Пропуски занятий в 2022-2023 учебном году', 'Процент посещения в 2022-2023 учебном году'])
```

Рис. 1.16. Команда для удаления столбец из второй таблицы

Во второй таблице с задолженностями используем метод `fillna()` для заполнения значений `NaN` в столбце с `id` студента значением из предыдущей строки. Удаляем дубликаты и приводим поле `id` типу `integer`.

```
df_debts[['ФИО студентов по отчету должников', 'Вид контроля']] = df_debts[['ФИО студентов по отчету должников', 'Вид контроля']].fillna(method='ffill')
df_debts = df_debts[:-1]
df_debts=df_debts.drop_duplicates(subset=['ФИО студентов по отчету должников', 'Вид контроля', 'Задолженность'])
df_debts[['ФИО студентов по отчету должников']]=df_debts[['ФИО студентов по отчету должников']].astype(int)
df_debts
```

	ФИО студентов по отчету должников	Вид контроля	Задолженность	Примечание	Unnamed: 12
0	1	экзамен	1. Коммуникационная политика в маркетинге\n2. ...	NaN	NaN
1	1	зачет	1. Инновационные компании - "единороги": от ид...	NaN	NaN
2	1	отчет по практике	1. Научно-исследовательская работа\n2. Практик...	NaN	NaN
3	6	зачет	Уголовное право	NaN	NaN
4	9	экзамен	Органическая химия	NaN	NaN
...
198	1150	экзамен	Источниковедение	NaN	NaN
199	1160	экзамен	1. Профессиональная компетентность педагога\n2...	NaN	NaN
200	1169	экзамен	Основы статистической обработки данных измерений	NaN	NaN
201	1176	экзамен	Аналитическая химия	NaN	NaN
202	1176	экзамен		NaN	NaN

202 rows x 5 columns

Рис. 1.17. Удаление дубликаты и приводимое поле `id` типу `integer`

Далее, чтобы разделить задолженности студентов в отдельные строки, используем регулярные выражения: удаляем цифры, заменяем их на запятую или пустую строку. С помощью метода `explode()` разбиваем строку. Результат изображен на рис. 1.18.

```
df_debts['Задолженность'] = df_debts['Задолженность'].replace(to_replace=',', value='', regex=True)
df_debts['Задолженность'] = df_debts['Задолженность'].replace(to_replace='^[\d].', value='', regex=True)
df_debts['Задолженность'] = df_debts['Задолженность'].replace(to_replace='\n[\d].', value='', regex=True)
df_debts['Задолженность'] = df_debts['Задолженность'].replace(to_replace='[\d].', value='', regex=True)
lst_col = 'Задолженность'
df_debts = df_debts.assign(**{lst_col:df_debts[lst_col].str.split(',')})
df_debts=df_debts.explode('Задолженность')
df_debts['Задолженность'] =df_debts['Задолженность'].replace(to_replace='\n. ', value='', regex=True)
df_debts = df_debts.replace(to_replace='^s*$', value=float('NaN'), regex=True)
df_debts=df_debts.dropna(subset=['Задолженность'])
df_debts
```

	ФИО студентов по отчету должников	Вид контроля	Задолженность	Примечание	Unnamed: 12
0	1	экзамен	Коммуникационная политика в маркетинге	NaN	NaN
0	1	экзамен	Международный бизнес: стратегии развития	NaN	NaN
0	1	экзамен	Традиционные и инновационные концепции менедж...	NaN	NaN
1	1	зачет	Инновационные компании - "единороги": от идеи...	NaN	NaN
1	1	зачет	Событийный маркетинг	NaN	NaN
...
199	1160	экзамен	Зоогеография и история фаун	NaN	NaN
199	1160	экзамен	Вариативность социальных норм и девиации	NaN	NaN
199	1160	экзамен	Физическая культура и спорт: элективные курсы...	NaN	NaN
200	1169	экзамен	Основы статистической обработки данных измерений	NaN	NaN
201	1176	экзамен	Аналитическая химия	NaN	NaN

296 rows x 5 columns

Рис. 1.18. Результат задолженности студентов в отдельные строки

Теперь после обработки столбца с задолженностями, мы можем создать справочник с названиями дисциплин из уникальных значений, отсортировав их в алфавитном порядке. Подготовленный справочник временно хранится в датафрейме `df_dbt` (рис. 1.19).

```
[ ] temp_lst=df_debts['Задолженность'].unique().tolist()
df_dbt=pd.DataFrame(temp_lst,columns=["title"])
df_dbt['title'] = df_dbt['title'].replace(to_replace ='\n', value = '', regex = True)
df_dbt['title'] = df_dbt['title'].replace(to_replace = ' $', value = '', regex = True)
df_dbt = df_dbt.replace(to_replace = '^s*$', value =float('NaN'), regex = True)
df_dbt=df_dbt.dropna()
df_dbt=df_dbt.drop_duplicates(subset=None, keep="first", inplace=False)
df_dbt=df_dbt.sort_values('title')
df_dbt
```

	title
91	Web-технологии
123	Анализ и визуализация открытых данных
76	Ахивоведение и музееведение
137	Безопасность публичного управления и противод...
160	Вариативность социальных норм и девиации
...	...
140	Философия: технологии мышления
48	Цифровая культура
16	Цифровой маркетинг: продвижение проектов в соц...
148	Экономический анализ
117	Языки программирования

161 rows × 1 columns

Рис. 1.19. Подготовленный справочник, временно хранящийся в датафрейме `df_dbt`

Заменяем во второй таблице наименования задолженностей на соответствующие id (рис. 1.20).

```
[ ] df_dbt['id'] = range(1, len(df_dbt ) + 1)
debt_dict = dict(zip(df_dbt['title'], df_dbt['id']))
df_debts['Задолженность'] = df_debts['Задолженность'].map(debt_dict)
df_debts=df_debts.dropna(subset=['Задолженность'])
df_debts['Задолженность']=df_debts['Задолженность'].astype(int)
```

Рис. 1.20. Замена во второй таблице наименования задолженностей на соответствующие

Таким же образом создаём справочник с видами контроля и заменяем наименования на соответствующие id (рис. 1.21).

```
temp_lst=df_debts['Вид контроля'].unique().tolist()
df_cntr=pd.DataFrame(temp_lst,columns=["title"])
df_cntr['id'] = range(1, len(df_cntr ) + 1)
cntr_dict = dict(zip(df_cntr['title'], df_cntr['id']))
df_debts['Вид контроля'] = df_debts['Вид контроля'].map(cntr_dict)
df_debts['id'] = range(1, len(df_debts ) + 1)
df_debts = df_debts.drop('Unnamed: 12', axis=1)
df_debts
```

Рис. 1.21. Справочник с видами контроля и заменой наименования на соответствующие

Итоговый датафрейм с задолженностями изображён на рис. 1.22.

```
[ ] df_debts
```

	ФИО студентов по отчету должников	Вид контроля	Задолженность	Примечание	id	
0		1	1	29	NaN	1
0		1	1	37	NaN	2
0		1	1	79	NaN	3
1		1	2	14	NaN	4
1		1	2	72	NaN	5
...
199	1160	1	120	NaN	284	
199	1160	1	13	NaN	285	
199	1160	1	5	NaN	286	
200	1169	1	129	NaN	287	
201	1176	1	97	NaN	288	

288 rows × 5 columns

Рис. 1.22. Итоговый датафрейм с задолженностями

Создаем курсор - это специальный объект для проведения SQL команд, который делает запросы и получает их результаты. (рис. 1.23).



Рис. 1.23. Код для проведения SQL команд

Создаём 2 таблицы и 5 справочников с нужными нам таблицами, не забывая указать первичные и внешние ключи, и наполняем их данными из соответствующих датафреймов (рис. 1.24, рис. 1.25, рис. 1.26, рис. 1.27, рис. 1.28, рис. 1.29, рис. 1.30).

```
[ ] create_ctrl="CREATE TABLE IF NOT EXISTS TypesOfControl (id INTEGER PRIMARY KEY,title TEXT NOT NULL)"
cursor.execute(create_ctrl)

[ ] for row in df_cntr.itertuples():
    insert_sql=f"INSERT INTO TypesOfControl (id,title) VALUES ({row[2]},'{row[1]}')"
    cursor.execute(insert_sql)
```

Рис. 1.24. Указание первичные и внешние ключи, и данный из соответствующих датафреймов 1

```
[ ] create_ctzsh="CREATE TABLE IF NOT EXISTS Citizenships (id INTEGER PRIMARY KEY,title TEXT NOT NULL)"
cursor.execute(create_ctzsh)
for row in df_citizenship.itertuples():
    insert_sql=f"INSERT INTO Citizenships (id,title) VALUES ({row[2]},'{row[1]}')"
    cursor.execute(insert_sql)
conn.commit()
```

Рис. 1.25. Указание первичные и внешние ключи, и данный из соответствующих датафреймов 2

```
[ ] create_d="CREATE TABLE IF NOT EXISTS Degrees (id INTEGER PRIMARY KEY,title TEXT NOT NULL)"
cursor.execute(create_d)
for row in df_degree.itertuples():
    insert_sql=f"INSERT INTO Degrees (id,title) VALUES ({row[2]},'{row[1]}')"
    cursor.execute(insert_sql)
conn.commit()
```

Рис. 1.26. Указание первичные и внешние ключи, и данные из соответствующих датафреймов 3

```
[ ] create_s="CREATE TABLE IF NOT EXISTS Specialities (id INTEGER PRIMARY KEY,title TEXT NOT NULL)"
cursor.execute(create_s)
for row in df_speciality.itertuples():
    insert_sql=f"INSERT INTO Specialities (id,title) VALUES ({row[2]},'{row[1]}')"
    cursor.execute(insert_sql)
conn.commit()
```

Рис. 1.27. Указание первичные и внешние ключи, и данные из соответствующих датафреймов 4

```
▶ create_db="CREATE TABLE IF NOT EXISTS Debts (id INTEGER PRIMARY KEY,title TEXT NOT NULL)"
cursor.execute(create_db)
for row in df_debts.itertuples():
    insert_sql=f"INSERT INTO Debts (id,title) VALUES ({row[2]},'{row[1]}')"
    cursor.execute(insert_sql)
conn.commit()
```

Рис. 1.28. Указание первичные и внешние ключи, и данные из соответствующих датафреймов 5

```
[ ] create_s="CREATE TABLE IF NOT EXISTS Students (id INTEGER PRIMARY KEY,name TEXT NOT NULL, citizenshipId INTEGER, course INTEGER, specialityId INTEGER, degreeId INTEGE
cursor.execute(create_s)
for row in df_new.itertuples():
    insert_sql=f"INSERT INTO Students (id,name, citizenshipId, course, specialityId, degreeId, truancy, attendance) VALUES ({row[7]},'{row[2]}',{row[3]},{row[4]},{row[
    cursor.execute(insert_sql)
conn.commit()
```

Рис. 1.29. Указание первичные и внешние ключи, и данные из соответствующих датафреймов 6

