**Reverse Engineering Authentication**

horizontal line

# 

# This is a tutorial on a reverse engineering node JS application using Sequelize and Passport

## Walkthrough to understand the codebase by Ryan Stout

**Overview and Purpose**

**This tutorial serves as a guide to help developers familiarize themselves with a new codebase. This codebase can be used to start a new project**.

What does this codebase do?

**This codebase is for password authentication. Allowing users to create an account to log in and out securely using website files. User data is stored in a mySQL database.**

User Story

President of the company SugarFlakes wants to hire a coding developer that will build him a website that has him safely store all his data in a password protected account. That password will then be stored into a database and lets him use that database to retrieve data to let him log in and out of his site safely and securely.

## Table of Contents

Overview and Purpose

File Directory Map

Walkthrough the Code

Quick Instructions

File Walkthrough With Steps

Testing

How to Improve Codebase

Resources

## 

## 

## 

## 

## 

**File Directory Map**

**Below is a directory of the file structure**

* + for new files/folders added in initialisation phase
* Highlighted folders require editing to make the code work

Folder-Develop

Folder-config

Folder-middleware

File- isAuthenticated.js

File-config.json

File- passport.js

Folder-models

File-Index.js

File-user.js

Folder-Node Modules (+)- (This folder has multiple subfiles that would make the tree to big to create)

