**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)

**ОТЧЁТ**

**по курсовой работе**

**направление 09.03.01 «Веб-технологии»**

**по теме: «Система управления студенческими проектами»**

Выполнил: студент группы 231-322

Данилов Тимофей Павлович

Руководитель: Даньшина М.В.  
- зам. декана по развитию факультета,

старший преподаватель

Москва

2024

**СОДЕРЖАНИЕ**

[**ГЛАВА 1. ВВЕДЕНИЕ**](#_8ex0ztf1gl6d) **3**

[1.1 Тема](#_53srnfmd19z8) 3

[1.2 Актуальность темы](#_sgoswjj8l675) 3

[1.3 Цель курсовой работы](#_1lu6afb3s4gm) 3

[1.4 Задачи](#_n9j46fj7ga4a) 4

[**ГЛАВА 2. ПРОЕКТ Django.**](#_hrv5aekfnhv4) **5**

[2.1 Основные характеристики Django:](#_gjdgxs) 5

[2.2 Установка django](#_40tjcx39ea0l) 6

[Шаг 1: Установка Python](#_1fob9te) 6

[Шаг 2: Установка виртуального окружения](#_3znysh7) 6

[Шаг 3: Установка Django](#_2et92p0) 7

[Шаг 4: Создание нового проекта Django](#_tyjcwt) 7

[Шаг 5: Запуск сервера разработки](#_3dy6vkm) 7

[2.3 Описание структуры проекта](#_djtocjmujovy) 8

[Основная директория проекта (library\_system/)](#_1t3h5sf) 8

[Внутренняя директория проекта (library\_system/library\_system/)](#_4d34og8) 8

[Создание приложений внутри проекта](#_2s8eyo1) 10

[2.4 База данных](#_opgdcydv5uff) 13

[2.5 ER-модель базы данных](#_9cxnk9pzi6ty) 15

[2.6 Настройки админки](#_c0slo5teu6yc) 15

[2.7 Добавление Debug Toolbar](#_4e3m8u7nmnpk) 17

[Установка django-debug-toolbar](#_ac5t4gxvm1ja) 17

Результаты…………………………………………………………………….….19

[**Заключение**](#_7zdyc2a1o0k6) **23**

[**Список используемой литературы**](#_fpeyyd9ze943) 24

# ВВЕДЕНИЕ

## **1.1 Тема:** Создание сайта бронирования номеров гостевого дома.

## **1.2 Актуальность темы**

Создание сайта для бронирования номеров в гостевом доме актуально по нескольким причинам. Современные потребители ожидают получать информацию онлайн, и сайт предоставляет доступ к информации о номерах, их характеристикам, ценам и акциям в любое время и из любого места. Он расширяет охват аудитории, включая удалённых клиентов, и улучшает обслуживание через онлайн-консультации, мгновенное бронирование и оплату.

В условиях высокой конкуренции сайт даёт значительное преимущество, позволяя выделиться среди конкурентов. Также он служит основой для цифрового маркетинга, включая SEO, контекстную рекламу и продвижение в социальных сетях, что помогает привлекать и удерживать клиентов. Сайт позволяет собирать данные о поведении пользователей для корректировки маркетинговых стратегий. В условиях пандемии онлайн-присутствие стало ещё важнее, так как многим клиентам удобнее и безопаснее бронировать номера онлайн. Создание сайта способствует росту бизнеса и улучшению взаимодействия с клиентами.

**1.3 Цель курсовой работы**

Целью данной курсовой работы является исследование и практическое применение фреймворка Django для разработки веб-приложения бронирования номеров. Работа будет посвящена изучению основных принципов работы Django, его компонентов и возможностей настройки, с акцентом на создании функционального веб-приложения. Основное внимание будет уделено вопросам работы с базой данных, аутентификации и авторизации пользователей, возможности добавлять отзывы, бронировать номера.

## **1.4 Задачи**

1. Разработать проект базы данных (минимум 6 таблиц, включить поля M2M)
2. Выполнить Django тьюториал.
3. Реализовать добавление, удаление, редактирование и просмотр записей из базы данных на страницах проекта.
4. Настроить админку для django проекта.
5. Модифицировать базу и проект для того, чтобы можно показать те или иные настройки.
6. Заполнить базу данных нетестовыми записями, минимум 10 записей в каждой таблице.

**ГЛАВА 2**

**ПРОЕКТ Django**

Django — это высокоуровневый фреймворк для веб-разработки на языке программирования Python. Он был разработан с целью облегчить создание сложных и высоконагруженных веб-приложений. Django предоставляет разработчикам мощные инструменты для быстрой и эффективной разработки, следуя принципам "Don't Repeat Yourself" (DRY) и "Convention Over Configuration" (COC).

