Author

Name: Shagun Dwivedi
Student Roll Number: 21F1001731
Student Mail ID: 21f1001731@student.onlinedgree.iitm.ac.in

Hi, I am Shagun Dwivedi, a third-year archaeology student at Lucknow University. I like coding and development. I also enjoy geology and traveling. I hope you will like my project.

Description

The project is aimed at creating an app, that I have named TrackOn, that allows multiple users to log in to create trackers to track their activities, keeps logs of their activities, and lets them see their progress via trendlines.

Technologies Used

Following are the technologies/dependencies used:

- **Flask** for main flask app;
- **Flask-Caching** for caching;
- **Flask-Restful** for REST API implementation;
- **Flask-SqlAlchemy** Object Relational Mapper in python for SQL;
- **JWT** for token based authentication;
- **Vue** for frontend UI;
- **Json** for returning json response from APIs;
- **Jinja2** for email templates;
- **Matplotlib** for graphs;
- **Redis** as message broker and caching;
- **Celery** as message broker;
- **WeasyPrint** to create PDF reports.

DB Schema Design

The database consists of 5 tables.

- **log** stores the logging by each tracker for every user
- **multiplechoice** saves values for multiple choice trackers
- **trak_type** is a static table that stores the available types of trackers
- **user** stores the details of the users
- **tracker** stores the details related to every tracker across every used

log

Column Names	Description	Constraint
log_id	Log ID	Integer, Not Null
trk_id	Tracker ID	Integer, Not Null
user_id	Username	String, Not Null
value	Logged in Value	String, Not Null
note	Note regarding Log	String, Not Null
time	Timestamp	String

multiplechoice

Column Names	Description	Constraint
trk_id	Tracker ID	Integer, Not Null
value	Values associated with Multiple Choice Type Trackers	String

trak_type

Column Names	Description	Constraint
traktypeid	ID for tracker type	Integer, Not Null
trak_type	Type Specification	Text

user

Column Names	Description	Constraint
id	username	String, Not Null, Unique
name	User's name	String
email	User's email	String, Unique
password	User's Encrypted Password	String
active	Boolean for User's Activity	Integer

tracker

Column Names	Description	Constraint
trk_id	Tracker ID	Integer
trk_name	Tracker Name	String
description	Tracker Description	String
trk_type	Type Specification	Integer
settings	Specifies values for Multiple Choice Trackers	String
user_id	Username	String

API Design

The REST API has been used for GET, PUT, DELETE, PATCH methods to perform CRUD operations on the User, Tracker, and Logs.

API	Endpoints	Methods	Purpose
RegisterAPI	/api/signup	POST	For User Registration
LoginAPI	/api/login	POST	For User Login
TrackersAPI	/api/trackers	GET, POST	To get and create Trackers
OneTrackerAPI	/api/tracker/<int:tracker_id>	GET, DELETE, PATCH	To view, delete, or update Trackers
LogsAPI	/api/tracker/<int:tracker_id>/logs	GET, POST	To get and create Logs
OneLogAPI	/api/tracker/<int:tracker_id>/logs/<int:log_id>	GET, DELETE, PATCH	To view, delete, or update Logs

Architecture and Features

The architecture is fairly simple, the backend is in the backend folder and frontend is in the frontend folder.
Features implemented are Secure Login, Dashboard, Tracker Management, Logging, Daily Reminders, Monthly PDF Reports, User-Triggered Export Jobs, Caching, Styling and Aesthetics.

Video

Here's a video demo for the project:
Video Link