

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ	5
1.1 Общее описание предметной области	5
1.2 Текущие решения в области автоматизации расписания	7
2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ	10
2.1 Система модуля и его функции	10
2.2 Требования к функциям, выполняемым системой	11
2.3 План тестирования.....	13
3 ОПИСАНИЕ СРЕДЫ РАЗРАБОТКИ	15
3.1 Выбор и обоснование программных инструментов	15
3.1.1 Выбор программных инструментов	15
3.1.2 Обоснование выбора инструментов	16
3.2 Разработка программного модуля	16
3.2.1 Реализация пользовательского интерфейса программы	17
3.2.2 Описание функциональных узлов	17
3.2.3 Результат работы и тестирования	18
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	22
ПРИЛОЖЕНИЕ 1 Диаграммы	24
ПРИЛОЖЕНИЕ 2 Тест-кейсы	25
ПРИЛОЖЕНИЕ 3 Интерфейс	4
ПРИЛОЖЕНИЕ 4 Функции кода	6

ВВЕДЕНИЕ

В современном образовательном процессе, характеризующемся высокой динамичностью и разнообразием форм обучения, эффективное управление временем и задачами становится одной из ключевых задач как для студентов, так и для преподавателей. Одним из основных аспектов получения прибыли является четко организованное расписание занятий. «Расписание занятий студента» представляет собой важный программный модуль, требующий упрощения и оптимизации процесса планирования и управления учебным временем.

Актуальность разработки обусловлена несколькими факторами. Во-первых, с определением числа теоретических дисциплин и разнообразия форм обучения, таких как лекции, семинары, лабораторные занятия, увеличение сложности в организации расписания. Студенты часто сталкиваются с проблемами, связанными с трудностями, нарушениями с несовпадением расписания, отсутствием информации о времени и месте, что может отрицательно сказаться процессе их обучения. Во-вторых, преподавателям также необходимы удобные инструменты для планирования своих занятий, которые позволяют им более эффективно экономить свое время и ресурсы.

Целью курсового проекта является создание функционального и удобного программного модуля, который будет интегрирован в существующую образовательную платформу. В рамках проекта предусмотрена разработка системы автоматического составления расписания, включающая функционал мгновенной генерации на всех зарегистрированных пользователей, а также дальнейшее его грамотное отображение.

Таким образом, разработка программной модуляционной системы «Расписание занятий студента» представляет собой важный шаг в направлении оптимизации текущего процесса и повышения его эффективности, что делает эту тему актуальной и инновационной для современного образования.

Для решения поставленной цели необходимо выполнить следующие задачи:

- 1) провести теоретический анализ текущих программ в области построения расписаний в образовательных учреждениях;
- 2) обосновать выбор инструментов и средств разработки программного модуля;
- 3) разработать архитектуру системы, включающую интерфейс пользователя и алгоритм формирования расписания;
- 4) реализовать функциональность программного модуля для создания и редактирования расписаний с учетом ограничений и требований;
- 5) провести обновление системы, включая тестирование разработки-кейсов и проверку работоспособности всех функций.

Объектом исследования является составление процессов и планирование управления в образовательных учреждениях.

Предметом исследования является разработка программного обеспечения для автоматизации этого процесса с учетом требований пользователей (студентов, преподавателей и администрации).

В первой части курсового проекта будет рассмотрено теоретическое обоснование проблем, включая подходы к автоматизации составления расписания, а также программное обеспечение анализа, уже примененного в этой области.

Во второй части проекта будет продемонстрирована функциональность кода, включая разработку и описание подключенной к системе базы данных.

В остальной части проекта будет проведено тестирование системы. Для этого будут разработаны тест-кейсы, проведены их проверки и анализ результатов с целью выявления возможных ошибок и улучшений.

В заключении будет подведен итог проделанной работы, а также предложены рекомендации по совершенствованию системы.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ

1.1 Общее описание предметной области

Современные технологии проникли во все сферы человеческой жизни, образование не является исключением. На фоне всех этих технических инноваций, прочно закрепившихся в жизни людей, традиционное расписание на доске объявлений расположенной недалеко от входа в образовательную организацию уже давно устарело. Каждая современная образовательная организация зачастую имеет своё информационно образовательное пространство. На фоне этого невозможность узнать своё расписание с применением современных гаджетов вызывает достаточно сильный дискомфорт.

В данном курсовом проекте предметной областью для решения поставленных задач является система автоматизации для образовательных учреждений, направленная на управление учебным процессом, а именно — составление расписания занятий для студентов и преподавателей. Это сложная задача, которая требует системы генерации планирования учебных дисциплин, распределение аудиторий и преподавателей с возможностью редактирования. Первоначально такие программы были предназначены для расчёта простых вариантов расписания, но со временем они развивались и становились более функциональными

Каждое учебное заведение стремится создать максимально эффективную организацию учебного процесса, чтобы обеспечить наилучшие условия для студентов и преподавателей. Преимущества автоматизации — это ускорение выполнения операций и снижение ошибок при их выполнении, снижение издержек на реализацию операций и повышение качества. Для успешного составления расписания необходимо учесть все эти факторы, что без автоматизации становится крайне сложным и время затратным процессом.

Для эффективной автоматизации процесса система должна обеспечить:

- автоматическое распределение учебных дисциплин по времени;
- оптимизацию использования аудиторий и ресурсов;
- учет предпочтений преподавателей и студентов;
- генерацию расписания с учетом всех ограничений и требований;
- автоматическое перераспределение занятий в случае совпадений;
- сохранение расписания в Excel;
- возможность внесения правок в расписании.

Система будет иметь удобный интерфейс для администратора, позволяющий на основе введенных данных о студентах, преподавателях и предметах автоматически сгенерировать оптимальное расписание, а также обеспечивать возможность корректировки в случае необходимости.

Состав предметной области:

- 1) Учебные дисциплины. Включают перечень всех дисциплин, которые преподаватели должны вести в течение семестра, с указанием длительности лекций, практических занятий и лабораторных работ;
- 2) Студенты и группы. Студенты распределены по учебным группам, каждая из которых занимается определенным набором дисциплин. Для успешного составления расписания необходимо точно учитывать количество студентов в каждой группе, а также их привязку к определенным дисциплинам. Это позволяет избежать конфликтов в расписании и гарантирует, что все студенты получат необходимое количество учебных часов по каждой дисциплине;
- 3) Преподаватели. Каждый преподаватель имеет свою специализацию и ограниченное количество рабочих часов в неделю. Система должна учитывать доступность преподавателей, их предпочтения по времени занятий, а также возможность проведения нескольких занятий в один день;
- 4) Аудитории. Они играют ключевую роль в процессе составления расписания. Каждая аудитория имеет определенную вместимость, а также оснащение для проведения тех или иных занятий. Система

должна эффективно использовать эти ресурсы, обеспечивая оптимальное распределение занятий по аудиториям. Интерфейс пользователя должен быть интуитивно понятным и простым в использовании. Его задача обеспечить возможность просмотра расписания по группам студентов, преподавателям удобным способом. Модель жизненного цикла программного продукта.

