To implement a login system, you would typically follow these steps:

1. \*\*Create a Login Request Model:\*\* This model will hold the data that the user enters on the login page (username and password).

2. \*\*Create a Login Method in the Controller:\*\* This method will handle POST requests to a login endpoint (e.g., `/login`). It will take the login request model as a parameter.

3. \*\*Check the Database for the User:\*\* In the login method, use the username from the login request to check if a user with that username exists in the database.

4. \*\*Validate the Password:\*\* If the user exists, compare the password in the login request with the password stored in the database for that user. If they match, the login is successful. If they don't match, the login is unsuccessful.

5. \*\*Return a Response:\*\* Depending on whether the login was successful or not, return an appropriate response.

In this example, `LoginRequest` is a class that has two properties: `username` and `password`. `userRepository` is an instance of `UserRepository`, which is an interface that extends `JpaRepository<User, Long>`. `passwordEncoder` is an instance of `PasswordEncoder`, which is a Spring Security interface for encoding passwords.

Please note that this is a very basic example and a real-world application would require more complex handling, such as generating and returning a JWT for authenticated users, handling roles and permissions, etc.