Manual-

Manual Testing-

- **Process**
  1. Basic SDLC (software development life cycle)
  2. Waterfall model/ process
  3. V-Model/ process
  4. **Agile Model/Process → 90 % to 95% process**

- **Testing type**
  1. Sanity testing, Smoke Testing, regression testing, etc

- **Manual part 2** – Test cases design, Test cases execution, review, defects, etc
- **Project management tool**- JIRA/ HPALM

API Testing-

- SOAP & REST service testing
- SOAPUI tool & POSTMAN tool

Database Testing-

- SQL quires

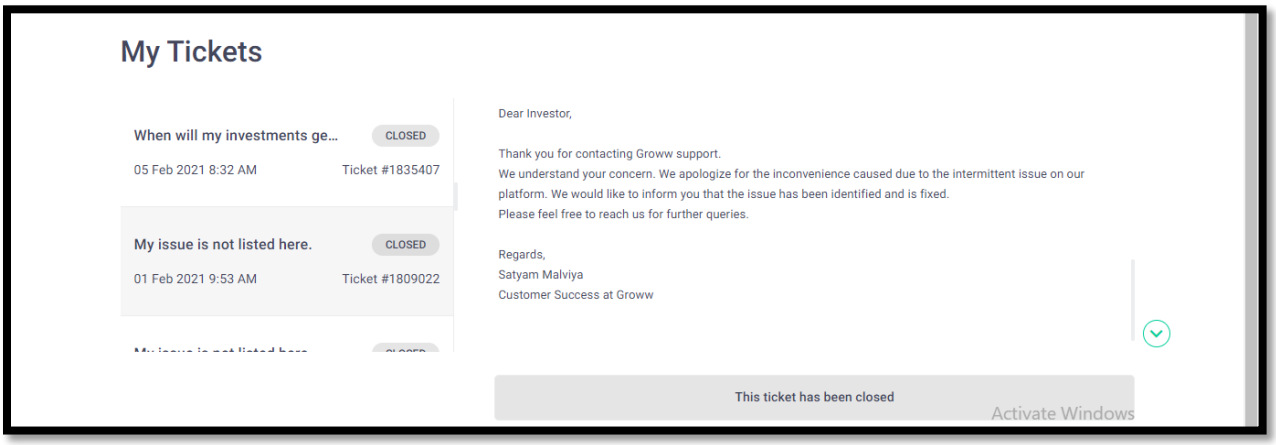
Project – 2live

Team Size–

- **Project Team-** New features/ functionality/ Module. Ex. Paytm – Invest in stock
- **Project team size is 24 to 26 people**
  1. **Deliver manager(1) –** Project delivery to client
  2. **Project manager (1)-** Project task assign/ work need perform with the time/ team handling
  3. **Business analysis (1)-** BA interaction client and collect the requirement/ functionality/ New features
  4. **Designer/ Solution archicture (1)-** project application design
  5. **Developer (14 to 16) –** Developer will code for application
  6. **Tester (5 to 6) –** Tester will do testing on developer application

- **Support Team –** Existing application issue/ defects/ end user quires/ Ticket Ex. Paytm- Recharge module
- **Support team size 9 peoples**
  1. **Project manager/ Support manager (1)-** Support task assign/ work need perform with the time/ support team handling
  2. **Developer (5 to 6) –** Developer will code for application
  3. **Tester (1 to 2) –** Tester will do testing on developer application

- **I have worked in Project team**
- Support Tickets-



- **Project –**
- **Wipro- Ex. Paytm = Project team & Support team**
- **Accenture- Ex. Groww= Project team & Syntel – Ex. Grow = Support team**

- **Service based company -** Infosys, Wipro, Persistence, etc
- **Product based company-** HSBC bank, IBM, Paytm, etc

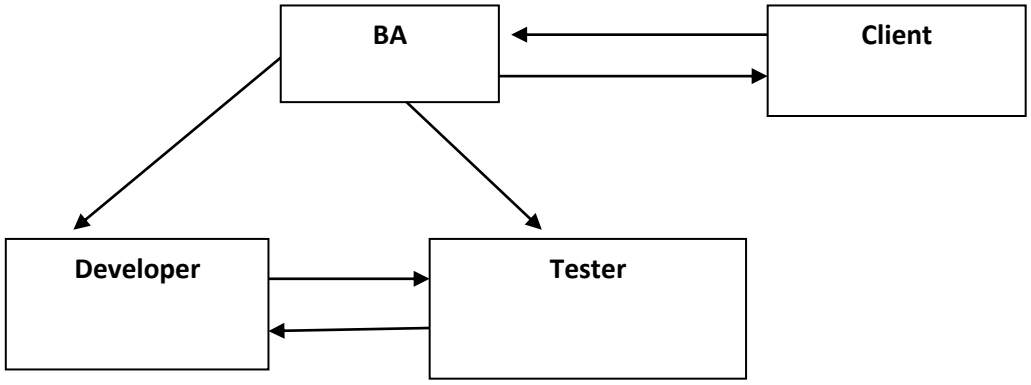
Manual Testing-

- **Defines-** Check / validation developed application will work as per requirements.
- Check completeness & correctness of functionality by manual

SQA process-

- **SQA process** defines to measure & monitor the developed application.
- SQA defines software quality assurance

- SQA process parameters-
  1. Clients/ customer requirements full fill
  2. Clients/ customer exception (security & performances)
  3. Clients/ customer delivery within time
  4. Risk managements



Process-

- Process deiced
- Ex. Client → HSBC company IT departments, Wipro company work → Client decide process
- Ex. Client → Cred application, No IT departments, Wipro company → Wipro decide process

SDLC process-

- SDLC- software development life cycle
- SDLC process development & testing application
- SDLC stage involved
  1. Information gathering
  2. Analysis
  3. Design
  4. Coding
  5. Testing
  6. Support/ Maintains

Information gathering-

- Information gathering will be **done by BA**
- BA will collect the **requirement from Client related to business of Client**
- In Information gathering stage **BA will papered a BRS documents**
- **BRS – business requirements specification**
- **BRS defines** clients business related requirements
- **BRS** documents will not get to tester
- **EX**. Client Business→ End user platform = Bidding/ Cricket biding →End user platform → End user brokerages charge → Crick, Football, etc→ Client business → Ex. Dream11, MPL

Analysis-

- In Analysis stage BA will work
- In Analysis stage **BA will collect the requirement from Client**
- BA will **collect requirements related the functionality of application/ software**
- In Analysis stage BA will prepared **SRS** (**Software requirements specification)**
- **SRS defies as functional requirements that will be implemented & system requirement that will be used.**
- **SRS also called FRS(**functional requirements specification**)/ CRS (**Customer requirements specification**)**
- **SRS will contains**
  1. **Functional requirements**
  2. **Functional flow diagram**
  3. **Use cases** (1 specific requirement)
     A. **Description** – Detail about the specific requirement
     B. **Acceptance criteria** – Does & don't about specific requirement/ Summery
  4. **Screenshot/ Snapshot/ prototype**
- **After completion of SRS documents BA will sent a mail these SRS documents to developer & tester**
- Developer & tester will **analysis/ understand/ study the SRS documents**
- If documents is not clear then we will do the meeting with BA

- **Ex. Use cases-BOQ- Stock Module- Intraday buy & Sell form new window**
- **Description**- User has to buy or sell throw intraday window. For end user intraday added which can buy or sell share / stock. In intraday, user can hold stock/share within 1 day. i.e. within trading time as per your country.

- **Acceptance criteria-**

1.	Intraday available in Stock module for all user
2.	Intraday for active with trading day (trade cycle time – 9.15 to 3.15 pm)
3.	Throw Intraday user can 1st buy then sell OR 1st sell then buy
4.	Intraday will provide margin for all stock/share
5.	Intraday will provide margin for all future & options
6.	After trade time, for Intraday error message will show – "Intraday not available you order will place after next trading day"

Design-

- In design stage designer or solution archicture will work.
- Designer will prepared the design same as SRS documents.
- Designer will prepared → **HLD** (high level design), **LLD** (low level design)

Coding-

- In Coding stage developer will work.
- Developer will work on **LLD** (low level design).
- LLD will implemented according to SRS/ as per Use cases

Testing-

- Tester will do the **Test cases design (TCD) & Test case execution (TCE)**
- Tester will check functionality as per SRS documents OR as per Use cases
- In TCE, if we **found a defect/ bug, tester will raised to developer**
- Developer will fix these defect/ bug
- At the end **application/ software deploy/ delivery to the client**
- Project team will give the warrant support for 1 month.

Support/ Maintains-

- **After completion of project team** warrant time/ period then project/ application went to support team
- Existing application **issue/ defects/ end user quires/ Ticket**

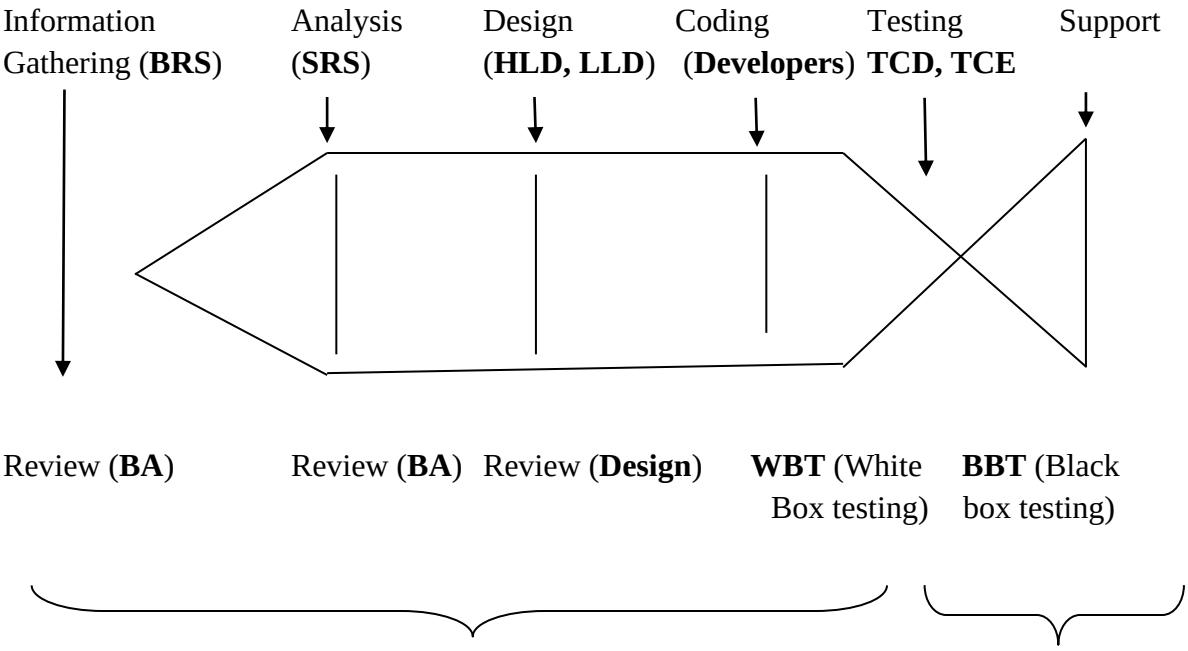
Interview Question-

1. **What is your team size? In last project what is your team size?**
2. **What is ratio of you developer & tester?**

- **Answer- In my Current/recent project** we have totals 25 to26 peoples. We have 16 developers and 6 testers. So we have ration of 3 developers: 1 tester

3. **What the SDLC process?**
4. **What is difference between SDLC & STLC?**
5. **What is SRS document and what it will contains?**
6. **If we don't clear the about requirements then what your approaches?**

Basic SDLC module / Fish module/Process –



- **Review-** To check the correctness & completeness of documents.

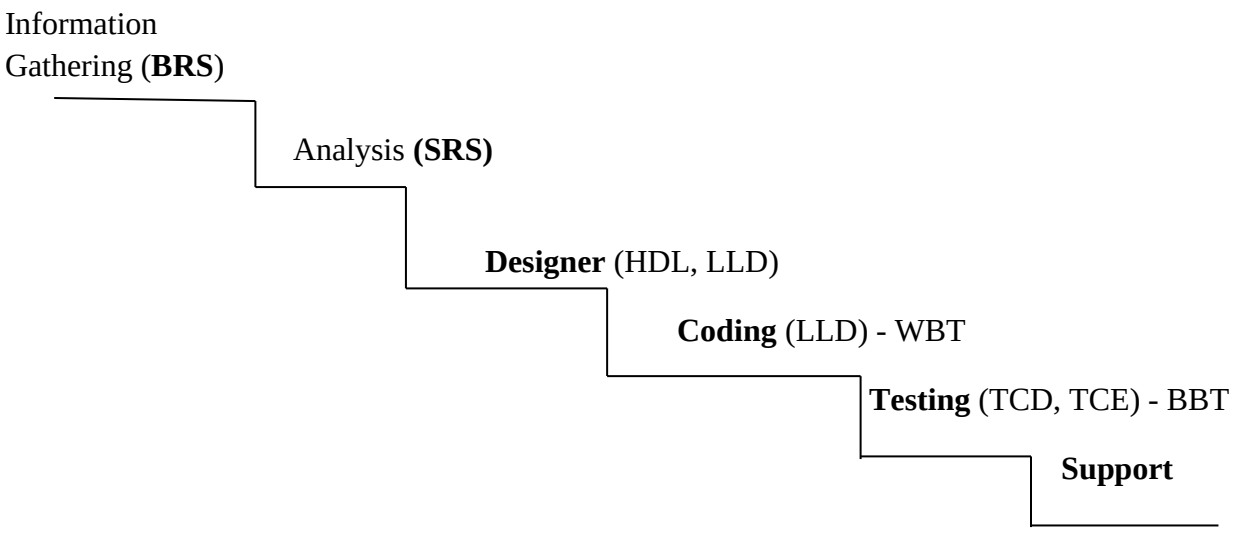
Static testing/ Verification	Dynamic testing / Validation
Definition- BA will check their documents, designer will check their design & developer will check code/ logic these process called Static testing/ Verification	Definition- Tester will check the functionality of application / software
Static testing/ Verification we will do/check **Quality control**	Dynamic testing / Validation we will do/ check **Quality assurance**
Static testing also called as Verification	Dynamic testing also called as Validation

Static testing/ Verification also called as **In-progress testing**	Dynamic testing / Validation also called **End progress testing**

