**Handling user Authentication**

**Project objective:**

Set up a standalone project to do unit testing of the user authentication class which is used in the main web application. The objective is to create a JUnit class that will test all aspects of the authentication class.

**Background of the problem statement:**

As a part of developing an ecommerce web application, a test-suite is being created to do unit testing of all backend components in the web application. This project will test the user authentication class. This project will be a standalone Java application, since Junit does not directly test servlets or web pages. We are only testing the classes that have the business logic.

**Code:**

package com.example;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table

public class user {

@Id

private String username;

private String password;

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

package com.example;

import java.util.List;

import org.springframework.orm.hibernate3.HibernateTemplate;

public class userdao {

private HibernateTemplate temp;

public void setTemp(HibernateTemplate temp) {

this.temp = temp;

}

//insert

public int insert(user u) {

return (Integer) temp.save(u);

}

//retrive

public List<user> getalluser(){

String sql="from user";

return (List<user>) temp.find(sql);

}

//delete

public int delete(user u) {

temp.delete(u);

return 1;

}

//update

public int update(user u) {

temp.update(u);

return 1;

}

}

package com.example;

import java.util.List;

import java.util.Scanner;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class userAuthentication {

public static String auth(String un,String up) {

ApplicationContext ac=new ClassPathXmlApplicationContext("spring.xml");

user u1=ac.getBean(user.class);

userdao dao1=ac.getBean(userdao.class);

int i=0;

String res = null;

List<user> uu=dao1.getalluser();

for(user usr:uu) {

if(usr.getUsername().equals(un) && usr.getPassword().equals(up)) {

i+=1;

res= "Authentication succesful";

}

}

if (i==0) {

res= "check credentials";

}

return res;

}

public static void main( String[] args ) {

ApplicationContext ac=new ClassPathXmlApplicationContext("spring.xml");

user u=ac.getBean(user.class);

userdao dao=ac.getBean(userdao.class);

Scanner sc=new Scanner(System.in);

System.out.println("enter the username");

String un=sc.next();

u.setUsername(un);

System.out.println("enter the password");

String up=sc.next();

u.setPassword(up);

auth(un,up);

}

}

package com.example;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

@Configuration

@ComponentScan(basePackages = {"com.example"})

public class config {

}

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns = *"http://www.springframework.org/schema/beans"*

xmlns:xsi = *"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation = *"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd"*>

<!-- pojo -->

<bean id=*"employee"* class=*"com.example.user"*></bean>

<!-- dao -->

<bean id=*"dao"* class=*"com.example.userdao"*>

<property name=*"temp"* ref=*"tempref"*></property>

</bean>

<!-- HibernateTemplate with session factory-->

<bean id=*"tempref"* class=*"org.springframework.orm.hibernate3.HibernateTemplate"*>

<property name=*"sessionFactory"* ref=*"sefref"*></property>

</bean>

<!-- session factory with the datasource -->

<bean id=*"sefref"* class=*"org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean"*>

<property name=*"dataSource"* ref=*"ds"*></property>

<property name=*"packagesToScan"* value=*"com.example"*></property>

<property name=*"hibernateProperties"*>

<props>

<prop key=*"hibernate.dialect"*> org.hibernate.dialect.MySQLDialect</prop>

<prop key=*"hibernate.hbm2ddl.auto"*>update</prop>

<prop key=*"hibernate.show\_sql"*>true</prop>

</props>

</property>

</bean>

<bean id=*"ds"* class=*"org.apache.commons.dbcp.BasicDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*></property>

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/db5"*></property>

<property name=*"username"* value=*"root"*></property>

<property name=*"password"* value=*"root"*></property>

</bean>

</beans>

<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>

<groupId>109</groupId>

<artifactId>HandlingUserAuthentication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-orm -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-tx -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>3.6.9.Final</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-entitymanager -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-entitymanager</artifactId>

