CUCUMBER & GHERKIN

1. **Tell me more about Cucumber, how did you guys decide to start using Cucumber ?**
   - In the past few years, more and more IT teams follow Agile methodology in their development process to adapt to the rapid changes of the market. This is also a challenge for the test team in managing test cases and test scripts which can be changed when the requirements are updated monthly. Finding a suitable testing method from the beginning is one of the keys to the success of an Agile software project.
   - Many Agile teams have successfully applied Behavior Driven Development (or BDD) approach in testing process using the Cucumber tool. So, what is Cucumber? And why is it one of the good approaches in Agile projects, used together with BDD?
   - Cucumber is a tool for running automated acceptance tests written in a behavior driven development style. One of its wonderful main features is the ability to execute plain text functional description (written in language named Gherkin) as automated tests. Here is an example:

```
Feature: Update password
Scenario: Admin user can update user password
Given I am in the HR system with an Admin account
When I update password of another "user"
Then I receive a message for updating password successfully
And user's password is updated to the new password
```

   - This great feature has played a primary role in supporting the BDD approach with the following **advantages**:
     - Writing BDD tests in Ubiquitous language, a language structured around the domain model and used by all team members including developers, testers, BAs, etc.
     - Building bridges between the technical and nontechnical members of a software team
     - Allows interaction directly with the developers' code, but written in a language that business stakeholders can understand
     - Last but not least, Cucumber is an Automated Acceptance Test Tool which running tests written in a behavior driven development (BDD) style.
   - **Cucumber Tool helps to improve communication between technical and non-technical members in a project.**

2. **Tell me what are the most important things in Cucumber, what makes it unique ?**
   - Features file, Step Defs, Runner Classes, Hook Class, Tags

3. **How to see your reports in cucumber?**
   - My framework generates cucumber reports folder in the target folder which contains the reports.
   - When we run the tests on Jenkins, Jenkins saves the report of every run.
   - Home page of the Jenkins job always points to the last run reports.
   - All the reports for previous runs can be found under the build number.
   - Go to target folder
   - Open with system explorer
   - Go to target>cucumber report>index shows the tests you ran

4. **What is Gherkin?**
   - Language used by feature files
   - Feature, Scenario, Given, Then, When, And, But, BackGround, Scenario Outline

5. **What are the components of Cucumber BDD framework?**
    1. **Feature files**
        - Consists of scenarios that test a certain feature or functionality
        - Feature is main story while scenarios are the test cases to the story(feature)
    2. **Cukes Runner**
        - A class that strictly runs the tests, generates codes for step definition
        - @smoketest
        - Cukesrunner → IN CUCKESRUNNER I HAVE A FEATURE LOCATION THAT SHOWS WHERE MY FEATURE ARE LOCATED
    3. **Step definition**
        - A class that made of steps that starts with Gherkin language
        - Make sure the step definition is in the same package as cukes Runner, or child package (not parent or sibling)
- FOR NON-TECH PPL TO UNDERSTAND
- DEPENDECY BDD IS A DEPEN
- MVN REPOSITORY IN THE POM.XML FILE
- CUCUMBER BDD FROM CUCUMBER.IO
- Combine techs of TDD
- Behavior driven
- Express the flow customer behavior → Don't focus on the elements

6. **What does @CucumberOptions do?**
    - Tag used to customize the running of the cucumber tests
    - Inside @CucumberOptions you can add:
        - dryRun
        - Plugin
            - "**Pretty**"
                - Adds more info in the console → Gives you tag, scenario, method info.
                - "html:target/cucumber report" → Generates html report located in target/cucumber report folder "json:target/cucumber.json
            - **Tags**
                - Tags must be located in feature path
                - Can add multiple tags…tags= "@Dog, @Cat"
            - **Features** location of where feature files are
            - **Glue** where to look for step definition steps. hook class is part of glue too.