WBT (white box testing)	**BBT (Black box testing)**
WBT is performed by **Developer**	BBT is performed by **Tester**
WBT testing 2 types 1. Unit testing 2. Integration testing	BBT testing types- 1. Sanity/ smoke testing 2. Functionality testing 3. Re-testing 4. Regression testing, etc
In WBT testing check – Logic, loop coverage, Brach converge, etc	In BBT testing check – Functionality, Behavioral coverage testing, input domain coverage, etc
WBT also called **code level testing**	BBT also called system & functional testing
WBT only performed in **+ve way** **Ex**. Paytm- Recharge Module- valid mobile no. testing	BBT performed in **+ve way & - ve way** **Ex.** Paytm- Recharge Module- valid mobile no. & invalid mobile no. testing

Waterfall Model/ Process-

- Waterfall model/ process it is also **sequential process**
- Sequential process after **completion of one next stage will start with next stage**
- **Ex**. Until Developer will complete, tester will not start test testing



Drawback-

- In Waterfall model **time is not fixed for deliver/ deployment**
- In Waterfall model, **backward/ backtrack is not possible** (in testing if we found defect then we can't go to coding stage)
- **Ex**. If in TCE, if we found a defect/ bug then we can't go to back stage (coding)
- **Ex**. Paytm – Recharge module - BSNL mobile no. accepting → defects note down. When new feature or new modules added in application/ software (New feature + old defects)

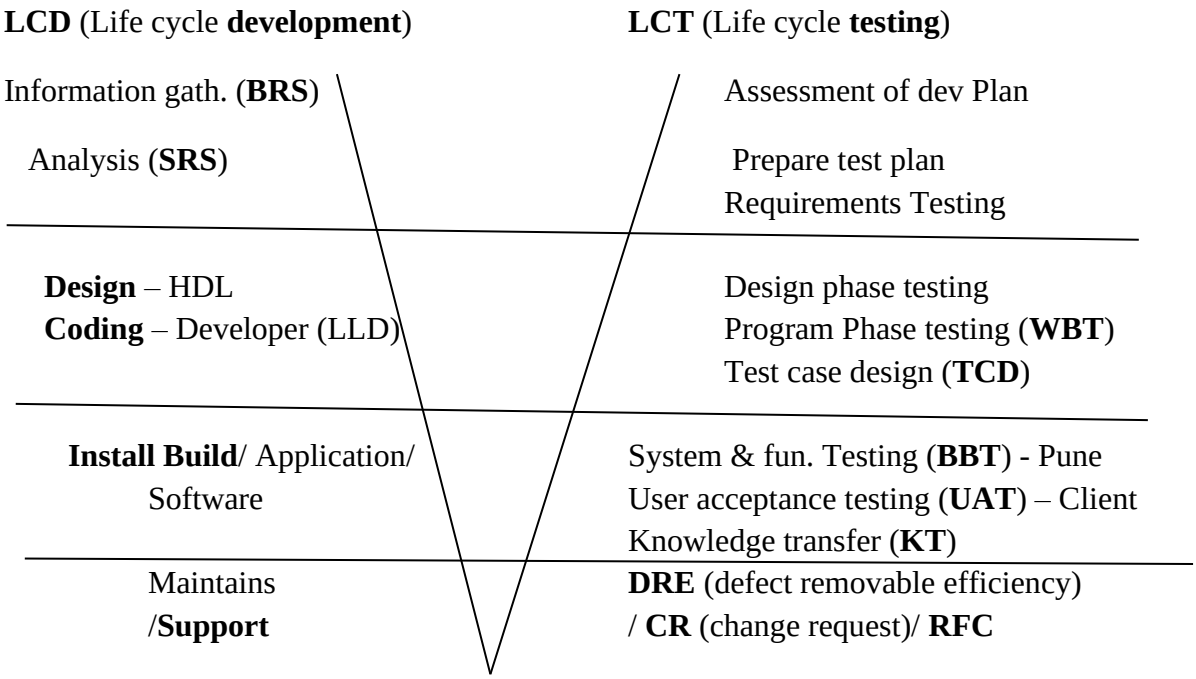
Q. What is an Entry criterion of Unit testing and what is an Exist criterion of unit testing?

- **Entry criteria-** When developer will completed coding, then developer will do unit testing **OR** when developer will sating the testing then it is called **Entry criteria for unit testing.**
- **Exit criterion –** When we will completed unit testing these is called **Exit criterion for unit testing OR** When we start with Integration testing it is called **Exit criterion for unit testing**

- **Unit Testing → Integration testing → System & function testing**

V-Module-

- V-module stand for **Verification & Validation**
- V-module / process → **Developer (LCD) and testing (LCT) are doing parallel**
- Developer & testing stages are work parallel
- In V-module / process **deployment / delivery process → 3 month**
- V- module it is **Plan driver process (**deployment / delivery process = 3 month)



- **DRE** (defect removable efficiency)- DRE = A/ (A+B)
- **A=** No. of defects found in BBT, **B**= No. of defects found in UAT
- **DRE** defines how much deeply we have tested application/ build
- **CR** (change request)/ **RFC (**request for charge)-
- **EX.** Paytm – Recharge module- Amount field under – Recharge Amount – description shows green text
- Id any CR comes from the client then your comply will **accept these CR and apply the extra charge to client**

Drawback-

- In V-module / process **deployment / delivery process → 3 month**
- **Extra money for CR**

Agile process/module-

- Agile defines **continues development & continues testing will happen in application/ build**
- Agile process it is **Values driven process** (Priority to **Client**)
- If any CR comes at any point of time the we will consider these CR
- When will **accept these CR** then we will check impact on current development, Testing & production.
    1. If CR has more impact on development/ testing/ production then we will inform to client
    2. If CR has less impact on development/ testing/ production then we will develop & testing and we will deploy to client.
- In Agile process/ module **deployment / delivery process → 2 week/ 3 week**
- **Agile subtypes/ framework/ methodology**
    1. XP (extreme programming) – Development & no testing
    2. **Scrum -** (Bunch of requirement → Sprint wise development & Testing & Delivery)
    3. Kanban – Support team (tickets/ Existing issue/ bugs/CR)
    4. Lean - Support team
    5. FDD – Future driven development

- I have worked in **Scrum agile methodology/ subtypes**

Agile architecture –

SDLC		**Agile**
Information Gathering (BRS)	**BA**	**Product Backlog (**Project = 2000 req**.)**
Analysis (SRS)	**BA**	**Sprint Backlog (**Sprint = 20 req.) – **Client priority** 1 sprint = 20 req./US – Decider PM, BA & designer 2 sprint = 18 req./US 3 sprint = 17 req./US
Use Cases (specific 1 req.) 1. Description 2. Acceptance criteria		**User Story (specific 1 req.)** 1. Description 2. Acceptance criteria
Designer (HDL)		**Designer**

Coding (developer - LLD) **Coding (developer - LLD)**

Tester – TCD, TCE **Tester – TCD, TCE**

Support **Support**

Agile people-

- **Stockholder → Client**
- **Delivery manager → Solution master**
- **Project manager → Scrum master**
- **Business annalist → Product owner**
- **Designer → Designer**
- **Developer → developer**
- **Tester → Tester**

Agile Meeting/ Ceremonies-

1. Grooming meeting
2. Sprint planning
3. Scrum meeting/ Daily stand up
4. Sprint review meeting
5. Sprint retrospective meeting

Agile meeting	Purpose	Involved
Grooming meeting (Before start of sprint /Any time in sprint)	**- US /Requirement clear/ doubt understand**	**1hr – BA,** PM, Designer, Dev & Tester
Sprint planning meeting One time- Start day of sprint	**- Current sprint=** 20 US Added/work – decided by PM, BA & Designer **- Estimation (**time span)/ **Story point Ex**. 1US/req. – 16hr Developer + 14hr Tester	**30Min- PM,** BA, Designer, Dev & Tester
Scrum meeting/ Daily stand up (Daily – 9.45 am to 10.00am)	**- What you have done yesterday** **- What are doing today** **- Issue/ roadblock**	**15 Min- PM,** BA, Designer, Dev & Tester
Sprint review meeting (Last day of sprint)	**- 4 US tested by tester – Demo/ Review to BA/ Client/ UAT**	**1hr- BA/Client/ UAT, PM,** Designer, Dev & Tester
Sprint retrospective meeting (Last day of sprint)	**- Current sprint – Good & Bad discussion** **1 Sprint → 2 sprint**	**30Min- PM,** Designer, Dev & Tester, BA (optional)

Agile Day wise Plane-

- Agile peoples / Team = 24 to 26 (16 developer & 6 Tester)
- Agile duration – **2 week (Monday to Friday)** = 2 * 5 = **10 days**
- Agile → Current Sprint= **Sprint 1 = 18 US/** requirements (For 4 module each4 to 5 US)
- 1 Team = 3 developer + 1 tester = Recharge module (4 US)
- 2 Team = 4 developer + 1 tester = Movies ticket module (4 US)
- 3 Team = 4 /5 developer + 2 tester = Investment in Stock (6US)
- **4 Team = 3 developer + 1 tester = Fasttag module (4 US)**

1 week – (Monday to Friday)

Monday –

- **Grooming meeting –** US doubt / clear about – **1hr**
- **Sprint Planning meeting (30 min)–**
  1. Current sprint = **Sprint 1= 18 US** – decide PM, BA, Designer & Task add
  2. **Estimation/ Story point** (time span) → 4 Team = 4 US assign to Team
     Ex. 1US = 12hr development + 10hr Testing, 2US= 16hr dev + 8hr testing, etc

- **1 US →** 1US = Coding (6/7hr-Inprogess) **+ 1US – TCD(5hr- Completed)**

Tuesday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**1US –TCD- Completed**)
  - What you are doing today (**1US- TCE**)
  - Issue/ roadblock

- **1US→** 1US = Coding (2hr- **Sent Build**), 2US – coding- (4hr- In-progress) + **1US – TCE (6/7hr- Completed)**

Wednesday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**1US –TCE- Completed**)
  - What you are doing today (**2US- TCD**)
  - Issue/ roadblock

- **2 US →** 2US = Coding (6hr- Completed) **+ 2US – TCD(6hr- Completed)**

Thursday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**2US –TCD- Completed**)
  - What you are doing today (**2US- TCE**)
  - Issue/ roadblock

- **2 US →** 2US = Coding (1hr- **2US Sent build**), 3US-coding – (6hr-In-progess) **+ 2US – TCE (6/7hr- Completed)**

Friday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**2US –TCE- Completed**)
  - What you are doing today (**3US- TCD**)
  - Issue/ roadblock

- **3US→** 3US-Coding (6hr- Completed) + **3US – TCD (4/5hr- In-progress)**

2 Week- (Monday to Friday)-

Monday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**3US –TCD- In-progress**)
  - What you are doing today (**3US- TCD completed and 3US- TCE**)
  - Issue/ roadblock

- **3US→** (3US-Build sent). 4US- Coding (6hr-In-progress) + **3US – TCD (1hr- Completed) & 3US- TCE (5/6hr- In-progress)**
- **3US- in TCE- defects (2 defects)- Raised to Developer → Develop fixed defect → Tester will test defects (2 defects- fixed/closed) (1/2hr)**

Tuesday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**3US –TCE- In-progress, 2 defects raised/ Closed**)
  - What you are doing today (**3US- TCE completed and 4US- TCD**)
  - Issue/ roadblock

- **4US→** 4US- Coding (6hr-In-progress) + **3US – TCE (1hr- Completed) & 4US- TCD (4/5hr- In-progress)**

Wednesday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**3US –TCE-Completed, 4US-TCD –In-progress**)
  - What you are doing today (**4US- TCD & 4US -TCE**)
  - Issue/ roadblock

- **4US→** 4US- Sent Build (2hr-Complted) + **4US – TCD (2hr- Completed) & 4US- TCE (4/5hr- In-progress)**
- **4US- TCE in defects (5 to 6 defects) → Raised to developer → Raised (3 to 4 defects fix- 3hr) → Sent for Testing & Developer is not accepting defects (5 to 6 defects)**

Thursday-

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**4US –TCE-In-progress, 4US- 6 defects found**)
  - What you are doing today (**4US- TCE & we will test defects)**
  - Issue/ roadblock (**Developer is not accepting the defects – 5 to 6**)

- **4US→** 4US- defect fix (defect 5 to 6) (4hr-In-progress) + **4US – TCE (2hr- Completed) & 4US- defect testing(defect 3 to 4) (3/4hr-Complated)**

Friday- (Last day of Sprint)

- **Daily stand meeting (15 min)**
  - What you have done yesterday (**4US –TCE-Completed)**
  - What you are doing today (**4US- defects 5 to 6 I will test)**
  - Issue/ roadblock

- **4US→** 4US- defect fix (defect 5 to 6) (1hr-Complated) + **4US defect testing(defect 5 to 5) (2hr-Complated)**

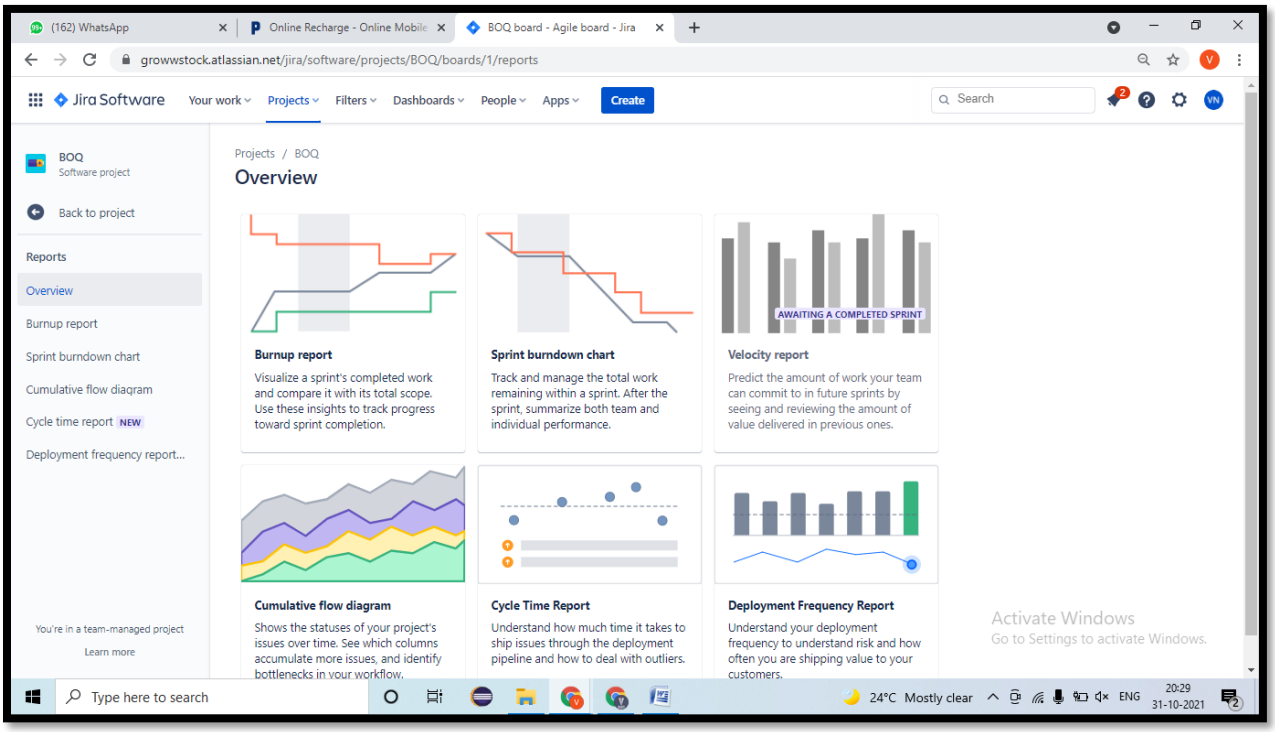
- **Sprint Review meeting (1hr)-**
  - **Tester** will give **Demo/review** to BA/Client/UAT of **→ 4US**

