RMIT UNIVERSITY

Further Web Programming
COSC2758/COSC2938 (semester 2, 2022)
Assignment 2

Assessment Type	To be attempted individually OR in a group of 2. Submit online via Canvas→Assignments→Assignment 2. Marks awarded for meeting requirements as closely as possible. Clarifications/updates may be made via announcements/relevant discussion forums.
Due Date	Week 12, Sunday 16 October 2022, 11:59 pm *Melbourne time*
Mandatory Demo	Week 13 (**NO DEMO → NO MARKS). *Schedule to be announced during 12.***
Marks	45

1. Overview (you must read this first)

In this assignment, you will develop a full-stack web application to complete the front-end prototype built from assignment 1. You are to use the following stacks:

- **Frontend:** ReactJS or ReactTS
- **Middle layer**: Node.js & Express.js with Sequelize ORM
- **Backend database**: MySQL

- You are **not** allowed to change technology stacks to suit your convenience and/or knowledge.
- Using stacks other than the ones mentioned above will **FETCH A ZERO** for the whole assignment.
- Use of Object-Oriented React will **FETCH A ZERO** for the whole assignment.

The tasks are divided into *four* parts: PA (Pass), CR (Credit), DI (Distinction) & HD (High Distinction).

> The DI & HD section tasks will require self-research, you will not get straight answers in the course material. While we are happy to assist you on those tasks, most of the work and research must be done by you. This is done on purpose to prepare for you future work and rigours of the IT industry.

If you find a specification open to interpretation, post a query identifying the specification in the corresponding discussion board for assignment 2. Software development in real life does not come with a definitive roadmap and flowcharts complete with instructions. More often than so, it is the job of the developer to clarify requirements from the client. For this assignment and course, the lecturer is considered as the client.

All of us have been affected by the unfortunate COVID-19 scenario and its aftermath. It is often hard to concentrate and study; but as a student enrolled in this course, it is your responsibility to regularly attend lectorial, lab and consultation session(s).

- Bring your questions to online discussion board, consultation sessions
- Watch the online recordings on a regular basis if you cannot attend the live sessions.
- Do NOT start the work on assignment at the last minute.
- Do NOT ask for last minute extensions, these are often rejected. Extensions can only be granted for personal and medical reasons, provided you can supply some evidence.

You are strongly recommended to work in a group as this is a longer assessment. If you want to join a group or change your assessment 1 group, please email lecturer at your earliest.

2. Learning Outcomes

This assessment relates to the following learning outcomes of the course which are:

CLO2: Demonstrate proficiency with a web application development framework

CLO3: Implement a range of techniques and procedures for developing a small to medium-scale web application

CLO4: Demonstrate knowledge of and utilise software engineering patterns in development

CLO5: Design and manage the development life cycle of a complete application.

3. Assessment details

The senior executive committee has accepted the prototype of LAN (**L**oop **A**gile **N**ow) website and now recommends that full stack version of website be developed. The committee recommends inclusion of extra features and constraints for the LAN website- a summary is presented here; you will find more details in **Section 4: Tasks**. The extra features and constraints include-

3.1. The committee wants the developer to use a Cloud MySQL database for the backend purposes.

3.2. Two users cannot have the same username. Password must be stored in a hashed format in the database. **Please do not use MD5 as it is no longer considered secure by industry professionals**.

3.3. The full stack version of LAN must be a multi-user website where users can follow and unfollow one or more other users. Posting form must have an option for uploading photos.

3.4. User can now submit interactions on posts such as: *more details in the tasks section.*

3.5. LAN website will have a separate *admin portal* for moderating posts and generating user statistics. Admin dashboard is accessible via a separate URL to LAN website.

3.6. Admin dashboard part must be implemented via GraphQL and not REST API.

3.7. Codebase must be well-commented.

3.8. GitHub should be used throughout the development lifecycle. The repository must be a part of **rmit-fwp-s2-2022** organisation account. You have been emailed an invite to join this organisation during week 1 of the semester; if you did not accept the invite, please email lecturer at shekhar.kalra@rmit.edu.au

3.9. The website must be fully styled and look professional. The content must make sense i.e., use of *lorem ipsum is not allowed.*

3.10. The digital assets (images, icons, audio & video) must be outsourced from free websites. You should not steal someone else's assets to enhance the look and feel of your website. High-quality & free assets can be obtained from:

https://unsplash.com/	(Images)
https://uifaces.co/	(Avatars)
https://fonts.google.com/icons?selected=Material+Icons:home	(Icons)
https://www.flaticon.com/	(Icons)

To proceed to higher parts, you must complete all the specifications in the lower part, you must not cherry pick specifications from various parts. As an example, complete all the specifications in PA part before proceeding to CR part and so on. Please proceed to next page for the tasks.