7. **How to run Cucumber with JUnit?**
    - Add cucumber JUnit dependency
    - Adding @RunWith (Cucumber.class) on top of cukesRunner class

8. **How to run Cucumber with TestNG?**
    - Add cucumber testNG dependency
    - Make CukesRunner extend to AbstractTestNG CucumberTests

9. **What happens we you run your runner class with no tags?**
    - All the feature files will run from top to bottom but only the feature files that are located in the @CucumberOptions "features="

10. **What are Hooks in cucumber?**
    - Cucumber hook allows us to better manage the code workflow and helps us to reduce the code redundancy. We can say that it is an unseen step, which allows us to perform our scenarios or tests.
    - Class that uses
        - @Before → runs before each cucumber scenario
        - @After → runs after each scenario (It will always run no matter if scenario passes or fails)
    - Class must be in same package as stepDefinition
    - I implemented screenshots inside hook class
    - Hook Class will not run if dryRun=true
    - I use Scenario as a parameter in my before/after method

11. **How do you take screenshots in cucumber?**
    - In my Aftermethod I use a code:
    - I use TakeScreenShot interface
    - You can store screenshot as a byte or file
        - @After

```
public void tearDown(Scenario scenario) {
   if(scenario.isFailed()) {
   //taking a screenshot
 final byte[] screenshot = ((TakesScreenshot)
                              Driver.getDriver()).getScreenshotAs(OutputType.BYTES);
 //adding the screenshot to the report
 scenario.embed(screenshot, "image/png"); }
```

12. **How to run a Cucumber with DDT?**
    - I use Cucumber tables:

      | Home | Emails | Documents | Projects |

    - You get the method with (DataTable arg1)
    - In the parameter DataTable you can change it to

      List<YourType>, List<List<E>>, List<Map<K,V>>, and Map<K,V>
    - Prints in order for list
    - No order for map

13. **What is Background?**
    - Cucumber has their own before method
    - The one in hooks is for java
    - A step that runs BEFORE a scenario inside the feature file
    - Can only put on top, before all scenarios
    - Cannot put pipelines in backgrounds (Only in scenario outline)

14. **What is Scenario Outline? vs Scenario?**
    - Scenario in cucumber runs once.
    - Used for data driven testing
    - Have the same cucumber steps but we provide data after the scenario as a table using keyword examples

15. **How do I limit the types of variables I can pass?**
    - In the gherkin parenthesis you can add (Collaboration | Sales | Marking, etc.)
    - Ex: @When("^I hover over the (Collaboration | Sales | Marketing | Activities | All) menu$")

```
public void i_hover_over_the_Collaboration_menu(String menu) {
switch(menu) {
case "Sales":
   BrowserUtils.hover(dashboard.sales); break;
case "Marketing":
   BrowserUtils.hover(dashboard.marketing);  break;
case "Collaboration":
   BrowserUtils.hover(dashboard.collaboration);  break;
case "Activities":
   BrowserUtils.hover(dashboard.activities);  break;
case "All":
   BrowserUtils.hover(dashboard.all); break;};
```

16. **What if you have a scenario that has two parameters (limiting parameter, table parameter)?**
    - Example :
        - Scenario: Verify Collaboration menu options
        - Given I logged into suiteCRM
        - When I hover over the Collaboration menu
        - Then the following menu options should be visible for Collaboration:
          | Home | Emails | Documents | Projects |
        - In this scenario i have a table, I want to limit collaboration to just collaboration and the other menus categories
    - Solution:
        - @Then("^following menu options should be visible for
          (Collaboration | Sales | Marketing | Activities | All):$")
        - public void following_menu_options_should_be_visisble_for_Collaboration(String menu, List<String> options) {
        - String menu represents the 5 menu options (Collaboration | Sales | Marketing | Activities | All)
          List<String>options represents the tables; | Home | Emails | Documents | Projects |

17. **How do I use cucumber scenario for DDT?**
    - In my current project I use Scenario Outline with Examples
    - In my scenario feature file, whenever I'm using a variable as a data driven, I use "<variable>"
    - Then in Examples:
      | variable | column name
      | data1. | row1
      | data 2 | row 2
      | data3 | row3

20. **Data driven**
    - Test data is separated from code and stored into external sources: Cucumber Examples table, Excel files, CSV files, Database.
    - If the amount of data is not that huge, then I use Cucumber Scenario outline with Examples table.
    - And other times I maintain test data in Excel files, and I use Apache POI library to read and write data
    - If data comes from a database, or I need to do database validation, I use SQL queries along with JDBC library in java.

