Software Design and Engineering

Lab Document

https://github.com/Owen-Carpenter/PERN-Stack-Authentication/tree/main

High Level Purpose Statement:	My goal for this lab is to create a user authentication system using Postgres. I will use NPM for dependencies using the PERN(Postgres, Express JS, React JS, Node JS) technology stack. I will also use NPM for configuring the run environment of the project. This project will allow a user to login or register an account.
Experimental Design:	First, I will use Vite to set up the React TSX app, and I need to create the backend using Postgres, Express, and Node. After the boiler plate directories have been made, 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. At the same time, I will style the Login and Register pages. After the frontend API routing has been set to localhost port 8080 for server sided logic, I will start to work on the backend. In the backend I need to first create the database model for the users table. This table will accept a primary ID, name, email, password, and timestamp. In order to use this Postgres database, I need to implement a connection pool. This is a user that will be able to operate the database using CRUD operations. Once the pool and SQL model has been made, I will use the Express framework to set up the server and connect to the database using the pool file. Now that the server is properly set up, it is time to create some controller files to authenticate user information while logging or registering in. These files will use functions to validate their process using async. Inside of the authentication and register controllers, Bcrypt is used as a dependency to encrypt passwords.
Resources Available:	References used using Postgres with the PERN Tech Stack: https://gist.github.com/manuelbieh/3864088 - Batch File Implementation https://dev.to/cwrcode/create-css-fireworks-animation-3nn7 - Firework Background animation https://alvarotrigo.com/blog/animated-backgrounds-css/ - #12 Particles Animation https://github.com/sambuddha92/pern-boilerplate - PERN 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 pool, sql model, and 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 ● I like how the Connection Pool can be modified inside code in order to setup the connection to Postgres ● It is very simple to make queries to the users table when it is SQL, although MongoDB in my experience is much 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. ● After using the PERN boilerplate, it is pretty straight forward at implementing a login/registration system.
Results:	The project now properly runs using NPM and the PERN 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.