TestNG Framework:

TestNG is a testing framework for the Java programming language created by Cedric Beust and inspired by JUnit and NUnit. The design goal of TestNG is to cover a wider range of test categories: unit, functional, end-to-end, integration, etc., with more powerful and easy-to-use functionalities.

1.Run multiple Test or parallel testing:

Parallel testing is a process where multiple tests are executed simultaneously/in parallel in different thread processes. With respect to Selenium and TestNG, it allows you to execute multiple tests on different browsers, devices, environments in parallel and at the same time, instead of running it sequentially

Step:1 write the code in TestNG Framework (package).

Step:2 select any two or more Classes and then convert to TestNG

(Remove static for eg: **public** **void** pageload() ) before that add @Test

Step :3 It will be stored in particular location and displayed in the left column

Step:4 While converting, it gives as TestNG XML file.

Step:5 Give run as TestNG Suite

Source code XML

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

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name=*"Suite"*>

<test name=*"Test"*>

<classes>

<class name=*"testNGFramework.pinespherefunc"*/>

<class name=*"testNGFramework.pinesphere"*/>

</classes>

</test> <!-- Test -->

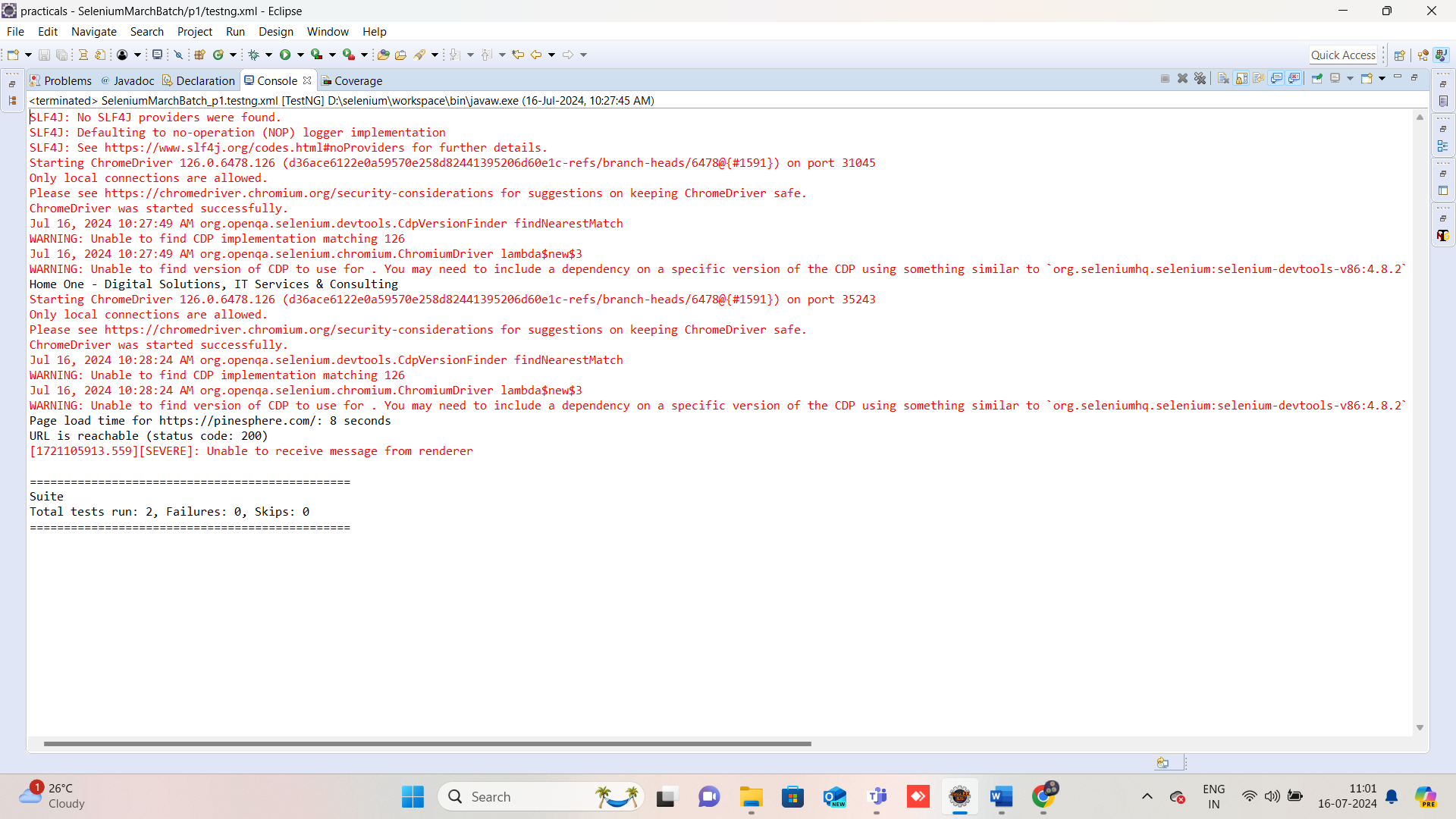
</suite> <!-- Suite -->

Output:

