**`------------------------------------------------------------------**

**Manual Testing by Vaibhav Sir**

**---------------------------------------------------------------------**

**Contents Page No.**

1. Test Documentation hierarchy. 01
2. Software Testing Life Cycle. 04
3. Test Case Design
4. Test Case Review
5. Traceability Matrix
6. Defect Life Cycle
7. Reproducible, Duplicate, Leakage of Defect
8. Test Report
9. Testing Principles 17

# Test Documentation –

**Q. What is your organization Test documentation?**

* Test Document hierarchy.

Quality control – QC/

Testing Head -TH

Company Level document

**Test Plan**

**Test Methodology**

**Test Strategy**

**Test Policy**

Test Strategist – TS & PM

Project manager – PM

TRM

Team Lead – TL

**Test Scenario/ Case**

Team Lead – TL

Tester – Tester

Project Level document

**Test Procedure/Design**

**Test Script/ Execution (Test Proof)**

**Defect Report**

**Test Summary Report**

**Final Report/ Test closer Report**

**Test Policy**

* Test policy defines the **Objective** of the Project.
* Test Policy is decided by **Test Head**
* Test Policy is **the company level document.**

**Test Strategy.**

* Test Strategy defines as which approach is going to apply to fulfill the project objective.
* Test strategy is a Company level document.
* E.g. which language is going used like Java, Python, JavaScript etc, Which Tools used Eclipse, JIRA, TestComplete. Automation Testing Tool
* Test Strategy is defined by Test Strategist.

**Test Methodology (PM)**

* Test methodology defines which environment which we are going to us for fulfill the strategy.
* **Project Manager** decides the test methodology.
* Test Methodology documents are Project level documents.
* PM will prepare TRM Test Responsibility Matrix
* In the **TRM it defines the development stages are mapped with the testing factors.**
* While Preparing TRM generally considers
  + Project Requirements.
  + Project Scope.
  + Risk in Project.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Development Stage  Testing Factor | Info Gathering / Analysis | Design | Coding | Testing | Maintenance |
| System and Functionality Testing. | No | No | YES | YES | YES |
| Security Testing | NO | No | No | YES | YES |
| Performance | NO | No | No | YES | YES |
| Usability Testing | NO | NO | YES | YES | YES |

**Test Plan**

* Test plan consist of **Resource Allocation, Job Allocation** ( 4 QA , 2 QA – Manual, 1QA – Manual/Automation – Lead : Automation)
* Test plan consist of the Estimation
* Test Plan documents are Project level document
* Test Plan document is prepared by **Team Lead.**

**Test Scenario/ Case or Test Case Design**

* QA will identify the testing scenarios.
* Based on the scenarios QA will design (Write) test cases.
* These documents are project level documents.

**Test Case Execution and Test Proof and Defect Logging.**

* In the test case execution; QA execute all the test cases designed as per the user story.
* While performing or While executing the test cases QA prepares the Test Proof (QA Captures the Screenshot or Videos if required)
* While executing the test cases if any of test cases fail or QA found any defect then QA will log a defect.

**Test Summary Report / Test Closure Report**

* Test Lead will create a Test Summary Report and Test Closure Report
* It consist of **No of Test Cases Designed, No Test Cases Executed, No Test Cases Pass, No Test Cases Fail, No Test Cases Skipped.**
* Test Summary Report and Test Closure reports are Project level document.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| QA | **No of Test Cases Designed** | **No Test Cases Executed** | PASS | FAIL | SKIPPED | US Assigned |
| Sachin | 50 | 50 | 49 | 01 | 0 | 4 |
| Prajakta | 25 | 25 |  |  |  |  |
| Nitin | 20 | 20 |  |  |  |  |
| Seema | 05 | 05 |  |  |  |  |

