Author

Swetha Suravajjula
21f1002451
21f1002451@student.onlinedegree.iitm.ac.in
I am a 3rd-year student of Computer Science Engineering at Mahindra University and Data Science student at IIT Madras. I enjoy learning new things and exploring computer science. I also enjoy singing western and Carnatic music.

Description

The goal of this project is to create a web application for creating blogs using the flask framework, similar to social media platforms like LinkedIn and Instagram, with a focus on implementing CRUD operations.

Technologies used

The technologies and libraries I have used to create this application are:
OS : This module is used to connect the current_app root to the static sub_folders for the image uploads
Datetime: This is used for the column in the post model called 'timestamp' for the datatype date
SQLite: Used as a Database Engine, **Python**: Programming language
dateutil.tz: this is used to convert the default timestamp into the local timestamp
Flask: This is used to build this web application using Python
Flask_login: this is used to build up a login framework in this web application
Flask-migrate: This is used to migrate the database in another file with the extension SQLite
Flask_sqlalchemy: this is used to build the database model and add them to the database
Werkzeug. security: this is used for password security such as creating a hash function for the passwords.
Werkzeug. utils: this is used to check whether the image that the user is uploading is a secure image.

DB Schema Design

My database has three tables

- **Followers** (Association table)
  - Attributes
    - follower_id(Foreign Key from User): to capture the followers user_id
    - followed_id(Foreign Key from User): to capture the following user_id

User (table) -· Attributes

id (PK)	username (unique)	password_hash	Post(FK from Post)	Followed(FK Followers)	Profile Pic

Functions to operate on these attributes
remove: this function removes the current_user record from the table if he/she wants to delete his/her account

check_password: checks whether the password that was given at the time of login is similar to the password that he/she added to their record into the database at the time of creating the account
is_following: checks whether the user is following the current_user
follow: this appends the user into the current_user followed attribute of the Users table if a user is not present in this list
unfollow: this removes the user from the current_user followed attribute of the Users table if the user is already present in the list.

Post (table) - Attributes

id (Primary Key)	user_id (FK from User)	timestamp	title	Text	pic

Functions to operate on the attributes - **local_timestamp:** this is to convert the default timestamp into the local timestamp

Architecture and Features

I designed this web application in such a way that I combined my models and controllers into a single Python file called **app.py.**

I made another file called **forms.py** in which I created all of the forms, such as **loginForm, SignUpForm, FollowForm, UnfollowForm, CreatePostForm, SearchForm, and UpdateForm.**

I imported these forms into the app.py file in order to use them to enable application functionality. I also made a templates folder, a static folder, and a migrations folder. The Templates folder contains 13 html templates, whereas the Static folder contains image subdirectories such as profile pics and post pics. The routes I've created are:
home function ('/'), which redirects to the login page,
login function('/login'), which redirects to the general feed upon successful login
signup function('/signup'), which allows users to create an account and is redirected to the login page upon successful registration.

I have used username attribute from user's table as unique so no two users can have the same username otherwise there would be a problem with the password in the backend.
I have disabled the option for users to change their username in the update account option. I have created a **logout** function that redirects to the login page, a **general feed** function that displays the posts of the current user and the people they follow, a save image function that stores images in the static folder, a **create post** function that uses the save image function, a following and follower function that shows a list of users, a **follow** and **unfollow** function to enable a follow button functionality, a **usernames** function to view user profiles, **delete** and **update** post functions, a search function, a delete account and update account function, and I added an additional feature where a user gets a "verified" star on their profile when they reach a certain number of followers.

Video

https://drive.google.com/file/d/1-MRp_94_GKsAj8q2BAcJNvNmMBjyG7S6/view?usp=share_link