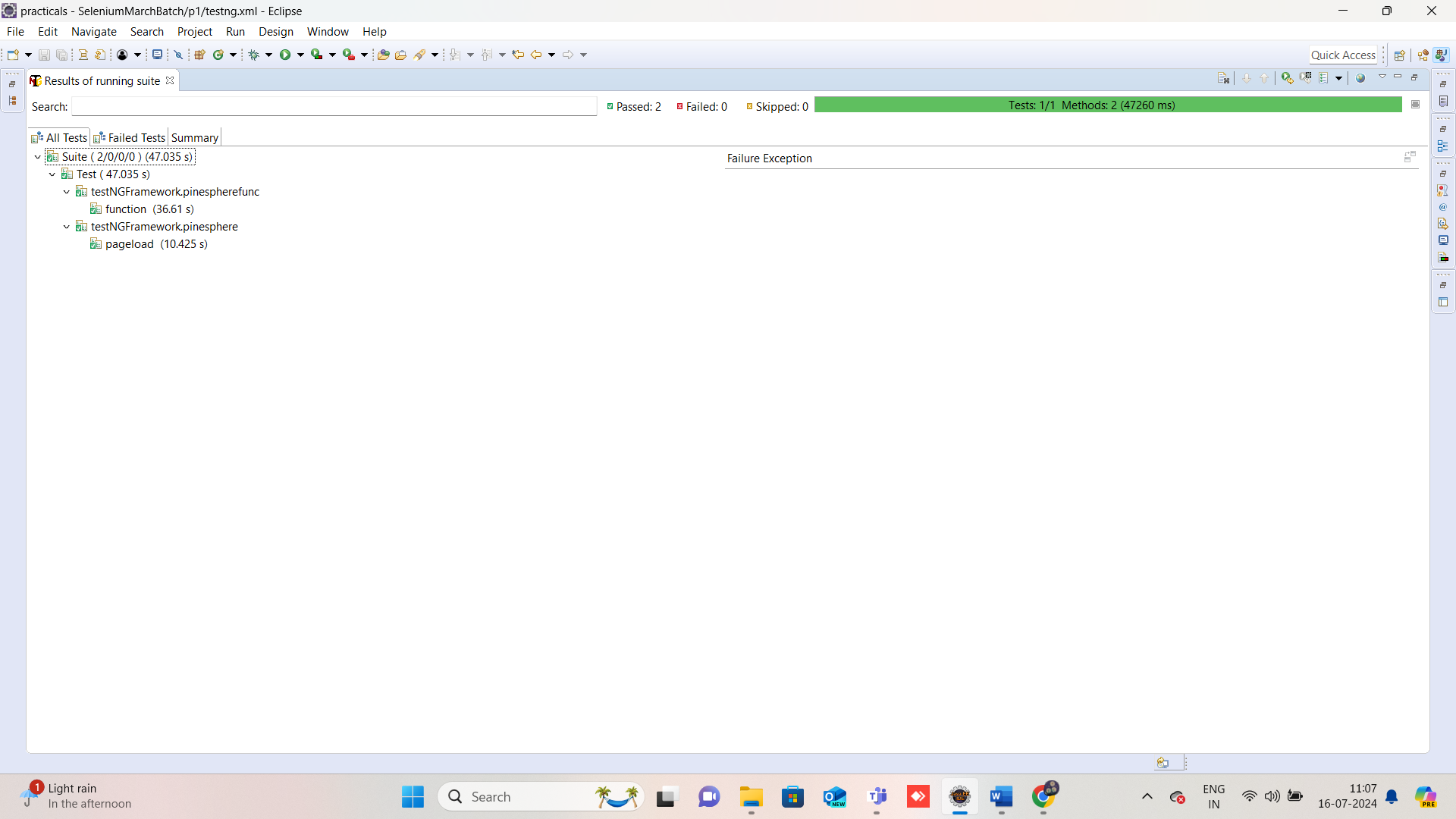
Gives as TestNG report

Eg : we can see in result of running suite :

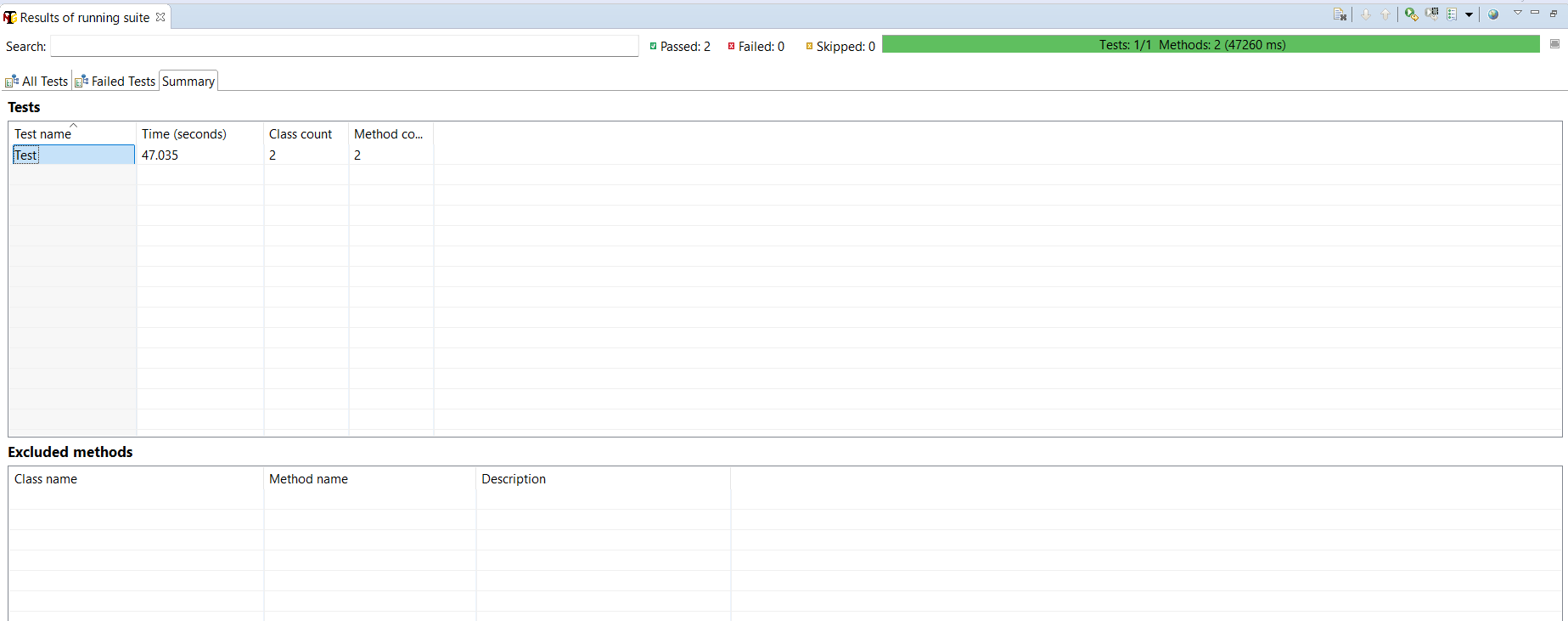
Facilities: search, All test, Failed Tests, Summary, report generation, rerun failed test , test history, previous and next failure are available.