<version>3.6.9.Final</version>

</dependency>

<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.19</version>

</dependency>

<!-- https://mvnrepository.com/artifact/commons-dbcp/commons-dbcp -->

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.4</version>

</dependency>

</dependencies>

</project>

import static org.junit.Assert.\*;

import org.junit.Test;

import com.example.user;

import com.example.userAuthentication;

public class authtest {

@Test

public void authentication(){

userAuthentication ua=new userAuthentication();

String result = userAuthentication.auth("ravi091","123ravi");

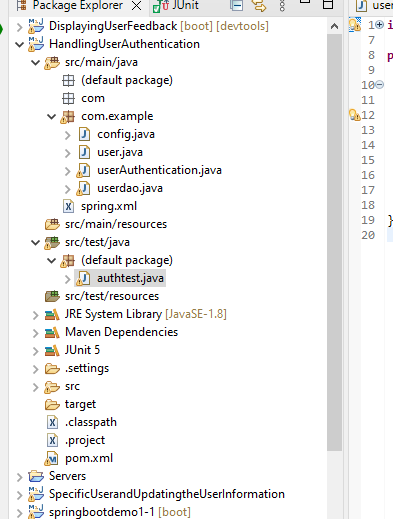
System.out.println(result);

assertEquals("Authentication succesful", result);

}

}

**Create a standalone Java application using Maven**



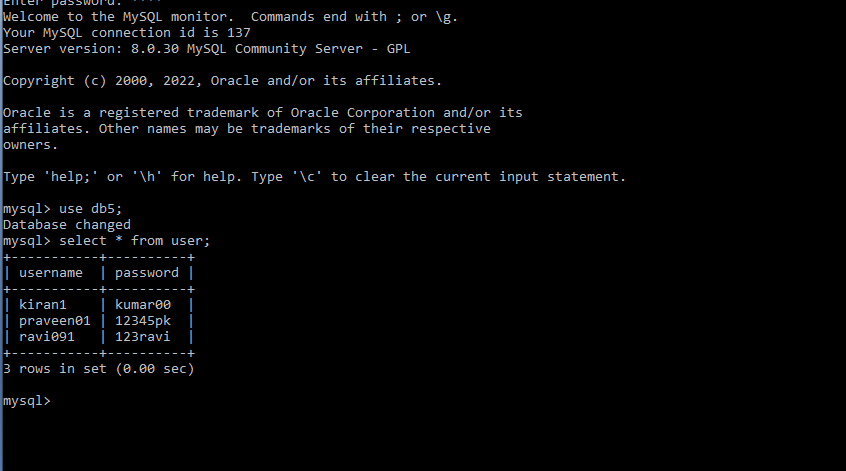
**Create an authentication class that has all the methods related to user authentication**