4. Tasks

Create a **full-stack** LAN website with – **ReactJS, Node.js, Express.js (with Sequelize ORM)** & **Cloud MySQL database. Here is the architecture diagram for the website:**

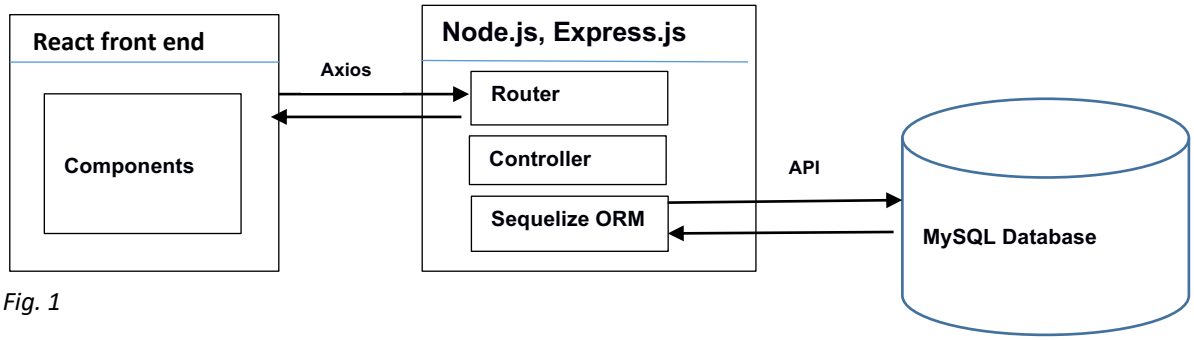

Fig. 1

React front-end app talks to API defined in Node + Express layer via Axios library. API created by you in Node + Express layer (*middle layer*) communicates with the backend database. Create separate projects – *one for React frontend app* & *one for Node + Express + Sequelize middle* layer. The tasks are shown below:

PA part [23 marks]

a. (6 marks) **Database schema model files**

Create an ER (Entity-Relationship) diagram that will represent the database schema for the website. *This is for your draft work*, there is no need to submit the diagram. It should display the tables with the fields, keys, constraints, and relationship(s) between the tables. Think of these points- *How many tables do I need? Which fields do I need in these tables? What datatypes should these fields use? What kind of relationships exist among these tables? Is the database normalized? (i.e., avoid duplicated data, do not use too few or too many tables)*.

Create model files that represents the above tables, keys and constraints using Sequelize in the Node.js + Express.js (middle layer) project.

b. (3 marks) **Sign-up page**

Implement the sign-up page from assignment 1. This time the user details are stored in the MySQL database. The API in the middle layer should handle all the database operations. All the form input validations must be handled on the React end.

c. (3 marks) **Sign-in page**

Implement the sign-in page with API handling authentication to verify the username and password. All the form input validations must be handled on the React end. Upon logging, the user must see a welcome message in the format of *Welcome* username. Introduce a **logout** link for a logged in user.

d. (6 marks) **Profile** & **Profile management features**

Implement the profile and profile management features from assignment 1. The details of user profile must be fetched and modified via the API.

e. (5 marks) **Unit tests**

Add five meaningful (*significant and contextual to the tasks b, c & d*) unit tests. You must explain the purpose of each of the test as comment entries in the test file.

CR part [8 marks]

f. (6 marks) **Posting feature**

Implement the posting feature from assignment 1 with the following additional requirements-

1) User post must be saved in the MySQL database via the API.
2) User may reply to their own post or to another user's post.
3) The maximum length of a post increased to 600 characters and it can now include formatted text.

(2 marks) Add a contextual unit test for this specification.

You may retain your image handling logic from assignment 1.

Note: For the following (DI and HD sections), you will need to do some independent research.

