Software Design and Engineering

Lab Document

https://github.com/Owen-Carpenter/MERN-Stack-Authentication

High Level Purpose Statement:	My goal for this lab is to create a user authentication system using MongoDB. I will use NPM for dependencies using the MERN(MongoDB, Express, React, Node) technology stack. I will use NPM for configuring the run environment (Needs to install directory dependencies after reviewing Lab 3). This project will all allow a user to login/register an account.
Experimental Design:	GOAL IS TO MAKE SIMILAR TO LAB3 FOR COMPARISON First, I will use Vite to set up the React TSX app and I need to create the backend using MongoDB, Express, and Node. After the boiler plate directories have been created, I will first start on the frontend. I can easily use form elements along with the Axios library to asynchronously handle server responses to the Login and Register pages (Much like lab3 w/ Postgres). While creating the markup language, I will style the pages. After the frontend API routing has been set to localhost port 8080 for server sided logic, I will start the backend. For the backend, I need to implement a MongoDB schema. This schema is very simple, as in it will only store the players *name, password, and email*. All of these fields are required in the schema and the inputs are all String types. Using MongoDB atlas to connect to the database, I need to create a connection string so that any user can connect from any IP address and are all given “user” access. Once this has been initialized inside of the .env, It is now time to set the express server up using the routes for each auth,register, and logout controllers. The controllers are used for the logic and NoSQL document queries for checking login/registry info.
Resources Available:	References used using MongoDB with the MERN Tech Stack: https://gist.github.com/manuelbieh/3864088 - Batch File Implementation https://github.com/djizco/mern-boilerplate/tree/master/server - MERN Boilerplate
Time Estimate:	I think that I will spend about eight hours on this project. For the first three hours I will set up the frontend using Vite and then I will create the Login and Register pages using TSX. For the next two hours, I will set up the connection string, NoSQL Schema, and Express server in the backend. For the last three hours, I will set up the batch file, NPM dependencies, and the NPM scripts.

Experiment Notes:	● Using NPM for adding various dependencies is very easy ● Vite allows a basic template for using TSX or JSX ● The connection string and Atlas make it much easier in my opinion to set up compared to Postgres. ● Using NoSQL you can find attributes by using the findOne function, this makes it super super simple to find what you are looking for in the document. ● MongoDB is quicker and has fewer lines of code. ● It took some time to set up some npm dependencies, each time the repo was cloned, they wouldn't save. In order to fix this, I saved the dependencies to –save-dev, also I called an npm command to install any faulty dependencies that don't save. This is all ran "concurrently" in the root json ● After using the MERN boilerplate, it is pretty straight forward at implementing a login/registration system.
Results:	The project now properly runs using NPM and the MERN stack to authenticate login or registration through a React form field. NPM acts a lot like Maven or Gradle in this case because it is used for dependency management and run configuration. Once I configured the scripts within NPM all the user has to do to run this project is open the terminal and type "npm start".
Consequences for the Future:	In the future, I should save some dependencies to the devDependencies so that they are not deprecated, and account for machines that do not already have NPM or NodeJS installed. I should also run npm install in each directory to ensure everything is properly set up before running (user now shouldn't run into any problems).