**Frameworks used:**

It is a Maven build Project where BDD (Behavioural Driven Development) – Cucumber and POM (Page Object Model), DDT( Data Driven Testing)- Excel are used.

**Design Pattern used:**

Factory Design Pattern used.

**Details:**

List of Files:

* A *Page Object Manager* class file initializing page objects of all pages on the website.
* A *Web Driver Manager* class file initializing the browser driver.
* A *Test Context Set Up* class file initializing *Page Object Manager* and *Web Driver Manager.* Also, the invoked browser driver object is passed to *Page Object Manager.*
* A *Feature file* representing all Test Cases as *Scenarios*.
* All *Step Definition* class files initializing the *Test Context Set Up* file through Dependency Injection and invoking the respective Page Object class files accordingly.
* A JUnit/TestNG *Test Runner* class file execute the actual test script written over the *Step Definition* class files by referring to the *Feature file.*
* A separate class file named *Hooks* to run hooks before and after each scenario.
* A class file named *Reusable Method* having frequently used methods.
* Two *properties* files *Extent Report* for Extent Reporting and another one *Parameter* file storing GUI locators/Elements and Global/Project Configuration details/fields like browser name, URL, and so on.

We do functional Testing using Selenium not Performance or any other testng

With cucumber bdd, 10 most common problems that need to be addressed in

framework design is solved WITHOUT ANY USE OF CODE..ie) in gherkin language

itself it is fulfilled..

Cucumber BDD:

Feature file: .feature extension...

tags: given, when, then, and, but

Scenario, Scenario Outline, examples, background

Tags: @Smoketest @SanityTest, @WebTest

Regular expressions: in scenario write within quotes and in step definition \"([^/"]\*)\" for one or two data

DataTable: data pipelined in scenario and in step definition in the method call, write (DataTable data)

Scenario Outline: parameterization of scenario with multiple data...use <> and examples in scenario

Examples:

and in step definition use (.+)

Background: pre requisite that is to be executed before all scenarios is written here

NOTE: background and hooks cant be used together..

Step definition file:

tags: given, when, then, and, but

hooks with tagnames

if pre-requisite for all scenarios differ, then hooks with tagnames used.

@Before("@SmokeTest")

TestRunner file:

Run with

two types: running with junit or testng(extends AbstractTestngCucumberTests)

cucumber options:

features---src/java/features

glue----stepdefinition, tags="@SmokeTest", dryRun=true, monochrome=true, strict(default)

plugin={"pretty","html:target/cucumber.html","junit:target/cucumber.xml","json:target/cucumber.json"}

dryRun: checks that for a given scenario any step definition is missed

monochrome: gives docs in the console in a human readable format

strict: runs only if all scenarios passed

10 quetions common problems:

1. How to write and maintain reusable code across the framework?

with the help of step defnition and regular expression context..

2. How to drive the data into the testcases?

dataTable helps to drive the data...

3. How to parameterize the testcases with multiple set of data?

Scenario Outline...

4. how to achiece 100% test coverage through the framework?

Cucumber-Gherkin scenario...

5. How to run all the tests on single click?

Test Runner...

6. How to control the test execution to select and run only few based on needs?

Tags in feature file and mention the same in testRunner

7. How to implement logic to separate test code from common post andpre requisite of the

test?

Hooks and Tags/Background...

8. How to generate HTML reports and Junit reports?

Plugin cucumber option in test Runner

9. How to develop Maven framework with existing code?

10. How to integrate developed framework to Jenkins Continuous Integration environment?

11. how to run testcases parallely in cucumner using testngrunner?

in testrunner, using object[][] call super.scenarios()...

@Override

@DataProvider(parallel=true)

Dependency Injection and Design Pattern for cucumber scenarios:

----------------------------------------------------------------

how to transfer the state of variables between stepDefinitions

eg: the driver in 1st file runs fine, but in the 2nd file shows error

bcz the driver is not said to be assigned to something like in 1st file..driver= new chromeDriver()

Qutn: when step definition file broken down to 2 files, what hapens?

nullpointer exception occur for the variables that are shared among them so

so that dependency injection used to fix it.

A class named TestContextSetUp is used and the variable shared initialized there.

in step definition files, the instance variable is initialized inside the constructor

so that it is used as instance variables like tcs.driver.... (object.method is refered as instance variable of testContextSetUp class)

==============================================================================================

Single Responsibility Principle:

----------------------------------

loosely coupled class files in separate step definition files

single class handles only those functions in that page not others...

=====================================================================

PageObject Pattern:

---------------------

Creating locators of different pages in different classes

using by, locators written, action method that returns the locator to step definition file.

in step definition file, class object instantiation done with drivers passed as argument

========================================================================

Factory Design Pattern:

-----------------------

a single class is created and given the responsibility of creating class objects for

all page object files classes

=======================================================================

TestContextSetup:-----------------------------this is used in Stepdefinition files where all the class objects are initialized

WebdriverManager class

pageobjectManager class

Reusable class

===============================

static scenario line in stepdefinition is a string..no cap and dollar inside tag...

but dynamic scenario line in stepdefinition is a regular expression.. so use cap nd dollar.

also parameter {string} should change with (.+)

========================================

Hooks:

driver quit written in @after hook.

here driver needs life from WebDriverManager's getDriver method. so using the object of the class it can be done.

but already its obj is in testContextSetUp class so link testContextSetUp in hooks class

@After-----------executed after all scenarios done

@AfterStep-------executed after every scenario line

============================================

Extent reports:

3 things:

1----2 dependencies: extent report by aventstack and

extent report cucumber7 adapter by grasshopper

2---in testrunner file: write keyword inside plugin cucumber option

3---in src--test--java---create resource folder. in it create extentreport.properties file

in it write some lines..basefolder.name, basefolder.datetimepattern, extent.reporter.spark.start, extent,reporter.spark.out...

-----------------------------------------

for failed scenario screenshot:

in hooks at hook named @AfterStep write some code:

1. in it, driver is converted to TakesScreenshot class and told to getscreenshot as output file type

2. then using scenario- class object, attaching screenshots to extent reports:

in hooks file: for that just attach screenshot to scenario object,

in the extent.properties file: then extent reports by defalut has the ability to read the scenario object

and find out any screenshot attached...if so it captures and attach it back into the extent report(for that write clearly the path)

3. in testrunner file: change the feature keyword value to "@target/failed\_scenarios.txt"(@ at first, path to .txt file)

==================================================================

Junit and Testng testRunner files:

-----------------------------------

Cucumber cant run alone so depends on junit or testng framework to run via testrunner file

while using junit use:

1. cucumber-junit and junit dependencies in pom.xml

2. and to import junit.cucumberoptions in testrunner file

3. use a 'test' word (at end for safer) in testrunner file

while using testng use:

1. cucumber-testng and testng dependencies in pom.xml

2. and to import testng.cucumberoptions in testrunner file

3. extends abstracttestngcucumbertests class in testrunner file

--------------------------------------------------------------------------

how to run parallely the tests with testng and junit testrunner file?

----------------------------------------------------------------------

using testng:

1.to override scenarios method returning object type

2. both scenarios and feaure files can run parallely

using junit:

1. junit cant run tests parallely, but with the help of 'maven' along with junit,

tests can run parallely ------by adding a small configuration in "pom.xml"

2. only feature files can run parallely, scenarios only execute sequentially

----------------------------------------------------------