```
[ ] create_sd="CREATE TABLE IF NOT EXISTS StudentsDebts (id INTEGER PRIMARY KEY,studentId INTEGER, typeOfControlId INTEGER,debtId INTEGER, note TEXT,FOREIGN KEY (studentId) F
cursor.execute(create_sd)
for row in df_debts.itertuples():
    insert_sql=f"INSERT INTO StudentsDebts (id,studentId, typeOfControlId, debtId, note) VALUES ({row[5]},'{row[1]}',{row[2]},{row[3]},{row[4]}'"
    cursor.execute(insert_sql)
conn.commit()
```

Рис. 1.30. Указание первичные и внешние ключи, и данные из соответствующих датафреймов 7

Закрываем соединение с базой данных (рис. 1.31).



Рис. 1.31. Команда для закрывания соединения с базой данных

Структура полученной базы данных в файле mydatabase.db предоставлена ниже (рис. 1.32).

Имя	Тип	Схема
Таблицы (7)		
Citizenships		CREATE TABLE Citizenships (id INTEGER PRIMARY KEY,title TEXT NOT NULL)
id	INTEGER	"id" INTEGER
title	TEXT	"title" TEXT NOT NULL
Debts		CREATE TABLE Debts (id INTEGER PRIMARY KEY,title TEXT NOT NULL)
id	INTEGER	"id" INTEGER
title	TEXT	"title" TEXT NOT NULL
Degrees		CREATE TABLE Degrees (id INTEGER PRIMARY KEY,title TEXT NOT NULL)
id	INTEGER	"id" INTEGER
title	TEXT	"title" TEXT NOT NULL
Specialities		CREATE TABLE Specialities (id INTEGER PRIMARY KEY,title TEXT NOT NULL)
id	INTEGER	"id" INTEGER
title	TEXT	"title" TEXT NOT NULL
Students		CREATE TABLE Students (id INTEGER PRIMARY KEY,name TEXT NOT NULL, citizenshipId INTEGER,
id	INTEGER	"id" INTEGER
name	TEXT	"name" TEXT NOT NULL
citizenshipId	INTEGER	"citizenshipId" INTEGER
course	INTEGER	"course" INTEGER
specialityId	INTEGER	"specialityId" INTEGER
degreeId	INTEGER	"degreeId" INTEGER
truancy	TEXT	"truancy" TEXT
attendance	TEXT	"attendance" TEXT
StudentsDebts		CREATE TABLE StudentsDebts (id INTEGER PRIMARY KEY,studentId INTEGER, typeOfControlId INT
id	INTEGER	"id" INTEGER
studentId	INTEGER	"studentId" INTEGER
typeOfControlId	INTEGER	"typeOfControlId" INTEGER
debtId	INTEGER	"debtId" INTEGER
note	TEXT	"note" TEXT
TypesOfControl		CREATE TABLE TypesOfControl (id INTEGER PRIMARY KEY,title TEXT NOT NULL)
id	INTEGER	"id" INTEGER
title	TEXT	"title" TEXT NOT NULL

Рис. 1.32. Структура полученной базы данных в файле mydatabase.db

	id	name	citizenshipId	course	specialityId	degreeId	truancy	attendance
	Фи...	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр
1	1	Абдивоихидов Бехруз Бахтиёр угли	0	2	0	0	38%	8%
2	2	Абдимиталипов Мустафа ...	1	1	1	1	nan	nan
3	3	Абдиреуков Самат Талгатулы	2	2	2	2	nan	nan
4	4	Бильдин Сергей Михайлович	2	2	2	2	nan	nan
5	5	Абдуалимзода Билолиддин ...	3	1	3	1	nan	nan
6	6	Абдуалимов Мираброржон Бехзод ...	0	3	4	2	24%	47%
7	7	Абдуганиев Илхомжон Холмурод угли	0	1	5	2	nan	nan
8	8	Абдуллаев Жахонгир Жалол угли	0	1	6	2	nan	nan
9	9	Абдуллаев Сардорбек Аброржон угли	0	4	7	2	77%	17%
10	10	Абдуллин Амаль Фуадович	0	4	8	3	nan	nan
11	11	Абдуллоев Мехрубон ...	3	1	9	2	nan	nan
12	12	Абдуллоев Шахзод Абдумаликович	3	4	7	2	94%	0%
13	13	Абдулов Некруз Нуралиевич	3	2	10	0	26%	50%
14	14	Абдуманнобов Хондамир Илхомжо...	0	3	11	2	nan	nan
15	15	Абдуманонов Шахбоз Раим угли	0	3	6	3	43%	7%
16	16	Абдуназаров Абдуназар ...	3	1	6	2	nan	nan
17	17	Абдураимов Фазлиддин Комилжон ...	0	2	4	2	nan	nan
18	18	Абдурахмонова Шириной Акмалжо...	0	2	9	4	nan	nan
19	19	Абдурашидова Асалгул ...	0	1	12	2	nan	nan
20	20	Абдусалимов Зухриддин Бахтиёр ...	0	1	13	2	nan	nan
21	21	Абдусаломов Муродулла Абдувохид...	0	1	9	2	nan	nan
22	22	Абдусаломов Садихжан ...	0	2	14	0	nan	nan
23	23	Абдусаломов Дилноза Баходир кизи	0	1	9	2	nan	nan

Рис. 1.33. Таблица Students

	id	studentId	typeOfControlId	debtId	note
	Фи...	Фильтр	Фильтр	Фильтр	Фильтр
1	1	1	1	29	nan
2	2	1	1	37	nan
3	3	1	1	79	nan
4	4	1	2	14	nan
5	5	1	2	72	nan
6	6	1	3	40	nan
7	7	1	3	55	nan
8	8	6	2	150	nan
9	9	9	1	127	nan
10	10	12	1	49	nan
11	11	12	1	88	nan
12	12	12	4	118	nan
13	13	13	5	123	Сдал
14	14	15	1	144	nan
15	15	27	6	119	nan
16	16	34	1	105	nan
17	17	34	2	159	nan
18	18	41	1	30	nan

Рис.1.34. Таблица StudentsDebts

1.5 ПРОБЛЕМЫ, СВЯЗАННЫЕ С ОСОБЕННОСТЯМИ ОРГАНИЗАЦИИ ИСХОДНЫХ ДАННЫХ

Исходные данные изначально располагались в файлах EXCEL, в формате, удобном для визуального изучения. Хранение информации в базе данных требует выполнения нормализации. Так как по ходу предварительной обработки данных они переводятся в вид датафрейма, возникают несколько проблем с однородностью данных.

1. Во входном документе не одна таблица, а две таблицы расположенных вплотную. При таком виде документа формата “.xlsx” считывание документа приводит к получению только одной из таблиц. Решением было создать новый документ куда записаны все данные в одну таблицу.
2. В документе есть ситуации, когда для одной строки входного документа указаны фиио, гражданство, специализация т.д., но при указании типа контроля для предмета, в некоторых случаях указано что все предметы будут «экзамен» или «зачет», как показано на рис.1.1. Либо как на рис.1.2, когда предметы и соответствующие им виды контроля разделены просто пронумерованы. Решением было программное разделение строк, и создание нового датафрейма в которой в одной строке будут только одна связка вид контроля – задолженность – дата\оценка ППА.
3. В документе много вариантов названий и орфографических ошибок в колонке «Вид контроля». Их можно увидеть на картинке 3. Проблема не решена. Программа является решением для данной проблемы так как можно будет не записывать вручную тип контроля предмета, а выбирать из нескольких изменяемых вариантов как показано на рис.1.1. Таким же образом можно будет выбирать предмет, гражданство, уровень подготовки и специальность\направление.
4. В таблице входных данных присутствовало множество пустых строк, что требовало нескольких дополнительных шагов при предобработке

данных. Пока данные небольшие эта проблема не сильно влияет на работу приложения, однако сложность алгоритма, используемого для этого $O(N)$ что означает что при больших данных может существенно увеличить время запуска программы.

ГЛАВА 2. РАЗРАБОТКА ПРИЛОЖЕНИЯ

2.1 ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИИ

Создано приложения для десктопных устройств на базе MacOS, Linux, Win 7,8,10. Использовался язык программирования python, так как он предоставляет множество удобных библиотек для работы с таблицами, создания интерактивных и наглядных графиков. Поддерживает возможность работы с базами данных.

В проекте использовались такие библиотеки как:

- 1) Pandas – работа с датафреймами. Предобработка и изменение данных.
- 2) PyQt – специальный фреймворк для создания межплатформенных десктопных оконных приложений. Из PyQt были взяты и использованы следующие модули:
 - QApplication – основной модуль для работы с приложением.
 - QDialog – модуль для создания диалоговых окон с вводом от пользователя
 - QVBoxLayout – модуль для задачи окну нужной разметки, лейаута
 - QWidget - модуль для создания и работы с компонентами.
 - QMainWindow – модуль для работы с главным окном, это специальный класс, с уникальным лейаутом, один из обязательных модулей без которых не будет работать никакое приложение на PyQt
 - QTableView – модуль для работы с таблицами
 - QPushButton – модуль для работы с кнопками
 - QLineEdit – модуль для работы с текстовыми полями для ввода пользователем
 - QLabel – модуль для работы с текстовыми полями
 - QHBoxLayout – модуль похожий на QVBoxLayout, толичается ориентаций, H - горизонтальный V - вертикальный
 - QDialog – модуль для работы с диалоговыми окнами без ввода от пользователя

- QFormLayout – модуль для работы с разметкой
- QListWidget – модуль для работы со списками
- QMessageBox – модуль для работы с компонентами оповещения пользователя

3) Sqlite3 – для работы с базой данных sqlite

4) Plotly – для работы с графиками.

2.2 АРХИТЕКТУРА ПРИЛОЖЕНИЯ

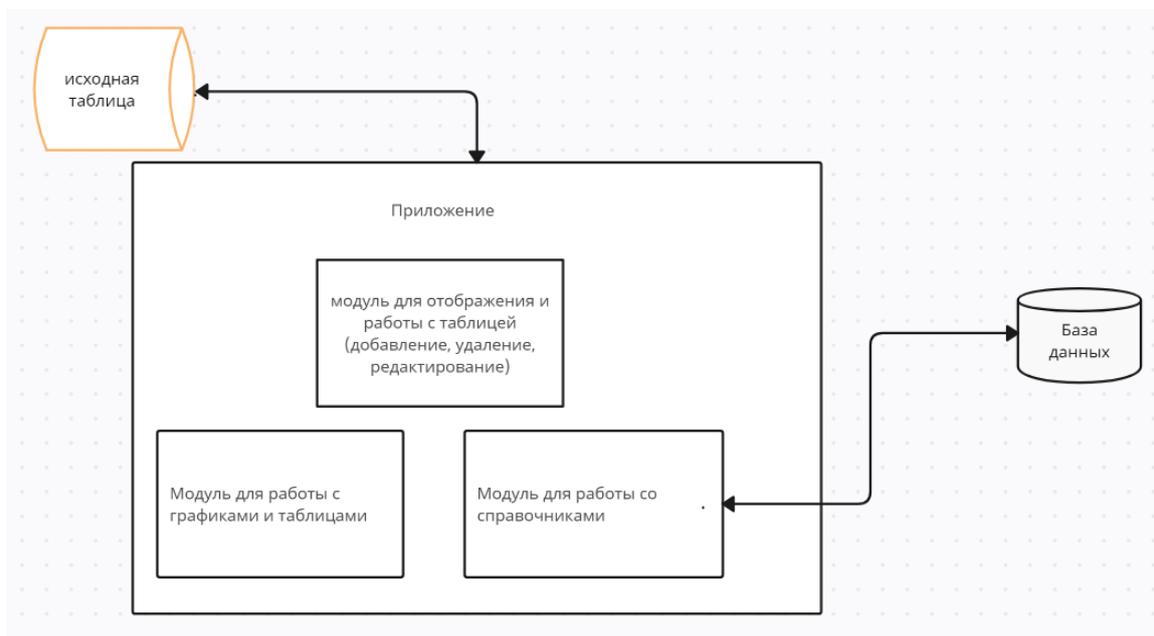


Рис 2.1. Схема архитектуры приложения.

Приложение состоит из 5 главных компонентов, краткое описание каждого из них

1) Компонент главного меню включает в себя:

- Таблицу, которая представляет датафрейм с информацией о студентах
- Кнопки по редактированию таблицы
- Кнопки для управления справочниками
- Кнопка для открытия окна с графиками.

- Компоненты для поиска по имени студента – текстовое поле для ввода ФИО и кнопка для начала поиска.

Главное меню является для пользователя начальным окном при запуске программы, позволяет изменять, добавлять и удалять строки студентов из таблицы. При вводе ФИО и нажатии кнопки поиска таблица редактируется и остаются только студенты чьё имя фамилию ввели. Для того чтобы удалить или отредактировать строку, её нужно выбрать на таблице. Работает при поиске студента.

- 2) Окно для просмотра и работы со справочниками. Справочники собираются из представленной таблицы, Справочники собираются по нескольким полям, таким как: специальность, курс, предмет задолженности, вид контроля по данному предмету. Можно удалять, редактировать или добавлять новые элементы справочника.
- 3) Модуль работы с графиками. С помощью технологий, предоставленных plotly и PyQt, создаются на основе представленной таблицы несколько графиков. График показывающий студента – количество пропущенных занятий, студента – процент посещений, график посещений в разрезе национальности. Данные графики преобразуются в html разметку с использованием js, далее с помощью WebEngineView представляются пользователю на экран в отдельном окне, графики частично интерактивные. Не требует подключения к сети.
- 4) Модуль для работы с базой данных. Все справочники сохраняются в базу данных. При первом запуске программы они пусты, происходит миграция таблиц. Что даёт возможность для разных таблиц иметь разные справочники, в этой программе, в зависимости какие данные содержатся в таблице. При изменениях в справочниках в программе изменяются таблица в базе данных.
- 5) Модуль предобработки таблиц. Из таблицы удаляются большинство ненужных нумераций. Таблица оформляется в вид где у каждого студента в одной строчке может быть только одна задолженность и результат. Далее будет рассмотрено детально как это достигается.

Приложение считывает таблицу при запуске программы, обрабатывает её для понятного и однозначного отображения для пользователя. Обработанная таблица отображается на главном окне. Любое редактирование таблицы в реальном времени будет изменять значения на таблице, которая показывается пользователю. Это позволяет пользователю исправлять ошибки. Сама таблица, которая отображается пользователю, показывает только самые важные столбцы. Двойное нажатие показывает полную информацию о строке. Значения справочников из таблицы загружаются в базу данных с проверкой уникальности. Это позволяет в справочниках всегда иметь полную информацию для всех таблиц, которые были загружены в программу на данном компьютере. Так же любые изменения справочников в приложении сразу же отражаются в базе данных.

2.3. ЭЛЕМЕНТЫ ПРОГРАММЫ

Таблица 2.1

Классы и методы `class MyWindow(QMainWindow)` класс для главного меню приложения методы

Классы и методы	Описание
<code>def __init__(self):</code>	Стандартный конструктор
<code>def init_ui(self):</code>	Метод, вызывается при в методе <code>__init__</code> , строит архитектуру окна, в нём объявляются все компоненты и события на Главном окне
<code>def open_graph_window(self):</code>	Вызывающийся при нажатии на одну из кнопок метод внутри которого строятся графики и вызывается открытие окна класса <code>GraphWindow</code>

Продолжение таблицы 2.1

Классы и методы	Описание
def open_input_dialog(self):	Метод, который вызывает открытия диалогового окна класса InputDialog, также внутри метода описывается добавление новой строки в таблицу.
def open_edit_dialog(self):	Метод тоже вызывает открытие окна InputDialog, но передаёт в него значения выбранной строки и индекс. Так же в нём описано изменение значений в строке.
def open_reference_editor(self, table_name):	Метод для открытия Редактора справочников.
def search_by_name (self):	Метод для поиска по имени студента, внутри метода описано изменение таблицы

Таблица 2.2

class GraphWindow(QMainWindow): Класс веб- окна программы для отображения графиков

Класс	Описание
def __init__(self, plot_html):	Конструктор в котором определяются компоненты окна и отрисовывается графики plot_html

Текст кода класса GraphWindow представлен в Приложении 1.

Таблица 2.3

class DataFrameModel(QAbstractTableModel): Класс – модель для работы с таблицей.

Класс	Описание
def __init__(self, data):	Конструктор, принимает данные таблицы.
def data(self, index, role):	Метод для заполнения таблицы данными

Класс	Описание
def rowCount(self, index):	Метод возвращает кол-во столбцов
def columnCount(self, index):	Метод возвращает кол-во строк

Таблица 2.4