Разработка технического задания состоит из сбора сведений, их обработки и описания всех функциональных требований, ограничений и архитектуры системы.

1.2 Текущие решения в области автоматизации расписания

На сегодняшний день в сфере образования существует несколько решений для автоматизации составления расписания. Эти решения могут быть как специализированными программами, так и модулями в рамках более крупных образовательных платформ.

Сейчас представлено большое разнообразие электронных систем в этой предметной области, которые направлены на создание расписания занятий для студентов и преподавателей, внесение изменений и уменьшение затрат времени и ресурсов.

Все они должны следовать законодательным аспектам электронных систем в данной области:

- 1) Единая система программной документации (ЕСПД):
ЕСПД — это система, которая стандартизирует документацию, связанную с разработкой программного обеспечения. Она включает в себя требования к оформлению и содержанию документации, что позволяет обеспечить единообразие и качество документации в процессе разработки. [1]
- 2) ГОСТ Р ИСО/МЭК 12207-2016 "Системы и программное обеспечение. Процессы жизненного цикла программного обеспечения": Этот стандарт описывает процессы, связанные с

жизненным циклом программного обеспечения, включая его разработку, эксплуатацию и сопровождение. Он устанавливает общие требования к процессам, которые должны быть учтены при разработке программного обеспечения; [2]

- 3) Федеральный закон от 6 июля 2016 года № 187-ФЗ "О безопасности критической информационной инфраструктуры Российской Федерации": Этот закон направлен на защиту критической информационной инфраструктуры, которая включает в себя системы, обеспечивающие жизнедеятельность общества и безопасность государства. Он устанавливает требования к безопасности таких систем и определяет полномочия государственных органов в этой области; [3]
- 4) Федеральный закон от 27 июля 2006 года № 152-ФЗ. «О персональных данных». Закон регулирует обработку персональных данных, устанавливает права субъектов персональных данных и обязанности операторов, которые обрабатывают такие данные. Он также определяет условия, при которых возможен сбор и обработка персональных данных, а также меры по их защите; [4]
- 5) Федеральный закон от 27 июля 2006 года № 149-ФЗ "Об информации, информационных технологиях и о защите информации": Этот закон регулирует отношения, связанные с информацией, информационными системами и технологиями. Он определяет права и обязанности субъектов в области обработки и защиты информации, а также устанавливает требования к безопасности информации и защите персональных данных; [5]

Для создания удобного модуля нужно проанализировать уже существующие предложения.

Moodle — система управления обучением, которая также включает модули для составления расписания, но фокусируется больше на

организационном аспекте обучения, чем на оптимизации распределения времени и ресурсов.

Google Calendar — простое и бесплатное решение, которое может использоваться для составления расписаний, но не имеет продвинутых функций для учета ограничений.

КлассИнфо — сервис, позволяющий составлять расписание занятий, вести журнал и дневник. Но он подходит больше для расписания в школах.

Free Timetabling Software — это бесплатное решение для составления расписаний, которое учитывает большое количество переменных, таких как типы занятий, группы студентов, доступность аудиторий и предпочтения преподавателей. Однако интерфейс этой программы сложен, а также она требует значительных усилий для настройки.

Timetabler — коммерческая платформа, которая позволяет автоматизировать процесс составления расписания с учетом множества параметров, таких как предпочтения преподавателей и студенческих групп, однако эта система также обладает сложным интерфейсом и высокой ценой.

Несмотря на существование этих решений, многие из них либо недостаточно гибки для различных типов учебных заведений, либо требуют значительных усилий для настройки и адаптации под специфические условия. Важно отметить, что все эти системы обычно имеют ограничения в плане учета индивидуальных предпочтений студентов и преподавателей, а также могут не эффективно управлять аудиторным фондом учебных заведений.

Проблемы, связанные с текущими решениями:

- 1) сложности в интеграции;
- 2) низкая автоматизация;
- 3) ограниченные функции для анализа данных.

2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ

2.1 Система модуля и его функции

Система управления учебным расписанием «Расписание занятий студента» разработана для комплексной автоматизации процесса управления расписанием учебных занятий, улучшения доступности информации о расписании для студентов и, как следствие, повышения общей эффективности учебного процесса в образовательном учреждении. Система обеспечивает удобный и оперативный доступ к актуальным данным о расписании, предоставляя пользователям интуитивно понятный интерфейс и возможности для эффективного планирования своего времени.

Основные преимущества и выгоды от внедрения системы «Расписание занятий студента» заключаются в следующем:

Система позволяет студентам с лёгкостью изучать свои учебные планы, что способствует значительно более эффективной организации учебного времени. Это, в свою очередь, снижает уровень стресса, связанного с организацией учебного процесса, и способствует лучшему усвоению материала.

Обеспечение постоянного доступа к актуальной информации. Студенты и преподаватели получают круглосуточный доступ к точной и обновленной информации о своих занятиях. Исключается необходимость уточнения расписания у преподавателей или административного персонала, что экономит значительное количество времени как для студентов, так и для сотрудников учебного заведения.

Повышение эффективности работы административного персонала. Автоматизация процесса управления расписанием освобождает административный персонал от рутинной работы по составлению и обновлению расписаний, что позволяет им сосредоточиться на более важных задачах, связанных с управлением учебным процессом.

Её использование способствует не только улучшению организации учебного процесса, но и повышению уровня удовлетворенности как студентов, так и преподавателей, благодаря удобству использования и постоянному доступу к актуальной информации о расписании занятий. В результате, внедрение системы способствует созданию более прозрачной и эффективной образовательной среды, что положительно сказывается на качестве образования в целом. Простой и интуитивно понятный интерфейс делает просмотр и использование расписания максимально удобным и быстрым процессом, что в свою очередь способствует созданию позитивного опыта использования системы и повышению мотивации студентов к обучению.

2.2 Требования к функциям, выполняемым системой

После анализа были определены следующие функции, реализация которых предстоит в системе «Расписание занятий студента»:

- 1) автоматическое эффективное распределение учебного графика;
- 2) добавление и редактирование предметов;
- 3) интуитивно понятный пользовательский интерфейс;
- 4) просмотр расписания любой группы;
- 5) сохранение расписания в Excel таблице;
- 6) регистрация и авторизация аккаунтов студентов, преподавателей и администраторов;
- 7) хранение информации о группах, предметах, преподавателях и аудиториях в базе данных.

Для начала определим основные объекты системы. Они представлены в Таблице 1.

