Дипломная работа по теме: Анализ и сравнение написания web-приложений с использованием разных фреймворков

1. **Введение**

Обоснование выбора темы:

1. В современном мире web-приложения занимают значительную часть ежедневно используемой людьми информации, благодаря чему, такое обыденное использование данного сектора является крайне важной задачей многих современных программистов.
2. Потребности рынка: в нынешнее время, когда веб-приложения становятся неотъемлемой частью бизнеса, роль Python и Django трудно переоценить. Они способствуют созданию высококачественных решений, которые могут адаптироваться к меняющимся требованиям пользователей и рынков. С помощью этих инструментов разрабатываются разнообразные проекты — от простых сайтов до сложных корпоративных систем.
3. Практическая значимость: работа в области разработки веб приложения позволит применить полученные знания на практике и даст опыт работы с современными инструментами. Это позволит создавать интересные и полезные для пользователей возможности.
4. Личный интерес и потенциальные перспективы: выбор данной темы позволил мне изучить вопрос создания web-приложений глубже благодаря тому, что база во время курсов была изучена достаточно хорошо. Использование различных инструментов для улучшения пользовательского опыта с небольшим размахом, дал мне ценный опыт, который востребован на рынке труда, что открывает перспективы для карьерного роста и развития.

Таким образом, выбор темы " Анализ и сравнение написания web-приложений с использованием разных фреймворков." обусловлен ее актуальностью, потребностями рынка, практической значимостью и личным интересом, что делает эту тему подходящей для проведения дипломной работы.

**Определение цели и задачи исследования:**

1. Обзор фреймворков для написания веб приложений и их возможности: провести анализ существующих инструментов, выбрать наиболее подходящие под имеющуюся задачу.
2. Разработать базовое представление о будущем web-приложении, создать основу в виде страницы регистрации, входа и основной страницы.
3. Улучшить пользовательский опыт и изучить более углублённо библиотеки Django и их взаимоотношение с html, css.
4. Написание дипломной работы: составить дипломную работу, включающую в себя введение, методологию и результаты исследования, анализ результатов, выводы и рекомендации.
5. **Основные понятия и определения**

**Обзор основных понятий в области разработки web-приложений**

1. Фреймворк (Framework): Программная платформа, которая предоставляет готовые компоненты и инструменты для разработки приложений. В контексте веб-приложений часто используются Django, FastAPI или Flask.
2. Веб-приложение (Web Application): Программное приложение, которое работает на веб-сервере и доступно через браузер. В контексте данного проекта, веб-приложение будет использоваться для сравнения и демонстрации эффективности различных предобученных моделей распознавания объектов.
3. База данных (Data base): упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД).
4. Языки программирования (programming languages): На стороне сервера обычно используются HTML, CSS и Python. HTML отвечает за структуру контента, CSS — за его стилизацию, а Python добавляет интерактивность.
5. **Методы и подходы к разработке**

**Выбор фреймворка**

Frontend и Backend: фронтенд (интерфейс пользователя) и бэкенд (серверная логика). Фронтенд может быть разработан с использованием HTML, CSS, а бэкенд может быть реализован на Python с использованием фреймворков, таких как Flask, Django, FastAPI. В данном случае мной был выбран Django, так как он имеет множество преимуществ, которые мне было интересно изучить.

**Обеспечение безопасности**  
  
Аутентификация и авторизация: можно настроить авторизацию пользователей, чтобы, например каждый пользователь был уникальным и имел возможность работать в приложении от своего лица.

1. **Обзор популярных инструментов для разработки разработки веб-приложений на Python**

Основные – Django, Flask, FastAPI

**Django**  
  
Django —фреймворк для веб-разработки на Python.  
  
Основные возможности:  
  
· Встроенная страница администрирования для управления данными.  
  
· ORM (Object-Relational Mapping) для работы с базами данных.  
  
· Поддержка аутентификации пользователей и управления доступом.  
  
· Мощная система маршрутизации URL.  
  
· Шаблонизатор для создания HTML-шаблонов.  
  
Особенности:  
  
Идеально подходит для больших и сложных проектов благодаря своей полной экосистеме.

**Flask**  
  
Это фреймворк для веб-разработки на Python. Обеспечивает простоту и гибкость.  
  
Основные возможности:  
  
Минималистичный подход: только основные компоненты без лишних зависимостей.  
  
Высокая гибкость для добавления нужных библиотек и модулей.  
  
Jinja2 для шаблонов и Werkzeug для маршрутизации и обработки запросов.  
  
