Final Project Report

BY PUSHPAK RUHIL

21f2001180 |Modern Application Development - II | May term, 2022.

AUTHOR

Name - PUSHPAK RUHIL

Roll No - 21F2001180

Email - 21f2001180@students.onlinedegree.iitm.ac.in

About me – I enjoy programming since my childhood. Complex problem solving makes me happy.

DESCRIPTION

Under this project, we are expected to make a version 2.0 of our MAD-I's WebApp. We have to focus more on the front-end as well async jobs.

TECHNOLOGIES USED

Here's are the technologies I used in this project.

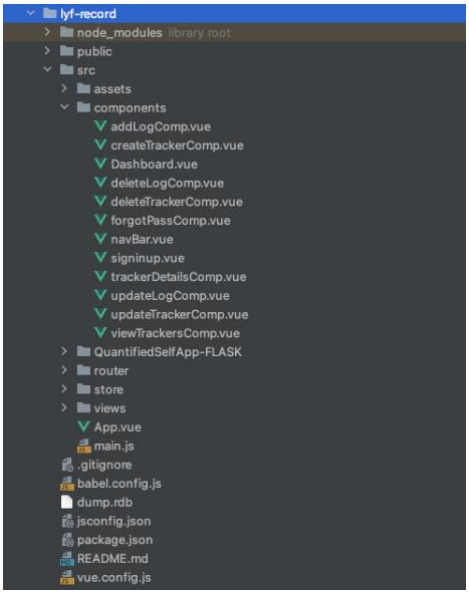
Existing in previous project:

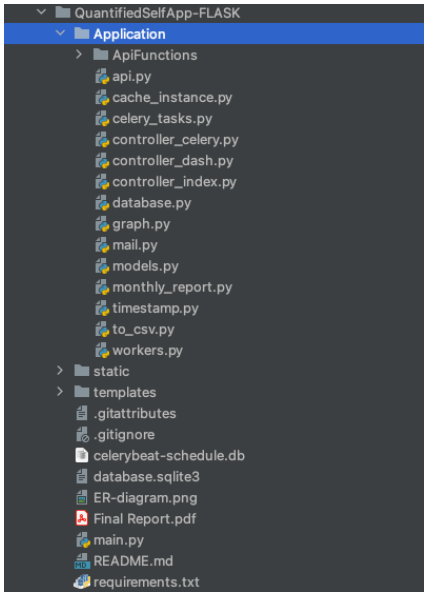
Python, HTML/CSS/JS, Jinja, bootstrap, Flask, SQLite, Flask-SQLAlchemy, Matplotlib's pyplot.

Additional in this project:

Celery, Redis, VueJS/Vue CLI, mailhog, pdfkit, flask-restful, flask-caching, flask-JWT-Extended

ARCHITECTURE





/Component/ folder – all the VueJS components

/QuantifiedSelfApp-FLASK/ folder – Flask backend folder.

/QuantifiedSelfApp-FLASK/Application/ folder – Backend server package. Controllers, API, Async worker, tasks, etc

FEATURES

Here's a list of features –

- User login, Dashboard, trendlines
- CRUD operations on tracker - Create, Read, Update, Delete
- CRUD operations on log
- Graphs for each tracker – summarizes all the logs graphically

- APIs
- Backend async jobs – Daily reminders, monthly reports, download trackers/logs
- Server-Side Caching using Flask-Caching

These are the **additional features** that I have added –

- **Streak** – A number representing the current longest login record.
- **Member since** – Represents the number of days since you've registered with the application.
- **Sign out option** – signs the user out
- **Forgot Password option** – Lets the user receive his password on mail if he forgets it.
- **Integrated sign in/sign up page** – Interactive page with animation where users can sign in and sign up.
- **Server Sent Events(SSE)**– Every time a user registers, a message is sent to the google chat space.
- **PDF Monthly Reports**
- **Single Page Application**

VIDEO LINK

Video demonstration of my project is available here.

https://www.youtube.com/watch?v=8HEiNq8lFkM

DB SCHEMA DESIGN

I have created a total of 9 tables to make my work easy and more organized. I have tried to Normalize the schema.

ER diagram for the schema has been attached by me on the last page, kindly refer to that for more details.

<<DB>>: Discussed Before

Table Name	Columns	details	Constraints
User	Username	Username of the user	Text, Primary key
	Password	Password of the user's account	Text, Not null
	Email	Email ID of the user, can be used to recover account.	Text
	Creation	Date of creation	Text
	Last logged	Last logged value to any tracker	Text
Tracker	Tracker_ID	Tracker's ID number	Integer, Primary key, Autoincrement
	Name	Name of the tracker	Text, Not null
	Description	Description of the tracker	Text
	Type	Type of the tracker (Numerical, multi choice, time duration, Boolean)	Text, Not null.
	Last_log	When was the value last logged for the tracker	Text, Default = "Not yet logged!"
Tracker_bool Tracker_num Tracker_mc	Log_ID	Log event's unique ID	Integer, Primary key, Autoincrement
	Tracker_ID	<<DB>>	<<DB>>
	Timestamp	Time at which the value was logged	Text, Not null
	Value	Value input for the log	Integer(num)/text(others), Not null
	Note	Note for the particular log	Text
Tracker_TD	Time_start, time_end	Start and ending time for the event being logged. In this table, we don't have a value attribute, but these 2 separate attributes.	Text, Not Null (Time_start only)
Multi_choices	Tracker_ID choices	<<DB>> Stores the choices for trackers with type multi choice!	<<DB>> Text
Streak	Streak_id	<<DB>>	<<DB>>
	Username	<<DB>>	<<DB>>
	Date	Date of last login	Text, not null
	Count	Streak count	Integer, default=1
User_Tracker	ID	Just the Primary key	Integer, Autoincrement
	Username	<<DB>>	<<DB>>
	Tracker_ID	<<DB>>	<<DB>>