```
class InputDialog(QDialog):
```

Класс для работы с окном при добавлении строки в таблицу, либо при редактировании

Класс	Описание
def __init__(self, connection):	Конструктор, вызывается при запуске программы, получает на вход connection для подключения к базе данных
def load_reference_data(self):	Метод, вызывающийся при запуске конструктора, запрашивает информацию о справочниках из базы данных
def init_ui(self):	Метод для построения окна диалога, тут описываются все компоненты и события.
def create_widget(self, column):	Метод, который заполняет выпадающий список элементами, на основе того какой это столбец.
def set_row_data(self, data):	Метод, описывающий как данные будут отображаться на диалоговом окне при запуске редактирования строки
def get_row_data(self):	Метод для возвращения данных и диалогового окна на родительский, в нашем случае MyWindow

class ReferenceEditorDialog(QDialog): Класс окна для работы со справочниками

Класс	Описание
def __init__(self, connection, table_name):	Конструктор, получает connection для работы с sqlite3 и table_name для того чтобы понимать с каким именно справочником будет работа
def load_data(self):	Метод для получения справочника из базы данных
def init_ui(self):	Метод для описания компонентов и событий окна
def add_item(self):	Метод для добавления в бд новой записи справочника, также перезагружает данные, чтобы пользователь всегда имел актуальные данные
def edit_item(self):	Метод похож на предыдущий однако, вместо добавления, новые данные перезаписывают имеющуюся запись.
def delete_item(self):	Метод удаляет запись из бд, обновляет данные на окне.

Общие методы

Класс	Описание
def def migration():	Заполняет базу данных информацией справочников. Вызывает при запуске программы после получения csv таблицы.
def migration():	Создаются все необходимые таблицы если их не было в БД, вызывает сразу перед def migration():

Вид контроля	Задолженность	Дата/ оценка ПИА
экзамен	маркетинге 2. Международный бизнес:	05.10.2022 - неявка 04.10.2022 - неявка
зачет	1. Инновационные компании - "единороги": от идеи до выхода на глобальный рынок 2. Событийный маркетинг	15.10.2022 - неявка 06.10.2022 - неявка
отчет по праактике	2. Практика по профилю профессиональной деятельности	19.10.2022 - неявка 20.10.2022 - неявка

Рис. 2.1. Часть таблицы, иллюстрирующая один из вариантов

Таким образом, эта часть таблицы представляет собой наглядный и компактный способ визуализации ключевых характеристик студентов, позволяя быстро оценить и сравнить их успеваемость, активность и прогресс. Это ценный инструмент для анализа данных, принятия решений и разработки индивидуальных стратегий развития студентов.

1. экзамен		1. 03.10.2022 - неявка
2. экзамен		2. 04.10.2022 - неявка
3. экзамен	1. Профессиональная компетентность педагога	3. 05.10.2022 - неявка
4. зачет	2. Школа вожатых	4. 10.10.2022 - неявка
5. экзамен	3. Летняя педагогическая (вожатская) практика	5. 11.10.2022 - неявка
6. экзамен	4. Познавательное развитие дошкольников (с практикумом)	6. 12.10.2022 - неявка
7. зачет	5. Социальная практика	7. 18.10.2022 - неявка
8. зачет	6. Научно-исследовательская работа (с научно- проектным семинаром)	8. 19.10.2022 - неявка
9. зачет	7. Зоогеография и история фаун 8. Вариативность социальных норм и девиации 9. Физическая культура и спорт: элективные курсы по видам спорта 4 семестр	9. 20.10.2022 - неявка

Рис. 2.2. Случай организации колонок вида контроля, предмета и оценки

В программе разработанного приложения предусмотрена функциональность, отображающая все доступные варианты типов контроля. В следующей части программы представлен список типов контроля, который включает:

1. Экзамен: данный тип контроля предполагает оценку знаний и умений студента в определенной предметной области путем проведения тестирования или письменного экзамена. Результат экзамена может влиять на итоговую оценку студента по предмету.
2. Зачет: зачет представляет собой форму контроля, где студент должен продемонстрировать свои знания и понимание учебного материала в устной или письменной форме. Зачет может быть промежуточным или итоговым и обычно несет определенное количество баллов или ставится оценка "зачтено" или "незачтено".
3. Отчет по практике: данный тип контроля предполагает подготовку и представление студентом отчета о выполненной практической работе или проекте. Оценка за отчет может основываться на качестве представленной работы, практических навыках студента и его способности анализировать и интерпретировать полученные результаты и т.д.

Таким образом, в программе предусмотрена возможность выбора из этих вариантов типов контроля, в зависимости от требований и особенностей учебного процесса. Это обеспечивает гибкость и адаптируемость приложения к различным образовательным программам и потребностям пользователей.

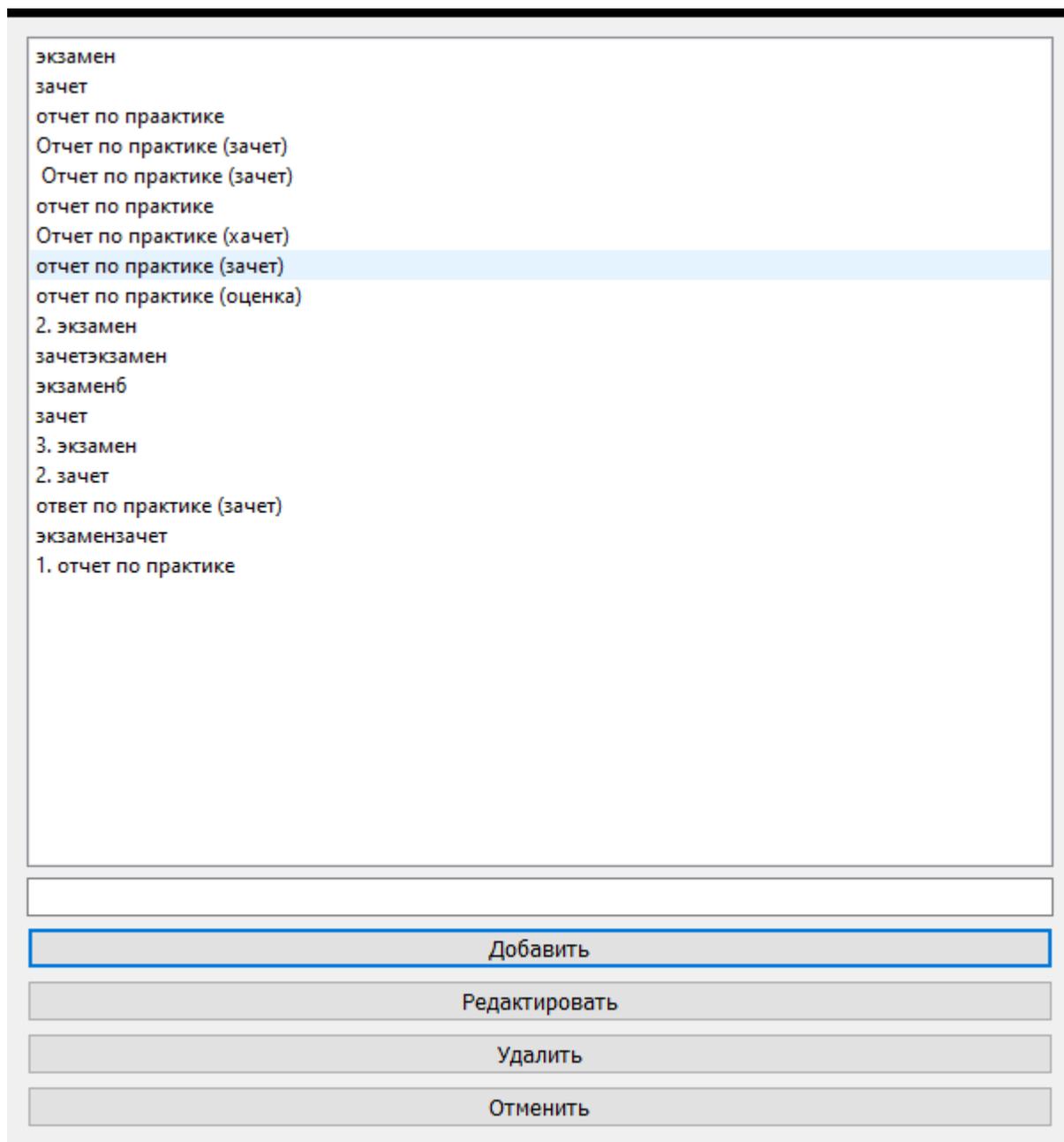


Рис.2.3 Часть программы показывающая все варианты типов контроля

В следующей части программы предусмотрена функциональность, позволяющая добавить новую строку с выбором типа контроля. Это позволяет пользователю гибко управлять типами контроля в системе.

Пользователь будет приглашён ввести новый тип контроля. После этого, новый тип будет добавлен в список доступных типов контроля, который может быть использован в других частях программы.

Таким образом, данная часть программы позволяет пользователю динамически расширять список доступных типов контроля, адаптируя его к конкретным требованиям и потребностям образовательного процесса.

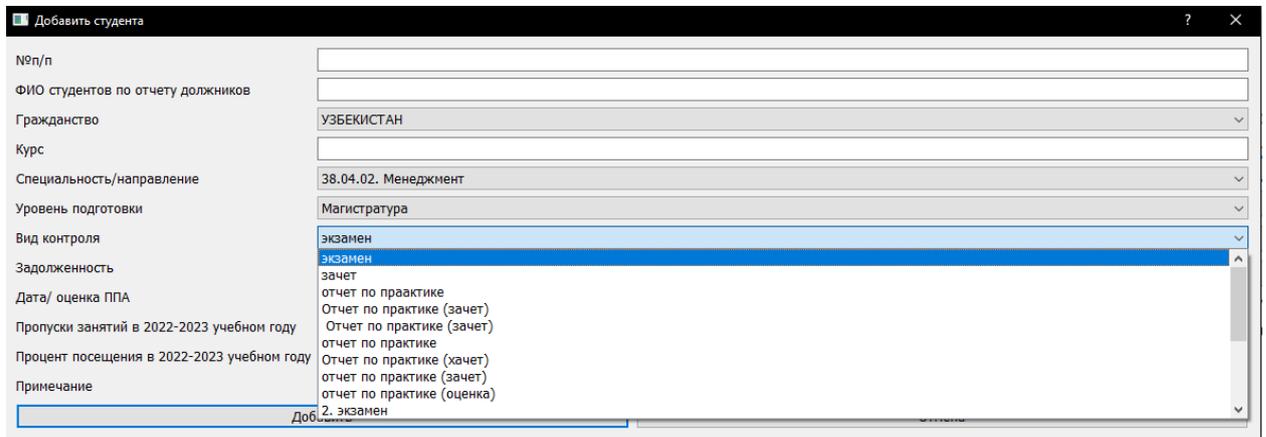


Рис 2.4. Часть программы, позволяющая добавить новую строчку с выбором типа контроля

На рисунке 2.5. представлен фрагмент кода, который демонстрирует функции для работы со справочниками. Справочники - это структуры данных, содержащие информацию, которая используется в приложении для быстрого доступа и удобного поиска.

