# Testing topics/ Syllabus-

* **Manual Testing part 1-**
* **Process** – SDLC, V-module/ Process, **Agile Process/ Module (90 to 95%)**
* **Different types testing** – Sanity testing, Smoke testing, System & functionality testing, Re- testing & Regression testing, etc
* **Manual Testing part 2-**
* **Tester real work** –Test cases design (TCD), Test cases review, Test cases execution, Defect, different types Report, etc
* **Databases testing –** SQL server
* **API Testing –** Manual testing
* **Project management tool/ Bug tracking tool –** JIRA/ Azure Devops tool/ HPALM, etc
* **Automation –** Java languages + Selenium tool
* **Framework** 🡪 Data driver, Hybrid, BDD/ Cucumber, etc
* **Rest assured** (API Testing automation) / **Cypress tool** / **Mobile Testing**
* **2 Project domain – Different types (**Retail, Banking, E-learning, Medium, Invements, Telecomm, Saleforce, etc)
* **Daily Work** – Topics discussion, Regularly class, Saturday, Weekend mock
* **Weekend mock/Interview (15 Mock)-English** **Langue’s**🡪 Real IT interview on Saturday/ Sunday
* **Client (HSBC Bank/ USB Bank- USA)** – Work application 🡪 Infosys, Pune India 🡪 Employees work (degrees, comm, etc)

# Team Size-

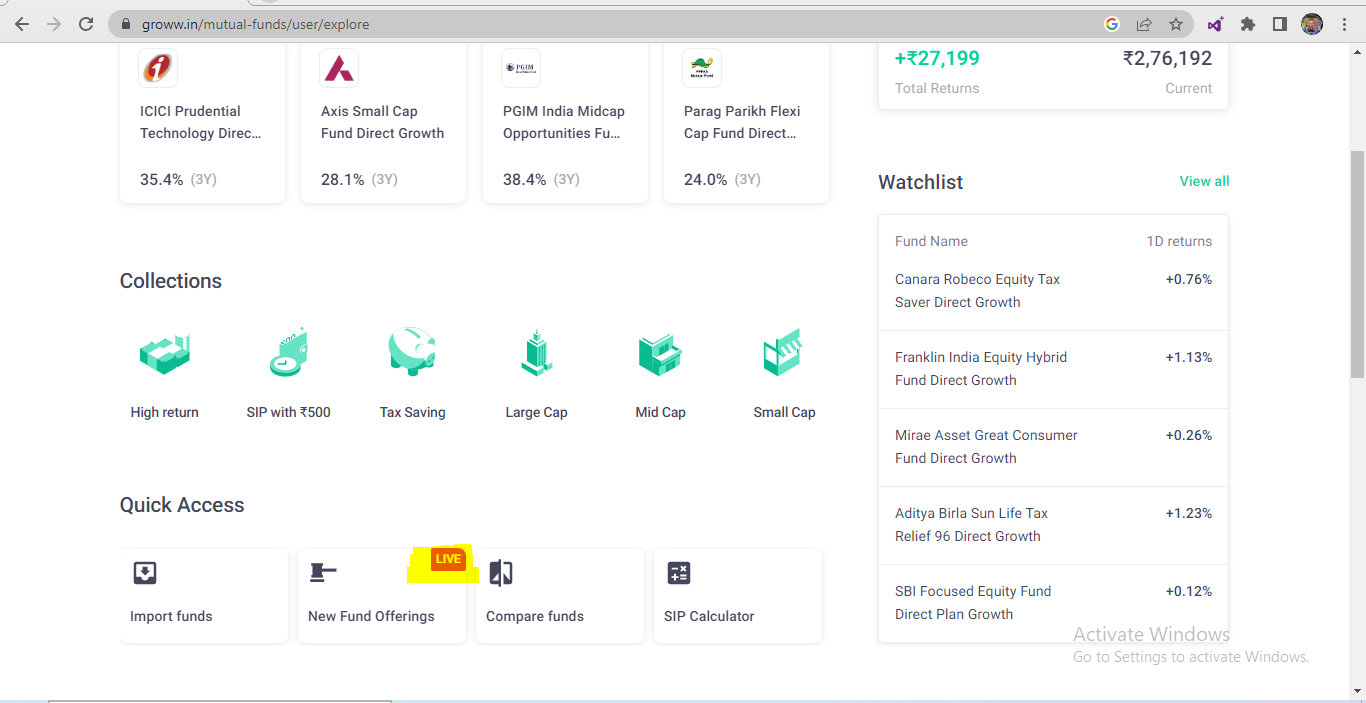
* Project 2 types of team will work

1. **Project team** – New functionality/ New feature/ New module added in application /project

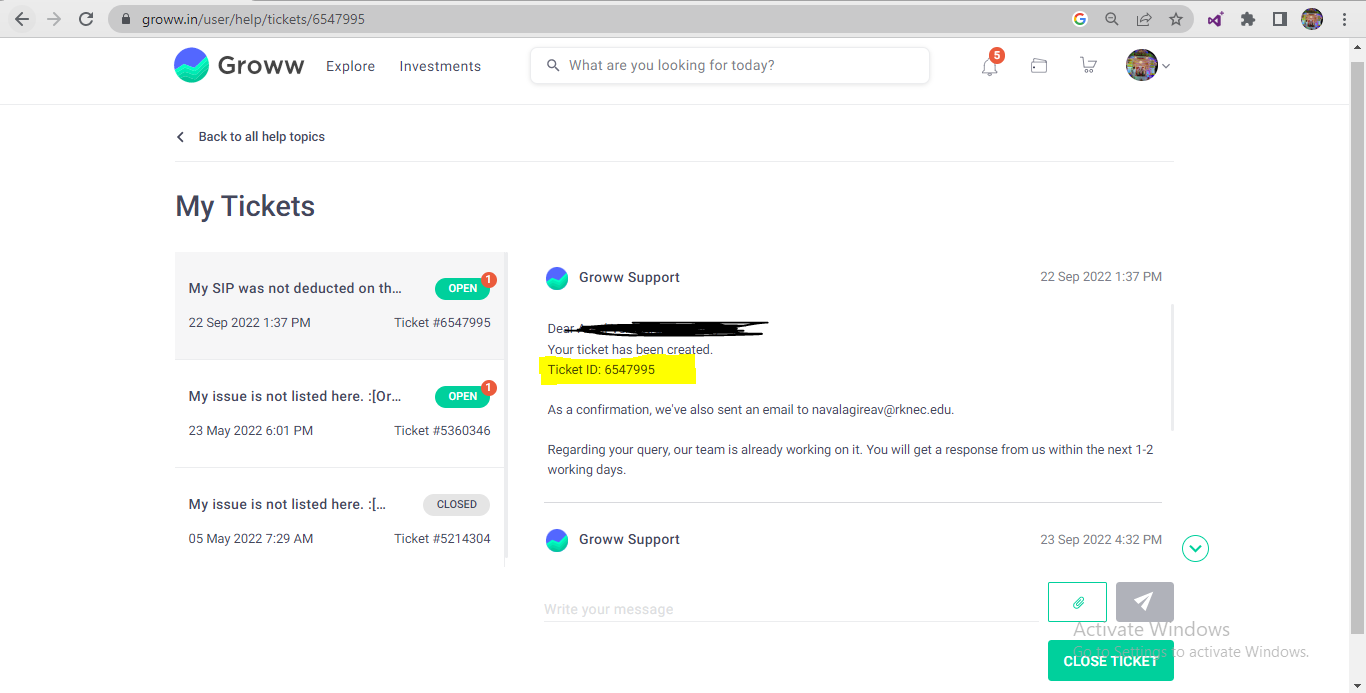
* **Ex**. Flipkard – New functionality/ feature – Glossary session, Profile page/ tab, etc

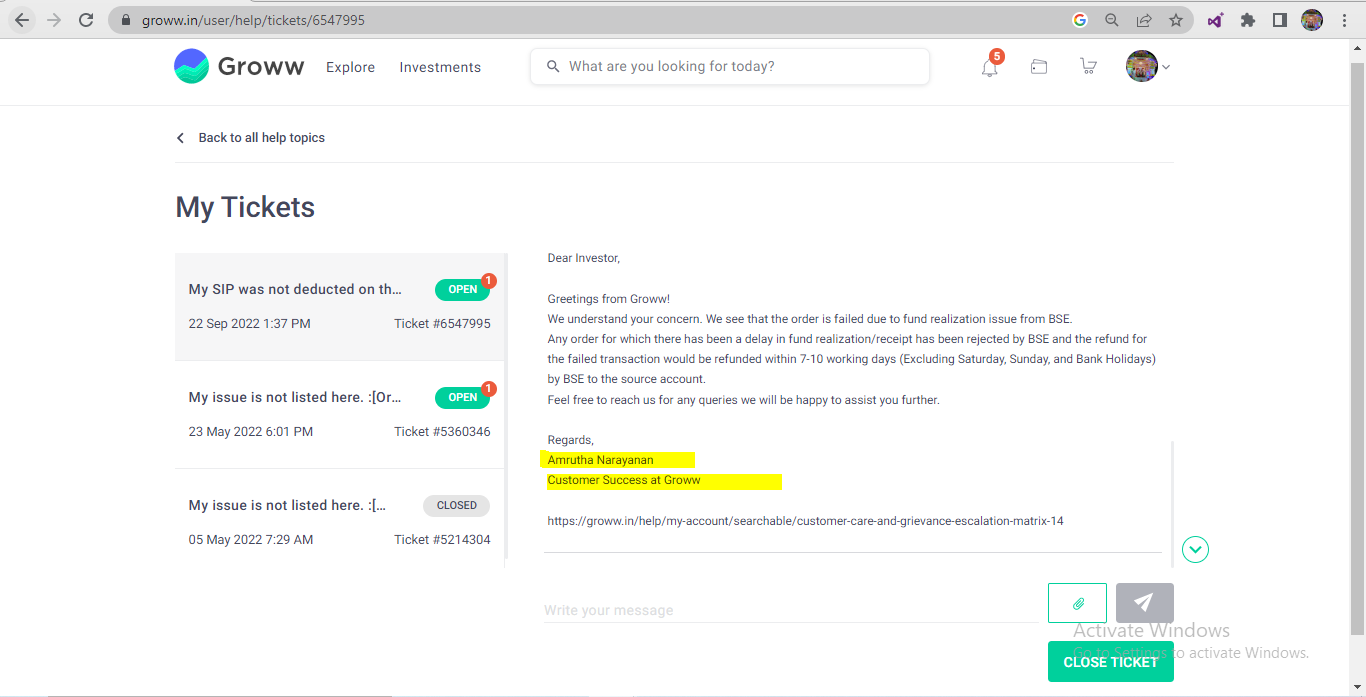
1. **Support team** – Existing application/ project Support - Defect/ bug fixed, Ticket, End user quires, enchantment, etc

