PART ONE: PSEUDO CODE

**Create and submit:**

1. A README.md document that contains the software requirements documentation:
   1. *System architecture:* 
      1. Describing the intended architecture for developing the application: provide motivation for the choice of architecture.
      2. What tools will be used to style the application: provide motivation for the choice?
   2. *System requirements specification:*
      1. Explain how the application will work, who will use the application and how they will benefit from using it.
      2. Explain what other software there is that does something similar, and the software will be different.
      3. list all the functional and non-functional requirements.
2. Create wireframes how the front-end (user-interface) will look and work.
   1. When creating/designing the wireframes/user-interface keep the following guidelines in mind:
      1. Keep the UI (*user interface*) simple.
      2. Create consistency and use the common UI elements.
      3. Be purposeful in page layout.
      4. Strategically use colour and texture.
      5. Use typography to create hierarchy and clarity.
      6. Make sure that the system communicates what’s happening.

PART TWO: PSEUDO CODE

**PROJECT REQUIREMENTS:**

Design any web application, as long as it meets the following requirements:

* It is built using Express.js, React.js, and MongoDB (the MERN stack),
* It creates, reads, updates, and deletes (CRUD) information from MongoDB.
* It has a custom server built using Express.
* It authenticates users using JWT.
* The front-end is built using React. Use a React framework (e.g. **Create React App** or Next.js) of your choice.
* The application should allow for normal end-user access and admin access.
  + An administrator should be able to monitor and make changes to users’ behaviour.

**Submission criteria**

* ***Code, test and debug the application designed in part 1.***

Submit:

1. The source code for the project. ***Delete/don’t submit the node\_modules*** directories and submit all the compressed project folders for the front-end and back-end. Ensure that the code meets the following criteria:
   1. The UI (*user interface*) should be attractive, easy to use and intuitive. Incorporate any suggestions received from the reviewer for part one regarding UI design.
   2. Ensure that the application has appropriately been tested include **at least** one **snapshot test** and the appropriate **unit tests** for **both** the front-end and back-end for the application.
   3. A reviewer should be able to launch the application by typing ‘*npm start*’ from the command line interface.
   4. Create a well organised and easy to understand and to use file structure of the project.
   5. The code should be well documented with appropriate comments.
   6. Write the code in a manner that is easy to read, adhering to Google’s style guide about indentation, **meaningful** variable names, and component names, etc.
   7. The code should be modular to make testing, debugging, code reuse and maintenance of the application easier.
2. An extended version of the README.md file in part 1 of the task. The README.md file must include:
   1. An clear explanation of how to use the application.
   2. Clear instructions that the end-user should be able to follow to install, test and run the application on their local machine – which includes:
      1. Instructions for modifying any MongoDB URI’s or API keys;
      2. A clear description of:
         * the measures taken to ensure the security of the application including a description of how API keys have been dealt with; any third-party API’s used in the code;
         * how the application has been deployed, how it was been deployed (*Back-end or front-end together or separately and why?* ); and
         * link or links to the deployed application.

Follow these steps:

1. Create a full-stack web application that meets ALL the listed criteria.
2. Deploy the application and add a link to the deployed application to the README.md for the project.
3. Push all the work generated for the project (including the design documentation generated in part 1) to GitHub.
4. Add a text file in DropBox with the GitHub link.