MAD - 1 REPORT

bloglite, only a blogging app

Tech used: Flask, Python, Jinja Templates, Bootstrap, SQL

Short Report

- It is a multi-user app
- Used for uploading blogs with images
- User can post multiple times
- Each post will have
  - ID
  - Title
  - Caption/Description
  - Timestamp
- A user can follow other users using the app

21f1007102
Arya Bhosale

- Each user will have
  - username
  - Password
  - No of followers
  - No of posts
- Every user will have its own feed
- System will automatically show the blogs from the users you follow in a particular sequence
- The recommended order of blogs in a user's feed is based on the timestamp of blogs

21f1007102@student.onlinedegree.iitm.ac.in

Application Structure

This structure has four top-level folders:

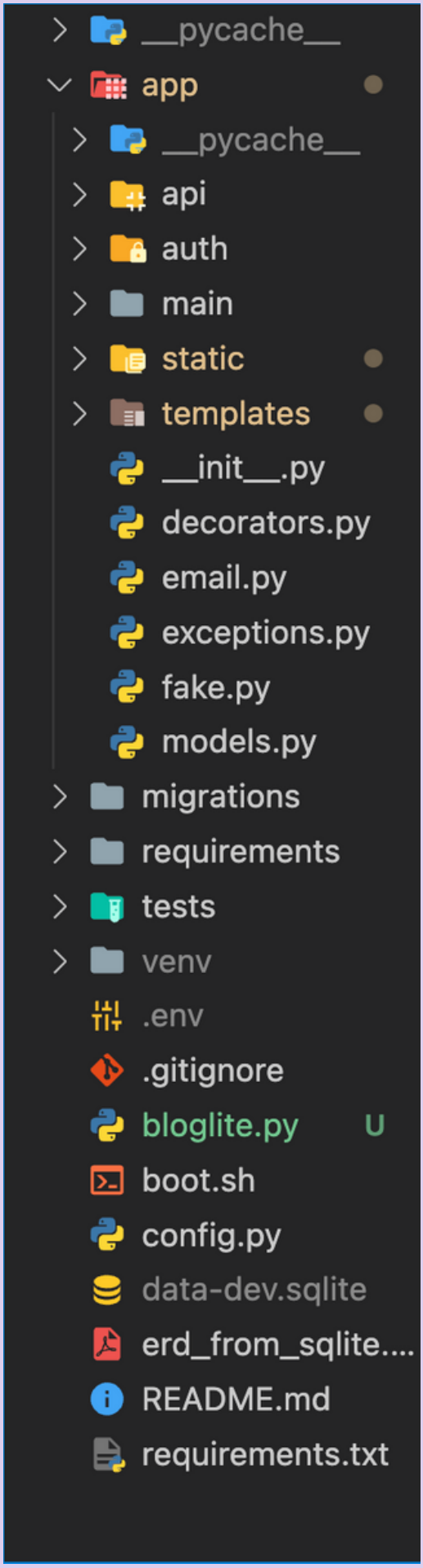
- The Flask application lives inside a package generically named app.
- The migrations folder contains the database migration scripts, as before.
- Unit tests are written in a tests package.
- The venv folder contains the Python virtual environment, as before.

There are also a few new files:

- requirements.txt lists the package dependencies so that it is easy to regenerate an identical virtual environment on a different computer.
- config.py stores the configuration settings.
- manage.py launches the application and other application tasks.