18. How to use Maps in cucumber?

- Using a nonScenario Outline
- Scenario: Create contact using a map
  - Given I logged into suiteCRM
  - When I create a new contact:

first_name	John	
last_name	Smith	
cell_phone	801 888 8889	

  - Then I should see contact information for "John Smith"
  - Left side is key, and right is value 2 columns only
- Using a Scenario Outline
  - Scenario Outline: Create contact using a map
  - Given I logged into suiteCRM
  - When I create a new contact:

first_name	<first_name>	
last_name	<lname>	
cell_phone	<cell_phone>	
office_phone	<office_phone>	

  - Then I should see contact information for "<first_name> <lname>"
  - Examples:

first_name	**lname**	**cell_phone**	**office_phone**	
Michael	Jackson	1234567890	2345678891	
Bonnie	Garcia	4569871234	4567890987	

- In step def I write;

```
@When("^I create a new contact:$")
 public void i_create_a_new_contact(Map<String,String>contact) {
 // open the create contact dialog
```

21. How to use POJO in cucumber?

- Create **contactBean** class
  - Add all variables
  - Add the getter/setters
- Create bean feature file
- Create a table with first row containing the variables in the contactBean class
  - Add values under the table
  - Implement method with parameter (List<ContactBean>contacts)
- Scenario: Create contact
  - Given I logged into suiteCRM
  - When I save a new contact:

firstName	**lastName**	**officePhone**	**cellphone**	**email**	
Steve	Gates	3456758888	1234329999	SteveGates123@gmail.com	

  - Then I should see contact information for "Steve Gates"

22. How to run a group of test case using TestNG?

```
@Test (groups={“smokeTest”,”FunctionalTest”})
public void loginTest(){
            System.out.println(“Logged in successfully”);
}
```

23. Data Driven Testing

- ***WHEN:*** Whenever a functionality or a module in an app requires testing with multiple sets of data(Parametrization), Multiple inputs then we need to perform data driven testing and automation.
- These scenarios are one of the things That must be automated.
- ***HOW:*** Test data is separated from code and stored into external sources: Cucumber Examples table, Excel files, CSV files, Database.
- ***BENEFIT:*** More organized, Data centralized, Collaboration on test data - it can come from BA, MTs etc

24. How can we create data driven framework using TestNG?

- By using @DataProvider annotation, we can create a Data Driven Framework

```
@DataProvider(name=”getData”) Public Object[][] getData(){ Object [][] data = new Object[2][2];
Data[0][0] = “firstUid”; Data[0][1] = “FirstPWD”;
Data[1][0] = “SecondUid”;
Data[1][1] = “SecondPWD”; Return data; }
```

25. How to create Group of Groups in TestNG?

- These groups are called metagroups.
- Example: you might want to define a group all that includes smokeTest and FunctionalTest.
  Let’s modify our testing.xml file:

```
<groups>
   <define name=”all”>
            <include name =”smoke Test”/>
            <include name = “functionalTest”/>
   </define>
   <run>
            <include name = “all”/>
   </run>
</groups>
```

26. How to run test cases in parallel using TestNG?

- We can use “parallel” attribute in testng.xml to accomplish parallel test execution in TestNG
- The parallel attribute of suite tag can accept four values:
  - Classes → All the test cases inside a java class will run parallel
  - Methods → All the methods with @Test annotation will execute parallel
  - Instances → Test cases in same instance will execute parallel but two method of two different instances will run in different thread. <suite name=”softwaretestingmaterial” parallel=”methods”>

27. How to ignore a test case in testNG?

- To ignore the test case, we use the parameter enabled = false to the
- @Test annotation @Test(enabled=false)

28. How to exclude a particular test method from a test case execution?

- By adding the exclude tag in the testing.xml

```
<classes>
   <class name="TestCaseName">
      <methods>
         <exclude name="TestMethodNameToExclude"/>
      </methods>
   </class>
</classes>
```

29. How to exclude a particular test group from a test case execution?

- By adding the exclude tag in the testing.xml

```
<groups>
   <run>
      <exclude name="TestGroupNameToExclude"/>
   </run>
</groups>
```

30. What are the different way to produce reports for TestNG results?

- TestNG offers two ways to produce a report
  - Listeners implement the interface **org.testng.testListener** and are notified in real time of when a test starts, passes, fails, etc...
  - Reporters implement the interface **org.testng.reporter** and are notified when all the suites have been run by TestNG.
- The IReporter instance receives a list of objects that describe the entire test run