- **Sprint Retrospective meeting (30 Min)-**
  - Current sprint (Sprint 1)- **Good & Bad** things about Process, Development & Testing

- Good things carried to next sprint

Agile Term-

- **Epic**- Main Module name - Epic → Multiples US present against that module
- **Burn down chart**- It is graph which defines how much **work is reaming w.r.t. time**
- **Burn up chart**- It is graph which defines how much **work is completed w.r.t. time**
- **Velocity**- Defines how much we will deployment/ delivering to Client
- **Estimation/ Story point**- Time span for US / Development & Testing time for a US
- Estimation will defines deposing on following on
    1. How much knowledge you have against US
    2. How much efforts will be required
    3. How much complexity of the US
- Estimation/ Story point based on/ unit → **Hours**



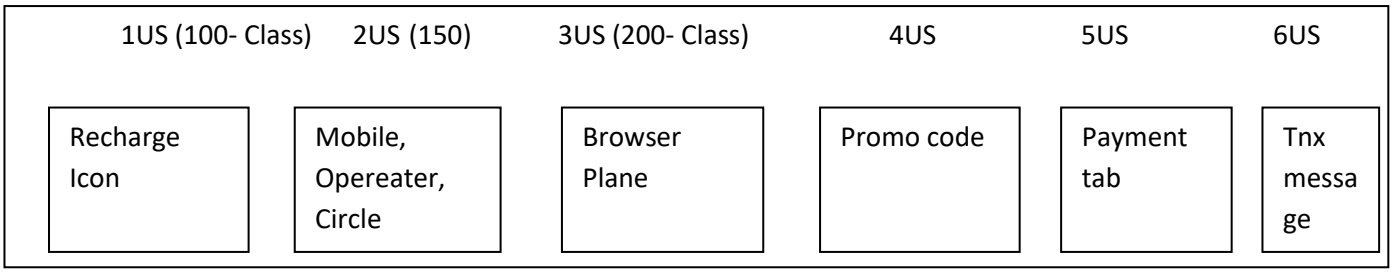
Agile Advantage-

- Agile advantage

1. Sprint wise delivery/ deployment- 2 week/ 3 week
2. Automation is possible is in Agile process

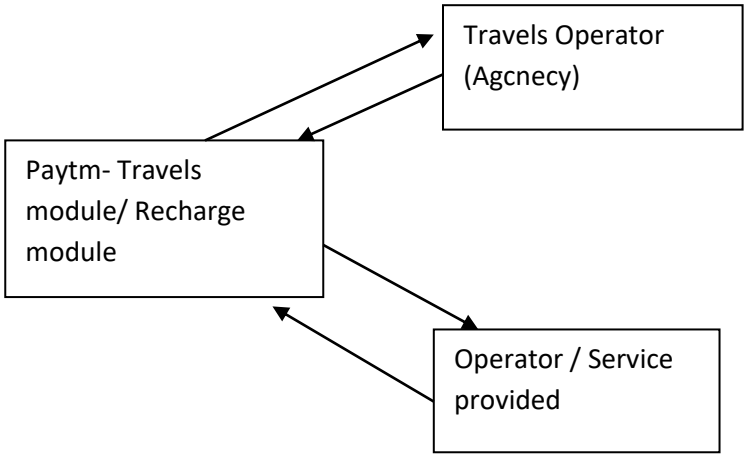
3. Daily stand up/ Scrum meeting- Work progress (In my project, we will do 2 times daily stand up- Moring & evening)
4. Check point is present in every module

Recharge Module-



Agile dis-Advantage-

- **If frequent changes in requirement/ US**, we can't deployment/ delivery these US to client
- If your **application/ module is depend on another** application/ module, we can't deployment/ delivery these US to client



Interview Question-

1. **What is difference between V-module & Agile**
2. **What is the Agile & what are agile frameworks?**
3. **What is the Agile, Scrum/ Sprint & Epic**
4. **What is Burn down chart, Burn up chart & Velocity**

5. **What are the Scrum Agile mythology/ architecture of Agile?**
6. **What are difference types meeting in agile/ what is ceremonies in the agile?**

7. **What is the Agile & How you are following agile process in your originations?**
    - Answer- Day wise plane in Agile (2 week)

8. **What are advances in the Agile & dis- advances agile process?**
9. **How you will decide the estimation in Sprint planning meeting?**
    - Answer- In my project, PM will open the **Agile Board (JIRA/HAPLM) in sprint planning meeting** then PM will ask to every developer & tester about the **time/ story point for every US** which is assigned to you.
    - Estimation will defines deposing on following on
        1. How much knowledge you have against US
        2. How much efforts will be required
        3. How much complexity of the US

10. **What is Sprint zero?**

- **Answer-** For **next sprint preparation whatever the extra effort** will be required for preparation these effort will called **Sprint Zero**
- Ex. PM will do → Sprint 1 (last day)→ Next sprint (Sprint 2) US & US (US against task) rough estimation

11. **When are a delivery/ deployment in your project?**

- **Answer-** In my project we are working in Scrum Agile methodology, where we will work for 2 week/ 3 week (Monday to Friday)
- In my project we will Delivery/ deployment on – **Saturday/ Monday**

12. **Who will do the Delivery/ deployment?**

- **Answer-** In my all project, **Developer** will do Delivery/ deployment to All environment & to the client

13. **What is Kanban & Lean**
14. **What is your approach if US is not completed (development/ testing) in current Sprint?**

- **Answer-** In my project, PM will tell to developer & tester will work on Saturday & Sunday and complete the work (More chances work will be complete)
- **But still If still work will not completed in** Saturday & Sunday then PM will prepared a US in next sprint – **US= "Reaming Work In Last Sprint"**

15. What is the Scrum of Scrum Master?

- **Answer-** Head of all Scrum Master (**1 Scrum of Scrum Master – 4 PM present under in that)**

Agile → Scrum →Bunch of US work = Sprint 1

EPIC- Name – New Electricity module in Paytm - 40 US – Product Backlog

Sprint Backlog – 4US – Electricity + 4US recharge module + 4US Rent payment +4US

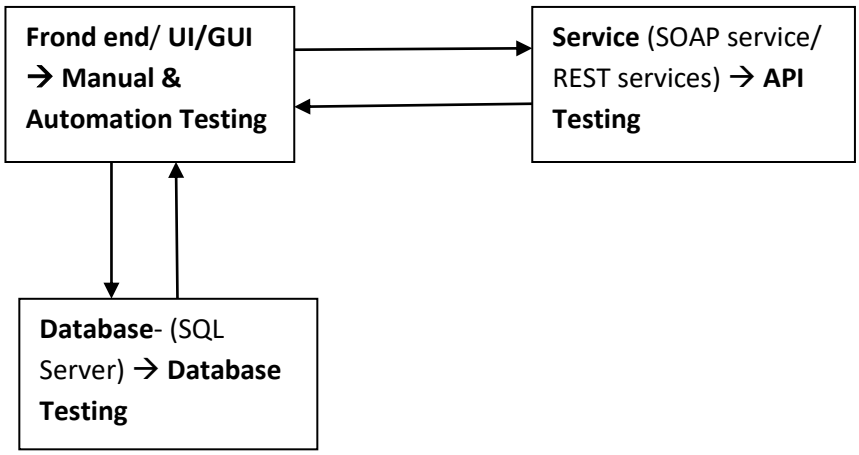
Sprint 1 = 16 US = Working

Sprint 2 = 18 US = Working

Reward → Preference (Food/ Entrainment/ Retails/ etc)

US- Preference in reward at end user

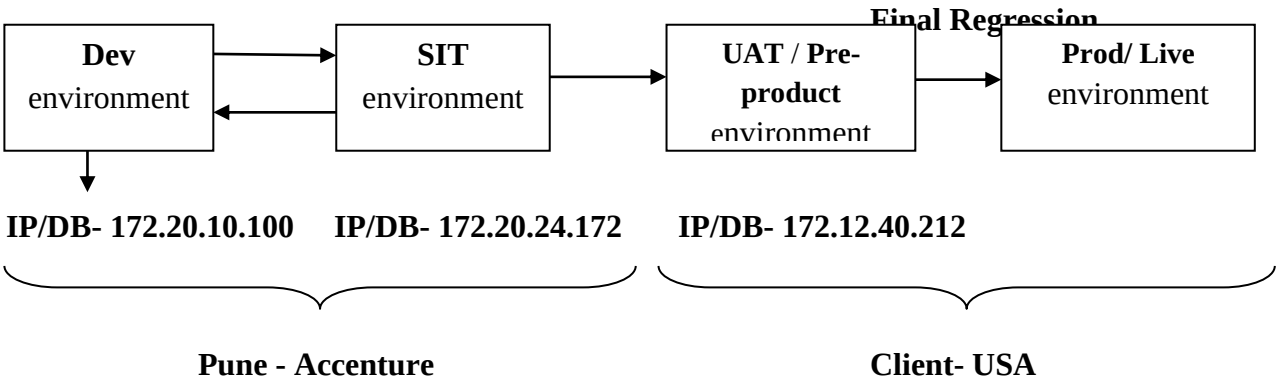
Project/ Application –



- Different types of **technology in your project**
  1. Frond end → **Dot net languages**
  2. Services → **Java languages**
  3. Backend/ Database → **SQL Server**

Environment-

- In my **Project 4 types environment**
  1. **Dev environment** - developer work → Coding + WBT
  2. **SIT environment** (System integration testing) – Tester → TCD, TCE, defect
  3. **UAT environment** → Client Location (Developer + tester)
  4. **Prod/ Live environment** →End user application



- **In Dev** environment → developer will do coding & WBT
- **Developer will sent the Build deploy/ delivery into SIT** environment
- **Dev** environment **(feature 250 line code**) → Build delivery (250 line) → SIT environment (f**eature -250 line code)**
- **Developer will sent Mail to Tester (throw JIRA/ HAPLM) → in Mail Attachment – Unit Testing documents**
- **In SIT** environment → Tester will work = **TCD, TCE**
- **In TCE if we found defects then we will raised to developer → Inform throw Mail (JIRA/ HAPLM)**
- **Defect will fix by developer in dev** environment → Modified Build sent → Tester will check defects
- **In SIT** environment→ TCE, tester will prepared the **test proof** → Mail sent to UAT team throw JIRA/ HPAML tool

Dev open →URL= 172.20.10.100:8080/paytm.com
SIT open →URL= 172.20.24.172:8080/paytm.com

Dev environment Testing-

1. Unit Testing
2. Integration Testing

SIT environment Testing-

1. Sanity Testing/ Smoke Testing
2. System & functional Testing (BBT)
3. Re-testing
4. Regression Testing, etc

UAT environment Testing-

1. Alpha Testing
2. Beta Testing

Prod/Live environment Testing-

1. Production Issue

Dev environment-

- In Dev environment , **developer will work**
- Developer will do **coding as per US**
- After completion of coding Developer will **do testing (WBT)**
    1. **Unit Testing**
    2. **Integration Testing**

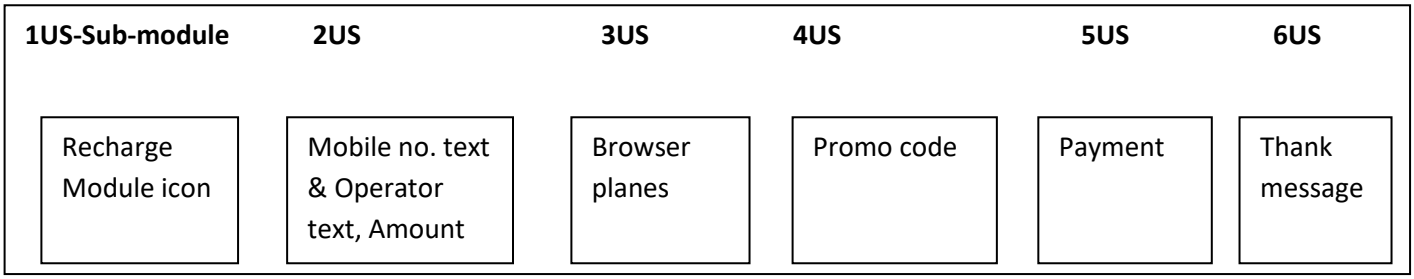
1. Unit Testing-

- Unit testing will be **performed on every Sub-module**
- Unit testing will be **performed on every US (Specific functionality)**
- **Ex**. Paytm → Recharge Module→US= Promo Codes tab in Recharge →developer has done the coding →Developer will do Unit Testing
- Unit Testing documents will contains → **Screenshot for testing WBT, Tables Name, URI/URL, etc**

2. Integration Testing-

- Integrations testing will **performed by developer**
- Integrations testing will **performed on Main Module**

- Integrations testing will **performed on by combing all sub-module**
- **Ex**. Paytm → **Main module**= **Recharge Module**



