**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**(Университет ИТМО)**

Факультет **Прикладной информатики**

Направление подготовки **09.03.03 Прикладная информатика**

Образовательная программа **Мобильные и сетевые технологии**

**К У Р С О В О Й   П Р О Е К Т**

Тема: «Веб-приложение поиска мобильных материалов»

Обучающийся: Мурашов Никита Александрович K3141

Санкт-Петербург 2024

# **СОДЕРЖАНИЕ**

[**СОДЕРЖАНИЕ 2**](#_heading=)

[**ВВЕДЕНИЕ 3**](#_heading=h.gjdgxs)

[**ОСНОВНАЯ ЧАСТЬ 5**](#_heading=h.fzwqynsg06nf)

[**1 Описание проекта 5**](#_heading=h.30j0zll)

[**2 Этапы работы над проектом 7**](#_heading=h.1fob9te)

[**3 Мой вклад в проект 9**](#_heading=h.3znysh7)

[**4 Анализ своей работы 12**](#_heading=h.2et92p0)

[**5 Взаимодействие с командой 13**](#_heading=h.tyjcwt)

[**6 Взаимодействие с руководителем проекта 14**](#_heading=h.3dy6vkm)

[**7 Оценка работы руководителя 15**](#_heading=h.1t3h5sf)

[**ЗАКЛЮЧЕНИЕ 16**](#_heading=h.4d34og8)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17**](#_heading=h.2s8eyo1)

[**ПРИЛОЖЕНИЕ 18**](#_heading=h.17dp8vu)

# **ВВЕДЕНИЕ**

В настоящем курсовом проекте рассматривается разработка удобного веб-приложения, которое поможет студентам и преподавателям быстро находить и просматривать мобильные материалы для обучения. Сегодня, при высоких темпах цифровизации образования, у многих студентов возникает вопрос: где оперативно взять качественные и полезные учебные ресурсы?

Столкнувшись с этой проблемой, мы создали прототип приложения, позволяющего осуществлять эффективный поиск по базе материалов, а также добавлять новые статьи или ссылки. Доступ к сервису обеспечивается через веб-интерфейс, что делает его лёгким в использовании для людей с разным уровнем технических навыков.

Цель проекта: создать доступную платформу, в которой все нужные материалы окажутся под рукой. Развитие такого приложения поможет значительно сократить время, затрачиваемое на поиск, и дать возможность сосредоточиться на самом процессе обучения.

Для достижения цели нужно было выполнить несколько основных шагов:

1. Разработать модель данных, описывающую структуру учебных материалов.
2. Создать REST API, позволяющий взаимодействовать с базой и проводить CRUD операции.
3. Реализовать пользовательский интерфейс, удобный для студентов разных курсов.

В процессе реализации мы также приняли во внимание, что некоторые материалы лежат на сторонних платформах. Это позволило расширить функционал нашего приложения, чтобы оно могло работать не только с мобильными файлами (PDF, презентации, тексты лекций), но и с другими видами образовательных ресурсов (например, ссылки на Google-диски или внешние веб-ресурсы). Благодаря этому наше решение стало более универсальным и охватывающим широкий спектр задач, связанных с хранением и поиском учебных материалов.

# **ОСНОВНАЯ ЧАСТЬ**

## **1 Описание проекта**

Наше веб-приложение предназначено для хранения и поиска учебных материалов, ориентированных на мобильные устройства (например, PDF-версии книг, презентации, тексты лекций). Главная задача — обеспечить быстрый доступ к важной информации и возможность управлять своими коллекциями файлов и ссылок.

Основные пользователи приложения:

* Студенты, которым нужно структурировать учебные материалы;
* Преподаватели, желающие легко делиться материалами со студентами;
* Любой пользователь, ценящий удобство одного сервиса для хранения учебных ресурсов.

Приложение состоит из двух частей:

* Серверная часть (Backend): Написана на языке Python [1] с использованием Django [2] и Django REST Framework [3]. Здесь реализованы модели данных, аутентификация, а также логика работы с запросами;
* Клиентская часть (Frontend): Отображает интерфейс пользователя в браузере. Планируется применить современные веб-технологии, что обеспечит удобство, простоту и быстроту интерфейса.