31. What is the use of @Listener annotation in TestNG?

- configure reports and logging.
- widely used listeners : ITestListener interface.
- It has methods like onTestStart, onTestSucess. onTestFailure, on TestSkipped...
- we should implement this interface creating a listener class of our own,
- Next, we should add the listeners annotation (@Listeners) in the class

32. What Is a Regular Expression, Regexp, or Regex?

- A regular expression is a special text string for describing a search pattern.
- You can think of regular expressions as wildcards on steroids.
- You are probably familiar with wildcard notations such as *.txt to find all text files in a file manager.
- Regex equivalent is.*\.txt.

33. How to write regular expression in testing.xml file to search @Test methods containing "smoke" keyword?

- Regular expression to find @Test method containing keyword "smoke" is a mentioned below

```
<methods>
   <include name=".*smoke.*"/>
</methods>
```

34. What is the time unit we specify in test suites and test cases ?

- We specify the time unit in test suites and test cases is in milliseconds.

35. What is the use of @Test(invocationCount= someInteger)?

```
@Test(invocationCount=10)
Public void testcase(){}
```

- //the invocation count attribute tells how many times TestNG should run a test method

36. What is the use of @Test(threadPoolSize=someInteger)?

- The threadPoolSize attribute tells to from a thread pool to run the test method through multiple threads
- Note: this attribute is ignored if invocation count IS NOT SPECIFIED

37. **What does the test timeout mean in testing?**
    - The maximum number of milliseconds a test case should take

```
@Test1(threadPoolSize=3,invocationCount=10,timeOut=10000)
public void test() {}
```

    - : // in this example: the function test1 will be invoked ten times from three different threads, Additionally, a time-out often seconds guarantees that none of the threads will block on this thread forever.

38. **What are @Factory and @DataProvider annotation?**
    - @Factory → executes all the test methods present inside a test class using a separate instance of the class with different set of data
    - @DataProvider → a test method that uses dataProvider will be executed the specific methods multiple number of times based on the data provided by the dataProvider.

39. **annotations - priority**
    - Doesn't matter what number you start Ex: @Test(priority=0)
    - DependsOnMethods = "test method name" You Can add multiple test names
    - If the first one fails, the 2nd test won't run at all
    - If the first method failed, your report will show that the 2nd test will be skipped

40. **parallel execution in testNG**
    - In xml file write.
        - parallel="tests"thread-count="4"
    - Thread-count is how many browsers you want to open same time
    - In xml file you can add .* to run everything
        - Ex:<package name=".*"></package>
    - TestNG has its own reports -When you run xml, it gives you the report in test-output folder
    - Contains the test report in html

41. **Framework Tools : Cucumber BDD framework**
    - Junit, Cucumber Java, Maven
    - Selenium, HTML reporting with screenshots Log4j,
    - JDBC, Rest Assured, Apache POI, Git, Jenkins

42. **Framework Tools: TestNG + Selenium**
    - Java, Maven, TestNG,
    - Selenium, Extend Reports with screenshots Log4J,
    - JDBC, Rest Assured, Apache POI, Git, Jenkins

43. **How does your framework generate reports?**
    - Our Cucumber BDD framework generates HTML reports.
    - The report shows the pass/fail coverage for feature files, tags, steps
    - The report contains all the steps for each test The report has screenshots for failures

44. **How to run tests selectively cucumber?**
    - tags keyword the cukesrunner
    - feature keyword the cukesrunner
    - tags and features can also be passed using the command line
    - mvn test -Dcucumber.options="--tag @smoke"

45. What do you use for logging?

- I use Log4J for logging. I always log important steps in the test execution. That helps me to debug when there is a failure.
- Log4J is not a replacement for HTML reports.

```
<dependency>
   <groupId>org.apache.logging.log4j</groupId>
   <artifactId>log4j- core</artifactId>
   <version>2.11.0</version>
</dependency>
```