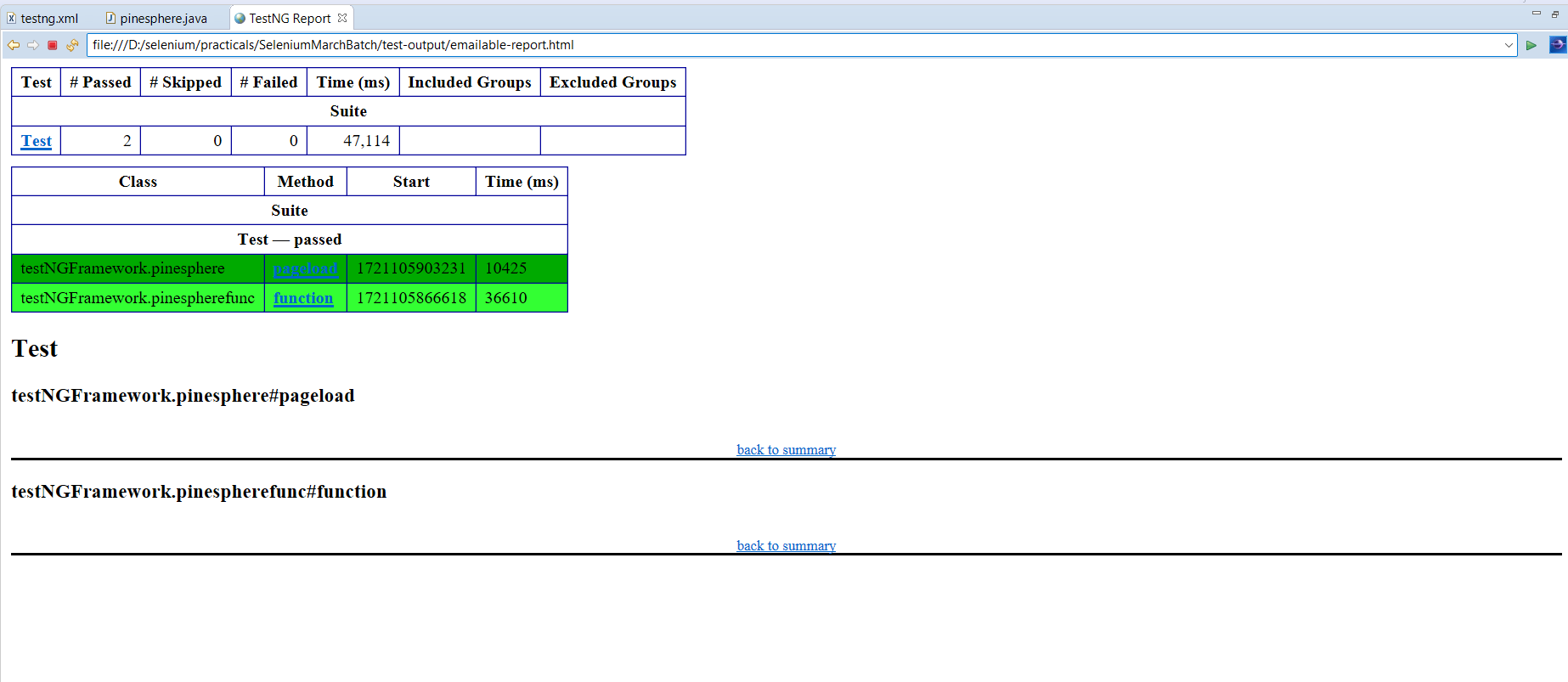
All test



Summary



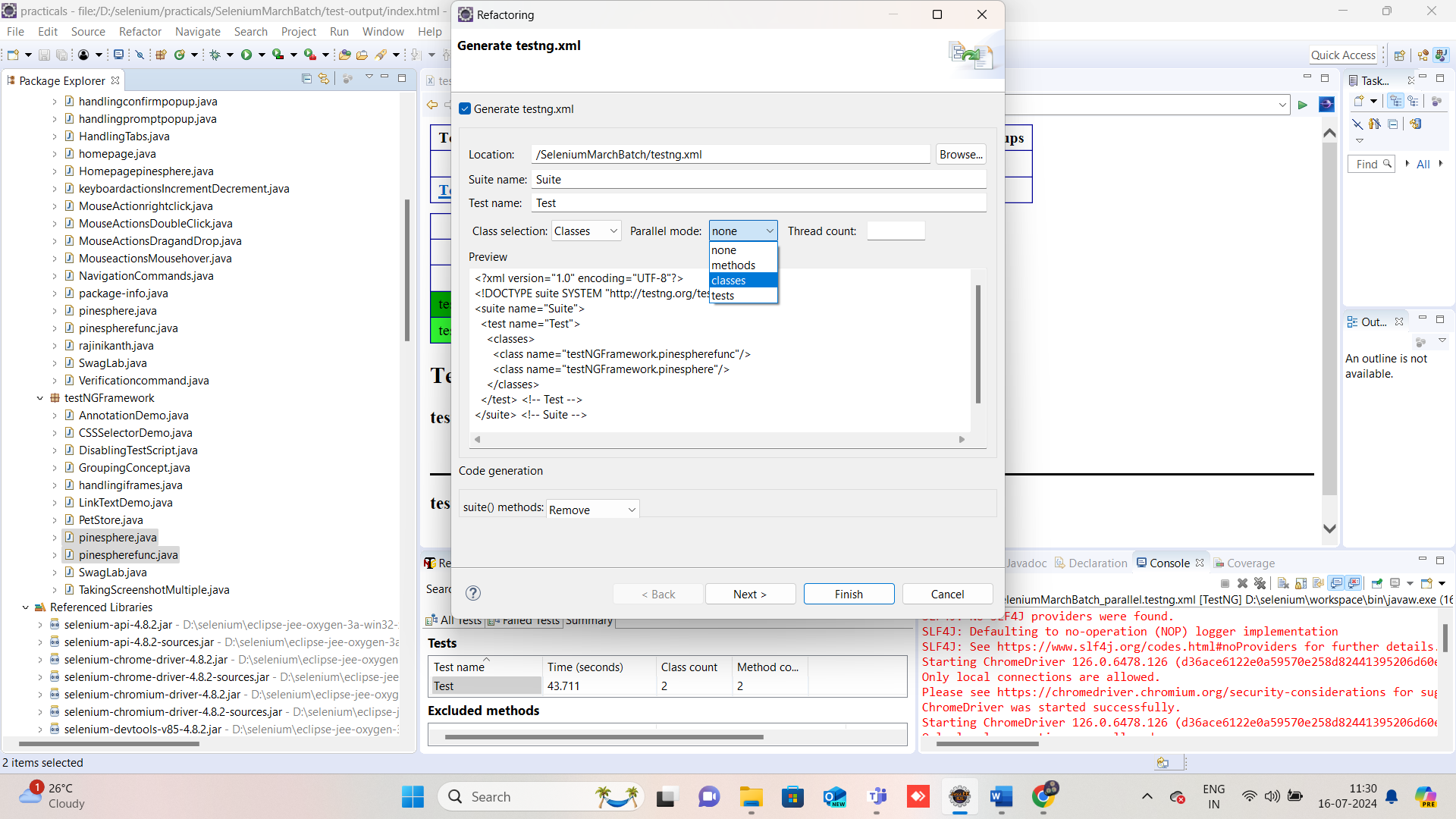
Report generation

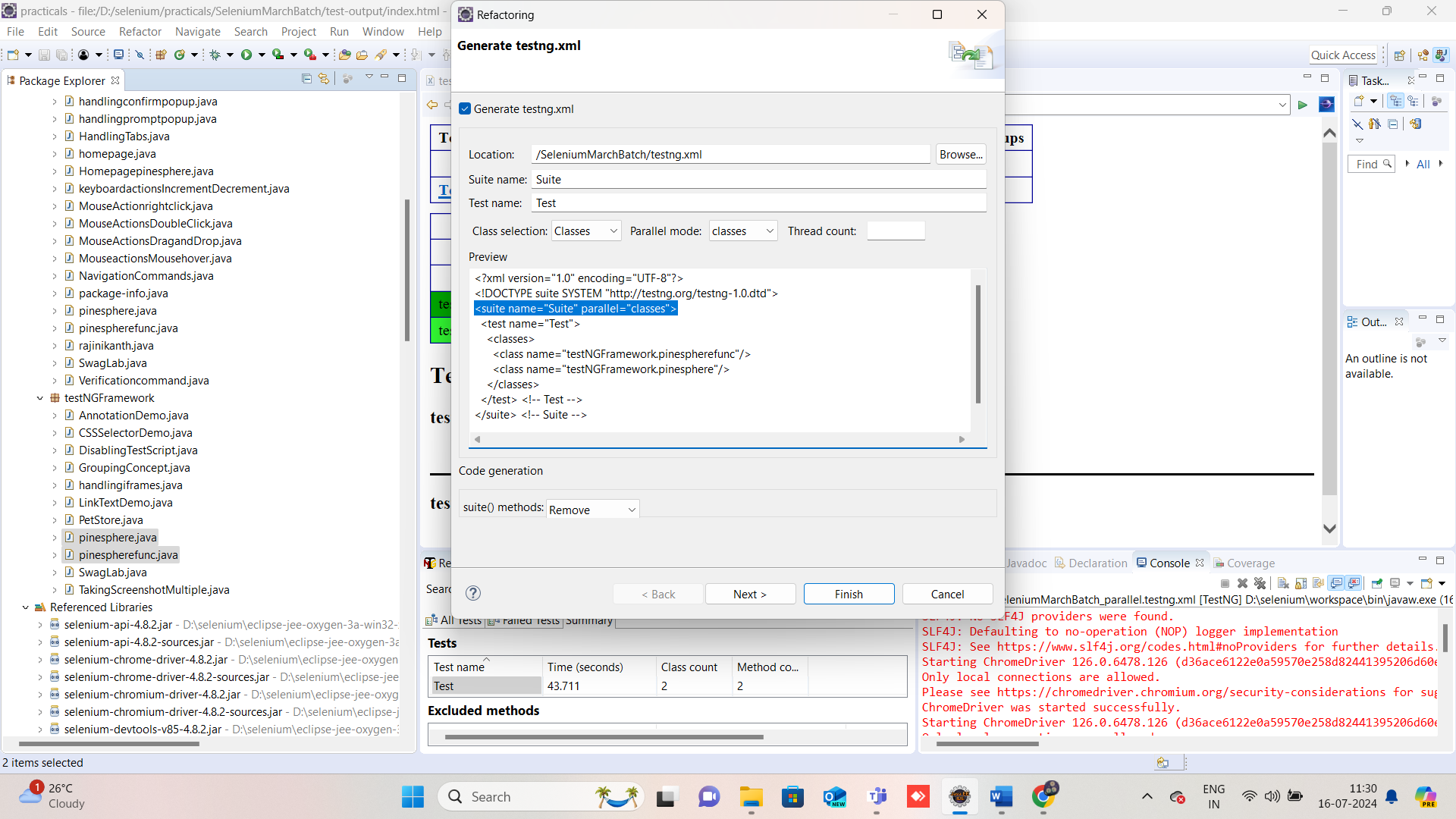


Parallel Testing or Sequential Testing:

Convert to testing ->give parallel->classes

Code changes to <suite name=*"Suite"* parallel=*"classes"*>





Thread count:

Thread count is basically number of instances running to execute multiple tests simultaneously or parallelly. The attribute thread−count allows the user to specify how many threads should be run for this execution.