Folder-bin

Folder- Public

Folder- js

File- login.js

File- members.js

File- signup.js

Folder- css

File- style.css

File- login.html

File- members.html

File- signup.html

Folder- routes

File- api-routes.js

File- html-routes.js

File- database\_production.sql

File- database\_test.sql

File- package-lock.json

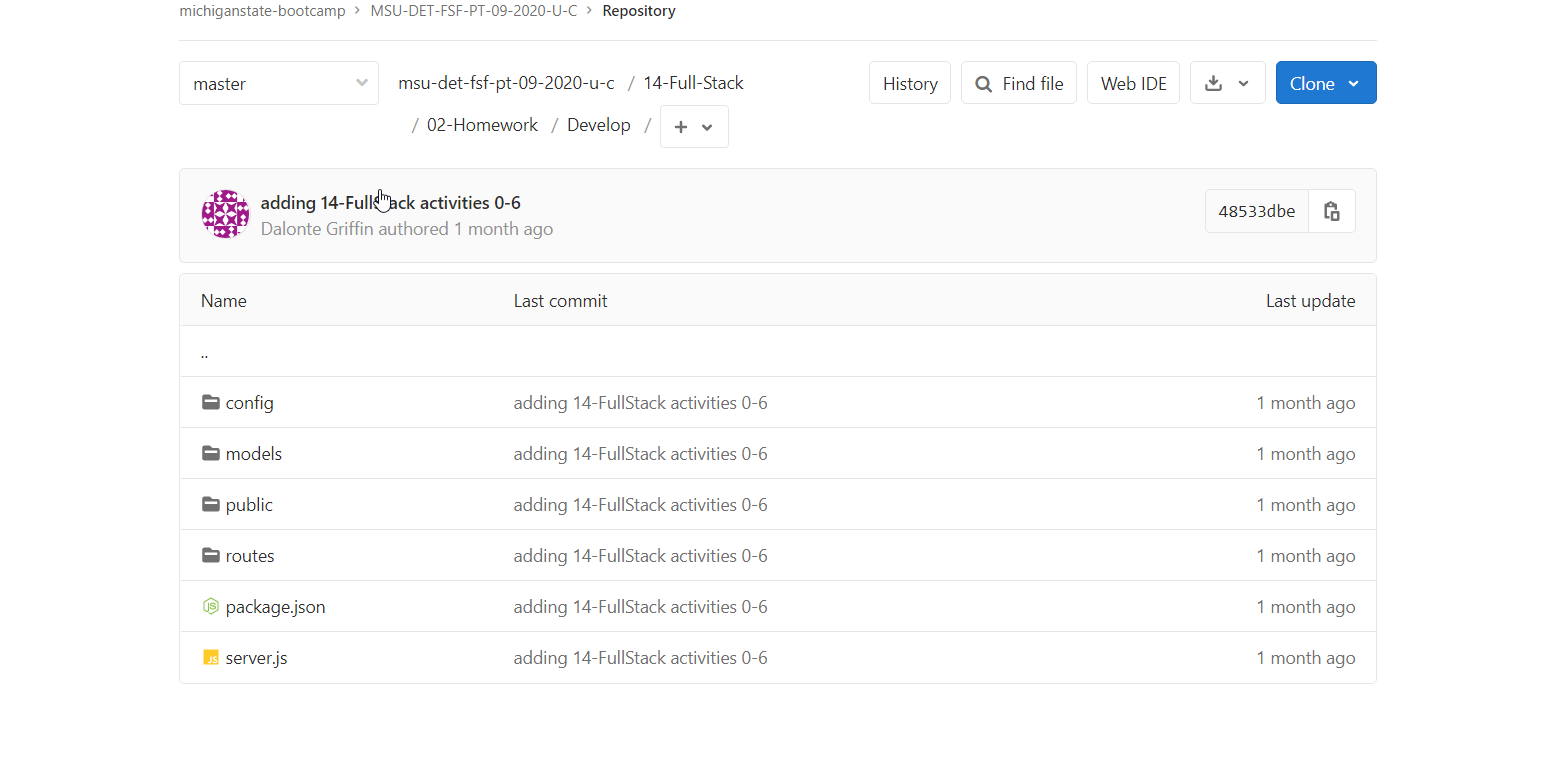
File- package.json

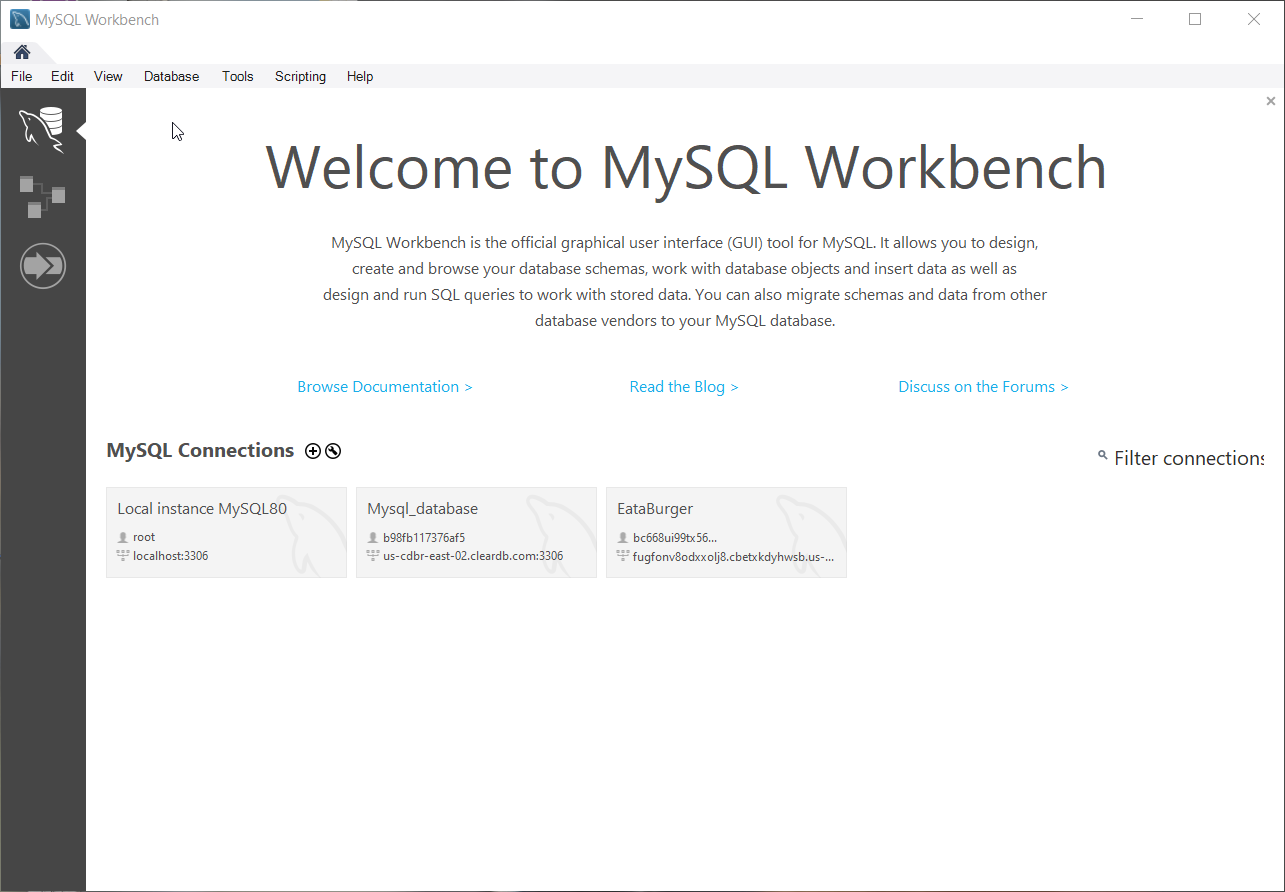
File- passport\_demo.sql

File- server.js

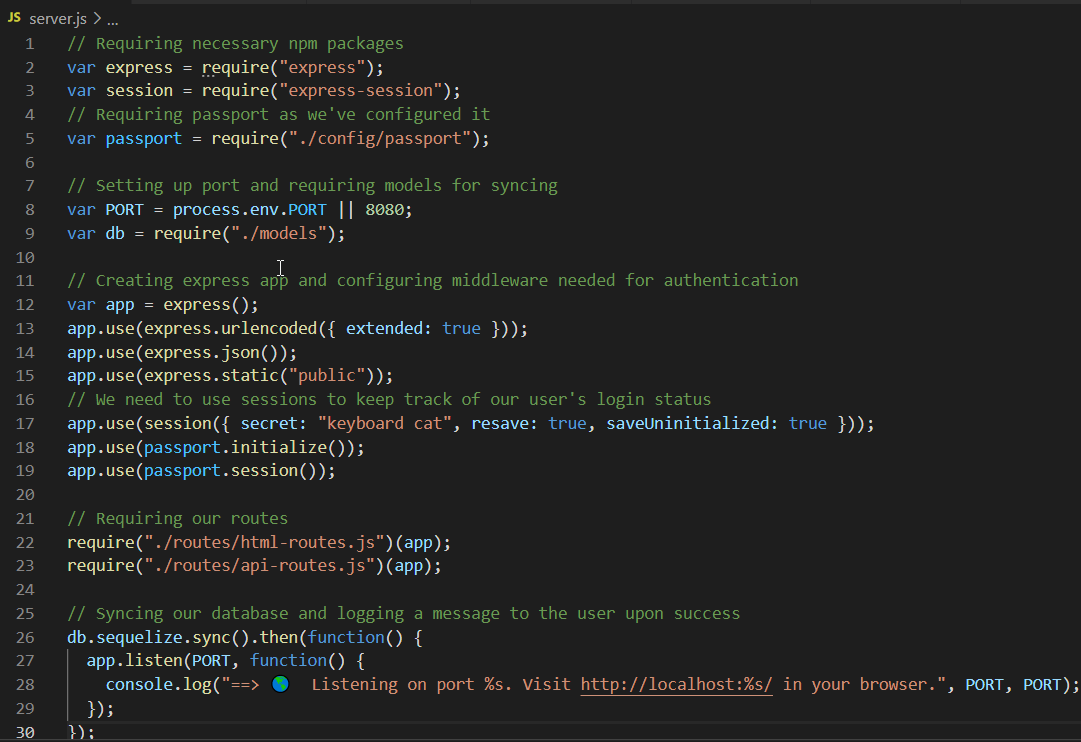
**Walkthrough of code**

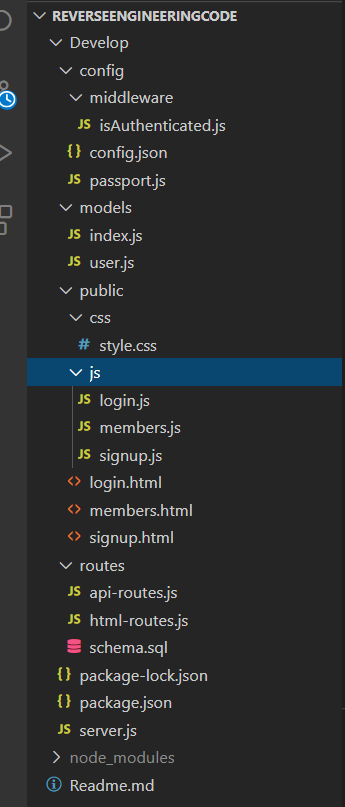
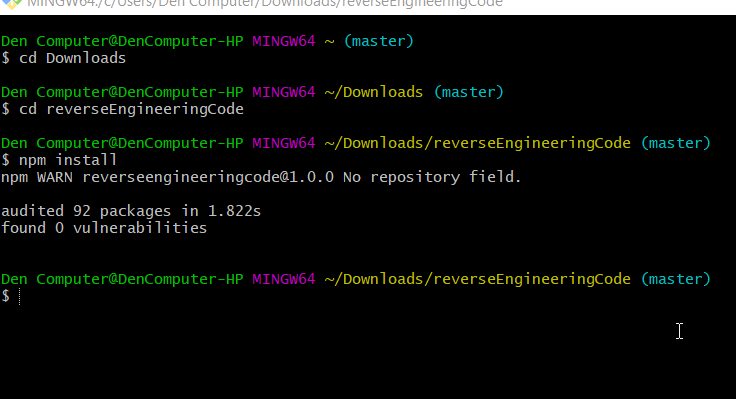
**Quick Instructions**

1. Use the codebase by first cloning the repository to your local storage. ****
2. Once complete- change const/lets in code to var to make code run.
3. Create SQL scripts for 3x databases as named in config.json file.