46. How does the FEATURE FILE WORK?

- **Feature** → description of what is being tested @tags. Sample feature file;
    - Feature: login functionality → Background:
    - Given I am on the login page → Scenario: 1, Scenario: 2
    - The background runs before both of the scenarios
- **Scenario** → description of the scenario being test
    - Given I am on the login page
    - And I enter username and password
    - When I click on the submit button
    - Then I should be able to see the profile picture
    - But the submit button should not be displayed
- **Given** → a precondition
- **When** → condition that triggers the expected result Then —> expected condition

47. What is test base Class ? and How do you implement in your framework ?

- Test Base class is class where I have most used methods in my tests.
- My test classes extend the Test Base class and thus have access to those methods. This helps me us make my code reusable
- Before/after test methods wait/synchronization utility methods.
    - SwitchToWindow(title)
    - WebDriver driver;

48. How to rerun the failed tests again in TestNG?

- In my TestNG framework, failed tests are reported in the testng_failed_.xml file in the target folder.
- We can add this file in the pom file so that maven will try to run the failed tests every time.
- If will only run when there are failures in the test.

49. How to rerun the failed tests again in Cucumber?

- we use the re-run option in the CukesRunner.
- Add the rerun to cukes runner.
- This option will create a file with a list of failed tests
- Create a second runner class which points to file with a list of failed tests
- Add the second runner in the pom file

50. How to rerun the failed tests again in Jenkins?

- In Jenkins there are plugin that re run the failed tests Unit cases.
- So you can configure your Maven build execution on Jenkins using the option:
- Dsurefire.rerunFailingTestsCount=2

51. RUNNING CUCUMBER TESTS IN PARALLEL?

There are couple options on how to make Cucumber + JUnit framework run in parallel

1. There is a plugin **cucumber-jvm-parallel-plugin**
   https://github.com/temyers/cucumber-jvm-parallel-plugin
   - This plugin automatically generates multiple cukes runner files.
   - Based on the configuration, this plugin creates one cukes runner per feature file.
   - Each runner will point to one feature file. and these cukes runners will run in parallel.
   - Normally cucumber runs feature files one after another. when we use this plugin, it runs starts them at the same time. we can specify how many tests running at the same time

2. **Cucumber 4.x parallel option**
   Starting from cucumber 4.0, cucumber supports parallellization natively.
   https://cucumber.io/blog/2018/09/24/announcing-cucumber-jvm-4-0-0
   By official documentation, in order to run tests in parallel, we have to add parallel option to the maven surefire plugin in pom file.

```
<build>
   <plugins>
      <plugin>
         <artifactId>maven-surefire-plugin</artifactId>
         <configuration>
            <parallel>both</parallel>
            <threadCount>4</threadCount>
         </configuration>
      </plugin>
   </plugins>
</build>
```

   But in my specific project, we added maven failsafe plugin to make sure that tests continue to execute even though some fail. This plugin makes sure that tests keep running

```
<plugins>
   <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-failsafe-plugin</artifactId>
      <version>2.18</version>
      <configuration>
         <testFailureIgnore>true</testFailureIgnore>
         <skipTests>false</skipTests>
         <includes>
            <include>**/runners/*TestRunner.java</include>
         </includes>
      </configuration>
   </plugin>
<plugins>
```

3. Second plugin is **maven-surefire-plugin**
   this plugin executes tests in parallel. in this plugin configuration, we indicate which runner files we want to run. we can also indicate how may simultaneous tests we want to run.

```
<include>**/runners/*TestRunner*.java</include>. → plugin will run these files
<threadCount>10</threadCount> → this shows how many browsers we want to have in at the same time.
<parallel>classes</parallel> → this line tells that cukes runner classes must run in parallel
```

We created cukes runner files based on how many tests we want to run and how we want to break down tests.
Each cukes runner will point to certain set up scenarios/feature files

How to run?

- we can execute tests in parallel in our framework only by running tests as a maven command
- **mvn verify** → this will run the tests and generate reports
- **mvn clean verify** → it will first delete the target folder, then runs tests, then generate reports

Benefits of parallelization:

- cuts down on the execution time. UI tests usually take a long time, especially in regression testing.

Challenges of parallelization?

- hard to implement --> not easy to do.
- load --> if we open too many instances in the same machine, it can overload the machine. it will result in tests running slow, and it increases the fail rate.
- this can addressed by running tests in different machines using GRID.
- certain tests cases did not work in parallel in my project.