1. **Frond integration testing-**

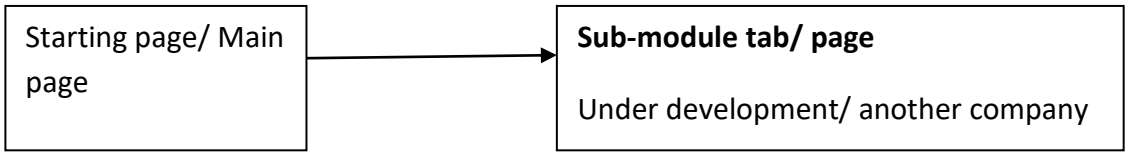
- For Frond interaction testing developer will use **call function (OOPS)**

2. **Backend integration testing-**

- For backend interaction testing developer will use **join Quires**

Integration testing approaches-

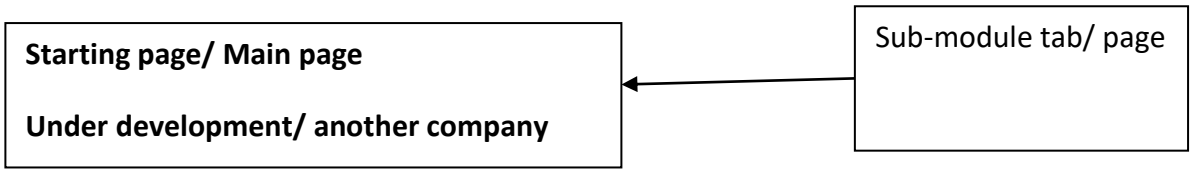
A. Top down approaches-



Stub- Temporary program → developed throw XML

- If we don't have sub-module then developer will prepared **Stub**
- **Stub-** It is Temporary program prepared by developer throw XML languages

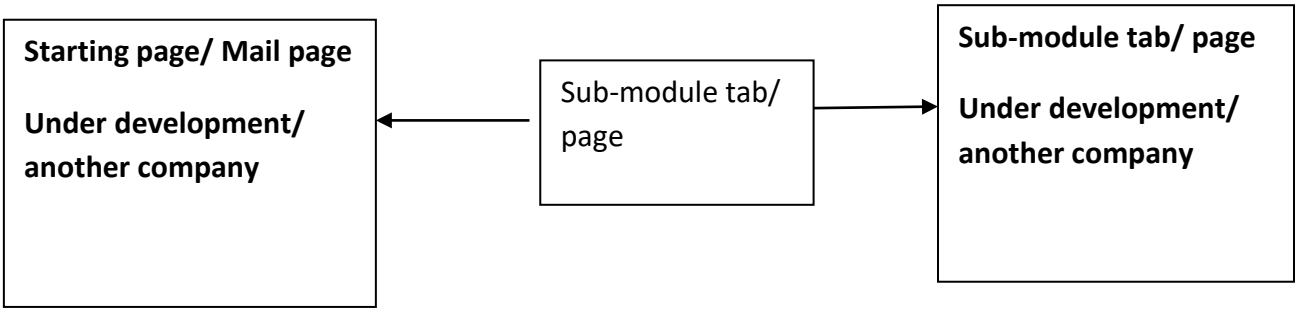
B. Bottom up approaches-



Driver- Temporary program → developed throw XML

- If we don't have sub-module then developer will prepared **Driver**
- **Driver -** It is Temporary program prepared by developer throw XML languages

C. Sandwich approaches-



Driver- Temporary program → developed throw XML

Stub- Temporary program → developed throw XML

Interview question-

1. **What are technologies used in your project?**
2. **How the deployment process in your organization?**
3. **Who will do the all deployment in your organization & how?**
4. **How you got the Build?**
5. **When you are starting the Testing?**

SIT environment-

- In SIT environment **tester will work**
- Tester will do **TCD & TCE in SIT environment**
- What is these testing, Why we will do these testing, What will we do in these testing, Preparation of defect/ TCD/ Documents/ mail in these testing, etc

Sanity Testing-

- When developer will **sent us a New build** and inform throw Mail (JIRA/ HPALM)
- In SIT environments, Sanity testing it is a first testing which is performed, Sanity testing is called **level zero testing**
- When we got the build, tester will **check build stability i.e. checking either these build is stable for testing or not**
- Sanity testing also called **Tester acceptance testing/ Build verification testing/ build test**
- In Sanity testing we will check
  1. Validation the **core functionality of application/ build**
  2. Validation the **GUI/ UI of application/ build**
  3. Validation the **link of application/ build**
  4. Validation the **tab/ pages**
  5. Validation the **navigation**

- In Sanity Testing if **build is not stable for testing** then **tester will reject the build**
- In sanity testing, tester will **not log/ create a defect**
- In sanity testing, if build is not stable for testing then we will reject build and **inform to developer throw Mail (JIRA/ HPALM)**
- In sanity testing, tester will **not writes test cases**

- **Ex. Paytm-** Rent Payment (Module) → **US** → Coding(500 line) & **Create New Build (V9.0)** →New build will sent/ Deploy → **Tester will do Sanity Testing** → Checking Build **stability for testing** → In Build **core functionality in not working** → Tester will

New Build reject (V9.0) → Developer will fix core functionality & **Create New build (V9.1)** → **New build (V9.1)** will sent/ Deploy → Tester will do Sanity Testing

- In Sanity testing build is not stable or Sanity testing issue due to **SIT environment problem, System hung out, Run time problem, Core functionality/ Basic functionality, etc**

- For Sanity testing, we required **2 to 3 hr for every New Build**
- In sanity testing, only tester is involved

Smoke Testing

- Smoke testing it is advance version of sanity testing.
- Smoke testing → When we got the **New build**, tester will **check build stability i.e. checking either these build is stable for testing or not**
- In Smoke testing we will check
  1. Validation the **core functionality of application/ build**
  2. Validation the **GUI/ UI of application/ build**
  3. Validation the **link of application/ build**
  4. Validation the **tab/ pages**
  5. Validation the **navigation**

- In smoke Testing if **build is not stable for testing** then **tester will reject the build and tester will give/ find the root cause of the defect/ issue**
- **In Smoke testing = Sanity Testing + Troubleshooting → Done by tester**
- **In Smoke testing = Sanity Testing + Package validation → Done by Developer**
- **How to do Troubleshooting-**
  1. Login into application or open application under test
  2. Pass data or verify functionality of application
  3. If we found error or defect then we will go Error logs (**172.20.24.172:**C:\application\Windows NT\Action\Logs.txt)
  4. Logs.txt file will contains all details about defect/ issue

- In smoke testing, **tester will not write test cases**
- In smoke testing build is not stable or smoke testing issue due to **SIT environment problem, System hung out, Run time problem, Core functionality/ Basic functionality, etc**

- In smoke testing, if we **found issue then we found root clause of the defect/ issue and inform to developer throw Mail (JIRA/ HPALM)**
- In smoke testing, **2 to 3hr for every New Build**
- In smoke testing, **Tester & developer both involved**
- **In my project, We are following Smoke testing**

Interview question-

1. What unit testing documents will contains?
2. What is integration testing? How you performed integration testing?

- **Answer-** Integrations testing will **performed on Main Module.** Integration testing will be performed by **both developer as well as tester**
- Sprint – US –Module -> **US name= Integration testing on Fastagemodule**

3. What is the difference between sanity testing & smoke testing?
4. What is Sanity testing & Smoke testing? Which testing is following in you origination?

- **Answer- In my project, We are following Smoke testing**
- In smoke Testing if **build is not stable for testing** then **tester will reject the build and tester will give/ find the root cause of the defect/ issue**
- If interview ask we will performed sanity testing, **in my project, we are following Smoke testing may be we are doing same work as you sanity testing doing.**

5. Which is the testing you will prefer after getting a new build?
6. Which types of defects you generally got in Smoke testing?
7. How you will inform to developer in sanity testing/ smoke testing?
8. What are you doing in sanity testing/ Smoke testing? Are you writing test cases for these testing?

System & Functional testing (BBT)-

- System and functional testing performed after completion of smoke testing/ After checking the build stability
- System and functional 4 types
  1. **Usability testing** } **90 to 95%**
  2. **System & Functionality Testing** }
  3. Performance testing - Jmeter, Loadrunner, etc
  4. Security testing - Jmeter, SOAPUI, etc

Usability testing-

- Usability testing, tester we will validate user friendlessness of the Screen/ application
- Usability testing 2 type,
  1. **GUI(graphical user interface) / UI(user interface)-**
  - In GUI/ UI testing, as tester validation
    - **A.** Validation **look & feel of the screen / application**
    - **B.** Validation **ease to use (End user ease) of screen / application**
    - **C.** **Speed of interface**

  2. **Manual support Testing-**
  - In manual support testing, as tester we will Validation , How manual text will be support in screen / sensitiveness on screen

System & Functionality Testing-

- 2 types of System & Functionality Testing-
  1. **Functionality testing** – validation **internal feature** of the application/ build
  2. **Non- Functionality testing -** validation **external feature** of the application/ build

Functionality testing (BIEBSC)-

- In Functionality, tester will validation **internal feature of the application/ build**
- For validating internal feature we will performed different coverage
- Functionality is different types
  1. **B**ehavioral coverage testing
  2. **I**nput domain coverage testing
  3. **E**rror handling coverage testing
  4. **B**ackend converge testing/ database coverage testing
  5. **S**ervice based coverage testing
  6. **C**alculation based coverage testing

1. **Behavioral coverage testing-**

- Tester will test/validate, behavioral of the object/ Web elements present in application
- Tester will check **behavioral of the object/ Web elements** & **property of the object/ Web elements**

Object/ Web elements	Behavioral & Property
Text box	Focus & Un- Focus
Check box	Check & un-check OR Tick & Un-tick

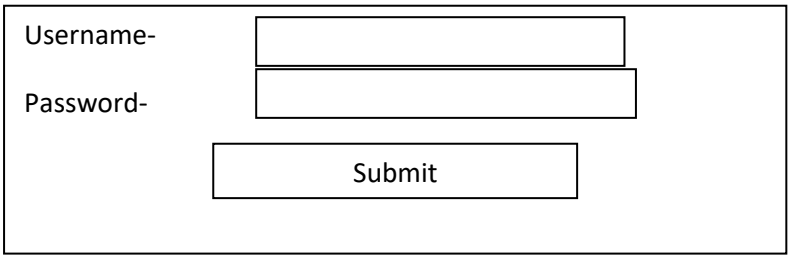
Button	Enables & Disables
Radio button	On & off

2. **Input domain converge testing-**

- Tester in input domain coverage testing, tester **validation different input size / length & data types**
- For input domain coverage testing maintaining these size/ length & Data types
  A. Bounder values analysis (**BVA**)
  B. Equivalent class partition (**ECP**)
  C. Decision table testing techniques

Bounder values analysis (BVA)-

- BVA we will check **Input size / length**
- **Ex**. Login page- Username- only Mobile no, Password- 4 to 6 charter combination (Capital, Small letters & No.)



BVA (Size/length)	**Pass**	**Fail**
Username text box -	10 digits	8, 11 digits
Password text box -	4 digits	3 digits
	5 digits	7 digits
	6 digits	8 digits

Equivalent class partition (ECP)-

- Equivalent class parathion validation input pass – **data type**
- Ex. Login page- Username- only Mobile no, Password- 4 to 6 charter combination (Capital letters, Small letters & 2 No.)

ECP (data type)	**Pass**	**Fail**
Username text box -	0-9 (int)	A-Z, a-z, Special chare
Password text box -	A-Z, a-z, 0-9	Special char (4 to 5 length)

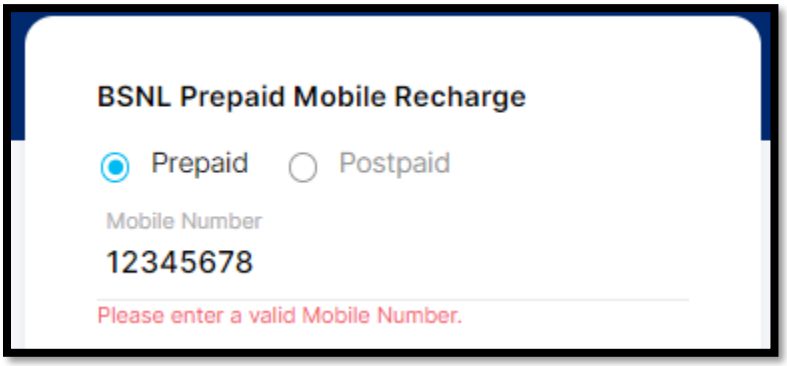
Decision table testing techniques-

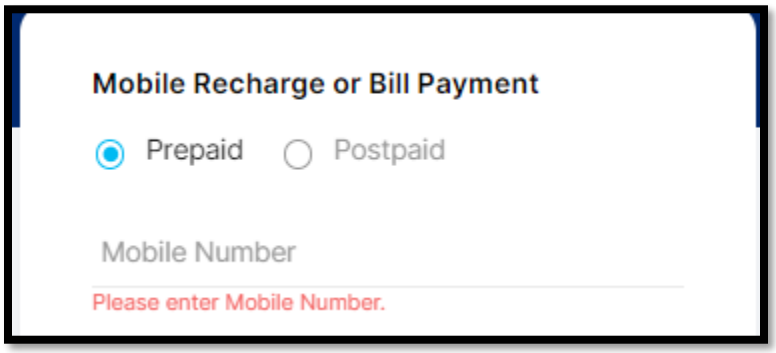
- Validation **different input combination on the object/ web elements → result**
- Ex. Ex. Login page- Username- only Mobile no, Password- 4 to 6 charter combination (Capital, Small letters & 2 No.), Submit

Objects	Rule 1/ input combination 1	Rule 2/ input combination 2	Rule 3/ input combination
Username	Valid	Valid	In-Valid
Password	Valid	In- Valid	Valid
Submit	Press	Press	Press
Result	Home	Error	Error

3. Error handling converge testing-

- Validating **different types error will generated in object** of application
- If we will pass **invalid/ wrong data** into object/web elements
- Ex. Paytm –Recharge module





4. Backend coverage testing/ Database converge testing-

- Validating **frond end operation that data stored into backend**
- Backend coverage testing also called **database testing**
- **Ex.** Paytm –Recharge module (frond end)- Transaction ID/ order ID- 56783408456

DB- **SQL server -** **Select * from Table**

Where Transaction ID/ order ID = 56783408456

5. **Service based converge testing-**

- Validating **sequential operation of the application**
- Ex. Paytm –Recharge module – Mobile no. & Circle & operator & Amount → Promo code → Payment tab → If payment done Thanks message OR If payment not done →Pop for cancellation –Yes → Home Recharge module
- Ex. Paytm –Recharge module – Mobile no. & Circle & operator & Amount & Click on fast forward→ Payment tab → If payment done Thanks message OR If payment not done →Pop for cancellation –Yes → Home Recharge module
-