4. Open and run scripts in mySQL to create the databases.



1. Edit the config.js file and include personal data (i.e. password for mySQL)
2. Open integrated terminal and run “NPM install“ to install required node modules for the project.





6. Run npm audit fixes if any issues.

1. Run Node server.js to check if the server is working.
2. Test HTML features ( 3 sites ) by opening the local server link in the browser.
3. Check the mySQL database to make sure login data is saved.
4. That’s it.

**File Walkthrough the steps**

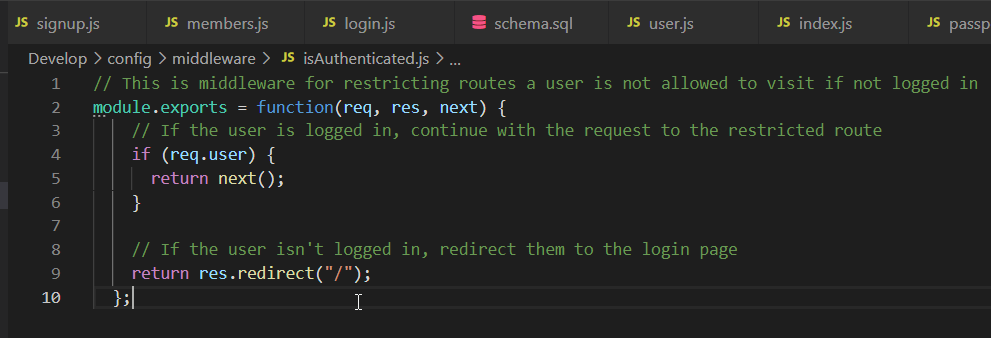
***Folder- Develop***

***Folder- config***

***Folder- middleware***

***File- isAuthenticated.js***

(Middleware prevents users unauthorized access to routes when they are not logged in.)



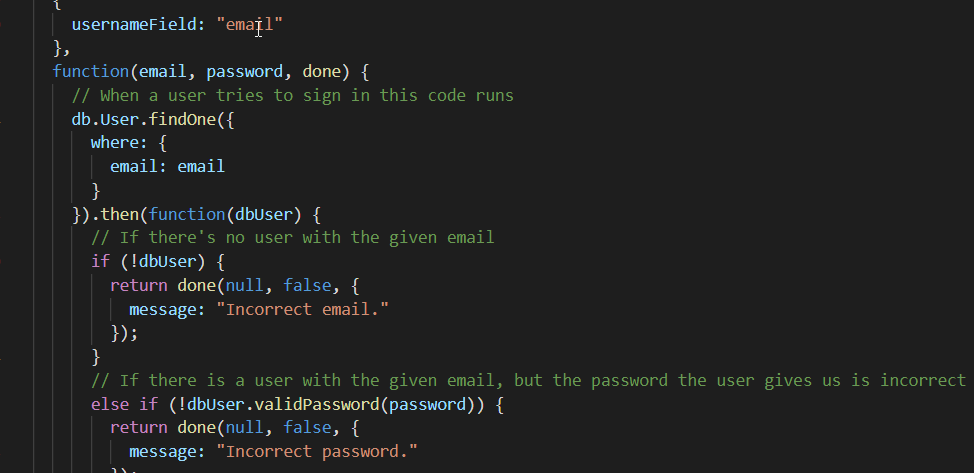
***File- config.json***

(Middleware connection configuration to connect to the server)



