CODE

HandlingUserAuthenticationApplication

```
package com.example.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class HandlingUserAuthenticationApplication {

   public static void main(String[] args) {

      SpringApplication.run(HandlingUserAuthenticationApplication.class, args);

   }

}
```

SpringInitializer

```
package com.example.demo;

import org.springframework.boot.builder.SpringApplicationBuilder;

import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;

public class ServletInitializer extends SpringBootServletInitializer {

        @Override

        protected SpringApplicationBuilder configure(SpringApplicationBuilder
application) {

                return application.sources(HandlingUserAuthenticationApplication.class);

        }


}
```

User

```
package com.example.demo;
public class User {
    private String email;
    private String password;
    public User() {
    }
    public User(String email, String password) {
        this.email = email;
        this.password = password;
    }
    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 [email=" + email + ", password=" + password + "]";
    }
```

```
}
```

UserAuthentication

```
package com.example.demo;

import java.util.ArrayList;

import java.util.List;

public class UserAuthentication {

    private List<User> users = new ArrayList<>();

    public boolean login(User user) {

        return users.stream()

            .anyMatch(u -> u.getEmail().equals(user.getEmail()) &&
u.getPassword().equals(user.getPassword()));

    }

    public List<User> listOfUsers() {

        return users;

    }

    public String register(User user) {

        if (users.stream().anyMatch(u -> u.getEmail().equals(user.getEmail()))) {

            return "Email already exists";

        } else {

            users.add(user);

            return "Registered";

        }

    }

}
```

UserAuthenticationTest

```
package com.example.demo;
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertTrue;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import com.example.demo.*;
class UserAuthenticationTest {
    static UserAuthentication ua;
    @BeforeAll
    static void setUpBeforeClass() throws Exception {
        ua = new UserAuthentication();
        ua.register(new User("demo@example.com", "demo@123"));
    }
    @AfterAll
    static void tearDownAfterClass() throws Exception {
        ua = null;
    }
    @Test
    void testLogin() {
        User user = new User("demo@example.com", "demo@123");
        assertTrue(ua.login(user));
    }
    @Test
    void testListOfUsers() {
        assertEquals(1, ua.listOfUsers().size());
```

```
    }

    @Test
    void testRegistration() {
        User user = new User("test@example.com", "test@123");
        String result = ua.register(user);
        assertEquals("Registered", result);
    }
}
```

Pom

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <parent>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-parent</artifactId>
                <version>2.7.17</version>
                <relativePath/> <!-- lookup parent from repository -->
        </parent>
        <groupId>com.example</groupId>
        <artifactId>handling-user-authentication</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>war</packaging>
        <name>handling-user-authentication</name>
        <description>handling-user-authentication</description>
```

```
<properties>
	<java.version>17</java.version>
</properties>
<dependencies>
	<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-data-jpa</artifactId>
	</dependency>
	<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-web</artifactId>
	</dependency>

	<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-devtools</artifactId>
		<scope>runtime</scope>
		<optional>true</optional>
	</dependency>
	<dependency>
		<groupId>com.mysql</groupId>
		<artifactId>mysql-connector-j</artifactId>
		<scope>runtime</scope>
	</dependency>
	<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-tomcat</artifactId>
```

```
                <scope>provided</scope>
            </dependency>
            <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-test</artifactId>
                <scope>test</scope>
            </dependency>
            <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.33</version>
            </dependency>
        </dependencies>

        <build>
            <plugins>
                <plugin>
                    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-maven-plugin</artifactId>
                    <configuration>
                        <image>
                            <builder>paketobuildpacks/builder-jammy-
base:latest</builder>
                        </image>
                    </configuration>
                </plugin>
            </plugins>
        </build>
```

</project>