* **Ex**. Phone pay – End user quires, Customer care call – ticket, Defect/ bug fixed, etc
* **Project team – (Team size = 14 to 15 people)**
* **Client –** Project / application / software development/ work (USA)
* **Delivery manager (DM) (1) –** To track/ check properly delivery to client
* **Project manager (PM) (1) –** PM is boss of project, Work allocation, Task allocation, etc
* **Business analyst (BA) (1) –** BA will always communicate to the client, for their requirement, etc
* **Designer / Sol architecture (1) (Developer exp. 10+) –** Project design against requirement
* **Developer (7 to 8 peoples)** (3 to 7 exp.) **–** Requirement against coding
* **Testing (2 to 3 peoples) –** Testing (Test cases design, Test cases execution, defect) against requirement

****

* **Support team – (Team size = 5 to 6 people) -**
* **Customer support manager (1) –** Defect/ bug raised, Ticket, End user quires, enchantment – work assignee
* **Developer (3 to 4 peoples)** (3 to 7 exp.) **–** Coding against Defect/ bug fix, Ticket resolving, enchantment, etc
* **Testing (1 to 2 peoples) –** Testing (Test cases design, Test cases execution, defect) against Coding against Defect/ bug fix, Ticket resolving, enchantment, etc

****

****

* **In my project,** I have worked in **Project team**

# Process/ Module-

* Process – Software implanted process types stage
* Different types

1. SDLC process/ Module
2. Waterfall process/ Module
3. V- process/ Module
4. **Agile process/ Module** (90 to 95%)

* Process decided –

1. If your client has own IT department – Client will decided

* **Ex**. HSBC, Paytm, Citi bank, etc

1. If your client don’t has own IT department – Company will decided

* **Ex**. Cred, Khaebook, etc

**HSBC Bank (Client) 🡪**

**HSBC software development company – Client IT**

**Vendor 🡪 Capgamini, TCS, Wipro, Infosys, etc**

# SDLC (Software development life cycle)-

* SDLC will contains 6 stage

1. Information gathering
2. Analysis
3. Design Project team work
4. Coding
5. Testing
6. Maintenance / Support 🡪 Support team

**Information gathering-**

* **In Information gathering, BA will work**
* BA will communicate to the client for collecting their **business related requirement**
* BA will prepared a documents, these documents are called as **BRS** (business requirement specification)
* Developer & Tester will not get these BRS documents
* **Ex.** Client business requirement – Application – Platform provided end user – End use bidding (Cricket, Kadbai, football, etc) – Bidding charge – Winner (total 90%) & 10% broker charge end user

**Analysis-**

* In analysis stage, **BA peoples will work**
* BA will communicate to the client for collecting their **software/ system related requirement**
* BA will prepared a documents, these documents are called as **SRS** (**software requirement specification**)
* SRS also called as **FRS** (functional requirement specification) / **CRS** (Customer requirement specification)
* SRS documents will contains

1. **Functional requirement –** (SRS = 1 module = Grocery module = 20 requirement)
2. **Functional flow diagram –** (requirement flow)
3. **Use Cases –** (1 requirement)
4. **Description –** 1 requirement all detail defines
5. **Acceptance Criteria –** Does & Don’t about 1 requirement
6. **Screenshot / Prototype –** Image / dummy design for these functionality

* After completion of these SRS documents, BA will send these documents to Developer & Tester. Developer & Tester will read/ understand these documents.
* If we have **doubt/ clarification against SRS** documents, Developer &Tester will set upon meeting with BA peoples.

**Design-**

* In design, Designer will work
* Designer will do the coding **against the HLD (High level design)**
* Project – Front end, Back end/ DB, Services/ API - Configuration – project functionality work

**Coding –**

* In Coding, Developer will work
* Developer will do **coding on** **LLD (Low level design)**
* Developer will of **coding against the requirement**/ functionality, these application/ software knows as **Build** (Under development / Testing)
* Developer will sent these build to Testing

**Testing –**

* In Testing, tester will work
* Tester will do **Test cases design (TCD) against the requirement**, **Test cases review**, **Test cases execution (TCE) on build (**application/ software)
* **In TCE**, if **functionality is not working properly** **as per the requirement** then tester will **raised/ create a defeat**
* These defect will inform to developer,
* Developer will fix these defect & Sent the Build
* Tester will do TCE/ check defect has been fixed or not
* **Support/ maintains-**
* Project team will provide 1 month warrant support for client against the complete module
* In 1 month warrant of period if any defect/ issue that will fixed project team
* After completion of warrant period, Support team will work in Support/ maintains
* Support team will work on existing application/ project Support - Defect/ bug fixed, Ticket, End user quires, enchantment, etc

# Fish Module / Basic SDLC module-

Information Analysis Design Coding Testing Support

Gathering (**BRS**) (**SRS**) (**HLD**) (**Developers- LLD**) **TCD, TCE**

Review (**BA**) Review (**BA**) Review (**Design**) **WBT** (White **BBT** (Black

Box testing) box testing)

Static testing/ Verification / Dynamic testing / Validation /

Quality assurance Quality Control

* **Review –** It is process to check correctness & completeness of your own work

|  |  |
| --- | --- |
| **WBT (White Box testing)** | **BBT (Black Box testing)** |
| WBT testing will be done by **Developer** | BBT testing will be done by **Tester** |
| WBT testing of 2 types   1. **Unit testing** 2. **Integration testing** | BBT testing of 2 types   1. **Functionality testing** 2. **Non-functionality testing** |
| In WBT testing developer will check, Code coverage, Looping statement, Conditional statement, etc | In BBT testing check, Error handling coverage, Input domain coverage, Backend coverage, etc |
| WBT testing also called as **Code level testing** | BBT testing also called as **system & functional testing** |

|  |  |
| --- | --- |
| **Static testing/ Verification** | **Dynamic testing / Validation** |
| In Static testing/ Verification, **BA** will check their own documents, **Designer** will check their own design and **Developer** will check their own code | In Dynamictesting/ Validation, Tester will check functionality of the application/ software/ build |
| Static testing also called as **Verification** | Dynamictesting also called as **Validation** |
| InStatic testing, for application/ software/ build will defines **Quality assurance** | InDynamictesting, for application/ software/ build will defines **Quality Control** |
| Static testing also called **In-progress testing** | Dynamictesting also called **End-progress testing** |

# Waterfall module/ Process-

* Waterfall module/process – it is **sequential process (step to step)**
* Sequential means after completion of 1st then move into 2nd stage **i.e. after completion of coding then only we will start testing**

Information

Gathering (**BRS**)

Analysis (**SRS**) – Module 1, Module 2 + old defect

Design – **HDL**

Coding – **LLD** – Build X Defect

Tester – **TCD, TCE on Build**

* **Dis-** **advantage –**

1. **Delivery / deployment duration = not fixed** OR more than 3 month
2. In waterfall module **backtracking is not possible**

# V-module/Process-

* V stand for **verification and validation**
* In V-module/ process, **parallel working is going on for development & Testing**
* V-module also called as **“Plane driven module”**
* In V-module/ process **delivery / deployment duration** = **3 month**

**LCD (Life cycle development)**  **LCT (Life cycle testing)**

Information gath. (**BRS**) Assessment of dev plane

Analysis (**SRS/FRS/CRS**) Prepare test plane

(SRS- 1 module) Requirements testing/ Doubt clear

Design – **HDL** Design phase testing

Coding – Developer -**LLD** Program Phase testing (**WBT**)

Test case design (**TCD**)

**Install Build**  System & fun. Testing (**BBT**) – Pune- FIS

User acceptance testing (**UAT**) - Client- USA

Knowledge transfer (**KT**)

Maintains **DRE** (defect removable efficiency) /

/Support **CR** (change request)/ **RFC/ Postmortem testing**

**DRE** (defect removable efficiency)-

* DRE defines how much thoroughly/ deeply we have tested the application
* **DRE = A / (A + B) = 9 / (9 + 1) = 9/10 = 0.9 = 90%**
* A= No of defect found in BBT
* B= No of defect found in UAT

**CR (change request)/ RFC (request for change) –**

* **CR –** If any functionality if client want to change then client sent the CR/ RFC
* In V-module, we will accept these CR/ RFC.
* For these CR/ RFC we will (company) take extra charge from the client
* **Ex.** Paytm – Initial Requirement – Recharge module functionality – Mobile no, Circle, operator, plane select amount enter 🡪 Development completed + Testing going
* **CR =** Requirement – Recharge module functionality – **Valid Mobile no** - automatic Circle, operator selected – automatic browser plane display 🡪

**Postmortem testing-**

* In V-module, before the application/ build is deployment to production/ end user, we will do one final testing these testing is called **Postmortem testing**
* InPostmortem testing, only main functionality of the application/ build test

**Dis-** **advantage –**

* For these **CR/ RFC we will (company) take extra charge from the client**
* **Delivery / deployment duration = 3 month**

# Agile Module/ process\*\*-

* Agile defines it is a **continue process for development and testing**
* In agile, if any CR comes from client side, these **CR will be accepted at any point of time** but we will check **impact of these CR** on current development, current Testing & application present on production.

1. If impact of CR is more 🡪 PM will inform to client (Current Sprint Or Next Sprint)
2. If impact of CR is less 🡪 These CR will be completed with Current Sprint

* In agile also called as **“values driven process”** (Client priority)
* In agile, **Delivery / deployment duration = 2 week (**service company) OR 3 week
* Agile have different sub-types/ Frameworks/ Methodologies-

1. XP (Extreme programming) – Only coding & Not testing
2. **Scrum** – **Sprint (bunch of User stories) wise delivery with 2 week**
3. Kanban- Support team
4. Lean - Support team Bunch of defect, Tickets, User request, enhancement, etc
5. FDD (Future driven development) -

* **In my project,** We have worked in **Scrum agile Methodologies**

# Agile Archicture-

Information **Product Backlog** (3 to 4 module = 400US)

Gathering (**BRS**)

Analysis (**SRS**) (1 SRS = 10 Use Cases) **Sprint** **Backlog (Client priority)**

