Міністерство освіти і науки України

Національний університет «Запорізька політехніка»

кафедра програмних засобів

**ЗВІТ**

з лабораторної роботи № 4

з дисципліни «Архітектура та проєктування програмного забезпечення» на тему:

**«Аналіз та проєктування архітектури веб-додатків на основі предметної області»**

Виконала

Студентка групи КНТ-129 А.К. Яценко

Прийняли:

асистент С.Д. Леощенко

2021

* 1. **Мета роботи**

Навчитися розробляти та проводити аналіз архітектури веб-додатків на основі предметної області.

Навчитися розробляти архітектуру веб-додатків з використанням фреймворків та CMS, провести порівняльний аналіз архітектур веб-додатків, розроблених на основі предметної області та за використання фреймворків або CMS.

* 1. **Тема, обрана для проєктування**

Обрана для проєктування тема – соціальний інтернет-сервіс для фотографів.

* реєстрація та авторизація користувачів;
* публікація власних фотоматеріалів;
* захист авторських прав (захист від копіювання фотоматеріалів) користувачів;
* перегляд та ознайомлення із роботами фотографів;
* можливість створення власних колекцій з фотоматеріалів інших користувачів;
* виставлення оцінок та коментування публікацій;
* опрацювання (адміністратором) коментарів;
* рекомендація та виведення фотографій за рейтингом найбільш популярних;
* адміністративна панель сайту.

Для розробки веб-застосунку було обрано мову програмування Python та фреймворк Django.

Python – це високорівнева мова програмування, його синтаксис легкий для сприйняття. Окрім того, для Python існує багато бібліотек, що полегшують веб-розробку.

Фреймворк Django написаний на мові програмування Python, тож його структура відповідає особливостям мови. Функціональність Django можна розширювати за допомогою чисельних плагінів та бібліотек, що дозволить розробити дійсно зручний веб-застосунок.

* 1. **Схематичне зображення архітектури веб-застосунку на основі предметної області**

Для розроблюваного веб-застосунку обрано архітектуру MVC – Model-View-Controller.

У компоненті Model представлено модель даних (базу даних). Тобто у цій компоненті знаходиться інформація щодо користувачів (логіни, паролі), публікацій (фотографія, кількість вподобань, автори, коментарі), колекцій користувачів (назви, фотографії в колекції, індекс першої доданої фотографії), листування користувачів з адміністратором (автори повідомлень, тексти повідомлень, дати відправки). Також у цій компоненті знаходиться логіка застосунку, тобто обробка даних, що вводить користувач.

У компоненті View представлено відображення моделі. Тобто у цій компоненті представлений графічний інтерфейс взаємодії користувача з системою.

У компоненті Controller представлена робота з вхідними даними. Тобто тут буде прийняття та перевірка даних, введених користувачем, та передача даних до моделі.

* 1. **Схематичне зображення архітектури веб-застосунку (на основі фреймворку або CMS)**

Для проєктування архітектури веб-застосунку було обрано фреймворк Django.

На рис. 1 наведено автоматично згенеровану структуру застосунку, що розробляється з використанням Django.

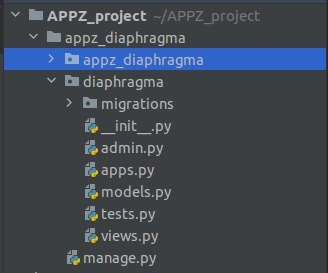


Рисунок 1 - Структура застосунку, що розробляється з використанням Django

Окрім наведених файлів, гарним тоном вважається винесення в окремий файл шару serializers / forms. Також при розробці буде створено папку templates, що буде знаходитися на одному рівні з папкою migrations та файлами застосунку.

Отже, виділимо в застосунках Django кілька шарів:

* front-end / templates;
* migrations;
* admin;
* apps;
* models;
* tests;
* serializers / forms;
* views.

У шарі templates знаходиться графічний інтерфейс користувача у вигляді html-файлів. Тобто тут буде знаходитися базова сторінка сайту та сторінки для реалізації функціональних вимог користувачів.

Шар migrations знаходяться файли міграцій. Ці файли слугують для внесення змін до схеми бази даних та створюються автоматично.

У шарі admin знаходиться код для сторінки адміністратора Django.

Шар models представляє модель даних системи. Тобто у цьому шарі розташовано моделі для таблиць бази даних - інформація щодо користувачів, публікацій, колекцій користувачів, листування користувачів з адміністратором.

У шарі tests знаходяться тестові приклади для перевірки працездатності веб-застосунку.

Шар serializers / forms слугує для валідації та передачі даних, введених користувачем, до бізнес-логіки застосунку для подальшої обробки. У цьому шарі може бути реалізовано перевірку на коректність завантаженого для публікації файлу (чи підходить цей формат файлу для публікації, перевірка розміру файлу), валідація логіну та паролю при реєстрації (не пусте поле, латиниця для паролю) та інше.

Шар views реалізує логіку роботи веб-застосунку. Тут оброблюються вхідні дані та реалізується задача отримання та відображення даних. Тобто у цьому шарі, наприклад, може бути реалізовано алгоритми авторизації та реєстрації; алгоритм створення колекції, додавання публікації, листування та інше.

* 1. **Обґрунтування вибору архітектури веб-застосунку**

Для розробки веб-застосунку було обрано архітектуру на основі фреймворку Django. Загалом архітектура MVC та архітектура на основі Django схожа, але Django дозволяє розділити модель збереження даних системи та логіку роботи застосунку. Це зручніше для роботи з даними, так як у такому разі буде легше сприймати модель зберігання даних та алгоритми роботи застосунку. В тому числі це добре для подальшого розвитку проєкту, так як модифікувати та доповнювати як модель зберігання даних, так і логіку роботи буде легше, ніж коли вони знаходяться в одному компоненті.

* 1. **Висновок**

У результаті роботи я навчилася розробляти та проводити аналіз архітектури веб-додатків на основі предметної області, а також розробляти архітектуру веб-додатків з використанням фреймворків та CMS Було проведено порівняльний аналіз архітектур веб-додатків, розроблених на основі предметної області та за використання фреймворку Django. Для розробки веб-застосунку було обрано архітектуру на основі фреймворку Django, так як дана архітектура дозволяє розділити модель збереження даних системи та логіку роботи застосунку.