# QuantifiedSelf App

## Problem definition

## Modern Application Development - I

Last Updated at : 2021-02-17T23:04:00.00+5:30

## Frameworks to be used

- Flask for application code
- Jinja2 templates + Bootstrap for HTML generation and styling
- SQLite for data storage
- All demos should be possible on a standalone platform like replit.com and should not require setting up new servers for database and frontend management

## QuantifiedSelf

- Used for self tracking - tracking habits, activities, other life parameters etc.
- User can have multiple trackers
- Each tracker will have a
  - ID
  - Name
  - Description
  - Tracker type
  - Settings
- User can log to one more tracker at any time, each time it's logged it will capture
  - TimeStamp
  - Tracker
  - Value (based on the corresponding tracker type)
  - Note
- System will track progress over time and shows graphs trend lines etc

Terminology

- Tracker - Corresponding to the
- TrackerType - Type says what data is captured
  - Numerical
  - Multiple Choice
  - Time Duration
  - Boolean
- Logging - Logging an event to a tracker by providing values
- Trendline - Shows the list of logged events and may be graphs

## Reference Material

- Quantified Self on Wikipedia
- Quantified Self - Awesome List - Curated by public

## Similar Products in the Market

1. Nomie - Life tracker
    - Open Source
    - Web
2. Loop Habit Tracker
    - Open Source
    - Android App
3. Tickmate - Journal and Tracker
    - Open Source
    - Android App
4. Dijo - Habit Tracker
    - Open Source
    - Command Line

- These are meant for exploring the idea and inspiration
- Don't copy

## Example - Temperature Tracker

- Can be used to log daily temperature by covid patients
- Tracker
  - ID : PK : tracker1
  - Name: Temperature
  - Description: Tracking body temperature in Fahrenheit
  - TrackerType: Numerical

- Users can log at any time of the day.
- Example log 1
  - TimeStamp : "2022-05-26T11:42:00.73+05:30"
  - Tracker: tracker1
  - Value : 98.3
  - Note : I was feeling okay
- Example 2
  - TimeStamp : "2022-05-27T10:42:00.73+05:30"
  - Tracker: tracker1
  - Value : 100.1
  - Note : Feeling tired and bit feverish

## Example - Running Tracker

- Can be used to log daily running by anyone
- Tracker
  - ID : PK : tracker2
  - Name: Running
  - Description: Tracking daily running in kilometers
  - TrackerType: Numerical

- Users can log at any time of the day.
- Example log 1
  - TimeStamp : "2022-05-26T11:42:00.73+05:30"
  - Tracker: tracker2
  - Value : 5
  - Note : It was a good run. Felt a little tired but okay.
- Example 2
  - TimeStamp : "2022-05-27T10:42:00.73+05:30"
  - Tracker: tracker2
  - Value : 2
  - Note : Couldn't run much today
- Example 2
  - TimeStamp : "2022-05-27T18:42:00.73+05:30"
  - Tracker: tracker2
  - Value : 3
  - Note : Making up because couldn't run in the morning

# Example - Mood Tracker

- Can be used to log mood multiple times a day
- Tracker
    - ID : PK : tracker3
    - Name: My Mood
    - Description: Tracking my mood multiple times a day
    - TrackerType: Multiple Choice
    - Settings: Angry, Sad, Happy, Calm, Okay, Meh

- Users can log at any time of the day.
- Example log 1
    - TimeStamp : "2022-05-26T11:42:00.73+05:30"
    - Tracker: tracker3
    - Value : Angry
    - Note : Not sure why, but I was angry
- Example 2
    - TimeStamp : "2022-05-27T10:42:00.73+05:30"
    - Tracker: tracker3
    - Value : Happy
    - Note : Good food and hence happy
- Example 2
    - TimeStamp : "2022-05-27T18:42:00.73+05:30"
    - Tracker: tracker3
    - Value : Calm
    - Note : Meditation did the trick

## Core Functionality

- This will be graded
- Base requirements:
  - User login
  - Dashboard and Trendlines
  - Tracker management
  - Tracker log events

## Core - User Login

- Form for username (and optional password)
- You can either use a proper login framework, or just use a field for username - we are not concerned with how secure the login or the app is
- Suitable model for user

## Core - Tracker management

- Create a new tracker
  - Storage should handle multiple languages - usually UTF-8 encoding is sufficient for this
- Edit a tracker
- View Tracker
  - View tracker
  - View logs related to that tracker
  - View stats and trendlines
- Remove a tracker

## Core - Dashboard

- Dashboard with list of trackers
- Time of last review, value on tracker
- Ability to go to logging view for any tracker
- Ability to go to create or edit tracker
- Ability to go the specific tracker details

## Core - Logging

- Click on a tracker, then log the values
    - Based on the TrackerType it should show the options to log
        - Numerical - Show the text box that takes numerical values only
        - MultipleChoice - present the options
        - Time Duration - Give the ability to select time duration, like 30 mins, 1 hr 29 mins etc.
    - The current timestamp needs to be picked up automatically. But the user should have the ability to edit
- Edit a log
    - Change value, timestamp or associate notes
- Remove a log

## Recommended (graded)

- APIs for interaction with trackers and logs
    - CRUD on tracker
    - CRUD on the tracker log event
    - Additional APIs for getting stats, trend lines or add other features
- Validation
    - All form inputs fields - text, numbers etc. with suitable messages
    - Backend validation before storing / selecting from database

## Example Wireframe

- Click this link to check the wireframes
- It is just given to gain a basic understanding, and not meant to be followed exactly

## Optional

- Styling and Aesthetics
- Proper login system
- Export/Import logs, so it can be analyzed in Excel or Calc

## Evaluation

- Report (not more than 2 pages) describing models and overall system design
  - Include as PDF inside submission folder
- All code to be submitted on portal
- A brief (2-3 minute) video explaining how you approached the problem, what you have implemented, and any extra features
  - This will be viewed during or before the viva, so should be a clear explanation of your work
- Viva: after the video explanation, you are required to give a demo of your work, and answer any questions
  - This includes making changes as requested and running the code for a live demo
  - Other questions that may be unrelated to the project itself but are relevant for the course