```
def add_item(self):
    item_name = self.line_edit.text()
    if item_name:
        cursor = self.connection.cursor()
        cursor.execute(f'INSERT INTO {self.table_name} (name) VALUES (?)', (item_name,))
        self.connection.commit()
        self.load_data()

def edit_item(self):
    selected_item = self.list_widget.currentItem()
    if selected_item:
        new_name, ok = QDialog.getText(self, 'Редактировать', 'Введите имя:', text=selected_item.text())
        if ok and new_name:
            cursor = self.connection.cursor()
            cursor.execute(f'UPDATE {self.table_name} SET name = ? WHERE name = ?', (new_name, selected_item.text()))
            self.connection.commit()
            self.load_data()

def delete_item(self):
    selected_item = self.list_widget.currentItem()
    if selected_item:
        reply = QMessageBox.question(self, 'Удалить', 'Вы уверены?',
                                     QMessageBox.Yes | QMessageBox.No, QMessageBox.No)
        if reply == QMessageBox.Yes:
            cursor = self.connection.cursor()
            cursor.execute(f'DELETE FROM {self.table_name} WHERE name = ?', (selected_item.text(),))
            self.connection.commit()
            self.load_data()
```

Рис 2.5. Фрагмент кода, представляющий функции для работы со справочниками

Фрагмент кода представляет собой набор функций, которые обеспечивают удобную работу с справочниками в приложении. Они предоставляют функциональность добавления, удаления, обновления и поиска записей в справочнике, что позволяет эффективно управлять информацией и использовать ее в других частях программы.

2.5 ИНТЕРФЕЙС ПОЛЬЗОВАТЕЛЯ

Пользователю представлено главное меню с возможностью выбрать одну из кнопок

- Кнопка для поиска по имени фамилии, выведет на экран только данные студента, которого вы ищете.
- Кнопка для редактирования данных выделенной на таблице строки.
- Кнопка для добавления новой строки.
- 4 кнопки для просмотра и редактирования каждого из справочников. При нажатии появится окно, в котором вы можете просмотреть какие есть записи в данном справочнике. Добавить либо редактировать любую запись.
- Кнопка «Построение табличных и графических отчетов» откроет для вас окно в которой вы можете просматривать и изменять параметры таблицы и графиков.

Параметры изменяются с помощью выпадающих списков.

Интерфейс приложения показан на рисунке 2.6

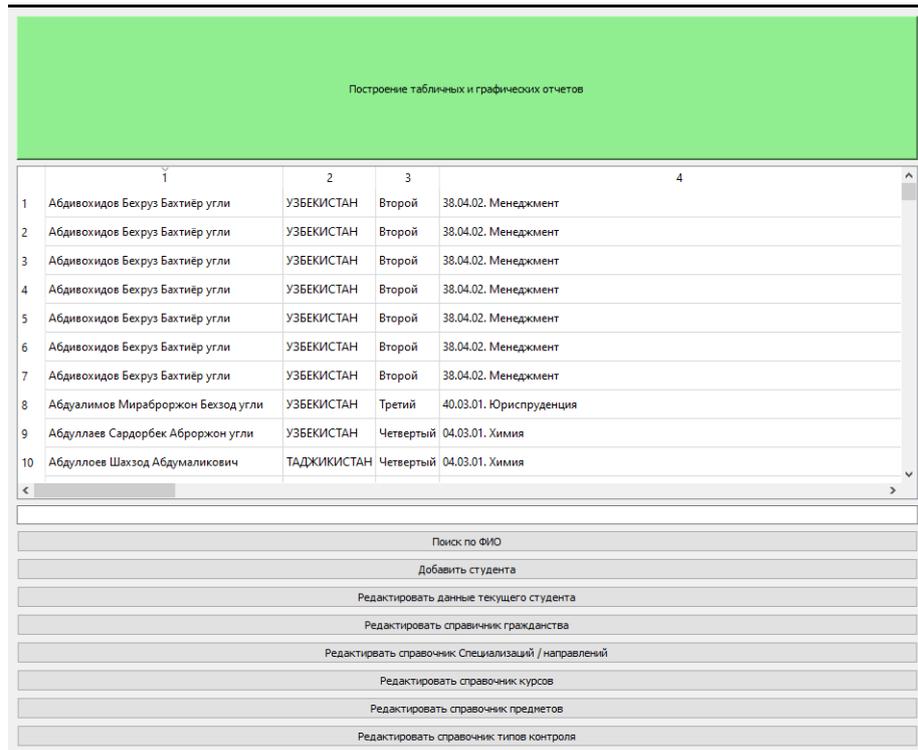


Рис 2.6. Интерфейс приложения

Интерфейс окна с таблицей и диаграммами обладает несколькими полезными функциями. Пользователь может выполнять различные операции, такие как сортировка, фильтрация, поиск и редактирование данных непосредственно в таблице. Кроме того, графики могут быть настроены с использованием различных параметров, таких как масштаб, цветовая схема, подписи осей и т.д., чтобы адаптироваться к потребностям и предпочтениям пользователя.

Такой интерфейс позволяет эффективно визуализировать и анализировать большие объёмы данных, облегчая процесс принятия решений, выявления тенденций и обнаружения паттернов в информации.

Элементы управления, позволяющие выполнить фильтрацию данных для построения таблицы с количеством студентов показан на рисунке 2.7.

Построить график анализа количества студентов сгруппированных по:

Курс

График будет отображать количество студентов:

Должников

Фильтр:

Гражданство

Курс	Гражданство	№п/п
Второй	АЗЕРБАЙДЖАН	18
Второй	БЕЛАРУСЬ	1
Второй	КАЗАХСТАН	90
Второй	ТАДЖИКИСТАН	34
Второй	УЗБЕКИСТАН	49
Пятый	КАЗАХСТАН	1
Пятый	УЗБЕКИСТАН	1
Третий	ГРУЗИЯ	1
Третий	КАЗАХСТАН	48
Третий	ТАДЖИКИСТАН	13
Третий	УЗБЕКИСТАН	16
Четвертый	АРМЕНИЯ	3
Четвертый	КАЗАХСТАН	15

Рис 2.7. Интерфейс окна с таблицей и графиками

2.6. ВЫВОД ИНФОРМАЦИИ В ТЕКСТОВОМ И ГРАФИЧЕСКОМ ВАРИАНТАХ

Приложение позволяет пользователю просмотреть таблицы и графики по разным разрезам. В данной программе реализовано построение графиков и таблицы по 2 разрезам и могут учитываться только должники или все студенты. Должниками считаются студенты, упомянутые в записях, в которых оценка – неуд, незачёт, неявка. Таблица, показывающая комбинацию 2 разрезов и количество задолженностей по данным разрезам как видно на Рис. 2.7. То-есть первая строчка означает, что на втором курсе граждан Азербайджана 18 должников. Реализованы 2 вида графиков, гистограмма с наростами на рисунке 2.8 и круговая диаграмма на рисунке 2.9.

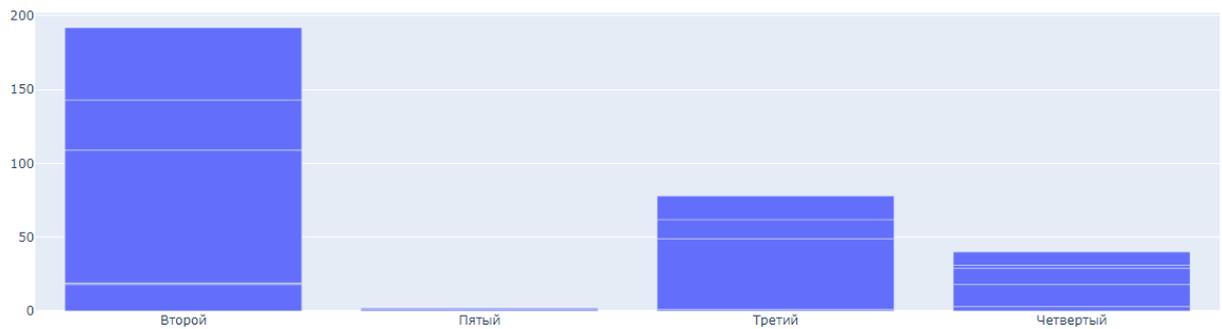


Рис 2.8. Гистограмма с наростами, наростами являются Гражданства, а столбцы номером курса

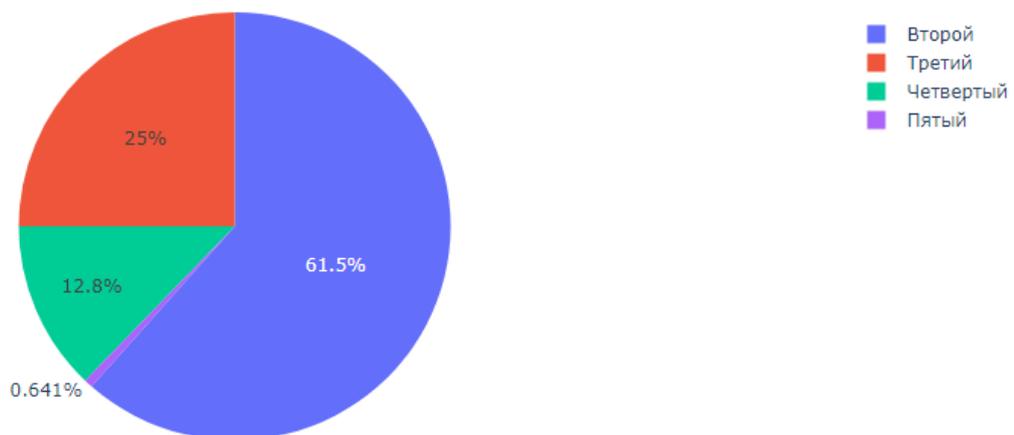


Рис 2.9. Круговая диаграмма по курсам

Другой пример таблицы по 2 разрезам – Вид контроля и гражданство. Таблица будет показывать количество должников каждого гражданства по каждому из видов контроля как видно на рисунке 2.10.

Гражданство	Вид контроля	№п/п
АЗЕРБАЙДЖАН	зачет	1
АЗЕРБАЙДЖАН	экзамен	17
АРМЕНИЯ	экзамен зачет	3
БЕЛАРУСЬ	экзамен	1
ГРУЗИЯ	экзамен	1
КАЗАХСТАН	3. Отчет по практике (зачет)	1
КАЗАХСТАН	зачет	2
КАЗАХСТАН	1. Экзамен	1
КАЗАХСТАН	2. Зачет	1
КАЗАХСТАН	4. зачет	1
КАЗАХСТАН	Зачет	2
КАЗАХСТАН	Отчет по практике (зачет)	1
КАЗАХСТАН	зачет	48

Рис. 2.10. Таблица по гражданству и виду контроля

Интерфейс пользователя для выбор категорий для разрезов показан на рисунках 2.11 и 2.12.

Построик график анализа количества студентов сгруппированных по:

Вид контроля
 Вид контроля
 Курс
 Должников

Фильтр:
 Гражданство

Гражданство	Вид контроля	№п/п
АЗЕРБАЙДЖАН	зачет	1
АЗЕРБАЙДЖАН	экзамен	17
АРМЕНИЯ	экзамен зачет	3
БЕЛАРУСЬ	экзамен	1
ГРУЗИЯ	экзамен	1
КАЗАХСТАН	3. Отчет по практике (зачет)	1
КАЗАХСТАН	зачет	2
КАЗАХСТАН	1. Экзамен	1
КАЗАХСТАН	2. Зачет	1
КАЗАХСТАН	4. зачет	1

Рис. 2.11. Организация возможности выбора первого разреза.

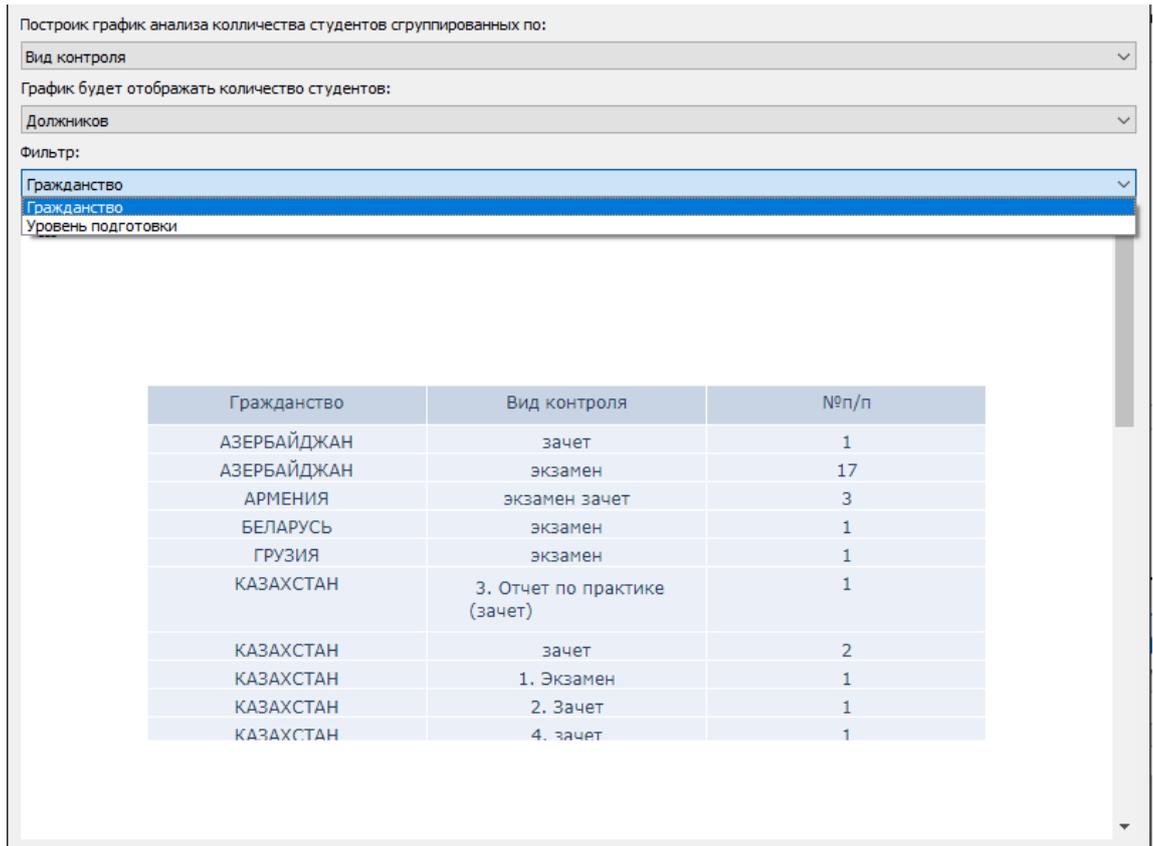


Рис. 2.12. Организация возможности выбора второго разреза.

Одновременно для визуализации данных также создаются две диаграммы, которые можно увидеть на рисунках 2.13 и 2.14.

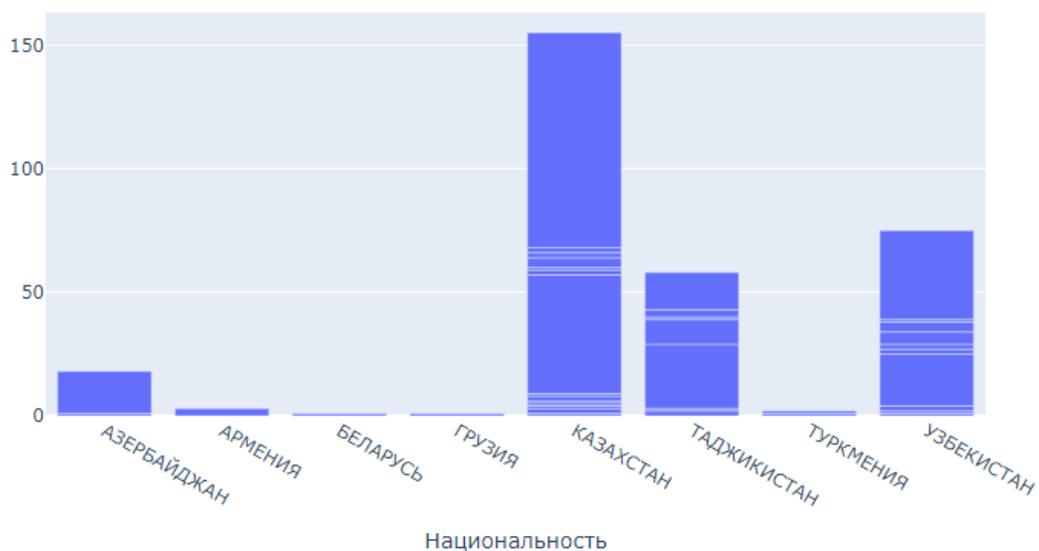


Рис. 2.13. Гистограмма с наростами по двум разрезам