Таблица 1. Основные объекты

№	Объект системы	Краткое описание
1	Пользователь	Человек, зарегистрированный в системе.
2	Предмет	Дисциплина, которая входит в учебный план, с указанием ее названия, кода, и аудиторий.

3	Расписание	График занятий с указанием времени, предмета, аудитории, преподавателя и задействованных групп.
---	------------	---

Каждому пользователю системы присваивается уникальный идентификатор (id). Пользователь может быть студентом, преподавателем или администратором, что определяется атрибутом role. Студенту также указывается его группа, специальность и курс обучения. Атрибут avatar используется для хранения изображения профиля. Атрибуты представлены в Таблице 2.

Таблица 2. Атрибуты сущности «Пользователь»

Атрибут	Тип	Описание
Id	INTEGER PRIMARY KEY	Уникальный идентификатор пользователя
real_name	TEXT	Полное имя пользователя
username	TEXT	Логин для авторизации
password	TEXT	Пароль пользователя
role	TEXT	Роль пользователя
group_name	TEXT	Название группы студента
specialty	TEXT	Специальность студента
course	INTEGER	Курс обучения студента
subjects	TEXT	Список преподаваемых предметов
birthday	DATE	Дата рождения
email	TEXT	Электронная почта
avatar	TEXT	Аватар пользователя

Каждый предмет имеет уникальный идентификатор (id), название, соответствующую специальность, курс и аудиторию, где проводятся занятия. Подробнее в Таблице 3.

Таблица 3. Атрибуты сущности «Предмет»

Атрибут	Тип	Описание
Id	INTEGER PRIMARY KEY	Уникальный идентификатор предмета
name	TEXT	Название предмета
specialty	TEXT	Специальность, для которой предназначен предмет
course	INTEGER	Курс обучения, на котором преподается предмет
classroom	TEXT	Аудитория, где проводятся занятия по предмету

Расписание занятий содержит информацию о времени и месте проведения занятий, а также сведения о группе студентов, преподавателе и предмете. Эти данные отражены в Таблице 4.

Таблица 4. Атрибуты сущности «Расписание»

Атрибут	Тип	Описание
day_of_week	TEXT	День недели, в который проводится занятие
time	TEXT	Время начала занятия
subject_id	INTEGER	Уникальный идентификатор предмета
subject_name	TEXT	Название предмета
group_name	TEXT	Группа студентов, посещающая занятие
teacher_id	INTEGER	Уникальный идентификатор преподавателя
teacher_name	TEXT	Имя преподавателя, ведущего занятие
week_type	TEXT	Тип недели (например, четная/нечетная)
classroom	TEXT	Аудитория, где проводится занятие

На основе выявленных сущностей представим ER-диаграмму базы данных «Расписание занятий студента». ER-диаграмма модели приведена в Приложении 1, Рисунок 1

Также в Приложении 1, Рисунок 2 представлена диаграмма сценариев. Она графическим способом описывает функциональное назначение системы.

2.3 План тестирования

Целью тестирования программного модуля «Расписание занятий студента» является обеспечение его корректной работы, высокой производительности и удобства использования. Это позволит обнаружить и устранить возможные ошибки и проблемы до ввода системы в эксплуатацию, что гарантирует её надежность и удобство для студентов, преподавателей и администраторов.

Объекты тестирования включают ключевые элементы модуля:

- Создание пользователя.
- Авторизация пользователя.
- Просмотр расписания занятий.

Таким образом, в данной главе рассмотрены цели, задачи и особенности системы «Расписание занятий студента», направленной на автоматизацию процессов управления расписанием и упрощение доступа к информации о занятиях.

Выделены основные преимущества внедрения системы, включая:

- Упрощение процесса планирования.
- Повышение доступности информации.
- Анализ учебной нагрузки.

Определены ключевые функции системы, такие как:

- Хранение данных о студентах, преподавателях и занятиях.
- Регистрация и авторизация пользователей.
- Реализация интуитивно понятного интерфейса.

Было описано устройство базы данных, включая сущности и их атрибуты, что обеспечивает понимание структуры и функциональных возможностей системы.

Разработанный план тестирования направлен на проверку функциональности и удобства системы, что позволяет устранить возможные ошибки на этапе разработки. Это гарантирует стабильную работу модуля после запуска и обеспечивает высокое качество обслуживания пользователей.

3 ОПИСАНИЕ СРЕДЫ РАЗРАБОТКИ

3.1 Выбор и обоснование программных инструментов

Программный модуль "Расписание занятий студентов" был реализован с использованием современных инструментов разработки, что обеспечило удобство работы, простоту в управлении данными и расширяемость системы.

3.1.1 Выбор программных инструментов

Для реализации системы были выбраны следующие инструменты и библиотеки:

- PyCharm: Среда разработки (IDE), которая предоставляет интеллектуальный редактор кода, встроенную отладку, поддержку виртуальных окружений и интеграцию с системами контроля версий. Эти возможности ускоряют процесс разработки и упрощают управление проектом.
- Tkinter: Библиотека для создания графического пользовательского интерфейса (GUI) на Python. Tkinter используется для создания окон, кнопок, текстовых полей и других элементов интерфейса, что делает приложение удобным и интуитивно понятным для пользователей.
- JSON: Формат для хранения данных, который предоставляет высокую читаемость, удобство интеграции и простоту работы с небольшими объемами данных. Использование JSON в качестве базы данных исключает необходимость установки реляционных СУБД, упрощая разработку.
- Pillow (PIL): Библиотека для работы с изображениями, которая используется для загрузки и отображения графических элементов интерфейса, таких как логотипы или иконки.
- Openpyxl: Библиотека для работы с файлами Microsoft Excel, применяемая для экспорта расписания в формат .xlsx, что позволяет пользователям сохранять и делиться расписанием.

- `os` и `random`: Встроенные модули Python. Модуль `os` используется для работы с файловой системой, а `random` — для генерации случайных данных, таких как уникальные идентификаторы записей.

3.1.2 Обоснование выбора инструментов

Выбор указанных инструментов был основан на нескольких ключевых факторах:

- Легкость использования: PyCharm предоставляет удобные инструменты для разработки, а Tkinter обеспечивает простой функционал для создания интерфейса, что позволяет легко интегрировать GUI с Python-кодом.
- Гибкость в хранении данных: JSON-файл как база данных идеально подходит для небольших проектов, не требующих сложной структуры данных.
- Экспорт в Excel: Orenpruhl позволяет удобно экспортировать данные в формат Excel, что значительно улучшает работу с расписаниями.
- Поддержка изображений: Использование Pillow помогает улучшить внешний вид интерфейса, делая его более привлекательным.
- Удобство управления: `os` и `random` обеспечивают автоматизацию управления файлами и данные.

Таким образом, выбранные инструменты полностью удовлетворяют требованиям проекта и обеспечивают гибкость и простоту разработки.