***File- passport.js***

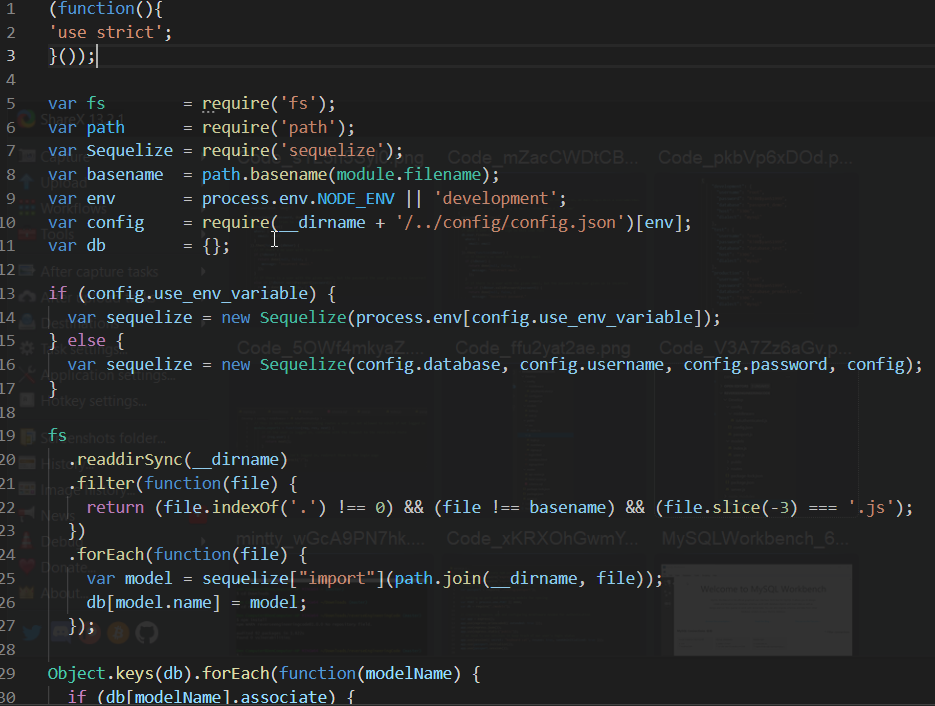
(Middleware contains javascript logic that tells passport that we want to login with an email address and password.)



***Folder- models***

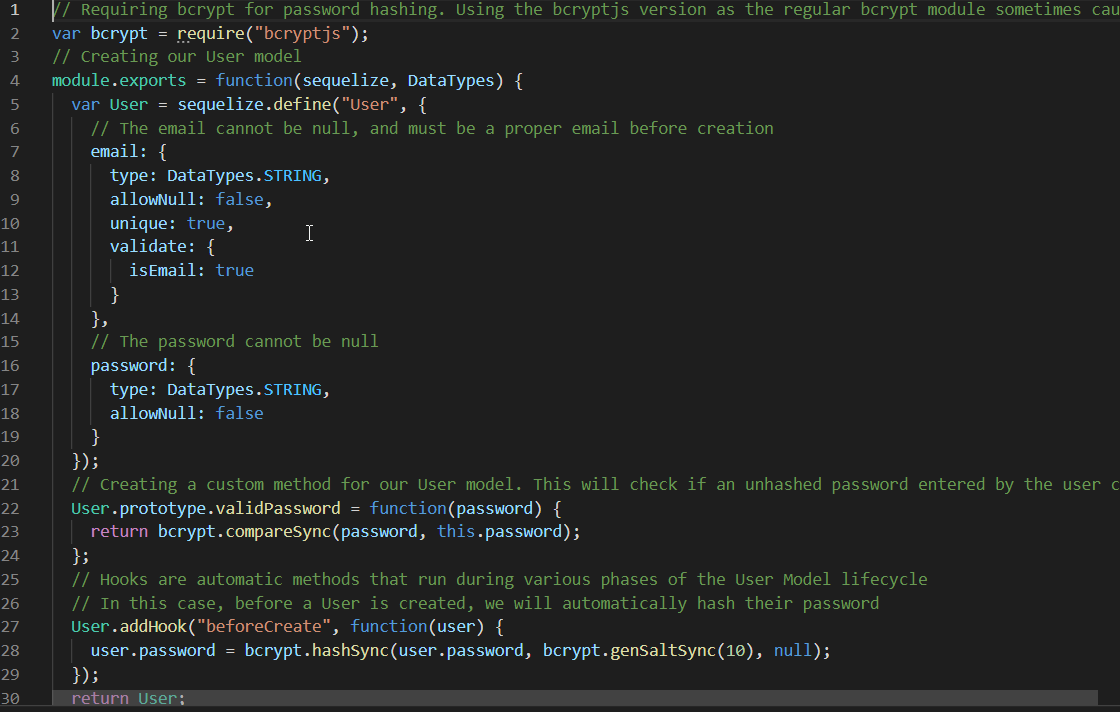
***File- index.js***

(This model connects to the database and imports users login data.)



***File- user.js***

(Requires “bcrypt” module for password hashing. This feature keeps the database secure even if compromised. The javascript defines what’s in the database.)



***Folder- node\_modules(+):*** (Sub folders are not included in the tree as it would be too large to display. Required installation modules are located in the package.json file. Run NPM install to install the required modules.)

