SIT209 - Individual Project

Topic 1: IoT Applications

- Create a new folder to store your project.
- Use git and github to create a new repository to manage the code versioning.
- Create a new web page as a welcome page for your smart-building.

Topic 2: IoT Data

- Add to the github repository you created for your individual project in topic 1.
- Add new pages to your individual project to allow the adding and removing of devices for your smart-building.
- Add a new page to display IoT device data. The suggestion is to think about the smart lighting and smart air-conditioning for the building.

Topic 3: Servers

- Add multiple pages for different activites, examples as follows.
    - Lighting
    - Air-conditioning
    - Security
    - others...
- Use a header and footer for the pages to have a consistent layout.
- Start creating each pages style, e.g. tables for each page to list the devices (Topic 4 will cover storing data)
- Style using bootstrap.
- Use github to manage this.

Topic 4: Service APIs

- Create an API server based on Node.js and Express.
- Use the API to provide access to a Cloud hosted MongoDB database.
- Use the database to provide data for your application, by creating mock data for your pages as following:

    - Lighting
    - Air-conditioning
    - others...
- Use github to manage this.

Topic 5: Automated Testing

- You should implement tests for any endpoints you have created in the API and use the Jest documentation, available at https://jestjs.io/docs/en/getting-started as a guide if you get stuck.
- Use github to manage this.

Topic 6: Documentation

- Use efficient comment in your code.
- Use automated API documentation by using apidoc.
- Create a technical report that summarises the features of the web site. Focus on a good style as well as good content.
- Use github to manage this.

Topic 7: M2M Communication

- Create a MQTT API server that will receive data on MQTT topics and insert it into your MongoDB database.
- Connect a sensor to an Arduino
- Read the sensor value and print it to the serial port
- Read this serial port in a node.js program.
- Use Axios to call the MQTT API.
- The MQTT API should save this data into the MongoDB based on an ID from the Arduino connected program.
- Modify your website to display the data.
- Use github to manage this.

Topic 8: IoT Lifecycle

- Add a plotly graph for some environmental data.
- Implemement a sense-think-act loop which will switch on a light based on data received from a MQTT device e.g. when a motion sensor is activated.
- Implement a sense-think-act loop which will activate a lighting scene when a user is present, based on their preferences.
- Use github to manage this.

Topic 9: Security

- Securing a web server with `HTTPS`.
- Using the `Helmet` middleware.
- Using the `Passport` middleware to login users.
- Use github to manage this.

Topic 10: Deployment

- Deploy your server.js to Heroku.
- Deploy your api.js to Amazon EC2.
- Deploy your mqtt.js to another cloud, either Heroku, Amazon EC2 or Microsoft Azure.
- As each server and the frontend need to know where each is installed, you will need to change the code for each to refer to the correct deployment location.
- Take screenshots to show these working together. Note that nothing should be running on your local machine.