Project Documentation

Author

Name: Shantaprasad Dhananjay Kamat
Roll number: 21f2000765
Student Email: 21f2000765@ds.study.iitm.ac.in
About me:
I am a Statistics student from Savitribai Phule Pune University. I have had some basic experience in programming in Python before, but this is my first experience in creating a full-fledged app on my own. Despite the obvious lack of experience in programming, I have tried my best to compile and present a working application.

Description

The aim of the project was to build BlogLite application, a social media platform, that would allow users to post images and follow other users to view their posts. We were assigned the task of building both the frontend and the backend of the app; the former using VueJS, and the latter using Python, Flask, and its various functionalities. The user should be able to view and edit their own profiles. The user should also be able to view any/all of the posts of the users that they follow in their feed in a specified order, along with their profiles. They should also be able to follow/unfollow any of the users, and like and comment on any post. The user should get a daily reminder on Google Chat. They should be alerted if they have not visited/posted anything. The user should receive a monthly engagement report on the first day of the next month. Posts should be able to be exported in CSV files. Caching should be done for users.

Technologies used

A variety of Python extensions and modules were used to build the backend of the app. Flask, Flask-Login, SQLAlchemy, Bcrypt, DateTime, Flask-WTF, Image, Redis, and Celery to name a few. Each of these packages had its own reasons for being included, for example, Flask-Login was used to set up user login, Bcrypt security was used to provide password authentication to the login process, DateTime was used to obtain the time a post/comment was posted, Image was used for displaying profile pictures and post pictures, Redis and Celery were used to handle scheduling tasks and caching, and the model was set up using a database from Sqlite3, which was manipulated using SQLAlchemy. The entire app was built locally on VSCode, and DB Browser for SQLite3 was used to build the database.

DB Schema Design

There are six main tables in the database, User, Post, Likes, and Comments, Followers and Actions.
User keeps track of all the users who have logged in to the app, with details including username, email, password, and the time that it was created. Email and name, and city are VARCHAR, with the UNIQUE constraint enforced on email and username since they are unique attributes for the user while logging in. The password column is VARCHAR, and it stores a UTF-8 encoding of the password set by the user. The image column is for the profile picture set by the user.
Post keeps track of the title, the post picture, the caption, the poster, and the time it was created.
Comment keeps track of the comments made by the users below the post, the commenter, the post under which it was commented, and the time it was created.
Like keeps track of the likes issued by the users on a post, the liker, the post which was liked, and the time it was done.
The followers table has two columns, follower_username and followed_username which keep track of who is following whom.
There is also an Actions table which keeps track of the user logins and if they have followed anyone with timestamp.
All the database schema is stored in the models.py file. It was a
very intuitive, simple design for the database.

API Design

RESTful APIs were implemented for Users, Posts, Comments, Likes, and Followers. For every function, the API returns the attributes of the respective class as JSON data which in turn is fetched at the client side of the application to implement basic functionalities.

Architecture and Features

The file hierarchy of the project is very basic. The backend and the frontend of the app are separate. There is an __init__.py file for the initialization of the app and the database, and also the Login Manager. There is a models.py file that contains all the database schema. There is a forms.py file for the initialization of wtforms for Login, SignUp, Post, and Follow. The route.py file has all the APIs of the app and is essentially the biggest file of them all. The main.py file is used to run the app. All the Vue templates used for frontend are stored in the views folder in Client.The main feature of the app is that it has CRUD function on all of users, posts, comments and likes. This means, that a user can create, delete and edit posts, write and delete comments, and they can view, update and delete their profiles as well. Scheduling jobs and Cacheing are also done.

Video:

https://drive.google.com/file/d/1m_WO29pwYwkrFZSqLJrFo6ZfwDgoJlVq/view?usp=sharing