Assignment # 4

Due Date Wednesday, April 9, 2022, @11: 59 PM

In this Assignment, you will build **the employee management (emApp)** app's backend using express and Node.js

Preparation

1. Sync cmps350-lab repo to get the Assignment files
2. Open the **assignment4/emApp** project
3. The project should be organized as follows:

    a. public folder contains the HTML pages, templates, CSS, and client-side JavaScript

    b. data folder contains **employees.json** files to be used in this Assignment.

4. Run **npm install express** to install the express package.

PART A – Creating the Server and Employee APIs

- Use the given **data/<u>employees. json</u>** file for persisting and reading the data for all the below methods.
- Create an **employee-repo.js** file and implement the EmployeeRepo class with the following methods listed below.

Method	Description
getEmployees()	Return all employees inside the employees.json file
getEmployee(eid)	Return an employee by employee id (eid)
addEmployee(employee)	Adds new employee to the employees.json file
updateEmployee(employee)	Updates existing employee in employees.json file
deleteEmployee(eid)	Deletes an employee from the **employees.json** file using the employee id (eid)
deleteEmployees()	Remove all the employees from employees.json file

Note : Test your **EmployeeRepo** class before moving to the next step to make sure everything works. You can create an object of the **EmployeeRepo** class and test all the methods or use mocha and chai. Once you finish the testing, comment out or remove the test code.

- Create a **employee-service.js** file and implement **EmployeeService** class with methods to handle the requests listed in the table 1 below. The handlers should use the provided methods from the EmployeeRepo class.
- Create a **router.js** file and implement handlers for the routes listed in the table below. The routes handlers should use the methods from the **EmployeeService** class.

HTPP Verb	URL	Functionality
Get	/api/employees	Return all employees
Get	/api/employees?eid={**employee id**}	Return a employee by eid
Post	/api/employees	Adds new employee
Put	/api/employees/:eid	Updates existing employee
Delete	/api/employees/:eid	Deletes a employee by eid
Delete	/api/employees	Remove all employees

- Finally, create **app.js** file, and import express package, instantiate an express app, configure it then start it at port 9090.
- Test your API's using **PostMan**

Update the Client App to use the Web API

Your task is to **replace** the Indexed DB with API calls to the server APIs you created above. The application should work the same way as Assignment 3 with one additional method called **showEmployee Detail**. The working and complete code of assignment 3 is given to you. You only need to re-write the **public/js/repository/employee-repo**.js file on the client-side. And make the **employee management app** communicate with the server instead of the indexedDB.

After completing the Assignment, fill in the **_TestingDoc-Grading-Sheet.docx_** and save it inside the **_Assignment4_** folder. Then, push your work to the Github repository.