3.2 Разработка программного модуля

В этой главе описывается процесс разработки системы "Расписание занятий студентов", ориентированной на работу с базой данных в формате JSON.

3.2.1 Реализация пользовательского интерфейса программы

Интерфейс был разработан с использованием библиотеки Tkinter, что позволило создать удобное приложение с понятным и простым пользовательским интерфейсом. Основные компоненты интерфейса включают:

- 1) Главное окно: реализовано методом `create_login_window`, предоставляющим пользователю возможность авторизоваться или зарегистрироваться. Оно представлено в Приложении 3, Рисунок 1.
- 2) Форма регистрации: открывается методом `create_register_window`, где пользователь вводит имя, email, группу и пароль. С формой регистрации можно ознакомиться в Приложении 3, Рисунок 2.
- 3) Просмотр расписания: после авторизации пользователю доступно расписание, которое отображается методом `view_schedule`. Данное окно представлено в Приложении 3, Рисунок 3.
- 4) Редактирование расписания: Администраторы могут редактировать расписание с помощью метода `edit_schedule`. Это окно приведено в Приложении 3, Рисунок 4.

3.2.2 Описание функциональных узлов

Создание JSON-файла для хранения данных
Файл создается автоматически при запуске приложения, если его нет.

Методы для работы с этим файлом:

- `insert_user`: добавляет новых пользователей.
- `generate_schedule_for_all_users`: создает расписание для пользователей.

1) функция login

Назначение: проверяет данные пользователя при авторизации.

Задачи:

- Чтение данных из JSON-файла.
- Сопоставление введенного логина и пароля.

— Открытие окна расписания при успешной авторизации.

Функция приведена в Приложении 4, Рисунок 1.

2) функция `view_schedule`

Назначение: отображает расписание для группы или пользователя.

Задачи:

— Извлечение данных из JSON-файла.

— Форматирование данных для отображения.

Ознакомиться с данной функцией можно в Приложении 4, Рисунок 2.

3) функция `export_schedule_to_excel`

Назначение: экспортирует расписание в файл Excel. Задачи:

— Чтение данных из JSON-файла.

— Создание Excel-документа с помощью `Openpyxl`.

— Сохранение файла на локальном диске.

Данная функция приведена в Приложении 4, Рисунок 3.

4) функция `edit_schedule`

Назначение: позволяет редактировать расписание. Задачи:

— Обновление данных в JSON-файле.

— Обновление графического интерфейса расписания.

Ознакомиться с данной функцией можно в Приложении 4, Рисунок 4.

3.2.3 Результат работы и тестирования

Тестирование системы проводилось для проверки правильности работы всех функций. Основные тестовые сценарии:

— Создание базы данных: проверка правильного создания JSON-файла.

Данный тест-кейс приведен в Приложении 2, Таблица 1

— Регистрация: тестирование ввода правильных и ошибочных данных через метод `insert_user`. Ознакомиться с этим тест-кейсом можно в Приложении 2, Таблица 2 и Таблица 3)

- Авторизация: проверка корректности обработки данных при успешной и неуспешной авторизации через метод login. Данный тест-кейс приведен в Приложении 2, Таблица 4
- Экспорт в Excel: проверка корректности формирования и сохранения Excel-файла. Тест-кейс приведен в Приложении 2, Таблица 5

Результаты тестов показали, что система функционирует стабильно и соответствует всем функциональным требованиям.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта на тему «Разработка программного модуля системы расписания занятий студента» была проделана значительная работа, направленная на создание эффективного инструмента для управления учебным временем как для студентов, так и для преподавателей. На начальном этапе исследования были тщательно проанализированы существующие проблемы в организации расписания учебных занятий, включая такие распространенные сложности, как частые несовпадения времени проведения занятий, недостаток доступной и актуальной информации, а также значительные трудности в процессе планирования учебного графика. Особое внимание было уделено необходимости обеспечения удобства использования системы расписания не только для студентов, но и для преподавателей, что позволяет им эффективно планировать свою работу, оптимизировать распределение времени и ресурсов, и, как следствие, повысить эффективность преподавания. Для решения выявленных проблем был разработан программный модуль расписания под названием «Компас».

Процесс реализации проекта включал в себя разработку функционального и интуитивно понятного пользовательского интерфейса, обеспечивающего лёгкость внесения изменений в расписание, а также удобство просмотра актуальной информации о расписании занятий. Разработанный интерфейс значительно упрощает процесс планирования и управления учебным временем, что, в свою очередь, способствует повышению успеваемости студентов и, что немаловажно, повышению уровня удовлетворенности преподавателей от процесса планирования и организации учебного процесса.

По окончании работы были достигнуты все поставленные задачи в начале. Был проведён теоретический анализ текущих результатов в области построения расписаний в образовательных учреждениях. Была разработана и

реализована архитектура системы, включающая в себя эффективный алгоритм формирования расписания с учётом различных ограничений и требований. Реализованная система предоставляет пользователям возможность редактировать расписание. Было проведено тестирование разработки-кейсов и проверку работоспособности всех функций. В результате выполнения проекта были достигнуты поставленные цели, создав инструмент, который не только улучшает качество образовательного процесса, но и способствует созданию более гармоничной и эффективной учебной среды.

Таким образом, разработка программного модуля системы «Расписание занятий студента» стала важным шагом к оптимизации учебного процесса, что делает его актуальным и значимым для современного образования. Внедрение данного модуля в учебных заведениях окажет позитивное влияние на организацию учебного времени, повысит общую эффективность образовательного процесса и создаст более комфортные условия для обучения как студентов, так и преподавателей. Перспективы дальнейшей разработки включают расширение функциональности модуля, интеграцию с другими системами управления учебным процессом и адаптацию под потребности различных типов учебных заведений.

Полный программный продукт доступен на GitHub [6].

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Стандарты и законодательные материалы

1. Единая система программной документации (ЕСПД)
2. Национальный стандарт РФ ГОСТ Р ИСО/МЭК 12207-2010".
Информационная технология. Системная и программная инженерия.
Процессы жизненного цикла программных средств"
3. Федеральный закон "О безопасности критической информационной инфраструктуры Российской Федерации" от 26.07.2017 N 187-ФЗ
4. Федеральный закон "О персональных данных" от 27.07.2006 N 152-ФЗ
5. Федеральный закон "Об информации, информационных технологиях и о защите информации" от 27.07.2006 N 149-ФЗ (последняя редакция)

Монография

6. GitHub Чугунова А. Б. Продукт курсового проекта на тему «Расписание занятий студента»:

Учебники и учебные пособия

7. Гуриков С. Р. Основы алгоритмизации и программирования на Python: учебное пособие /— Москва: ИНФРА-М, 2023.
8. Любанович Б. Простой Python. Современный стиль программирования. — (Серия «Бестселлеры O'Reilly»). — СПб.: Питер, 2022
9. Мэтиз Э. Изучаем Python. Программирование игр, визуализация данных, веб-приложения. — СПб.: Питер, 2021.

