Author
Somnath Bose
21F1006656
21f1006656@ds.study.iitm.ac.in
I am a 3rd year undergraduate student from West Bengal. I am currently pursuing B.Tech. in Mechanical Engineering from IIEST Shibpur alongside the Programming and Data Science Online Degree programme at IIT Madras.

Description

The goal of the project is to create a lightweight social media web app. Users can post content consisting of text and images which can be viewed by users who follow them. In addition to this features such as daily reminders, monthly reports and export data.

Technologies used

- Flask- Python framework for back-end.
- Additional Flask libraries- Flask-Security-Too, Flask-SQLAlchemy, Flask-Caching, Flask-Restful
- Redis- For caching and message broker for Celery
- Celery- Message queues for backend jobs
- SQLite3- Database
- VueJS- Javascript framework for frontend.
- HTML for creating web pages and CSS, Bootstrap 5 and Bootstrap Icons for styling

DB Schema Design

Table: auth

- id INTEGER AUTONUMBER PRIMARY-KEY
- username TEXT UNIQUE
- email TEXT UNIQUE
- password TEXT
- active INTEGER
- fs_uniquifier TEXT (used by Flask token based Authentication)
- whooks TEXT (stores webhooks for reminder)
- log_flag INTEGER

Table: posts

- id INTEGER AUTONUMBER PRIMARY-KEY
- username TEXT FOREIGN KEY auth.username
- caption TEXT
- body TEXT
- image TEXT
- timestamp INTEGER

Table: profiles

- id INTEGER AUTONUMBER PRIMARY-KEY
- name TEXT
- dp TEXT

- username TEXT FOREIGN-KEY auth.username
- location TEXT
- about TEXT
- following_count INTEGER
- follower_count INTEGER
- posts_count INTEGER

Table: relations

- id INTEGER AUTONUMBER PRIMARY-KEY
- follower TEXT FOREIGN-KEY auth.username
- followed TEXT FOREIGN-KEY auth.username

The tables “roles” and “roles_user” are not used to store any functionality, and only present as a setup convention for the flask-security library.

API Design

APIs are implemented for getting the logged user, exporting user data, and performing CRUD operations for Posts and Profiles.

- authAPI- returns data of logged user
- feedAPI- generates user specific feed
- profileAPI- CRUD for profiles
- postAPI- CRUD for posts
- searchAPI- returns user search results
- followingAPI- Handles followings
- followerAPI- Handles followers
- exportAPI- Triggers export job
- hookAPI- Handles webhooks

Architecture and Features

The root folder ‘/’ contains the main.py file where the flask app is defined.The controllers are stored in controllers.py and APIs are stored in api.py in application directory ‘/application’. The database functions are stored in db_func.py in application directory ‘/application’.The models are stored in models.py in application directory ‘/application’. The config variables are stored in config.py in application directory‘/application’. The database file is stored in bd directory‘/db’.The VueJS script is stored in spa.js in static directory ‘/static. Permanent image assests are stored in ‘/static/imgs’. Templates for login page, registration page and index page are stored in ‘/templates/security’ and ‘/templates’.

In the backend the main flask app calls for the controllers and apis as per requests from the client, which perform operations on the database and renders the result to the frontend using VueJS component and bootstrap.

In addition to basic profile creation and posting features, the app can export user data as csv, and send reminders and monthly reports.

Video

https://drive.google.com/file/d/1Xi3lZLgiGQMLIGXZ7ljmXajMMGmqa8VK/view?usp=sharing