SELF INTRODUCTION

1. Hi, I am S.Arunsivadas working as a software Engineer at Accenture Solutions Pvt Ltd with 4 **year** of experience in software Testing.
2. I had worked in different domains such as Ecommerce and Airlines.
3. I have good experience in core-Java.
4. Also I have experience in web application testing using selenium webdriver
5. I worked on different frameworks such as cucumber with Junit, TestNG with integration of Page Object model and Data-Driven framework.
6. In cucumber ,I am good in creating feature file and step definition file
7. As a manual Tester I am good in creating test cases
8. I have good experience in handling tools like
    a. Maven as build Management,
    b. GIT as source code Management,
    c. Jenkins as Continuous integration tool and
    d. JIRA as defect Management tool.
9. In JIRA, I am good in raising defects and track it to the closure
10. I had worked on both waterfall and Agile methodologies
11. In Agile I will be actively participate in Agile ceremonies like
    a. sprint grooming,
    b. sprint planning,
    c. sprint review,
    d. sprint retrospective and daily standup.

ROLES AND RESPONSIBILITIES

1. Analyzing the user stories
2. Based on the acceptance criteria I will do feasibility check. that is what all the part going to do by automation and manual.
3. Then I will create the test cases for manual testing and feature file for automation testing.
4. Also I will prepare test data.
5. Once I complete scripting, I will review it from the lead.
6. After getting reviewed I will start the execution and analyze the report for defect and failure.
7. If any scenario getting failed I will raise the defect and assign it to the developer.
8. Once the developer fix the bug I will be retesting it and verify the bug is fixed.
9. If the bug is fixed I will close the story to done status, If it is not fixed I will again assign it to the developer.
10. On daily basis I will push my code to my branch.
11. Once the user story completed I will push it to the branch and give pull request to the branch.
12. Also I will prepare defect log and execution result with screenshot and then I will mail those reports to client on weekly basis.

DAILY ACTIVITIES

1. I will start my day by checking the mails to know what all the tasks assigned to me.
2. Then I will be attending daily standup meeting with my team and discuss what had done what I am going to do.
3. And then I will take a new pull from the master to check the updated code is there and I will start doing the scripting and execution.
4. In that for some test cases I will be doing manual testing also.
5. If any test case failed I will raise defect and I will be following the defect until defect is fixed.
6. Once my work is done at the end of the day I will push up code to the branch.
7. And once my user story is completed I will give pull request to add my code to master.
8. Once the day is completed I will send task mail to my manager.

CHALLENGES FACED

Initially When I was joined as a Fresher, I was not familiar with domain, so I struggled lot in preparing test cases. Later when I was familiar in Domain Knowledge, I can able to write test cases.

Then When I moved to automation tester I faced difficulty in handling dynamic web elements. Later I upgraded myself and handled dynamic web elements using Parent, Child, Ancestor and Sibling Concepts.

Also I have faced conflicts while giving pull request in GIT. Then I learned to update codes in eclipse by giving git pull command which will give updated codes. I can avoid conflicts by giving pull request after merging updated code with my code.

The major challenge I have faced in my project is time out issue. Due to network issue or delay loading many of the Test execution got failed by throwing No such element exception. I avoided those exceptions by using waits concept here. I started using implicit wait as well as Explicit wait .

Initially we used driver.navigate().to() method for launching url. At the time the execution continues even the page isn't loaded completely. So after we started using driver.get() method for launching url.

JENKINS

1. We are using Jenkins as continuous integration and continuous deployment tool.
2. Usually we use Jenkins for regression testing .
3. Initially we will add new item in the tool where we will integrate our git repository url and configure as project in Jenkins.
4. As we are using Maven framework we are giving it as Maven in Build tool.
5. Also we have schedule our regression on weekly basis with the help of CRON expressions.
6. CRON expression is nothing but five star Expression
7. First star indicates minutes, second star indicates hours, Third star indicates date, fourth star indicates month and fifth star indicates day of the week.
8. Usually we have scheduled regression on every Thursday 10pm and its CRON expression will be
    i. 22 * * * 4
9. Also we can run manually at any time by giving build giving option.

PAGE OBJECT MODEL[POM]

The design approach we are using in our project is Page Object Model. We have implemented encapsulation in POM by using POJO class. Pojo class is used to store locators in page wise. We are having many number of pages in our project. For each page we will maintain locator in different Pojo class.

Here in the pojo class we will declare or web element as private and using getters we can access our private webelements in other class.

In pojo class we have a constructor. Because we are using Pagefactory class in the constructor. The main purpose of the Page factory is to capture bulk of webelements. Also we have annotations like @Findby, @Findbys, @FindAll.

@FindBy it is used to declare only one locator for one webelement and @FindBys to declarethe more thanone locator for one webelement it should be work on AND basis and we have another annotations like @FindAll it work on OR operator basis if one locator is true means it will find that webElement.

Then we are using initElement method in the Pagefactory class which is used to re-initialize the webelement to avoid stale element reference exception.

Framework Explaination

We have implemented our cucumber framework in the maven. First we will create Hooks class where we use @Before and @After annotations which will execute before and after each scenario, we used to have launch browser script in @Before and quite browser script in @After.

