**MavenWrapperDownloader.java**

**import** java.net.\*;

**import** java.io.\*;

**import** java.nio.channels.\*;

**import** java.util.Properties;

**public** **class** MavenWrapperDownloader {

**private** **static** **final** String ***WRAPPER\_VERSION*** = "0.5.6";

**private** **static** **final** String ***DEFAULT\_DOWNLOAD\_URL*** = "https://repo.maven.apache.org/maven2/io/takari/maven-wrapper/"

+ ***WRAPPER\_VERSION*** + "/maven-wrapper-" + ***WRAPPER\_VERSION*** + ".jar";

**private** **static** **final** String ***MAVEN\_WRAPPER\_PROPERTIES\_PATH*** =

".mvn/wrapper/maven-wrapper.properties";

**private** **static** **final** String ***MAVEN\_WRAPPER\_JAR\_PATH*** =

".mvn/wrapper/maven-wrapper.jar";

**private** **static** **final** String ***PROPERTY\_NAME\_WRAPPER\_URL*** = "wrapperUrl";

**public** **static** **void** main(String args[]) {

System.***out***.println("- Downloader started");

File baseDirectory = **new** File(args[0]);

System.***out***.println("- Using base directory: " + baseDirectory.getAbsolutePath());

File mavenWrapperPropertyFile = **new** File(baseDirectory, ***MAVEN\_WRAPPER\_PROPERTIES\_PATH***);

String url = ***DEFAULT\_DOWNLOAD\_URL***;

**if**(mavenWrapperPropertyFile.exists()) {

FileInputStream mavenWrapperPropertyFileInputStream = **null**;

**try** {

mavenWrapperPropertyFileInputStream = **new** FileInputStream(mavenWrapperPropertyFile);

Properties mavenWrapperProperties = **new** Properties();

mavenWrapperProperties.load(mavenWrapperPropertyFileInputStream);

url = mavenWrapperProperties.getProperty(***PROPERTY\_NAME\_WRAPPER\_URL***, url);

} **catch** (IOException e) {

System.***out***.println("- ERROR loading '" + ***MAVEN\_WRAPPER\_PROPERTIES\_PATH*** + "'");

} **finally** {

**try** {

**if**(mavenWrapperPropertyFileInputStream != **null**) {

mavenWrapperPropertyFileInputStream.close();

}

} **catch** (IOException e) {

}

}

}

System.***out***.println("- Downloading from: " + url);

File outputFile = **new** File(baseDirectory.getAbsolutePath(), ***MAVEN\_WRAPPER\_JAR\_PATH***);

**if**(!outputFile.getParentFile().exists()) {

**if**(!outputFile.getParentFile().mkdirs()) {

System.***out***.println(

"- ERROR creating output directory '" + outputFile.getParentFile().getAbsolutePath() + "'");

}

}

System.***out***.println("- Downloading to: " + outputFile.getAbsolutePath());

**try** {

*downloadFileFromURL*(url, outputFile);

System.***out***.println("Done");

System.*exit*(0);

} **catch** (Throwable e) {

System.***out***.println("- Error downloading");

e.printStackTrace();

System.*exit*(1);

}

}

**private** **static** **void** downloadFileFromURL(String urlString, File destination) **throws** Exception {

**if** (System.*getenv*("MVNW\_USERNAME") != **null** && System.*getenv*("MVNW\_PASSWORD") != **null**) {

String username = System.*getenv*("MVNW\_USERNAME");

**char**[] password = System.*getenv*("MVNW\_PASSWORD").toCharArray();

Authenticator.*setDefault*(**new** Authenticator() {

@Override

**protected** PasswordAuthentication getPasswordAuthentication() {

**return** **new** PasswordAuthentication(username, password);

}

});

}

URL website = **new** URL(urlString);

ReadableByteChannel rbc;

rbc = Channels.*newChannel*(website.openStream());

FileOutputStream fos = **new** FileOutputStream(destination);

fos.getChannel().transferFrom(rbc, 0, Long.***MAX\_VALUE***);

fos.close();

rbc.close();

}

}

**Maven-wrapper-properties:**

distributionUrl=https://repo.maven.apache.org/maven2/org/apache/maven/apache-maven/3.6.3/apache-maven-3.6.3-bin.zip

wrapperUrl=https://repo.maven.apache.org/maven2/io/takari/maven-wrapper/0.5.6/maven-wrapper-0.5.6.jar

**LoginController.java:**

**import** org.apache.catalina.User;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

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

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.ModelMap;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PostMapping;

**import** org.springframework.web.bind.annotation.RequestParam;

**import** org.springframework.web.bind.annotation.ResponseBody;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

**public** **class** LoginController {

@Autowired

UserRepository userRepository;

@GetMapping(value="/")