(1SRS = 1 module - Movies) **Sprint 1 = 10 US –** PM, BA, Designer & Client

(1SRS = 2 module – Rent module) **Sprint 2 = 9 US – 2 week**

(1SRS = 3 module – Fasttag module) **Sprint 3 = 11 US – 2 week**

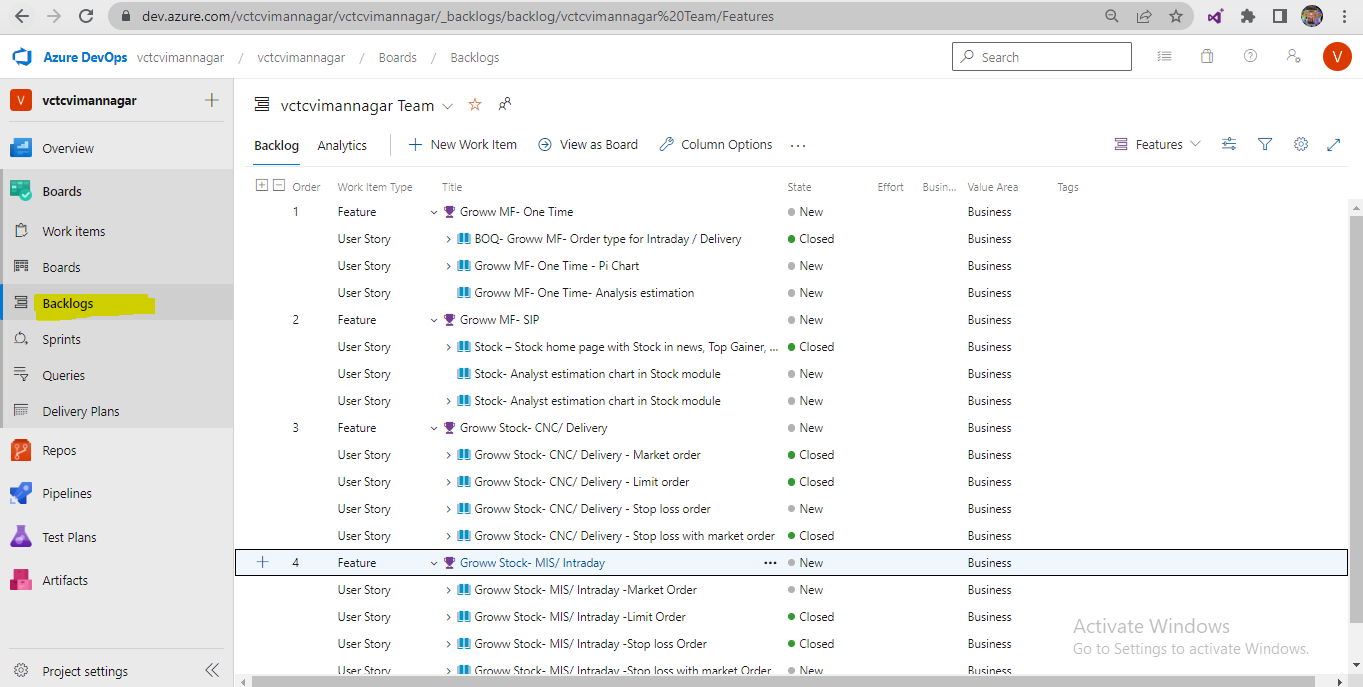
**Use Cases** (1 specific requirement) BA **User Stories** (1 use cases = 5 to 6 US)

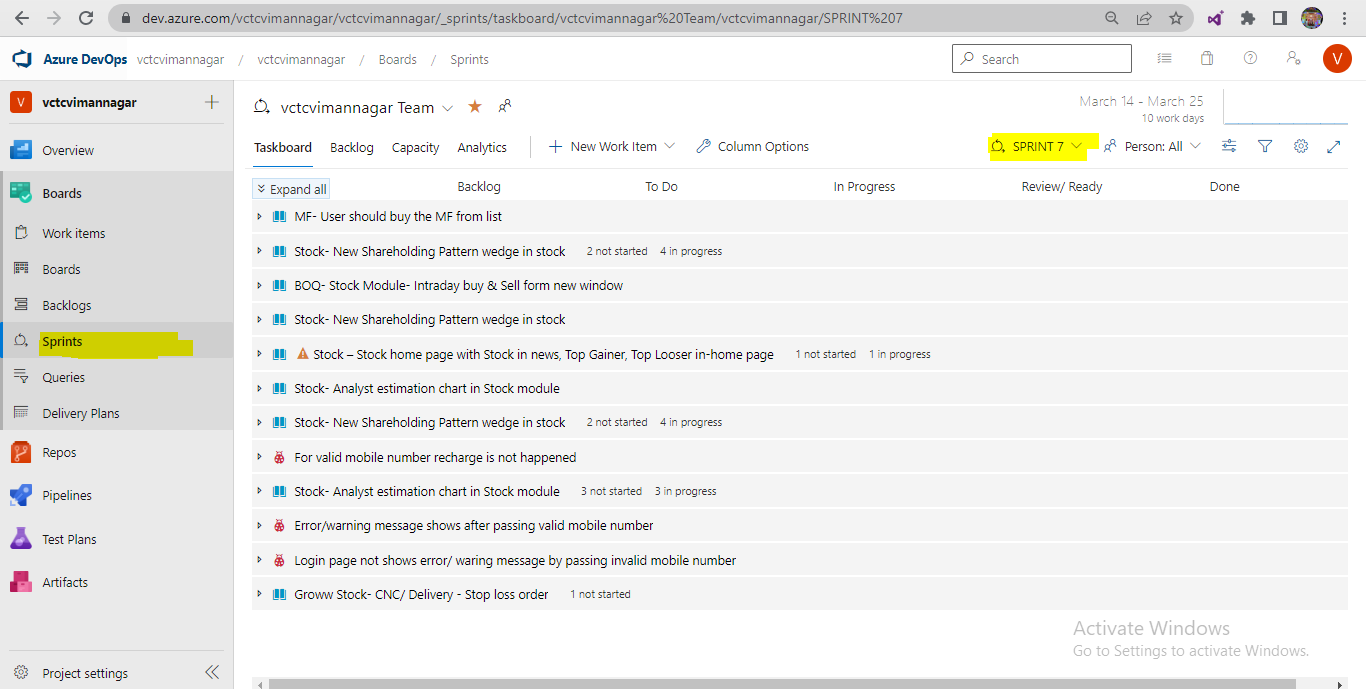
1. Description 1. Description
2. Acceptance criteria 2. Acceptance criteria

**Designer against User Stories**

**Coding against User Stories**

**Testing against User Stories**

**Product Backlog- **

**Sprint** **Backlog- **

# Agile meeting/ Ceremonies\*\* –

|  |  |  |
| --- | --- | --- |
| **Meeting** | **Purpose** | **Involvement** |
| **Sprint planning meeting**  1st day of Sprint | - Current sprint – **Task / work allocation** - **Estimation / story point** defines against the US  **Ex**.1US = 24hr dev + 14hr testing | **1hr/2hr – PM**, BA, Designer, Dev Team, Testing Team |
| **Grooming meeting**  1st day of Sprint | **-** Current sprint – **US doubt/ clarification** | **1hr – BA**, PM, Designer, Dev Team, Testing Team |
| **Daily stand up/ Scrum meeting**  Daily meeting **Morning** -  (Time = 10am to 10.15am)  **Evening** -  (Time = 5pm to 5.15pm) | **- What you have done yesterday**  **- What are you doing today**  **- Issue/ Roadblock**  **-** Daily work statusfor dev & tester | **15min- PM**, BA, Designer, Dev Team, Testing Team |
| **Sprint Review meeting**  Last day of Sprint | **-** Current sprint 4 US testing – respective **tester US against** **demo/ review to Client /UAT** | **1hr/2hr - Client / UAT,** PM, BA, Designer, Dev Team, Testing Team |
| **Sprint Retrospective meeting**  Last day of Sprint | **-** Current sprint **good & Bad work discussion**  Current sprint – Next sprint | **30min/ 1hr- PM**, BA, Designer, Dev Team, Testing Team |

# Agile Daily wise plane\*\*-

* Team size = 14 to 15 people = Tester 2 to 3 present
* Agile individual person work = 2 week = work day = 10 days \* 8hr = **80hr**
* **Team 1** = 3 Developer + **1 Tester (You/Me)** = Recharge module
* **Team 2 =** 2 Developer + 1 Tester = Movies module
* **Team 3** = **3 Developer (Onside)** + 1 Tester = Fasttag module
* Agile = **Current Sprint 1 = 10 US** - (3 US Recharge, 3 US Movies, 4 US Fasttag)

**1 week- (Mon to Friday) (office regular time= 9.30 am to 6 pm)**

**Monday – (1st day of Sprint)**

* **Sprint planning meeting-** (1hr/2hr) (10am to 11.30am)
* Current sprint 1– 3 US Recharge **- Task / work allocation**
* **For** 3 US Recharge **- Estimation / story point** defines (**Ex**.1US = 38hr dev + 14hr testing, 2US = 30hr dev + 18hr testing)
* **Grooming meeting (30min/ 1hr)-**
* 3 US Recharge against doubt/ clarification
* 3 Developer = **1US** - Coding (15hr) – In-progress
* **1 Tester = 1US - TCD (5hr) - Completed**

**Tuesday –**

* **Daily stand up / Scrum meeting-** (15 min)
* What you have done yesterday (1US – TCD – Completed)
* What are you doing today (2US – TCD – Start)
* Issue/ Roadblock
* 3 Developer = **1US** – Coding (18hr) – **Completed & 1 US build sent for testing**
* **1 Tester = 2US– TCD (6hr) - Completed**

**Wednesday -**

* **Daily stand up / Scrum meeting-** (15 min)
* What you have done yesterday (2US – TCD – Completed)
* What are you doing today (1US – TCE – Start)
* Issue/ Roadblock
* 3 Developer = **2US**– Coding – **In-progress**
* **1 Tester = 1US – TCE (6hr) – In-progress**

**Thursday**-

* **Daily stand up / Scrum meeting-** (15 min)
* What you have done yesterday (1US – TCE – In-progress)
* What are you doing today (1US – TCE – try to completed)
* Issue/ Roadblock
* 3 Developer = **2US**– Coding (18hr) – **In-progress**
* **1 Tester 1US – TCE (5hr) – Completed**

**Friday –**

* **Daily stand up / Scrum meeting-** (15 min)
* What you have done yesterday (1US – TCE – Completed)
* What are you doing today (3US – TCD – Start)
* Issue/ Roadblock
* 3 Developer = **2US** – Coding (14hr) – **In-progress**
* **1 Tester = 3US – TCD (4hr) – In-progress**

**2 week –**

**Monday –**

* **Daily stand up / Scrum meeting-** (15 min) (Time = 10am to 10.15 am)
* What you have done yesterday (3US – TCD – In-progress)
* What are you doing today (3US – TCD – Reaming work completed)
* Issue/ Roadblock
* **3 Developer** = **2US** – Coding (14hr) – Completed & **2US build sent for testing**
* **1 Tester = 3US – TCD (3 to 4hr) – Completed**

**Tuesdays –**

* **Daily stand up / Scrum meeting-** (15 min) (Time = 10am to 10.15 am)
* What you have done yesterday (3US – TCD – completed)
* What are you doing today (2US – TCE – start)
* Issue/ Roadblock
* **3 Developer** = **3US** – Coding (16hr) – In-progress
* **1 Tester = 2US – TCE (5hr) – In-progress & 2 defect in 2US in TCE**
* Tester will inform to developer & Developer will fix & Tester will re-test **(2hr)**

**Wednesday –**

* **Daily stand up / Scrum meeting-** (15 min) (Time = 10am to 10.15 am)
* What you have done yesterday (2US – TCE – In-progress)
* What are you doing today (2US – TCE – Reaming work completed)
* Issue/ Roadblock
* **3 Developer** = **3US** – Coding (18hr) – Completed & **3US build sent for testing**
* **1 Tester = 2US – TCE (4hr) – Completed**

**Thursday –**

* **Daily stand up / Scrum meeting-** (15 min) (Time = 10am to 10.15 am)
* What you have done yesterday (2US – TCE – completed)
* What are you doing today (3US – TCE – start)
* Issue/ Roadblock
* **3 Developer** = **Waiting for defect**
* **1 Tester = 3US – TCE (5hr) – In-progress & 5 to 6 defect raised in 3US while TCE**
* Tester will inform (5 to 6 defect) to developer & Developer **not accepting (invalid) 2 defect** & Developer fixed only 4 defect & Tester will re-test only 4 fixed defect **(2hr)**