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

**Software Testing Life Cycle (STLC)**

**BRS**

Testing

Development

**SRS/FRS/CRS**

**Design Test Initiation Stage**

**Coding Test Plane**

**STLC**

**Unit Testing Test Case Design**

**SDLC**

**Integration Testing Test Case Execution & Closer**

**(Install Build)**

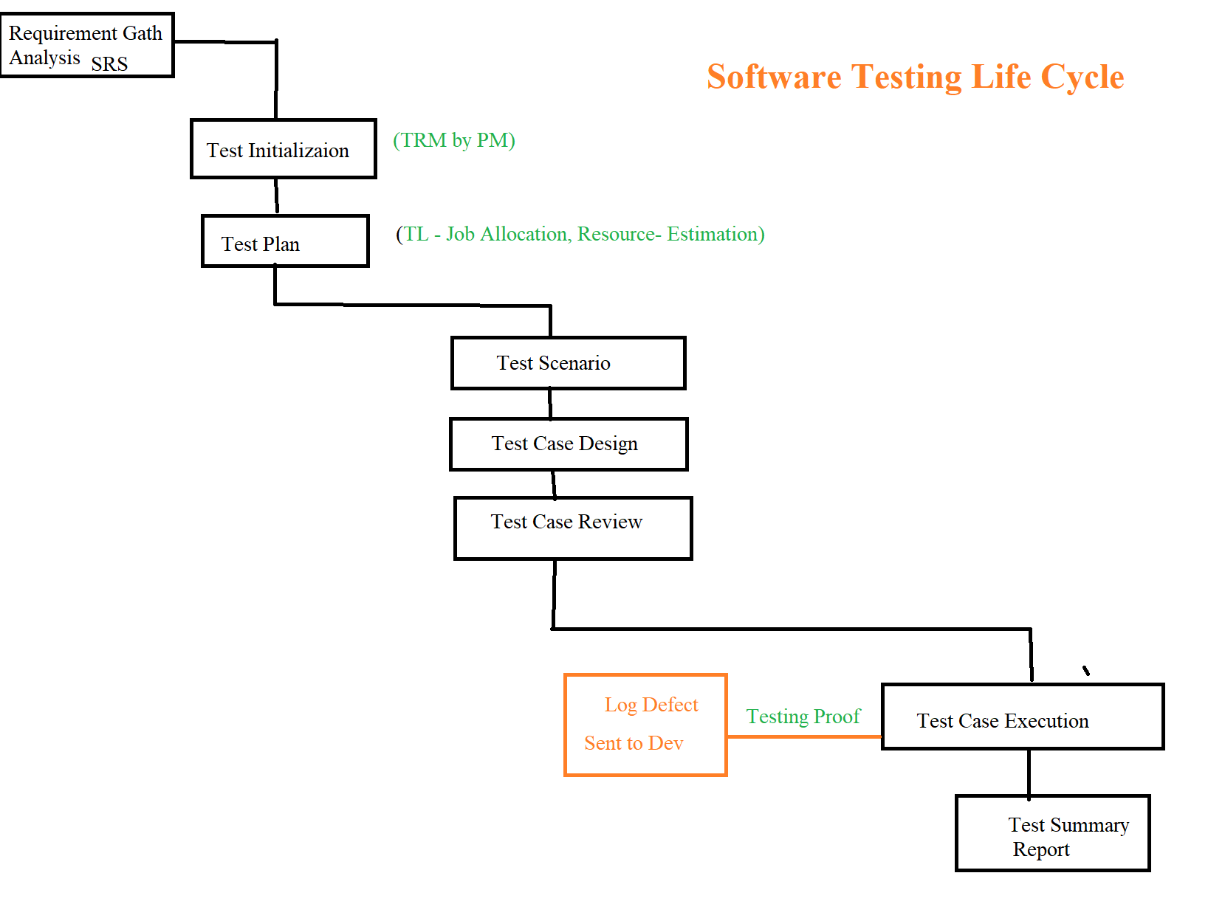
**Level 0 Sanity/ Smoke Testing (**Check theStability of build)

**Level 1 BBT/ System & function testing (**Intern & External**)**

**(**If found **defect** sent to developer) (Inform throw **JIRA**)

**Level 2 Retesting / Regression Testing on Modified built.**

**Level 3 Final Regression Testing.**



**Test Initialize. (PM - TRM)**

* In the Test Initialization Test Responsibility Matrix TRM document is prepared by PM.
* Once TRM document is finalizes it send to the Team (QA and Dev)
* In the TRM, Dev Stages are mapped with the testing factors.

**Test Plan.**

* The purpose of the Test plan is to Plan the Sprint from Starting Date to ending date of the Sprint.
* E.g. Sprint Start Date – 28/11/2022 and Sprint End Date – 12/12/2022.
* 10 Days. – 4 QA - (Work?? 16 US, Automation – 15 TC Automate) – Work Distribute.
* Test Plan is prepared by Team Lead (TL) QA.
* It consist of No of QA available, Environment, Work Allocation and Estimation.

Resource Allocation – JOB Allocation – Estimation

What to Test – When to Test – How to Test

**Test Scenario**

* When any user story assigned to QA for the testing; QA first understand the User story (Requirement)
* QA will find the different ways to test the application.
* Test Scenarios means simply finding the number of ways I can perform/Test the functionality.
* Test cases are depends upon the Test Scenarios.

**Test Case Design**

* Depends upon the functionality and Test Scenarios; Test Cases are Designed.
* For Design Test Cases – **Excel Sheets** / (X-ray Tool Configure with JIRA) prefers.
* When I Complete the Test Case Design part then this Test Cases are attached to the User Story.