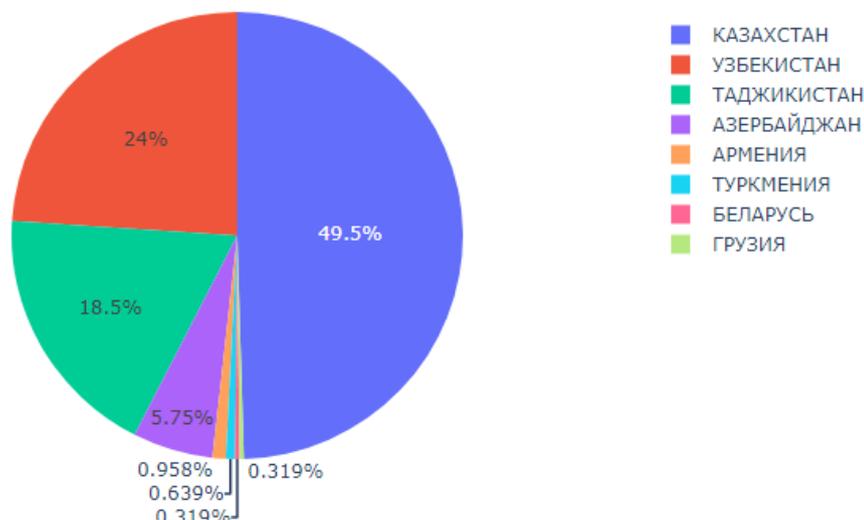


Рис 2.14. Круговая диаграмма по количеству задолженностей в зависимости от гражданства

Предоставляется возможность использовать таблицу с двумя разрезами: "Вид контроля" и "Гражданство". В этой таблице отображается количество должников каждого гражданства по каждому виду контроля, как показано на рисунке 2.11. Для визуализации данных также создаются две диаграммы, которые можно увидеть на 2.13 и 2.14.

Пользователь может выбирать, какие разрезы или какие данные он хочет увидеть с помощью интерфейса выбора колонки, по которой будет происходить группировка. Кроме того, пользователь имеет возможность сохранить таблицы и графики в формате PNG-изображения, как показано на рисунке 2.15. Эти функции позволяют получать разнообразные данные для отслеживания наихудших и лучших направлений обучения.

Средства сохранения изображения в виде графического файла являются встроенными в библиотеку plotly.

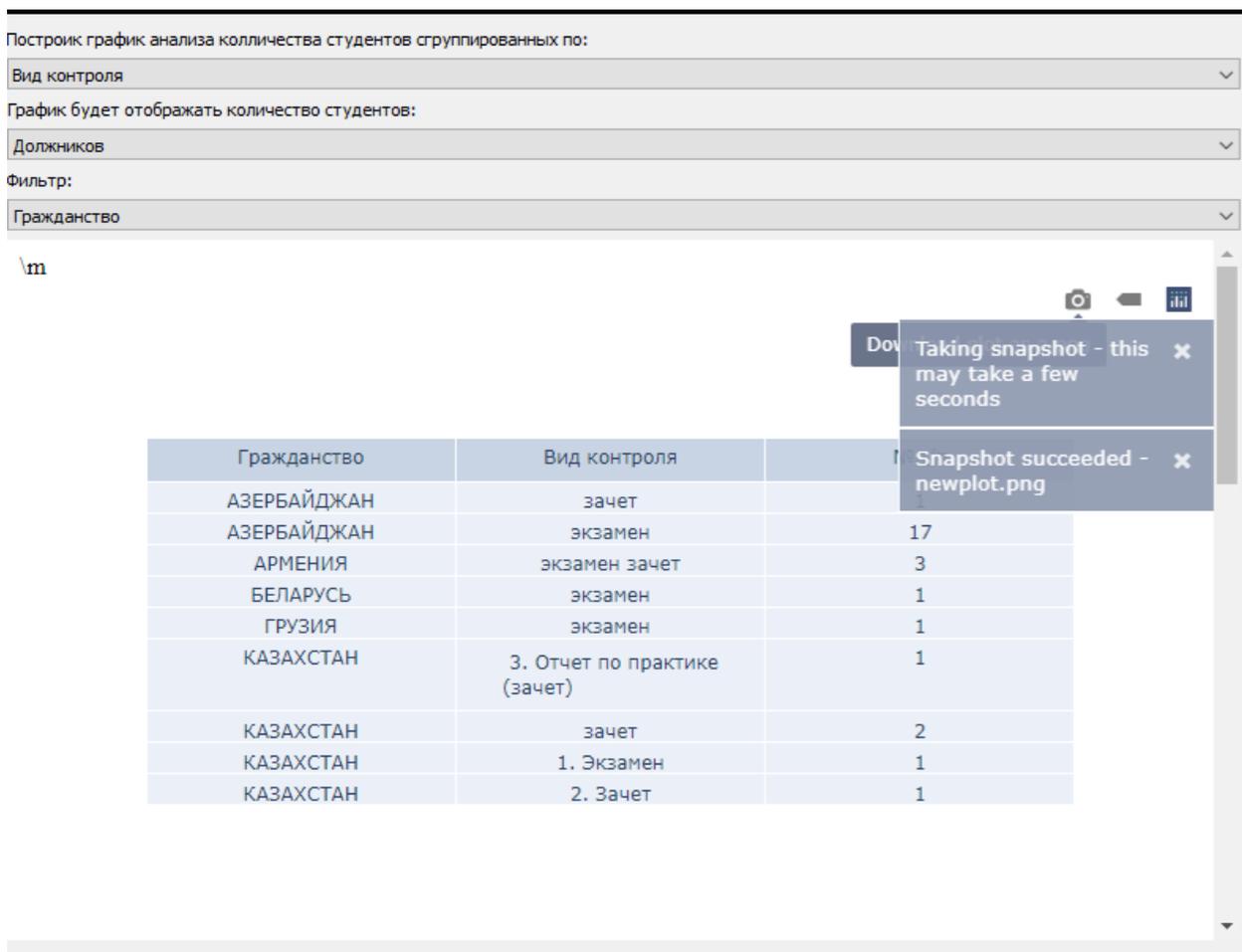


Рис 2.15. Возможность сохранения табличных или графических данных в виде файла формата PNG.

С помощью полученных данных можно определить, какие студенты определенного гражданства имеют наибольшее количество долгов. Также можно выявить, какие направления чаще всего вызывают проблемы у учащихся. Важной информацией является возможность фильтровать учащихся по наличию неудовлетворительных оценок. Это позволяет сравнивать показатели успеваемости на различных направлениях и определить, где учащиеся лучше всего справляются с нагрузкой.

Все данные, полученные с помощью приложения, можно сохранить в виде изображений, что позволяет сравнивать показатели в разные периоды времени, например, сравнивать результаты первого и второго семестров. Это позволяет делать выводы о том, какие направления улучшаются, а какие, наоборот,

ухудшаются. Делать выводы о том какие направления улучшаются, а какие наоборот ухудшаются. Такие сравнения помогут выявить тренды, улучшения или ухудшения в успеваемости и предоставят ценную информацию для определения эффективности образовательных программ и направлений обучения. В итоге, приложение предоставляет мощные инструменты для анализа данных об успеваемости студентов, позволяя пользователю получить глубокое понимание о текущем состоянии, а также прогнозировать и принимать решения на основе данных для улучшения образовательного процесса.

ГЛАВА 3. ОРГАНИЗАЦИЯ ПРОВЕДЕНИЯ АНКЕТИРОВАНИЯ ИНОСТРАННЫХ СТУДЕНТОВ ТЮМГУ

Было проведено анкетирование среди студентов вуза "ТюмГУ". Анкета состояла из нескольких вопросов, направленных на определение социально-демографических характеристик и мнений студентов по ряду вопросов. Всего было собрано 42 ответа.

Анкетирование было проведено в период с февраля по марте 2023 года на территории вуза. Анкеты заполнялись анонимно и добровольно. В анкете содержались вопросы о возрасте, поле, факультете, курсе, и мнении студентов по ряду вопросов, связанных с учебным процессом и общественной жизнью вуза.

Собранные данные будут проанализированы для выявления основных тенденций и мнений студентов вуза. Полученные результаты будут использованы для улучшения образовательного процесса и повышения качества жизни студентов на территории вуза.

Вот наш вопросы и результаты анкетирование.

Первый вопрос был связан с полом студентов, которые проходили анкетирование. Ответили 42 студента. Из них 64,3% (27 студентов) указали, что они мужчины, а 35,7% (15 студентов) - женщины.

Ваш пол:

42 ответа

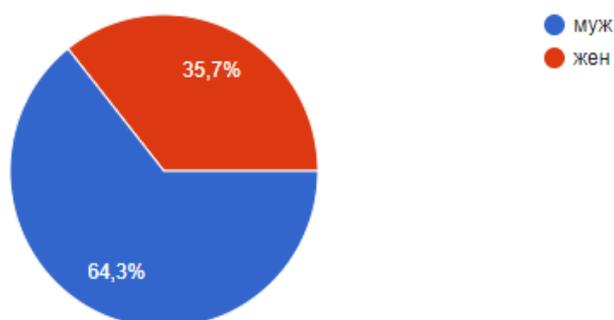


Рис 3.1. Первый вопрос анкетирования

Второй вопрос был связан с институтом, в котором учатся студенты, которые проходили анкетирование. Ответили 42 студента. Из них:

- 28,6% (12 студентов) учатся в Институте биологии ТюмГУ.
- 31% (13 студентов) учатся в Институте математики и компьютерных наук ТюмГУ.
- 9,5% (4 студента) учатся в Институте государства и права ТюмГУ.
- 9,5% (4 студента) учатся в Институте наук о Земле ТюмГУ.
- 7,1% (3 студента) учатся в Институте социально-гуманитарных наук ТюмГУ.
- 7,1% (3 студента) учатся в Финансово-экономическом институте ТюмГУ.
- 4,8% (2 студента) учатся в Институте химии ТюмГУ.
- 2,4% (1 студент) учится в Институте физической культуры ТюмГУ.

Институт:

42 ответа



Рис. 3.2. Второй вопрос анкетирования

Третий вопрос был связан со специальностью (направлением) студентов, которые проходили анкетирование. Всего было получено 42 ответа. Вот перечень специальностей (направлений) и количество студентов, которые выбрали каждую из них:

- Экономист - 4.8% (2 студента)
- МОиАИС - 14.3% (6 студентов)
- Биология - 21.4% (9 студентов)
- Биоинженерия и биоинформатика - 7.1% (3 студента)
- Прикладная информатика - 4.8% (2 студента)
- ИСиТ - 2.4% (1 студент)
- ФХА 22 - 2.4% (1 студент)
- Таможенное дело - 7.1% (3 студента)
- Экономическая безопасность - 2.4% (1 студент)
- Филология - 4.8% (2 студента)
- ИБ - 2.4% (1 студент)
- Туризм - 4.8% (2 студента)
- Экономическая социология - 2.4% (1 студент)
- География - 4.8% (2 студента)
- Менеджмент - 2.4% (1 студент)
- Компьютерная безопасность - 2.4% (1 студент)
- Правовое обеспечение национальной безопасности - 2.4% (1 студент)
- Математика - 2.4% (1 студент)
- Информационные системы и технологии - 2.4% (1 студент)
- ПО: русский язык и литература - 2.4% (1 студент)
- Мехатроника и роботехника - 2.4% (1 студент)
- Химия - 2.4% (1 студент).

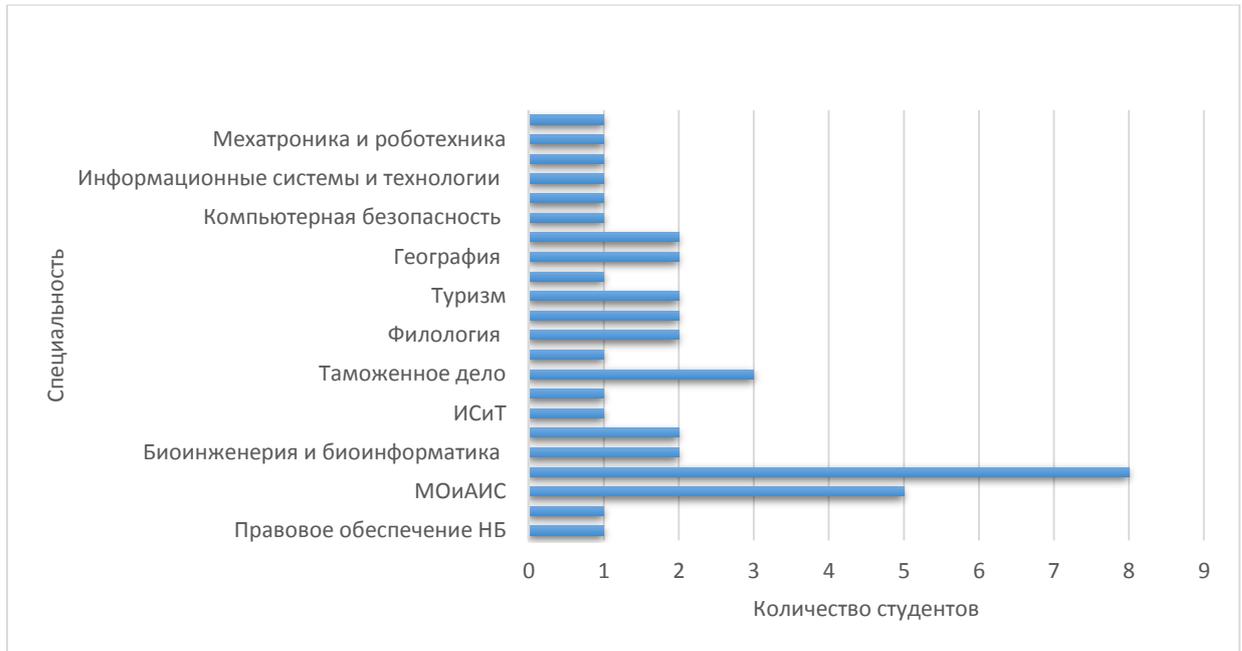


Рис. 3.3. Третий вопрос анкетирования

Из ответов 42 студентов на вопрос "Престижно ли учиться в ТюмГУ?" 88,1% ответили "Да", а 11,9% ответили "Нет". В количественном выражении, это означает, что 37 студентов ответили "Да", а 5 студентов ответили "Нет".

Престижно ли учиться в ТюмГУ?

42 ответа

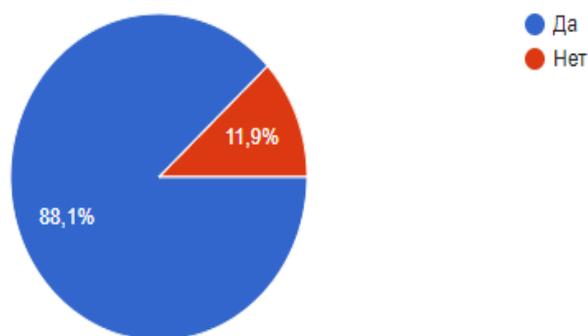


Рис. 3.4. Четвёртый вопрос анкетирования

