**Дипломная работа по теме:** Web-приложения с использованием Django, Flask и FastAPI, а так же телеграмм бот, целевое значение которых оформление списка литературы согласно требованиям ГОСТ для студентов и научных сотрудников.

**Автор:** Дивеев Данила Васильевич

**Оглавление дипломной работы:**

**1. Введение3**

**2. Основные понятия и определения 5**

**3. Методы и подходы к разработке7**

**4. Обзор популярных инструментов для разработки Telegram бота8**

5.

6.

7.

8. Заключение…

**1. Введение**

**Обоснование выбора темы:**

1. Веб-приложения, разработанные на Django, Flask или FastAPI, обеспечивают доступ к функционалу из любого места с интернет-соединением. Telegram-бот позволяет пользователям получать помощь и оформлять списки литературы прямо в мессенджере, что увеличивает удобство использования. Оформление списка литературы — это рутинная задача, которая требует точности и соблюдения стандартов. Использование веб-приложений и ботов позволяет автоматизировать этот процесс, снижая вероятность ошибок и экономя время пользователей.

2. Потребности IT рынка: разработчики Web-приложений и Telegram-ботов востребованы, так как компании все чаще используют автоматизацию для улучшения взаимодействия с клиентами и оптимизации бизнес-процессов. Современные веб-приложения могут предложить интуитивно понятный интерфейс с возможностью интерактивного ввода данных, что упрощает процесс для пользователей. Telegram-боты могут использовать кнопки и меню для упрощения взаимодействия. Все это позволяет эффективно обрабатывать запросы, предоставлять информацию и улучшать клиентский сервис, что делает специалистов по их разработке востребованными на рынке труда.

3. Практическая значимость: автоматизация и упрощение процесса создания корректных ссылок. Такой сервис помогает студентам и научным сотрудникам сэкономить время. Автоматическое оформление позволяет избежать рутинной работы и сосредоточиться на содержании исследования. Требования ГОСТ к оформлению ссылок могут быть сложны и изменяться. Веб-приложение и Telegram-бот смогут обеспечить актуальность и корректность оформления, автоматически адаптируясь к изменениям в стандартах. Помогает студентам лучше понять структуру и форматирование ссылок, что способствует их обучению.

.

4. Личный интерес и потенциальные перспективы: Соприкасается с профессиональной областью, личностно и практически значима, являюсь аспирантом 3 года выпуска, занимаюсь научной деятельностью. При написании работ всегда имеются требования к оформления источников определённым образом, зависящим от их типа. Сервис поможет сократить время на оформление библиографического списка в несколько раз, как для деятелей науки, так и студентов.

Таким образом, сервис для оформления списка литературы значительно облегчает работу пользователей, повышая качество и скорость подготовки научных материалов, тем самым выбор нашей темы обусловлен ее актуальностью, потребностями рынка, практической значимостью и личным интересом, что делает эту тему подходящей для проведения дипломной работы.

**Определение цели и задач исследования:**

Цель исследования: Провести практический анализ удобства и эффективности разработки Web-приложение на базе разных фреймворков и телеграмм-бота по оформлению списка литературы согласно требованиям ГОСТ.

Задачи исследования:

1. Создание интерфейса для ознакомления и ввода информации о книгах, статьях и других источниках, включая авторов, названия, годы публикации и т.д..

2. Разработка алгоритма, который будет автоматически форматировать введенные данные в соответствии с требованиями ГОСТ.

3. Создание удобного и интуитивно понятного интерфейса для взаимодействия с пользователями, включая команды и меню.

4. Написание дипломной работы: Составить дипломную работу, включающую в себя введение, обзор литературы, методологию и результаты исследования, анализ результатов, выводы и рекомендации.

Цели и задачи исследования направлены на создание нашего проекта, перспективы исследования заключаются в разработке средств оценки эффективности и последующей доработке сервисов.