6. **Calculation based converge testing-**

- Validating **arithmetic calculation of application**
- **Ex**. Paytm → Recharge module 499rs →Promo code (10%) → Payment – 499-49 = 450rs

Non- Functionality testing (RCCIISPG)-

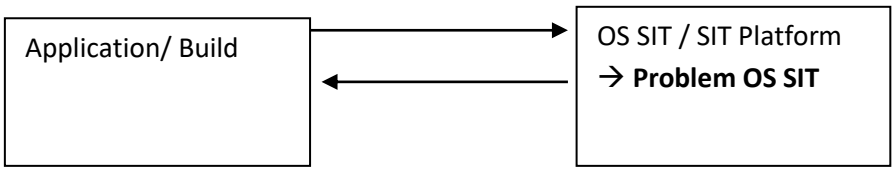
- In Non-Functionality, tester will validation **External feature of the application/ build**
- Non- Functionality testing, different testing types
  1. **R**ecovery coverage testing
  2. **C**ompatibility coverage testing (Browser **C**ompatibility testing)
  3. **C**onfiguration coverage testing/ hardware coverage testing
  4. **I**nstallation coverage testing → **We have not done**
  5. **I**ntersystem coverage testing
  6. **S**anitization coverage testing
  7. **P**arrelaization coverage testing → **We have not done**
  8. **G**lobalizations coverage testing

1. **Recovery coverage testing-**

- Validating either application is handling **abnormal situation**
- Ex. Filpkart- Payment page/ tab→While doing Payment close or refresh the page → Close application → Add to card page/ tab

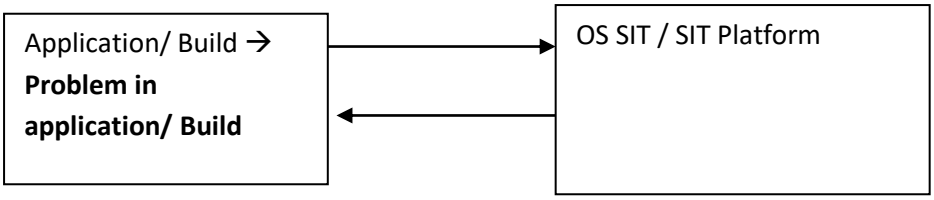
2. **Compatibility coverage testing-**

- Validating the application → Client excepted platform work
- Compatibility 2 types
  1. **Forward Compatibility testing –**



- If we have problem in OS of SIT environments → **Network team**

2. **Backward Compatibility testing –**



- If we have problem in application of SIT environments → **Tester**
- Backward Compatibility testing → In my project I have done "**Browser Compatibility testing**"

A. **Cross browsing Compatibility testing** –
   - Validating application/ build it support **different browser**
   - **Ex**. Different browser – Chrome, Firefox, Opera, Edge, suffari, etc

B. Version control Compatibility testing-
   - Validating application/ build it support for **different version on same browser**
   - **Ex. Chrome browser – V90.10, V89.00, V85.00, V70.00, etc**

3. Configuration coverage testing/ hardware coverage testing-

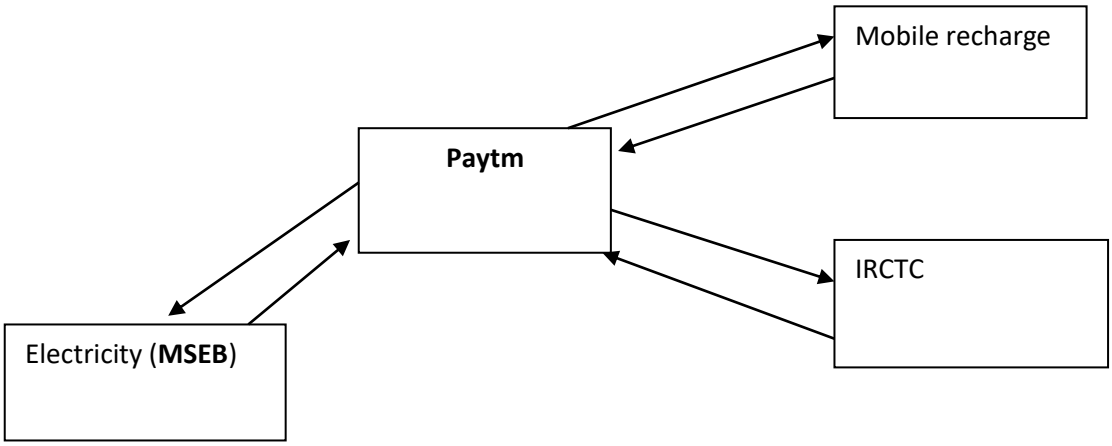
- Validating the application I **supporting the hardware** (Print, Blutthoot, etc)
- Configuration coverage testing also called hardware coverage testing

4. **I**nstallation coverage testing → **We have not done**

- I have not done, these testing

5. **I**ntersystem coverage testing-

- Validating application → **different service / application connect or data interchange**
- Ex. Paytm- Recharge mobile →



6. Sanitization coverage testing-

- Validating the application **extra feature/ Garbage values**
- Ex. Paytm → Recharge module → Mobile no. text box

Mobile no. text box- 10 digits

- Extra feature added → +91

Mobile no. text box- **+91** - 10 digits

- For extra feature/ Garbage values tester will raised defects

7. **Parrelaization coverage testing → We have not done**

- I have not done these testing

8. Globalizations coverage testing-

- Validating application is supporting to all langue's
  1. **Standard langue's/ Universal langue's** – English langue's support
  2. **Regional langue's –** Hindi, Marthi, Tamil, etc langue's support
- For regional testing we will Google translator

Re-testing-

- In Re-testing tester validating the **functionality of the application by passing multiple test data**
- Testing which is performed by **passing multiple test data, these testing called re-testing**
- Ex. Paytm – FastTag module → Multiples vehicle no. enter (4 tiers, 6 tiers, 8 tiers, 10 tiers, etc)
- Test data we will get from **Databases (SQL server) -**
- If we have old project → **Existing data in Databases (SQL server)** OR Tester in SIT environment will create the test data
- If we have new project → For testing we required test data then **Tester in SIT environment will create the test data**
- If we have project → Test data depends on other application → Ex. Phone pay – feature UPIID created → Mobile no., Bank No., Debit card, etc these data related to bank →These **Test data will provided by BA**
- **Ex.** Paytm → Credit card new functionality → Payment for bus ticket → **Credit card dummy card= 4000111100001111**

Regression Testing-

- In **re-testing or in BBT**, when **tester found a defects** then tester will raised to developer
- Then **developer fix or resolved the defect** and developer will sent **Modified build** and **tester will do the testing (regression testing)**
- Regression testing – **Re-execution** of test on **modified build**, to **check defect has been fixed/ work properly** & **there is no side impact on interconnected modules**
- **Regression testing = Regret + Action**
- **Ex.** Paytm – Recharge Module → Recharge module **build (V9.0 - 500 lines codes)** → Mobile working Recharge → VI, Airtel, JIO, MTNL but **BSNL mobile no. is not working- defect** (6 to 7 hr) → defect raised to developer → developer will fix or resolved **modified build (V9.0 - 550 lines codes),** BSNL should work → Developer will sent **modified build (V9.0 - 550 lines codes) →On modified build (**re-testing + Regression testing**)** tester will do **Regression testing →** check **defect has been fixed/ work properly, BSNL mobile no.** work (10 to 12 test data - multiples BSNL no.) (1 to 2hr) **& no side impact on interconnected modules i.e.** VI, Airtel, JIO mobile work

- **Regression testing will performed in 2 times**
  1. **In SIT environments -** if we found defects
  2. **Final Regression testing-** When application is moving from **one environment to another environment**

- **In Regression testing** we will **execute these test cases**
  1. **Failed test cases** (Ex. BSNL mobile no.)
  2. **High priority test cases** (Ex. VI, Airtel, JIO, MTNL mobile no)
  3. **Extra features test cases or Extra functionality test cases (**Ex. **+91** mobile)
  4. If times permits, we will remaining test cases

- **Regression testing** performed within 2 to 3hr

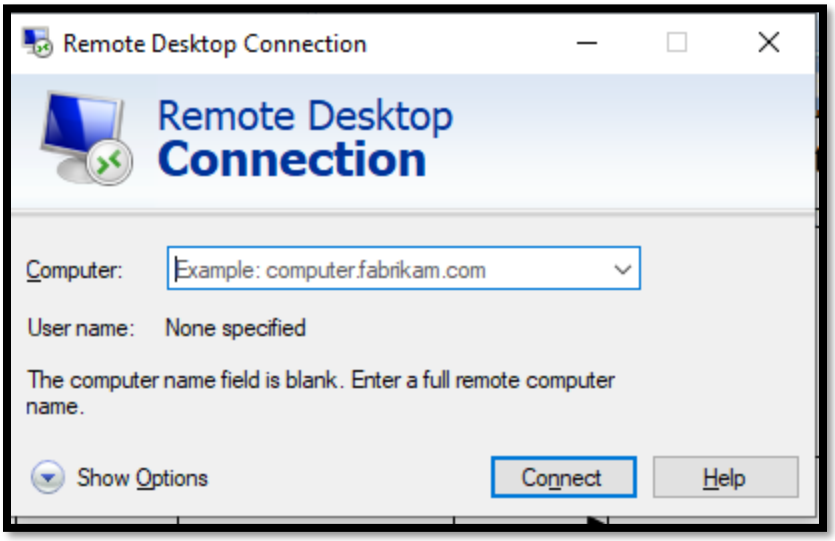
UAT/ Pre-product Testing-

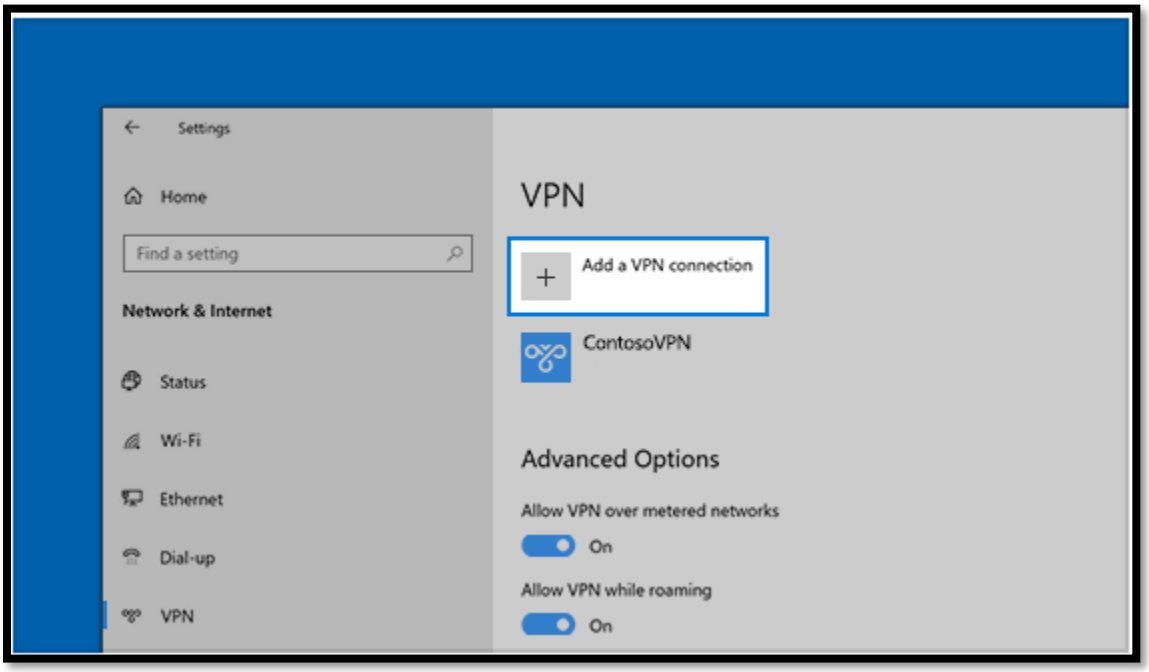
- When UAT testing, will start after completion of BBT in SIT environment
- Tester who is working in SIT environment →prepared the test proof and sent o UAT team
- **UAT (User acceptance testing)** also called **Custer acceptance testing**
- **UAT team works-** 2 developer + **1 tester (Client location)**
- **UAT check/testing system & functional**
- **UAT 2 types**
  1. **Alpha Testing**
  2. **Beta Testing**

Alpha Testing	**Beta Testing**
Alpha testing performed on s**ervice based application** Ex. Paytm/ Zerodha/ Upstock etc	Beta testing will performed for **product based application** Ex. Splitewise, Adoberedaer, Khatabook, etc
In UAT, for Alpha testing **developer + Tester are present**	In UAT, for Beta testing **developer + Tester are not present**
If defects/ Issue application→ **immediate fixed/ resolved**	If defects/ Issue application→ **fixed/ resolved in next version of the application**
Client interaction is more	**End User interaction is more**
Alpha Testing → check system & functional	Beta Testing → check system & functional

- In my project, we will do **Alpha Testing → Service based project**
- I have do these testing by **accessing remote desktop**
- Throw remote desktop, we will access UAT sever & do the testing
- Tester will not work in **both environments (SIT & UAT)**
- In project, **UAT tester is on leave (4 to 5 month) that I have done UAT testing.**
- I same project we have UAT testing also, these time I have not SIT environments

- In one module I have performed SIT testing & in another module we have performed UAT testing
- If we **found issue/defect in UAT testing**, UAT tester will raised these defects to respective developer & SIT Tester.
- SIT tester, he **re-produce the defects in SIT environment.**
    1. If defect found in SIT environment then we will inform developer and say that fix the defect ASAP
    2. If defect not found in SIT environment then SIT tester shows Test proof to UAT tester





Final Regression –

- **Regression testing will perfumed**
  1. **Final Regression testing-** When application is moving from **UAT to prod/ Live environment**

- **In Final Regression testing** we will **execute these test cases**
  1. **Failed test cases** (Ex. BSNL mobile no.)
  2. **High priority test cases** (Ex. VI, Airtel, JIO, MTNL mobile no)
  3. **Extra features test cases or Extra functionality test cases (**Ex. **+91** mobile)
  4. If times permits, we will remaining test cases

Production testing-

- After completion UAT, then developer will deployed the code/ build from UAT to production
- If we **found a defects in Production environment** these defects called **production defect**
- Production defect will occurred due to 2 reason
  1. If **Tester have missed functionality while doing the testing**, then these production defect "**Hot Fix**"
  2. If **Client has missed some functionality** and these defects found in production issue, then these production defect "**CR (Change request)**"

1. If **Tester have missed functionality while doing the testing**, then these production defect "**Hot Fix**"
   A. If defects/ issue found in production then Client will sent a "**Escalate Mail**" to project team → Tester will **re-produce the defects in SIT environment** → If defects/ issue is **found in SIT environment** → He will inform to developer and say fix these issue ASAP and deployment to client (Project Manager ask to Tester sent me apology mail / Reasoned why these has been happed against these production issue)
   B. If defects/ issue found in production then Client will sent a "**Escalate Mail**" to project team → Tester will **re-produce the defects in SIT environment → If defect is not present in SIT environment** → Tester will Mail/Call to PM, Developer & Designer, these issue is not present in SIT environment & attached test proof → Developer will **check the deployment process/ environment problem/ Configuration file** → Developer will these issue & deployment of client

2. If **Client has missed some functionality** and these defects found in production issue, then these production defect "**CR (Change request)**"
    A. If we will get CR from client then we will accept these CR but if CR is impact more on Developer, Testing & Production. Client inform
    B. If we will get CR from client then we will accept these CR and if CR is having the less impact Developer, Testing & Production. CR deployment & Testing and deployment to client.