**Friday – (last day of Sprint)**

* **Daily stand up / Scrum meeting-** (15 min) (Time = 10am to 10.15 am)
* What you have done yesterday (3US – TCE – In-progress)
* What are you doing today (3US – TCE – Reaming work completed)
* Issue/ Roadblock **(2 defect developer not accepting)**
* **3 Developer** = 2 defect will fixed (2hr)
* **1 Tester = 3US – TCE (3hr) – Completed & 2 defect re-test (1hr)**
* **Sprint review meeting (1hr/2hr)-**
* **3US Demo/ review to Client / UAT team**
* **Sprint Retrospective meeting (30min/ 1hr)-**
* **Discussion on Good & Bad work in current sprint**
* **Good –** Tester has done found more defect in US
* **Bad –** Developer has not done proper coding

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1 week** | | | | | |
| **Day** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** |
| **Developer** | 1US – In-progress | 1US – In-progress | 1US – In-progress & 1US Completed | 2US – In-progress | 2US – In-progress |
| **Testing** | 1US – TCD – In - progress | 1US – TCD – Completed | 2US – TCD – In - progress | 2US – TCD – Completed | 1US – TCE – In - progress |
| **2 week** | | | | | |
| **Day** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** |
| **Developer** | 2US – In-progress & 2US Completed | 3US – In-progress | 3US – In-progress | 3US – In-progress & 3US Completed | **Waiting for defect** |
| **Testing** | 1US – TCE – In – progress & & 1US TCE Completed | 2US – TCE – In - progress | 2US – TCE – In – progress & 2US TCE Completed  3US – TCD – In - progress | 3US – TCD - Completed  3US – TCE – In – progress | 3US – TCE – In – progress & 3US TCE Completed |

**OR**

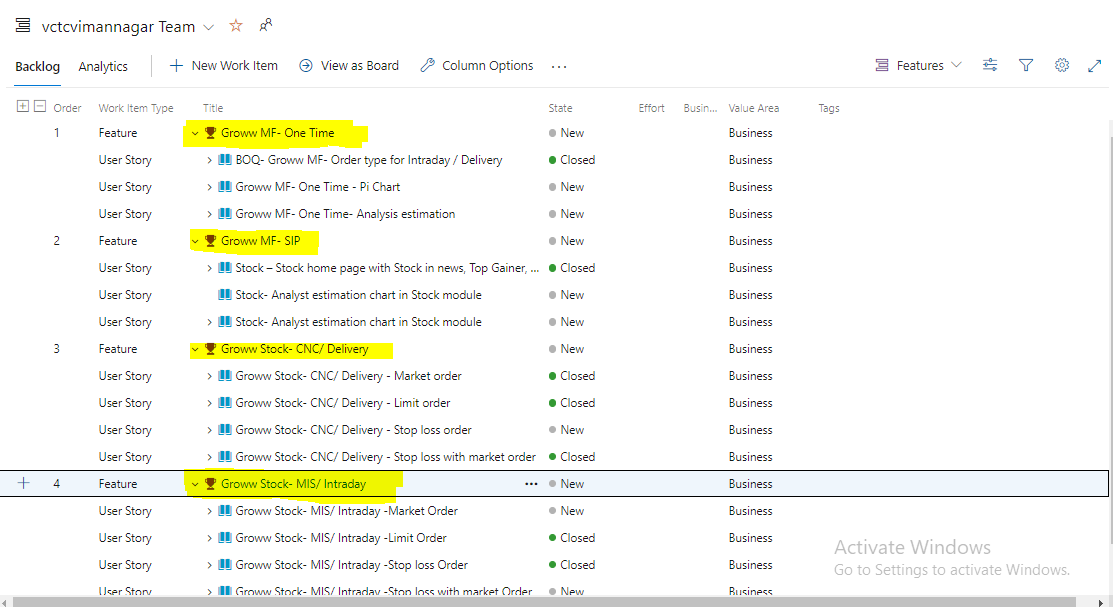
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1 week** | | | | | |
| **Day** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** |
| **Developer** | 1US – In-progress | 1US – Completed | 2US – In-progress | 2US – In-progress | 2US – In-progress |
| **Testing** | 1US – TCD – Completed | 1US – TCE – Completed | 2US – TCD – In - progress | 2US – TCD – Completed | 3US – TCD – In - progress |
| **2 week** | | | | | |
| **Day** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** |
| **Developer** | 2US – In-progress & 2US Completed | If less impact CR implement + 3US – In-progress | 3US – In-progress & 3US Completed | **Waiting for defect** | **Waiting for defect** |
| **Testing** | 3US – TCD - Completed | 2US – TCE – In - progress | 2US – TCE – In – progress & 2US TCE Completed | 3US – TCE - In – progress | 3US – TCE – In – progress & 3US TCE Completed |

# Agile Term\*\*-

* **Burn down chart –** It is chart which defines **how much work is reaming w. r. time**. It is present in Project management tool (JIRA/ Azure develop)
* **Burn up chart –** It is chart which defines **how much work is completed w. r. time**. It is present in Project management tool (JIRA/ Azure develop)
* **Velocity -** It is chart which defines **how much work/ time will required to complete the sprint** OR How much time taken to completed the sprint and how much will exactly required

****

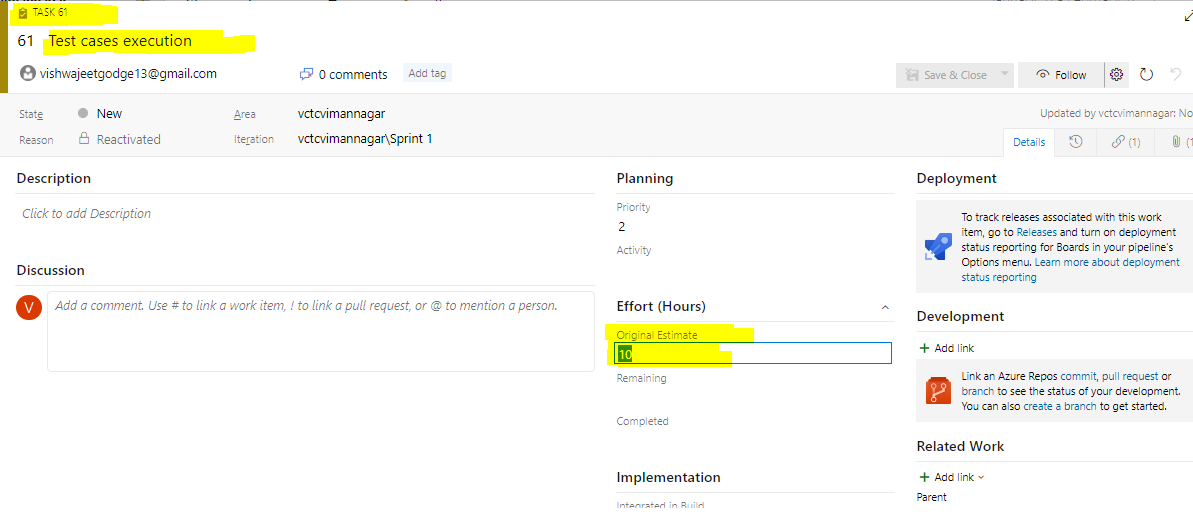
* **Epic =** Main US of a module. These epic will contains multiples US related to these module

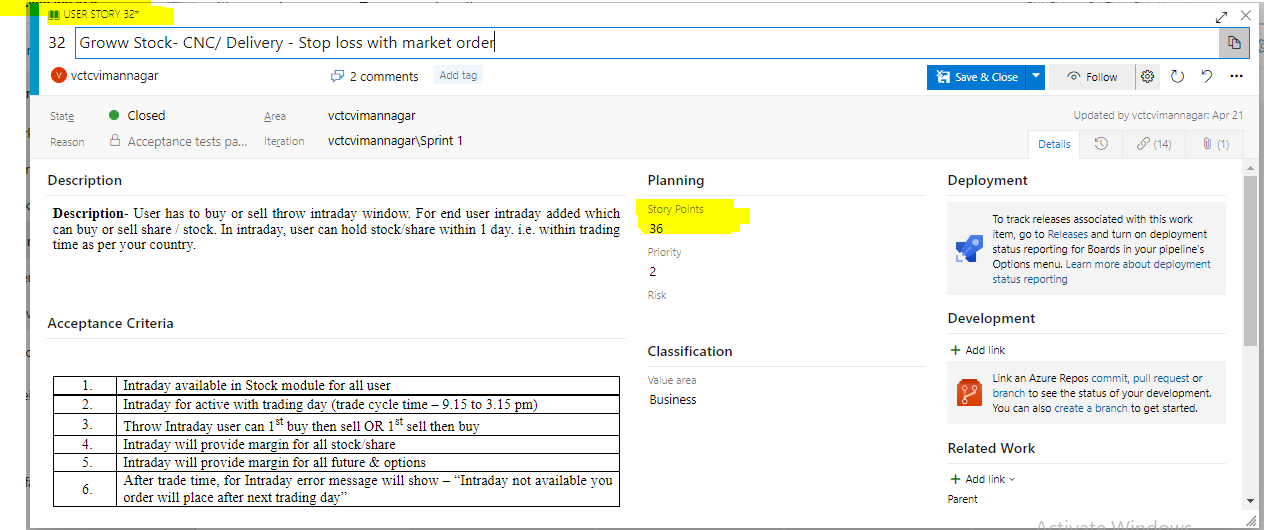
****

* **Estimation/ story point – Time required for** US against development & testing time is called estimation.
* **Ex. 1US =** Developer 36hr coding + Testing 24hr TCD, TCE = **60hr**
* **Factor to decide estimation/ story point**

1. How much knowledge we have against the US
2. How much complexity the US for testing
3. How much efforts required for testing

* **Story point = Time define for a US**
* **Estimation = Time define for a task** which present against the US

****

****

* **Sprint refinement meeting/ Scrum zero-** InSprint refinement meeting **PM, BA, Designer & Client present,** which decides **planning for next sprint.**
* **Ex.** CurrentSprint 4 = 10 US 🡪2 week working. On 2 week Wednesday/ Thursday Sprint refinement meeting will be conducted
* **Scrum of scrum meeting –** Scrum of Scrum master will conduct meeting with all PM/ Scrum of these big project.
* **Ex.** Paytm project
* Project 1 = Paytm.com – Fintech project 🡪 PM/ Scrum master
* Project 2 = Paytmmall.com – Reatil project 🡪 PM/ Scrum master Scrum of
* Project 3 = Paytmmoney.com – Finance project 🡪 PM/ Scrum master Scrum master
* **Agile peoples-**

1. Client 🡪 Stakeholder
2. Delivery manager 🡪 Solution master
3. Project manager 🡪 Scrum master
4. Business annalist 🡪 Product owner

# Advantages & dis- advantages of agile process-

* **Advantages –**

1. In agile, Delivery / deployment duration = **2 week**
2. In agile, If any CR will comes then we will accepted these CR
3. Automation is possible over the manual
4. Daily stand up meeting/ Scrum meeting (Daily work progress)
5. Checkpoint is present in every module

**Ex. Recharge module-**

**1US**- Sub-module **2US**- Sub-module **3US**- Sub-module **4US**- Sub-module **5US**- Sub-module **6US**- Sub- module

250 line code 200 line code 210 line code 260 line code 340 line code 140 line code

Browser planes/ Planes details

Thank message

Payment tab

**Promo code**

Mobile no, Operator, Circle

Recharge Icon

**Dis-Advantages-**

1. If frequently changes occurred in a US from client (Delivery / deployment will be delay)
2. If your project/ functionality is depends on another project/ functionality (Delivery / deployment will be delay)

INOX/ PVR – Ticket

Paytm – Movies ticket, Electricity, etc

MSEB/ Adani/ Tata power – Electric

**Interview Question –**

1. What is your team size
2. How many developer are present in a project
3. What is SDLC?
4. What is SDLC & STLC?
5. What is a SRS document & what it will contain?
6. If we have doubt against the SRS documents so what you will do?
7. What is difference between WBT & BBT
8. What is difference between Verification & Validation
9. What is V-module & What are dis-advantages over the agile process
10. What is postmortem testing?
11. What is Agile, Scrum & Sprint?
12. What are the agile ceremonies?
13. What are agile artifacts?

