Lab 5: Assignment Set Up

1. Purpose of this lab

This lab runs through the set up necessary for setting up the assignment server and provides some hints and tips on how to get started. By this point, you should have completed the previous four labs that contain most of what you need to know in order to develop a good assignment. The assignment requires you to write a web application, which can interface with an existing API using React Native.

2. Before going any further…

Before going any further with this lab, it is important that you ensure that you have completed all of the following:

1. Complete all of the previous labs
2. Download the server code from Moodle
3. Ensure that you have access to the Mudfoot server
4. Download the API specification YAML file from Moodle
5. Read and understand the assignment specification

3. Setting up the server

1. Open your terminal and navigate to the directory where your server code is.
2. Run 'npm install' (If you get any errors, try running it twice)
3. Open up the './config/config.js' file. You need to edit this file to point at your Mudfoot database:
   - Host: mudfoot.doc.stu.mmu.ac.uk
   - Port: 6306
   - User: Your Mudfoot username
   - Password: your Mudfoot password
   - Database: Should be the same as your username
4. Save that file and go back to the terminal window
5. Run 'npm start' – your server should start up successfully and tell you that it is listening on port 3333.
6. We can test that the server is running by opening a browser and navigating to 'http://localhost:3333/api/v0.0.3/' – you should see a message saying that the server is up.
7. Finally, to ensure everything is working, open a second terminal window and navigate to the same directory. Run 'npm test' – this will send a series of requests to the API and ensure that the responses coming back are correct.

4. Understanding the API specification

Open the API specification in Swagger (http://editor.swagger.io/) by copying the contents of the YAML file into the code box on the left hand side of your browser. Familiarise yourself with the different end-points.

4.1 Authentication

You will notice in Swagger that some of the end-points contain a pad lock icon on the right hand side. This means that the user must be authenticated (logged in) before calling these end-points.

1. Click on any one of these end-points and look at the 'Responses' section. Note that any user calling the end-point without authentication will receive a '401 Unauthorised' response.
2. Click on the 'POST /login' end-point. Notice that the request body expects an email and password, and the success response provides the users ID and a token.

All end-points that require the user to be authenticated verify the authentication using this token. Once a user logs in, the token needs to be sent in the header of every request that requires authentication. The name of the token element to include in the header (the key) is 'X-Authorization'

3. Use Postman to create a new user by querying the API
4. Login as this new user, again using Postman
5. Use the token to try accessing an API end-point that requires authentication (e.g. POST /chits)

5. Populating the database

Open Postman and spend some time familiarising yourself with the API by populating the database with some test data. Add some users, chits, followers and chits with photos.

6. Getting started with the assignment

6.1 Design and Wireframing

Taken from: https://www.usability.gov/how-to-and-tools/methods/wireframing.html

A wireframe is a two-dimensional illustration of a page's interface that specifically focuses on space allocation and prioritization of content, functionalities available, and intended behaviours. For these reasons, wireframes typically do not include any styling, colour, or graphics. Wireframes also help establish relationships between a website's various templates.

There are a number of benefits to creating wireframes:

- Connect the site's information architecture to its visual design by showing paths between pages
- Clarify consistent ways for displaying particular types of information on the user interface
- Determine intended functionality in the interface
- Prioritize content through the determination of how much space to allocate to a given item and where that item is located

There are two types of wireframes:

- Low-fidelity wireframes help facilitate project team communication and are relatively quick to develop. They tend to be more abstract because they often use simple images to block off space and implement mock content, or Latin (lorem ipsum) text as filler for content and labels.
- High-fidelity wireframes are better for documentation because of their increased level of detail. These wireframes often include information about each particular item on the page, including dimensions, behaviour, and/or actions related to any interactive element.

Exercise: Plan your applications design and structure by sketching some low fidelity wireframes. This can be done on paper and/or online wireframing tools such as Lucid Chart. Think about the ways in which users will interact and navigate through your application.

6.2 Next steps

The next steps are ultimately up to you. However, here is a recommended order to get you started:

1. Create an application
2. Implement your basic navigation structure and your project structure (i.e. how your JS files will be organised)
3. Implement the GET /chits end point.

Implementing these three ensures that you are set up with an organised project structure, have thought about design and navigation and get interact with the API. All of the information required to achieve this can be found in the previous labs.