REPORT

Author :

- *Name*- Shaifali Vashistha
- *Roll number*- 21f1003257
- *Student email*- 21f1001231@student.onlinedegree.iitm.ac.in.
- *Details*- I am a degree level student in this Online BS degree program. I have a keen interest in exploring the different realms of Technology and thus I wish to accomplish this degree with utmost dedication and hard work. I am very thankful to the **IIT Madras** for providing me this opportunity to develop such an interesting web application and making my journey interesting!

Description :

In this project I have designed a KanBan Application which will track progress of a list of task over time, maintains task deadlines and shows graphs. Each List has its unique ID, description and stores task card under it. Similarly, each cards has its unique ID, ListName(name of the list under which the card is stored), title, content, deadline and status. User can add, delete or update the task lists and cards accordingly.

Technologies used :

1. *Flask*- used for developing web applications using python.
2. *Flask-SQLAlchemy*- used to create database schema and tables using SQLAlchemy with Flask by providing defaults and helpers.
3. *Flask-Security*- Flask extension used to add basic security, password hashing, user verification and authentication features to the application.
4. *Flask-CORS*-Flask extension used for handling Cross Origin Resourse Sharing(CORS), making cross origin AJAX possible.
5. *Flask-Caching*- Flask extension supports Server-side session to our application.
6. *Celery*- Open source python library used to run the tasks asynchronously.
7. *Vue js*- Vue framework used as single file components to build user interface.
8. *smtplib*- Used to define an SMTP client session object that can be used to send mail to the client.
9. *Redis*- it is an in-memory data store that can be used for a high-performance key-value store or Caching.
10. *Datetime*- a python module which supplies classes to work with date and time. Here, datetime module to create timestamp and set deadlines in the application.
11. *matplotlib.figure*- a python module is used to create dynamic figures based on the information provided by the user.

Database Schema :

1. *Relation-* A user can have several Task Lists and thus we have one-to-many relationship between User and TaskList table and are connected to eachother with uid and tid as foreign keys in UserTaskList table. Similarly, a TaskList can have multiple Cards, so we have one-to-many relationship between TaskList and Cards table and are connected to eachother with tid and cid as foreign keys in ListCards table.
2. *ER Diagram* - ER Diagram Link

Architecture and Features:

The project code is organised based on its utility in different files. I have named my project KanBan ToDo Application. Inside this folder there are 2 folders i.e., backend and frontend containing all the required files for the application, a README.md file that covers all the information about the application in detail and a project Report pdf. The backend folder contains 3 folders, application that have all the required python backend files, template that contains HTML template files to be used to send emails to the client, instances that contains sqlite database for the user data storage, and a python file main.py . The frontend folder contains the myapp folder that is the main vue app for the client side. It contains 2 folders, public containing file a html file, index.html and src containing files, App.vue, main.js and 4 folders i.e,assets, components, views and router containing vue components, component views and index.js containing all router paths respectively.

Fetch Request Routes

- @app.route("/register", methods=["POST", "GET"])
- @app.route("/login_page", methods=["POST", "GET"])
- @app.route("/dashboard/<string: username>", methods=["GET"])
- @app.route("/logout_page", methods=["GET"])
- @app.route("/dashboard/<string: username>/create_list", methods=["POST", "GET"])
- @app.route("/<string: username>/update_task_list", methods=["POST"])
- @app.route("/<string: username>/create_card", methods=["GET"]))
- @app.route("/<string: username>/<int: listID>/bounce_card_cache", methods=["POST"])
- @app.route("/<string: username>/<int: listID>/delete", methods=["GET"])
- @app.route("/<string: username>/<int: listID>/<int: cardID>/delete", methods=["GET"])
- @app.route("/<string: username>/update_card", methods=["POST"])
- @app.route("//summary_page", methods=["GET"])
- @app.route("/<string: username>/export_summary", methods=["GET", "POST"])
- @app.route("/<string: username>/<int: listID>/export_card", methods=["GET", "POST"])

Video link: Video Link