Интернет-источники

10. Руководство по программированию на Tkinter и Python.
<https://metanit.com/python/tkinter/1.1.php>
11. Python и Tkinter | Кнопки. <https://metanit.com/python/tkinter/2.2.php>
12. Введение в библиотеку openpyxl.
https://www.geeksforgeeks.org/introduction-to-python-openpyxl/?ref=oin_asr20

13. Python GUI Programming with Tkinter. <https://realpython.com/python-gui-tkinter/>
14. Работа с JSON в Python: основы и примеры.
<https://pythonworld.ru/moduli/modul-json.html>
15. Использование модуля os для работы с файловой системой в Python.
<https://pythonworld.ru/moduli/modul-os.html>
16. Генерация случайных данных с использованием модуля random в Python.
<https://docs-python.ru/standart-library/modul-random-python/>
17. Руководство по модулю JSON в Python. <https://realpython.com/python-json/>
18. Обработка изображений с помощью библиотеки PIL (Pillow) в Python.
<https://habr.com/ru/articles/312154/>
19. Работа с изображениями в Python: от загрузки до изменения размера.
<https://pillow.readthedocs.io/en/stable/>
20. Объектно-ориентированное программирование в Python: классы и методы.
<https://metanit.com/python/tutorial/7.1.php>

ПРИЛОЖЕНИЕ 1 Диаграммы

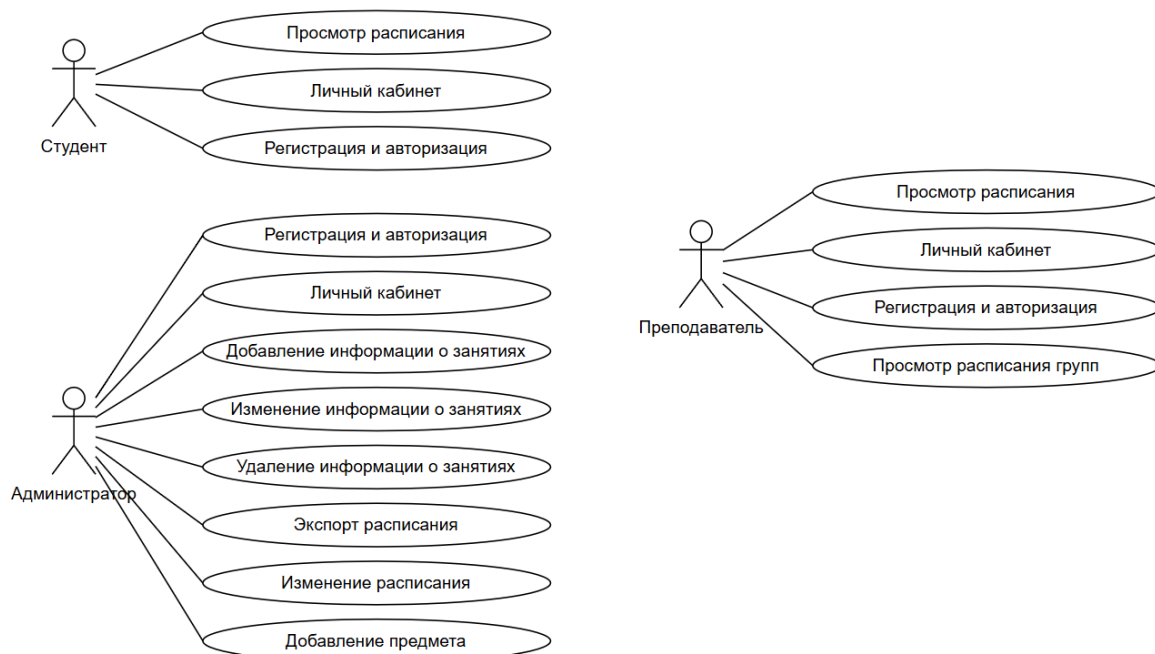


Рисунок 1 – диаграмма сценариев

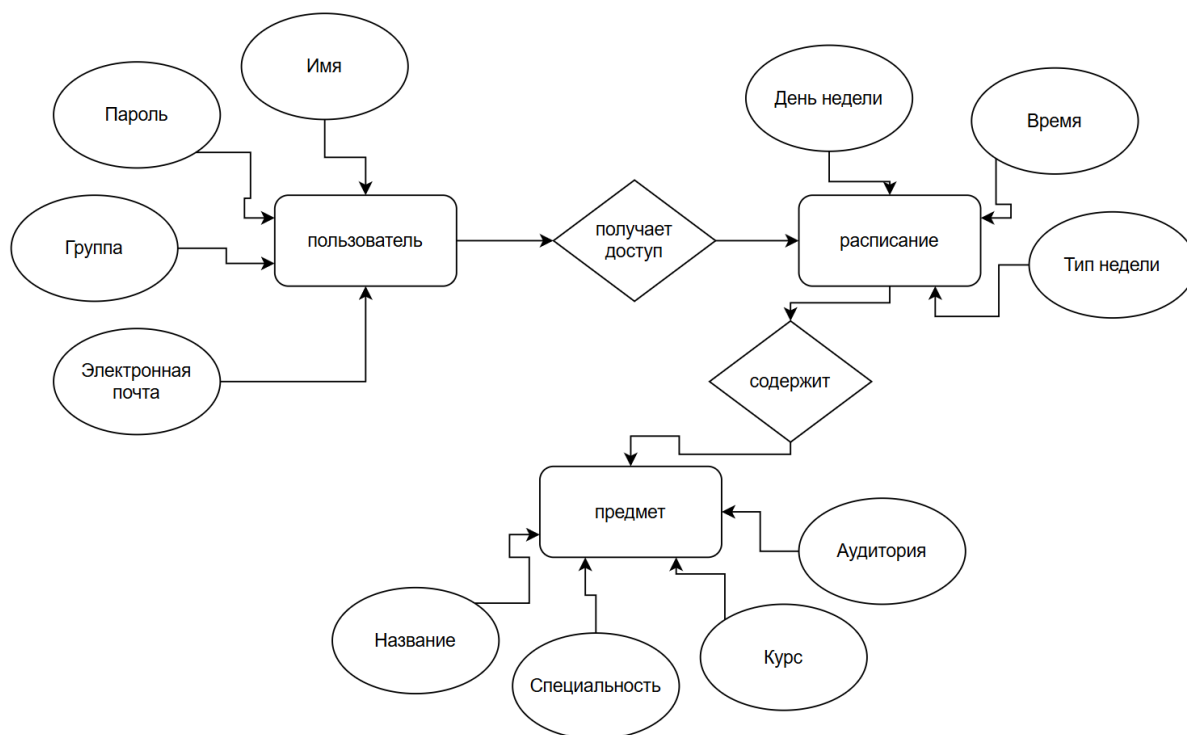


Рисунок 2 – ER диаграмма

ПРИЛОЖЕНИЕ 2 Тест-кейсы

Таблица 1 - тест-кейс №1

№	Наименование функциональности	Наименование поля	Тестовый набор	Ожидаемый результат
1	Проверка правильного создания JSON - файла	Название	Два зарегистрированных пользователя и генерация расписания	Успешное отображение данных
		Формат		
		Содержимое		