**Test Case Execution.**

* When Test cases are reviewed and Build sent to you after Dev complete the Coding part; QA starts Test Case Execution.
* Test Case Execution means to Perform the Activity (Designed Test Cases) on the UI of the Application and Confirm the Testing Steps mentioned in the Test Cases matched to the Result.
* While Test Case Execution if any Test Case fails then QA has to Log a Bug.

**Test Proof**

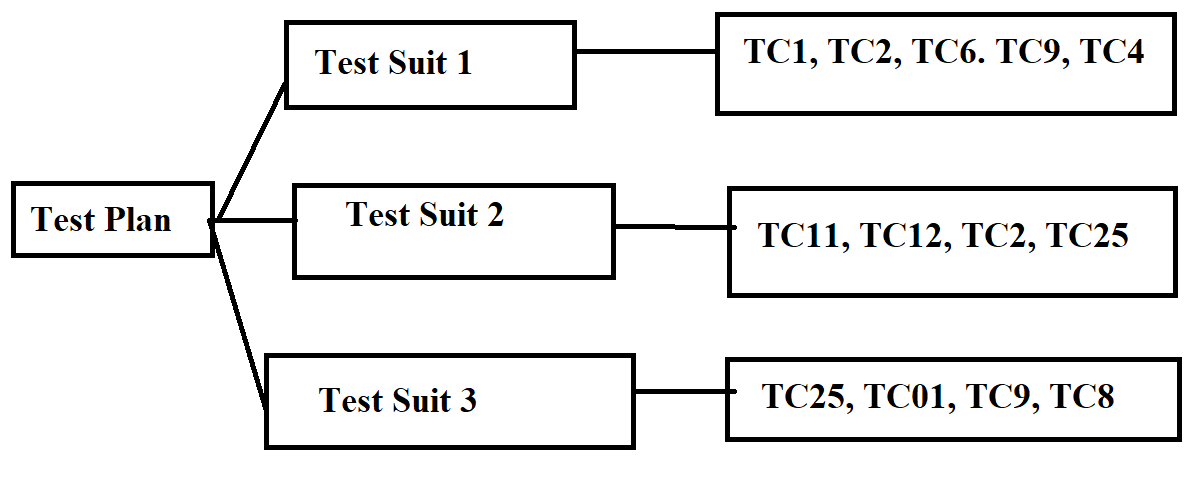
* While Test Case Execution; If Functionality is working as Intended then QA captures the Screenshots/Videos.
* Captured Screenshots/videos stores in particular/specific location that is referred as Test Proof.
* Usually Word Document is used to stored screenshots.

**Test Summary Report / Test Closure Report**

* Test Lead will create a Test Summary Report and Test Closure Report
* It consist of **No of Test Cases Designed, No Test Cases Executed, No Test Cases Pass, No Test Cases Fail, No Test Cases Skipped.**
* Test Summary Report and Test Closure reports are Project level document.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| QA | **No of Test Cases Designed** | **No Test Cases Executed** | PASS | FAIL | SKIPPED | US Assigned |
| Sachin | 50 | 50 | 49 | 01 | 0 | 4 |
| Prajakta | 25 | 25 |  |  |  |  |
| Nitin | 20 | 20 |  |  |  |  |
| Seema | 05 | 05 |  |  |  |  |

**Test Suit / Test Bed / Test Cases Bunch**



**Test Plan Document.**

1. **Test Plan ID**

* ID of the document (Test Plan) E.g - PID-37, AN-54, AN-55

1. **Testing Items**

* Items to be tested i.e User stories/Bugs that will consider in current sprint.

1. **Features to Test**

* Features that need to be Test ( E.g If there are 6 Providers in Recharge Module; Functionality implemented for only three Providers)

1. **Features Not to Test**

* Those feature will not Implement in Current sprint but part of the same module.

1. **Pass/ fail Criteria**

* Based on the Acceptance criteria; User stories – PASS / Fails.

1. **Testing Environment**

* The Env in which QA have to test functionality. **(QA Has access of only SIT Env)**

1. **Road Blocker / Suspension Criteria / Shows Topper Bug**

* Major bugs because of that Testing blocked or stopped.

1. **Test Deliverables.**

* Before the Testing; QA Prepares the Test Deliverables on which QA Performs the Testing. (Test Cases should be designed before the testing).

1. **Role and Responsibility**

* Roles and Responsibility of the QA ( Manual / Automation / Hybrid)
* Amit - TL – Test Automate -
* Mahesh – Hybrid – Automation + Manual
* Akash – Manual.

