**Lab Assignment #3 – Micro-Frontends and Microservices with GraphQL**

Due Date: Week 10, Thursday, 2:30pm.

Purpose: The purpose of this assignment is to:

1. Extend the existing micro-frontends and microservices architecture to develop user – vital signs system.
2. Utilize GraphQL for communication between micro-frontends and microservices.

References: Read the reference textbooks, lecture slides, and class examples. This material provides the necessary information that you need to complete the exercises.

Be sure to read the following general instructions carefully:

- This assignment may be completed using the **pair programming technique** (https://en.wikipedia.org/wiki/Pair\_programming).

- See the naming and **submission rules** at the end of this document

- You will have to **provide a demonstration for your solution** and upload the solution on eCentennial through the assignment link.

**Exercise 1:**

In this exercise you will set up the project infrastructure and begin backend development, using cutting-edge technologies, including Micro-Frontends with Vite Module Federation for the React user interface and Microservices with Express.js and Graph QL for the backend.

1. **Backend Development**
2. **Authentication Microservice -** Develop a microservice using Express and GraphQL to handle **user registration and login**.
   1. Implement MongoDB schema for users.
   2. Implement mutations for **signup**, **login**, and **logout**.
   3. Ensure security measures for user authentication (e.g., hashing passwords).
3. **Vital Signs Microservice** - Develop a microservice using Express and GraphQL to handle vital signs operations (add, display, update)
4. **Initial Frontend Development**
5. **Authentication Micro Frontend -** Develop a micro frontend using React Vite and Apollo Client.
   1. Create a micro frontend responsible for user authentication (signup, login, logout).
   2. Integrate with the Authentication Microservice for user-related operations.
6. **Vital Signs Micro Frontend -** Develop a micro frontend using React Vite and Apollo Client.
   1. Create a micro frontend responsible for handling vital signs (create, update, read).
   2. Integrate with the Vital Signs Microservice for vital signs-related operations.

Use **functional components**, client-side composition, and **React Hooks** for the Micro Frontends. Design a visually appealing and user-friendly UI.

(10 marks)

**Evaluation:**

|  |  |
| --- | --- |
| **Functionality(including code explanation during demonstration):** |  |
| **Micro frontends (Authentication, Vital Signs)** | 30% |
| **MongoDB database** (config files, models) | 5% |
| **GraphQL Microservices (Authentication, Vital Signs**) | 30% |
| **Integration using Vite Module Federation plugin** | 20% |
| **Friendliness** (using CSS to align the React elements, React-Bootstrap, etc.) | 5% |
| Code **demonstration** and brief explanation during demonstration in class | 10% |
|  |  |
| **Total** | **100%** |

**VS Code Project Naming rules:**

You must name your **VS Code** project/folder according to the following rule:

**YourFullName\_COMP308LabNumber\_ ExNumber**.

Example: **JohnSmith\_JaneSmith\_COMP308Lab3\_ Ex1**

**Submission rules:**

**Remove the node\_modules folder before zipping the project.** Submit your project as a **zip file** that is named according to the following rule:

**YourFullName \_COMP308LabNumber\_ExNumber.zip**

Example: **JohnSmith\_JaneSmith\_COMP308Lab3\_Ex1.zip**

**DO NOT use RAR or other types of archives.**