Priority:

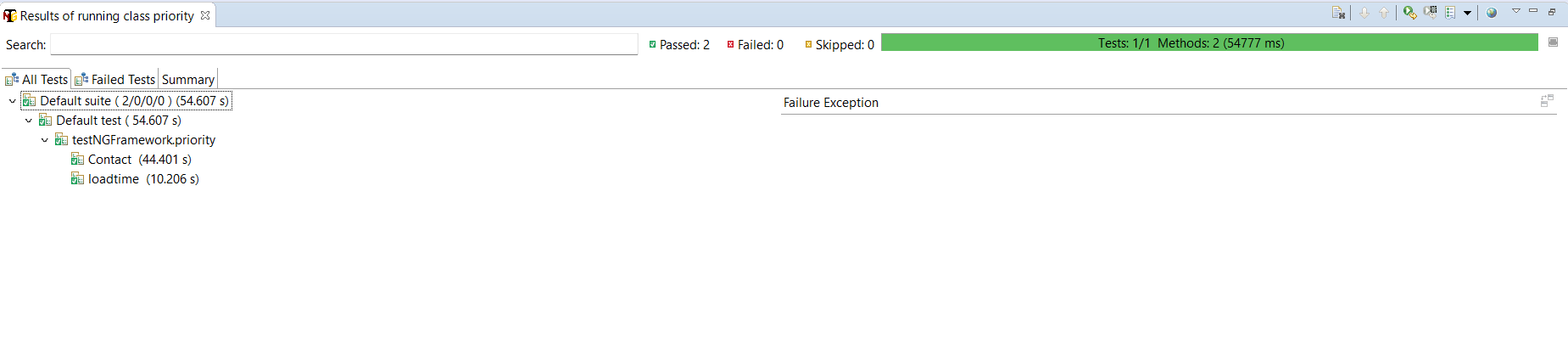
Priority is an attribute that tells TestNG which order the tests need to follow. When we have multiple test cases and want to execute them in a particular order, the TestNG priority attribute helps in executing the test cases in that order. The test cases get executed in ascending order of the priority list.