1. **Any QA Needs Training.**

* Any QA Need any type Training – Training (Client Business / Technology) (JIRA Tool Use, HPALM/Dev Azure - Use)

1. **Risk in Testing**

* Lack of Knowledge QA fail to Test efficiently etc factors / What if testing not completed within Time.

1. **Signature/Approval.**

* Test Plan prepared by TL is whether correct and complete is approved by Scrum Master.

**---------------------------------------------------------------------------------------------------------------------**

**Test Case Format**

1. **Test Case ID**.
2. Priority/Severity.
3. Reference.
4. Test Scenario.
5. Pre-requisite.
6. **Test Case Title / Test Case Heading.**
7. **Test Data.**
8. **Testing Steps.**
9. **Expected Result.**
10. **Actual Result.**
11. **PASS/Fail.**
12. Comments.
13. Regression Suit.

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

**User Story:**

* **User Story** is derived from the SRS Document / Clients Requirement.
* **User Story is** designed by **Product Owner** / Scrum Master.
* **User Story** defines the specific requirement/information about the functionality.
* **User Story** consist of the factors – Description, Acceptance Criteria, and Screenshots/Videos.

**User Story** MI-996 , US – 115, UP-1176

**Description**

For the Login; Username and Password are the mandatory fields. User can login by using Registered Phone Number / Registered Email Address.

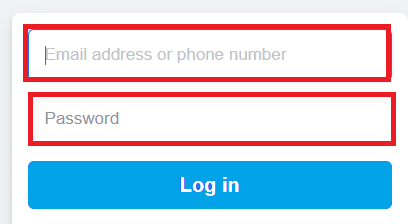
Email Id / Username. Username should not be case sensitive

**Acceptance Criteria**

|  |  |
| --- | --- |
| **1.3.1** | **ID – valid domain name (gmail.com, yahoo, rediffmail, Hotmail, outlook etc)** |
| **1.3.2** | **Username should be anything but not repetitive. (4 character to max 60 char)** |
| **1.3.3** | **When Mobile Number used for the login must be registered one.** |
| **1.3.4** | **Password – Minimum 8 Characters and Maximum 16 Characters** |
| **1.3.5** | **Pass – Must be alphanumeric, contains at least one special character, and One Capital Letter**  **Trimuri@123** |
| **1.3.6** | **Textbox should be Focused when enter value in it.** |

**(Hover over – move cursor (Mouse) on any element)**

**Screenshot.**



**Login Functionality Testing.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Valid** | **EmailID / Mobile /  Username** | [**abc@gmail.com**](mailto:abc@gmail.com)  [**abc@rediffmail.com**](mailto:abc@rediffmail.com)  [**abc@hotmail.com**](mailto:abc@hotmail.com)  [**abc@yahoo.com**](mailto:abc@yahoo.com)  [**abc@outlook.com**](mailto:abc@outlook.com)  **8669076673**  **Onlyone.** | **Valid** | **Password**  **Pass@123**  **Pass@123**  **Pass@123**  **Pass@123**  **Pass@123**  **Pass@123** | **Expected Result  User Should logged In** |
| **Invalid** | **EmailID / Mobile /  Username** | **abc@gyagciamail.conhj**  123456789 | **Valid** | **Password** | **NOT LOG IN** |
| **Valid** | **EmailID / Mobile /  Username** |  | **Invalid** | **Password** | **NOT LOG IN** |
| **Invalid** | **EmailID / Mobile /  Username** | 12345678901 | **Invalid** | **Password** | **NOT LOG IN** |
| **Null** | **EmailID / Mobile /  Username** |  | **Valid** | **Password** | **NOT LOG IN** |
| **Null** | **EmailID / Mobile /  Username** |  | **Invalid** | **Password** | **NOT LOG IN** |
| **Valid** | **EmailID / Mobile /  Username** |  | **Null** | **Password** | **NOT LOG IN** |
| **Invalid** | **EmailID / Mobile /  Username** |  | **Null** | **Password** | **NOT LOG IN** |
| **NULL** | **EmailID / Mobile /  Username** |  | **NULL** | **Password** | **NOT LOG IN** |

**Test Case Review.**

**Review: Review is the process where documentation is to be tested.**

1. Self-Review.
2. **Peer-Review.**
3. Internal Review.
4. External Review.
5. **Self-Review.**

* QA will design the Test Cases and Same QA check/review Test cases.
* Self-review is done by QA who has strong knowledge about the Testing, How to Design Test Cases and have ample years of Exp.