**2. Основные понятия и определения**

**Основные понятия в создании Telegram-ботов с использованием библиотеки Aiogram включают:**

1. **Веб-приложение**: Программное приложение, доступное через веб-браузер и работащее на сервере.
2. **Django**: Высокоуровневый веб-фреймворк на Python, который упрощает создание сложных веб-приложений благодаря встроенным компонентам и инструментам.
3. **Flask**: Легковесный веб-фреймворк на Python, который предоставляет минималистичный подход к созданию веб-приложений и API.
4. **FastAPI**: Современный веб-фреймворк для создания API на Python, который основан на стандартных аннотациях Python и поддерживает асинхронное программирование.
5. **Telegram-бот**: Программа, которая взаимодействует с пользователями через Telegram, выполняя автоматические задачи и обрабатывая команды.
6. **API (Application Programming Interface)**: Интерфейс, который позволяет различным программам взаимодействовать друг с другом, отправляя и получая данные.
7. **ORM (Object-Relational Mapping)**: Технология, которая позволяет работать с базами данных на уровне объектов, абстрагируя SQL-запросы.
8. **База данных**: Структурированное хранилище данных, используемое для хранения информации о пользователях, литературе и других элементах.
9. **Сервер**: Компьютер или система, предоставляющая ресурсы и услуги для других компьютеров или клиентов в сети.
10. **HTTP (Hypertext Transfer Protocol)**: Протокол, который используется для передачи данных в Интернете, включая запросы и ответы между клиентами и серверами.
11. **Frontend**: Часть веб-приложения, с которой взаимодействует пользователь, включая интерфейс и элементы дизайна.
12. **Backend**: Серверная часть веб-приложения, отвечающая за обработку данных и логику приложения.
13. **Маршрутизация**: Процесс определения, как обрабатывать входящие запросы на основе URL и методов HTTP.
14. **Клавиатура (Keyboard)**: Элементы интерфейса в Telegram, которые позволяют пользователям взаимодействовать с ботом, выбирая опции или отправляя команды.
15. **Асинхронное программирование**: Метод программирования, который позволяет выполнять несколько задач одновременно, улучшая производительность приложений.
16. **Документация**: Описание функционала и использования приложения, включая инструкции и примеры.
17. **Валидация данных**: Процесс проверки введенной пользователем информации на корректность и соответствие заданным правилам.
18. **Стандарты ГОСТ**: Нормативные документы, регулирующие оформление научных работ и список литературы в России.

**3. Методы и подходы к разработке**

1. **Анализ требований:** Определение функциональных и нефункциональных требований к боту, включая необходимые команды, форматирование ссылок по ГОСТ и пользовательский интерфейс.
2. **Проектирование архитектуры**: Использование модульного подхода, разделив код на логические модули (например, обработчики, модели, маршруты, интерфейсы, управление состоянием, работа с базой данных). Это улучшит читаемость и поддержку кода.
3. **Frontend**: Создание интуитивно понятного интерфейса для пользователей с возможностью ввода данных и просмотра оформленных списков литературы.
4. **Интерфейс пользователя:** Проектирование интуитивно понятного интерфейса с использованием кнопок и меню, чтобы пользователи могли легко вводить данные о литературе и получать результаты.
5. **Форматирование данных:** Реализация алгоритма для автоматического форматирования введенной информации в соответствии с требованиями ГОСТ.
6. **Подключение к базам данных**: Использование ORM (например, SQLAlchemy или Django ORM) для работы с базами данных, хранения информации о пользователях и литературе.
7. **Бот-команды:** Создание команд для взаимодействия с пользователями, таких как /start, /help, и команд для оформления списка литературы. Интерфейс бота: Использование клавиатур и инлайн-кнопок для упрощения ввода данных и выбора опций.
8. **Документация:** Создание подробной документации для пользователей и разработчиков, включая инструкции по использованию бота и описание его функционала.