**2.1 Основные характеристики Django:**

1. **Быстрая разработка:** Django существенно ускоряет процесс разработки за счёт встроенных инструментов и библиотек. Он предоставляет множество готовых решений для типичных задач, таких как управление пользователями, создание форм и взаимодействие с базой данных.
2. **Масштабируемость**: Django идеально подходит для разработки как небольших, так и крупных проектов. Он обеспечивает поддержку масштабируемости на уровне кода и инфраструктуры, что позволяет легко расширять проект с ростом требований.
3. **Безопасность**: Django включает в себя встроенные механизмы для защиты от распространенных веб-угроз, таких как CSRF (межсайтовая подделка запросов), XSS (межсайтовый скриптинг), SQL-инъекции и кликджекинг. Благодаря этим механизмам Django становится надежным выбором для создания безопасных веб-приложений.
4. **Административный интерфейс**: Одной из ключевых особенностей Django является автоматическая генерация административного интерфейса для управления данными. Этот функционал позволяет легко выполнять CRUD-операции (создание, чтение, обновление, удаление) с моделями данных через удобный веб-интерфейс.
5. **ORM (Object-Relational Mapping)**: Django включает мощную систему ORM (Object-Relational Mapping), которая позволяет вам работать с базой данных на уровне объектов Python. Это значительно упрощает написание запросов и управление данными, делая код более читаемым и поддерживаемым.
6. **Поддержка шаблонов**: Django предоставляет собственный шаблонный движок, который помогает разделить логику приложения и представление данных. Это способствует созданию чистого и поддерживаемого кода, а также упрощает работу над дизайном интерфейса.

**2.2 Установка Django**

Установка Django действительно проста и может быть выполнена несколькими способами. Вот пример процесса установки Django с использованием pip, стандартного менеджера пакетов для Python:

**Шаг 1: Установка Python**

Перед установкой Django убедитесь, что у вас установлен Python. Для этого выполните следующую команду в командной строке.

python --version.

**Шаг 2: Установка виртуального окружения**

Использование виртуального окружения рекомендуется для изоляции зависимостей проекта. Чтобы создать виртуальное окружение, выполните следующие команды: Установите модуль virtualenv, если он еще не установлен: pip install virtualenv

Создаем виртуальное окружение: virtualenv myenv

1. Здесь myenv — это имя виртуального окружения. Вы можете использовать любое имя.
2. Активируйте виртуальное окружение: myenv\Scripts\activate

**Шаг 3: Установка Django**

Теперь, когда виртуальное окружение активировано, устанавливаем Django с помощью pip:

pip install django

**Шаг 4: Создание нового проекта Django**

После установки Django вы можете создать новый проект с помощью команды django-admin startproject:

django-admin startproject course\_work

Здесь course\_work — это имя проекта. В каталоге, где выполнена команда, будет создана структура файлов и папок для нового Django-проекта.

**Шаг 5: Запуск сервера разработки**

Переходим в директорию вашего проекта:

cd course\_work

Запускаем сервер разработки Django:

python manage.py runserver

Если все прошло успешно, мы видим сообщение о запуске сервера:

Starting development server at http://127.0.0.1:8000/

Открываем браузер и переходим по адресу http://127.0.0.1:8000/. Мы должны увидеть страницу приветствия Django, что означает успешную установку и запуск нашего первого Django-проекта.

