Instructions

The instructions can also be found in the readme.txt.

-> Create a new database (ideally with the name ‘social_media’)
-> run ‘createTable.sql’
-> run ‘importData.sql’ (populates the tables using the provided csv files)

-> edit ‘config.py’ and specify the host, port, database name, user, and password of the mysql database created in the previous step.

Note: you might want to create a new user in your mysql and provide it all privileges if you don’t want to use your root user.

-> open a terminal/command prompt in the ‘v9malhot_s6bhatna/python’ folder
-> run the command to install pip packages dependencies: pip install -r req.txt
-> run the following command to start using the application: python cli.py

Demo

Please refer to demo.mp4 in the video folder. The application is fully functioning. The demo videos goes through the following functionalities of the application:

1. Sign up
2. Login (includes validation)
3. Create a post
4. View posts from topics you are following
5. View posts from users you are following
6. Create a new topic
7. Follow a new topic
8. View comments on a post
9. View likes on a post
10. Add a comment on a post
11. Add a like on a post

SQL

The data is retrieved and updated by the python application in real-time using SQL queries. These queries can be found in connector.py as well as queries1.sql and queries2.sql.

Python Code

There are 6 main components of the application: cli, config, connector, middle, objects, and state.

cli.py Contains all the front end code which runs the interactive CLI application using a state machine to render the views

state.py Contains the classes for each state and the behaviour according to each state

middle.py Contains the middleware and provides a session provider to cli.py which is used to communicate to the backend which retrieves the data.

connector.py Contains the backend code which communicates with the MySQL database and retrieves the data by sending the appropriate queries to the database.

objects.py Contains data model classes which mimic structure of the entities returned from the database

config.py Contains the configuration information about the database which is required to make a connection