***Key Node modules are bcryptjs, express, express-session, mysql2, passport, passport-local, and sequelize***

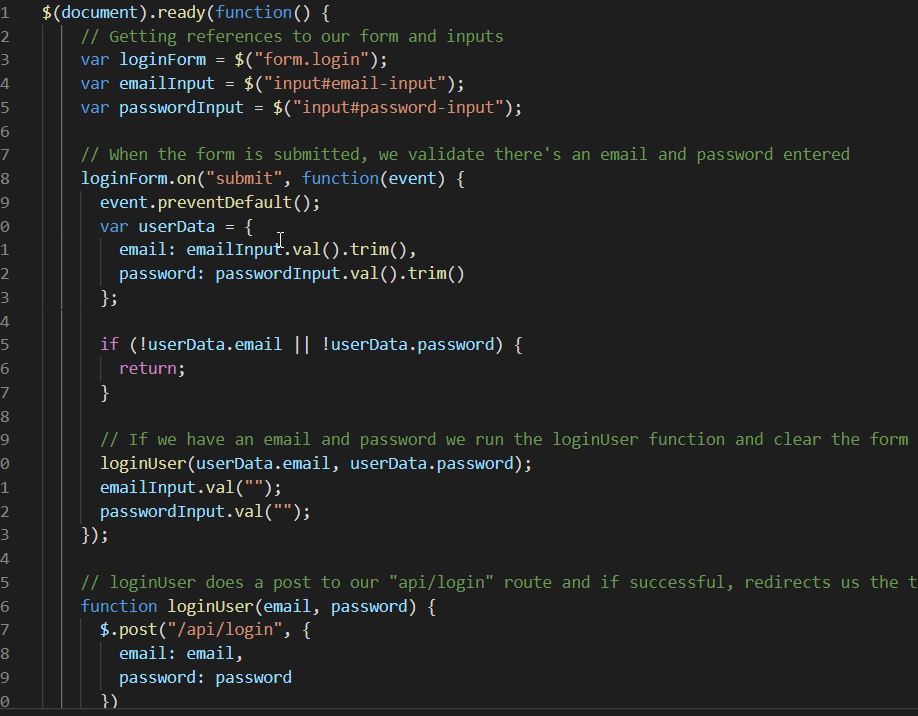
***Folder- bins***

(Stores executable files so node module functions can work when on the local server.)