1. **Peer Review. (In my Organization Peer Review Follows)**

* In the Peer review Designed Test Cases by you is checked/review by another **Senior QA. (Who has sufficient years of Exp.)**
* QA will Design Test cases in Excel Sheet 🡪 Sent the Excel sheet to Senior QA (Reviewer) 🡪 After the Review, He/she will send back the Excel File with Corrections/Comments if any.
* Comments/Modifications are added by the Reviewer in the Comment Column of the Test Case File

1. **Internal Review.**

* In the Internal Review Test Cases are checked by the **BA**.
* When QA finish the Design Test Case Part 🡪 QA Will send the Excel file of Test Cases to BA 🡪 Further BA schedule a Call / Check the Designed Test Cases and Send Back to you.
* Comments/Modifications are added by the BA in the Comment Column of the Test Case File

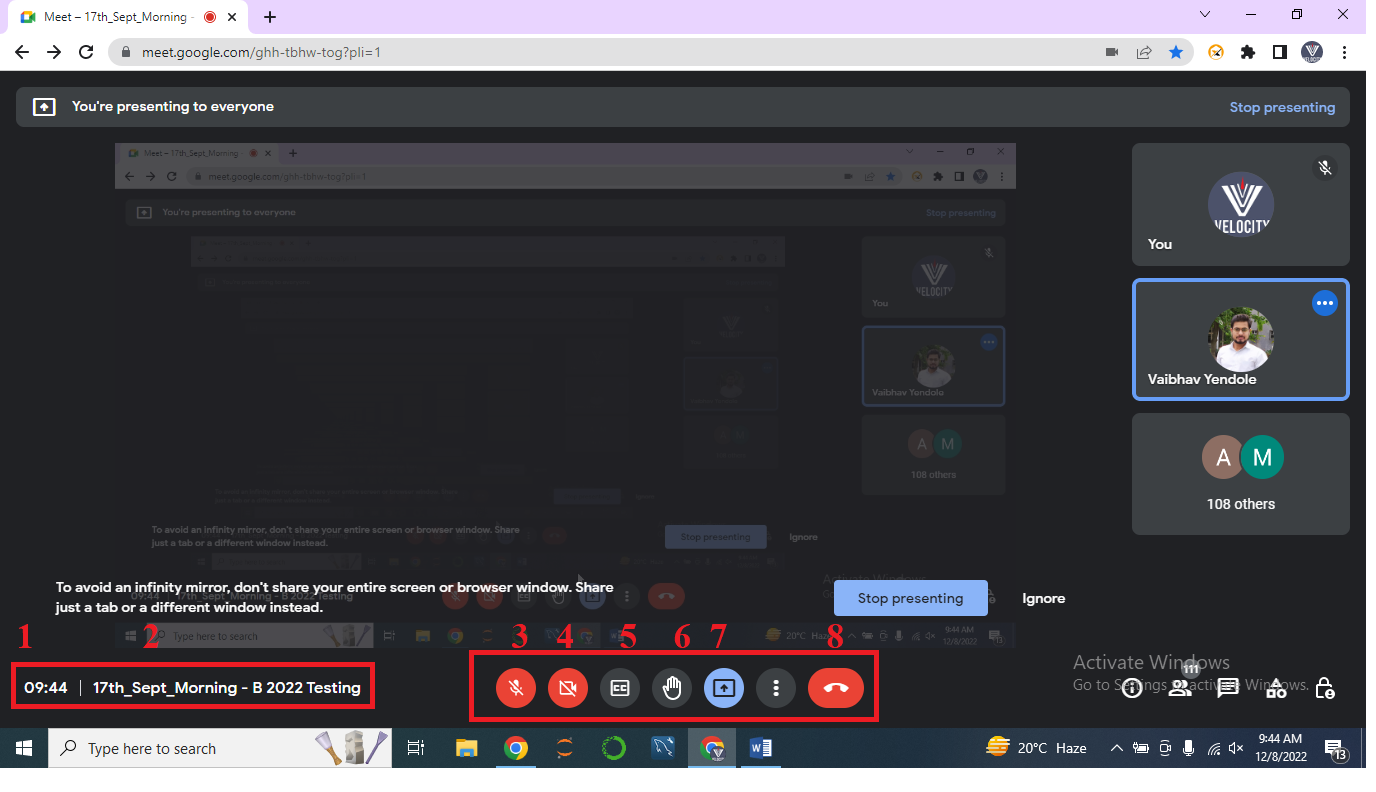
1. **External review.**

* In the External Review **UAT QA / Client** will review Designed Test Cases.
* When QA finish Design Test Cases 🡪 You will Inform to TL 🡪 TL will set up a Meeting with Client.

