

Authentication(Source Code)

src/main/java

com.project.Authentication(Package)

AuthenticationApplication.java(Class)

```
package com.project.Authentication;
```

```
import org.springframework.boot.SpringApplication;
```

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
import org.springframework.context.annotation.Import;
```

```
import
```

```
com.project.Authentication.controllers.AuthenticationController;
```

```
import com.project.Authentication.entities.User;
```

```
import com.project.Authentication.exceptions.UserNotFoundException;
```

```
import com.project.Authentication.services.AuthenticationService;
```

```
@SpringBootApplication
```

```
@Import({
```

```
    AuthenticationController.class,
```

```
        UserNotFoundException.class,  
        AuthenticationService.class,  
        User.class  
    })  
  
    public class AuthenticationApplication {  
  
        public static void main(String[] args) {  
            SpringApplication.run(AuthenticationApplication.class, args);  
        }  
  
    }
```

com.project.Authentication.controllers(Package)

AuthenticationController.java(Class)

```
package com.project.Authentication.controllers;
```

```
import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
```

```
import com.project.Authentication.entities.User;
import com.project.Authentication.services.AuthenticationService;
```

```
@Controller
```

```
public class AuthenticationController {
```

```
    Logger logger =
    LoggerFactory.getLogger(AuthenticationController.class);
```

```
    @Autowired
```

```
    AuthenticationService authService;
```

```
    @GetMapping("/")
```

```
    public String showGreeting() {
```

```
        return "greeting";
```

```
    }
```

```
@GetMapping("/Auth")  
  
public String showLogin() {  
    return "authenticate";  
}
```

```
@PostMapping("/Auth")  
  
public String authenticateUser(@RequestParam("username")  
String username, @RequestParam("password") String pswd) {  
    User user = authService.getUserByName(username);  
  
    logger.info(user.getName() + " attempted to login with " +  
user.getPassword());  
  
    String path = (authService.isValidPassword(pswd,  
user.getPassword())) ? "success" : "failure";  
  
    logger.info("The path return: " + path);  
  
    return path;  
}  
}
```

com.project.Authentication.entities(Package)

User.java(Class)

```
package com.project.Authentication.entities;
```

```
import javax.persistence.Column;
```

```
import javax.persistence.Entity;
```

```
import javax.persistence.GeneratedValue;
```

```
import javax.persistence.GenerationType;
```

```
import javax.persistence.Id;
```

```
import javax.persistence.Table;
```

```
import javax.validation.constraints.NotNull;
```

```
@Entity
```

```
@Table(name = "user")
```

```
public class User {
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.AUTO)
```

```
    @NotNull
```

```
    private Integer id;
```

```
@Column(name = "name")
```

```
@NotNull
```

```
private String name;
```

```
@Column(name = "email")
```

```
@NotNull
```

```
private String email;
```

```
@Column(name = "password")
```

```
@NotNull
```

```
private String password;
```

```
public User() {
```

```
    super();
```

```
}
```

```
public User(@NotNull String name, @NotNull String password) {
```

```
    this.name = name;
```

```
    this.password = password;
```

```
}
```

```
    public User(@NotNull String name, @NotNull String email,  
@NotNull String password) {
```

```
        super();
```

```
        this.name = name;
```

```
        this.email = email;
```

```
        this.password = password;
```

```
}
```

```
    public Integer getId() {
```

```
        return id;
```

```
}
```

```
    public void setId(Integer id) {
```

```
        this.id = id;
```

```
}
```

```
    public String getName() {
```

```
        return name;
```

```
}
```

```
public void setName(String name) {  
    this.name = name;  
}
```

```
public String getEmail() {  
    return email;  
}
```

```
public void setEmail(String email) {  
    this.email = email;  
}
```

```
public String getPassword() {  
    return password;  
}
```

```
public void setPassword(String password) {  
    this.password = password;  
}
```



```
@Override  
  
public String toString() {  
  
    return "User [id=" + id + ", name=" + name + ", email=" +  
email + ", password=" + password + "];"  
  
}  
  
}
```

com.project.Authentication.exceptions(Package)

UserNotFoundException.java(Class)

```
package com.project.Authentication.exceptions;  
  
public class UserNotFoundException extends RuntimeException {  
  
    private static final long serialVersionUID = 1L;  
  
}
```

com.project.Authentication.repositories(Package)

AuthenticationRepository.java(Class)

```
package com.project.Authentication.repositories;
```

```
import java.util.Optional;
```

```
import org.springframework.data.repository.CrudRepository;
```

```
import org.springframework.stereotype.Repository;
```

```
import com.project.Authentication.entities.User;
```

```
@Repository
```

```
public interface AuthenticationRepository extends  
CrudRepository<User, Integer> {
```

```
    public Optional<User> findUserByName(String name);  
}
```

```
com.project.Authentication.services(Package)
```

```
AuthenticationService.java(Class)
```

```
package com.project.Authentication.services;
```

```
import java.util.Optional;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.stereotype.Service;
```

```
import com.project.Authentication.entities.User;
```

```
import com.project.Authentication.exceptions.UserNotFoundException;
```

```
import
```

```
com.project.Authentication.repositories.AuthenticationRepository;
```

```
@Service
```

```
public class AuthenticationService {
```

```
    @Autowired
```

```
    AuthenticationRepository authRepo;
```

```
    public User GetUserByName(String name) {
```

```
        Optional<User> found = authRepo.findUserByName(name);
```

```
        if(found.isPresent()) return found.get();
```

```
        else throw new UserNotFoundException();
```

```
}
```

```
public Boolean isValidPassword(String cmp, String actual) {  
    return ((cmp.equals(actual)) ? true : false);  
}
```

