

Дипломная работа

Панова Александра Анатольевича

на тему:

Анализ и сравнение написания web-приложений с использованием разных фреймворков: разработать простые веб-приложения с использованием Django, Flask и FastAPI, провести их сравнение

г. Гатчина 2025г.

**1. Введение**

Современные технологии искусственного интеллекта и машинного обучения активно внедряются в различные сферы жизни, от медицины до бизнеса. Одним из ключевых направлений является интеграция пред обученных моделей в веб-приложения, что позволяет создавать интеллектуальные системы, доступные через интернет. Python, как один из самых популярных языков программирования для разработки и анализа данных, предоставляет широкие возможности для реализации таких решений.

**2. Определение задач исследования**

В данной дипломной работе мы напишем web приложение для загрузки, просмотра и удаления фотографий на основе трех популярных фреймворков: Анализ и сравнение написания web-приложений с использованием разных фреймворков: Разработать простые веб-приложения с использованием Django, Flask и FastAPI, провести их сравнение.. Проведем сравнение работы приложения и анализ работы фреймворков.

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

Django

Высокоуровневый фреймворк, который подходит для создания сложных веб-приложений.

Flask

Микрофреймворк, который предоставляет базовые инструменты для разработки.

FastAPI

Современный фреймворк для создания высокопроизводительных API.

В результате дипломной работы получили приложение на основе каждого из вышеперечисленных фреймворков.

**4. Сравнение фреймворков Django, Flask и FastAPI**

**Производительность**

Производительность Django ниже, чем у Flask и FastAPI, из-за его монолитной архитектуры.

Однако для большинства веб-приложений производительности Django достаточно, особенно если использовать кэширование и оптимизировать запросы к базе данных.

Flask быстрее Django, так как он минималистичен и не имеет лишних зависимостей.

Производительность зависит от выбранных библиотек (например, SQLAlchemy для ORM или Jinja2 для шаблонов).

FastAPI — самый быстрый из трёх фреймворков благодаря асинхронной обработке запросов и использованию ASGI-сервера (например, uvicorn).

Идеально подходит для высоконагруженных API и приложений, работающих с большим количеством одновременных запросов.

**Удобство разработки**

Django обладает встроенной ORM с поддержкой миграций, что упрощает работу с базой данных.

Встроенная административная панель в Django позволяет быстро создавать интерфейсы для управления данными. Также придает удобство в работе готовые инструменты для работы с формами и валидацией данных, есть встроенный движок шаблонов (Django Templates) для создания HTML-страниц и автоматическая маршрутизация через файл urls.py.

Flask не имеет встроенной ORM, но можно использовать SQLAlchemy или другие библиотеки. Административную панель нужно создавать самостоятельно или использовать сторонние расширения (например, Flask-Admin). Имеется Flask-WTF для работы с формами. Jinja2 используется по умолчанию, однако, нужна ручная настройка маршрутов через декораторы.

FastAPI также не имеет встроенной ORM, но можно использовать SQLAlchemy, Tortoise-ORM или другие библиотеки. Административную панель тоже нужно создавать самостоятельно. Нет встроенной поддержки форм, но можно использовать сторонние библиотеки. FastAPI не предназначен для рендеринга HTML-страниц, но можно использовать Jinja2. Декларативная маршрутизация через декораторы.

**Гибкость**

Django меньше гибкости из-за монолитной архитектуры, однако он предоставляет множество готовых решений, что ускоряет разработку.

Flask имеет довольно высокую гибкость: можно выбрать любые библиотеки и инструменты. Подходит для небольших проектов или случаев, когда требуется точная настройка.

FastAPI - высокая гибкость, но ориентирован на создание API. Не подходит для классических веб-приложений с рендерингом HTML-страниц.

**Асинхронность**

В Django поддержка асинхронности ограничена. Версия 3.1+ имеет асинхронные представления, но они всё ещё не так мощны, как в FastAPI.

Flask поддерживает асинхронность, но её использование ограничено и требует дополнительных настроек.

FastAPI имеет полную поддержку асинхронных операций, он идеально подходит для приложений, работающих с WebSockets, фоновыми задачами и высоконагруженными API.

**Скорость разработки**

Django: быстрая разработка благодаря множеству встроенных инструментов. Подходит для стандартных задач, таких как CRUD-операции, аутентификация, управление пользователями.

Flask: требует больше времени на настройку, так как многие функции нужно добавлять вручную. Подходит для небольших проектов или случаев, когда требуется точная настройка.

FastAPI: быстрая разработка API благодаря автоматической генерации документации. Не подходит для классических веб-приложений с рендерингом HTML-страниц.

**Документация и сообщество**

У Django обширная документация и большое сообщество, множество примеров и готовых решений.

Flask хорошая документация, но меньше готовых решений по сравнению с Django. Сообщество активное, но меньше, чем у Django.

FastAPI отличная документация с акцентом на современные технологии. Сообщество растёт, но пока меньше, чем у Django и Flask.

**Пример использования**

Django: идеально для больших веб-приложений, таких как интернет-магазины, социальные сети, блоги.

Flask: идеально для небольших проектов, микросервисов, REST API.

FastAPI: идеально для высоконагруженных API, машинного обучения, IoT.

9. Заключение

Если вам нужна полноценная платформа с множеством встроенных инструментов.

Для крупных проектов, где важна скорость разработки и готовые решения, тогда лучше использовать Django.

Если вам нужна гибкость и контроль над каждым аспектом приложения, необходимо сделать небольшой проект или микросервис, тогда лучше использовать Flask.

Если вам нужна высокая производительность и асинхронная обработка запросов, создание API, микросервисов и приложений, работающих с большим количеством одновременных запросов, тогда выбор лучше сделать в пользу FastAPI.

**4. ВЫВОД:**

Выбор фреймворка зависит от ваших требований к проекту. Если вам нужна универсальность и готовые решения, выбирайте Django. Если важна гибкость и контроль, выбирайте Flask. Если ваш проект требует высокой производительности и асинхронности, выбирайте FastAPI. Хотя программирование интересно именно как раз тем, что в нем нет одного ответа на один вопрос. Какой фреймворк выбирать, решает программист. При этом, одну задачу можно решить разными способами и с помощью разных фреймворков, что и доказала эта дипломная работа.