**Name – Mahir Khurana**

**Mobile – 9953746574**

**Mail –** [**17mahirkhurana@gmail.com**](mailto:17mahirkhurana@gmail.com)

**ASSIGNMENT**

**Q1)** **Test Scenarios Vs Test Cases Vs Test Plan**

**Test Scenario :-**

~ Test Scenario is a High level document of all the customer business workflow according to the Customer.

~ It is a one line Scenario to test the Feature.

~ We Can write Test Scenarios by looking into the Requirement Document.

~ By looking into the Test Scenarios, we can’t Test any project until we have great product knowledge.

**Test Case:-**

~ Test Case is a Detailed Document of the scenarios which helps us to Test the Application.

~ We can write as many Test Cases as we can think of to test the Feature.

~ We can write Test cases by looking into both Requirement Document & Test Scenario.

~ By looking into the Test Cases, we can Test any project whether we have Product knowledge or not.

**Test Plan :-**

~ Test Plan is a document which derives the future testing activities.

~ It is generally prepared by Test Lead/ Sr Test Engineer/ Test Manager.

~ It is like a Roadmap of whole testing activities that we are planning to conduct to test the Software.

~ It has got different sections like:-

Objective, Effort Estimation, Scope, Assumption, Risk, Mitigation plan, Testing Methodology, Entry & Exit Criteria, Test Stop criteria, etc.

Q2) **Test Case Template (Format)**

**HEADER:-**

1. Test Case Name
2. Requirement Number
3. Test data
4. Pre-Condition
5. Test Case Type
6. Severity
7. Brief Description

**Body:-**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S.No** | **Description** | **Test Steps** | **Input** | **Expected Output** | **Actual Output** | **Status** | **Comments** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

**Footer:-**

1. Author
2. Reviewer
3. Approved Date
4. Approved By

Q3)  **Software Testing Techniques**

These are the Techniques which are used by test Engineers while writing the test Cases in order to improve the Test case Coverage.

There are 3 Types of Test Case Design Techniques:-

1. Error Guessing
2. Equivalence Class Partition
3. Boundary Value Analysis

1) **Error Guessing**

In this technique, all the Test Engineers guess all the possible errors and then based on that, they derive Test Scenarios & write the Test Cases.

Test Engineers guess errors based on :-

1. Requirement
2. Intuition
3. Experience

2) **Equivalence Class Partition :-**

There are 2 Types of Equivalence Class Partition :-

i) Pressman Rules

ii) Practice Method

i) **Pressman Rules**

1. If the Input is in the range of values, then we have to design the test case for 1 Valid & 2 Invalid Inputs.
2. If the Input is in a set of values, then we have to design the test case for 1 Valid & 2 Invalid Inputs.
3. If the input is in Boolean, then we have to design the test Cases for both True & False values.

ii) **Practice Method**

If the Input is in the range of values, then we have to divide the range into Equivalent parts, and then test for all the values & also test at least 2 Invalid values. This is called the Practice Method.

3) **Boundary Value Analysis**

If the input is in the range of values between A to B then we have to Design the Test Cases for A, A+1, A-1 and B, B+1, B-1.

Q4) **API Method**

As of now, I am not familiar with this topic. But I am eager to learn it and expand my skill set.

Q5) **Banking Domain System Scenarios**

1. Verify that the User is able to Login successfully by Entering Valid Credentials.
2. Verify that the User is not able to login by entering either wrong Email/Mobile or wrong Password.
3. Verify that the User is able to access every feature of the Application by logging into the Application.
4. Verify that the User is able to Check his/her Balance by entering Valid Password/PIN.
5. Verify that the user is able to Transfer the Amount to the Account within that same Bank by entering Account Number, IFSC code, OTP, Transaction PIN, etc.
6. Verify that the user is able to Transfer the Amount to the Account in some other Bank by entering Account Number, IFSC code, OTP, Transaction PIN, etc.
7. Verify that the User is not able to Transfer the Amount when he/she enters Same Details in both Remitter/Payer & Beneficiary/Payee.
8. Verify that No Amount is deducted when we Transferred the Amount to the Bank Account which does not exist.
9. Verify that No Amount is deducted when we Transferred the Amount to the Bank Account with Incorrect details. Ex- Account No, IFSC Code, etc.
10. Verify that the User is not able to Transfer Amount which exceeds their Account Balance.
11. Verify That the User is able to Logout Successfully without any hassle.
12. Verify That the User is not able to Transfer Zero Balance (0) or Negative Amount.