**4. Обзор популярных инструментов для разработки веб-приложения с использованием Django, Flask и FastAPI, а также Telegram-бота**

Обзор популярных инструментов для разработки Telegram-бота с помощью библиотеки Aiogram включает в себя следующие ключевые компоненты и ресурсы:

1. **Python**: Основной язык программирования для разработки ботов на Aiogram. Python обладает простой и понятной синтаксисом, что делает его идеальным для быстрого прототипирования и разработки.
2. **Django ORM:** Встроенный ORM в Django, который позволяет легко работать с базой данных, используя объекты Python.
3. **SQLite:** Легковесная база данных, которая хорошо подходит для разработки и тестирования
4. **Swagger/OpenAPI:** Инструменты для автоматической генерации документации API, которые могут быть интегрированы с FastAPI.
5. **SQLAlchemy**: ORM-библиотека для Python, которая упрощает работу с базами данных, позволяя разработчикам манипулировать данными на уровне объектов вместо написания сырого SQL-кода.
6. **Git**: Система контроля версий, которая позволяет отслеживать изменения в коде, совместно работать над проектом и управлять версиями бота.
7. **PyCharm**: Популярные IDE (интегрированные среды разработки), которые предлагают инструменты для разработки, отладки и тестирования Python-кода, поддерживают работу с Git и имеют плагины для улучшения производительности.
8. **Telegram Bot API Documentation**: Официальная документация Telegram Bot API, которая содержит всю необходимую информацию о доступных методах, параметрах и примерах использования API.

**5. Проектирование приложения**

**1. Функциональные требования**

* **Оформление списка литературы**: Автоматическое создание списков литературы в соответствии с требованиями ГОСТ.
* **Ввод данных**: Пользователь должен иметь возможность вводить информацию о книгах, статьях и других источниках.
* **Вывод данных**: Возможность получать оформленный список литературы из введенной информации.
* **Интерактивный интерфейс**: Удобный пользовательский интерфейс для ввода и редактирования данных.

**2. Архитектура приложения**

* **Backend**:
  + **Django**: Для создания полноценного веб-приложения с админ-панелью и ORM.
  + **Flask**: Для создания легковесного API или микросервиса, если нужно минимальное решение.
  + **FastAPI**: Для высокопроизводительного API, если требуется асинхронная обработка.
* **Telegram-бот**: Использование библиотеки Aiogram для создания бота, который будет взаимодействовать с пользователями.

**3. Моделирование будущих данных**

**Например:**

* **Модель "Источник"**:
  + Название
  + Автор(ы)
  + Год издания
  + Издательство
  + Тип (книга, статья и т.д.)
  + Другие атрибуты согласно требованиям ГОСТ

**4. Схема базы данных**

* Разработка схемы базы данных.

**4. Реализация функционала**

* Создание RESTful API для взаимодействия с фронтендом или Telegram-ботом.
* Реализация алгоритма для форматирования списка литературы в соответствии с требованиями ГОСТ.
* Валидация вводимых данных на корректность и соответствие стандартам.

**5.1 Создание пользовательского интерфейса**

* Проектирование интуитивно понятного интерфейса для ввода данных о источниках.
* Реализация форм для добавления, редактирования и удаления информации.

**5.2. Интерфейс Telegram-бота**

* Создание команд и меню для взаимодействия с пользователями.
* Реализация обработки сообщений и команд для добавления источников и получения оформленного списка.

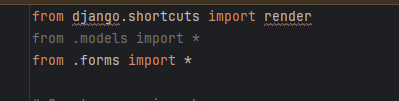
**6. Документация**

* Создание документации для разработчиков и пользователей, включая инструкции по использованию приложения.