**public** String showIndexPage(ModelMap model){

**return** "<html>\n"

+ "<head>\n"

+ " <style>\n"

+ " .center {\n"

+ " text-align: center;\n"

+ " }\n"

+ " \n"

+ " </style>\n"

+ "</head>\n"

+ "<body style=\"background-color:lightblue;\">\n"

+ " <div class=\"center\">\n"

+ " <h1>User Login Page</h1>\n"

+ " \n"

+ " <h2 class=\"hello-title\">Welcome</h2>\n"

+ " \n"

+ " <a href=\"/allusers\">View all users</a>\n"

+ " <br><br>\n"

+ " <form method=\"get\" action=\"login\">\n"

+ " <br><h3>Login below:</h3>\n"

+ " <input type=\"text\" id=\"name\" name=\"name\" placeholder=\"Name\" required>\n"

+ " <input type=\"text\" id=\"email\" name=\"email\" placeholder=\"Email\" required>\n"

+ " <input type=\"text\" id=\"password\" name=\"password\" placeholder=\"Password\" required> \n"

+ " <input type=\"submit\" value=\"Enter\" />\n"

+ " </form>"

+ " </div>\n"

+ "</body>\n"

+ "</html>";

}

@GetMapping("/login")

**public** String showLogin(@RequestParam("name") String name, @RequestParam("email") String email, @RequestParam("password") String password, ModelMap map) {

User u = **new** User(name,email,password);

userRepository.save(u);

**return** "<html>\n"

+ "<head>\n"

+ " <style>\n"

+ " .center {\n"

+ " text-align: center;\n"

+ " }\n"

+ " \n"

+ " </style>\n"

+ "</head>\n"

+ "<body style=\"background-color:lightblue;\">\n"

+ " <div class=\"center\">\n"

+ " <h1>Logged In</h1>\n"

+ " \n"

+ " <h2 class=\"hello-title\">Successfully Added Your Information</h2>\n"

+ " </div>\n"

+ "</body>\n"

+ "</html>";

}

@GetMapping("/allusers")

**public** @ResponseBody String getAllFeedbacks() {

Iterable<User> allUser = userRepository.findAll();

**return** "<html>\n"

+ "<head>\n"

+ " <style>\n"

+ " .center {\n"

+ " text-align: center;\n"

+ " }\n"

+ " \n"

+ " </style>\n"

+ "</head>\n"

+ "<body style=\"background-color:lightblue;\">\n"

+ " <div class=\"center\">\n"

+ "<h1>Feedback Table</h1>\n"

+ allUser.toString()

+ " </div>\n"

+ "</body>\n"

+ "</html>";

}

@PostMapping("/login")

**public** String submitLogin(@RequestParam String username, @RequestParam String password){

//**TODO**:

**return** "Success";

}

}

**User.java :**

**package** entity;

**import** jakarta.annotation.Generated;

@Entity

**public** **class** User {

@Id

@Generated(value=Generated.IDENTITY)

**private** Integer id;

**private** String name;

**private** String email;

**private** String password;

**public** User()

{

**public** User(String name, String email, String password) {

**this**.name = name;

**this**.email = email;

**this**.password = password;

}

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**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;

}

@Override

**public** String toString() {

**return** "<br><h3>" + name + " [" + id + "]:" + "</h3><h4>email: " + email + "</h4><h4>password: " + password + "</h4><br>";

}

}

**UserNotFoundException.java :**

**package** exceptions;

**public** **class** UserNotFoundException **extends** RuntimeException {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

}

**userRepository.java :**

**package** repository;

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

**import** com.example.Authentication.entities.User;

**public** **interface** UserRepository **extends** CrudRepository<User, Integer> {

**public** User findByName(String name);

}

**UserService.java :**

**package** services;

**import** java.util.Optional;

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

**import** org.springframework.stereotype.Service;

**import** com.example.Authentication.entities.User;

@Service

**public** **class** UserService {

@Autowired

**private** UserRepository userRepository;

**public** Iterable<User> GetAllUsers()

{

**return** userRepository.findAll();

}

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

User foundUser = userRepository.findByName(name);

**return** foundUser;

}

**public** User GetUserById(**int** id) **throws** services.UserNotFoundException {

Optional<User> foundUser = userRepository.findById(id);

//**TODO**:

// handle a "Not Found"

**if** (!foundUser.isPresent()) {

**throw** **new** UserNotFoundException();

}

**return**(foundUser.get());

}

**public** **void** UpdateUser(User usertoUpdate) {

userRepository.save(usertoUpdate);

}

}

**AuthenticationApplication.java :**

package services;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class AuthenticationApplication {

public static void main(String[] args) {

SpringApplication.run(AuthenticationApplication.class, args);

