Author

MEWADA SHUBBH RAJESH
21F1005150
21f1005150@student.onlinedegree.iitm.ac.in
I'm a Final-year Computer Engineering student at Gujarat Technological University.
I chose to pursue B.S. in Data Science and its Application from IIT-Madras parallelly because of its comprehensive and more intensive focus on Machine Learning and Data Science based curriculum.

Description

As this is a Multi-User Social Media Platform, it should be able to handle substantial amounts of data in an optimised manner and should be visually appealing at the same time. It should also be concerned about the security of the user's sensitive data. I've tried my best to implement them all .

Technologies used

- **For Front-End/User-Interface development:**
  - **HTML, CSS, Bootstrap.**
    - I've used HTML for structuring ".html" pages and designing them using CSS. All the elements, classes and ids used in ".html" file are given specific attributes in a common ".css" file.
    - I've used Bootstrap for improving the Styling , Responsiveness and Aesthetics of the website. Some elements which are included from Bootstrap are Sidebar, Buttons, Icons, etc.
  - **Jinja2.**
    - I'm so glad that we got to use this templating language. It made the structuring of repetitive required elements like
      - Sidebar for user's interaction.
      - A specific interface design which should be constant throughout the user's experience.

      easy and was possible due to the template inheritance feature of Jinja2.
    - I've also used it to execute basic database queries to avoid rendering or redirecting the data to the same ".html" file again and again.
- **For Back-End development:**
  - **Python**
  - **Flask**

    Flask-Bcrypt -(For hashing the sensitive user's information while storing it into the database.)
    Flask-Login -(For improving the session management activities like Sign In and Log Out functionalities.)
    Flask-RESTful -(For building rest API's . This has features similar to ORM mapping functionality of Flask-SQLAlchemy, only difference is that Flask-RESTful ORM needs to return the values in JSON type format as per the methods (POST, GET, PUT, DELETE) used).
    Flask-SQLAlchemy -(This is majorly responsible for performing CRUD operations on the database using various queries via ORM functionality.

- **For Database Management:**
  - **sqlite3**

API Design [(My Published Collection of BLOG LITE API)](#).

To perform CRUD operations on our database using API tools, I've used POSTMAN. There are two most important tables USER and POST. I've performed CRUD operations using respective methods (POST, GET, PUT, DELETE) by passing a JSON file as input with all the required parameters. So Intotal, I've created 8 methods for interacting with USER and POST.

DB Schema Design

PRIMARY KEY FOREIGN KEY

USER

id	Integer
username	String
email	String
image_file	String
password	String

image_file : Profile image of a user.
Constraints:
Unique: Username, email.
Password: It is hashed before storing.

POST

id	Integer
title	String
date_posted	DateTime
content	String
user_id	Integer
image_file	String

Parent Table : USER.
image_file : Post image of a user.
date_posted helps to sort the posts(newest first).

MASTER_SERVENT

id	Integer
master	Integer
servant	Integer

Parent Table : USER.

LIKE

id	Integer
post_id	Integer
user_id	Integer

Parent Table : USER, POST.

COMMENT

id	Integer
post_id	Integer
user_id	Integer
comments	String
date_posted	DateTime

Parent Table : USER, POST.
date_posted helps to sort the comments.

- Special Case: image_file on submission is stored at a special location with a new filename. This filename is then stored as a string format.

Description:

- **USER** : This stores user's data while new user registers for an account. It is also used for checking authentication while user login.
- **POST** : All the details about the post are stored in this table. "user_id" is the Foreign Key referenced from USER which helps us to backtrack for the author of the post.
- **MASTER_STUDENT** : This table is essentially used for storing followers and following data of a particular user. Here both "master" and "servant" are referenced from USER. This table helps us to create the feed page (Home Page) for users.
- **LIKE**, **COMMENT** : This table stores all the activities performed by a single user on another user's post.

Architecture and Features

├——**flaskblog**(This package has all the important files like routing files(routes.py), ORM files(models.py), form files(forms.py) and (init.py))
│ │——**static**(This folder saves all the Front-End development stuff including the ".css" file.)
│ │ ├——**post_images**(All the image files uploaded are stored here, which makes it easier to access.)
│ │ └——**profile_pics**(All the image files uploaded are stored here, which makes it easier to access.)
│ └——**templates**(All the ".html" pages are saved inside this folder.)
│——**run.py** (This file is used to run initialise the localhost server and gets the app running and initialises the package named **flaskblog**.)
│——**api.py** (This file is not a part of website development, it initialises the localhost server and waits for API method calls (POST, GET, PUT, DELETE).)
│——**requirements.txt**(This file saves all packages and dependencies used while developing the project.)
└——**readme.txt**(This file explains how to run the code.)

All the Core, Recommended and Optional Functionalities are successfully implemented in this project. Some Additional functionality which I thought was important are

- Reset Password link sent to the user through Email using **Flask-Mail** -(For sending sensitive password reset mail to the user using SMTP.).
- Tried my best to make this project look aesthetically pleasing and easy to use.

Video [(My Project Presentation Video)](#).