Особенности:  
  
Подходит для небольших и средних проектов, где требуется большая гибкость.

**FastAPI**  
  
Это современный, высокопроизводительный фреймворк для создания API с Python 3.7+ на основе стандартов OpenAPI и JSON Schema.  
  
Основные возможности:  
  
· Высокая производительность благодаря использованию асинхронного программирования (async/await).  
  
· Автоматическая генерация интерактивной документации API.  
  
· Поддержка валидирования и аннотаций типов данных.  
  
· Интеграция с различными библиотеками и инструментами, такими как Pydantic и SQLAlchemy.  
  
Особенности:  
  
Часто используется для создания API и микросервисов.

1. **Проектирование приложения**

**Планирование и анализ требований**

Выбор фреймворка и инструментов: Определение наиболее подходящего фреймворка (например, Django, Flask или FastAPI) и инструментов для разработки.  
  
Определение структуры приложения: Разработка схемы архитектуры, включающей фронтенд, бэкенд, базу данных.

Разработка прототипа  
  
Реализация основного функционала: Создание базовой версии приложения поэтапно создавая страницы с минимально необходимым функционалом для тестирования и демонстрации.

Разработка полноценно работающей базовой модели

Создание API для взаимодействия с фронтендом: Разработка API для обработки запросов от пользовательского интерфейса. Параллельная настройка зависимостей

Реализация пользовательского интерфейса: Создание формы создания поста в которую входят: создание названия и описания, возможность для загрузки изображений. Отображение результатов взаимодействия с пользователя с интерфейсом. Отображение окна онлайн пользователей. Улучшение визуальной части приложения

**Технические требования**  
  
Фронтенд: использовать HTML, CSS  
  
Бэкенд: Python с использованием фреймворков Django.  
  
База данных: Использование базы данных (например, SQlite3) для хранения данных о пользователях и постах.

API: Разработка API для взаимодействия между фронтендом и бэкендом.

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

**Разработка**

**Планирование разработки**  
  
Разработка была разделена на несколько основных этапов: проектирование интерфейса, реализация серверной логики и улучшение основного функционала. Планирование задач и отслеживание прогресса осуществлялось во время разработки.

Бекэнд-разработка:

Реализована серверная логика с использованием фреймворка Django.  
  
Создана главная страница, которая встречает пользователя на основной ссылке “/”.

Создана страница регистрации с использованием встроенной в Django модели и подгоном её под необходимые нужны.

Создана страница авторизации.

Подключена и протестирована база данных.

Создана основная страница с постами.

Создана страница для загрузки постов, после тестировки была интегрирована возможность добавлять изображения при создании постов.

Дальнейшее улучшение главной страницы с использованием встроенных в Django инструментов, таких как пагинация, возможность взаимодействия с БД и пр.

Настроены маршруты для обработки загрузок изображений, взаимодействия с основной страницей.

Фронтенд-разработка:

Создан пользовательский интерфейс с использованием HTML для шаблонов и стили css для стилизации.

1. **Анализ и интерпретация результатов**

Использование Django значительно упростило процесс разработки web-приложения благодаря его богатому набору встроенных функций, таких как система аутентификации пользователей, ORM. Это позволило сократить время создания и тестирования приложения. Также работа с Django обеспечила доступ к большому количеству ресурсов, обучающих материалов и библиотек, что облегчило процесс разработки и решение возникающих проблем. Сообщество разработчиков активно делится знаниями и опытом, что способствует быстрому обучению.

**Рекомендации по выбору API в зависимости от проекта**

Хотя создаваемый форум был одностраничным, архитектура Django позволяет легко масштабировать приложение в будущем, добавляя новые функции и улучшения, такие как поддержка многоуровневых обсуждений, интеграция с внешними API, или возможность добавления более глубокого взаимодействия пользователей.

1. **Заключение**

**Обзор выполненной работы**

Проектирование и разработка веб-приложения для были успешно завершены в соответствии с изначально созданной документацией. Приложение включает функционал создания постов с возможностью загрузки изображений, пускай и не прямое, но взаимодействие пользователей онлайн.

Реализованное приложение достигло того функционала, который я хотел от него видеть и во время разработки благодаря идеям добавить данный функционал, мной были более углублённо изучены библиотеки Django.

**Дальнейшие планы**

Данная модель пускай и крайне проста, но благодаря многогранности Django, её можно развить как минимум до полноценного форума.

Планируется добавление новых функций для развития взаимодействия пользователей, таких как: личные сообщения, комментарии под постами, а так же реакции там же.