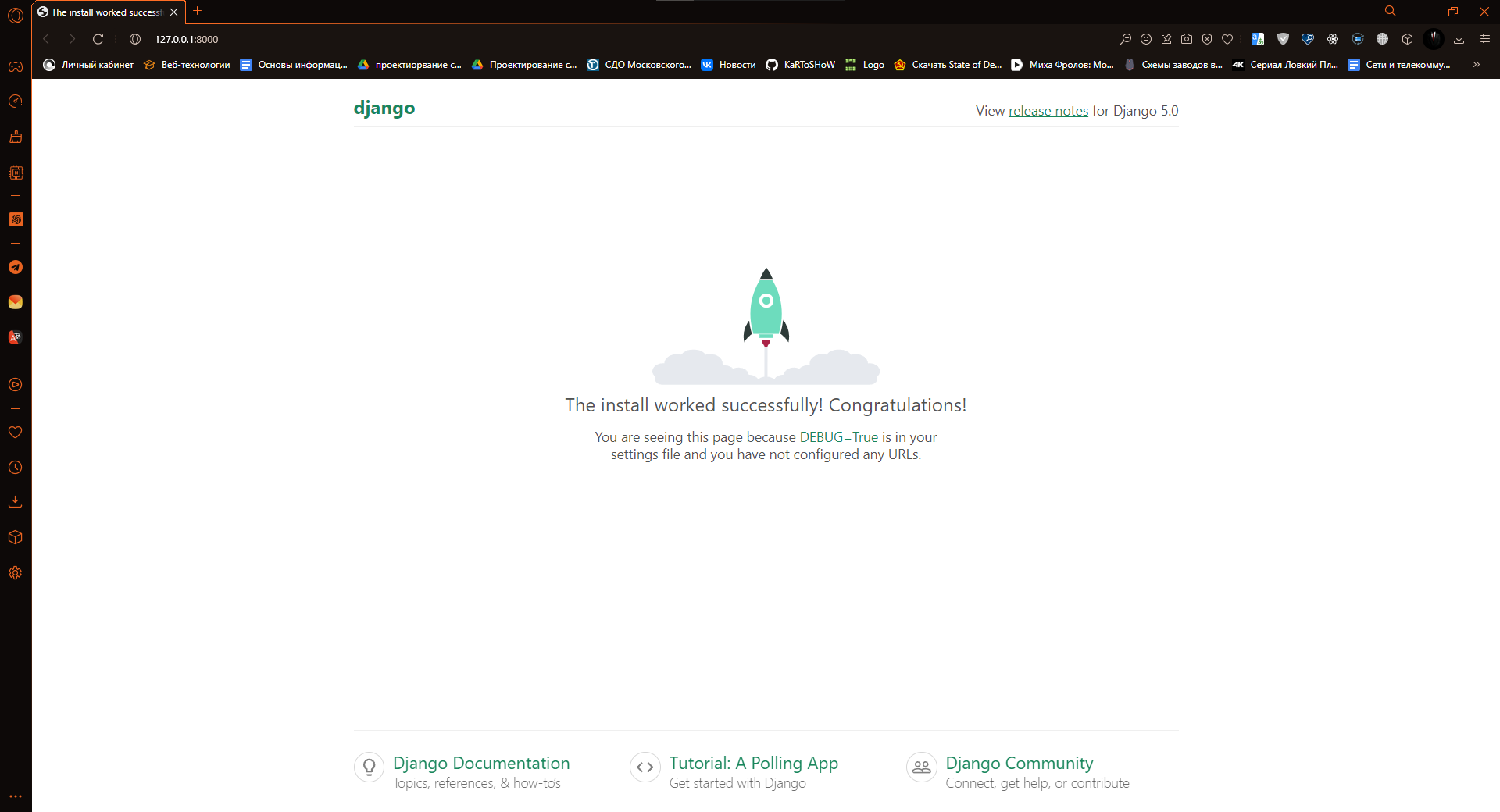


Рис.1

**2.3 Описание структуры проекта**

**Основная директория проекта (course\_work/)**

* manage.py: Этот файл является скриптом командной строки для взаимодействия с проектом Django. С его помощью можно выполнять различные задачи, такие как запуск сервера разработки, миграции базы данных, создание новых приложений и другие команды.

**Внутренняя директория проекта (course\_work/course\_work/)**

* \_\_init\_\_.py: Этот пустой файл указывает Python, что данный каталог должен рассматриваться как пакет. Это необходимо для того, чтобы Python мог корректно импортировать модули из этого каталога.
* settings.py: Это файл настроек проекта. Здесь находятся все конфигурационные параметры вашего проекта Django, такие как подключение к базе данных, параметры установленных приложений (apps), настройки middleware, статических и медиа файлов, шаблонов и т.д.

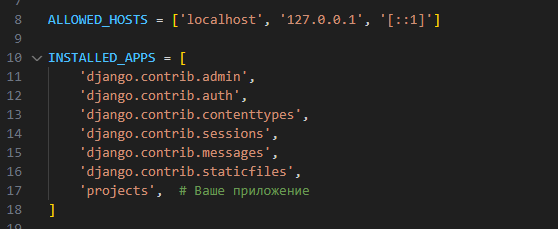


Рис.2

* urls.py: Файл маршрутизации URL. Здесь определяются маршруты (URL patterns) для проекта, связывая URL-адреса с соответствующими представлениями (views). Этот файл служит центральным маршрутизатором для приложения.

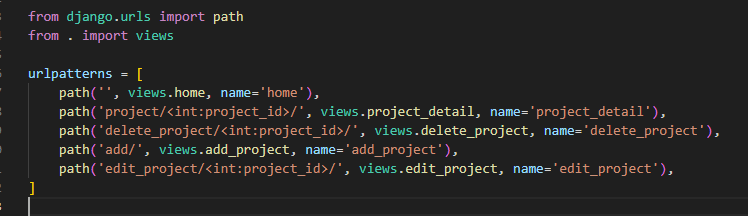


Рис.3

* wsgi.py: Файл конфигурации для интерфейса Web Server Gateway Interface. WSGI — это стандартный интерфейс между веб-серверами и веб-приложениями или фреймворками на языке Python. Этот файл используется для развертывания проекта на производственном сервере.

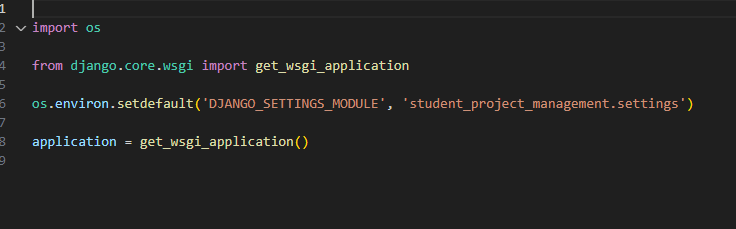


Рис.4

* asgi.py: Файл конфигурации для интерфейса Asynchronous Server Gateway Interface. ASGI предназначен для поддержки асинхронных веб-приложений на Python, что позволяет использовать веб-сокеты и другие асинхронные протоколы.

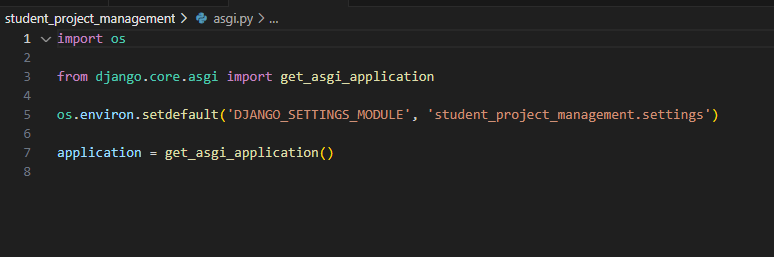


Рис.5

**Создание приложений внутри проекта**

Проект Django может содержать несколько приложений. Приложение в Django — это самостоятельный модуль, который выполняет конкретную задачу или предоставляет определенную функциональность. Чтобы создать новое приложение, выполните команду:

python manage.py startapp booking

Эта команда создаст структуру каталогов и файлов для нового приложения:

booking /

\_\_init\_\_.py

admin.py

apps.py

models.py

tests.py

views.py

migrations/

\_\_init\_\_.py

Описание файлов внутри приложения:

* \_\_init\_\_.py: Указывает, что данный каталог является пакетом Python.
* admin.py: В этом файле регистрируются модели, чтобы они отображались в административном интерфейсе Django.
* apps.py: Содержит конфигурацию приложения. Здесь можно задать метаинформацию и настройки для вашего приложения.

