### Author

Name: Maheedhar A

Roll number: 21f1006544

student email: 21f1006544@student.onlinedegree.iitm.ac.in

I am a student of SSN College of Engineering , Chennai. I am very enthusiastic full stack developer. My hobbies are listening to music, coding.

### Description

From the problem statement we received from IIT , I understood that we need to make a

website which allows the user/client to track his day-to-day activities and allows him/her

to visibly track her progress over time using Vue.js for the frontend and Flask API for the backend

### Technologies used

In my code, I have used the following technologies:

1) flask, flask-sqlalchemy ,

2) sqlite ,

3) numpy ,

4) matplotlib ,

5) os library from python

6) datetime library from python,

7) HTML,

8) CSS and

9) Bootstrap

10) Vue.js

11) Celery, redis and smtplib

12) Pandas

13) flask-jwt-extended

Their applications:

1) Flask and flask-sqlalchemy for basic routing and database related operations and to provide CORS access to the Vue.js application

2) numpy , matplotlib for plotting the graphs which shows the progress of the user.

3) Os library to set the secret key for our application.

4) HTML,CSS,Bootstrap to design beautiful and interactive web pages.

5) Datetime library to get date and time in format of datetime with respect to sql- alchemy and to set expiry for javascript web tokens

6) Vue.js for creating frontend and the interface to connect with the flask API.

7) Celery, redis and smtplib to schedule asynchronous tasks and to create reports and send mails to the user

8) Pandas to export the trackers in CSV format.

9) flask-jwt-extended to create and manage javascript web tokens.

### DB Schema Design

Database Structure:

There are totally three tables in my database:

i) user

This table stores the data of users who have created an account in my application.

The attributes of this table are:

1) User\_id: This is the primary key of this table and has an auto increment feature .

2) Username = this stores the name of the user who registered on our website and has not NULL feature enabled

3) Mail= this stores the email-ID of the user who registered on our website and has not

NULL feature enabled

4) password= this stores the password of the user’s account who registers on our website, which the user has to input later to login to the website and has not NULL feature enabled

ii) trackers

This table stores the information about the trackers the user creates.

The table includes the following attributes:

1) Tracker\_id= this stores the id of the tracker the user created. This is the primary key for the trackers table and has auto increment enabled.

2) u\_id= this stores the id of the user who created that particular tracker, This is a

foreign key reference to the username attribute of the user table.

3) tracker\_name= this stores the name of the tracker the user created. It has not NULL

enabled.

4) tracker\_type= this stores the type of the tracker the user created. It has not NULL

enabled.

5) tracker\_settings= this stores the values/choices of the tracker a user creates if the tracker is not of type numerical.

6) date\_created = This stores the date of creation of the trackers

iii) log table

This table stores the values the user logs into a particular tracker.

The table includes the following attributes:

1) Log\_id: this stores the id of the log that the user logs into the tracker. This is the primary key for the log table table and has auto increment enabled.

2) T\_id: this stores the id of the tracker the user created. This is a foreign key for the log

table table that references the tracker\_id of trackers table.

3) User\_id : this stores the name of the user. This is a foreign key which references the

username of the user table.

4) Note = this stores the note the user inputs while logging into the table.

5) value= this stores the value the user wants to input. It has not NULL enabled.

6) Timestamp = This stores the timestamp in which the logging has been done

### API Design

<<Briefly describe what elements you have created an API for and how it was implemented. The YAML file for API to be submitted separately>>

### Architecture and Features

**The Architecture of my project folder:**

Project folder

|\_ backend

|\_ static

|\_ templates

|\_ flask

|\_ application.py

|\_ frontend

|\_ node\_modules

|\_ public

|\_ index.html

|\_ src

|\_ App.vue

|\_ assets

|\_ components

|\_ views

|\_ router

|\_ store

|\_ utils

|\_ pics

|\_ main.js

Features implemented:

1) Register your account in the webpage to track your progress.

2) Login page: This searches the database for your credentials and helps you sign in to the

website and provides you with a javascript web token which you require for accessing the application’s core features.

3) Home page: A home page that welcomes you to the website and gives a short description

about the website and its use.

4) Trackers page: This page searches the database based on the username and filters all

the trackers that the user has created and represents it in the form of a table.

5) Trackers creation page: This page uses Bootstrap forms and helps the user create trackers

for his own.

7) Trackers update page: This page allows the user to edit the trackers he has created and

updates it in the database.

8) Trackers delete page: This page allows the user to delete an entire tracker. Deleting a

tracker deletes all the records it has on the log table.

9) Logging page: Users can log values to a particular tracker in this page. The user is asked

to input the value and note that they want to add while adding the value to the tracker.

10) Tracker info page: This page provides information about the tracker to the user. This

page contains the logged information as well as graphs to indicate trends in the tracker.

11) Log update page: This page allows the user to update the logged values in case the

previously logged values were wrong or if the user wishes to change the values.

12) Log delete page: This page allows the user to delete a particular log in the log table.

Finally, whatever changes we do to the log table is dynamically reflected in the trackers

info page.

Additional features:

1) Styling and aesthetics: I have included the styling and aesthetics using CSS and bootstrap

to make the page stunning and aesthetically pleasing.

2) A working forgot password page: I have worked on making the forgot password page work using the API and vue.js. The whole forgot password option is split into three views , one to input your email ID, one to verify your identity by using the otp that was sent to your email and the third page is where the password changing process will take place.

### Video

<<Link to your online video of not more than 3 minutes length>>