Author

Swastik Guha Roy
22f2001624
22f2001624@ds.study.iitm.ac.in
I am from Siliguri , Darjeeling District , West Bengal. Have been pursuing this degree since May 2022. Now I am expecting to complete my Diploma in Programming in this current term. Also I am a 2nd Year Student in B Tech in IT & MI at University Of Delhi

Description

The problem statement (ticket booking application) was an extension of our MAD1 project.Here we were to focus on advanced and dynamic functionality , use of javascript , redis , celery was to be learned and implemented. Also token-based authentication was to be implemented which was new.

Technologies used

1. Ubuntu 22.04.1 (64 bits) USB-Booted
2. SQLite for Database
3. VueJS2 as Javascript Framework
4. Jinja2 & Bootstrap CSS for handling the HTML
5. Package files used:
    - flask : for api calling , routing & framework
    - flask_cros : for api calling
    - date_time : for getting the date & time
    - sqlite3 : for database management
    - secrets: for generating app secret key
    - jwt: for token based authentication
    - uuid : for generating unique booking id
    - flask_mail , smtplib , email.mine.mutlipart/text: for email management
    - redis: for caching
    - celery: for batch jobs
    - celerytask: for batch jobs

DB Schema Design

Click To Find The DataBase Schema

API Design:

Most of the features are implemented through api calling. Fetch is used in my VUEJS2 scripts to call the API sending a JSONFIED payload which returns the appropriate jsonified message.

Architecture and Features

My project contents a template folder where all the pages in their HTML format can be found. The HTML contains inline CSS implemented mostly with Bootstrap , and their VUEJS2 scripts. Since the scripts handle most of the work such as checking for credentials , displaying appropriate messages etc. There are around 3 scripts on average in each page. There are three .py files , the main.py is the backend of my web application , tasks.py and celerytask.py are two files used for celery jobs.

The implementation is simple with a flask backend , and a frontend heavily dependent on VUEJS scripts and works mostly on API calling. Celery jobs are handled by two .py files as mentioned. Emailing to users on a schedule basis is implemented through tasks.py. Also after signup , a welcome message is sent via email. Token based authentication is implementation and ‘role’ helps in RBAC.

Video

Folder Link To Video