COURSE PROJECT:
MAPPING EVERYTHING

RELEASED: **9 MAR 2020 (MON)**
UPDATED: **16 MAY 2020 (SAT)**

SYNOPSIS

In a group of 3–4 students, you are going to set up a site for a tiny social network for locations. Your project app will retrieve location details from open datasets, and allow users to favourite and comment on locations.

DATA SOURCE

You have to access some open API for actual data. You can pick among these three topics:

1. Bus arrival time for Citybus routes
   https://data.gov.hk/en-data/dataset/ctb-eta-transport-realtime-eta
2. Residential Buildings of Conoravirus cases
   https://data.gov.hk/en-data/dataset/hk-dh-chpsebcddr-novel-infectious-agent
3. Businesses in an area on Yelp
   https://www.yelp.com/developers/documentation/v3/business

Only the main link is given above, but you will likely need to ***access related data*** as well.

If you identify a comparable dataset of location-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 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 location (bus stop/building/business) details, maintain a list of favourite locations, and leave comments on locations. [Note: reviews/comments from the original dataset are not needed in this work.]

Admins – Admins will be able to perform arbitrary CRUD actions to the location 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. Bus arrival time: you'll choose 10 bus routes to store their stops, yet the ETA is always live!
2. Coronavirus buildings: all cases from data source need to be stored, and you need to lookup the coordinates from building names (e.g. using Google Maps API), but be warned that this dataset is updated daily
   [Note: Since number of cases is recently decreasing, you are advised to work with historical data, and perhaps allow easy access of data on a different date with your own design.]
3. Yelp business: only businesses within 10km from CUHK need to be stored, yet the rating is always live!

For the ***"live" data*** *indicated above, you only need to get it from API when the user loads your page.*

You need to design the data schemas and models for storing (caching) items. For the locations, you are required to maintain at least:

- Latitude and longitude
- Name
- Buses arrival time (*for buses*) / related patient age (*for buildings*) / photo(s) (*for businesses*)
- Two more attributes you find useful from the dataset

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 model for users and other social networking components, you may design freely to suit your needs.

You may need to consult data dictionaries and related data for location details, and can feel free to use extra APIs for your app. ***You definitely should use nothing more than free tier***!

APPLICATION REQUIREMENTS

User actions:

1. List all locations in a table, and allow sorting of the table with ***one*** of the listed fields, linking to single locations
2. Show all locations in a map, with links to each single location
   [Suggested APIs: Google Maps, MapBox]

3. Search for locations which contain keywords in ***one*** field chosen by the user which will result in a table of location results
4. A separate view for one single location, containing:
    a. a map showing the location
    b. the location details
    c. user comments, where users can add new comments (*non-threaded*)
5. Add location into a list of favourite locations, and see the list in another view (*flexible implementation*)
6. See the username in the top right of screen, and be able to log out

Admin actions:

1. Flush 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 location data in the local database
3. CRUD user data (username and password only) in the local database
4. Create location data from CSV file upload (sample needs to be provided for user on data format)
5. Log out as admin

Non-user actions:

1. Log in as user with username and password
2. Log in as admin via a link (*yes that's very insecure*)

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

USER LOCATION

Each user should have an attribute of "home location". This should be updatable by the user by using a map interface, and stored as a pair of longitude and latitude coordinates. Then, this user location should allow query in the "search" view (user action #3), specifying "nearby x km to home location" of specified x.

You may use any external API for displaying a map for accepting location input.

CHARTING STATISTICS

Using any JS charting library (e.g. Chart.js), include two charts for user, and two charts for admin view. At least these should be included:

- User view: top 5 locations with most comments (e.g. bar chart)
- Admin view: top 5 active users with comments and favourites (e.g. bar chart)

Please prepare enough local data to show the chart features meaningfully. You may create more than the required charts.

SYSTEM REQUIREMENTS

Your app will need to be built on a Linux virtual machine provided by the department. It should be hosted in the account of one of the members. *Details are provided in Lab 6.5. Please read it carefully. The following facilities are provided:*

- *Node v12.16.1 + npm 6.13.4*
- *MongoDB server 4.2.3 / MySQL server 5.7.29*

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 using Google Chrome (*almost-newest versions*), so your app should at least serve HTML and relevant styling and scripting codes.

"ABOUT THIS PROJECT"

You need to include one extra section in your project app, and name it "**About This Project**". This article should describe:

1. Names and workload distribution of each group member
2. Basic "how-to" of your project app
   — *If there are requirements that is not implemented, please indicate.*
3. Design of data schemas and models of your database (figures are welcome)
4. Technologies and frameworks/libraries in use
   Show a 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?"*
5. Indicate clearly whether or not you have read this article carefully:
   http://www.cuhk.edu.hk/policy/academichonesty

PREPARING A RESTFUL API

~~Inside your site, you MUST include a description of your location data schema as `/locschema.txt`, which clearly lists out the structure of the location items in your database.~~

After the project demo, you need to work out a RESTful API for your own group for accessing location data (where *loc-id* represents a valid *number/string* pointing to a location):

- List all locations: **`GET /api/locations`**
- Add a new location: **`POST /api/locations`**
  - With location details in request body, and URI of created location returned in response header, e.g. `Location: http://.../api/locations/`*`loc-id`*
- Retrieve a location: **`GET /api/locations/`***`loc-id`*
- Update a location: **`PUT /api/locations/`***`loc-id`*
  - With updated location details in request body
- Delete a location: **`DELETE /api/locations/`***`loc-id`*
- Input/output format: XML inside request/response body, with elements of `<name>`, `<id>`, `<latitude>`, `<longitude>` inside `<location>` elements.

This unencrypted HTTP header must be used for all requests to your API, otherwise a response with status code 401 must be returned: **`Authorization: Bearer csci2720`**

You must use the HTTP methods indicated above. You may refer to ***Lecture 17 RESTful API***, slide 11. The API is expected to be accessed as *http://csci2720.cse.cuhk.edu.hk/????/api/locations/...*

SUBMISSION AND ASSESSMENT

PROJECT OUTLINE [**MARCH 31**]

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 AND SUBMISSION [**MAY 15**, 23:59]

You must leave your files and database unmodified on the VM account after the submission deadline. You will also need to submit your site URL and start up commands. Your accounts will not be available after the deadline of the API Extension.

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.***

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

- Technical requirements – *fulfilment* and *complexity* (42%)
- Usability – *look and feel* (10%)
- Project demo (10%)
- Project proposal (2%), Project report (20%)
- API extension (16%)

PROJECT REPORT [**MAY 22**, 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
- **Background**
  - Describe how your project is relevant to the course lectures/labs
  - Motivation of your project ideas, and similar work on the market
- **Methodologies** (with subsections)
  - Discuss the programming language(s) and important algorithms you have used (and you may reference the course materials if needed)
  - Elaborate on your project work
- **Conclusion**
  - A summary of what you have done in this project
- **References**
  - Cite all materials which are not originally written by you, including all 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

The report should naturally be an extension from your "About This Project" section in your web app. You may feel free to include more figures for this report. *You are not required to include details on the project API extension.*

Please use 6–8 pages for the report. Penalties will be applied for anything out of the allowable range.

PROJECT API EXTENSION [**MAY 22**, 23:59]

The section of RESTful API will be graded ***using scripts*** after the due date. All groups need to submit to Blackboard a **`readme.txt`** including the username on the VM, URL and commands to start the API server, and anything else important to note.