**Create a JUnit test class to create unit tests for the authentication class**

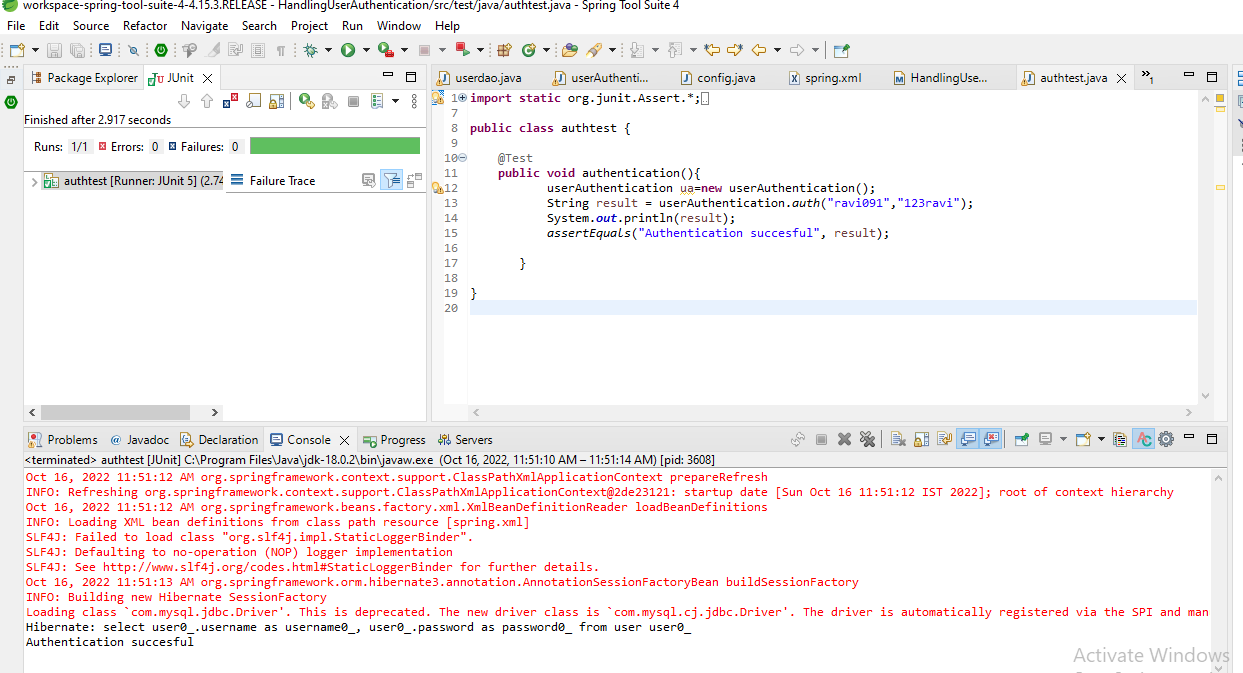


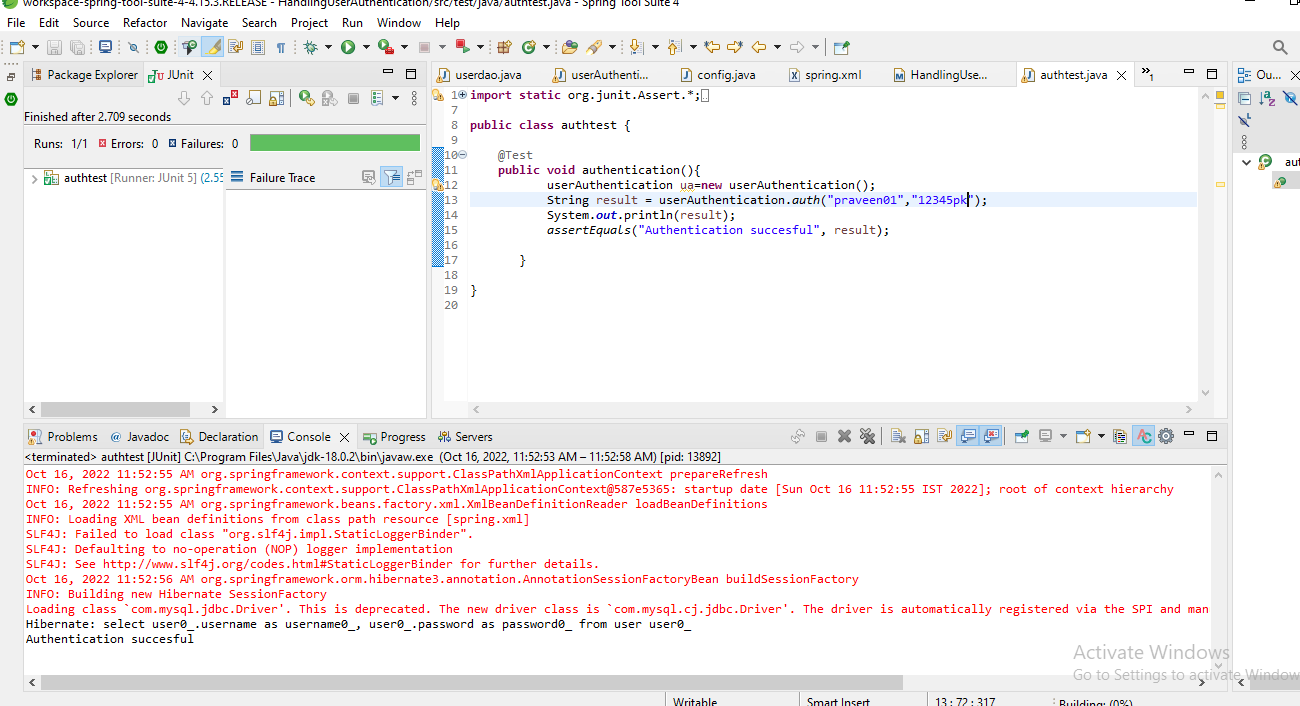
**Run the test class directly as a JUnit and check if all the tests pass**