```
}
```

src/test/java

com.project.Authentication(Package)

AuthenticationApplicationTests.java(Class)

package com.project.Authentication;

import org.junit.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

```
public class AuthenticationApplicationTests {
```

```
    @Test
```

```
    public void contextLoads() {
```

```
    }
```

```
}
```

```
AuthenticationCodeTests.java(Class)
```

```
package com.project.Authentication;
```

```
import static org.junit.jupiter.api.Assertions.assertEquals;
```

```
import static org.junit.jupiter.api.Assertions.assertFalse;
```

```
import static org.junit.jupiter.api.Assertions.assertTrue;
```

```
import java.util.Optional;
```

```
import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

import org.springframework.beans.factory.annotation.Autowired;

import
org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;

import
org.springframework.boot.test.autoconfigure.orm.jpa.TestEntityManag
er;


import com.project.Authentication.entities.User;

import com.project.Authentication.exceptions.UserNotFoundException;

import
com.project.Authentication.repositories.AuthenticationRepository;

import com.project.Authentication.services.AuthenticationService;


@DataJpaTest

public class AuthenticationCodeTests {

    @Autowired

    private TestEntityManager entityManager;


    @Autowired
```

```
private AuthenticationService authService;
```

```
@Autowired
```

```
private AuthenticationRepository authRepo;
```

```
private User testUser;
```

```
@BeforeEach
```

```
private void Setup() {
```

```
    testUser = new User("dummy", "dummy@testdummy.edu",  
"TestDummy4Life");
```

```
    System.out.println(testUser.toString());
```

```
    entityManager.persist(testUser);
```

```
    entityManager.flush();
```

```
}
```

```
@Test
```

```
public void shouldGetUserByName() {
```

```
    User test = authService.GetUserByName("dummy");
```

```
    assertEquals(testUser.getName(), test.getName());
```

```
}
```

@Test

```
public void shouldFindUserByName() throws
UserNotFoundException {

    Optional<User> temp =
authRepo.findUserByName("dummy");

    User tempUser = (temp.isPresent()) ? temp.get() : new
User();

    assertEquals(testUser.getName(), tempUser.getName());

    tempUser = new User();

    assertFalse(testUser.getName().equals(tempUser.getName()));
}
```

@Test

```
public void shouldValidateUser() {

    // incorrect username

    User input = new User("dumbo", "BigEars");

    Optional<User> temp =
authRepo.findUserByName(input.getName());

    User tempUser = (temp.isPresent()) ? temp.get() : new
User();
```



```
        assertFalse(testUser.getName().equals(input.getName()));

        // incorrect password but correct username
        input.setName("dummy");

        temp = authRepo.findUserByName(input.getName());
        tempUser = (temp.isPresent()) ? temp.get() : new User();

        assertFalse(authService.isValidPassword(tempUser.getPassword(),
input.getPassword()));

        //correct username and password
        input.setPassword("TestDummy4Life");

        assertTrue(authService.isValidPassword(tempUser.getPassword(),
input.getPassword()));
    }

}
```

AuthenticationWebTesting.java(Class)

```
package com.project.Authentication;
```

```
import static
```

```
org.springframework.test.web.servlet.request.MockMvcRequestBuilder  
s.get;
```

```
import static
```

```
org.springframework.test.web.servlet.result.MockMvcResultHandlers.p  
rint;
```

```
import static
```

```
org.springframework.test.web.servlet.result.MockMvcResultMatchers.s  
tatus;
```

```
import org.junit.jupiter.api.Test;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import
```

```
org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigur  
eMockMvc;
```

```
import org.springframework.boot.test.context.SpringBootTest;
```

```
import org.springframework.boot.web.server.LocalServerPort;
```

```
import org.springframework.test.web.servlet.MockMvc;
```

```
@SpringBootTest(webEnvironment =  
SpringBootTest.WebEnvironment.RANDOM_PORT)
```

@AutoConfigureMockMvc

public class AuthenticationWebTesting {

 @Autowired

 private MockMvc mockMvc;

 @LocalServerPort

 private int port;

 @Test

 public void shouldGetDefaultMessageFromGreetings() throws
Exception{

 this.mockMvc.perform(get("/")).andDo(print()).andExpect(status()
isOk());

 }

 @Test

 public void shouldGetDefaultMessageFromAuthenticate() throws
Exception {

 this.mockMvc.perform(get("/Auth")).andDo(print()).andExpect(sta

```
tus().isOk());
```

```
}
```

```
}
```

jsp

authenticate.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="ISO-8859-1">
```

```
<title>Authentication Page</title>
```

```
</head>
```

```
<h2>Login Page</h2>
```

```
<body>
```

```
welcome to the authentication page
```

```
<form:form action="Auth" method="post" commandName="Login">

    <label for="username"> Username:</label>

    <input name="username" id="username" type="text" placeholder="Username"
required/>

    <label for="password">Password:</label>

    <input name="password" id="password" type="password" placholder="Password"
required/>

    <input type="submit" name="Submit"/>

</form:form>

</body>

</html>
```

failure.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Failed Login</title>

</head>

<body>
```

```
<h1>You failed your login pal!  
  
</h1><br/>  
<a href="/Auth">Attempt Login again</a>  
  
</body>  
  
</html>
```

greeting.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
    pageEncoding="ISO-8859-1"%>  
  
<!DOCTYPE html>  
  
<html>  
  
<head>  
  
<meta charset="ISO-8859-1">  
  
<title>Landing Page</title>  
  
</head>  
  
<h2>Welcome Page</h2>  
  
<body>  
  
you reached the landing page  
  
<a href="Auth">Login</a>  
  
</body>  
  
</html>
```

success.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Successful Login Page</title>

</head>

<body>

<h1>Successful Login</h1>

</body>

</html>
```

