Retesting

- Re-testing, it is a part of functional testing
- Re-testing defines **check/ validating the same functionality/ feature by passing multiple test data or**
- **Re execution of same application / build with multiple test data** to validate functionality of the application is known as **retesting**
- **Ex.** Paytm – Recharge module – Recharge mobile functionality → Test data = BSNL, Airtel, JIO, VI, MTNL and different state
- **Ex.** Paytm – Login page validation- Mobile no. / Email id- Test data Mobile no –Idea, Vodafone, JIO, BSNL, etc.
- For Re-testing we will get **Test data from Database (DB -172.10.31.124) (**exist project)
- If functionality depends on another application than **test data provided by BA**
- For test date **BA will sent Mail to tester**
- **Retesting** is performed only **two times**
  1. Before we lock the defect- we do retesting
  2. After developer solve defect (after getting fix)- we do retesting
- Before lock the defect, we validate whether the **defect is valid or not**
- Before raising the defect, we have to check it, whether its **good defect or bad defect**
- While executing test cases / test scenarios / test steps, if we found defect then we have to execute test cases / test scenarios / test step with **multiple test data** & still there is defect **or** still defect is reproducing **or** defect occurs repeatedly then it's a **good defect**
- While executing test cases / test scenarios / test step with **one data** & if we found defect, so it is considered as **bad defect**
- **Process**

  We do resting- we got the defect- we lock defect on JIRA & give remark to that test case as 'fail'- We got defect ID to that defect- developer solve that defect- developer again assign to tester- tester retest the test case & ensure that it work well- if not working tester again assign to developer

Note-Create defect in JIRA

- If defect is valid, then we have a JIRA tool. So we simply 'select project & project No.' then we select issue type as 'bug'.
- Then we write summary of that defect in the summary tab 'submit button functionality is not working'.
- Then we write description in the description tab 'having entered UN & PW. When I click on submit button then I faced/got this defect **submit button functionality is not working**'. So find below attached screenshot.
- Then we select or give the **priority** of that defect like highest, high, medium, & low
- Then we **assign** that particular **defect to the developer**
- Then enter the **environment** means which environment got the defect like- SIT, UAT…etc.
- Then **link the issue to the epic or user story**
- Then we **select the sprint & create the defect**
- After creating the defect then **defect id is generated**

- When developer feels that, it's a **valid defect then he changes some code** & **deploy that code on build**. so, after getting the fix, so, it's a **new build**
- There is **developer comment** on JIRA, **"the issue/bug/defect has been solved, in this particular package I have changed this code now it's working fine, please rest it & also I have given new testing build & testing data"** etc.
- So, it is actually assigned to us or present in the testing column. We execute that particular test scenarios/cases/step which was earlier failed with multiple test data to **ensure defect has been fixed/solved.** So, simply we have to check it is solved or not. If it is solved then we have to **close it**
- If it is **solved** then **tester comments**. Where we write closing comment, **"we have verified build version 4.1. Its working fine as expected. The defect has been resolved"**
- If it is **not solved (fail)** then **tester comment "the issue/bug/defect has been still existing in the environment. Kindly look in to this issue, please find attached below screenshots"**. When we click on the reassign it will be **reassign to the developer**
- Valid defect- developer change code- deploy code- new build-developer comment- assigned to tester- execute test cases which were failed- validate whether defect has been solved or not – yes (closing comment) – No (comment + screenshots)- reassign

Regression Testing

- During BBT (System & functional testing) & retesting, we execute test cases, if we found defect, than tester will create defect in JIRA & assign it to the developer
- If developer feels it's a valid defect then developer changes the code & deploy the code
- Developer writes a comment – in this package, I have changed this particular code. The defect has been solved & now it's working fine please test it
- Developer assign to us, we get new build & we **execute test case / scenario / test step** which was **earlier failed with multiple test data** on the **new build** to **ensure whether defect has been solved or not**
- As well as we **verify** the **impact of new code on interconnected modules, on main modules & sub modules** or
- We simply **verify**, whether it is **hampering other modules or not** or
- In Regression testing **check/ validation defect has been working correct or not &** There is **not side impact on interconnected module or**
- Regression testing is the process in which we are **testing modified build / newly corrected build to ensure that they are working well/fine** & also to **check their impact / side effect on other working module**
- **Ex.** Paytm- Recharge module- Mobile no. object→ Build (V9.0) (1000 line code) **Retesting**- VI, Airtel, JIO, MTNL it is working but **BSNL is not working** → Tester will raised defect to developer → developer will fixed the defect, **modified build (V9.1)** (1050 line code) & sent modified build → On **modified build**, we will perform **Regression testing (Re-testing + Regression testing)** → BSNL is working (10 BSNL Test data – **Retesting**) or not & check side effect / impact on VI, Airtel, JIO, MTNL
- Regression testing is performed **3 times**
  1. **During SIT-** Whenever we **found a defects /** When we got defects
  2. **After completion of SIT & UAT-** Whenever build is moving from **SIT/ UAT environment into prod environment then we performed Final regression testing(End to End testing) /** when **build is moving** from **one environment to another environment** (SIT to UAT OR UAT to Prod/ Live environment)

1. **During SIT-** in SIT, we get no. of defects as well as if CR comes then every time we get new build. So, we have to validate whether due to CR & this defect fix other functionalities have been hampered or not / impact of CR on other modules
   i.e. in SIT:- Defect fix + CR = new build = regression testing

2. **After SIT**- all fix= all defect= regression testing
   Check all other modules & functionalities are working fine or not

3. **After UAT**- in UAT, if client found defect then developer has to provide fix of it after that also we have to perform regression testing there
   i.e. in UAT:- New suggestions + CR = new fix = regression testing

- In Regression testing, we are preparing / create **Regression suite (bunch of test cases)**

  1. **Failed Test cases** (Ex. BSNL Test cases)
     When we get fix of the defect (i.e. new build), we execute test cases / scenarios / step which was failed earlier with multiple data to ensure defect has been fixed / solve or not & also check side effect of other module

  2. **High priority Test cases** / **Core functionality test cases** (Ex. JIO, VI etc. Test cases)
     Main functionality of main & sub modules test cases are executed here. From this we get we get idea whether system / application is ok or not

  3. **Extra feature or extra functionality Test cases** (Ex. **+91-** Mobile no text box → IND added, +91 feature test cases)
     If there is CR, due to that whether any functionality or modules are hampered or not is tested / verified here by executing test cases

  4. If **time permits**, we will consider reaming test cases (**we execute low & medium priority test cases**)
- For regression testing we required **2 to 4 hrs**.

- In **My project**, for Regression testing (Regression suite) 2 to 4hr.
- **Ex**. In manual testing – Module – 900 Test cases → Regression suite – 200 to 250 test cases

Or

- **In My project**, whenever we found defects, then we required **2 to 4 hrs. For regression testing.**
- **1Module –** Total 20US – Manual test cases – Total 450 test cases – Regression suite will contains 160 Test cases

Types of RT

Unit RT-We are testing only changes, or modification part done by developer

A

Regional RT-We are testing only changing part & also impacted area

A B C D

Full RT-We are testing the changed features & also remaining part of an application

A B C D E F