* **Answer** – Product Backlog, Sprint Backlog, Agile ceremonies, Scrum, Sprint

1. How agile process flow in your organization?

* **Answer –** Agile day wise plane

1. What is daily routine in you project?

* **Answer –** Work office = 9.30 am to 5.30 pm
* 9.30 am = Mail check (Developer mail, Client/PM mail, Important mail)
* 10 am = Morning stand up- 15 min to 30 min – Daily stand meeting (work status)
* Accordingly to work = We will write TCD or TCE against the US
* In TCE, if we got the defect then we will inform to developer, these is daily routine for 2 week
* 5 pm =Evening stand up - 15 min to 30 min – Daily stand meeting (work status)
* Last day of Sprint – Sprint review, Sprint retrospective meeting

1. What we have discussed in last Sprint retrospective meeting?

* **Answer-** Last sprintDeveloper has not done proper coding/ not done unit testing so we have raised 6 to 7 defect.
* In Sprint retrospective meeting, PM told to all developer unit testing is compulsory before deployment to testing

1. What is Burn down chart, Burn up chart, velocity, Epic
2. What are scrum zero/ refinementmeeting?
3. What is estimation and how you will define estimation?
4. What is your process, if we got CR in middle of sprint?

* **Answer**- In agile, if any CR comes from client side, these **CR will be accepted at any point of time** but we will check **impact of these CR** on current development, current Testing & application present on production.
* If impact of CR is more 🡪 PM will inform to client (Current Sprint Or Next Sprint)
* If impact of CR is less 🡪 These CR will be completed with Current Sprint

1. What is advantages & dis- advantages of Agile process

# Project technology/ Techniques –

* **In my project**, we have different technology/ techniques
* **Frond end /GUI 🡪 Angular framework**
* **Service/ API 🡪Java languages**
* **Database/ SQL Server 🡪 SQL languages**

PVR/ INOX – Ticked not

**Service/ API** –

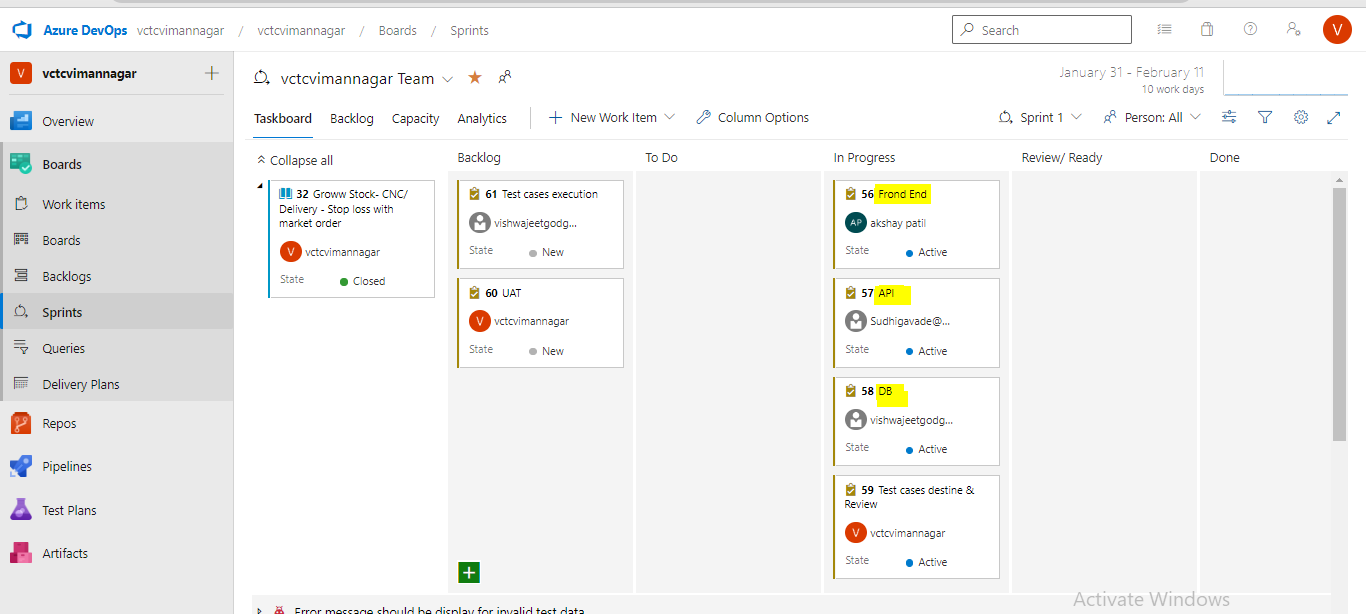
API Testing, Restassured

**Frond End/ GUI** –

Manual Testing, Automation Testing

**Database** –

DB Testing/ ETL Testing



# Environments in project-

* **In my project**, we have **4 environments**

1. **Dev environments 🡪** Only Developer are working
2. **SIT/ Test environments 🡪** Only Tester are working **Pune - Infosys**
3. **UAT / Pre-prod/ Staging 🡪** Only Tester are working
4. **Prod/ Live environments 🡪** End user will use the application **USA – BOQ**

* **Testing in project-**
* **In Dev environments (only developer) –** Unit Testing, Integration Testing
* **SIT/ Test environments (Only tester) –** Sanity testing/ Smoke testing, System& Functional testing, Re-testing & Regression testing
* **UAT Testing (Client side tester) –** Alpha & Beta Testing
* **Production –** Production issue

**Prod/ Live environments**

**3US = 8650 code**

**UAT/ Pre-prod/ Staging environments**

**3US = 8650 code**

**SIT / QA/ Test environments 3US = 8650 code**

**Dev environments**

**3US = 250 code + 8400 existing code**

IP/ DB = **172.10.21.161** IP/ DB = **172.20.10.141** IP/ DB = **192.10.62.190 www.Paytm.com**

**Infosys, Pune BOQ Banking, USA**

* Dev environments URL = **172.10.21.161:8080/paytm.com**
* SIT environments URL = **172.20.10.141:8080/paytm.com**
* UAT environments URL = **192.10.62.190s:8080/paytm.coms**

**Interview question –**

1. Which technology present in your project
2. What is URL of your project

* **Answer**- SIT environments URL = **172.20.10.141:8080/paytm.com**

# Dev environments-

* In Dev environments, **Developer will work**
* Developer will do coding against the US for preparation of build (Functionality without test)
* After completion of coding developer will do testing, these testing are called WBT
* 2 types WBT

1. **Unit Testing**
2. **Integration Testing**

**Unit Testing-**

* Unit testing will performed **on every US.**
* When developer will done with **coding against US / Sub-module** then he will do unit testing
* When developer will do unit testing then he will prepared a **Unit testing documents**
* Unit Testing documents will **contains Screenshot new implanted functionality, Tables name, etc**
* **Ex**. Paytm - Recharge module – Browser planes tab display

**Integration Testing-**

* Integration testing will performed on **completed module**
* **Ex**. Paytm - Recharge module – 6 US need to completed

**Promo code -**

Browser planes/ Planes details

Mobile no, Operator, Circle

**Recharge Icon - Driver**

**1US**- Sub-module **2US**- Sub-module **3US**- Sub-module **4US**- Sub-module **5US**- Sub-module **6US**- Sub- module

250 line code 200 line code 210 line code 260 line code 340 line code 140 line code

Thank message

**Payment tab – Stub**

* Integration testing approaches –

1. **Top down approaches** – If we don’t have any sub-module, for these sub module developer will prepared a stub. **Stub** it is temporary program developer in xml langue’s
2. **Bottom up approaches -** If we don’t have any starting sub-module, for this sub module developer will prepared a Driver. **Driver** it is temporary program developer in xml langue’s
3. **Sandi watch/ Bi-directional -** If we don’t have any starting sub-module, for this sub module developer will prepare a driver. If we don’t have any sub-module, for these sub modules developer will prepared a stub. Driver/ Stub it is temporary program developer in xml langue’.

# SIT ENVIORMNETS-

* In SIT environments Tester will work
* Tester will do TCD & TCE against the US
* Developer do coding against the US (1US= 200 line code) and complementation these code will me **merge** with existing application (app = 8400 line code), these merge will do in **“GitHub tool”** (code repository= 8600 line code).
* For deployment these all new feature from Dev environments into SIT environments developer will **run “Jenkins Job”.**
* When we got the Build in SIT environments then Tester will start the testing
* In SIT environments, we will do testing

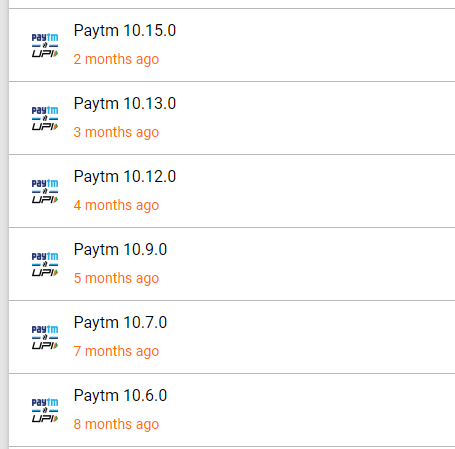
1. **Sanity/ Smoke testing** –No TCD, New Build , Check build stability
2. **System & functional Testing** –TCD, Stable Build, Intern & external feature Check
3. Usability Testing
4. **Functional testing** – Functionality & Non – Functionality testing ()
5. Performance testing
6. Security testing
7. **Re-testing –**TCD, Stable Build, Same Functionality with different test data
8. **Regression Testing –** No TCD, Modify Build, Check defect fixed or Side impact