После тщательного анализа ответов, можно сделать вывод, что основными причинами выбора данного вуза являются высокое качество образования, удобное территориальное расположение и много положительных отзывов. Большинство респондентов (19 человек) указали именно высокое качество образования как основной фактор своего выбора. Однако, также заметно, что многие студенты выбрали университет на основании рекомендаций от знакомых или совета родителей. Важным фактором также является удобное территориальное расположение, что подтверждается 9 ответами, а также привлекательность университета из-за множества положительных отзывов (10 ответов). В целом, можно сделать вывод, что выбор вуза основывается на нескольких факторах, таких как качество образования, рекомендации от знакомых и родителей, а также удобное расположение и репутация университета.

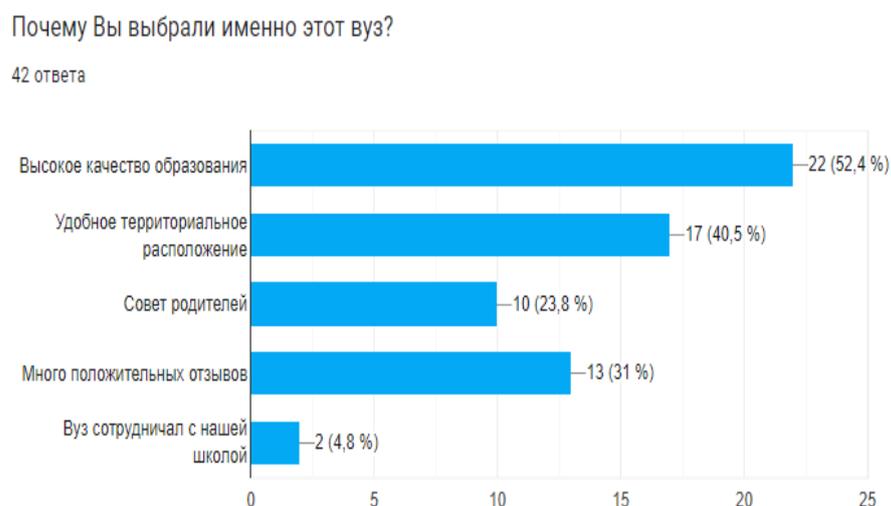


Рис. 3.5. Пятый вопрос анкетирования

После анализа ответов на вопрос "Почему вы выбрали данную специальность?" можно сделать вывод, что большинство людей выбрали свою специальность из-за интереса к будущей профессии и/или мечты с детства, а также возможности заработать хорошие деньги. Меньшее количество людей выбирали специальность из-за престижа в обществе, под влиянием родителей, или чтобы получить отсрочку от армии. Одна из причин выбора специальности была бюджетность обучения.

Почему вы выбрали данную специальность?

42 ответа

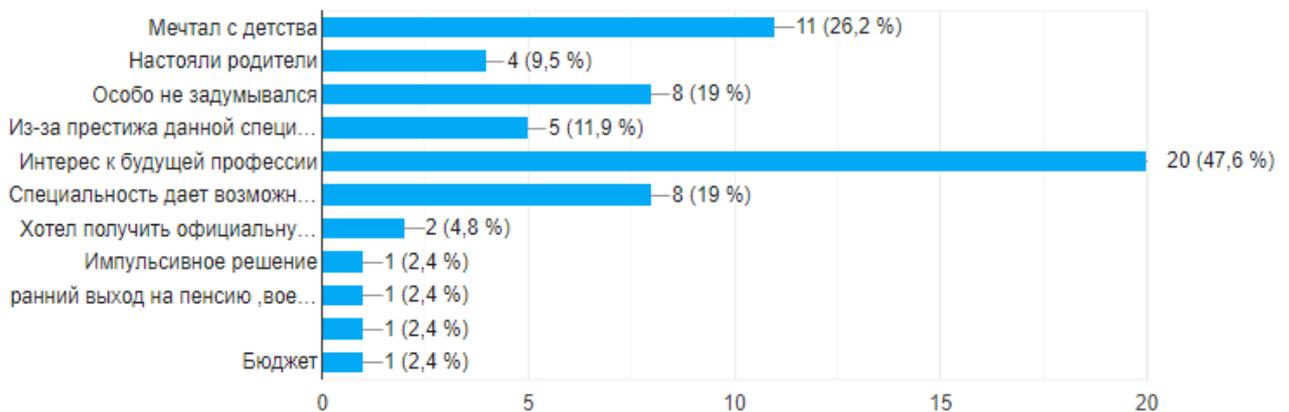


Рис. 3.6. Шестой вопрос анкетирования

Исходя из предоставленных ответов, можно сказать, что большинство опрошенных (57,1%) собираются работать по специальности, 38,1% собираются работать "как получится", а всего лишь 4,8% не собираются работать по своей специальности.

Собираетесь ли вы работать по специальности?

42 ответа

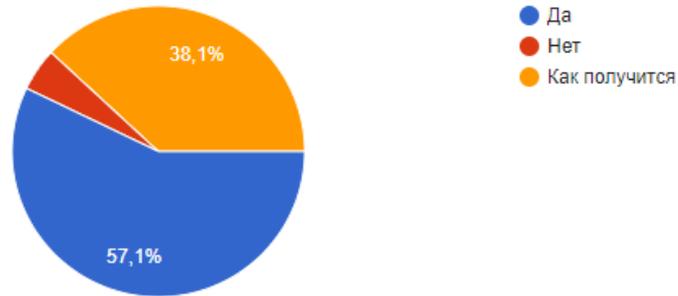


Рис. 3.7. Седьмой вопрос анкетирования

Следующий вопрос - изменилось ли ваше отношение к выбранной специальности после поступления? Судя по ответам, у большинства людей отношение к выбранной специальности осталось без изменений или даже улучшилось после поступления. Однако у некоторых людей отношение к специальности ухудшилось. Возможно, это связано с тем, что реальная работа в профессии отличается от ожиданий или же из-за сложностей, с которыми сталкиваются в процессе обучения. Но в целом, результаты говорят о том, что большинство студентов удовлетворены своим выбором специальности.

Изменилось ли ваше отношение к выбранной специальности после поступления?

42 ответа



Рис. 3.8. Восьмой вопрос анкетирования

Большинство ответов указывают на то, что если бы посещение занятий было необязательным, то большинство студентов посещали бы более 50% занятий, а 50% ответов указывают на то, что студенты могут посещать занятия как менее, так и более чем на половину. Это может зависеть от конкретного курса и преподавателя, а также от личных интересов и мотивации студента.

Если бы посещение занятий было бы необязательным, а по вашему выбору, то какой процент занятий вы бы посещали?

42 ответа

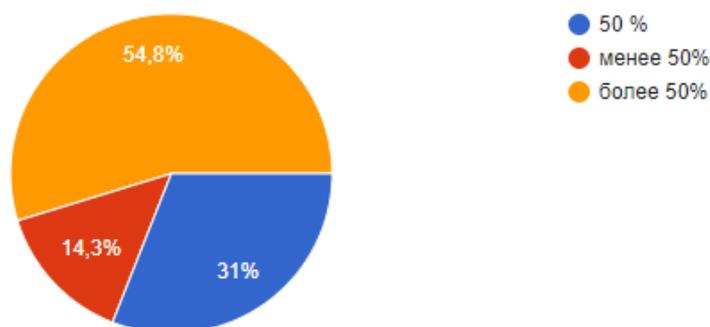


Рис. 3.9. Девятый вопрос анкетирования

Из представленных высказываний можно сделать вывод, что организация учебного процесса на вашем факультете/институте в целом оценивается как на удовлетворительном или высоком уровне. В то же время, есть некоторые замечания относительно несоответствия изучаемых дисциплин получаемой специальности и неудовлетворённости качеством преподавания в целом. Однако, это не отражает общей картины, и многие студенты считают, что организация учебного процесса в целом является на высоком уровне.

Выберите высказывания, характеризующие организацию учебного процесса на вашем факультете (институте)

42 ответа



Рис. 3.10. Десятый вопрос анкетирования

Большинство оценок указывает на то, что отношения "преподаватель - студент" на вашем факультете (институте) находятся на высоком уровне (оценка 4 и выше). Однако есть и несколько оценок на уровне "удовлетворительно" (оценка 3), а также одна оценка на уровне "низко" (оценка 2). В целом, стоит обратить внимание на эти отзывы и попытаться выяснить, в чем заключаются проблемы и что можно улучшить для улучшения отношений между преподавателями и студентами.

Отношения «преподаватель – студент» (в учебном процессе)

42 ответа

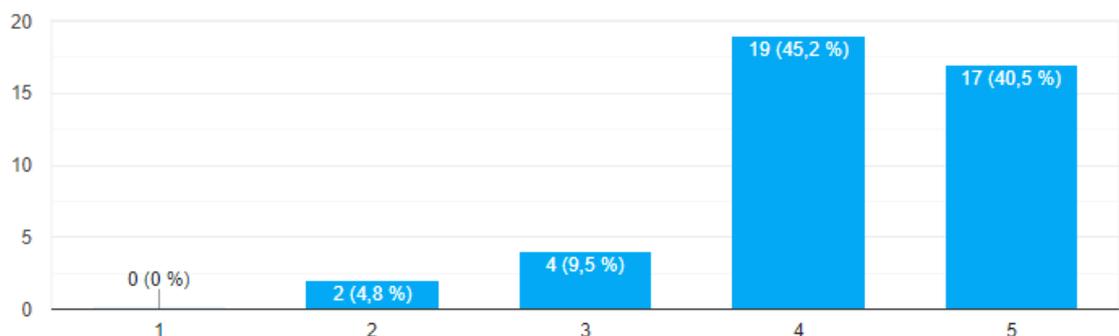


Рис. 3.11. Одиннадцатый вопрос анкетирования

Из ответов можно сделать вывод, что отношения студентов с администрацией (декан, заместители декана, заведующие кафедрами) в целом хорошие. Большинство студентов оценили эти отношения на оценку 4 или 5 из 5, что говорит о высоком уровне удовлетворённости. Тем не менее, есть и те, кто оценил эти отношения на оценку 3, что может указывать на какие-то проблемы в отношениях. Общая картина показывает, что администрация университета и студенты могут работать вместе, чтобы обеспечить наилучшие условия для обучения и участия в университетской жизни.

Отношения «студент – администрация» (декан, заместители декана, заведующие кафедрами)

42 ответа

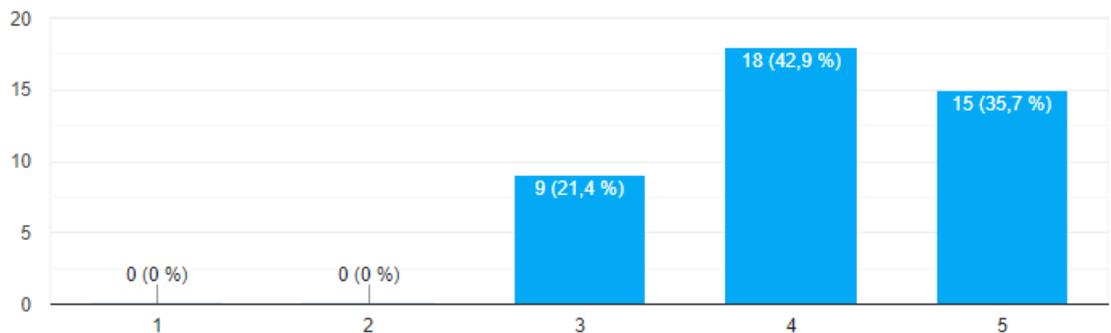


Рис. 3.12. Двенадцатый вопрос анкетирования

Анализ ответов на вопрос об отношениях "студент-сотрудники" показывает, что большинство студентов оценивают эти отношения положительно, со средним баллом 4.28 из 5. Некоторые сотрудники, такие как экономисты и воспитатели в общежитии, получили более высокие оценки, чем другие, такие как кассиры и секретари деканата. Оценки также указывают на то, что некоторые студенты испытывают трудности во взаимодействии с некоторыми сотрудниками, такими как вахтёры и секретари деканата. В целом, однако, ответы указывают на то, что отношения между студентами и сотрудниками являются достаточно хорошими.

Отношения «студент – сотрудники» (кассир, бухгалтер, экономист, вахтёр, воспитатель в общежитии, секретарь деканата и т.д.)

42 ответа

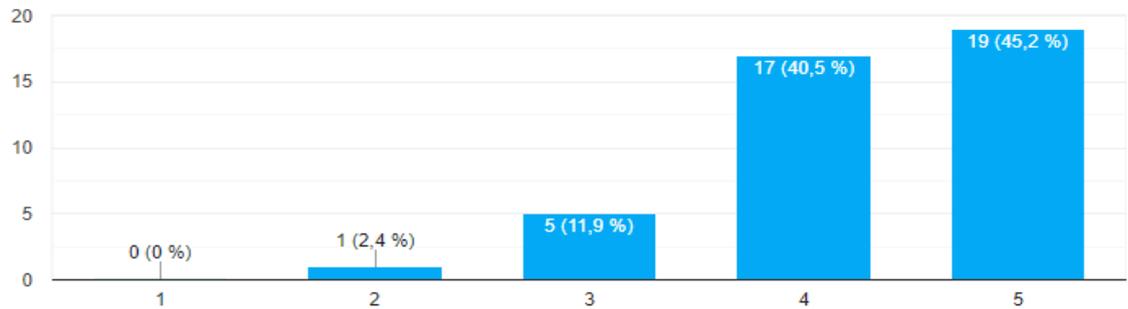


Рис. 3.13. Тринадцатый вопрос анкетирования