Q6) **Automation Frameworks**

I haven't covered this topic yet, so I haven't had the chance to study it. However, I'm eager to learn it soon, as I'm nearing completion of my study in Selenium.

**Q7) Life Cycle of a Bug:-**

Life Cycle of a bug defines the Stages that happen in the whole Life Cycle of a Bug ie- Defect.

It includes many stages, some of them are:-

1. New/Open – When the bug is discovered for the first time, its stage is New, and when we are working on it its Stage is Open.

This status is Given by Test engineer.

1. Assigned – When the Bug is found, the Test Engineer will prepare Defect Report, and send it to Development Lead.

Development Lead will identify the Development Engineer who has done the Mistake and will assign work to him.

Development Lead will put the Status as Assigned.

1. Fixed – When the Bug is fixed, the Development Engineer will put the Status as Fixed.
2. Closed – When the Bug is fixed and is also Retested by the Test Engineer,

Test Engineer will put the Status as Closed.

1. Re-Open/ Not Fixed – While doing Retesting, If the Test Engineer finds that the Bug is not yet fixed properly, He will put the Status as Re-Open.
2. Defect Can’t be Fixed – If the Test Engineer finds the defect in the Root of the Product, and if it is a minor Defect.

And,

If the Cost of fixing the Defect is more than the Cost of the Defect,

In these 2 cases, the Development Engineer will give the Status as Defect can’t be fixed.

1. Duplicate – If the Test Engineer tracks and Reports a Bug, and if the same bug is already reported by some other Test Engineer, then the Developer will give the status as Duplicate.
2. Issue not reproducible – If the Test Engineer is able to see the defect, but Developer is not able to see the same Defect, then the status will be Issue not Reproducible.
3. Deferred – If the Development Engineer is accepting that this is a Defect, but they want to fix it a little late and are postponing it for future, In this case, They put the status as Deferred.
4. Reject – If the Test Engineer finds a defect and sends it to the Development Engineer, the Development Engineer is saying that it is not a Defect, it is a feature. They put the status as Reject.
5. RFE- If a Test Engineer finds a defect in the feature which is not a part of the Requirement, then it is called RFE(Request for Enhancement).

**Severity –** Severity is defined as the Impact of the Defect on the Customer Business.

It is divided into 4 Categories:-

1. Blocker – When TE is 100% sure that this defect is Affecting my Customer Business for sure and is also blocking the TE from Testing Features further. It is Blocker Defect.
2. Critical – When TE is 100% sure that this defect is Affecting my Customer Business for sure but is not blocking the TE from Testing Features further. Its is Critical defect.
3. Major – If TE is not sure that, How this defect may affect my Customer business workflow, such defects are called Major defects.
4. Minor – When TE is 100% sure that this defect is Not Affecting my Customer Business for sure. Such Defects are Minor Defects.

**Priority –** It is the Importance given to the Defect to Fix the Defect.

It is divided in 3 Categories:-

P1 – Must be fixed on urgent basis.

P2 – Can be Fixed within some cycles or within a release.

P3 – Can be fixed in the next upcoming release.

**High Severity & Low Priority Example-**

When Help or Privacy Policy Page is Crashing any website, we can say this defect is of High Severity & Low Priority.

Severity (High) – Because it is crashing, and not allowing the user to access the page.

Priority (Low) – Because it is accessed by very few People and is not impacting the Primary Functionality of the website.

**Low Severity & High Priority Example –**

In an Ecommerce Website, (Add To Cart) Button is slightly misaligned,

We can say it is of Low Severity & High Priority.

Severity (Low) – Because it is not affecting the Actual use of the Button.

Priority (High) – Because it is affecting the Usability of the website.

**Q8)** **Smoke Testing -** 1)Testing the Basic or Critical features of an Application before we do Thorough Testing is called Smoke Testing.