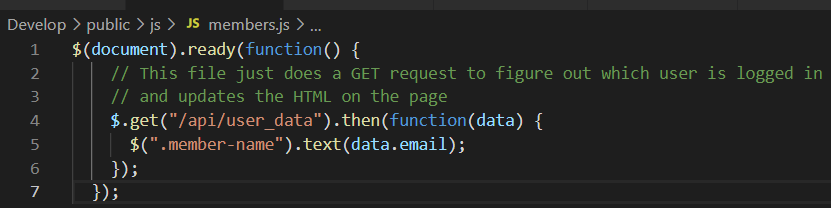
***Folder- Public***

***Folder- js***

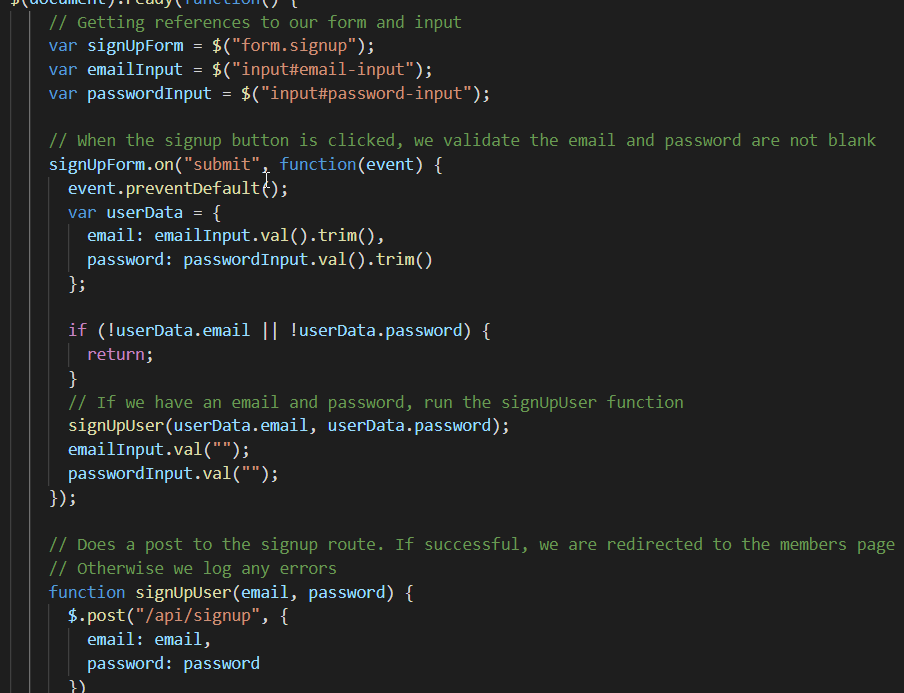
***File- login.js***



***File- members.js***

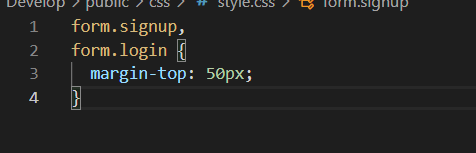
******

***File- signup.js***

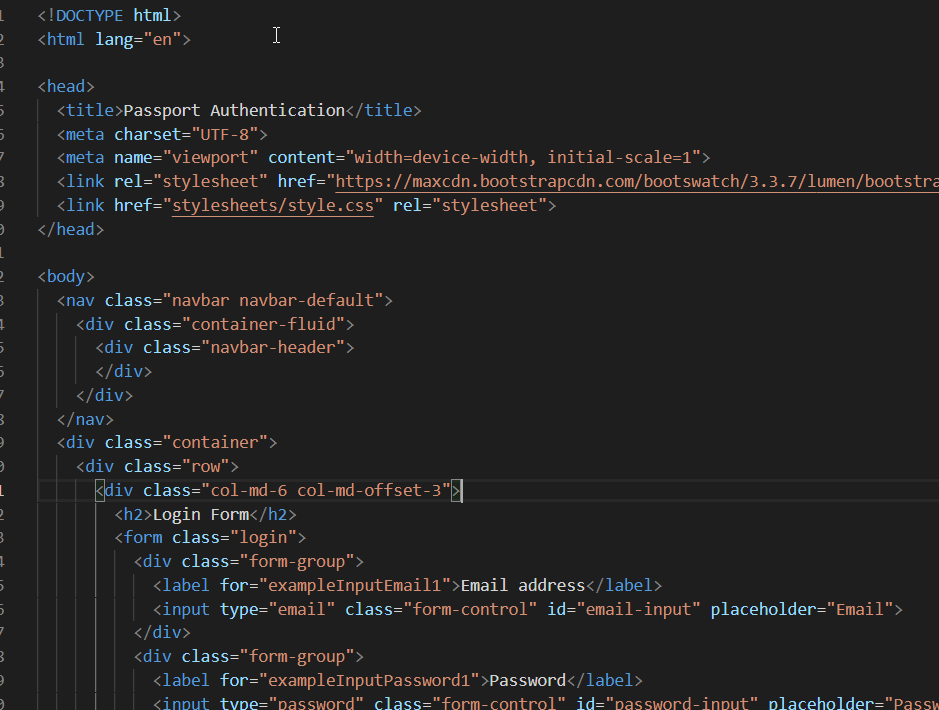
******

***Folder- css***

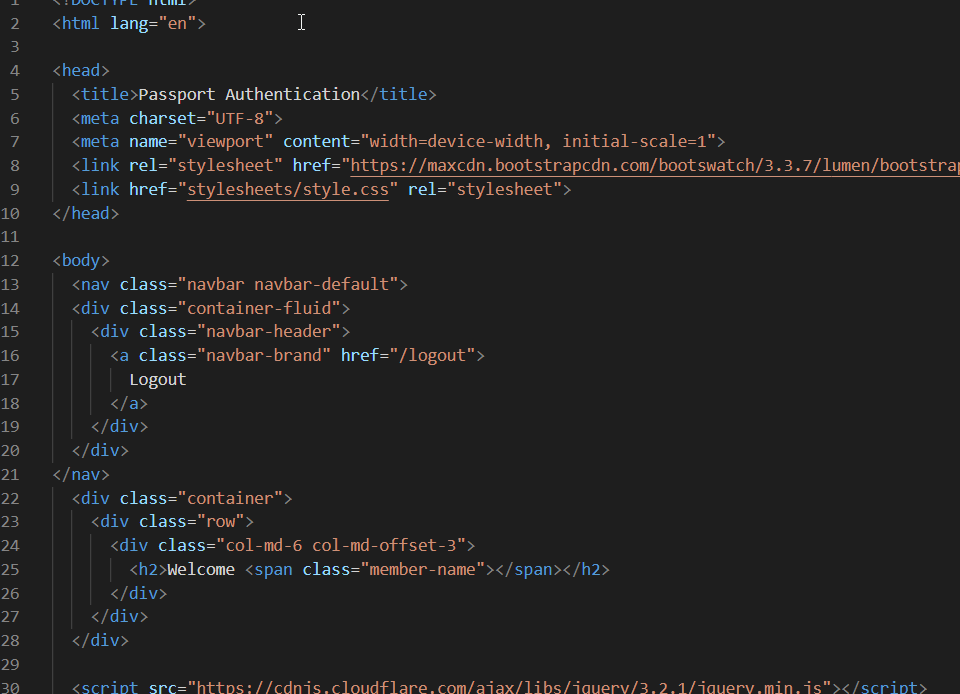
***Folder- style.css***

******

***File- login.html***

******

***File- members.html***

******

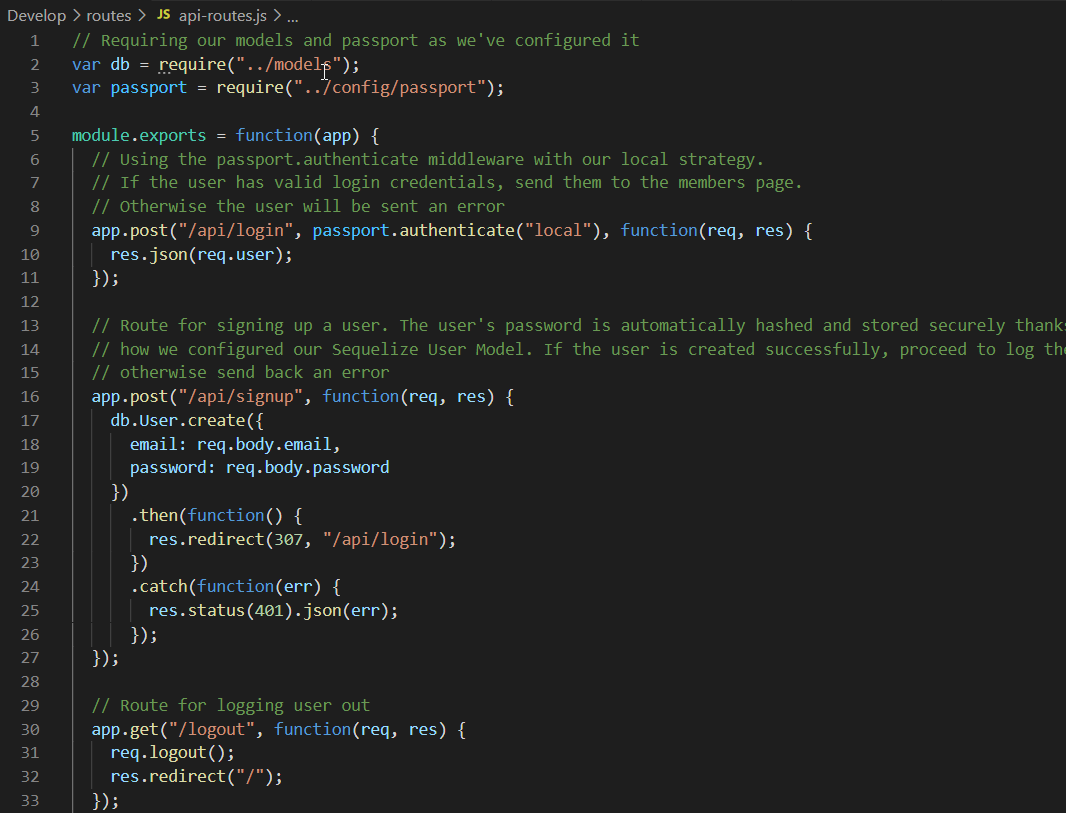
***File- signup.html***

******

***Folder- routes***

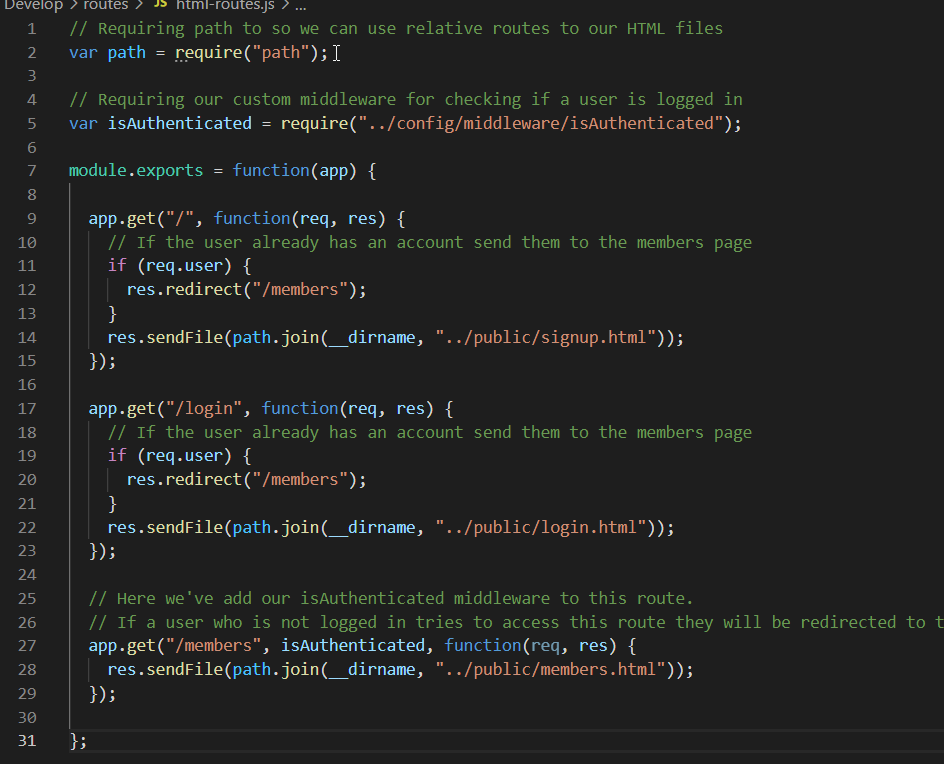
***File- api-routes.js***

(Contains routes for signing in, logging out and getting users specific data to be displayed client on the client (browser) side.)