DI part [4 marks]

g. (1.5 marks) **Follow** & **Unfollow**

Add a new feature to LAN- a user should be able to follow and unfollow another user. The follow and unfollow logic must be written in the API and appropriate details stored in the MySQL database.

For this part, a user should be able to see a list of users they can follow. For each user you will need to keep track of whom they are following. When a user is followed, all their posts can be accessed.

The user should be able to view a list of the users they are following in the React app with an option of unfollowing them.

The design of the user interface is left for you.

h. (1.5 marks) **Post reactions**

A core feature of any social media platform is the ability for users to interact with shared content. For the posts that are created on the LAN website, add the options to like and dislike individual posts. The logic of post reactions may be written in the API and the appropriate details stored in the MySQL database.

Do some research, look at various social media platforms. Each post or comment must support two types of reactions.

The design of the user interface is left for you.

(1 mark) Add a contextual unit test for this specification.

Please proceed to next page for the HD tasks →

HD part [10 marks]

Please note the requirements for this part:

Your task is to create an Admin Dashboard with the following features-

Requirement # 1: Create separate project(s) for this section.

Requirement # 2: The look and appeal of the admin dashboard must be different to the LAN website. Spend some time thinking of an appropriate user interface.

Requirement # 3: The admin dashboard must consist of multiple components with a nested hierarchy. The hierarchy must be at least **3-4** levels deep (as an example App > Dashboard > Component 1 > Component 2). This is to make sure that you understand and correctly implement hooks for the state management.

Requirement # 4: You must utilise **useReducer** and **useContext** hooks for the state management for this part.

Requirement # 5: **For the data fetching part, you must use GraphQL- i.e.,** use of **REST API** is **not allowed.**

i. (2 marks) Admin should be able to perform the following (**database fetch via GraphQL**)-

1) Delete posts(s) if deemed inappropriate (you will need to explain during demo why a post is considered inappropriate?)
   Deleting a post must show some corresponding message on the user page in LAN website such as- *[**** This post has been deleted by the admin ***]*

2) Block and unblock a user account: *blocking a user will not allow a user to login until the admin unblocks the account*

ii. (4 marks) Admin should also be able to generate the following data analytics visually (**database fetch via GraphQL**):

1) Number of users using LAN per day.
2) User post reaction metrics
3) Profile visits
4) Follower metrics for each of the user

Each of the metric must be graphically depicted. No marks will be offered for a pure tabular representation.

Note: do you understand the exact nature of these metrics? You need to do some reading first and find out how social media websites/apps use these metrics. Then define your understanding of the metric in the corresponding code file(s).

k. (4 marks) Create a GraphQL **subscription** so that admin can find out if a post is getting too many *dislikes or downvotes*. This real time feature will alert the admin of a possible lynching, boycotting or harassment of a particular user and their post. Admin can then take an action of blocking responsible users or deleting the post.

You will need to add some feature on the dashboard where admin can see the alert message raised by the underlying subscription.

Submission instructions →

5. Submission & **Mandatory Demo**

- Zip all website files **EXCEPT NODE MODULES FILES** and submit single zipped archive with .zip extension via Canvas submission link for this assignment.
- You must demo your assignment in week 13 - a schedule and a booking link will be published closer to the deadline. **No demo will lead to no marks**. **You must submit the assignment prior to the demo**.

After the due date, you will have 5 business days to submit your assignment as a late submission. Late submissions will incur a penalty of 10% per day. After these five days, Canvas will be closed, and you will lose ALL the assignment marks.

Assessment declaration:

When you submit work electronically, you agree to the assessment declaration:

https://www.rmit.edu.au/students/student-essentials/assessment-and-results/how-to-submit-your-assessments

6. Academic integrity and plagiarism (standard warning)

Academic integrity is about honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e., directly copied), summarised, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods,
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviours, including:

- Contract cheating- paying someone to do your work
- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students
- Posting assignment tasks on technical forums (*reddit, stack exchange, etc.*) and asking for solution(s)

For further information on our policies and procedures, please refer to:

https://www.rmit.edu.au/students/student-essentials/assessment-and-results/academic-integrity

7. Marking Guidelines

The marks allocated have been added to each of the tasks. **Please read rubrics for details**.