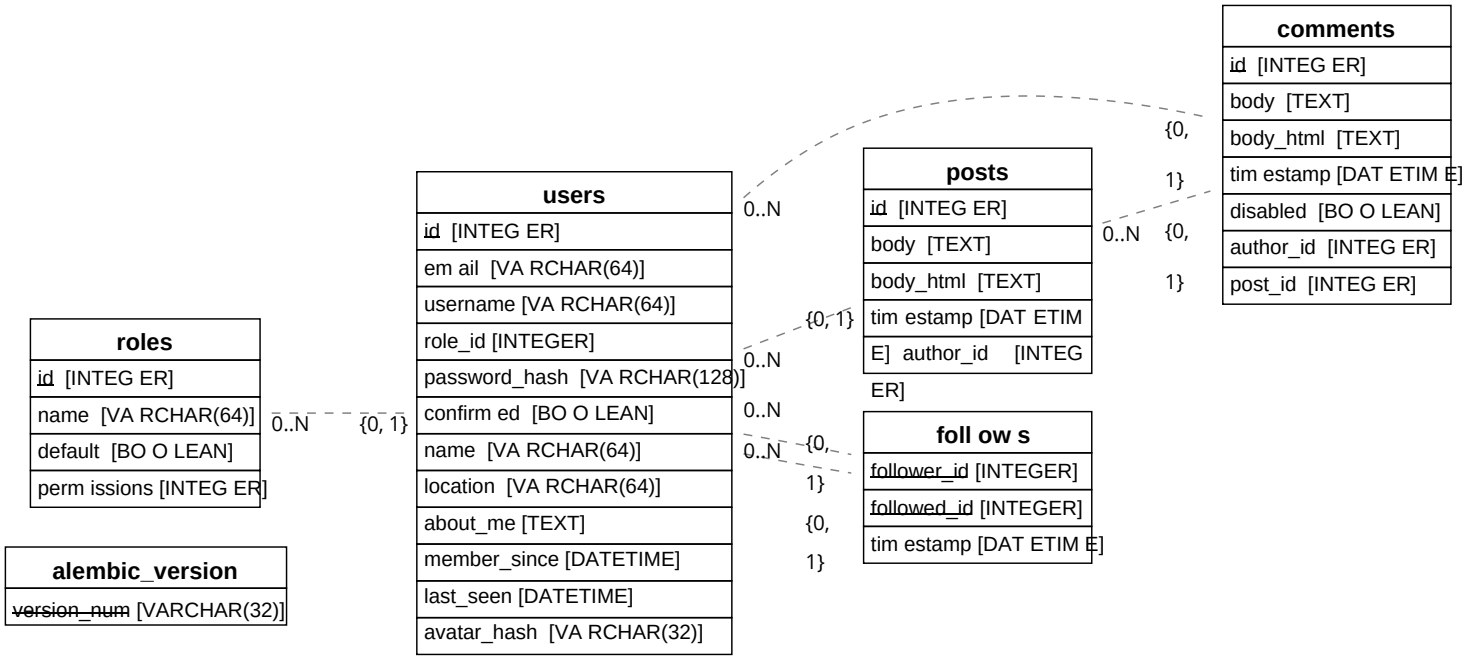
The API endpoints are as follows:

- /users/<int:id> GET A user
- /users/<int:id>/posts/ GET The blog posts written by a user
- /users/<int:id>/timeline/ GET The blog posts followed by a user
- /posts/ GET, POST All the blog posts
- /posts/<int:id> GET, PUT A blog post
- /posts/<int:id/>comments/ GET,POST The comments on a blog post
- /comments/ GET All the comments
- /comments/<int:id> GET A comment
- /token token based user authentication



Given below is the ER diagram of all the models used in the application. Primarily I have used 5 tables in sqlite i.e

- users
- posts
- comments
- roles
- follows



The application uses MVC architecture.
Model-View-Controller (MVC) is an architectural pattern for implementing user interfaces. It divides an application into three interconnected parts: the Model, the View, and the Controller.
Flask is actually not an MVC framework. It is a minimalistic framework which gives you a lot of freedom in how you structure your application, but MVC pattern is a very good fit for what Flask provides, at least in the way that MVC pattern is understood today in the context of web applications (which purists would probably object to).
Video:
https://drive.google.com/file/d/1Z-mGi_R-O_pyhLRu3bzrJIP6CNcAL-RR/view?usp=sharing