Фактический результат. Тест-кейс 1 – создание базы данных

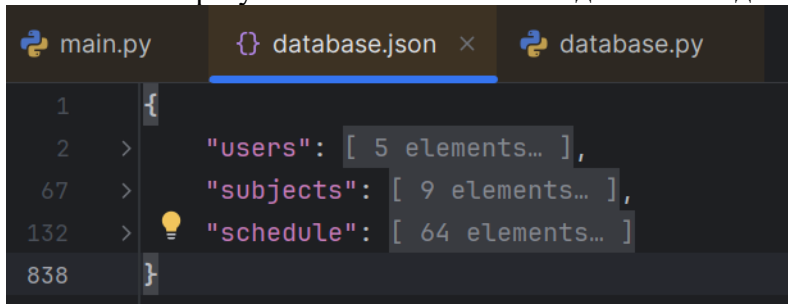


Таблица 2 - тест-кейс №2

№	Наименование функциональности	Наименование поля	Тестовый набор	Ожидаемый результат
2	Регистрация пользователя с корректными данными	ФИО	Чугунова А. Б.	Успешная регистрация
		Логин	shishaysno	
		Пароль	sasha135	
		Группа	312ИС-22	

Фактический результат. Тест-кейс 2 – Регистрация пользователя

РЕГИСТРАЦИЯ

ФИО: Чугунова А. Б.

Логин: shishaysno

Пароль: *****

Введите вашу группу:

312ИС-22

Зарегистрироваться

Успех

Студент зарегистрирован!

ОК

Таблица 3 - тест-кейс №3

№	Наименование функциональности	Наименование поля	Тестовый набор	Ожидаемый результат
3	Регистрация пользователя с некорректными данными	ФИО	Чугунова А. В.	Ошибка
		Логин	chugunovameow	
		Пароль	barkbark	
		Группа		

Фактический результат. Тест-кейс 3 – Регистрация пользователя

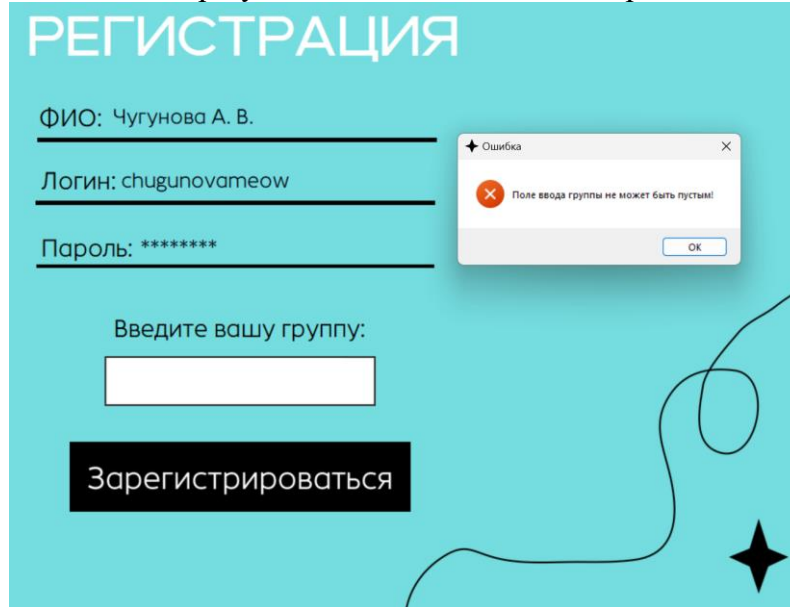


Таблица 4 - тест-кейс №4

№	Наименование функциональности	Наименование поля	Тестовый набор	Ожидаемый результат
4	Авторизация пользователя с корректными данными	Логин для входа	shishaysno	Успешный вход
		Пароль для входа	sasha135	

Фактический результат. Тест-кейс 4 – Авторизация пользователя



Таблица 5 - тест-кейс №5

№	Наименование функциональности	Наименование поля	Тестовый набор	Ожидаемый результат
5	Экспорт расписания в Excel	Данные расписания	Генерация расписания	Успешный экспорт. Данные в файле Excel отображаются корректно

Фактический результат. Тест-кейс 5 – Экспорт в Excel

The screenshot shows an Excel spreadsheet with a schedule for the 30th and 31st of December 2022. The spreadsheet is divided into two main sections, one for each day. Each section has a header row for the days of the week (Понедельник, Вторник, Среда, Четверг, Пятница, Суббота) and a table of activities. The activities are listed in columns, with their start and end times in the first column. The activities include: Разработка программных модулей (K65), Основы алгоритмизации и программирования (K53), Внедрение и поддержка компьютерных систем (K505), Иностранный язык в профессиональной деятельности (K406), Технологии разработки и защиты баз данных (K615), Численные методы (K56), Биологические основы прикладного использования (K616), Поддержка и тестирование программных модулей (K515), Физическая культура (Спортзал), and Иностранный язык в профессиональной деятельности (K406).