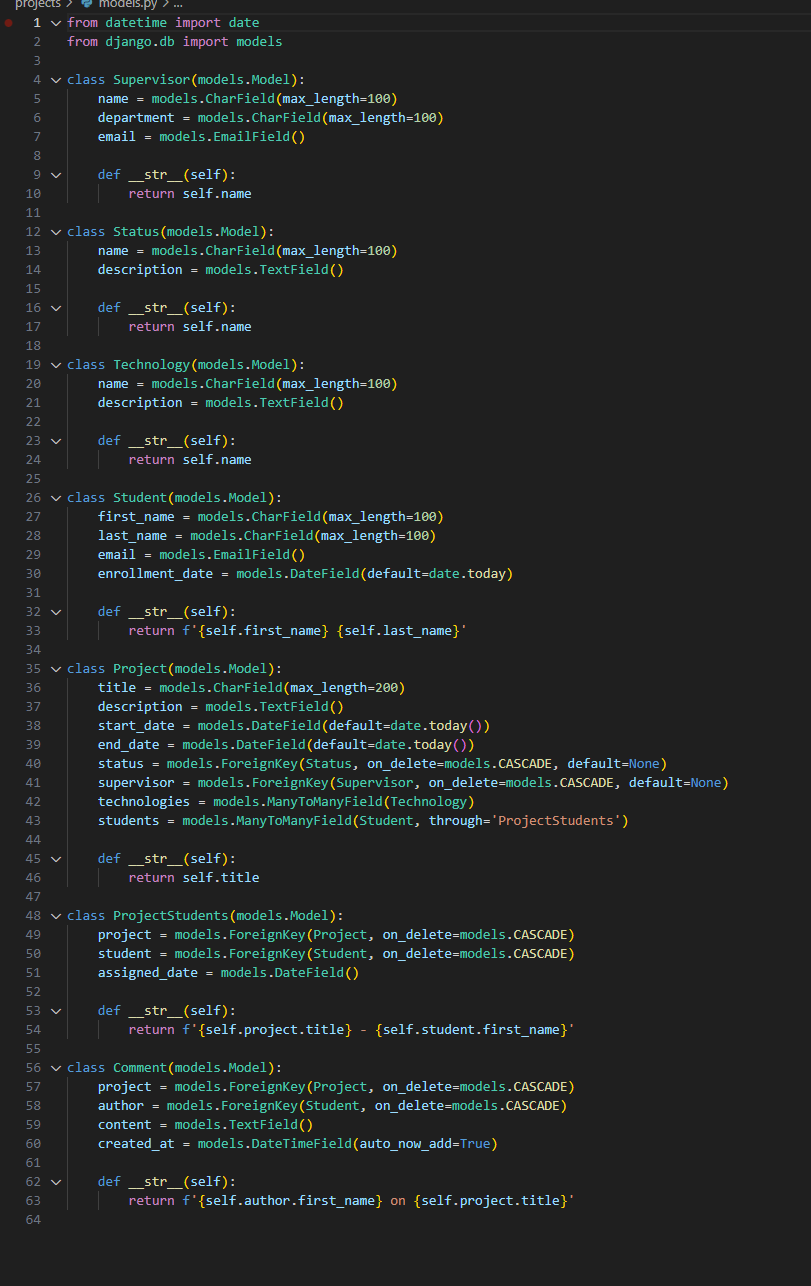


Рис.6

* models.py: В этом файле определяются модели данных. Модели описывают структуру и логику работы с данными в приложении, а также их связь с базой данных.
* tests.py: Здесь пишутся тесты для вашего приложения. Тесты позволяют автоматизировать процесс проверки правильности работы кода.
* views.py: В этом файле определяются представления (views), которые обрабатывают запросы и возвращают ответы. Представления связывают логику вашего приложения с соответствующими шаблонами и данными.
* migrations/: Этот каталог содержит файлы миграций базы данных, которые создаются при изменении моделей. Миграции позволяют синхронизировать структуру базы данных с определениями моделей в коде.

## **2.4 База данных**

1 Проекты

* **Атрибуты:**
  + id (Первичный ключ): Целое число, автоинкремент.
  + Title: Поле типа varchar, максимальная длина 200 символов.
  + Description: тип: text
  + Start\_date: type: date
  + End\_date: type: date

2 Статус

* **Атрибуты:**
  + id (Первичный ключ): Целое число, автоинкремент.
  + name: Поле типа varchar, максимальная длина 100 символов.
  + Description: type: text

3 **Students**

* **Атрибуты:**
  + id (Первичный ключ): Целое число, автоинкремент.
  + name: Поле типа varchar (100).
  + last\_name: Поле типа varchar (100).
  + email: Поле типа varchar (254).
  + enrollment\_date: type: date

4 **Supervisors**

* **Атрибуты:**
  + id (Первичный ключ): Целое число, автоинкремент.
  + Name: varchar (100).
  + Department: varchar (100).
  + email: Поле типа varchar (254).

5 **Technologys**

* **Атрибуты:**
  + id (Первичный ключ): Целое число, автоинкремент.
  + name: Поле типа varchar (100).
  + Description: text

6 **Comment**

* **Атрибуты:**
  + id (Первичный ключ): Целое число, автоинкремент.
  + Content: text.
  + Created\_at: datetime.
  + Project\_id: bigint.
  + Author\_id bigint.

