Author

Name : Kaustav Goswami
Roll number : 21F1001588
Email id : 21f1001588@student.onlinedegree.iitm.ac.in
I love to do maths as well as programming, which data science provides me in the perfect proportion. I'm currently pursuing MSc in Computer Science from NBU. I love to take part in new projects.

Description

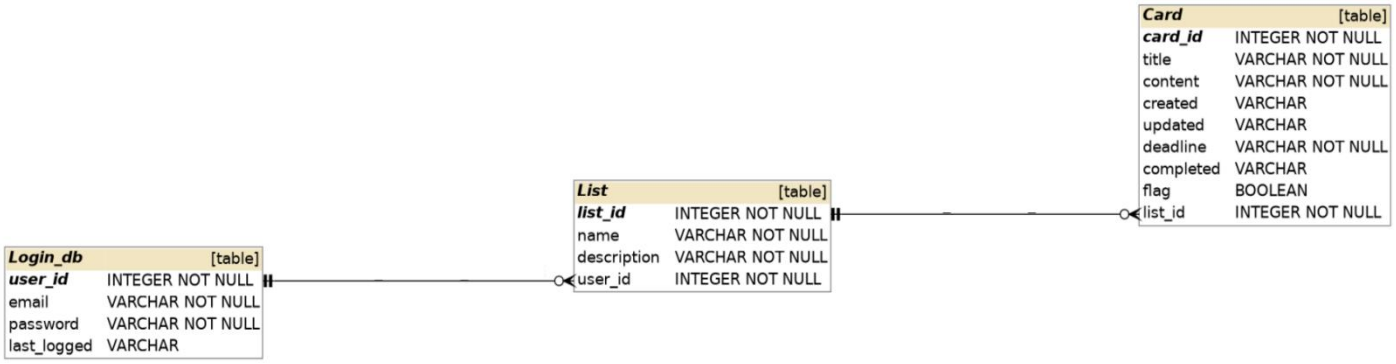
The main aim of this project is to help the user maintain a list or notes which will remind the user to complete the tasks set by the user. A user can login or create a new account to use the application. The user will be able to do **CRUD** operations on each **List** and also on each of the **cards** in the **List**. However, a user will not be able to edit the List/Card title after it has been created. The user will get daily reminder in the email provided and also a detailed monthly report of the his/her current status of tasks. A CSV file will be sent to the email, if the user wants to export the data on the application.

Technologies used

The technologies I used in this project are Python, HTML/CSS/JS, Jinja, bootstrap, Flask, Flask-Login , Flask-SQLAlchemy, Flask-caching, Flask-restful, weasyprint, celery, Matplotlib's pyplot.

- Python is the core programming language used.
- Flask is the main framework for the server.
- Flask-Login is used for managing multiple user login and keeping a session alive.
- Flask-SQLAlchemy is the SQL toolkit used to connect with the database file.
- Flask-caching for making cache and improving performance.
- Flask-restful for creating REST architecture based API
- Weasyprint for creating PDF's from HTML .
- Celery for back-end asynchronous jobs.

ER Diagram



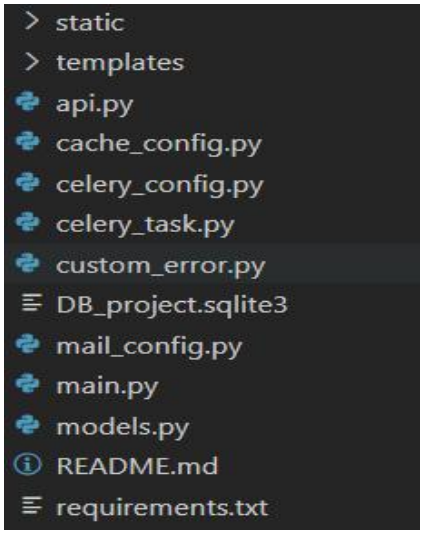
API Design

I have implemented Get, Put, Delete and Post for Login_db table as User_api class, List table as List_api class, Card table as card_api class. Each time any end-point gets called the last_logged from the login_db gets updated for that user who had created that list or card.

DB Schema Design

Table Name	Columns	Description	Constraints
Login_db	user_id	User's unique id	Integer, Primary key, Auto increment
	email	Email of the user	Text, Unique , Not Null
	password	Password of the user's account	Text, Not Null
	Last_logged	Last time user had logged in	Datetime
List	list_id	List's unique ID	Integer, Primary key, Auto increment
	user_id	User's id who created the tracker	Foreign key to Login_db.user_id
	name	Name of the list created	Text, Not Null
	description	Description of the list created	Text, Not Null
Card	card_id	Unique ID of the card	Integer, Primary key, Auto increment
	list_id	List's ID under which the card is	Foreign key to List.list_id
	title	Title of the card	Text, Not Null
	content	Content of the card	Text, Not Null
	created	Time of creation	Datetime
	updated	Time at which the card was updated	Datetime
	deadline	Deadline for this task	Datetime
	completed	Time of completion	Datetime
	flag	Stores whether this task is completed or not	Boolean

Architecture



Here, I have 2 folders:

- **static** - which holds the JS, CSS and image files.
- **templates** - which holds all the HTML files.

Then I have made 8 python files :

- **api.py** file has all the code related to API's.
- **main.py** file has the code for the main code to start the Web App.
- **models.py** file has the all the code related to the different models.
- **custom_error.py** file has the code related to the custom errors that I have created for my Web App.
- **cache_config.py**, **celery_config.py**, **mail_config.py** files have the configurations for the respective parts.
- **celery_task.py** file has all the asynchronous tasks to be done by celery.

DB_project.sqlite3 is the database file. **README.md** has the instructions on how to start the Flask Web App. **requirements.txt** has the required packages name.

Features

Here's a list of features :-

- Multiple users can use the Web App at the same time.
- Users can access the Web App even after closing and re-opening the browser and it wouldn't take the user to the login page unless the user have logged out or have accessed the login page explicitly.
- CRUD operations on List and Card - Create, Read, Update, Delete.
- Daily reminder and monthly report will be sent to the user's email-id.

Video

Link to my video