ПРИЛОЖЕНИЕ 3 Интерфейс

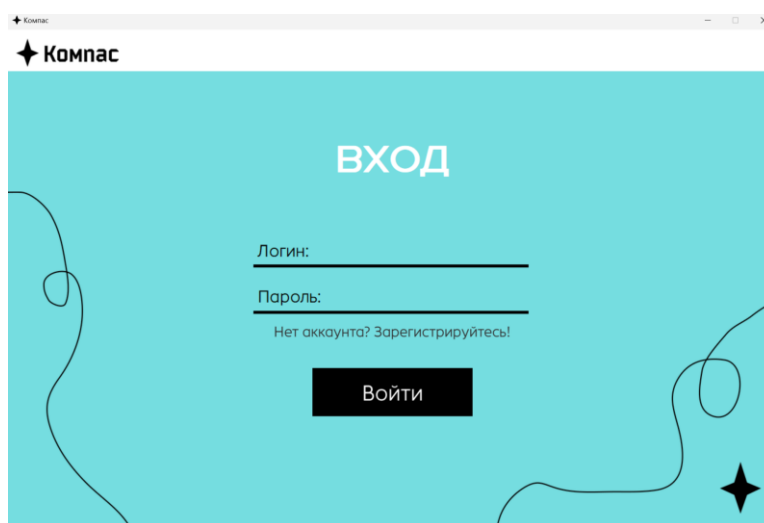


Рисунок 1 – Главное окно

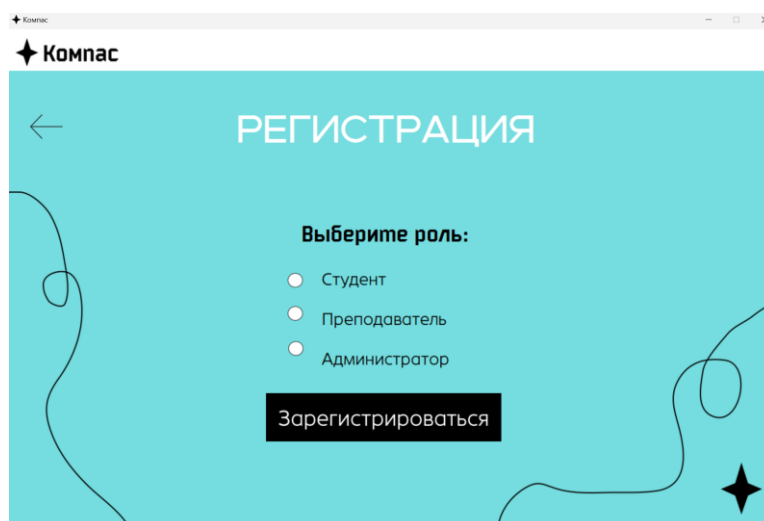


Рисунок 2 – Форма регистрации

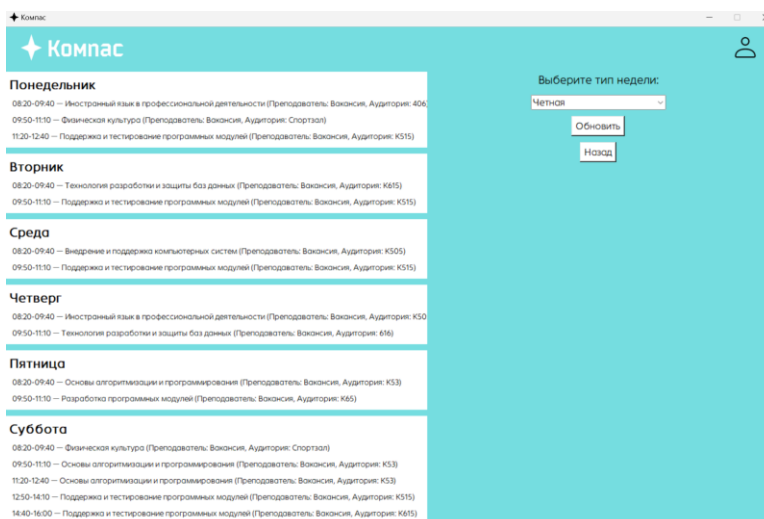


Рисунок 3 – Просмотр расписания

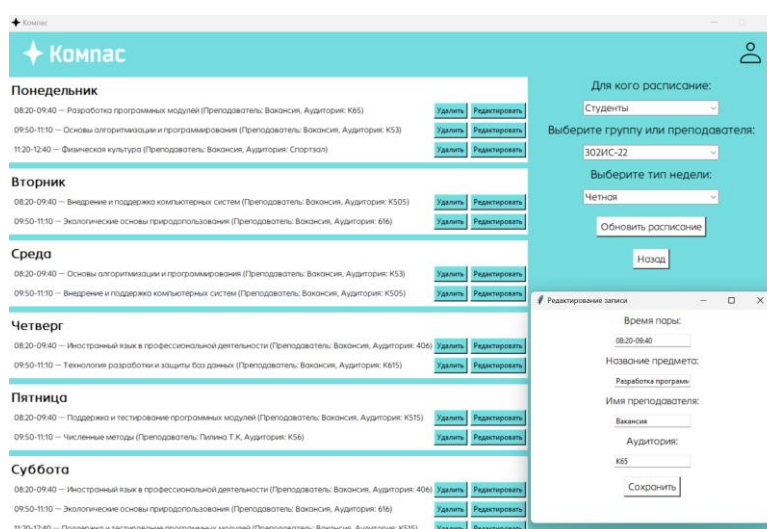


Рисунок 4 – Редактирование расписания

ПРИЛОЖЕНИЕ 4 Функции кода

```
def login(self):
    username = self.username_entry.get()
    password = self.password_entry.get()
    user = self.db.check_login(username, password)
    if user:
        self.current_user = user
        self.create_dashboard()
    else:
        messagebox.showerror( title: "Ошибка", message: "Неверные данные")
```

Рисунок 1 – функция login

```
def view_schedule(self):
    header_canvas = Canvas(self.root, bg="#FFFFFF", height=69, width=1280, bd=0, highlightthickness=0, relief="ridge")
    header_canvas.pack(side="top", fill="x")
    header_canvas.create_rectangle(0.0, 0.0, 1280.0, 69.0, fill="#75D0E0", outline="")

    self.button_image_3 = PhotoImage(file=self.relative_to_assets("button_11.png"))
    button_3 = Button(image=self.button_image_3, borderwidth=0, highlightthickness=0, command=self.open_user_profile, relief="flat")
    button_3.place(x=1193.0, y=6.0, width=60.0, height=57.0)

    self.button_image_4 = PhotoImage(file=self.relative_to_assets("button_25.png"))
    button_4 = Button(image=self.button_image_4, borderwidth=0, highlightthickness=0, command=self.reate_dashboard, relief="flat")
    button_4.place(x=0.0, y=0.0, width=241.0, height=69.0)

    scroll_canvas = tk.Canvas(self.root, bg="#75D0E0", highlightthickness=0, bd=0)
    scroll_canvas.pack(side="left", fill="both", expand=True)
    self.scroll_frame = tk.Frame(scroll_canvas, bg="#75D0E0", highlightthickness=0, bd=0)
    self.scroll_bar = tk.Scrollbar(self.root, orient="vertical", command=scroll_canvas.yview)
    scroll_bar.pack(side="right", fill="y")
    scroll_canvas.configure(yscrollcommand=scroll_bar.set)
    scroll_canvas.create_window((0, 0), window=self.scroll_frame, anchor="nw")

    def update_scroll_region(event):...
    self.scroll_frame.bind("<Configure>", update_scroll_region)
    def on_mouse_scroll(event):...
    scroll_canvas.bind_all("<MouseWheel>", on_mouse_scroll)
    def update_schedule():...

    tk.Label(self.root, text="Выберите тип недели:", bg="#75D0E0", font=("AA Stetica Regular", 14)).pack(pady=5, padx=180)
    week_type_var = tk.StringVar(value="Четная")
    week_type_dropdown = ttk.Combobox(self.root, font=("AA Stetica Regular", 12), textvariable=week_type_var, values=["Четная", "Не-"])
    week_type_dropdown.pack(pady=5, padx=20)
    week_type_dropdown.bind("<<ComboboxSelected>>", lambda event: root.focus())

    tk.Button(self.root, text="Обновить", bg="#FFFFFF", font=("AA Stetica Regular", 12), command=update_schedule).pack(pady=5, padx=10)
    tk.Button(self.root, text="Назад", bg="#FFFFFF", font=("AA Stetica Regular", 12), command=self.create_dashboard).pack(pady=5, padx=10)

    update_schedule()
```