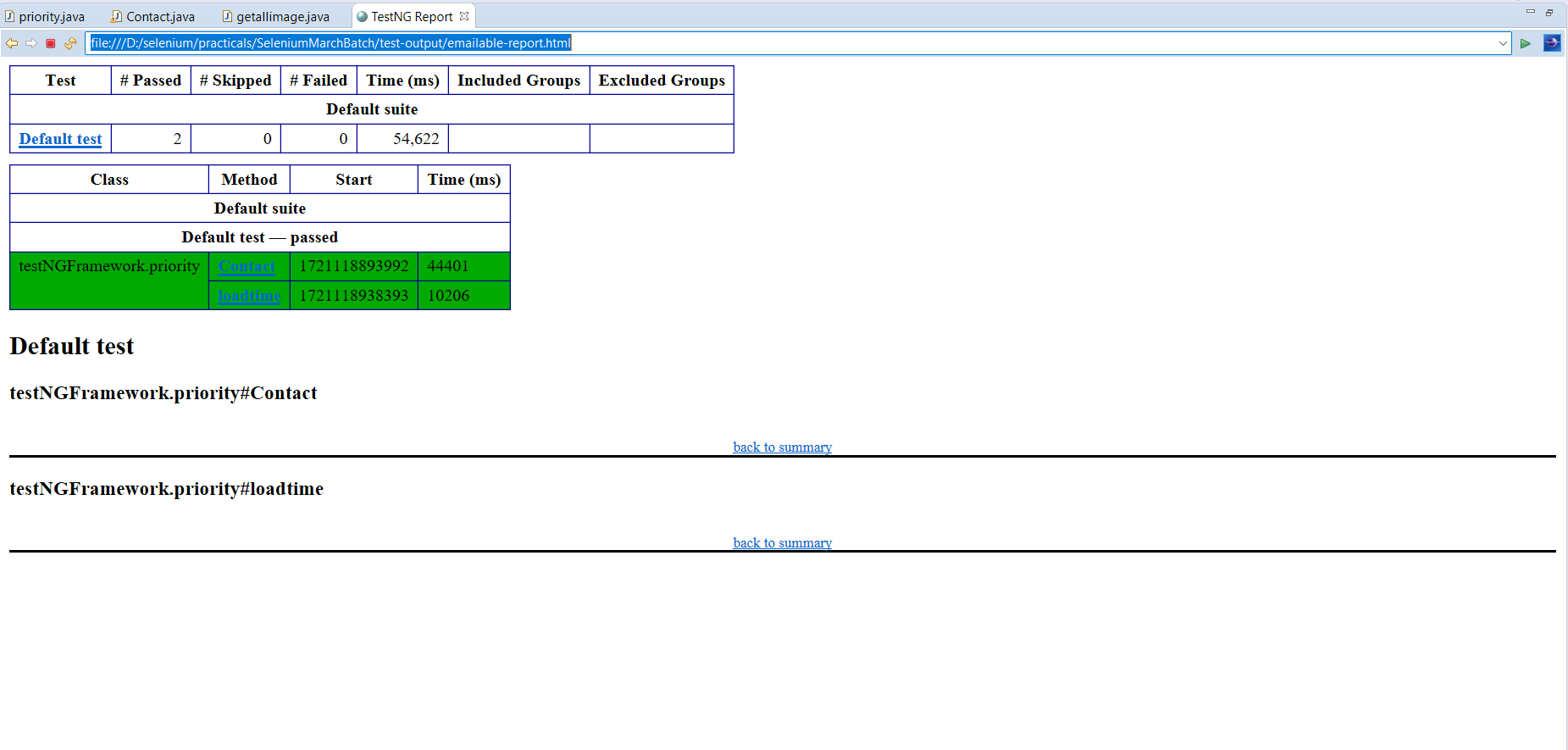
Step:1 Add the multiple test cases

Step:2 Give priority according to the convenience.

Steo 3: @Test (priority=1)

No need for XML conversion





Method dependency:

It allows a test method to depend on some other method or methods. Test Dependency allows us to make one test method dependent on one or multiple other test methods.

Source code:

**public** **class** MouseActionrightclick {

**private** WebDriver driver;

@BeforeMethod

**public** **void** setUp() {

System.*setProperty*("webdriver.chrome.driver", "D:\\\\selenium\\\\eclipse-jee-oxygen-3a-win32-x86\_64\\\\eclipse\\\\chromedriver-win64 (1)\\\\chromedriver-win64\\\\chromedriver.exe");

driver = **new** ChromeDriver();

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

}

@Test

**public** **void** rightclick() {

driver.get("https://demo.guru99.com/test/simple\_context\_menu.html");

WebElement rightClickButton = driver.findElement(By.*xpath*("//span[contains(text(),'right')]"));

Actions act = **new** Actions(driver);

act.contextClick(rightClickButton).build().perform();

// Rest of your right-click functionality here...

}

@Test(dependsOnMethods= {"rightclick"}) Basic syntax for dependsonmethods

**public** **void** doubleClick() {

driver.get("https://demo.guru99.com/test/simple\_context\_menu.html");

WebElement doubleClickButton = driver.findElement(By.*xpath*("//button[contains(text(),'See')]"));

Actions act = **new** Actions(driver);

act.doubleClick(doubleClickButton).build().perform();

// Rest of your double-click functionality here...

}

@AfterMethod

**public** **void** tearDown() {

driver.quit();

}

}

Invocation Count and Invocation Timeout

Here we are telling TestNG that we want it to run “testMethod” to run 5 times but the overall **timeout** across all the **invocations** would be 1000 milli seconds.

Source code:

package testNGFramework;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.Test;

public class SwagLab {

WebDriver driver;

@Test(invocationCount=2,invocationTimeOut=30000)

Basic syntax for dependsonmethods

public void a\_pageload() throws InterruptedException {

System.setProperty("webdriver.chrome.driver","D:\\selenium\\eclipse-jee-oxygen-3a-win32-x86\_64\\eclipse\\chromedriver-win64 (1)\\chromedriver-win64\\chromedriver.exe");

driver=new ChromeDriver();

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

driver.get("https://www.saucedemo.com/");

driver.findElement(By.id("user-name")).sendKeys("standard\_user");

driver.findElement(By.name("password")).sendKeys("secret\_sauce");

driver.findElement(By.xpath("//input[@id='login-button']")).click();

driver.findElement(By.xpath("//button[@id='react-burger-menu-btn']")).click();

Thread.sleep(5000);

driver.findElement(By.xpath("//a[@id='logout\_sidebar\_link']")).click();

}

}