}

}

**Application.properties:**

spring.jpa.hibernate.ddl-auto=update

spring.datasource.url=jdbc:mysql://${MYSQL\_HOST:localhost}:3306

spring.datasource.username=root

spring.datasource.password=root

logging.level.org.springframework.web: DEBUG

server.port=8090

server.error.whitelabel.enabled=false

**greetings.jsp**

<html>

<body>

<h2>Spring Application</h2>

</body>

</html>

**error.jsp**

<html>

<body>

<h2>Page not found</h2>

</body>

</html>

**AuthenticationApplicationTests.java :**

**package** com.authentication;

**import** org.junit.Test;

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

**import** com.example.Authentication.entities.User;

**import** services.UserService;

@SpringBootTest

**class** AuthenticationApplicationTests {

@Autowired

**private** UserService userService;

@Test

**void** contextLoads() {

}

@Test

**void** testServiceCall() {

Iterable<User> users = userService.GetAllUsers();

Integer count = 0;

**for**(User u: users)

count++;

assertNotEquals(count, 0);

}

**private** **void** assertNotEquals(Integer count, **int** i) {

// **TODO** Auto-generated method stub

}

@Test

**void** countUsers() {

Iterable<User> users = userService.GetAllUsers();

Integer count = 0;

**for**(User u: users)

count++;

assertEquals(count, 4);

}

**private** **void** assertEquals(Integer count, **int** i) {

// **TODO** Auto-generated method stub

}

@Test

**void** checkAllUsers() {

Iterable<User> users = userService.GetAllUsers();

**for**(User u: users)

assertNotNull(u);

}

**private** **void** assertNotNull(User u) {

// **TODO** Auto-generated method stub

}

}

**AuthenticationTests.java :**

**package** com.authentication;

**import** **static** org.junit.Assert.*assertEquals*;

**import** **static** org.junit.Assert.*assertNotNull*;

**import** org.junit.Test;

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

**import** com.example.Authentication.entities.User;

**import** repository.UserRepository;

@DataJpaTest

**public** **class** AuthenticationTests {

@Autowired

**private** TestEntityManager entityManager;

@Autowired

**private** UserRepository userRepository;

@Test

**public** **void** returnUserFromName() {

User testUser = **new** User();

testUser.setName("newTest");

testUser.setEmail("test@email.com");

testUser.setPassword("testpw");

entityManager.persist(testUser);

entityManager.flush();

User found = userRepository.findByName(testUser.getName());

*assertEquals*(found.getName(), testUser.getName());

}

@Test

**public** **void** passwordNotNull() {

Iterable<User> users = userRepository.findAll();

**for**(User u: users)

*assertNotNull*(u.getPassword());

}

@Test

**public** **void** nameNotNull() {

Iterable<User> users = userRepository.findAll();

**for**(User u: users)

*assertNotNull*(u.getName());

}

@Test

**public** **void** emailNotNull() {

Iterable<User> users = userRepository.findAll();

**for**(User u: users)

*assertNotNull*(u.getEmail());

}

}**EntityTests.java :**

**package** com.authentication;

**import** **static** org.junit.Assert.*assertEquals*;

**import** **static** org.junit.Assert.*assertNotNull*;

**import** org.junit.Test;

**import** com.example.Authentication.entities.User;

**public** **class** EntityTests {

@Test

**public** **void** getAndSetPassword() {

User testUser = **new** User();

testUser.setPassword("mypassword");

*assertEquals*(testUser.getPassword(),"mypassword");

}

@Test

**public** **void** getAndSetName() {

User testUser = **new** User();

testUser.setName("joe");

*assertEquals*(testUser.getName(),"joe");

}

@Test

**public** **void** getAndSetEmail() {

User testUser = **new** User();

testUser.setEmail("joe@email.com");

*assertEquals*(testUser.getEmail(),"joe@email.com");

}

@Test

**public** **void** checkToString() {

User testUser = **new** User();

*assertNotNull*(testUser.toString());

}

@Test

**public** **void** checkConstructor() {

User testUser = **new** User("joe","joe@email.com","123");

User checkUser = **new** User();

checkUser.setName("joe");

checkUser.setEmail("joe@email.com");

checkUser.setPassword("123");

*assertEquals*(testUser.getName(), checkUser.getName());

*assertEquals*(testUser.getEmail(), checkUser.getEmail());

*assertEquals*(testUser.getPassword(), checkUser.getPassword());

}

@Test

**public** **void** testDefaultConstructor() {

User testUser = **new** User();

*assertNotNull*(testUser);

}

}

Pom.xml:

<?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>2.4.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.example</groupId>

<artifactId>Authentication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Authentication</name>

<description>Test</description>

<properties>

<java.version>1.8</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>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

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

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<configuration>

<excludes>

<exclude>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

</exclude>

</excludes>

</configuration>

</plugin>

</plugins>

</build>

</project>