**Authentication and Cookies in Selenium**

This tutorial demonstrates how to handle API authentication in Selenium using cookies. We'll walk through logging into a website, retrieving cookies, and making authenticated API calls. We'll use the petstore swagger (<https://petstore.swagger.io/>) for practical examples.

**1. Introduction to API Authentication in Selenium**

API authentication ensures secure access to resources. When testing web applications, cookies stored after logging in can be used to authenticate API calls. Selenium enables interaction with web applications to log in and retrieve cookies for use in further testing or API requests.

**2. Using Cookies for API Calls**

**What Are Cookies?**

Cookies store session information after authentication. These cookies can be passed to REST API requests to simulate an authenticated user session.

**Workflow Overview:**

1. Use Selenium to log into the website.
2. Retrieve cookies from the browser session.
3. Use these cookies in a REST API request (via REST Assured).

**3. Handling API Authentication in Selenium**

**Steps for Authentication:**

1. **Log in to the petstore swagger** with a username and password (provide in the swagger).
2. **Retrieve the cookies** after successful login.
3. **Use the cookies** for authenticated requests in subsequent API calls.

**Example Setup:**

We'll use **Selenium WebDriver** to automate the login process and retrieve cookies. Then, we will make an API call using the retrieved cookies.

**Step 1: Setup WebDriver and Navigate to the Login Page**

The tutorial starts by setting up the Selenium WebDriver to navigate to the login page and perform a login action to get an authenticated session (with cookies).

**Step 2: Create a BaseTest class extends it with the rest of the classes.**

package base;

import io.restassured.RestAssured;

import org.openqa.selenium.Cookie;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeClass;

import java.util.Set;

public class BaseTest {

protected WebDriver driver;

protected Cookie sessionCookie;

@BeforeClass

public void setup() throws InterruptedException {

// Step 1: Set up WebDriver

System.setProperty("webdriver.chrome.driver", "src/main/resources/chromedriver.exe");

driver = new ChromeDriver(); // Create a new instance of ChromeDriver

driver.manage().window().maximize();

// Step 2: Navigate to the Swagger Petstore Login Page

driver.get("https://petstore.swagger.io/#/user/login");

// Step 3: Log in with username and password (passed as query parameters) and then minimize the window

String username = "username"; // Example username

String password = "password"; // Example password

driver.get("https://petstore.swagger.io/v2/user/login?username=" + username + "&password=" + password);

Thread.sleep(2000);

driver.manage().window().minimize();

// Step 4: Retrieve all the cookies after login

Set<Cookie> cookies = driver.manage().getCookies();

System.out.println("Retrieved Cookies:");

// Step 5: Print cookie(s) details including Name, Value, Domain, Path, and Expiry

for (Cookie cookie : cookies) {

System.out.println("Name: " + cookie.getName());

System.out.println("Value: " + cookie.getValue());

System.out.println("Domain: " + cookie.getDomain());

System.out.println("Path: " + cookie.getPath());

System.out.println("Expiry: " + cookie.getExpiry());

System.out.println("-----------------------------");

}

// Step 6: Use the desired cookie for authenticated API call

sessionCookie = driver.manage().getCookieNamed("userId");

RestAssured.baseURI = "https://petstore.swagger.io/v2";

//assert sessionCookie != null; // It checks whether the variable sessionCookie is not null.

}

@AfterClass

public void teardown() {

if (driver != null) {

driver.quit();

}

}

}

**Explanation for the Base Class:** The BaseTest class serves as the foundational class for setting up and managing the WebDriver instance and retrieving the session cookie for authenticated API calls. It is structured to handle the common setup and teardown tasks required for Selenium WebDriver-based tests.

