Group Project – Developing Apps Using Emerging Web Technologies

Due Date: Group presentation and demonstration in Week 14

Purpose: The purpose of this project is to:

- Design and code web apps using emerging web frameworks
- Build a Graph QL API using Express
- Build a Front-End that utilizes the Graph QL API
- Apply appropriate design patterns and principles
- Use Deep Learning to make intelligent use of data

References: Read the reference textbooks, lecture slides, class examples, and additional references provided here. This material provides the necessary information that you need to complete the project. You may need to read and use more materials and tools to implement a good solution.

Be sure to read the following general instructions carefully:

- This Project **may be completed in groups of 5-6 students**.
- This project can be **replaced with your capstone project** (COMP-231 or COMP-313), if you use and implement the **same front-end/back-end technologies shown in this document**.
- You will have to **present and demonstrate your solution in Week 14** and upload the solution on eCentennial through the assignment link on D2L. Bonus marks will be given if you also **publish the app on Heroku, Microsoft Azure, Amazon, or another Cloud platform.**
- **Your VS Code project should be named "YourGroupName_COMP308Project_SectionName" and should bezipped in a file YourGroupName_COMP308Project_SectionName.zip**.

Project Specifications

Your client needs an application to help nurse practitioners to monitor patients during the first weeks of their release from the hospital and also help the patients to monitor their daily activities. Develop a modern web app that implements the following functionalities:

1. User **registration/login**
2. If the user is a **nurse**:
   a. Allow the user to **enter vital signs**: *body temperature*, *heart rate*, *blood pressure*, or *respiratory rate*.
   b. Allow the user to **access information** captured during a previous clinical visit, vital signs: *body temperature*, *heart rate*, *blood pressure*, or *respiratory rate.*
   c. Allow the user to **send daily motivational tips** to the patient (by storing them in the database and providing a *daily tips* page for the patient to view, etc.).
   d. Allow the user to generate a list of possible medical conditions, and advise the patient to see a doctor if necessary - **intelligent use of symptoms** or other data using deep learning and publicly available datasets.

3. If the user is a **patient**:
    a. Allow the user **to create and send an emergency alert** to first responders (by storing this in a separate collection)
    b. Allow the user to access **fitness games page** designed to encourage patients to exercise at home. The **Gaming students** are encouraged to design/incorporate their own games/interactive pages.
    c. Allow the user to **enter daily information (**for example *pulse rate*, *blood pressure*, *weight*, *temperature*, *respiratory rate*).
    d. Allow the user to use a **checklist of common signs and symptoms (COVID-19 for example)**, and submit the choices.
4. Use **MongoDB** for storing the information.
5. **Use Express to implement Graph QL API to implement theGraph QL API.**
    a. **Choices for front-end frameworks:**
        i. **React 18** or higher, using **functional components**

Apply **MVC** for the Express part and **responsive** web design principles. Use CSS and React Bootstrap to create a nice look and feel of your app. Display the logo for the application, other images, game objects, etc.

(100 marks)

Evaluation of software solution for each component (all items need to be shown during the group presentation):

Evaluation Component	Percentage
Functionality:	
MongoDB database (proper use of document structure/model)	10%
Correct Graph QL implementation (proper use of design patterns)	35%
Correct Front End (proper use of architecture/libraries/frameworks)	30%
Use of naming guidelines for functional components, variables, methods, comments.	10%
Innovation (intelligent use of symptoms or other data using deep learning)	15%
Total	**100%**

Evaluation rubric with criteria and level of achievement:

Criteria	Level of Achievement

	Failure to Minimal: 0 - 59%	Satisfactory: 60% - 69%	Good to Excellent: 70% - 79%	Excellent to Outstanding: 80 - 100%
MongoDB database (proper use of document structure/model)	Incorrect model, errors.	Model has some missing or incorrect fields.	Model has the correct fields, some constraints are missing	Excellent design/implementation of the document model
Graph QL API Design and implementation	Incomplete design/implementation of most CRUD functionalities, errors.	The design/implementation of some functionalities is missing or has errors	The design/implementation of functionalities is correct	Excellent design Graph QL API as per specs. Excellent use of design patterns.
Front End (proper use of architecture/libraries/frameworks)	Incomplete front-end, most components are not implemented, errors	The design/implementation does not follow the specs and UI is not friendly. Some elements are not aligned. Some components have errors.	The design/implementation mostly follows the specs. UI is not very friendly.	Excellent design/implementation of components, very friendly UI
Use of naming guidelines for functional components, variables, methods, comments.	No use of naming guidelines in most components	Some functional components, methods, variables are not named correctly	Most functional components, methods, variables are named correctly	Excellent use of naming guidelines
Intelligent use of symptoms or other data using deep learning	Not implemented	Does not work properly	The AI implementation is mostly correct.	Correct implementation and excellent use of AI, efficient model and high accuracy.

References:

https://docs.mongodb.com/manual/data-modeling/
https://expressjs.com/en/4x/api.html
http://mongoosejs.com/docs/guide.html
https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs
https://reactjs.org/

https://js.tensorflow.org/
https://www.tensorflow.org/js/tutorials/training/nodejs_training
https://vitalflux.com/key-deep-learning-techniques-medical-disease-diagnosis/
https://link.springer.com/article/10.1007/s00521-021-06426-4