7 ProjectsStudents

* Атрибуты:
  + id (Первичный ключ): Целое число, автоинкремент.
  + Assigned\_date: date
  + Project\_id: bigint.
  + Student\_id bigint.

**Связи**

1. **Project и Student через ProjectStudent** 
   * **Тип**: Много ко многим (Many-to-Many)
   * **Описание**: Эта связь описывает, что один проект может быть назначен многим студентам, и один студент может быть назначен на многие проекты. Связь осуществляется через промежуточную таблицу ProjectStudent, которая также хранит дополнительную информацию, такую как дата назначения (assigned\_date).
2. **ProjectStudent - Project**
   * **Тип**: Один ко многим (One-to-Many)
   * **Описание**: Эта связь описывает, что каждый объект ProjectStudent связан с одним объектом Project. То есть, каждый проект может иметь множество записей в ProjectStudent, но каждая запись в ProjectStudent относится только к одному проекту. Это реализуется с помощью поля ForeignKey в ProjectStudent, которое ссылается на Project.
3. **ProjectStudent - Student**

* **Тип**: Один ко многим (One-to-Many)
* **Описание**: Эта связь описывает, что каждый объект ProjectStudent связан с одним объектом Student. То есть, каждый студент может иметь множество записей в ProjectStudent, но каждая запись в ProjectStudent относится только к одному студенту. Это реализуется с помощью поля ForeignKey в ProjectStudent, которое ссылается на Student.

1. **Projects - Status**
   * **Тип:** Много ко многим
   * **Описание:** Каждый Статус может быть связан с многими проектами

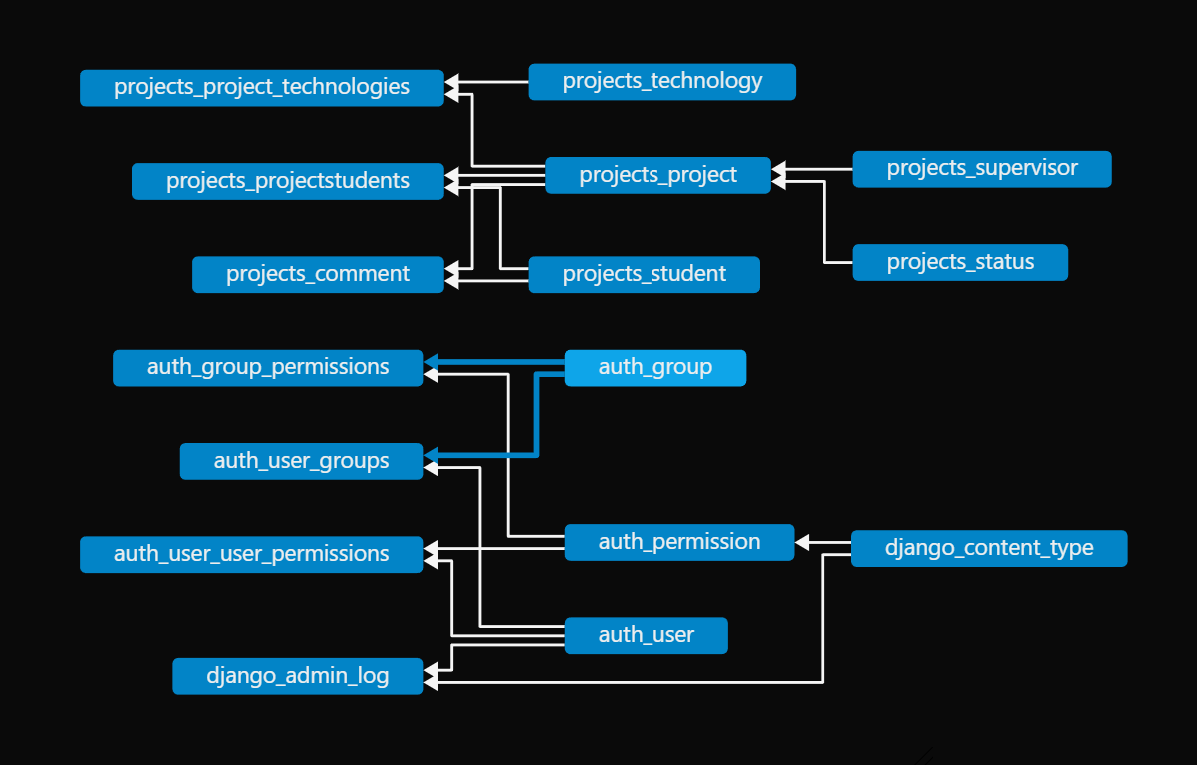


Рис.7

## **2.6 Настройки админки**

list\_display - отображает указанные поля в списке объектов.

list\_filter - добавляет боковую панель с фильтрами.

inlines - позволяет редактировать связанные объекты на одной странице с основным объектом.

date\_hierarchy - добавляет навигацию по датам сверху страницы.

@admin.display - позволяет определить метод для отображения вычисляемого поля.

short\_description - позволяет задать читаемое название для метода.

filter\_horizontal - добавляет горизонтальный виджет для выбора связанных объектов.

list\_display\_links - делает указанные поля ссылками на страницы редактирования объектов.

raw\_id\_fields - отображает поле ForeignKey как поле ввода id.

readonly\_fields - делает указанные поля только для чтения.

search\_fields - добавляет строку поиска по указанным полям.