Преимущества выбранного стека технологий:

* Python — высокоуровневый язык, поддерживающий большие сообщества разработчиков;
* Django — популярный фреймворк, предоставляющий административную панель, систему шаблонов и ORM для быстрого запуска проектов [4];
* Django REST Framework — дополнение для создания RESTful API, упрощающее интеграцию с клиентской частью;
* React с TypeScript — дают возможность типизации и упрощают поддержку кода;
* ShadcnUI — ускоряет разработку однообразных и стилистически целостных интерфейсов;
* React Router — обеспечивает удобную навигацию между страницами;
* Zod — упрощает валидацию данных при вводе (например, при регистрации пользователей или добавлении материалов).

Благодаря перечисленному стеку, приложение можно легко адаптировать для различных образовательных сценариев. Даже если основная задача — поиск мобильных материалов, архитектура проекта позволяет гибко интегрировать и более общие типы учебных ресурсов.

## **2 Этапы работы над проектом**

Весь процесс разработки можно условно разделить на несколько ключевых этапов:

1. Сбор требований и анализ

На старте команда обсуждала концепцию: какой объём информации будет обрабатываться, как именно пользователи должны взаимодействовать с материалами. Обсудили детали интерфейса, способы хранения и форматы файлов, а также возможности поисковой системы.

1. Создание моделей данных

Разработали подробную схему для представления учебных материалов. Модель Material описывает основные поля (заголовок, описание, файл, автор), а модель Tag отвечает за категории и метки. Были учтены ограничения, чтобы не перегружать базу данных лишней информацией.

1. Настройка REST API

На базе Django REST Framework реализовали CRUD [5] операции: добавление, редактирование, удаление и просмотр материалов. Дополнительно внедрили систему фильтрации и пагинации, что помогает справляться с обширными списками файлов.

1. Разработка интерфейса

Фронтенд-проект предполагается сделать удобным для начинающих пользователей. Уже созданы макеты основных страниц (поиск материалов, личный кабинет, страница добавления материалов), хотя детальная реализация ещё ведётся.

1. Тестирование и отладка

Написаны тесты для проверки корректности API, включающие сценарии на добавление новых записей и удаление существующих. При обнаружении ошибок проводилась отладка кода. Также создавались тестовые учётные записи, чтобы проверить возможность доступа разных пользователей.

1. Управление проектом и контроль версий

Основной инструмент управления задачами — Git. Проект разделён на ветки, каждая из которых предназначена для отдельной задачи. Команда регулярно обсуждает прогресс в общем чате и по необходимости корректирует сроки.

## **3 Мой вклад в проект**

1. Моделирование структуры данных  
    В ходе проектирования базы данных я уделил особое внимание тому, каким образом будет храниться информация о материалах и тегах, а также как они будут взаимодействовать между собой.
   1. Для каждого учебного материала (предполагая, что это могут быть файлы, ссылки или текстовые материалы) я определил набор критически важных полей: название, описание, ссылка на сам материал (или путь к загруженному файлу), автор, уровень сложности и счётчик просмотров. В дополнение к этому было учтено, что разные материалы могут относиться к нескольким категориям (тегам).
   2. При разработке структуры я предусмотрел, что в перспективе база может значительно расшириться. Поэтому я уточнил типы полей, выбрал оптимальные размеры для текстовых полей (например, CharField vs TextField для описаний) и продумал систему связей «многие-ко-многим» между материалами и тегами. Кроме того, взял в расчёт потенциальную нагрузку (например, при поиске или фильтрации) и постарался заложить возможность индексации некоторых полей.
2. Реализация CRUD операций в Django REST Framework  
    Чтобы обеспечить удобное взаимодействие с моделью данных (как для фронтенда, так и для внешних сервисов), я сосредоточился на создании REST API.
   1. Сначала я определил сериализаторы (serializers) для модели Material, чтобы переводить объекты учебных материалов в формат JSON [6] и обратно. При этом я описал поля, которые должны либо быть обязательными для заполнения, либо допускать пустые значения. Это позволило чётко контролировать валидацию данных и упростить разработку фронтенда.
   2. Затем реализовал эндпоинты (views) для основных CRUD операций. Это включало:
      * Создание нового материала (POST), где пользователь может передать все необходимые данные для записи;
      * Редактирование (PUT или PATCH), чтобы при необходимости менять отдельные поля (например, заголовок, описание);
      * Удаление (DELETE), обеспечивающее корректное удаление записей из базы данных;
      * Чтение (GET) — выдача списков материалов и отдельных записей.
   3. Я также добавил базовые фильтры, чтобы пользователи могли искать по названию, тегам и дате добавления. Это оказалось особенно важным при большом количестве записей. Фильтрация строилась в том числе на встроенном функционале Django REST Framework (например, django-filter).
3. Поддержка документации  
    Для команды и будущих разработчиков крайне важно иметь понятную документацию, описывающую, какие методы доступны в API и какие данные ожидаются.
   1. В рамках базового описания я добавил докстринги к ключевым методам API, а также оформил подробный README (или дополнительный файл), где перечислены пути к эндпоинтам, тела запросов и форматы ответов. Такой подход облегчает погружение в проект новых участников и интеграцию с другими системами.
   2. Кроме того, я частично внедрил механизмы автодокументирования, используя возможности Django REST Framework. Это помогает автоматически генерировать схемы (OpenAPI/Swagger) на основе сериализаторов и вьюшек. Благодаря этому любой заинтересованный пользователь API может быстро посмотреть, какие ресурсы есть в системе и как к ним обращаться.