# Sanity testing\*\*-

* When developer has deployed all code in SIT environments then tester will do testing
* When developer will sent **New Build** then tester will do **Sanity testing**
* In sanity testing, Tester will **check** **build stability i. e. check either the build is stable for testing or not**
* In sanity testing tester will check

1. Validation the **GUI/ UI** of application/ Build
2. Validation the **Core / main functionality** of application/ Build
3. Validation the **Link present** in application/ Build
4. Validation the **Tab/Page present** in application/ Build
5. Validation the **Navigation present** application/ Build

* In sanity testing, tester **never write test cases**
* For sanity testing, We required **1hr to 2hr (only checking build stability)**
* In sanity testing, if we **found defect or build is not stable** then tester will **reject the build**
* In sanity testing, We got defect due to these reasons, **Core/ main functionality not working, Environments problem** (occurred due configure file), **Run time problem** (500-internal server problem, 404- not found, 400- bad request)**, Database connectivity**, etc
* **Ex.** Paytm –Recharge module – US = Browser plane **New build V9.0** 🡪 Developer sent for testing 🡪 Tester will do **sanity testing** **on new build** 🡪 If we **found defect** (Browser plane not display for BSNL/JIO/VI operator) 🡪 Tester will **reject the build (V9.0)** & tester will inform to developer throw mail 🡪 Developer will **fix and prepared a new build (V9.1)**

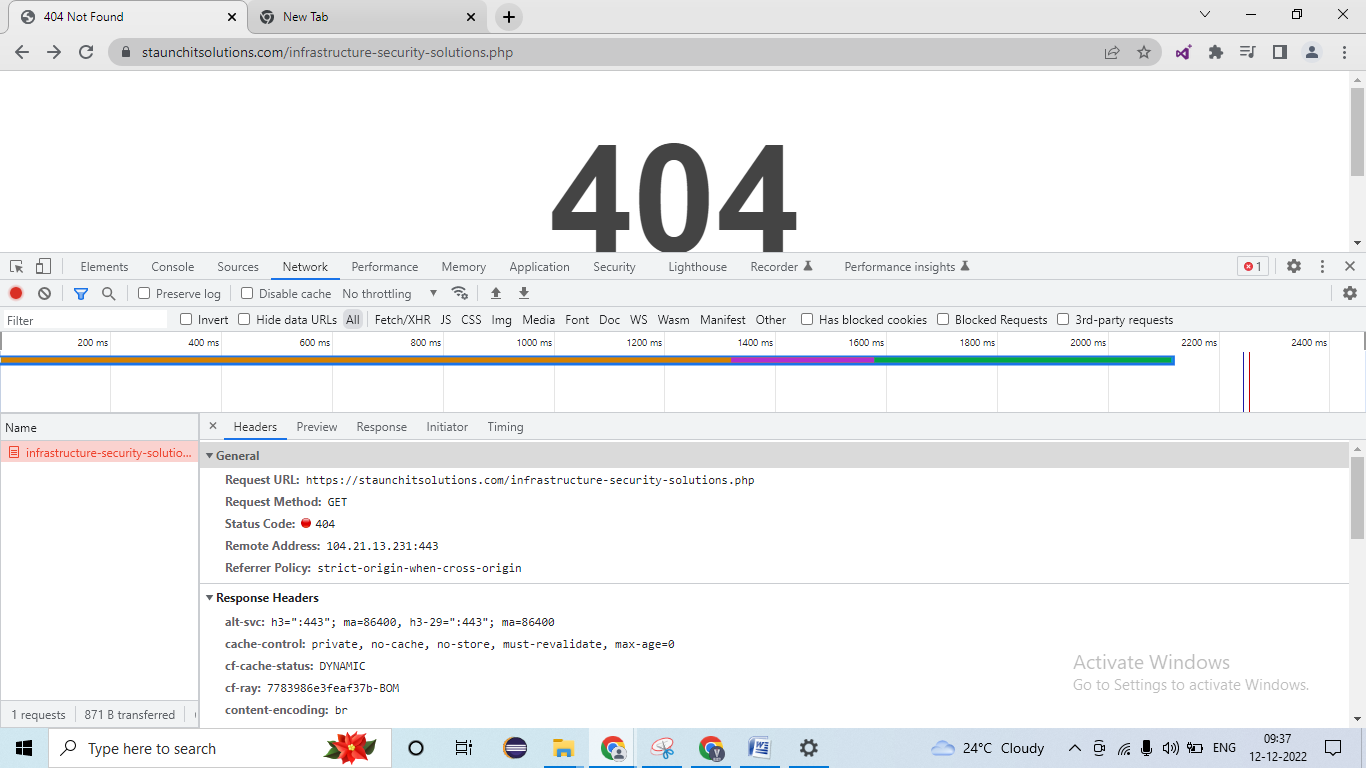


# Smoke testing\*\*-

* Smoke testing, It is a **advance version of Sanity testing**
* When developer will sent **New Build** then tester will do **Smoke** **testing**
* In Smoke testing, Tester will **check** **build stability i. e. check either the build is stable for testing or not**
* In Smoke testing tester will check

1. Validation the **GUI/ UI** of application/ Build
2. Validation the **Core / main functionality** of application/ Build
3. Validation the **Link present** in application/ Build
4. Validation the **Tab/Page present** in application/ Build
5. Validation the **Navigation present** application/ Build

* In Smoke testing, tester **never write test cases**
* For Smoke testing, We required **1hr to 2hr (only checking build stability)**
* In Smoke testing, if we **found defect or build is not stable** then tester will **reject the build** and Tester will found the **root cause of the defect**.
* Tester will inform to developer throw mail
* Smoke testing = Sanity testing + **Root cause / Troubleshooting** – Tester
* Smoke testing = Sanity testing + **Package validation** – Developer
* **In my project**, I have performed **Smoke testing**



* In Smoke testing, We got defect due to these reasons, **Core/ main functionality not working, Environments problem** (occurred due configure file), **Run time problem** (500-internal server problem, 404- not found, 400- bad request)**, Database connectivity**, etc

# System & functional Testing-

* When tester found that **build is stable** then we will **do System & functional testing**.
* If we got the old build (enhancement of existing functionality) for testing, then tester will **do System & functional testing**
* System & functional testing 4 types

1. **Usability testing**
2. **Functional testing 90 to 95%**
3. ~~Performance testing/ Load testing /Stress testing~~ **Jmeter**, Load runner tool
4. ~~Security testing~~ **5 to 10%**

# Usability testing-

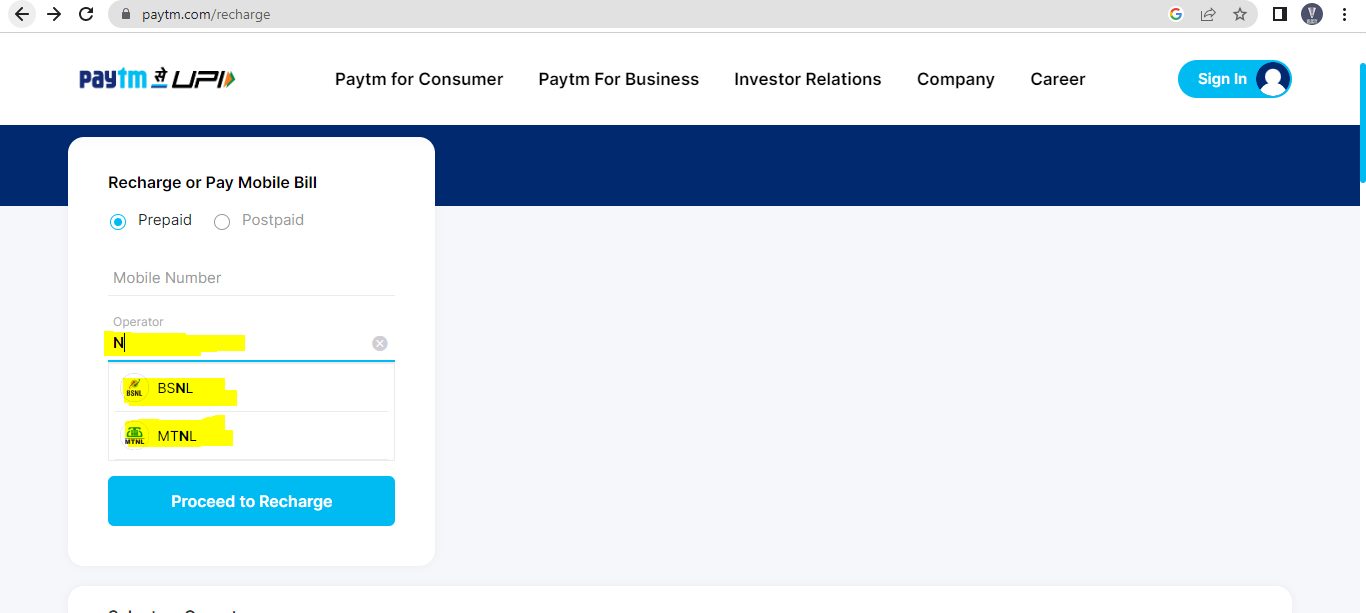
* In Usability testing, tester will check **user** **friendly of the application/ build**
* 2 Usability testing,

1. **GUI** (graphical user interface) **/ UI** (user interface)/ **Front end Testing –**

* Check **look & feel** of the application/ build
* Check **easy to use** of the application/ build
* Check the **speed of interface** present in application/ build

1. **Manual support testing-**

* In manual support testing, we will check **manual input values enter into object** (application/ build elements), how the **sensitivity/ sensitiveness of the theses input values**
* **Ex. Drop down –** Values

****

# Functional testing\*-

* In functional testing, Tester will check **internal & external feature of the application**.
* 2 types functional testing

1. **Functionality testing** - Checking **internal feature** of the application
2. **Non- Functionality Testing** - Checking **external feature** of the application

# Functionality testing-

* In Functionality testing**, tester will c**hecking **internal feature of the application**
* For internal feature we have performed **different coverage** (**BIEBSC**)

1. **B**ehavioral coverage testing
2. **I**nput domain coverage testing
3. **E**rror handling coverage testing
4. **B**ackend coverage testing
5. **S**ervice level coverage testing
6. **C**alculation based coverage testing

**Behavioral coverage testing-**

* In Behavioral coverage testing, Tester will **check property & behavioral of the object/ web elements** present in the application/ build
* **Ex.**

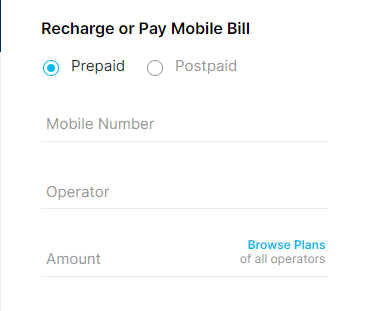
|  |  |
| --- | --- |
| **Object/ web elements** | **Property & behavioral** |
| Text box | Focus & Un-focus |
| Check box | Check & Un-check |
| Radio Button | On & Off |
| Button | Enables & Disables |
| Drop down | Values selection |

**Input domain coverage testing-**

* In input domain coverage testing, Tester will **check input data types & size/ length pass into object/ web elements** present in the application/ build