Анализ ответов на вопрос об отношениях между студентами показывает, что большинство респондентов имеют положительное отношение к своим однокурсникам. Большинство ответов составляют оценки 4 и 5 из возможных 5. Также заметно, что есть небольшое количество ответов со средними оценками, что может указывать на некоторую неудовлетворенность или недоверие к определенным группам студентов. В целом, можно сделать вывод, что отношения между студентами воспринимаются как положительные и дружеские.

Отношения между студентами

42 ответа

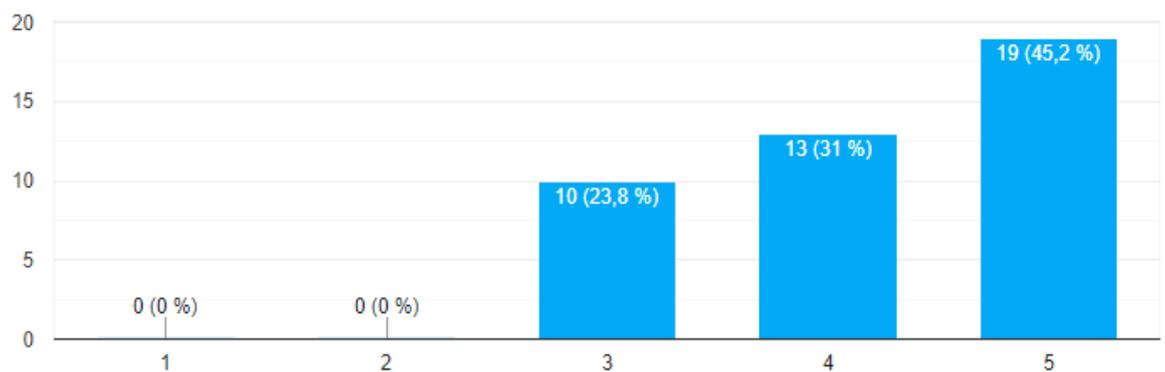


Рис. 3.14. Четырнадцатый вопрос анкетирования

Из ответов видно, что в группе есть кураторы у 18 студентов, нет кураторов у 13 студентов, и 10 человек затруднились ответить.

Есть ли в вашей студенческой группе куратор?

42 ответа

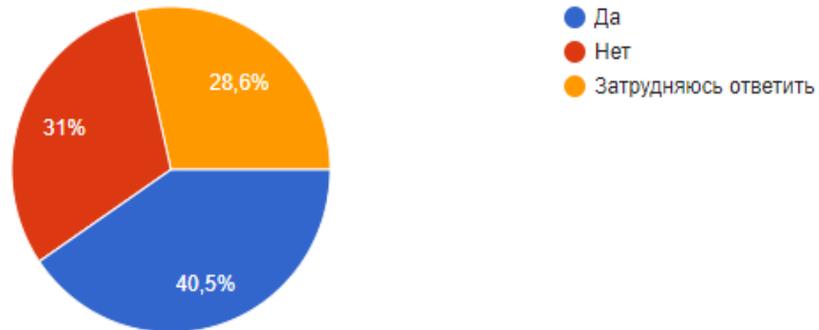


Рис. 3.15. Пятнадцатый вопрос анкетирования

Судя по ответам, большинство студентов удовлетворены работой куратора своей студенческой группы, но также есть и те, кто ответил, что удовлетворен частично или неудовлетворен. Важно узнать причины неудовлетворенности и принять меры для улучшения работы куратора.

Удовлетворенность работой куратора студенческой группы

42 ответа

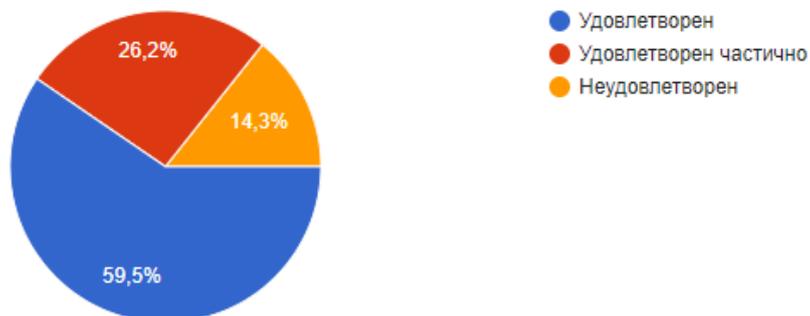


Рис. 3.16. Шестнадцатый вопрос анкетирования

Из предоставленных ответов можно сделать вывод, что большинство опрошенных имеют какую-то информацию о коллективах художественной самодеятельности, научных кружках и молодежных клубах по интересам в университете. Однако, не все опрошенные владеют этой информацией полностью. Некоторые люди не имеют никакой информации о таких коллективах и кружках.

Поэтому, можно предположить, что информация о таких организациях не всегда является достаточно доступной для студентов, или же университет не оказывает должной поддержки таким сообществам. В целом, стоит продолжать работу по повышению информированности студентов об этих организациях и их возможностях.

Владеете ли Вы информацией о существующих в университете коллективах художественной самодеятельности, научных кружках, молодежных клубах по интересам?

42 ответа

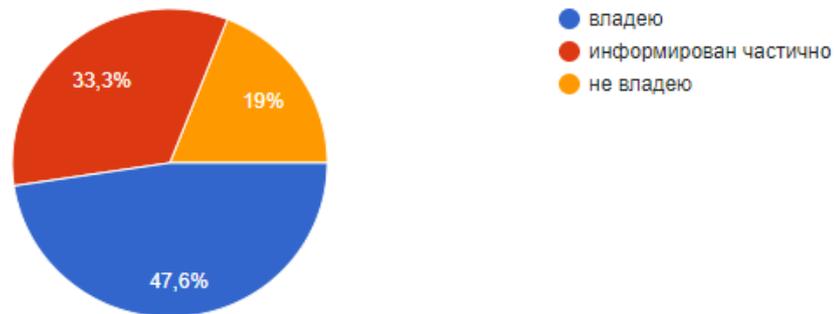


Рис. 3.16. Семнадцатый вопрос анкетирование

Из последнего вопроса видно, что большинство студентов не принимают участия во вне учебной общественной работе, но некоторые участвуют в организации различных мероприятий, таких как праздники, вечера, дискотеки и спортивные соревнования. Также некоторые студенты принимают участие в политических акциях и митингах, волонтерской деятельности, а также работе в студенческом совете. Большинство студентов не участвуют в спортивных соревнованиях и не принимают участия в политических мероприятиях.

Принимаете ли Вы участие во вне учебной, общественной работе?

42 ответа

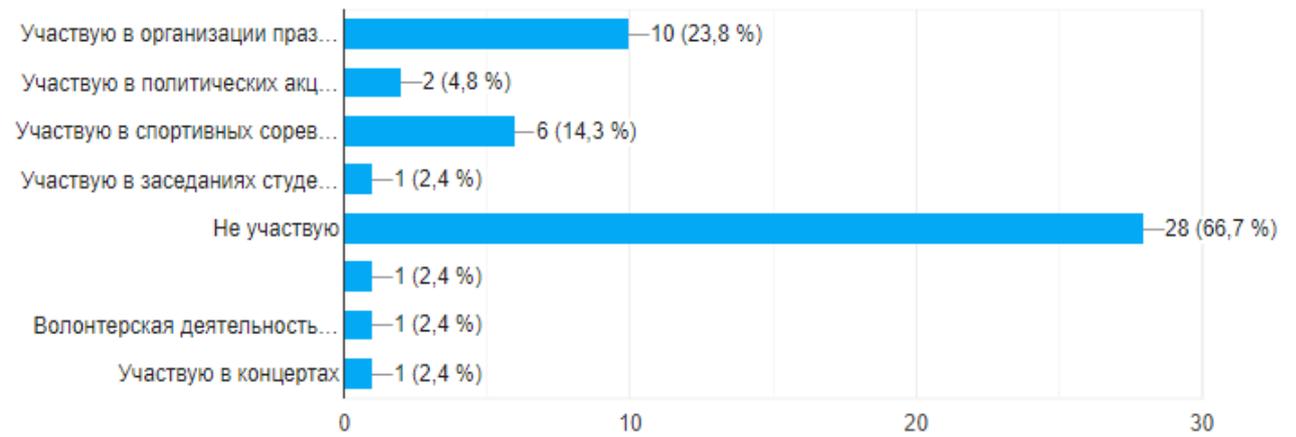


Рис. 3.16. Восемнадцатый вопрос анкетирования

ЗАКЛЮЧЕНИЕ

В рамках выполнения выпускной квалификационной работы были успешно решены следующие задачи:

1. Организация загрузки данных из Excel-таблицы: был разработан и реализован механизм для эффективной загрузки данных из Excel-файлов, позволяющий удобно и быстро импортировать информацию в нашу систему.
2. Предварительная обработка данных: важным этапом работы было проведение предварительной обработки данных. Мы учли наличие пропусков, осуществили выделение экземпляров сущности и проверили корректность данных, применяя соответствующие алгоритмы и методы.
3. Вычисление видов показателей для студентов: в рамках разработки были выбраны и реализованы различные виды показателей, позволяющие оценить успеваемость и прогресс студентов. Были учтены требования и особенности нашей системы образования.
4. Возможности выбора правил фильтрации: для облегчения работы с данными была создана возможность выбора правил фильтрации для отбора студентов. Это позволяет пользователям нашей системы гибко настраивать параметры отображения данных в соответствии с их потребностями.
5. Представление данных: мы успешно реализовали функционал для представления данных в текстовом и графическом форматах. Теперь пользователи могут в удобном виде просматривать и анализировать информацию о студентах и их результаты.
6. Обработка результатов анкетирования студентов: важным аспектом нашей работы была обработка результатов анкетирования студентов с привязкой к их атрибутам. Мы разработали и внедрили механизм, который позволяет

эффективно анализировать полученные данные и выявлять паттерны и тенденции.

В результате успешной реализации всех поставленных задач в рамках ВКР, мы создали систему, способную эффективно управлять данными о студентах, обрабатывать и анализировать результаты и предоставлять пользователю удобный интерфейс для работы с информацией. Разработанные функциональные возможности значительно облегчат процесс анализа и мониторинга успеваемости студентов, а также помогут выявлять их потенциал и оптимизировать образовательный процесс.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Ключникова Е.В. Проблемы адаптации иностранных студентов в России // Вестник Тверского государственного университета. Серия: Педагогика и психология. 2018. № 1. С. 133–140
2. Python 3.11.4 documentation // Сайт разработчиков Python Software Foundation [Электронный ресурс]. URL: <https://docs.python.org/3/> (Дата обращения: 20.05.2023).
3. SQLite documentation// Сайт поддержка SQLite консорциума разработчиков SQLite [Электронный ресурс]. URL: <https://www3.sqlite.org/docs.html> (Дата обращения: 17.05.2023).
4. Чернышев, С. А. Основы программирования на Python : учебное пособие для вузов / С. А. Чернышев. — Москва : Издательство Юрайт, 2021 — 286 с.
5. Абдрахманов М.И. Devpractice Team. Pandas. Работа с данными. 2-е изд. / Абдрахманов М.И.- Изд-во devpractice.ru. 2020. – 170 с.
6. 4. Абдрахманов М.И. Devpractice Team. Python. Визуализация данных. Matplotlib. Seaborn. Mayavi. / Абдрахманов М.И.- Изд-во devpractice.ru. 2020. – 412 с.
7. Букунов С. В. Разработка приложений с графическим пользовательским интерфейсом на языке Python : учебное пособие для СПО / С. В. Букунов, О. В. Букунова. — Санкт-Петербург : Лань, 2023. — 88 с.

Текст кода класса GraphWindow(QMainWindow), представляющего окно программы для отображения графиков.

