Reverse Enginering code

At the start of the Develop folder there’s a configuration folder which is mainly used to implement Passport within a project, passport is a package for javascript that allows the project to be authenticated meaning it enforces the database and the front end to only be accessed to the right user. As seen in the passport JS inside is a code enforcing the authentication.

First it uses the package and it uses Local strategy meaning that it wants a log in with a username/email and password. When a user tries to sign in the project would call the findOne function in the User database where email is equal to email , the authentication comes into play in the next line where it is stating that if there isn’t any email in the dbUser then a message of incorrect email would show up meaning that the email they enter is invalid. In the next step it would also check if the email matches the password if it is not then it will show a message of incorrect password. If none of the above is true then it returns the certain user from Db.User.

Inside the models folder this specify the structure of the backend database, which in this case would be the User database. It uses bycript package to hashed the password in the backend for security purposes. The addHook method is a method that will be triggered during the phases of the User model lifecycle which in this case mean before the User is created and it will automatically hash their password.

In the public folder is where all the html, front end js and and css is applied. Which then is connected to the backend through the routes. Inside the login, members, signup html it has all the front end structure of the design of the login page, with a stylesheets that details the design. In the js page is the front end javascript codes.

It is split into three file which are login, members and signup. In the Signup.js first it gets reference to all the form input that is used, then it uses a on eventlistener where it would get the value of email and password and saves it in an object called userData. Then if email and password exist then we would run the signUp user function which is a post request to the backend in respective of email and password. Inside the members.js is just getting all the user data from the backend. In the login.js it’s the main authentication js but in this case we are using the login/api route.