**ДИПЛОМНАЯ РАБОТА**

по теме: «**Анализ и сравнение написания веб-приложений с использованием разных фреймворков**»

Автор: Баранов Владимир Константинович

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc183332590)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 5](#_Toc183332591)

[2. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 7](#_Toc183332592)

[3. ПРАКТИЧЕСКАЯ ЧАСТЬ 9](#_Toc183332593)

[3.1 Требования к интернет-ресурсу 9](#_Toc183332594)

[3.2 Результаты реализации 9](#_Toc183332595)

[ЗАКЛЮЧЕНИЕ 12](#_Toc183332596)

[ПРИЛОЖЕНИЯ 13](#_Toc183332597)

# ВВЕДЕНИЕ

Веб-интерфейсы представляют собой мощный инструмент анализа данных, визуализации и взаимодействия с пользователями. Как правило, наибольшей популярностью пользуются те ресурсы, которые представляют конкретные функциональные элементы, представленные в наглядной и интуитивно близкой форме. Для тренировки создания таких ресурсов, как правило, можно решать относительно простые задачи, в рамках которых практиковаться в визуализации и интерпретации конечных результатов решения таких задач.

Одним из наиболее распространённых классов задач программирования являются веб-приложения. В данной работе хотелось бы рассмотреть задачу по разработке и сравнению трёх простых веб-приложений с использованием фреймворков Django, Flask и FastAPI и языков HTML, Python и CSS с параметрами визуализации: возможность ввода данных со страницы, нахождение пользователя в базе данных по заданным параметрам, предоставление своеобразного личного кабинета, а также имитация процесса предоставления услуг и товаров.

Таким образом, можно сформулировать **цель работы** — разработка трёх простых веб-приложений на трёх разных фреймворках (Django, Flask и FastAPI) каждое из которых будет иметь отличный друг от друга функционал:

- имитация сайта ресторана с возможностью заказа еды;

- интернет-ресурс по созданию заметок с возможностью прикрепления фотографий;

- приложение для бронирования отелей.

В соответствии с целью можно сформулировать следующие **задачи**:

1. Провести теоретический обзор по области разработки продукта и по его теме.

2. Составить список основных требований к разрабатываемому интернет-ресурсу.

3. Разработать три многостраничных веб-интерфейса посредством языков HTML, Python и CSS, и их инструментов

4. Разработать веб-приложения с использованием фреймворков Django, Flask и FastAPI.

5. Составить отчёт по выполненным работам со скриншотами результатов разработки и исходным кодом страниц.

Далее в отчёте будут приведены результаты выполнения данных задачи и итоги разработки продукта.

# 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Задача по предоставлению услуг посредством веб-приложений не является уникальной в контексте имеющихся по ней решений. Рассмотрим наиболее популярные интернет-ресурсы, решающие поставленные задачи.

**Сайт ресторана «Вкусно и точка».**

Данный сайт обладает широким спектром функциональных возможностей, что позволяет работать с его интерфейсом для непосредственного заказа еды и дополнительного спектра раскрываемых системой задач.

Скриншот ресурса представлен на Рисунке 1.1.

|  |
| --- |
|  |
| **Рисунок 1.1 – Блок сайта «Вкусно и точка»** |

В качестве основного недостатка данного ресурса можно выделить, как обилие рекламы на страницах сайта (не была включена в скриншот), так и отсутствие возможности настройки тёмной темы, которая существует уже на многих интернет-ресурсах.

**Онлайн-бронировщик «Яндекс Путешествия».**

Данный веб-интерфейс функционально не похож на предыдущий, но является не менее функциональным, так как он не представляет возможности заказа еды, а предназначен только для бронирования отелей, гостиниц и др.

На Рисунке 1.2 представлен скриншот работы данного ресурса.

|  |
| --- |
|  |
| **Рисунок 1.2 – Блок сайта «Яндекс Путешествия»** |

В отличии от предыдущего ресурса здесь присутствует возможность включения тёмного оформления, построение параметров графика, но большим минусом является обилие рекламы на сайте, которое без специальных сервисов блокировки, затрудняет пользование данным ресурсом.

Таким образом, на текущий момент существует большое количество ресурсов, решающих задачи по предоставлению услуг посредством веб-приложений, но практика в решении данных задач и создание подобного рода полезных ресурсов позволяет улучшить свои навыки в создании веб-интерфейсов, а также отработать изученные механизмы в рамках прохождения программы «Python-разработчика» на практике.