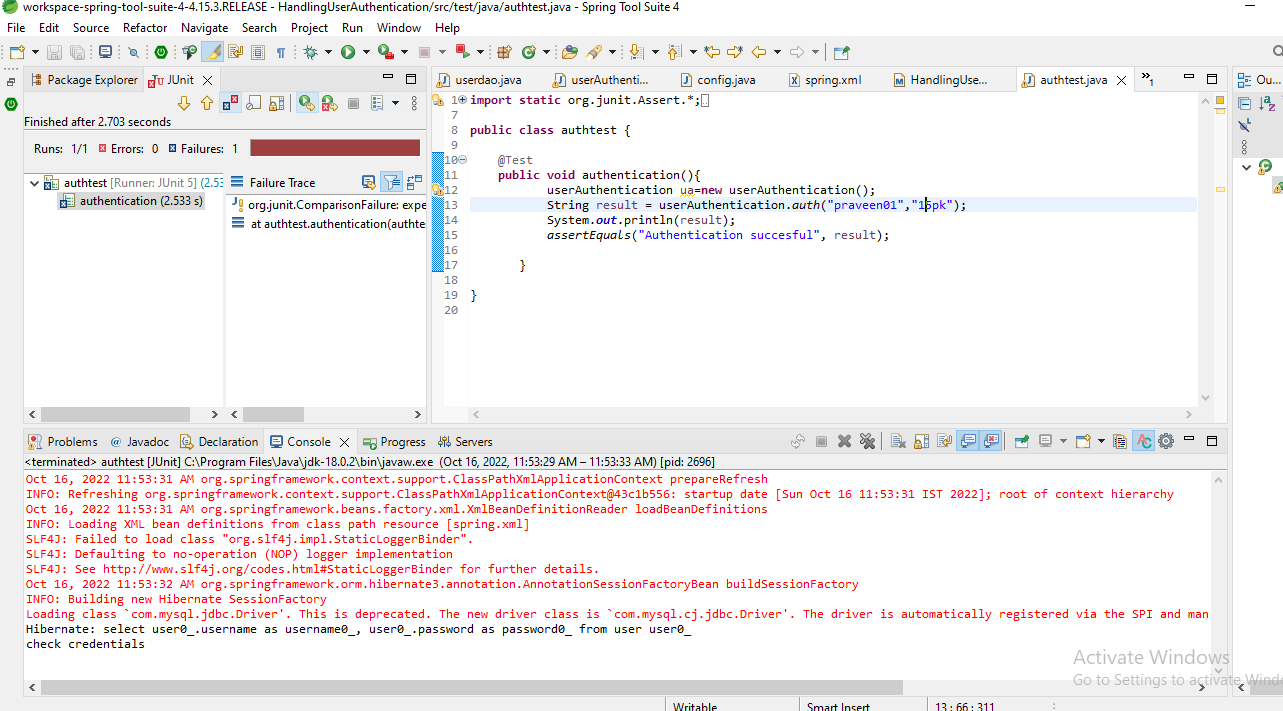
**Try to check this username and password from database in junit test case**

**Trying with right password**



****

**Trying a wrong password**



**Git**