Production Issue

You side (Company)
↓
Production issue (Hot Fix)
↓
Impact
↓
Development
↓
Testing
↓
Deployment / delivery

Client
↓
Production issue (CR)
↓
Impact
↓
Development
↓
Testing
↓
Deployment / delivery

Testing Terminology-

- Testing we will scenario/condition

Monkey Testing-

- If we have **more test cases (70 to 100 test cases) & we have less time (1hr to 2hr) for testing then Monkey testing OR** If we have **get the build at last moments (in Friday) & we have less time (1hr to 2hr) for testing we follow Monkey testing approaches**
- In Monkey Testing, we will **execute high priority/ Core functionality application/ build (system & functionality testing)**
- We will inform to client about miner defect may be present in the application/ build
- If we got issue/ defects (high priority/ Core functionality) in money testing then we can't deployed these US to client

1. If client say I want these deployment with these sprint, developer & tester will work Saturday & Sunday → Developer & test & deployment to client
2. If client say not you can deployment to next sprint

Exploratory Testing-

- If we have **more test data** but we have **less knowledge about the application** then **Exploratory Testing approaches**
- **Ex**. If your collogues is absent (2 days immediate leave) and testing will be done by another tester
- Tester will do testing by passing more data on the functionality & we will Exploratory (application functionality) functionality understand

Ad-hoc Testing-

- If we have **knowledge about the application** but we have **less test data** then **Ad-hoc Testing**
- Ex. Application payment tab → More credit care/ debit card for testing
- Credit card = 1 no & Debit card = 1no.

Priority & Severity-

- Priority & Severity terms it will be **defined against defect**
- **Priority** term → **defect impact on client business**
- **Severity** term → **defect impact on functionality application**
- Defect status on Priority → Very High, High, Medium & Low
- Defect status on Severity → Critical, High, Medium & Low

1. **High Priority & High Severity –**

- **Ex.** Login page is not working- defects
- **Ex.** Core functionality application in project - defects

2. **High Priority & Low Severity-**

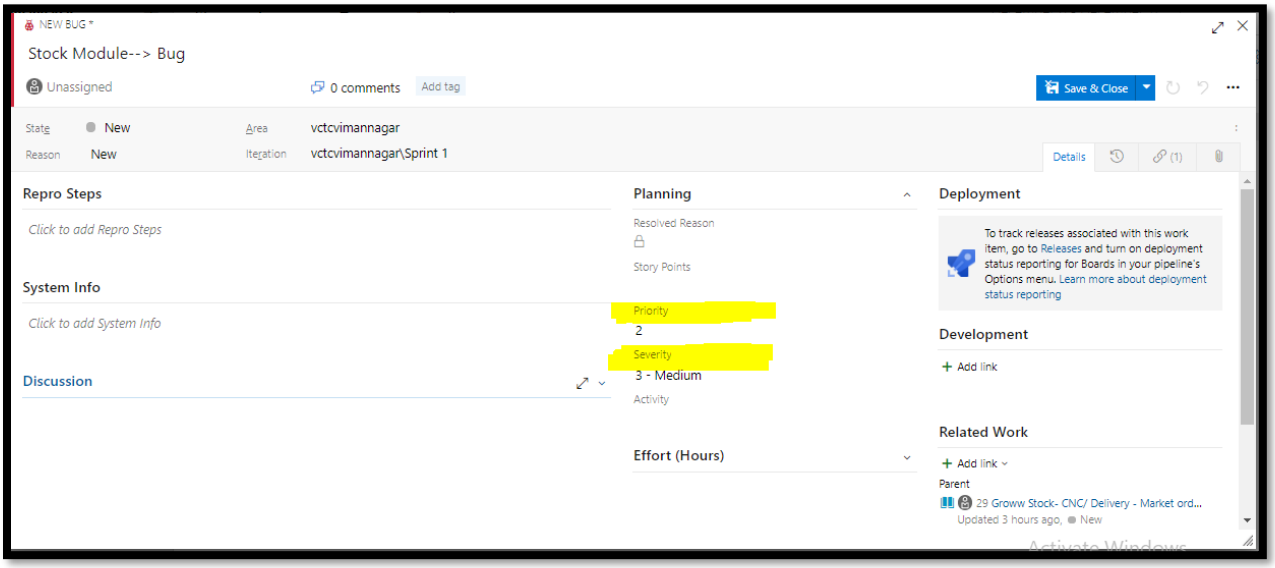
- **Ex.** Application client logo is not present
- **Ex.** Application text are overlapped with each other

3. **Low Priority & High Severity-**

- **Ex.** Application rarely used functionality is not working (Ex. Paytm – Promo code)
- **Ex.** Invoice download is not working in application

4. **Low Priority & Low Severity-**

- **Ex.** Spelling mistakes in Application (Ex. Submit text changes – Ok text)
- **Ex.** Button color is not proper (Ex. Button Blue color into red color)



- **Error-** If some wrong coding happens application, if we found mistakes in the application error
- **Defect –** In testing, if we found a error in the application these error are called defect
- **Bug-** If developer is accepting these defects then it is called Bug
- **Issue-** If we found major bug in the application these bug, it is called issue

Integration Testing/ End to End testing-

- Integration testing/ also called **end to end testing**
- Integration testing will **performed on complete module**
- Integration testing performed by Tester

Interview question-

1. What testing you have performed in the project?
2. What is System & functional testing
3. What is difference between system & functional testing
4. What is boundary values analysis, ECP, Decision table testing technique?
5. What is retesting & Regression testing?
6. Which testing you will perform after fixing a defects?
7. What is UAT testing & how you worked in that?
8. What is final regression testing?
9. What is hot fix & have you handled any hot fixed?
10. Give me the example for high Priority & high Severity?
11. Give me the example for high Priority & Low Severity And Low Priority & High Severity
12. What is your approach, if we have less time and more test cases to be executed?
13. What is your approach, if we your colleagues' is on leave and you have to do testing
14. What is error, bug, defects & issue
15. What is an approach if we found issue/ defects in UAT?
16. Who is responsible for deployment?

- **Answer-** Deployment/ Delivery all will be done by developer

17. When you will do the Deployment/ Delivery to client?

- **Answer-** Saturday

18. What is your approach, when SIT environment is crash?

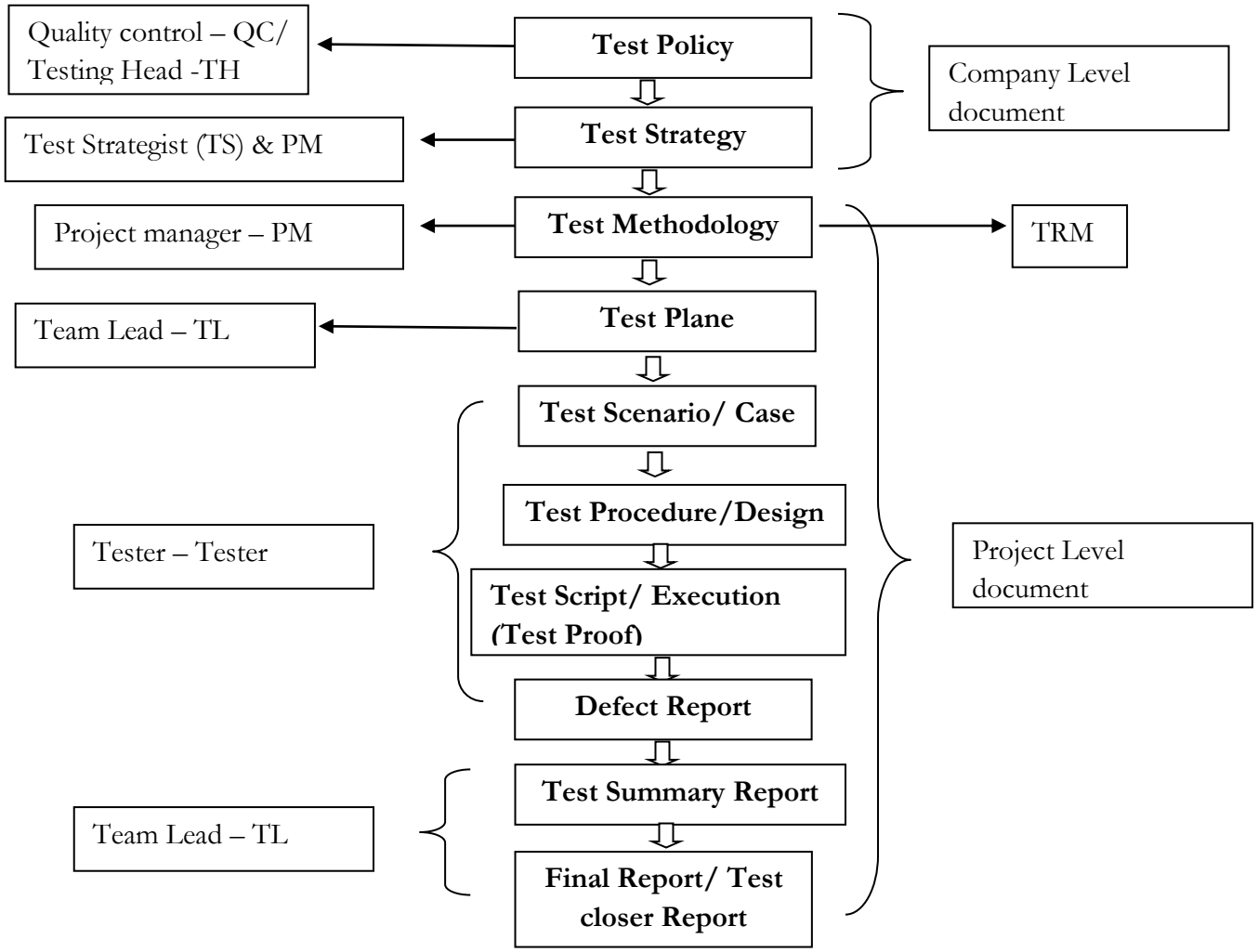
- **Answer-** If client want Deployment with the sprint and SIT environment is crash, then tester will access the UAT environment.
- **UAT** environment access throw remote desktop

10000 line (Master Braches) → Rent payment (500 line code) = 10500 line

Manual Part 2

Q. What is your organization Test documentation?

- Test Document hierarchy



Test Policy-

- **In Test policy stage,** Testing Head & Quantity control peoples will work
- In Test policy documents **defines objective of the project**
- Simple term how much revenue we will generated from the project
- Test policy documents, it is company level documents

Test Strategy-

- Test s Strategist person will work in Test Strategist
- Test Strategist documents defines **which strategy we will consider for the project**
- It is company level documents

Test Methodology-

- Test s **Methodology** documents will be **prepped by PM**
- Test **Methodology defines mapping between development stages & testing factor**
- Test **Methodology stage PM will papered a TRM (Test reasonability matrix) documents**
- **TRM documents also called as Test matrix**
- It is Project level documents
- While preparing TRM document, focus on parameters
    1. Functional / requirements of application
    2. Risk in the Project

Development (Stage) → **Testing factor ↓**	BRS, SRS	Design	Coding	Testing	Support
Requirement phase Testing	Yes	Yes	Yes		
UI Testing	No	Yes	Yes	Yes	Yes
Functional Testing	No	No	Yes	Yes	Yes

Test Plane-

- **Test plane** documents will be prepped by Test Lead
- **Test plane defines → Job allocation, Resources allocation, estimation**
- It is Project level documents

Test Scenario & Test cases design, Test cases execution-

- According to US tester will identify the Test **Scenario & Test cases**
- **TCD will design against the US**
- Test **Scenario & Test cases** will be prepped by Tester
- It is Project level documents

Defect Report-

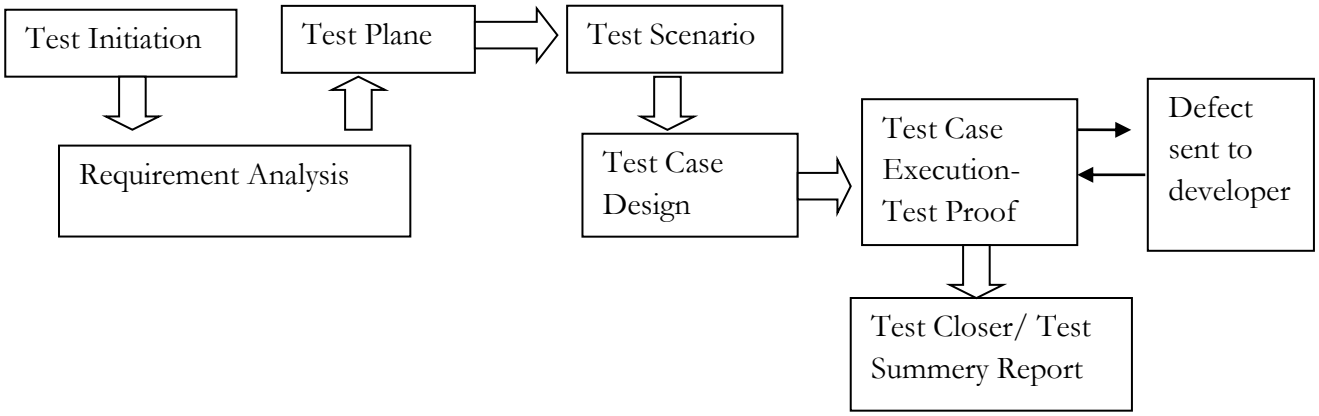
- In TCE, if we found defects, the we will created defects in JIRA/ TFS
- Inform to Developer throe Mail
- After defects fix then we will performed e-testing & regression testing
- It is Project level documents

