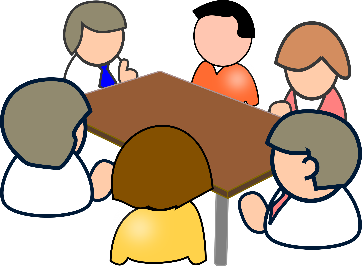
National University of Computer and Emerging Sciences Chiniot-Faisalabad Campus



**META-BASE**

Software Quality Engineering

### Semester Project

### Phase 1ST

### BS (SE) 5A

**API AUTOMATED TEST CASES**

import io.restassured.RestAssured;

import io.restassured.response.Response;

import org.junit.jupiter.api.Test;

import static io.restassured.RestAssured.given;

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

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

public class ApiTests {

private static final String BASE\_URL = "your\_base\_url";

@Test

public void testRetrieveDashboardData() {

RestAssured.baseURI = BASE\_URL;

Response response = given()

.when()

.get("/dashboard");

assertEquals(200, response.getStatusCode());

assertTrue(response.body().asString().contains("expected\_dashboard\_info"));

}

@Test

public void testExecuteQueryViaApi() {

RestAssured.baseURI = BASE\_URL;

Response response = given()

.when()

.post("/executeQuery");

assertEquals(200, response.getStatusCode());

assertTrue(response.body().asString().contains("expected\_query\_results"));

}

@Test

public void testFilterDataViaApi() {

RestAssured.baseURI = BASE\_URL;

Response response = given()

.param("filterParameter", "specific\_value")

.when()

.get("/filterData");

assertEquals(200, response.getStatusCode());

assertTrue(response.body().asString().contains("filtered\_data"));

}

@Test

public void testExportQueryResultsViaApi() {

RestAssured.baseURI = BASE\_URL;

Response response = given()

.when()

.get("/exportQueryResults");

assertEquals(200, response.getStatusCode());

assertTrue(response.body().asString().contains("downloadable\_link"));

}

@Test

public void testUserPermissionsViaApi() {

RestAssured.baseURI = BASE\_URL;

Response response = given()

.auth().basic("limited\_user", "password")

.when()

.get("/userPermissions");

assertEquals(403, response.getStatusCode());

}

}

**GUI AUTOMATED TEST CASES**

import org.junit.jupiter.api.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

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

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

public class WebUITests {

private static final String BASE\_URL = "your\_metabase\_url";

private static final String USERNAME = "your\_username";

private static final String PASSWORD = "your\_password";

@Test

public void testLoginFunctionality() {

WebDriver driver = new ChromeDriver();

driver.get(BASE\_URL);

WebElement usernameInput = driver.findElement(By.id("username"));

WebElement passwordInput = driver.findElement(By.id("password"));

WebElement loginButton = driver.findElement(By.id("loginButton"));

usernameInput.sendKeys(USERNAME);

passwordInput.sendKeys(PASSWORD);

loginButton.click();

String dashboardUrl = driver.getCurrentUrl();

assertTrue(dashboardUrl.contains("dashboard"));

driver.quit();

}

@Test

public void testInvalidLoginAttempt() {

WebDriver driver = new ChromeDriver();

driver.get(BASE\_URL);

WebElement usernameInput = driver.findElement(By.id("username"));

WebElement passwordInput = driver.findElement(By.id("password"));

WebElement loginButton = driver.findElement(By.id("loginButton"));

usernameInput.sendKeys("invalid\_username");

passwordInput.sendKeys("invalid\_password");

loginButton.click();

WebElement errorMessage = driver.findElement(By.id("errorMessage"));

assertTrue(errorMessage.isDisplayed());

driver.quit();

}

@Test

public void testDashboardNavigation() {

WebDriver driver = new ChromeDriver();

driver.get(BASE\_URL);

// Assume login functionality is already tested in a separate test

WebElement dashboardLink = driver.findElement(By.id("dashboardLink"));

dashboardLink.click();

String dashboardUrl = driver.getCurrentUrl();

assertTrue(dashboardUrl.contains("dashboard"));

driver.quit();

}

@Test

public void testQueryExecution() {

WebDriver driver = new ChromeDriver();

driver.get(BASE\_URL);

// Assume login functionality is already tested in a separate test

WebElement dashboardLink = driver.findElement(By.id("dashboardLink"));

dashboardLink.click();

WebElement queryInput = driver.findElement(By.id("queryInput"));

queryInput.sendKeys("SELECT \* FROM your\_table");

WebElement executeButton = driver.findElement(By.id("executeButton"));

executeButton.click();

WebElement queryResults = driver.findElement(By.id("queryResults"));

assertTrue(queryResults.isDisplayed());

driver.quit();

}

@Test

public void testQueryVisualization() {

WebDriver driver = new ChromeDriver();

driver.get(BASE\_URL);

// Assume login functionality is already tested in a separate test

WebElement dashboardLink = driver.findElement(By.id("dashboardLink"));

dashboardLink.click();

WebElement queryInput = driver.findElement(By.id("queryInput"));

queryInput.sendKeys("SELECT \* FROM your\_table");

WebElement executeButton = driver.findElement(By.id("executeButton"));

executeButton.click();

WebElement visualizeButton = driver.findElement(By.id("visualizeButton"));

visualizeButton.click();

WebElement chartVisualization = driver.findElement(By.id("chartVisualization"));

assertTrue(chartVisualization.isDisplayed());

driver.quit();

}

}