Дополнительные аспекты

Я затратил значительное время на изучение рекомендаций по производительности и безопасности, чтобы при загрузке файлов не возникало риска испортить базу, а пользователи не могли внедрять вредоносный код или выполнять несанкционированные операции. Проверял, как приложение ведёт себя при большом количестве записей, анализировал логику фильтрации и старался минимизировать количество SQL-запросов при простом просмотре материалов. Заботился об ограничении типов загружаемых файлов, настраивал валидацию полей и следил за тем, чтобы случайные пользователи не имели доступа к чужим материалам или административной панели.

Таким образом, моя работа охватила создание надёжной и расширяемой модели данных, разработку и документирование API на базе Django REST Framework, а также тщательную проработку моментов, связанных с безопасностью и производительностью приложения.

## **4 Анализ своей работы**

Что получилось:

* Создать готовую для использования модель данных;
* Настроить API, позволяющее взаимодействовать с материалами;
* Организовать первоначальное тестирование работы кода.

С какими сложностями столкнулись:

* Выбор оптимальной структуры хранения информации, чтобы база данных не засорялась лишними полями;
* Изучение некоторых аспектов настройки JWT-токенов для аутентификации;
* Настройка тестовой среды — пришлось тратить время на конфигурацию отдельных профилей для разработки и тестирования.

Что можно улучшить:

* Дополнить фильтрацию материалов более гибкими критериями (например, по дате добавления или популярности);
* Рассмотреть возможность интеграции с системами рекомендаций, чтобы подсказывать пользователям новые подходящие материалы.

## **5 Взаимодействие с командой**

Вся команда взаимодействовала с помощью группового чата и Git. В чате обсуждали детали по мере возникновения вопросов. Периодические «созвоны» устраивались, когда нужно было согласовать архитектурные решения или дедлайны.

Принципы совместной работы:

* Поддерживать единый стиль кода;
* Закрывать каждую задачу отдельной веткой, затем делать merge через merge request;
* Не бояться задавать вопросы, если что-то не ясно.

Такой подход помог уменьшить количество конфликтов кода при одновременной работе нескольких человек над одним проектом.

## **6 Взаимодействие с руководителем проекта**

Руководитель проекта координировал распределение задач, помогал определить приоритеты. Регулярно проверял выполненный функционал и оставлял комментарии по улучшению кода. Благодаря советам руководителя:

* Оптимизировали структуру базы данных;
* Сократили избыточные запросы;
* Навели порядок в нейминге моделей и полей.

## **7 Оценка работы руководителя**

Я оцениваю работу руководителя проекта как отличную. Он проявил высокий профессионализм, организованность и заинтересованность в успешном выполнении проекта. Руководитель всегда был готов помочь и ответить на мои вопросы, предоставлял необходимую поддержку и ресурсы.