Test Summery report & Test closer report-

- **Test summery report –** In a sprint how much US test, TCD, TCE, TCS & defect found
- **Test closer report – Sprint wise report (Sprint1, Sprint2, Sprint3, etc)**
- **Test Summery report & Test closer report will prepared by test Lead**

Software Testing Life Cycle (STLC) –

Q. What is your organization Test process?
Q. What is STLC?



Test Initiation-

- In Test Initiation stage **PM will work**
- PM will **prepared TRM**
- After completion of TRM, PM will sent to Test lead

Test Plane-

- Test plane will **prepared by test lead**
- Test plane will contains **– Job allocation, resource allocation & estimations**
- Test planes will contains – **who will test, when to test, how to test, defect, etc**

Test Scenario & Test Case Design-

- For a US tester will identify test Scenario
- Test cases design from test Scenario
- Test cases design we will write in excel sheet

Test Case Execution-

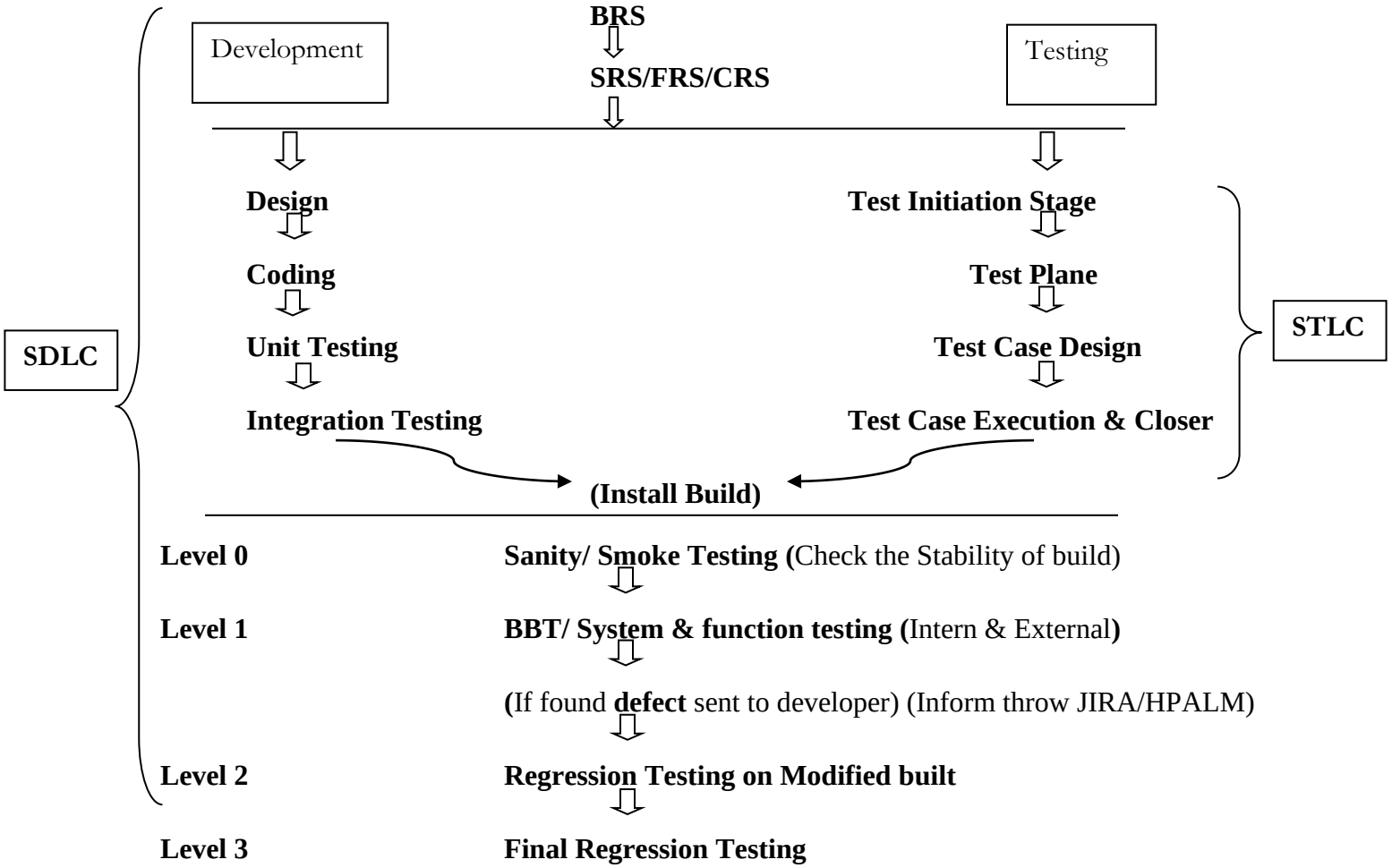
- In TCE, when we got build from developer then **we will start the TCE**
- In TCE, if we found defect then we will **create defects in JIRA/TFS tool & inform to developer throw Mail**

- After fixing defects we will perfumed **re-testing & regression Testing**

Test Closer/ Test Summery Report-

- Test Closer/ Test Summery Report will be **prepared by test lead/ Team lead**
- Test Closer/ Test Summery Report will contains Sprint → US – TCD,TCE, TCS, defect, etc

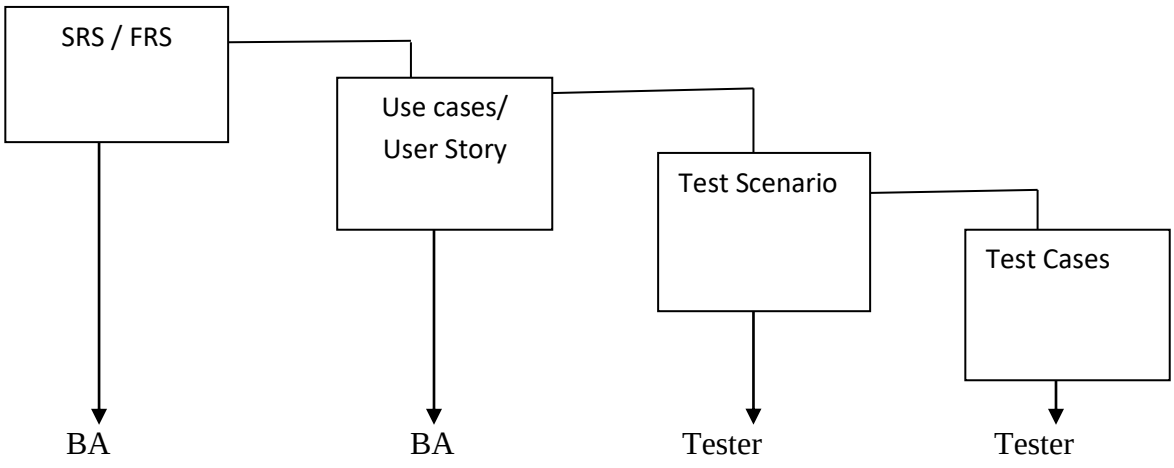
Testing Process-



Test Plane-

- Test plane will be **prepared by Test lead**
- While prepping Test planes will consider following point
  1. Job allocation
  2. Resource allocation
  3. Estimation
- Main purpose of Test plane is **decide start & end date of testing**
- **Test plane contains-**
  1. Test plane ID
  2. Test items
  3. Feature to Tested
  4. Feature not to be Tested
  5. Test pass/ fail
  6. Test deliverable (Test cases design → TCE)
  7. Test Environment
  8. Critical bug/ Issue/ Major blocker/ Defect life cycle
  9. Testing Task
  10. Staff training (New module)
  11. Roles & Responsibility
  12. Sing & Approvals

Test scenario & Test cases-



SRS & Use Case -

SRS-

- SRS defines functional requirement to developer & System requirement that will be used
- SRS will be **prepared by BA**
- SRS **derived from BRS**
- SRS will contains
    1. Functional requirement
    2. Functional full diagram
    3. Use case
    4. Screenshot

Use Case-

- Use cases defines a single/ specific requirement
- Use cases will be **prepared by BA**
- Use cases **derived from SRS**
- Use cases will contains
    1. Description
    2. Acceptance criteria

Test Scenario-

- Test scenario will be **prepared by Tester against the US**
- Test scenario will be **derived from US**
- Test scenario defines **"What to Test"**
- Test scenario defines **ways to test the functionality**
- Form one Test scenario, multiples test cases will prepared
- Test scenario defines → **High level test cases**
- Test scenario **always defines in +ve ways**
- **Ex. US-** Login Page – Username text box, Password text box & Submit button
- Description- Username accept Mobile no & email id. Password – 4 to 6 charter
- **Test scenario-**

1. Verify username text box by passing valid mobile no
2. Verify username text box by passing valid email id
3. Verify Password text box by passing valid 4 to 6 charter combination

Test case-

- Test cases will be **prepared by Tester against the Test scenario**
- Test cases will be **derived from Test scenario**
- Test cases defines **"How to Test"**
- Test cases defines **Input, Process, Output**

- Form one Test cases, Test data, Result, etc
- Test cases defines → **Low level test cases**
- Test cases **always defines in +ve ways & -ve ways**
- **Ex. US-** Login Page – Username text box, Password text box & Submit button
- Description- Username accept Mobile no & email id. Password – 4 to 6 charter
- **Test scenario-** Verify username text box by passing valid mobile no
- **Test cases-**
  1. Verify username text box by passing valid Airtel mobile no
  2. Verify username text box by passing valid BSNL mobile no
  3. Verify username text box by passing valid JIO mobile no
  4. Verify username text box by passing valid VI mobile no
  5. Verify username text box by passing valid MTNL mobile no
  6. Verify username text box by passing In-valid Airtel mobile no
  7. Verify username text box by passing In-valid BSNL mobile no
  8. Verify username text box by passing In-valid JIO mobile no
  9. Verify username text box by passing In-valid VI mobile no
  10. Verify username text box by passing In-valid MTNL mobile no
  11. Verify username text box by passing valid landline no
  12. Verify username text box by passing Null/Blank mobile no

- **Test scenario** - Verify username text box by passing valid email id
- **Test cases-**

1. Verify username text box by passing valid gmail id
2. Verify username text box by passing valid outlook id
3. Verify username text box by passing valid rediffmail id
4. Verify username text box by passing valid yahoo id
5. Verify username text box by passing valid HSBC id
6. Verify username text box by passing In-valid gmail/ outlook/ rediffmail/ yahoo id
7. Verify username text box by passing Null / Blank id

- **Test scenario** - Verify Password text box by passing valid 4 to 6 charter combination
- **Test cases-**
  1.

Interview Question-

1. What is difference SDLC & STLC
2. What is testing process in your origination

3. What is Agile Test process?
4. What is Test Plane & have you prepared a Test plane?
5. What is difference SRS & Use case
6. What is difference Test Scenario & Test Cases

Test Cases-

- Test cases will be written by Tester
- While preparing Test cases, we will consider
    1. **Business logic related test cases**
    2. **Input domains related test cases**
    3. **UI/GUI related test cases**
- **In my project**, we will **write Test cases in excel sheet.**
- **Ex. US**

User Story- 30074	
Description	**Acceptance criteria**
Login page of VCTC application contains Login page/ tab username text box, password text box & sign in button. This login page will provided for VCTC student & staff login, from which they login into their VCTC account page. Login page will contains all page as per standard color as suggested.	Username text box accept only mobile no. Existing in your domain
	Username text box will unmasked
	Username text box doesn't accept other then VCTC mobile number (error message- enter valid data)
	Password should contains of 6 to 8 char combination
	Password text box will accept One Capital letter, One Small letter, One special char, One number
	Password text box will masked
	Password text box doesn't accept One special char, One number these combination (error message- enter valid data)
	Password text box doesn't accept One Capital letter, One number these combination (error message- enter valid data)

Test scenario-

1. Verify username text box by passing mobile no
2. Verify Password text box by passing valid combination of charter
3. Verify submit button by passing data in username & Password text box

Test cases-

Test scenario- Verify username text box by passing mobile no

1. Verify username text box by passing valid student Airtel mobile no
2. Verify username text box by passing valid student BSNL mobile no
3. Verify username text box by passing valid student JIO mobile no
4. Verify username text box by passing valid student VI mobile no
5. Verify username text box by passing valid staff Airtel mobile no

6. Verify username text box by passing valid staff BSNL mobile no
7. Verify username text box by passing valid staff JIO mobile no
8. Verify username text box by passing valid staff VI mobile no
9. Verify username text by passing valid data→ EC= Mobile no should be unmasked
10. Verify username text box by passing In-valid staff VI/ Airtel/ BSNL mobile no → EC= Error message should display “Enter valid data”
11. Verify username text box by passing In-valid student VI/ Airtel/ BSNL mobile no → EC= Error message should display “Enter valid data”
12. Verify username text box by passing charter → EC= Error message should display “Enter valid data”
13. Verify username text box by passing Null/ blank → EC= Error message should display “Enter valid data”
14. Verify username text box by passing mobile no of other than 10 digits blank → EC= Error message should display “Enter valid data”

Test cases-

Test scenario- Verify Password text box by passing valid combination of charter

1. Verify Password text box by passing valid charter combination (Aa@123)
2. Verify Password text box by passing valid charter combination (aA@123)
3. Verify Password text box by passing valid charter combination (@Aa123)
4. Verify Password text box by passing valid charter combination (1Aa@23)
5. Verify Password text box by passing valid charter combination (AA@@11aa)
6. Verify Password text box by passing valid charter combination (aa@AA3)
7. Verify Password text box by values → EC= All enter values should be masked
8. Verify Password text box by passing in-valid charter combination (A@123) → EC = error message should be display “enter valid data”
9. Verify Password text box by passing in-valid charter combination (a@123) → EC = error message should be display “enter valid data”
10. Verify Password text box by passing in-valid charter combination (@123) → EC = error message should be display “enter valid data”
11. Verify Password text box by passing in-valid charter combination (Aa@) → EC = error message should be display “enter valid data”
12. Verify Password text box by passing in-valid charter combination (Aa@1) → EC = error message should be display “enter valid data”
13. Verify Password text box by passing in-valid charter combination (AAAaaa@@@123) → EC = error message should be display “enter valid data”