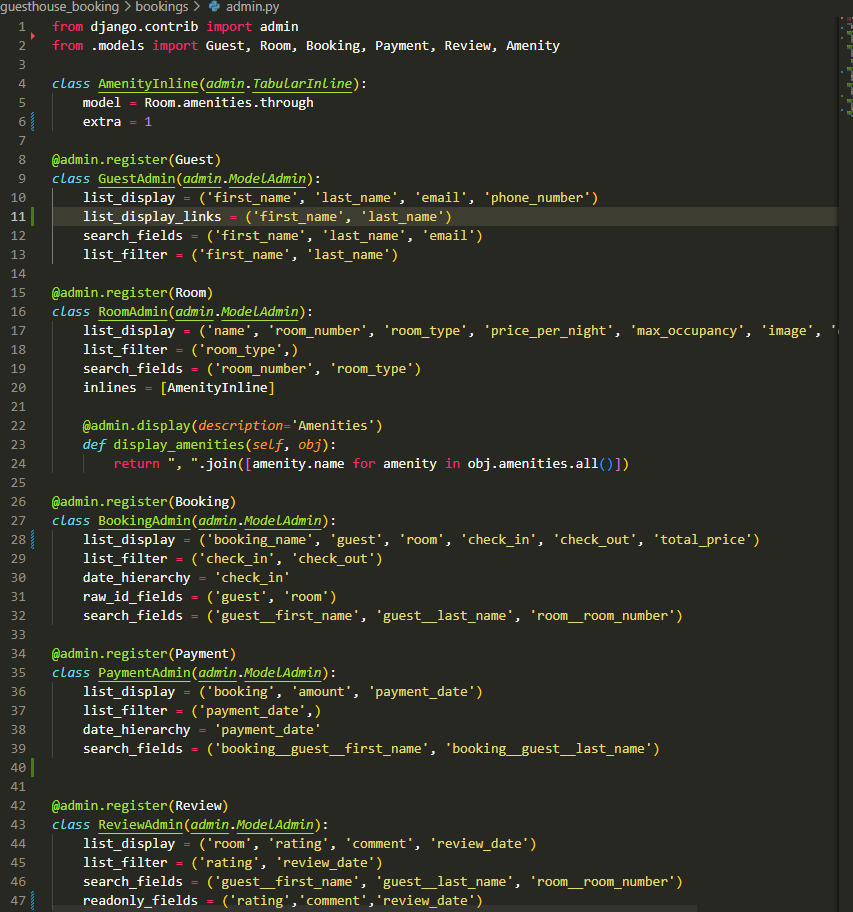


Рис.8

**Результат**

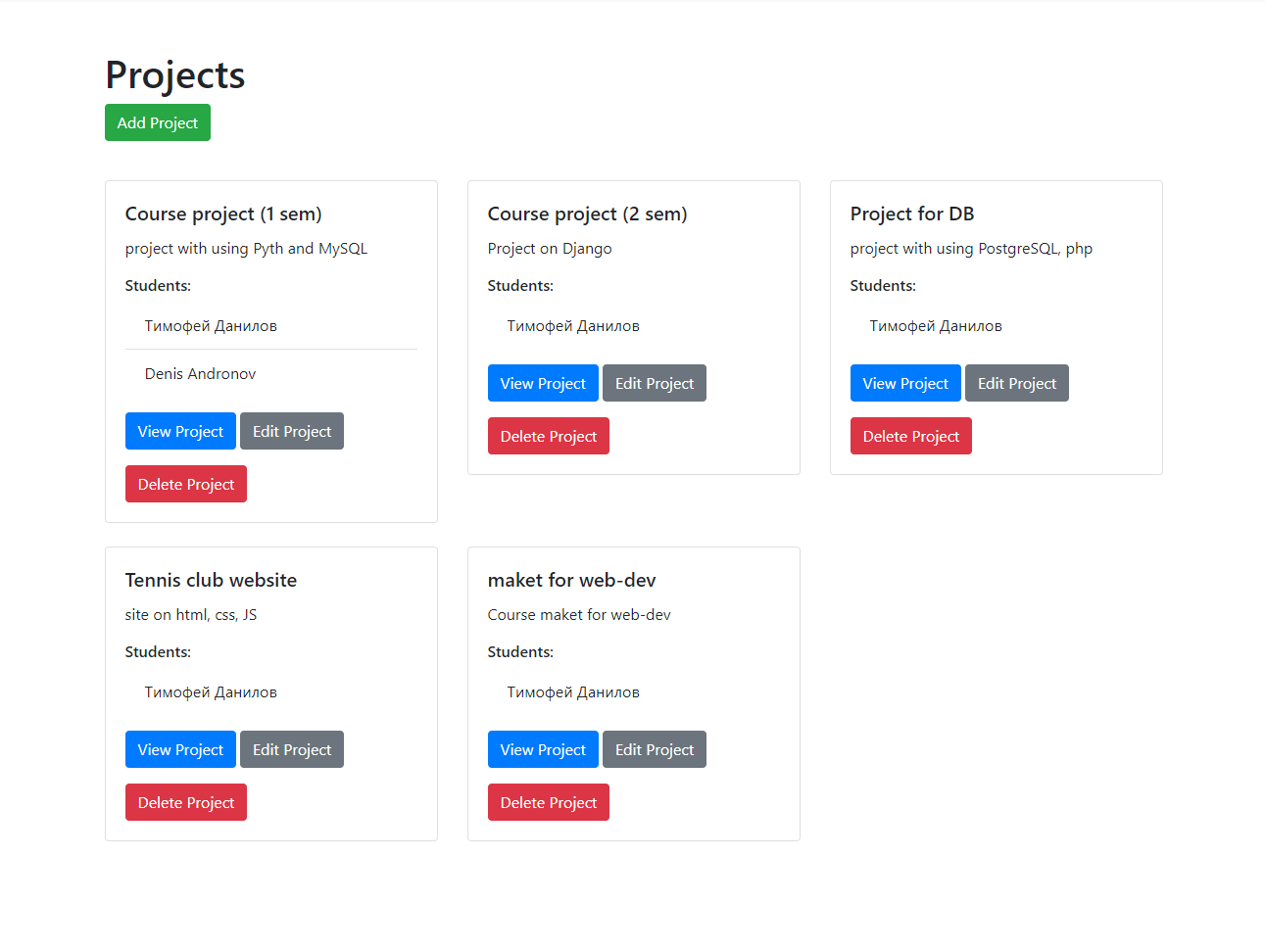


Рис.10

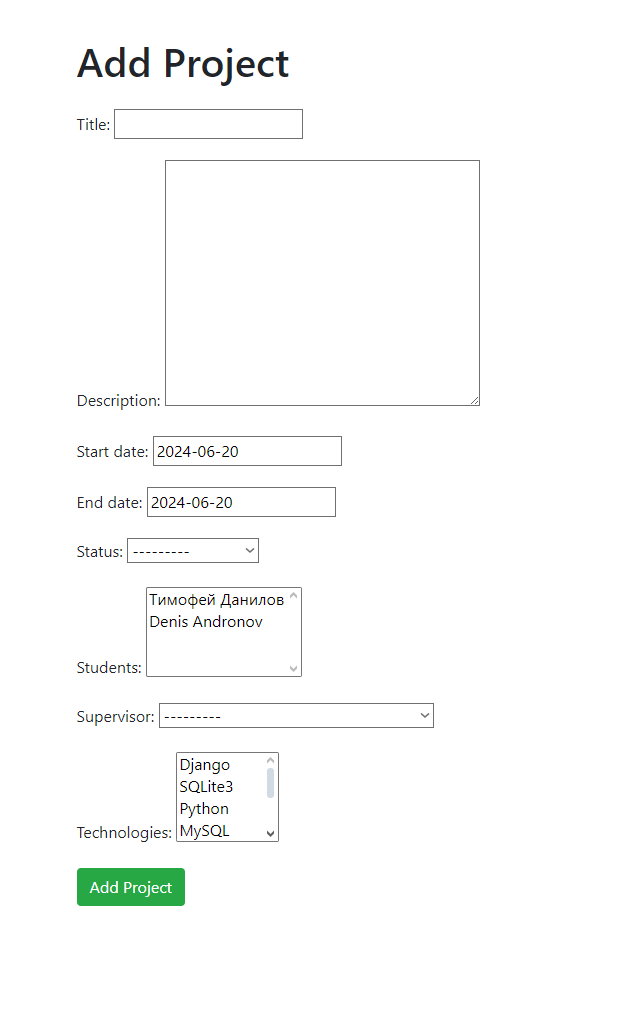


Рис.11

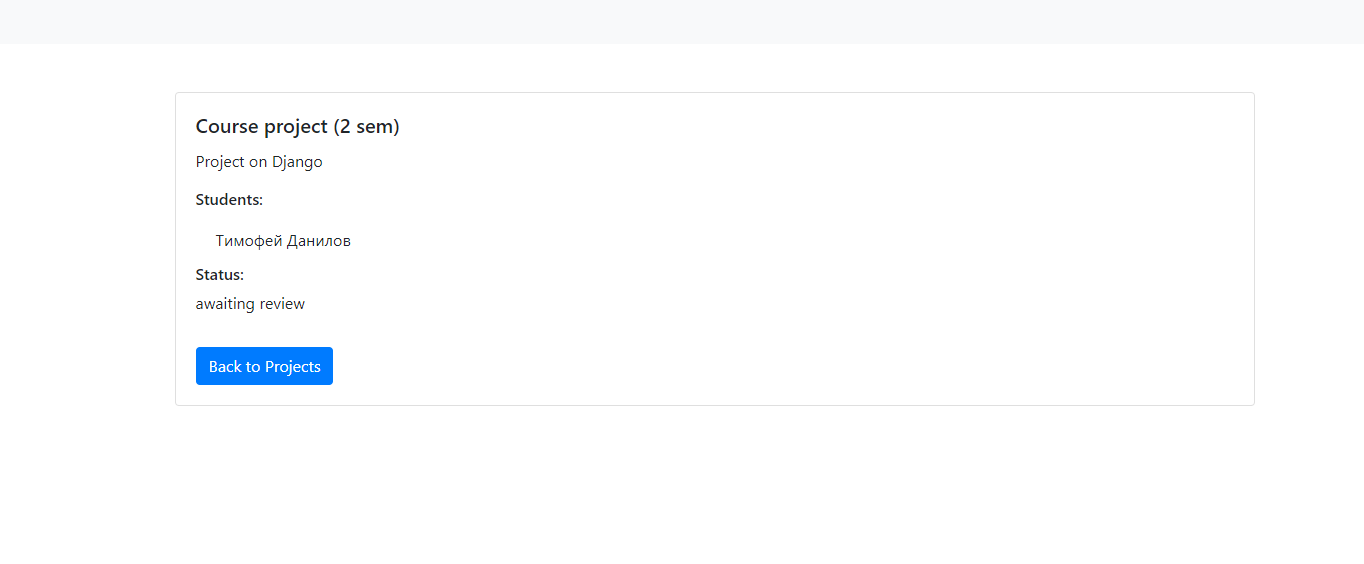


Рис.12

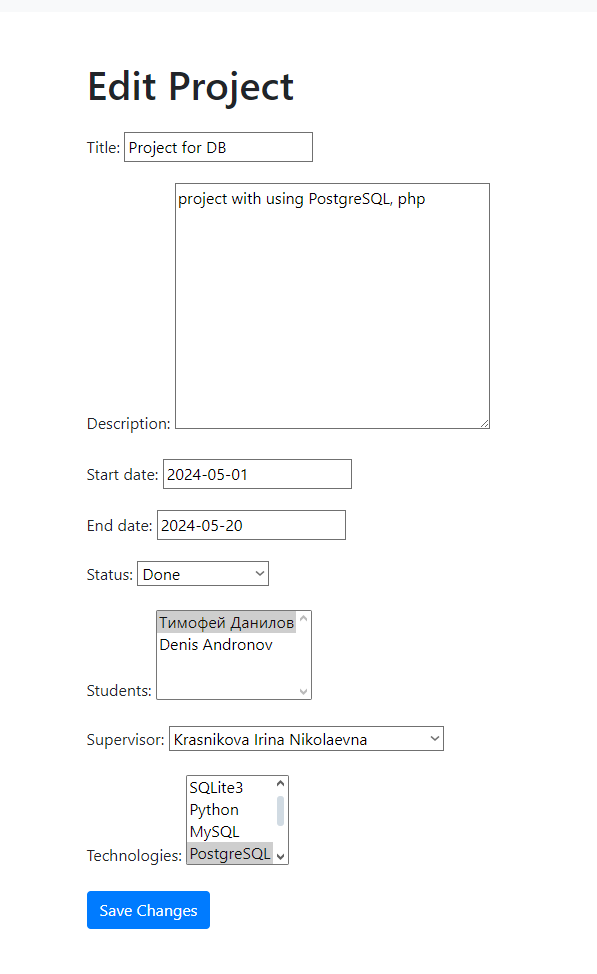


Рис.13

# Заключение

В рамках проекта "Система управления студенческими проектами" были реализованы следующие ключевые элементы и функциональности:

**Цель проекта**

Целью проекта было создание системы управления студенческими проектами, которая позволяет администраторам и преподавателям эффективно организовывать и отслеживать студенческие проекты, назначать студентов на проекты и управлять статусами проектов. Система должна обеспечивать удобный интерфейс для просмотра, добавления, редактирования и удаления информации о проектах и студентах.

**Реализованные функциональности**

1. **Модели данных**
   * **Student**: Хранит информацию о студентах, включая имя, фамилию и электронную почту.
   * **Project**: Хранит информацию о проектах, включая название, описание, дату начала и статус проекта.
   * **Status**: Хранит возможные статусы проектов, такие как "Запланирован", "В процессе", "Завершен".
