Management System

CS 340 Databases - Assignment 4

Due Date: 5th December

Lead TA's: Aima Shahid, Zoha Hayat Bhatti

1. Introduction

This manual serves as a thorough guide for the development of a Management System using the MERN stack (MongoDB, Express.js, React, and Node.js). Your task is to develop a management system that incorporates essential functionalities such as user authentication, user management, and data manipulation. The primary aim of this assignment is to facilitate your comprehension of real-world applications that use noSQL Databases like MongoDB.

This assignment builds upon the work you did in PA2, where you conceptualized the schema and outlined the various use cases for your designated management system. Your task now is to take those concepts and transform them into a functional web application for your management system. You have to choose one management system to implement from the management systems you and your partner were allocated in PA2. You have to enter this information here.

2. System Setup

The provided zip file contains the pdf for this manual as well as a folder. The folder contains subfolders for both the client and server sides of the application.

2.1 The client-side:

React is an open-source JavaScript library used for building user interfaces in client-side web applications. To help you get started, the boiler code for react has already been provided to you in the client folder. To be able to run the code without errors, you will need to install node modules inside the client folder. Follow the steps below:

1. Navigate inside the client folder by running **cd client** in the terminal

2. Execute **npm i** to install the required node modules and dependencies listed in the package.json file.
3. Start the React application by running **npm start**, which will automatically open the application in your web browser. You should be able to see the login page.

2.1 Server-side, Creating Database and Connecting to MongoDB

In the server-side folder, you will write server-side code and establish the database connection. For this assignment, we will use MongoDB as our database. You can set up your server-side environment by following the first four videos in this playlist**.**

You will have to make an account on MongoDb Atlas before creating a new cluster for your database.

To run the server side:

1. Navigate inside the server folder by running **cd server** in the terminal.
2. Start the server using **npm start.**

Once you have made the connection, you will first have to start the server and then start the client-side in parallel.

3. Use Cases

Your Management System should cater to at least **three** main user roles, each with specific actions within the system. You should come up with at least 3 use cases per actor in addition to the mandatory ones listed down below. You can simply implement your use cases from PA2. All use cases **must be documented in a separate file**, which should be submitted along with your code files. This can be in the same format as PA2. After outlining the use cases, you are responsible for implementing them in your web application.

Mandatory Use cases:

Here are a few use cases that are necessary to incorporate in your management systems.

3.1 Signup

- Members can register using their email and password.
- Passwords should adhere to security standards and specific format.
- This email and password should be stored in the database.
- You will have to implement a signup page for this.

3.2 Login:

- Only registered members can log in using their email and password.
- You will have to implement a login page for this. Only logged in users should be navigated to the Welcome/Home page.

3.3 Search for items:

- Your web app should include a search bar where you can search for items. (These items may vary depending on what management system you choose to go for. E.g. in a library management system you can search for available books)

3.4 Sign Out

- The user should be navigated back to the login page when they sign out.
- The user should no longer be able to navigate to the home page unless they log in again.

4. Checks

- Implement checks during registration to ensure password complexity and valid email formats.
- Mandatory fields must be completed during both registration and login processes (No form should be capable of being submitted if it has empty fields).
- Only registered students and staff members should be granted access to the system (Jumping from the login page to the internal pages through the URL should not be possible without logging in).

5. Submission:

- You are to submit a zipped folder containing all the code files and a pdf with a short description of all your use cases.
- Please make sure you delete node modules before zipping the files.

6. Grading:

This assignment is out of **100 marks**. The grading breakdown is as follows:

Breakdown	**Marks**
Setting up MongoDB	10
Login	10
SignUp	5
Search	10
Signout	5
Other Use cases	9 * 5 = 45
Implementing checks (input format + required fields + authorized access)	2 + 2 + 6
Viva	5
Total	**100**

Please note that there are no marks for the UI of your web application. Your web application does not need to look pretty. It just needs to be functional.