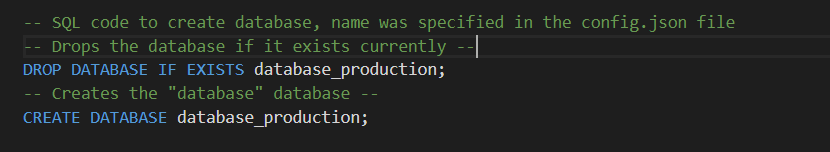
******

***File- html-routes.js***

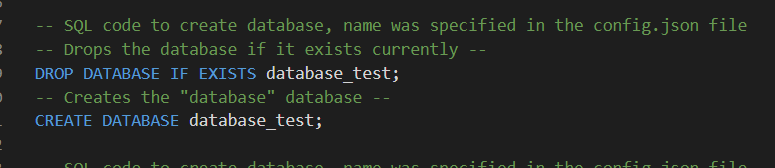
(Routes that check whether the user is signed in, whether the user has an account and sends them to the correct page.)



***File- database\_production.sql(+)***

******

***File- database\_test.sql(+)***

******

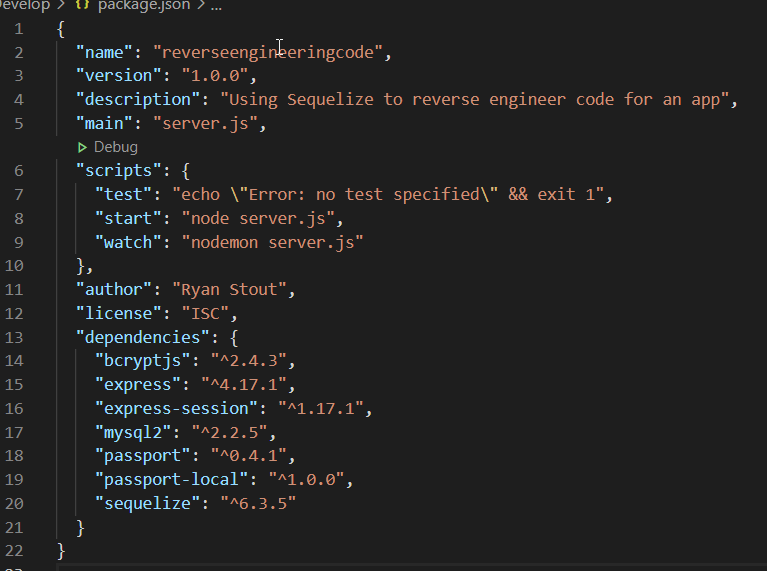
***File- package-lock.json***

(*This only contains one view of the file, code goes down to line 693 once all node modules are installed.)*

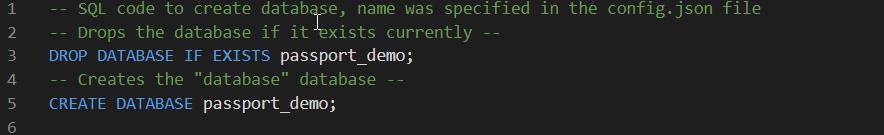
**

***File- package.json***

(Contains all pack info, node modules used, version info, etc.)