```
class GraphWindow(QMainWindow):
    def __init__(self, plot_html):
        super().__init__()
        self.setWindowTitle("Окно графиков и таблиц")
        self.setGeometry(100, 100, 800, 600)

        self.layout = QVBoxLayout()
        central_widget = QWidget(self)
        central_widget.setLayout(self.layout)
        self.setCentralWidget(central_widget)

        self.x_axis_label_1 = QLabel("Построик график анализа количества
            студентов сгруппированных по:")
        self.x_axis_combo_1 = QComboBox()
        self.x_axis_combo_1.addItem(CONTROL_TYPE)
        self.x_axis_combo_1.addItem(COURSE)

        self.y_axis_label_1 = QLabel("График будет отображать количество
            студентов:")
        self.y_axis_combo_1 = QComboBox()
        self.y_axis_combo_1.addItem("Должников")
        self.y_axis_combo_1.addItem("Всех")

        self.filter_label_1 = QLabel("Фильтр:")
        self.filter_combo_1 = QComboBox()
        self.filter_combo_1.addItem(CITIZENSHIP)
        self.filter_combo_1.addItem(ED_LEVEL)

        self.plot_button = QPushButton("Построить график")
        # self.plot_button.clicked.connect(self.plot_graph)

        # Создание компоновщика и добавление элементов управления
        self.layout.addWidget(self.x_axis_label_1)
        self.layout.addWidget(self.x_axis_combo_1)
        self.layout.addWidget(self.y_axis_label_1)
        self.layout.addWidget(self.y_axis_combo_1)
        self.layout.addWidget(self.filter_label_1)
        self.layout.addWidget(self.filter_combo_1)
        self.x_axis_combo_1.currentIndexChanged.connect
            (self.update_graph)
        self.y_axis_combo_1.currentIndexChanged.connect
            (self.update_graph)
        self.filter_combo_1.currentIndexChanged.connect
            (self.update_graph)
```

```

self.webview = QWebEngineView()
self.webview.setHtml(plot_html)

self.layout.addWidget(self.webview)
def update_graph(self):
    x_axis = self.x_axis_combo_1.currentText()
    y_axis = self.y_axis_combo_1.currentText()
    filtered_by = self.filter_combo_1.currentText()

    # Update the graph based on the selected options
    # ...
    if y_axis == "Должников":
        filtered_df = displayed_df[displayed_df
            ['Состояние долга'] == 'Есть']
    else:
        filtered_df = displayed_df

    filtered_df = displayed_df.groupby([x_axis, filtered_by])
        [NUM].count().reset_index()
    fig = go.Figure(data=[go.Bar(x=filtered_df[x_axis],
        y=filtered_df[NUM])])
    fig_html = plot(fig, include_plotlyjs=False, output_type="div")
    fig3 = go.Figure(data=[go.Pie(labels=filtered_df[x_axis],
        values=filtered_df[NUM])])
    fig_html3 = plot(fig3, include_plotlyjs=False, output_type="div")
    table = go.Figure(data=[go.Table(
        header=dict(values=filtered_df.columns),
        cells=dict(values=[filtered_df[column] for column in
            filtered_df.columns]))
    ])
    fig_table = plot(table, include_plotlyjs=False, output_type='div')

    html_header = '<script src="https://cdn.plot.ly/plotly-
        latest.min.js"></script>'
    combined_html = f"{html_header}\n{fig_table}\n{fig_html}\n{fig_html3}"
    # Update the HTML content and reload the webview
    self.webview.setHtml(combined_html)

```