   * **ProjectStudent**: Промежуточная модель, связывающая проекты и студентов, хранит информацию о дате назначения студента на проект.
2. **Связи в базе данных**
   * **Многие ко многим** (Many-to-Many) между проектами и студентами через модель ProjectStudent.
   * **Один ко многим** (One-to-Many) между проектами и статусами, что позволяет каждому проекту иметь один статус, но каждый статус может быть назначен множеству проектов.
3. **Функционал для работы с данными**
   * **Добавление, удаление, редактирование и просмотр записей**: Веб-интерфейс предоставляет функциональность для добавления новых проектов и студентов, редактирования существующих записей, удаления записей и просмотра подробной информации о проектах.
   * **Формы для ввода данных**: Были созданы формы для удобного ввода и редактирования данных о проектах, студентах и назначениях студентов на проекты.
4. **Интерфейс администратора**
   * **Настройки админки**: Админка была настроена для удобного управления всеми моделями, включая добавление, редактирование и удаление записей, а также просмотр связанных данных.
   * **Фильтрация и поиск**: Реализованы функции фильтрации и поиска данных в админке для упрощения навигации и управления большими объемами данных.
5. **Документация**
   * Полная документация по проекту включает описание моделей данных, их связей, функциональности веб-интерфейса, инструкции по установке и запуску проекта, а также примеры использования системы.

**Заключение**

Проект "Система управления студенческими проектами" достиг поставленных целей, обеспечив удобный инструмент для управления студенческими проектами. Система позволяет эффективно организовывать работу с проектами, назначать студентов и отслеживать прогресс проектов, что способствует более структурированному и организованному образовательному процессу.

В результате был создан надежный и удобный для пользователей веб-сайт, отвечающий современным требованиям.

# Список используемой литературы

1. [https://docs.djangoproject.com/en/5.0/](https://docs.djangoproject.com/en/4.2/intro/overview/) - Сайт официальной документации фреймворка Django (.
2. <https://developer.mozilla.org/ru/docs/Learn/Server-side/Django> - Сайт с документацией на русском языке.
3. <https://docs.djangoproject.com/en/4.2/ref/contrib/admin/> - Сведения о настройке панели администратора
4. <https://goo.su/IgZqdDG> - настройка панели администратора.
5. <https://tutorial.djangogirls.org/ru/> - сайт-помощник для создания сайтов на Django.