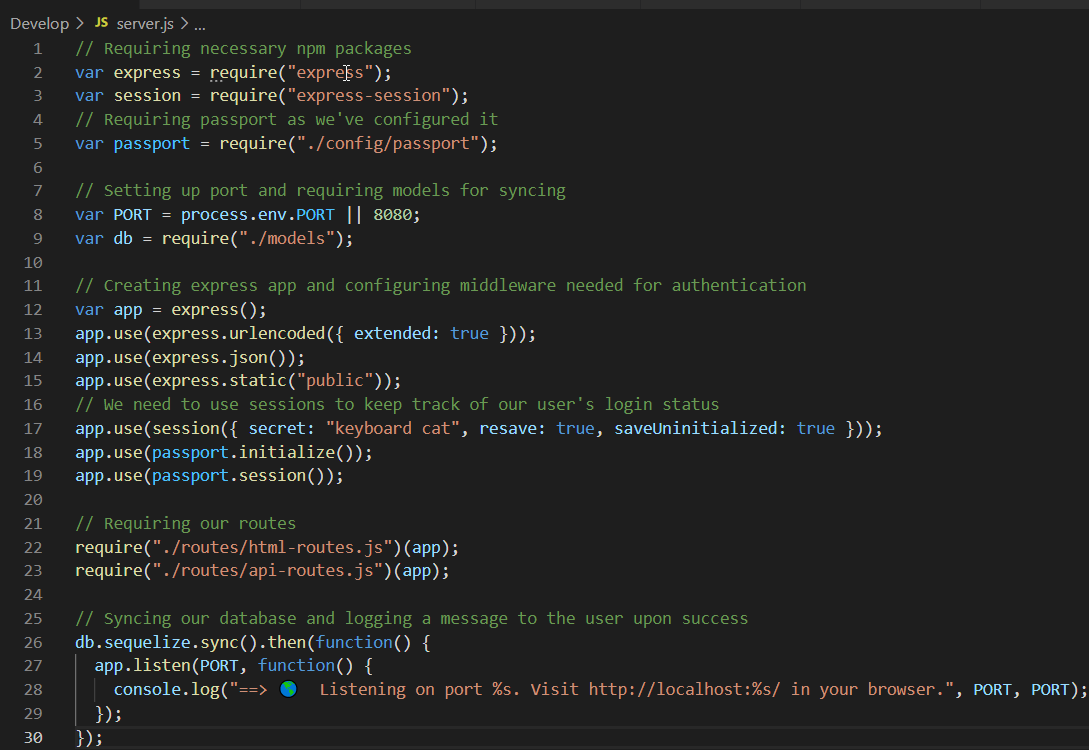


***File- passport\_demo.sql(+)***

******

***File- server.js***

(Requires packages, sets up PORT, creates express and middleware, creates routes and syncs database/ logs message in terminal on successful connection to server.)

******

**Testing**

***Where to test***

1. ***Console.log(connection)***
2. ***Browser( HTML, routing)***
3. ***Database(routing)***

**How to Improve Codebase**

***Describes what could be done right away and how to improve code for next time***

To vars to const/let when necessary.

Set up a notification when the user tries to sign up with an email that’s already in the database to prevent duplication.

Add warning when password fails minimum requirements.

Add a feature that lets the user edit their password or date joined to the members page.

Use AJAX to set API calls and functions.

Move isAuthenticated to html-routes file.

Display the integrated app onto a public website and demonstrate the features for exposure and mentoring people who don’t know as much. Ideas could be shopping/checkout center, wishlist access, or a forum.

Create github and heroku repos to host the new project and ALWAYS create a .gitignore file for node\_modules.

**Resources**

* **Griffin, Dalonte. Unit 14 Sequelize Homework: Reverse Engineering Code. (2020). Retrieved January 10, 2021 from** [**14-Full-Stack/02-Homework · master · michiganstate-bootcamp / MSU-DET-FSF-PT-09-2020-U-C · GitLab (bootcampcontent.com)**](https://michiganstate.bootcampcontent.com/michiganstate-bootcamp/msu-det-fsf-pt-09-2020-u-c/-/tree/master/14-Full-Stack/02-Homework)**.**
* **NPM. npm-package-lock.json | npm Documentation. (2020). Retrieved Jan 10, 2021, from** [**https://docs.npmjs.com/configuring-npm/package-lock-json.html#:~:text=Description,regardless%20of%20intermediate%20dependency%20updates.**](https://docs.npmjs.com/configuring-npm/package-lock-json.html#:~:text=Description,regardless%20of%20intermediate%20dependency%20updates.v)
* **Open JS Foundation.Path | Node.js v.14.8.0 Documentation. (2020). Retrieved Jan 10, 2021, from** [**https://nodejs.org/api/path.html**](https://nodejs.org/api/path.html)**.**
* **Open JS Foundation. What is the file ‘package.json’? | Node.js. (2020). Retrieved Jan 10, 2021, from** [**https://nodejs.org/en/knowledge/getting-started/npm/what-is-the-file-package-json/#:~:text=All%20npm%20packages%20contain%20a,as%20handle%20the%20project’s%20dependencies.&text=The%20package.,-json%20file%20is**](https://nodejs.org/en/knowledge/getting-started/npm/what-is-the-file-package-json/#:~:text=All%20npm%20packages%20contain%20a,as%20handle%20the%20project%E2%80%99s%20dependencies.&text=The%20package.,-json%20file%20is)**.**
* **W3 Schools. Node.js File System Module. (2020). Retrieved Jan 10, 2021, from** [**https://www.w3schools.com/nodejs/nodejs\_filesystem.as**](https://www.w3schools.com/nodejs/nodejs_filesystem.as)**.**