Then we will create base class where we maintain all resuable codes like sendkeys, click, scroll-up, scroll-down etc.

Then we will create feature file in src/test/resources. In the feature file we will add Feature and Scenario using Gherkin Keywords like Given, when, And, Then, Feature, Scenario, Scenario outline, Examples, Background etc.Once the feature file is created we will create TestRunner class where we use two annotaion such as @Runwith and @cucumberoptions.

The cucumber options we are using are Feature, Glue, Monochrome, dryRun, Tags, Plugin, snippets, strict etc.

Features Option is used to connect feature file with test runner class and glue option is used to connect test runner class with step definition.

Once we execute the test Runner class snippets will be generated in the console which is called skeleton script. Then we copy it and paste it in the stepdefintion class and we will write our scripting part.

We integrate this stepdefintion class with baseclass also with POJO class. . Pojo class is used to store locators in page wise. We are having many number of pages in our project. For each page we will maintain locator in different Pojo class.

Here in the pojo class we will declare or web element as private and using getters we can access our private webelements in other class.

In pojo class we have a constructor. Because we are using Pagefactory class in the constructor. The main purpose of the Page factory is to capture bulk of webelements.

Also we have annotations like @Findby, @Findbys, @FindAll. @FindBy it is used to declare only one locator for one webelement and @FindBys to declare more than one locator for one webelement it should be AND basis and we have another annotations like @FindAll it work on OR operator basis if one locator is true means it will find that webElement.

Then we are using initElement method in the Pagefactory class which is used to re-initialize the webelement to avoid stale element reference exception.

And at last we Create a separate class for Reporting .There we use generateReport method from ReportBuilder class and we will give file path for storing our Report. It will create a detailed reported as HTML File. Where as it shows the graphical representation of our execution results.

AIRLINE DOMAIN

I have worked in airline domain. _____ is the leading _____ airlines with headquarters in _____. In our project there are modules like Booking, Manage Module, Login Module, Check-in, Business module etc. Here I have worked in most of the module and I have majorly worked in the Booking module. In booking module, we are having pages such as Home page, Flight details page. Initially I will be in the home page. In homepage I can enter source, destination, date of travel, mode of travel (either one way or round trip), and departure date for one way trip, return date if we have round trip and also we have number of passenger input.

Once I clicked the Book now button the page will be navigated to flight details page. In the flight details page Flight name with number, departure and arrival time, travel duration, and price will be displayed. Price will be varied based on the cabin. Cabins we are having are Comfort, Express and Economy. After selecting the cabin the amount will be added in the billed amount on the left side.

After selecting the flight the page will be navigated to the Passenger details page. In that page I should enter First name, last name, phone number, email address etc.

Then I will be moving to Seat selection page. Here I can select the seat based on the comfort. Following to the page we have luggage page.

In the luggage page we can add our extra luggages including pets, special equipments. Also we can add special assistance, add meal etc.

On continuing the last page will be Payment page. I can either pay either by credit card or offline banking.

Maven

Maven is the build management tool as well as Project management tool. Because maven gives good folder structure and also no need to download jar files here. We will add dependency files in pom.xml file.

With the help of maven goal, that is clean- install- run all the required jar files will be automatically downloaded and in case ok any version update we can change the version number in the dependency file is alone needed.

Also maven gives good folder structure such as src/test/Java where we will write our scripts src/test/resources where we will maintain our resources files like features files, reports etc and Dependency libraries, and then the execution starts from pom.xml file.

In the pom.xml file only we are going to add dependencies like Selenium Java, Apache Poi, Cucumber Java, Cucumber Junit, JVM reporting.

GIT

We use GIT as a source code management tool. In Git only we are maintaining our source code in remote repository. On daily basis the day begins with pulling updated codes from the master using git pull command. This is because to avoid conflict while giving pull request to the master from branch. Once my collegue code was merged before my pull request, when I try to give pull request without updating code with latest my update from the master I will get conflict. So we have to use git pull command. And then we will get the updated code we can reallign the updated code with my code and then I will push the code with git add ., Then to commit the code we use git commit -m "update message", and then we can give git push origin master command to push the code to the branch. And then once the user story is completed we can give pull request.

POM

Page Object Model is the design pattern we are using in our project. We have implemented Page object Model in pojo classe using Encapsulation. Pojo class is used to store locators in page wise. We are having many number of pages in our project. For each page we will maintain locator in different Pojo class.

Here in the pojo class we will declare or web element as private and using getters we can access our private webelements in other class.

In pojo class we have a constructor. Because we are using Pagefactory class in the constructor. The main purpose of the Page factory is to capture bulk of webelements.

Also we have annotations like @Findby, @Findbys, @FindAll. @FindBy it is used to declare only one locator for one webelement and @FindBys to declare more than one locator for one webelement it should be AND basis and we have another annotations like @FindAll it work on OR operator basis if one locator is true means it will find that webElement.

Then we are using initElement method in the Pagefactory class which is used to re-initialize the webelement to avoid stale element reference exception.