**What things Cover in the Test Case review.**

1. Designed Test Cases covered all the Functionality.
2. Designed Test Cases covered according to the Client’s Business Requirement.
3. TC follows the Set/Standard format.
4. TC doesn’t have duplicate test cases.
5. TC grammatically Correct.
6. Designed TC must be simple, easy to understand. (Complex keywords should be avoided)

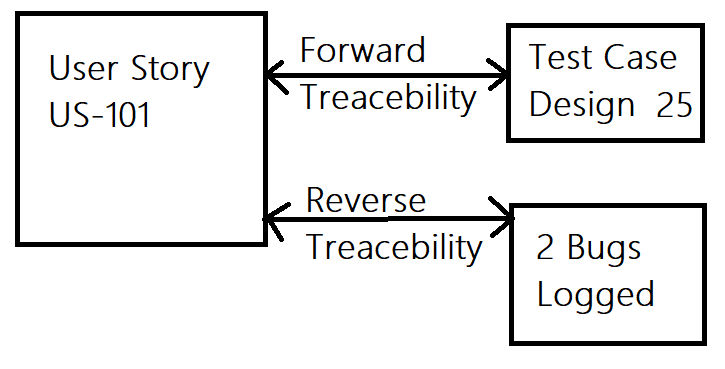
**Assignment No. 2**: **Design Test Cases for the Google Meet Functionality.**



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

**Traceability Matrix.**

* Traceability Matrix defines the Test Cases, Defects mapped with User Story.
* Traceability Matrix used for the Tracking.
* There are two types of Traceability Matrix.
  + Forward Traceability Matrix
  + Backward/Reverse Traceability Matrix.



1. **Forward Traceability Matrix.**

* In the Forward Traceability Matrix Test Cases are linked to the User Story.
* QA will design the Test Cases in the Excel Sheet further Excel Sheet is attached to the User Story.
* **QA will design Test Cases in JIRA using Tool X-ray further Number/Id of the Test Cases are attached to the User Story.**

1. **Backward/Reverse Traceability Matrix.**

* In the Backward Traceability Matrix Logged bugs are attached to the User Story.
* While Test Case Execution; if any Test Case fails then QA will log a bug related to the User Story.
* For the defect logging; JIRA tool is used in my Organization.

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

**Defect Life Cycle**

* While Test Case Execution; if any Test Case fails then QA will log a bug related to the User Story.
* Defect Life Cycle is Journey of a Bug from Beginning to Closing of the Bug.

1. NEW (**QA**) – While Testing the User Story; if any Bug found then QA will log the Bug and it is referred as **NEW** Bug.
2. OPEN (**Dev**) – When QA Assigned the Logged Bug to the Dev; Dev will analyzes the Bug (Whether is correct or not, whether it is reproducible) This Stage is referred as **OPEN**.

After Analysis of the Bug (Dev) 🡪

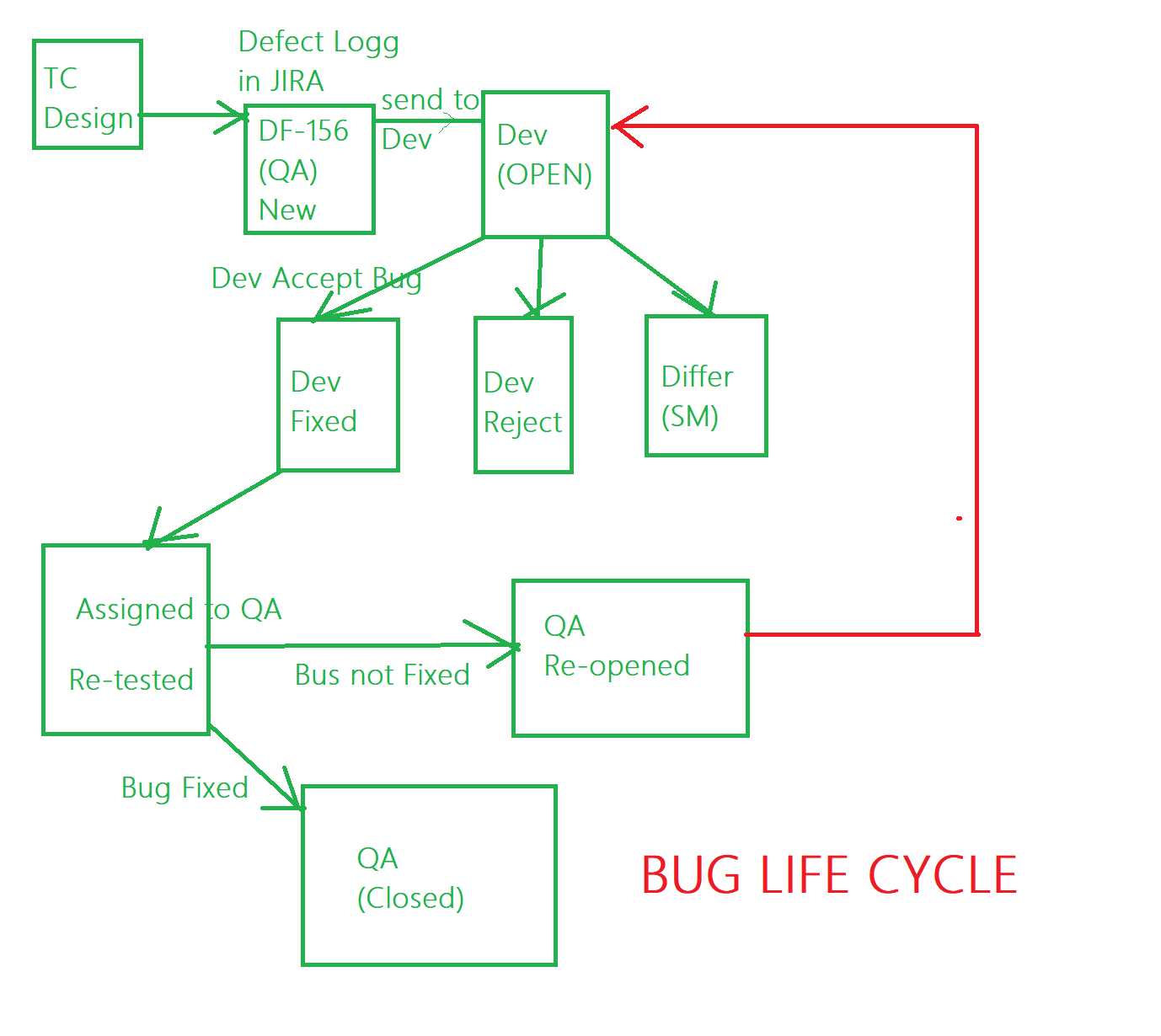
1. **Fixed** 🡪 When Dev Accept the Bug (Mistake in Code) 🡪 Dev will fix the Bug by change in Code.
2. **Reject** 🡪 If logged Bug by QA is not valid then **Dev** will **reject** the Bug.
3. **Differ** 🡪 If the logged Bug is Valid and Not possible for dev to fix within current sprint because of lack of time. In such condition Scrum Master has Right to differ a Bug/User story.
4. **Closed**.

