**Handling User Authentication**

**Github link:**

**Writeup:**

* Created a maven project as UserFeedback.
* Then add the dependencies to the pom.xml file and update the project by giving forceful update.
* Create the Application Properties file to connect to my database.
* Create files to serve as the homepage and added functionality to direct the user to the all users page, which shows all the users in the database.
* Then add a form for the user to input their name, email and password which adds them to the database.
* Create a controller class which handles mapping for the homepage, login page, and all users page.
* Create a user class and initialized it with the necessary components (id, name, email, password).
* Then add getters and setters for all these elements.
* Create a class called UserRepository which extends CrudRepository and allows me to make changes to the database.
* Then add a class called userService that used userRepository to find from the database.

**Testing:**

* Wrote 4 tests in the AuthenticationApplicationTests class: one that checks if the context loads, one that ensures that the number of users in the database is not 0, one that checks that the count in the database is the same as the expected value, and one that verifies that the users in the database have values not null.
* Wrote 4 tests in the AuthenticationTests class to verify that the user information was being added correctly. One method searches the userRepository for a created user and checks that its data is the same as what was entered during creation. The remaining methods ensure that the password, name, and email are all not null for any of the users.
* Wrote 3 tests in the AuthenticationWebTests class; one for each of the web pages. The first checks the homepage (“/”), the second checks the users page (“/allusers”), the third checks the login page for a 4xxClientError as it is not supposed to be accessed in this manner without the necessary login information.
* Wrote 6 tests in the EntityTests class to check all the getters and setters, as well as the constructors. The first 3 methods check the getters and setters for name, email, and password respectively. The next method checks the output of the toString method. The 5th method checks that the secondary constructor correctly initializes new users by cross referencing it with a user initialized using set methods. The last method tests the default constructor.