**Step 3: Create Separate Classes for GET, POST, PUT, and DELETE Requests**

**1. GET Request Class:**

package tests;

import base.BaseTest;

import io.restassured.RestAssured;

import io.restassured.response.Response;

import org.testng.annotations.Test;

public class GetRequestAuth extends BaseTest {

@Test

public void getPetsByStatus() {

// GET Request - Retrieve pets by status

Response response = RestAssured.given()

.cookie(sessionCookie.getName(), sessionCookie.getValue())

.when()

.get("/pet/findByStatus?status=pending");

System.out.println("GET API Response: " + response.getBody().asString());

response.then().statusCode(200);

}

}

**Explanation for GET Request:** From the BaseTest class, we used Selenium WebDriver to log in to the Swagger Petstore API and retrieve the session cookie (userId). In this class, we then pass this cookie to the GET API request to retrieve pets by their status. The response is validated to ensure the status code is 200.

**2. POST Request Class:**

package tests;

import base.BaseTest;

import io.restassured.RestAssured;

import io.restassured.response.Response;

import org.json.JSONObject;

import org.testng.annotations.Test;

public class PostRequestAuth extends BaseTest {

@Test

public void postPetByStatus() {

// POST Request - Create a new pet

JSONObject postParams = new JSONObject();

postParams.put("id","12345");

postParams.put("name","snowy");

postParams.put("status","sold");

Response postResponse = RestAssured.given()

.cookie(sessionCookie.getName(), sessionCookie.getValue()) // Pass the cookie to the request

.contentType("application/json")

.body(postParams.toString())

.when()

.post("/pet"); // Example API call to add a new pet

System.out.println("POST API Response: " + postResponse.getBody().asString());

postResponse.then().statusCode(200); // Validate successful response

}

}

**Explanation for POST Request:** From the BaseTest class, we log in to the Swagger Petstore API and retrieve the session cookie. Then, in this class, we create a new pet using a POST request by passing the session cookie to authenticate the request. The response is checked for a 200 status code.

**3. PUT Request Class:**

package tests;

import base.BaseTest;

import io.restassured.RestAssured;

import io.restassured.response.Response;

import org.json.JSONObject;

import org.testng.annotations.Test;

public class PutRequestAuth extends BaseTest {

@Test

public void putPetByName() {

// PUT Request - Update pet details

JSONObject putParams = new JSONObject();

putParams.put("id","12345");

putParams.put("name","fluffy(updated from snowy)");

putParams.put("status","pending");

Response putResponse = RestAssured.given()

.cookie(sessionCookie.getName(), sessionCookie.getValue()) // Pass the cookie to the request

.contentType("application/json")

.body(putParams.toString())

.when()

.put("/pet"); // Example API call to update a pet

System.out.println("PUT API Response: " + putResponse.getBody().asString());

putResponse.then().statusCode(200); // Validate successful response

}

}

**Explanation for PUT Request:** Here, after logging in and retrieving the session cookie from the BaseTest class, we make a PUT request to update the details of a pet. The cookie is passed to authenticate the request, and we validate that the response code is 200.

**4. DELETE Request Class:**

package tests;

import base.BaseTest;

import io.restassured.RestAssured;

import io.restassured.response.Response;

import org.testng.annotations.Test;

public class DelRequestAuth extends BaseTest {

@Test

public void delPetByid() {

// DELETE Request - Delete a pet

Response deleteResponse = RestAssured.given()

.cookie(sessionCookie.getName(), sessionCookie.getValue()) // Pass the cookie to the request

.when()

.delete("/pet/12345"); // Example API call to delete a pet by ID

System.out.println("DELETE API Response: " + deleteResponse.getBody().asString());

deleteResponse.then().statusCode(404); // Validate successful response

}

}

**Explanation for DELETE Request:** In this class, we authenticate using the session cookie and make a DELETE request to remove a pet by its ID. The response is validated to ensure it returns a 404 status code.

**Conclusion:**

In this tutorial, we demonstrated how to authenticate via a login page using Selenium WebDriver, retrieve session cookies, and use those cookies to authenticate API requests (GET, POST, PUT, DELETE) using RestAssured. This technique is useful when dealing with applications that require authentication to access certain resources.