- **Test cases Excel sheet fields-**
  1. Sr. No.
  2. Test case ID
  3. Priority
  4. Reference
  5. Test scenario
  6. Pre-request
  7. Test data
  8. Test cases
  9. Test steps
  10. Expected result
  11. Actual Result
  12. Pass/ Fail

- **Writes Test cases for following things**
  1. Whatup, Instgram, Facbook (Meta), Flipkard/ Amzone
  2. Pen, Chair, Table, Fan, Laptop, Mobile, ATM, etc
  3. Scenarios – Collage degree admission form, Page/ Home side test case
- **https://artoftesting.com/chair**

- **Test cases Whatup (Group message)-**

1. Verify side of sent text message in the group → EC= Sent text message will shows on right side
2. Verify times present in sent text message in the group → EC= Current Times should display of Sent text message
3. Verify single tick marks present in sent text message in the group → EC= Single tick should display of Sent text message when message not deliver to all members
4. Verify double tick marks present in sent text message in the group → EC= Double tick should display of Sent text message when message will deliver to all members
5. Verify color of double tick marks present in sent text message in the group → EC= Double tick should display of Sent text message when message will deliver & read to all members
6. Verify advance feature present in sent text message in the group → EC= Advance feature will be as Message info, Delete message, Forward message, Replay, Star Message
7. Verify Message info advance feature present in sent text message in the group → EC= Message info in Advance feature will shows information about delivery, Read, un-delivery, time also shows

8. Verify Delete message advance feature present in sent text message in the group → EC= Delete message Advance feature will shows pop which contains Cancel, Delete for Me & Delete for everyone
9. Verify Cancel button in Delete message advance feature pop present in sent text message in the group → EC= Delete message Advance feature pop will shows
10. Verify Delete for Me button in Delete message pop present in sent text message in the group → EC= Delete message Advance feature pop will detect message and shows pop about delete message
11. Verify Delete for everyone button in Delete message pop present Delete message advance feature in sent text message in the group → EC= Delete message Advance feature pop will detect message and shows pop about delete message & Message about defect
12. Verify Reply advance feature present in sent text message in the group → EC= Reply advance feature will shows sender message & name
13. Verify Forward message advance feature present in sent text message in the group → EC= Reply advance feature will shows sender message & name
14. Verify Star advance feature present in sent text message in the group → EC= Reply advance feature will shows sender message & name

Test cases review-

- Review defines **correctness & completeness of the documents**.
- 4 type of Test cases
    1. Self review
    2. Peer review
    3. Internal review
    4. External review

1. Self review-

- Tester will do test cases review, these review is called **self review**

2. Peer review/ Cealage review-

- **Test Lead/ Senior Tester** will do review the test cases
- Tester will sent the **Mail to Test Lead/ Senior Tester**

3. Internal review-

- **In Internal review, BA people** will review test cases
- Tester will sent the **Mail to BA**

- **BA set up the meeting** with Tester for test cases review

4. **External review-**

- In External review- **Client/ UAT tester** will review the test cases
- Tester will sent the **Mail to Test Lead**
- In External review for test cases review for that **Test lead set up meeting** with **Client/ UAT tester**
- In these meeting, Tester will shows/ explain their test cases to Client/ UAT Tester

- **In My project,** We have done **Internal Review / External Review**

- In review, If we **got the comments/suggestion from client/ UAT Tester/ BA→** Tester will accept these comments/suggestion →**In Excel sheet**, we will accept these comments/suggestion and write in **comments/suggestion column** →Tester, will change test case as per comments/suggestion

- **Which parameter consider in Test cases review-**

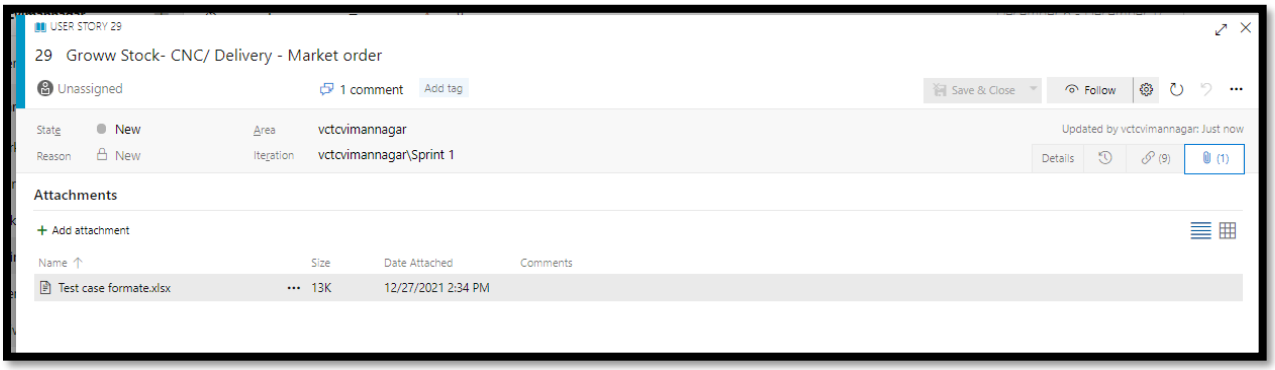
1. Test case should be **simple**
2. Test cases should be **understandable**
3. Test cases should be **functionality against the US**
4. Test cases should be **business logic related test cases**
5. Test cases not should be **duplicated**
6. Test cases should **follow standard format**
7. Test cases should **grammatically correct**
8. Test cases should be with **spelling mistake**

Test cases execution-

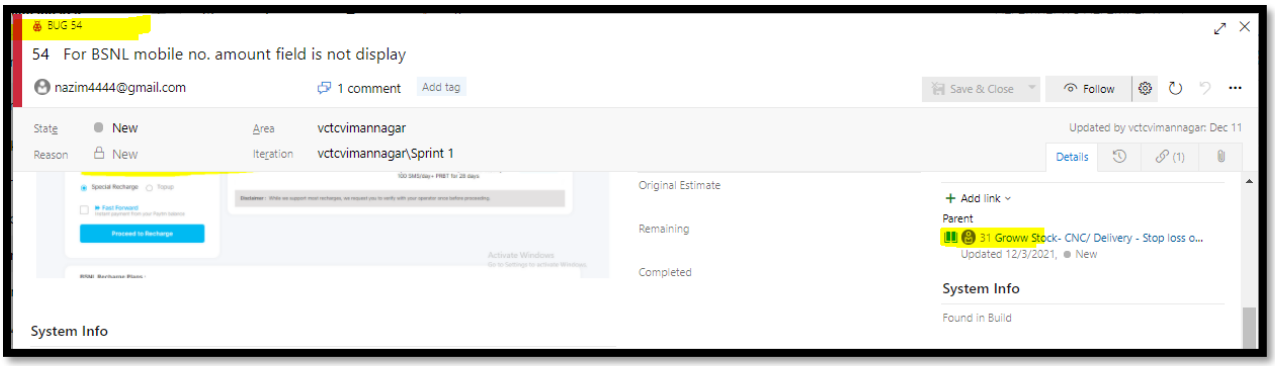
- As per Test cases functionality test → TCE, **Tester will prepared → Test Proof**
- In TCE, if we found defects then raised/ create in JIRA/ TFS tool. Inform to developer throw Mail (JIRA/ TFS)
- Test proof will contains- Test cases against Screen shot

Tractability Matrix-

- Traceability matrix → **Test cases & defects link/mapped with US**
- 2 types Traceability matrix
  1. **Forward Traceability matrix-**
     - In Forward Traceability matrix, **Test cases should be link/mapped with the US**
     - Test cases written in excel sheet, **these excel sheet attached with US**

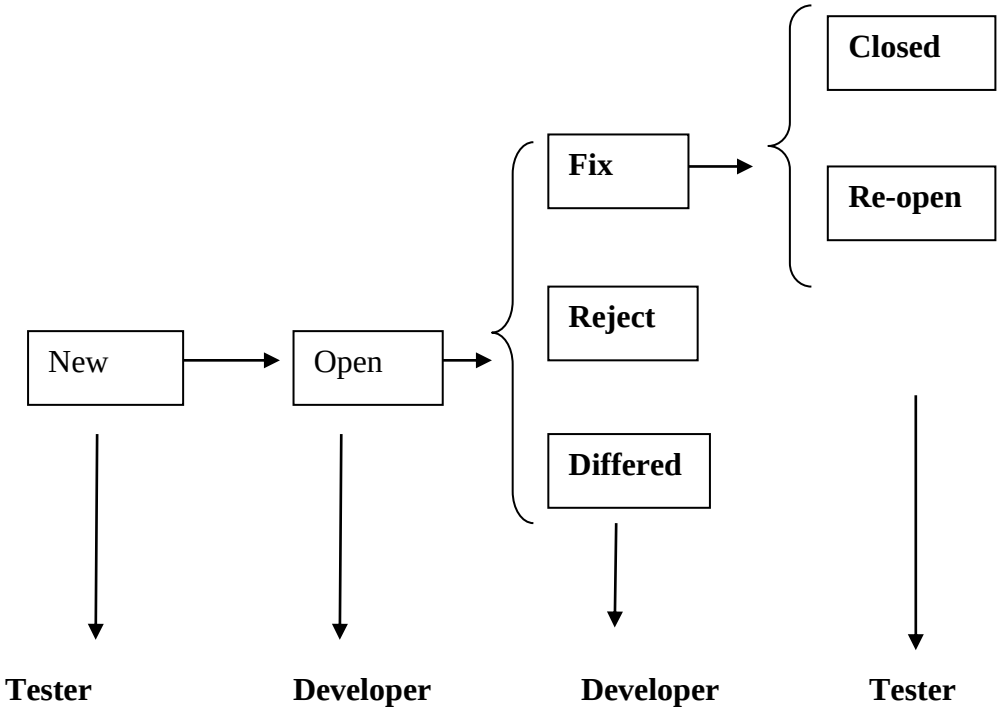


2. **Backward Traceability matrix-**
   - In Backward Traceability matrix, **defects should be link/mapped with the US**
   - Defects will be created in JIRA/ TFS tool. In **JIRA/ TFS tool defects link with US**



Defect-

- Defect – When tester found error in application then it is called as defects
- **Defect life cycle –** The **journey of the defects** from their start to end it is called defects life cycle



- **New-** When **tester will found defect** in the application/ build, then tester will create a defects in JIRA/ TFS tool & mark defect **status = New**
- Tester will inform to developer throw Mail

- **Open-**When developer is looking into these defects, developer will mark defect **status = Open**

- **Fix –** When developer found it is valid defect then developer will fix and sent to tester & mark defect **status = Fix**
- Developer will inform to Tester throw Mail
  A. **Closed –** On fixed defects, tester will do re-testing & regression testing. If we found that defects has been working fine & mark defect **status = Closed**
  B. **Re-Open –** On fixed defects, tester will do re-testing & regression testing. If we found that defects has not been working fine & mark defect **status = Re-Open.** Tester will inform to developer throw Mail

- **Reject -** When developer checking the defect & found that defects it is invalid defects & mark defect **status = Rejected**
- Developer will inform to Tester throw Mail
- **Differed –** When developer found, these **defect can't be fixed in current sprint. These defects has been** mark defect **status = Differed**

- Defect **status as Differed, will be decided by PM, BA, Designer**
- **Differed defects** will be move into next Sprint. **PM** will prepared a US for these Differed defects, **US = "Carried over bug"**

Duplicate defects-

- Duplicate defects are same defects created with different test data
- Ex. Paytm – Recharge Module → Airtel mobile no. not working – tester will create defect in JIRA/ TFS tool defect -1021→ BSNL(maha) mobile no is not working – defects created – 1022, BSNL(MP) mobile no is not working – defects created – 1023 (Duplicate defects), BSNL(UP) mobile no is not working – defects created – 1024 (Duplicate defects)

Re-producible defects-

- Re-producible defects are those **which is producing again and again**
- For Re-producible defects, In JIRA/ TFS tool we will marked as Re-producible defects = Y/ N

Blocker defeats/ Show stopper defects -

- Those defect, **which block the testing (TCE)**
- Ex. Paytm – Recharge Module → mobile no. not accepting – tester will create defect in JIRA/ TFS tool defect -1021

Bug leakage-

- Defect those found which is **missed from SIT environment & found in UAT environment**
- Tester in SIT environment, we will re-produced the defect

Report-

- In my project,
  1. Test cases execution → Tester → Test proof
  2. Defect Report → Tester → Tester All defects related to 4US
  3. Test summery report → Test Lead → Current sprint (TCD, TCE, TCE, defect)
  4. Test closer report → Test lead → Module wise report for testing

Defect Report-

Defect ID	Defect Description Title	Created By	Assigned To	State	Root cause
61295	Before 8:00 off-peak and student limited contract access the access application	Malik Patil	Akit partil	Done	configuration
61351	on public holiday access did not get for Off-peak and student limited contract	Malik Patil	Akit partil	Done	Coding Issue
59415	when create one year contract using MembershipAPI for created contract showing wrong initial period dates and increase dates	Malik Patil	Akit partil	Diffred	Conding error
61431	when create one year and two year contract using MembershipAPI for created contract showing wrong end dates and increase dates	Malik Patil	Akit partil	Rejected	Deployment Issue

Test summery report-

Sprint	User Story ID	User Descrip[tion	Assigned To	Test case Desgin	Test case Execution	Test case Skipped	Defect
Sprint 21	GR0023	Groww with Auto pay options	Maik patil	38	38	0	5
Sprint 21	GR0031	Groww with Mf	Maik patil	23	23	0	4
Sprint 21	GR0032	Groww with Mf	Maik patil	19	19	0	5
Sprint 21	IPO0021	IPO with Mf phase 1	Ankit Rodge	21	20	1	8
Sprint 21	IPO0022	IPO with Mf phase 2	Ankit Rodge	27	27	0	6
Sprint 21	IPO0023	IPO with Mf phase 3	Ankit Rodge	23	23	0	3
Sprint 21	GR0036	SGB with Mf	Priti Patil	35	32	3	2
Sprint 21	GR0037	SGB with Mf	Priti Patil	23	23	0	3

Test closer report –

- Module wise report for testing

Interview question-

1. What is different test cases review? Which is following in you origination?
2. What you will do if we got comments or suggestion in test cases review?

3. What is Tractability matrix? How you will prepare?
4. What is defect? What is defect life cycle?
5. Who will decide defect status as differed?
6. What is duplicate defect, re-produce defect & Block defect?
7. How many reports you will prepare in project?