Особенно хочу отметить его способность давать своевременные и полезные советы, которые помогали мне находить оптимальные решения. Руководитель был достаточно гибким, чтобы адаптировать план проекта под возникающие обстоятельства, и в то же время был достаточно требовательным, чтобы мотивировать нас на достижение поставленных целей.

Также руководитель активно участвовал в подготовке к защите проекта, давал советы по выступлению и наполненности презентации. Он очень верил в нас, и знал, что у нас всё получится, что придавало нам сил во время близкого дедлайна.

В целом, работа с таким руководителем была очень позитивным и ценным опытом. Возможно, на будущее стоило бы еще более детально прорабатывать сроки и цели каждого этапа проекта для более точного понимания дедлайнов и распределения ресурсов, а также более активно участвовать в соединении работы каждого участника в единое целое.

# **ЗАКЛЮЧЕНИЕ**

Итоговая реализация:

* Приложение даёт возможность хранить и искать мобильные учебные материалы, связывать их с тегами и просматривать детальную информацию о каждом файле;
* Создан базовый функционал аутентификации и личных кабинетов;
* Проведено частичное тестирование, вскрывшее несколько незначительных недочётов, которые были оперативно исправлены.

Основные результаты:

* Цель по упрощению процесса поиска учебных ресурсов достигнута;
* Пользователь может быстро находить нужную информацию, а также добавлять собственные материалы.

Перспективы развития:

* + Введение системы рекомендаций и рейтинга материалов для улучшения удобства;
  + Расширение списка поддерживаемых форматов файлов;
  + Глубокая интеграция с социальными платформами, чтобы студенты могли делиться ссылками.

Таким образом, проект уже полезен тем, кто ищет удобный способ организовать свои учебные материалы. Планируется дальнейшее улучшение UI и добавление новых функций.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Python Documentation [Электронный ресурс] – URL: <https://docs.python.org/3/index.html> (дата обращения 27.11.2024).
2. Документация Django [Электронный ресурс] – URL: <https://docs.djangoproject.com/en/5.1> (дата обращения 27.11.2024).
3. Документация Django Rest-framework [Электронный ресурс] – URL: <https://www.django-rest-framework.org/topics/documenting-your-api/> (дата обращения 27.11.2024).
4. Принципы работы с Django проектом [Электронный ресурс] – URL: <https://docs.djangoproject.com/en/5.1/intro/tutorial01/> (дата обращения 27.11.2024).
5. Документация по CRUD операциям [Электронный ресурс] – URL: <https://www.geeksforgeeks.org/django-rest-api-crud-with-drf/> (дата обращения 27.11.2024).
6. Что такое JSON? [Электронный ресурс] – URL: <https://www.json.org/json-en.html> (дата обращения 2.12.2024).

# **ПРИЛОЖЕНИЕ**

Техническое задание на разработку веб-приложения

1. Общие положения
   1. Название проекта: веб-приложение поиска учебных материалов.
   2. Цель (назначение): разработка веб-приложения, которое позволит пользователям добавлять, искать и просматривать учебные материалы по различным предметам, улучшая доступность и удобство получения образовательной информации.
   3. Сроки выполнения: начало - 2024-11-01, конец - 2024-12-20.
   4. Команда проекта: исполнитель проекта (руководитель проекта) - Прохоров Николай Игоревич, backend разработчики - Левахин Лев, Богдан Мезенцев, Никита Мурашёв, frontend разработчики - Мария Фёдорова, Михаил Бай.
   5. Этапы задач: TO DO, Development, Review, Done.
2. Технические требования
   1. Регистрация и авторизация пользователя
   2. Поиск и просмотр учебных материалов
   3. Просмотр списка доступных предметов и добавление предметов в избранное
   4. Фильтрация результатов поиска по категориям
   5. Модерация материалов через админ панель. Управление пользователями(блокировка, изменение прав доступа)
   6. Поддержка одновременной работы не менее 1000 пользователей
   7. Интуитивно понятный и современный дизайн
   8. Регулярное резервное копирование базы данных
   9. Сервер с поддержкой Python и Django. База данных PostgreSQL
   10. Браузеры с поддержкой современных веб-технологий. Поддержка JavaScript и TypeScript.
3. Ожидаемый результат

MVP системы, отвечающей описанным в ТЗ функциональным требованиям, развернутой на сервере и доступной для пользователей.