1. **BVA (boundary values analysis) – Input size/** **length** pass into object/ web elements
2. **ECP (Equivalents class partition)- Input data types** pass into object/ web elements
3. **Decision table testing techniques – Different Input combination** pass into object/ web elements to the result

* **Ex. Recharge module-**

****

|  |  |  |
| --- | --- | --- |
| **BVA object (Length/ Size)** | **Pass criteria** | **Fail criteria** |
| Mobile number | 10 digits values | 9 digits values, 11 digits values, etc |
| Amount | 2 digits values & 5 digits values | 1 digits values, more than 5 digits values |

|  |  |  |
| --- | --- | --- |
| **ECP object (Data types)** | **Pass criteria** | **Fail criteria** |
| Mobile number | Integer values | Decimal, Character, Symbol, etc |
| Amount | Integer values | Decimal, Character, Symbol, etc |
| Operator/Circle | Character (A-Z, a-z) | Decimal, Integer, Special Symbol, etc |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Decision table testing techniques** | **Input 1** | **Input 2** | **Input 3** | **Input 4** | **Input 5** |
| Mobile number | Valid | In-Valid | Valid | Valid | Null/Blank |
| Operator/Circle | Valid | Valid | In-Valid | Valid | Null/Blank |
| Amount | Valid | Valid | Valid | In-Valid | Null/Blank |
| Recharge button | Click | Click | Click | Click | Click |
| **Result** | Next page | Error message | Error message | Error message | Error message |

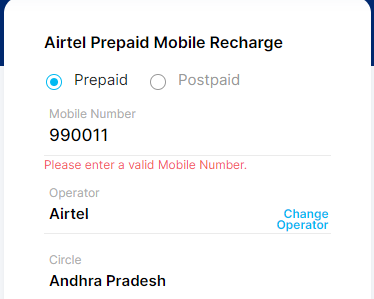
**Q. Text box = Starting values not 0, 1 accepted and only 4 digit value accepted, BVA & ECP**

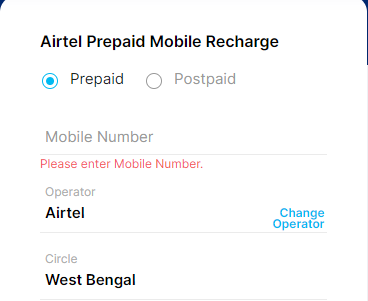
**Answer-** BVA – Pass criteria = Min = 2000 & Max = 9999

ECP – Data types – Pass criteria = only integer values

**Error handling coverage testing-**

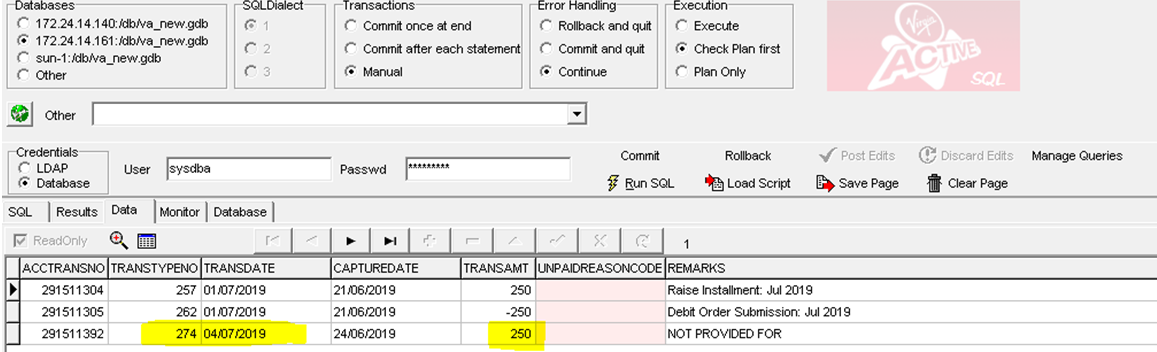
* In Error handling coverage testing, tester will check **different error message present** in the application/ build **by passing invalid test data/ -ve test data**
* Error message will be shows as per US
* Ex. Recharge module –



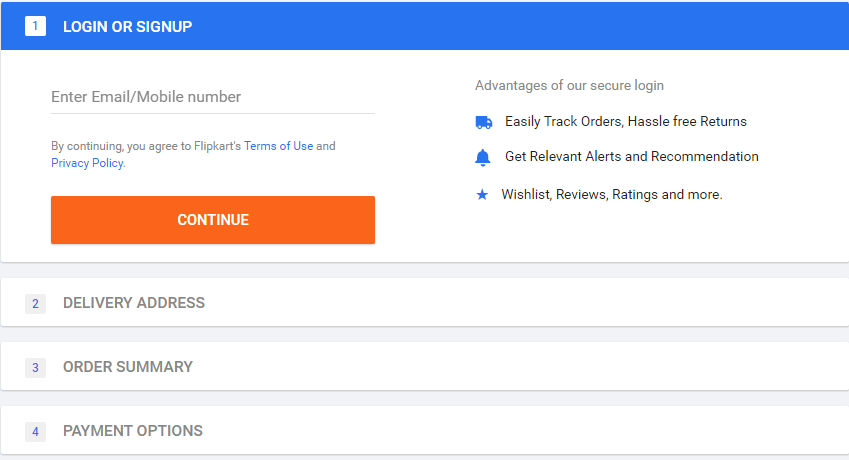


**Backend coverage testing/ Database testing-**

* In Backend coverage testing, Tester we will **check application/ build front end operation that frond data will be data backend stored**
* Backend coverage testing also called **Database testing**
* **Ex**. Recharge module – Mobile = 99001122333, Operator = Airtel, Circle = Maha, & Recharge 499rs – Order ID/ Transaction ID = 47349992
* For checking database / backend data – **SQL quires run**
* **Ex**. SQL quires **= SELECT \* FROM TN WHERE ORDER ID = 47349992 ;**

****

**Service level coverage testing-**

* In Service level coverage testing, Tester will check application/ build **service sequential working/ flow**
* **Ex**. Recharge module – Mobile, operator, Circle, Amount 🡪 **Check box Uncheck** = Fast forward 🡪 Click process to recharge 🡪 Promo tab 🡪 Payment tab 🡪 Thanks message
* **Ex**. Recharge module – Mobile, operator, Circle, Amount 🡪 **Check box check** = Fast forward 🡪 Click process to recharge 🡪 ~~Promo tab~~ 🡪 Payment tab 🡪 Thanks message
* **Ex. Flipkard- **

**Calculation based coverage testing-**

* In **Calculation based coverage testing, Tester will check arithmetic calculation**
* **Ex**. Recharge module – Mobile, operator, Circle, Amount= 499rs 🡪 🡪 Click process to recharge 🡪 Promo tab = 10% cash back 🡪 Payment tab = 499rs – 49rs = 450rs 🡪 Thanks message

# Non- Functionality Testing-

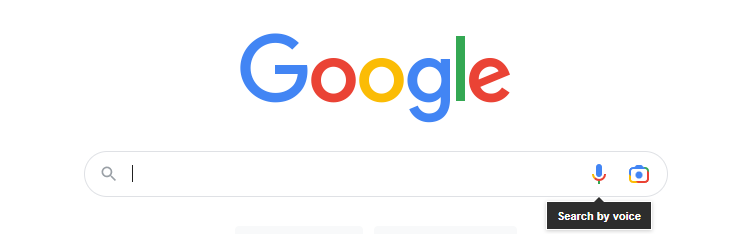
* In Non-Functionality testing**, tester will c**hecking **external feature of the application**
* For externalfeature we have performed **different coverage** (**RCCIISPG**)

1. **R**ecover coverage testing
2. **C**onfiguration coverage testing
3. **C**ompability coverage testing
4. **~~I~~**~~nstallation coverage testing~~ – Not performed
5. **I**ntersystem coverage testing
6. **S**anitization coverage testing
7. **~~P~~**~~arellzation coverage testing~~ - Not performed
8. **G**lobalization coverage testing
9. **Recover coverage testing-**

* In recover coverage testing, Tester check application/ build **abnormal situation**, application/ build recover
* **Ex**. PVR web site – Booking movies 5 ticket 🡪 Payment tab debit card details & Submit button 🡪 **abnormal situation (bank server not responding/ Message don’t press back button/ don’t refresh/ don’t close application- Press back button** 🡪 Previous page (order detail tab)

1. **Configuration coverage testing-**

* In Configuration coverage testing, Tester will check application/ build **support to hardware device** (Voice, Bluetooth, Printer, etc)
* These Configuration coverage testing also called **Hardware converge testing**
* **Ex.** Amzone application, Google search engine -



1. **Compability coverage testing-**

* In Compability coverage testing, Tester will check application/ build is **Compability/ conformant with client expected environments/ platform (different OS support)**
* **Ex.** Paytm - expected environments (Windows, Linux, etc)

1. **Forward Compability coverage testing-**

* If we have problem with OS (Network team will work)

**OS support - Problem**

Application/ build

1. **Backward Compability coverage testing-**

* **I have worked**, in **Browser** **Compability coverage testing**

OS support

**Application/ build - Problem**

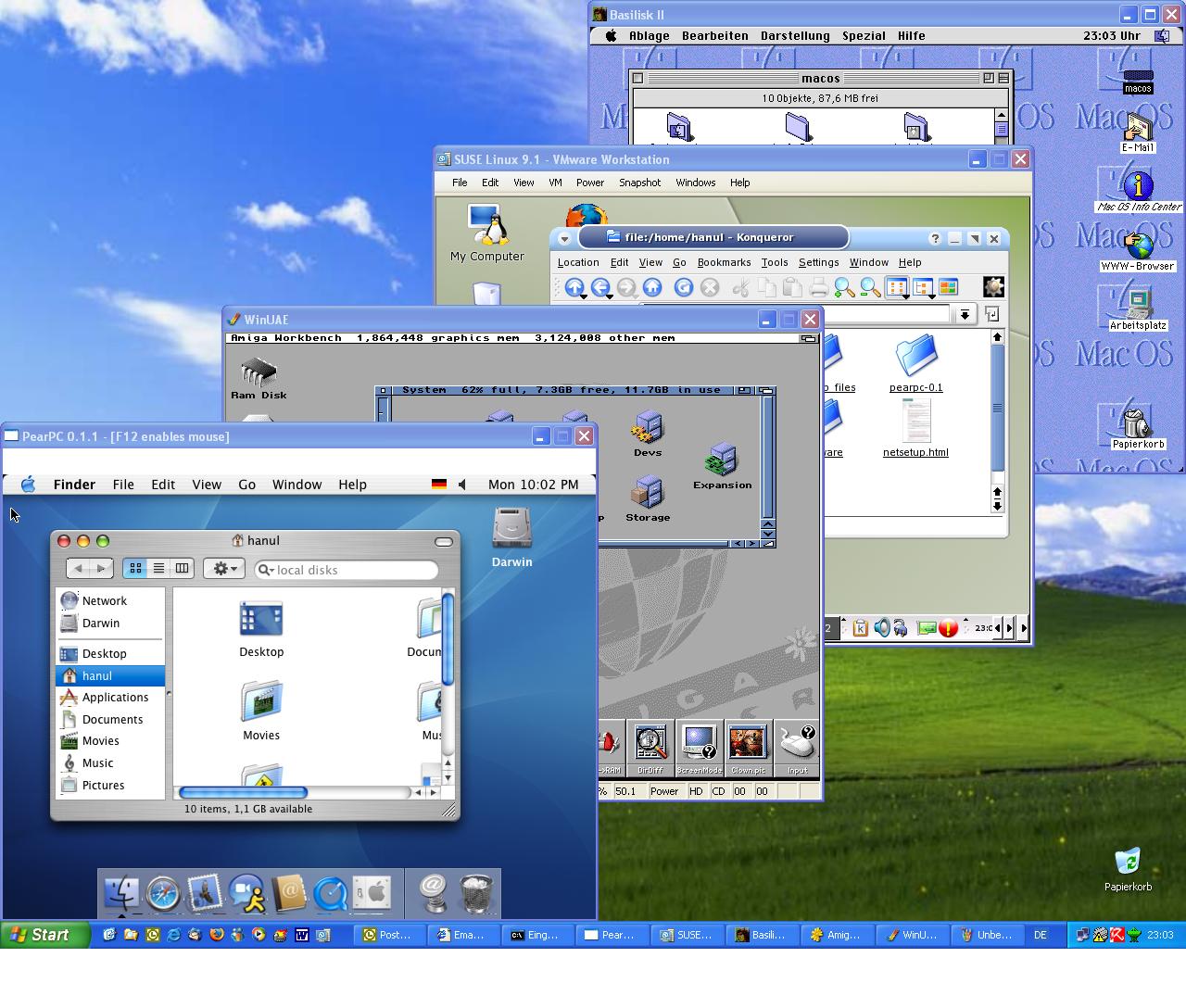
* Different Browser Compability coverage testing-