* When Bug Fixed by Dev 🡪 Fixed Bug is assigned /send to **QA for Testing**. 🡪 When QA Test the Bug 🡪 If Bug is Fixed then It referred as **Closed**.

1. **Re-opened**

* When Bug Fixed by Dev 🡪 Fixed Bug is assigned /send to **QA for Testing**. 🡪 When QA Test the Bug but Bug is not still fixed 🡪 at this stage Bug is referred as Re-open.

<https://artoftesting.com/test-scenario-examples>



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

**Reproducible Defect** 🡪

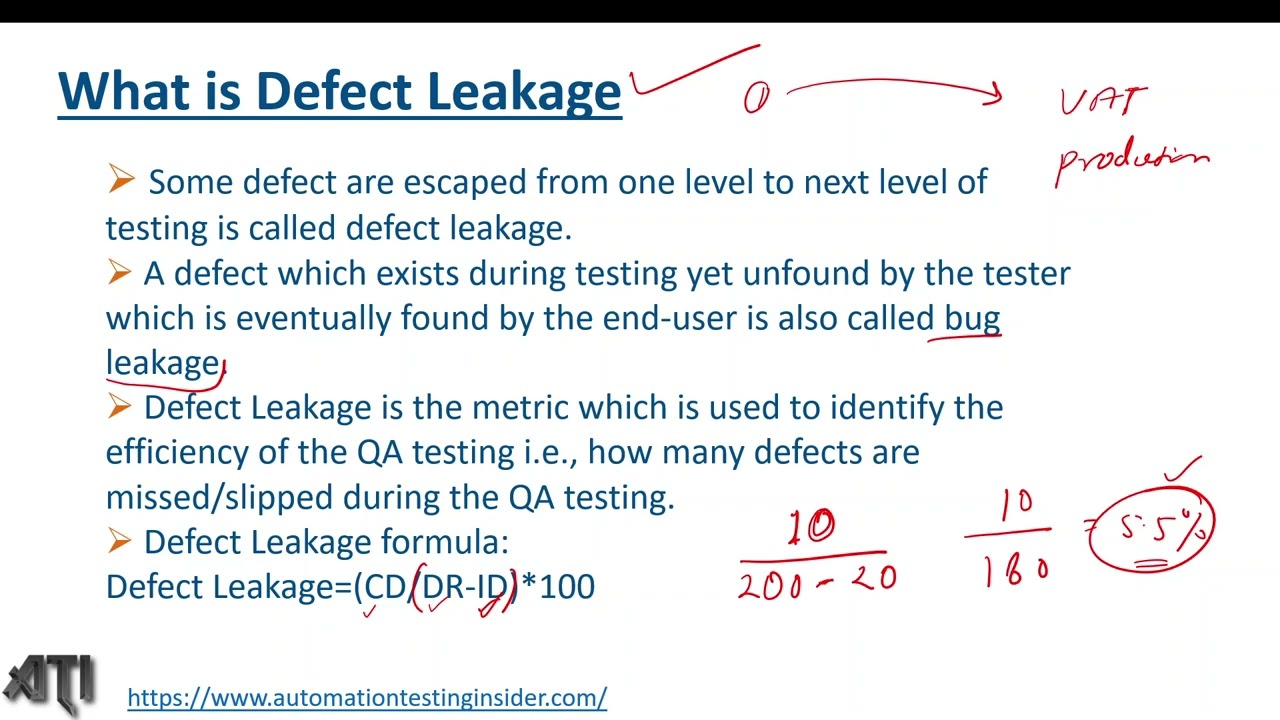
* The bug which produce again and again E.g If QA perform the Bug Steps then Bug Observed each and Every time 🡪 Suppose if QA Perform the Bug Reproduce Steps for 100 Times then Bug observed 100 Times.