# 2. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

**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.

|  |
| --- |
|  |
| **Рисунок 2.1 – Логотип Django** |

**Flask -** 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** - 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.

# 3. ПРАКТИЧЕСКАЯ ЧАСТЬ

## 3.1 Требования к интернет-ресурсу

По поставленным задачам можно сформулировать следующие требования к разрабатываемым продуктам:

1. Наличие на странице возможности ввода данных от пользователя (имя, фамилия, пароль и др.).

2. Наличие на странице кнопки для отправки фото на сервер.

3. Наличие на странице кнопки для выхода из одного аккаунта и входа под другим.

4. Наличие на странице возможности удаления или оплаты заказа из своей корзины, удаления файлов из личного кабинета, а также просмотра текущих заказов.

Реализация данных пунктов при наличии соответствующих стилистических особенностей и будет являться готовым программным решением для отработки навыков по языкам HTML, Python и CSS и фреймворкам Django, Flask и FastAPI.

## 3.2 Результаты реализации

В рамках выполнения дипломной работы были выполнены поставленные задачи перед системой и программно реализованы её основные элементы. Для выполнения работы была выбрана среда разработки PyCharm Community Edition, были использованы языки программирования HTML, Python и CSS, и фреймворки Django, Flask и FastAPI соответственно. Исходный код страниц представлен в приложении.

Результат работы приведён на Рисунках 3.1-3.3.

|  |
| --- |
|  |
| **Рисунок 3.1 – Результат реализации страницы личного кабинета пользователя с использованием фреймворка Django** |

|  |
| --- |
| Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение  Автоматически созданное описание |
| **Рисунок 3.1 – Результат реализации страницы для создания заметок с использованием фреймворка Flask** |

|  |
| --- |
| Изображение выглядит как текст, снимок экрана, Веб-сайт, веб-страница  Автоматически созданное описание |
| **Рисунок 3.1 – Результат реализации страницы бронирования отеля с использованием фреймворка FastAPI** |

Как видно из рисунков, на страницах представлены все необходимые функциональные части с соответствующими подписями. Результаты взаимодействия с сайтом сохраняются и представляется на отдельной странице пользователя со всеми необходимыми данными для отображения в контексте решаемой задачи.

# ЗАКЛЮЧЕНИЕ

Таким образом, в результате написания дипломной работы был проведён теоретической обзор области разработки продукта, в рамках которого были получены выводы не только о большом количестве имеющихся интернет-ресурсов по теме, но и о практической пользе решения данного цикла задач в рамках изучения программирования.

На основе результатов теоретической части и специфики темы были сформированы основные требования к интернет-ресурсу и описаны в виде текстового набора пунктов-задач перед системой.

Фреймворки Django, Flask и FastAPI были выбраны для реализации самого веб-приложений, содержащих возможности отображения нескольких страниц спроектированных страниц:

• страница основного функционала;

• страница личного кабинета;

• страница обработки данных от пользователя (логин, пароль, имя и т.д.).

Согласно требованиям, были выполнены работы по проектированию и программной реализации страниц интернет-ресурса, функционал которых целиком покрыл поставленные перед системой задачи. Веб-приложения работают корректно и прошли необходимое тестирование.

Фреймворк Django показал высокую удобность работы с базами данных, и хорошую работу при реализации крупных проектов, однако для создания микросервисов выбор будет идти в пользу FastAPI а для средних проектов подойдёт Flask.

Так, в ходе выполнения дипломной работы были закреплены и отработаны навыки теоретического исследования по теме, формирования и составления основных требований к интернет-ресурсам, их проектирования и программной реализации на языках программирования, изученных в течение курса «Python-разработчик».

# ПРИЛОЖЕНИЯ

Приложение 1.А

|  |
| --- |
| Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение  Автоматически созданное описание |
| **Файловая структура проекта на Django** |

Приложение 2.А

|  |
| --- |
|  |
| **Список необходимых библиотек проекта на Django** |

Приложение 1.Б

|  |
| --- |
|  |
| **Файловая структура проекта на Flask** |

Приложение 2.Б

|  |
| --- |
| Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение  Автоматически созданное описание |
| **Список необходимых библиотек проекта на Flask** |

Приложение 1.В

|  |
| --- |
|  |
| **Файловая структура проекта на FastAPI** |

Приложение 2.В

|  |
| --- |
| Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей  Автоматически созданное описание |
| **Список необходимых библиотек проекта на FastAPI** |