COURSE PROJECT:
WELL WORTH THE WAIT

RELEASED: **25 MAR 2021 (THU)**
DUE: **9 MAY 2021 (SAT)**

SYNOPSIS

In a group of 4–5 students, you are going to set up a webapp to check waiting time for places. Users will be able to log in and choose a subset of favourite places. Your project app will retrieve place details from open datasets.

DATA SOURCE

You have to access some open API for actual data. You can pick among these three topics, based on your interest, ability, and preference:

1. Accident and Emergency Waiting Time by Hospital
   https://data.gov.hk/en-data/dataset/hospital-hadata-ae-waiting-time
2. Journey time indicators
   https://data.gov.hk/en-data/dataset/hk-td-sm_2-journey-time-indicators
3. Popular time/ Waiting time of Restaurants in Google Maps
   https://developers.google.com/maps/documentation/places/web-service/overview
   https://github.com/m-wrzr/populartimes

Only the main link is given above, you may or may not need to ***access related data*** as well.

If you identify a comparable dataset of waiting-time-related information from another source, you may propose to use it by emailing to chuckjee@cse.cuhk.edu.hk.

ACCESS MODES

Your app will provide two modes of access:

Users – Only authenticated users have access to the app contents. A user is recognized using a username (unique string of 4–20 characters) and password (string of 4–20 characters, hashed) pair. The user will be able to see and search for place details, maintain a group of favourite places, and leave comments on places.

Admins – Admins will be able to perform arbitrary CRUD actions to the place data and the user data on your database.

THE DATA

There are some differences in pre-processing of data from the 3 given sources for our local storage and display:

1. ***Hospital***:
    - only the hospital name is given, so you have to manually (or programmatically) query for the latitude and longitude and store it
2. ***Traffic***:
    - the data set for one "place" includes journey time towards multiple destinations, so you have to see how to handle and visualize that
    - the latitude/longitude is in the data dictionary
3. ***Restaurants***:
    - the GitHub repo mentioned above is unofficial, and is using Python
    - only a small subset of restaurants have data for popular time/ waiting time, so you may run some tests to see what is indeed available
    - data contains a value for each hour, and only the ***current*** popularity/waiting time is to be shown

For datasets 1 and 2, you need to get the waiting time from API when the user loads your page and show the last updated time clearly.

For the project purpose, you are only required to have at least ***10 places*** in your app. It does not matter if you store or show more. You need to design the data schemas and models for storing (caching) items. For the places, you are required to maintain at least:

- Latitude and longitude
- Name

These data may be named and formatted differently in the above datasets. You need to design your database access according to that. Only English data is required for the project app. For the schema and models for users and other data, you may design freely to suit your needs.

You may need to consult data dictionaries and related data for place details, and can feel free to use extra APIs for your app. ***Never use anything more than FREE TIER***!

APPLICATION REQUIREMENTS

User actions:

1. List all places in a table, and allow sorting of the table with ***one*** of the listed fields, linking to single places
2. Show all ~~*available*~~ places in a map, with links to each single place
   [Suggested APIs: Google Maps, MapBox]
3. Search for places which contain keywords in ***one*** field chosen by the user which will result in a table of place results
4. A separate view for one single place, containing:
   a. a map showing the place
   b. the place details
   c. user comments, where users can add new comments (*non-threaded*)
5. Add place into a list of user's favourite places, and see the list in another view (*flexible implementation*)
6. See the username in the top *left/right* of screen, and be able to log out

Admin actions:

1. Refresh data, i.e. reload from the online dataset, *without affecting data which does not come from API (e.g. user comments within your app)*
2. CRUD place data in the local database
3. CRUD user data (username and password only) in the local database
4. Log out as admin

Non-user actions:

1. Log in as user with username and password
2. Log in as admin using username and password both as `admin`

You may introduce pagination if you see fit, but it is not a requirement.

CHARTING HISTORICAL DATA

Using any JS charting library (e.g. Chart.js), include these two charts:

- Waiting time in the past 10 hours
- Waiting time in this hour of past 7 days

Historical data is available for all the 3 datasets. If you want, you may create more than the required charts.

SYSTEM REQUIREMENTS

Your app will need to be built on the CSCI2720 VM provided by the CSE department. It should be hosted in the account of one of the members. *Details are provided in Lab 6 and 7. Please read it carefully. The following facilities are provided:*

- *Node v15.11.0 + npm 7.7.0*
- *MongoDB server 4.2.11 / MySQL server 5.7.33*

Your project app needs to be a ***Single Page Application***, without refreshing the page for any internal links. However, visits to all different views should be reserved in the browser history, with a proper URL.

You will decide the complexity and aesthetics of your work. Make sure it is clear and useable by average users (e.g. your TAs) without much guesswork. You can make decision on anything not specified in this document, and extend beyond the basic requirements.

You may freely decide the choice of technologies and frameworks to be used in this project. The grading will be done on a desktop computer using Google Chrome (*almost-newest versions*), so your app should at least serve HTML and relevant styling and scripting codes.

ASSESSMENT AND SUBMISSION

All technical features would be graded during the demo. Your project will be graded by:

- Technical requirements – *fulfilment* and *complexity* (50%)
- Usability – *look and feel* (20%)
- Project demo (10%)
- Project outline/report (20%)

PROJECT OUTLINE [**APRIL 9**]

Submit a one-page document discussing the followings:

- Group members, data source, data schema plan, APIs to be used, reasons for your chosen platform and technologies comparing to others, and anything you consider relevant

PROJECT DEMO [**MAY 8** AFTERNOON] AND SUBMISSION [**MAY 8**, 23:59]

Details on the project demo will be announced in the end of April.

Include full names and student IDs of all members in ***all code files*** using comments. Zip all your files and submit it on the course site on Blackboard.

If there is one, you do not need to submit the **`node_modules`** ***folder but please keep the files*** **`package.json`** ***and*** **`package-lock.json`**.

Submit also a **`readme.txt`** to state the Node server start commands, as well as your site URL. Indicate clearly whether or not you have read this article carefully: *http://www.cuhk.edu.hk/policy/academichonesty*

PROJECT REPORT [**MAY 8**, 23:59]

You need to submit a document to describe your project, ***with reference to the CSCI2720 course materials***, and other online references. Here are suggested components for your report:

- **Abstract**
  - A summary of your work in no more than 100 words, *with one screenshot of one representative screen of your site*
- **Methodologies**
  - Discussion on the programming language(s) and important algorithms you have used (and you may reference the course materials if needed)
  - Design of data schemas and models of your database
  - A comparison table of at least two advantages and two disadvantages (specific to your project app) of your chosen platform and technologies comparing to others, e.g. "*Why React+MySQL?*"
- **References**
  - Citation of all materials which are not originally written by you, including teaching materials in and out of our course
  - You must use the IEEE style (*Ref: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/conferences/style_references_manual.pdf*)
- **Appendix**
  - Anything you consider supplementary, e.g. workload distribution

You may feel free to include more figures for this report.

Please use 2–4 pages for the report. Penalties will be applied for anything out of the allowable range.