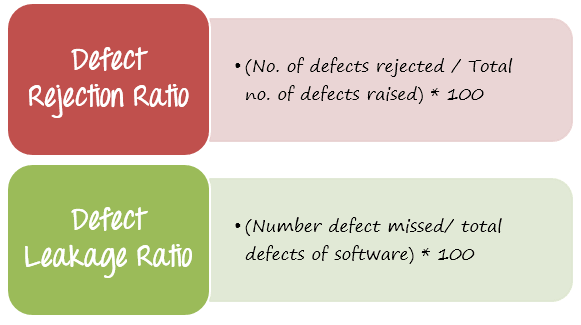
**Intermittent Defect** 🡪

* The defects which produces sometimes when we tried to reproduce defect. This kinds of bug observed Sometime and sometimes not observes.

**Defect Leakage.**

If the Bug is missed by QA in SIT Env and Found that Bugs in UAT Env 🡪 that is referred as Defect Leakage.





**Latent Defect.**

The bugs which are not identify by the Observations/Inspections 🡪 such kind of Bugs are referred as Latent Defect.

**Duplicate Bug.**

* Bug which are similar/duplicate.

**Test Report Formats.**

**Test Cases – Excel**

**Test Proof – Word File**

**Test Summary Report – Excel /** Word

Test Summary Report –**Prepare after every Sprint(2 Weeks).**

Test Closure Report – (**When one module is complete**)

* **TCD, TCE, Test Cases Skipped, TC PASS/Fail, No of Defect Logged.**

**WSR – Weekly Status Report.**

**Title – WSR Starting date and Ending Date –**

**Road Block**

* **Sanjay has Accees related Issue (VPN)**

**Report**

1. **Akshay –**
2. **Akaksh –**

**Goal for Next Week**

1. **Akshay –**
2. **Akash**

**Holiday -**

**Leave –**

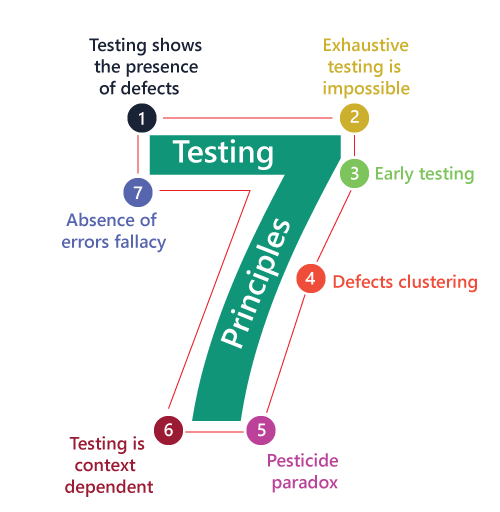
1. **Akshay – Next Week Leave on 01/01/2023 (One Day) ! Current week - NA**
2. **Akash**

**------------------------------------------------------------------------------------------------------------------**

**Defect Report**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Defect ID** | **Defect Description** | **Raised by** | **Assigned to** | **Status** | **Root Cause** |
|  |  |  |  |  |  |

**--------------------------------------------------------------------------------------------------------------------**

**Testing Principles.**

1. **Testing shows presence of defects**
2. **Exhaustive testing is not possible**
3. **Early testing.**
4. **Defect clustering (Pareto Principle).**
5. **Pesticide paradox.**
6. **Testing is context dependent.**
7. **Absence of errors fallacy.**

<https://www.guru99.com/software-testing-seven-principles.html>