Рисунок 2 – функция view_schedule

```

def export_schedule_to_excel(self, filename="schedule.xlsx", database_path="database.json"):
    wb = Workbook()
    even_fill = PatternFill(start_color="D6EAF8", end_color="D6EAF8", fill_type="solid")
    odd_fill = PatternFill(start_color="FAD7AB", end_color="FAD7AB", fill_type="solid")
    bold_font = Font(bold=True)
    group_font = Font(size=14, bold=True)
    border = Border(left=Side(style="thin"),right=Side(style="thin"),top=Side(style="thin"),bottom=Side(style="thin"),)
    def adjust_column_width(sheet):...
    def fill_schedule(sheet, schedule, groups_or_teachers, key, week_type, fill, row_offset):...
    students_sheet = wb.active
    students_sheet.title = "Студенты"
    schedule = self.get_schedule()
    even_week_schedule = [entry for entry in schedule if entry["week_type"] == "Четная"]
    odd_week_schedule = [entry for entry in schedule if entry["week_type"] == "Нечетная"]
    groups = sorted(set(entry["group_name"] for entry in schedule if entry["group_name"]))
    row_offset = 1
    for week_type, week_schedule, fill in zip(["Четная", "Нечетная"], [even_week_schedule, odd_week_schedule], [even_fill, odd_fill]):
        row_offset = fill_schedule(students_sheet, week_schedule, groups, key="group_name", week_type, fill, row_offset)
    adjust_column_width(students_sheet)
    teachers_sheet = wb.create_sheet(title="Преподаватели")
    with open(database_path, "r", encoding="utf-8") as db_file:
        database = json.load(db_file)
    teacher_id_to_name = {user["id"]: user["real_name"]}
    for user in database["users"]:
        if user.get("role") == "Преподаватель" and "real_name" in user:
            filtered_schedule = [entry for entry in schedule if entry.get("teacher_id") in teacher_id_to_name]
    for entry in filtered_schedule:
        teacher_id = entry["teacher_id"]
        entry["teacher_name"] = teacher_id_to_name[teacher_id]
    teachers = sorted(set(entry["teacher_name"] for entry in filtered_schedule))
    row_offset = 1
    for week_type, week_schedule, fill in zip(["Четная", "Нечетная"], [e for e in filtered_schedule if e["week_type"] == "Четная"], [odd_fill, even_fill]):
        row_offset = fill_schedule(teachers_sheet, week_schedule, teachers, key="teacher_name", week_type, fill, row_offset)
    adjust_column_width(teachers_sheet)
    wb.save(filename)
    os.startfile(filename)
    print(f"Расписание успешно экспортировано в файл {filename} и открыто!")

```

Рисунок 3 – функция export_schedule_to_excel

```

def edit_schedule(self):
    if self.current_user["role"] != "Администратор":
        messagebox.showerror(Имя: "Ошибка", message: "Недостаточно прав.")
        return
    self.clear_window()
    header_canvas = Canvas(self.root, bg="#FFFFFF", height=69, width=1280, bd=0, highlightthickness=0, relief="ridge")
    header_canvas.pack(side="top", fill="x")
    header_canvas.create_rectangle(0.0, 0.0, 1280.0, 69.0, fill="#750DE0", outline="")
    self.button_image_3 = PhotoImage(file=self.relative_to_assets("button_11.png"))
    button_3 = Button(image=self.button_image_3, borderwidth=0, highlightthickness=0, command=self.open_user_profile, relief="flat")
    button_3.place(x=1193.0, y=6.0, width=68.0, height=57.0)
    self.button_image_4 = PhotoImage(file=self.relative_to_assets("button_25.png"))
    button_4 = Button(image=self.button_image_4, borderwidth=0, highlightthickness=0, command=self.reate_dashboard, relief="flat")
    button_4.place(x=0.0, y=0.0, width=241.0, height=69.0)
    def on_mouse_scroll(event):
        scroll_canvas.yview_scroll(-1 * int(event.delta / 120), "units")
    scroll_canvas.bind_all("<<MouseWheel>", on_mouse_scroll)
    def update_schedule():...
    tk.Label(self.root, text="Для кого расписание:", bg="#750DE0", font=("AA Stetica Regular", 14)).pack(pady=5, padx=30)
    role_var = tk.StringVar(value="Студенты")
    role_dropdown = ttk.Combobox(self.root, font=("AA Stetica Regular", 12), textvariable=role_var, values=["Студенты", "Преподаватель"])
    role_dropdown.pack(pady=5, padx=20)
    role_dropdown.bind("<<ComboboxSelected>", lambda event: root.focus())
    role_var.trace("w", lambda *args: update_group_options())
    tk.Label(self.root, text="Выберите группу или преподавателя:", bg="#750DE0", font=("AA Stetica Regular", 14)).pack(pady=5, padx=30)
    groups = sorted(set(entry["group_name"] for entry in self.db.data["schedule"]), key=lambda x: (int(''.join(filter(str.isdigit, x))), x))
    group_var = tk.StringVar(value=groups[0] if groups else "")
    group_dropdown = ttk.Combobox(self.root, font=("AA Stetica Regular", 12), textvariable=group_var, values=groups, state="readonly")
    group_dropdown.pack(pady=5, padx=20)
    group_dropdown.bind("<<ComboboxSelected>", lambda event: root.focus())
    tk.Label(self.root, text="Выберите тип недели:", bg="#750DE0", font=("AA Stetica Regular", 14)).pack(pady=5, padx=30)
    week_type_var = tk.StringVar(value="Четная")
    week_type_dropdown = ttk.Combobox(self.root, font=("AA Stetica Regular", 12), textvariable=week_type_var, values=["Четная", "Нечетная"])
    week_type_dropdown.pack(pady=5, padx=20)
    week_type_dropdown.bind("<<ComboboxSelected>", lambda event: root.focus())
    tk.Button(self.root, text="Обновить расписание", bg="#FFFFFF", font=("AA Stetica Regular", 12), command=update_schedule).pack(pady=5, padx=30)
    tk.Button(self.root, text="Назад", bg="#FFFFFF", font=("AA Stetica Regular", 12), command=self.create_dashboard).pack(pady=5, padx=30)
    update_schedule()

```

Рисунок 4 – функция edit_schedule