Author

Mohammed Asadullah Sayeed
22dp2000212
22dp2000212@student.onlinedegree.iitm.ac.in
I am a 3rd Year Computer Science Engineering student as well and I like to play badminton and am fond of reading self help books.

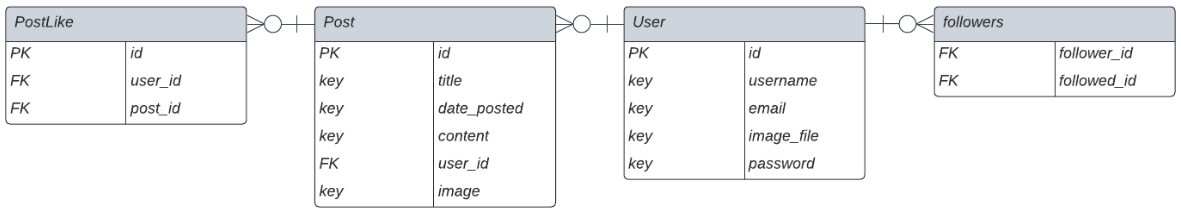
Description

Basically we are trying to create a blogging application that involves user login/registration after login the user can create ,like and update posts. The post contains a title, content and an image file. The user can also delete the post as well. In the account section the user can update his/her details and delete the account. The user can also reset the password. The user can follow and view posts from other users aswell.

Technologies used

- Flask: for backend stuff
- Flask SQLAlchemy: for handling database management
- SQLite: Database
- Flask login: User login management
- Flask WTForms: For form management of the application
- Flask bcrypt: For hashing user passwords and checking it
- Flask migrate: For handling database migrations
- Werkzeug: For handling secure filenames
- PIL: For image saving

DB Schema Design



user	
id	Integer
username	String(20)
email	String(120)
image_file	String(20)
password	String(60)

post	
id	Integer
title	String(100)
date_posted	DateTime
content	Text
user_id	Integer
image	String(200)

post_like	
id	Integer
user_id	Interger
post_id	Interger

followers	
follower_id	Integer
followed_id	Integer

This was done this way to create one to zero or many relationships between Post and PostLike, User and Post, User and followers.

API Design

```
@main.route("/")
@main.route("/home")
```

This route directs to the home page which consists of all posts from different user. Login is required in order to access this route.

```
@users.route("/register", methods=['GET', 'POST'])
```

This route directs the user to the registration page.

```
@users.route("/login", methods=['GET', 'POST'])
```

This route directs the user to the login page.If user is accessing something where login is required it redirects to this route.

```
@users.route("/logout")
```

This route is used to logout the current user.

```
@users.route("/account", methods=['GET', 'POST'])
```

This route directs the user to user account page which is used to update user account details.

```
@users.route("/user/<string:username>")
```

This route directs user to user profile page which consists of all the post by that user.

```
@users.route("/reset_password", methods=['GET', 'POST'])
```

This route directs the user to reset password page which is used to reset the password of the user.

```
@users.route('/follow/<int:user_id>/<action>')
```

This route helps in performing the action of follow a user.

```
@users.route('/<int:user_id>/following')
```

This route directs the user to the user’s list of following

```
@users.route('/<int:user_id>/followers')
```

This route directs the user to user’s list of followers

```
@users.route('/search', methods=['GET']
```

This route is used in searching for a user.

```
@users.route("/delete/<int:id>")
```

This route helps in deleting of the user account

```
@posts.route("/post/new", methods=['GET', 'POST'])
```

This route directs the user to new post page to create a new post.

```
@posts.route("/post/<int:post_id>")
```

This route directs the user to specific post on which it is clicked.

```
@posts.route("/post/<int:post_id>/update", methods=['GET', 'POST'])
```

This route directs the user to update post page where the user can update the post.

```
@posts.route("/post/<int:post_id>/delete")
```

This route helps in deleting the post of the user

```
@posts.route('/like/<int:post_id>/<action>')
```

This route helps in performing the action of liking a post of a user.

```
@main.app_errorhandler(404)
```

This route manages 404 error in html page

```
@main.app_errorhandler(403)
```

This route manages 403 error in html page

```
@main.app_errorhandler(500)
```

This route manages 500 error in html page

Architecture and Features

The project is organised into different folders such as

1. Main(Controller): This folder contains main routes
2. User(Controller): This folder contains user,util routes and forms
3. Post(Controller): This folder contains post routes and forms
4. Templates: This folder contains all the html pages
5. Static: This folder contains all static files such as images and css

Some features implemented:

Like: The like feature was implemented using the post_like table which has 3 fields - id which is the corresponding like to the post, user_id is the id of the user who liked the post, post_id is the id of the post which was liked by the user.

Follow: The follow feature was implemented using the many to many relationship corresponding to the follower table which has 2 fields - followed_id and follower_id where follower_id is the id of the user who is following the followed_id user.

Video

https://drive.google.com/file/d/1vv7qODbyS6EYsUJZjvKaDGo3hOlPAVzw/view?usp=sharing