Group Project – Developing Apps Using Emerging Web Technologies

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** and upload the solution on eCentennial through the assignment link on D2L. (Evaluation will not be done if this is missed and no marks will be awarded)
- You must publish the app on Heroku, Microsoft Azure, AWS, or any another Cloud platform.

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 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 **enter daily information (**for example *pulse rate*, *blood pressure*, *weight*, *temperature*, *respiratory rate*).
   b. Allow the user to use a **checklist of common signs and symptoms (COVID-19 for example)**, and submit the choices.
   c. 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.

4. Use **MongoDB Cloud** for storing the data.
5. **Use Express to implement Graph QL API and React for front-end (must use functional components)**

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

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

Evaluation Component	**Marks**
MongoDB Cloud database & Usage (proper use of document structure/model)	1
Registration & Login Feature	2
Correct Backend implementation	3
Correct Front End implementation	4
Elegant UI and user-friendly design	1
Use of proper naming guidelines for functional components, variables, methods, comments.	1
Innovation (intelligent use of symptoms, other data using deep learning / Game)	3
Proper usage of Github for collaborating with the team members (Commits, Pull requests, gitignore etc.)	1.5
Wireframes, Readme file, Project management (User stories and sprint management) in JIRA	1.5
The whole project is deployed to cloud properly	2
Total	**20**

	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 / Game	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