2) It is also called as Positive Testing or Build Verification testing or Health Check-up of the Product.

3) It is done by both Development Engineer & Test Engineer.

4) In Smoke Testing, We document Test Scenarios and test Cases.

5) In Smoke Testing, We Go for Automation.

**Sanity Testing -** 1) Testing the specific features after the changes have been made to the Product.

2) It is both Positive and Negative Testing as it is deep & Narrow Testing.

3) It is done only by the Test Engineers.

4) In Sanity Testing, we do not document the Test Scenarios and Test Cases.

5) In Sanity Testing, We do not go for Automation.

**Regression Testing -** 1) Testing the unchanged features to make sure that it is not affected or broken by the changes (Adding, Modifying, removing or bug fixing) is known as regression testing.

2) Re-Execution of same test cases in different test cycles or sprints or releases to make sure that changes are not introducing any new defects in the Unchanged Features.

3) It is done only by Test Engineers.

4) In regression Testing, We document the Test Scenarios and Test Cases.

5) For Regression Testing, we generally go for Automation.

6) We have further 3 types of regression Testing, ie - Unit Regression testing, Regional regression Testing, Full regression testing.

7) In regression testing, an Indirectly Whole Application will be tested in order to test the Changes and the Impacted Areas.

**Q9) Selenium Task on Demoblaze :-**

public class Assignment {

public static void main(String[] args) throws InterruptedException {

//Opening the Browser

WebDriver driver = new ChromeDriver();

//Maximize

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

//Enter url

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

Thread.sleep(3000);

//Adding Product 1

driver.findElement(By.xpath("//a[text()='Samsung galaxy s6']")).click();

Thread.sleep(3000);

driver.findElement(By.xpath("//a[@onclick='addToCart(1)']")).click();

Thread.sleep(3000);

Alert a = driver.switchTo().alert();

a.accept();

driver.findElement(By.xpath("//a[contains(text(),'Home ')]")).click();

Thread.sleep(3000);

//Adding Product 2

driver.findElement(By.xpath("//a[text()='Nokia lumia 1520']")).click();

Thread.sleep(3000);

driver.findElement(By.xpath("//a[@onclick='addToCart(2)']")).click();

Thread.sleep(3000);

Alert b = driver.switchTo().alert();

b.accept();

driver.findElement(By.xpath("//a[contains(text(),'Home ')]")).click();

Thread.sleep(3000);

//Adding Product 3

driver.findElement(By.xpath("//a[text()='Sony vaio i5']")).click();

Thread.sleep(3000);

driver.findElement(By.xpath("//a[@onclick='addToCart(8)']")).click();

Thread.sleep(3000);

Alert c = driver.switchTo().alert();

c.accept();

//Going Inside Cart

driver.findElement(By.xpath("//a[text()='Cart']")).click();

Thread.sleep(3000);

//Verifying total price

driver.findElement(By.id("totalp"));

Thread.sleep(3000);

//Removing 1st added Product from Cart

driver.findElement(By.xpath("(//a[text()='Delete'])[3]")).click();

Thread.sleep(3000);

//Verifying total price

driver.findElement(By.id("totalp"));

Thread.sleep(3000);

//Closing Browser

driver.close();

}

}

**Q10)** **Absolute XPath Vs Relative XPath**

**Absolute XPath - ~** It is represented by a single slash (/).

~ To find the Web-Element, we have to come from the Root of the Path.

~ Root is the HTML Tag.

~ It specifies the complete path of the web element starting from the Root.

~ It is Too Lengthy to write.

~ Test Engineer will get confused.

~ If any path is getting changed, At that time, you will get -> NoSuchElementException.

**Relative XPath -** ~ It is represented by Double Slash (//).

~ To find the web Element, we are using the Direct Path in an HTML.

~ It does not specify the Complete Path, rather specifies the path with respect to other element.

~ It is relatively short & easy to use.

~ Relative Path is having further 5 Types:-

1. X-Path by Attribute.
2. X-Path by text() Function.
3. X-Path by contains() Function.
4. Independent & Dependent X-Path.
5. X-Path by Group of Index.