Coursework – REST API and Progressive Web Apps (PWA) (25%)

Deadline: 5pm on the Friday of Week 14

Task

For this coursework, you need to:

- Create the back-end for the web app build in Coursework 1;
- Make the app `progressive'.

Submission

- All the necessary files should be included in a zip file (max 10MB), including the html, css, javascript, a text file containing the GitHub repository URL, and external library if there is any.
- Please make sure the GitHub repository is public so it can be checked.

A submission will receive zero mark if it fails any of the criteria below:

- The backend server must use `Node.js`; Apache or Xampp is not allowed;
- The data must be stored in `MongoDB`; any other database is not allowed;
- All database access, such storing and retrieving data, must be achieved through `REST API`; any other type of access is not allowed, including direct database access;
- The `REST API` must be developed with `Express.js`;
- The front-end data access must be achieved with `promise` using `fetch` function; `xmlHttpRequest` is not allowed.
- The code must be hosted in a GitHub repository with at least 10 commits;
    - You can continue using the same repository created for cw1, in which case 10 new commits are required.
- The work must be demonstrated in lab in person.
    - Coursework code will not receive any mark, even it works fine if It cannot be explained satisfactorily during the in-lab demonstration, i.e., student cannot explain what the code does.

Marking scheme (total 25%)

Data storage and access (8%):

– Store in MongoDB user profile - minimal fields: email, password, and user type; – Store in MongoDB course information - minimal fields: topic, price, location, provider, review rankings and the author for each ranking;

- Connect to MongoDB with native Node.js driver for MongoDB;
- Access database in the front end using `promise` with `fetch()` function.

REST API (9%):

– For user, there should be at least the follow REST routes:

- Create a new user;
- Retrieve user information by email;
- For courses, there should be at least the follow REST routes:
    - Retrieve all courses;
    - Retrieve all courses created by a provider;
    - Create a course;
    - Update a course;
    - Delete a course;
- For review, there should be at least the follow REST routes:
    - Save a ranking;
    - Update a ranking.

Progressive Web Apps (8%):

– Use service worker to cache app files:

- Each course must have an unique images, with at least 10 different images;
- Always load app from cache first. – Make the app 'installable':
- Can be added to home screen;
- Run in full-screen mode; – Send user notification;
- When cache is created;
- Whenever data is retrieved from cache; – Progressive loading
- Only load an image when it appears in browser window;
- Display a place-holder image first, and then replace it with the actual image.