**6. Разработка в соответствии с созданной документацией**

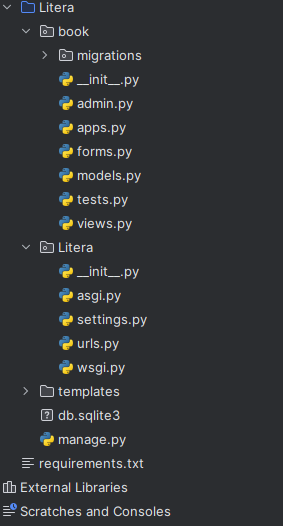
**1. Подготовка окружения**

**1.1. Установка необходимых инструментов**

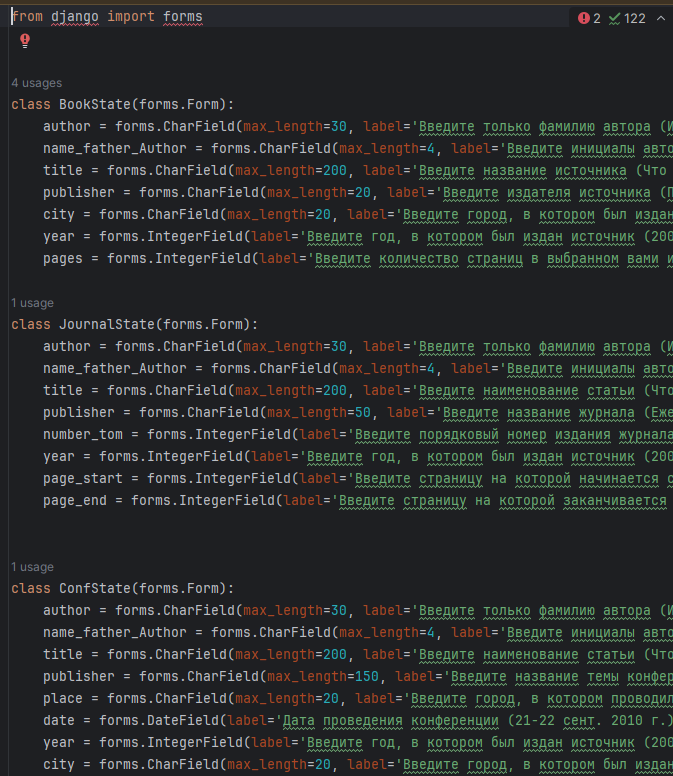
****

**2. Разработка веб-приложения**

**2.1. Создание проекта Django**

****

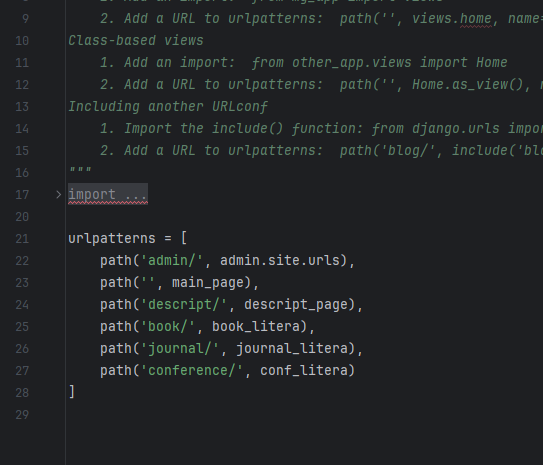
**2.2. Определение форм**

****

**2.3. Настройка их обратной связи после заполнения и вывод результата на экран**

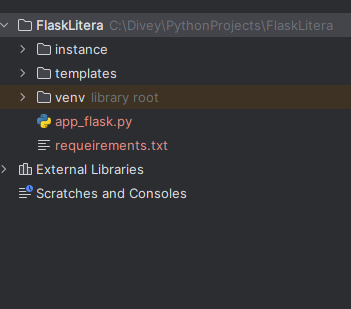
****

**2.5. Настройка URL-ов**

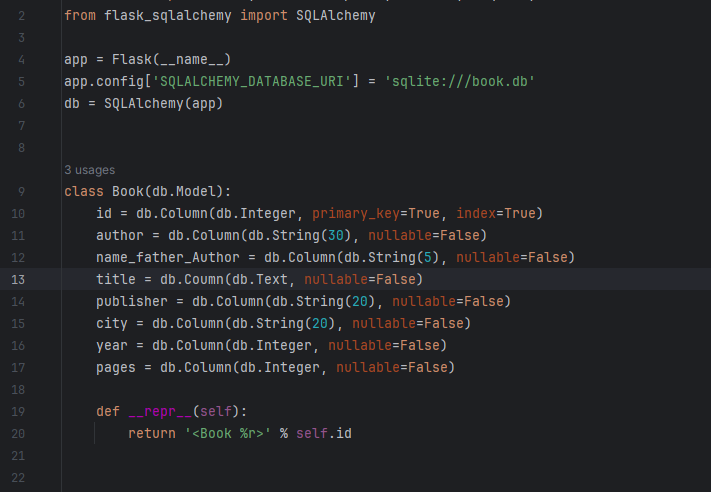


**3. Разработка Flask API**

**3.1. Создание проекта Flask**

****

**3.2. Реализация CRUD-операций**

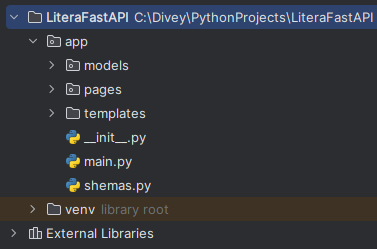


**3.3. Настройка обратной связи от базы данных после заполнения и вывод результата на экран**

****

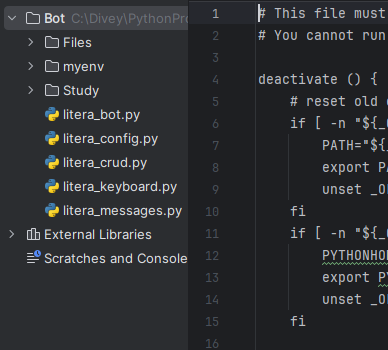
**4. Разработка FastAPI**

**4.1. Создание проекта FastAPI**



**5. Разработка Telegram-бота**

**5.1. Создание бота**



**8. Заключение**

**Основные выводы:**

1. **Django как оптимальный выбор**:
   * **Полнофункциональность**: Django предоставляет все необходимое для быстрого создания полноценного веб-приложения.
   * **Структурированность**: Благодаря строгой архитектуре и принципам MVC, Django позволяет легко поддерживать код и расширять функциональность приложения по мере необходимости.
2. **Flask**:
   * **Flask**: Подходит для создания легковесных приложений и микросервисов, где требуется больше гибкости и минимализма. Однако, если проект требует более сложной логики и интеграции, Flask может потребовать больше времени на разработку дополнительных компонентов.
3. FastAPI отличный выбор для бэкэнда:
   * Для разработки полноценного приложения, которое включает в себя как бэкэнд, так и фронтэнд, может быть разумно использовать FastAPI в сочетании с другими технологиями, такими как Django или Flask для бэкэнда, а также библиотеками для фронтэнда (например, React или Vue.js), чтобы обеспечить полное решение для пользователей.
4. **Telegram-бот**:
   * Интуитивно понятен для пользователей, но схема создания досрочно трудоемка, а на выхлде малоэффективна, возможно потому что не использовал БД.

**Заключение:**

В результате разработки приложения с использованием Django, Flask и FastAPI, а также Telegram-бота для оформления списка литературы, можно сделать вывод, что Django является наиболее удобным и целесообразным выбором для создания полноценного веб-приложения. Его мощные возможности, встроенные функции и безопасность делают его идеальным для разработки сложных и многофункциональных приложений, таких как система управления литературой. Использование Flask и FastAPI также имеет свои преимущества, но их применение более целесообразно в специфических задачах. В конечном итоге, выбор инструмента зависит от конкретных требований проекта и предпочтений команды разработчиков