1. **Cross Browser** **Compability coverage testing –**

* In Cross Browser Compability coverage testing, tester will check application/ build support for **different browser as Chrome, Firefox, IE, Opermini, Edge, Safrai, etc**
* **Ex.** Application/ build – Support on chrome browser

1. **Version control Browser** **Compability coverage testing -**

* In Cross Browser Compability coverage testing, tester will check application/ build support for **same browser with different version**
* To install same Chrome browser with different version we will use **Virtual machines OR Remote desktop**
* **Ex.** Paytm – Chrome browser –V75, V80, V85, V90, V95, V100, V105, V108, etc



1. **~~I~~**~~nstallation coverage testing~~ – Not performed

* Application – Old version V17.30 🡪 ~~New version V17.35 install – problem~~ 🡪 Old version V17.30 🡪 New version V17.35 install

1. **Intersystem coverage testing-**

* In Intersystem coverage testing, Tester will check application/ build other application data inter change / connected
* Ex. Paytm – Movies – PVE/INOX, Electricity – MSDCL, TATA powser, etc

Movies – PVE/INOX

Paytm – Electricity, Movies, Recharge

Electricity – MSDCL, TATA power

1. **Sanitization coverage testing –**

* In Sanitization coverage testing, Tester will **check application/ build unnecessary/ extra feature/ functionality which is not mentioned in US**
* For these unnecessary/ extra feature/ functionality which is not mentioned in US, Tester will **raised/ create defect in JIRA/ Azure DevOps tool**
* Ex. Paytm – Recharge module – Mobile text box – 10 digits mobile no.
* Development - Mobile text box = **+91** - 10 digits mobile no

1. **~~P~~**~~arellzation coverage testing~~ - Not performed
2. **Globalization coverage testing-**

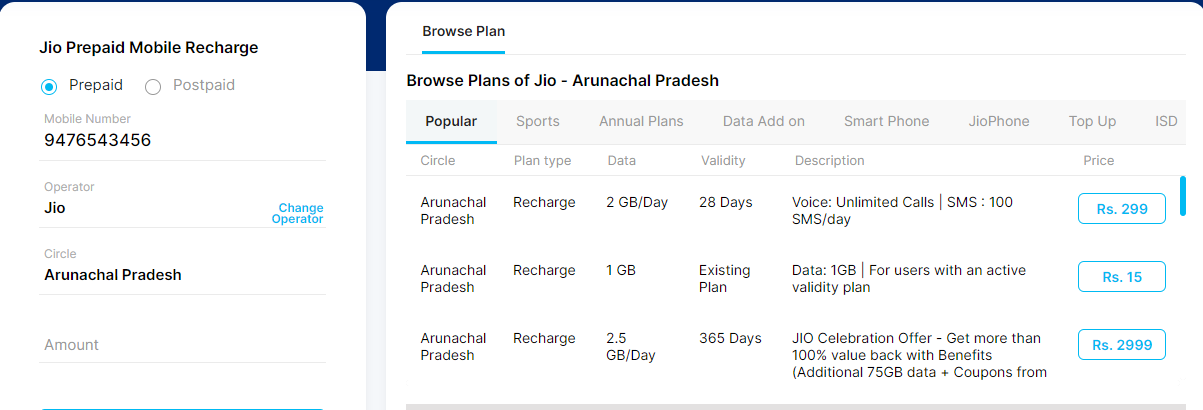
* In Globalization coverage testing, Tester will **check application/ build support with different languages**

1. **Standard Globalization coverage testing –** Support with English languages
2. **Regional Globalization coverage testing -** Support with Regional languages (Marathi, Hindi, Tamil, Tealgy, Kanada, etc)

* For checking Regional languages, we are using “**Google translator”**

# Re-testing\*\*-

* **In Re-testing**, it is part of system and functional
* In Re-testing, Tester will check **same functionality by passing multiple test data**
* Project USA banking = For **existing project**, Test data we will get from **database**
* Project USA banking = For **New project**, Test data we **have create** (Front end/ SQL quires backend insert/ modify)
* For re-testing, we are writing test case
* **Ex. US =** Paytm – In Recharge module, Check browser planes display 🡪 Pass test data = Different Mobile, Operator, Circle – Check different planes Browser planes

****

# Regression Testing\*\*-

* When Tester are **system & functional/ Re-testing** on **stable build,** if we **got the defect** then tester will create / raised into JIRA/ Azure DevOps tool and inform to developer
* Developer will check & **fix** these defect then developer will **prepared a Modified build**
* When tester will **got** **Modified build** for the testing then we will **do Regression testing**
* In regression testing, tester will check **defect has been fixed and check there is not side impact on interconnected module**
* **In Regression testing, we will check/ We will execute test cases (Regression Suit)**

1. Validation the defect related to test cases
2. Validation the core / main functionality related to test cases
3. Validation extra feature added in the functionality
4. If time permit, we will execute reaming test cases

* **Ex.** Paytm –US = In Recharge module, Check browser planes display 🡪 Pass test data = Different Mobile, Operator, Circle – developer done coding **stable** **build (V14.00 = 270 code)** & sent the build for testing 🡪 Tester doing re-testing & in re-testing For operator Airtel, JIO, BSNL different browser planes shows but **VI operator no browser planes will display** 🡪 Tester will **raised a** **defect** & Inform to developer 🡪 Developer will fix the defect & prepared a **modified build (V14.00 = 320 code)** sent for testing 🡪 tester will do **regression testing** 🡪

Tester will check –

1. **Defect has been fixed -** VI operator shows browser planes will display
2. **Side impact on interconnected module -** Airtel, JIO, BSNL different browser planes shows

* **Ex**. **1US = 30 TCD** 🡪 In TCE, TC no. -20 Defect found & inform to developer 🡪 Developer will fix these defect and sent the **modified build** 🡪 Tester will do **Regression testing**, In regression testing we will execute **Regression Suit = 1TC (Defect) + 3 to 4TC (high priority/core) + 1TC (extra feature) = 5 to 6TC**
* **For Regression testing, 1 to 2hr** will be required when we found defect in a build
* Types Regression testing types-

1. **Normal Regression testing –** When **got defect** against the US while doing TCE
2. **Final Regression testing –** When we have **completed a module** then we have to check complete functionality (ex. After 6 to 7 month we will do final regression testing)

* For a completed module, Final Regression testing will be done, All regression suite

**Testing Flow –**

1. New Build – Smoke Testing –If Build stable – System & functional Testing, Re-testing- If We got defect in System & functional Testing/ Re-testing – On Modified Build - Regression Testing

**OR**

1. New Build – Smoke Testing –If Build is not stable – Reject Build- developer sent New build - Smoke Testing - If Build stable - System & functional Testing, Re-testing- If We got defect in System & functional Testing/ Re-testing – On Modified Build - Regression Testing

**OR**

1. New Build – Smoke Testing –If Build stable – System & functional Testing, Re-testing- Testing completed for US

**OR**

1. Old Build – System & functional Testing, Re-testing- Testing completed for US

**OR**

1. Old Build – System & functional Testing, Re-testing- If We got defect in System & functional Testing/ Re-testing – On Modified Build - Regression Testing

**Interview Question –**

1. **Which testing you will performed when you will get defect?**

* **Answer-** We will do Re-testing & Regression testing
* **Ex.** Paytm –US = In Recharge module, Check browser planes display 🡪 Pass test data = Different Mobile, Operator, Circle – developer done coding **build (V14.00 = 270 code)** & sent the build for testing 🡪 Tester doing re-testing & in re-testing for operator Airtel, JIO, BSNL different browser planes shows but **VI operator no browser planes will display** 🡪 Tester will **raised a** **defect** & Inform to developer 🡪 Developer will fix the defect & prepared a **modified build (V14.00 = 320 code)** sent for testing 🡪 tester will do **regression testing** 🡪

Tester will check –

1. **Defect has been fixed -** VI operator – 9911223334, 9899001122, 9495969688, etc ensued defect has been fixed – **Retesting**
2. **Side impact on interconnected module -** Airtel, JIO, BSNL different browser planes shows- 9900223344, 9566990022, 9411223344, etc – **Regression Testing**
3. What are different environments present in your project
4. What are Unit Testing & what it will contains
5. What is Sanity testing & Smoke Testing? Which testing you will do?
6. Which testing you will perform when you got builds from developer?
7. **What is zero level testing**

* **Answer-** In SIT environments, first testing is sanity testing. These sanity testing called as level zero testing

1. How you are doing troubleshooting/ root clause?
2. How to identify the defect is it is not present in front end?

* **Answer**- **Frond end – Inspected elements – Network tab – Defect object click – JSON Data – Failure**

1. What you will check in smoke testing
2. What is different testing you have performed in your project?
3. What is re-testing & Regression
4. What is regression suite
5. What is final regression testing

# UAT-

* UAT means User acceptance testing
* UAT testing will be performed on client side
* After completion of all testing in SIT environments (In TCE Tester will prepared a **Test poof documents**), SIT tester will inform to developer & UAT tester throw mail (JIRA/ Azure DevOps tool)
* Developer will sent the build to UAT environments
* In UAT environments, Tester is present from **client side (USA)**
* In UAT (Client side) from Pune side developer also present (Dev environments access throw VPN)
* In UAT Testing, UAT Tester will check system & functional testing (End user used/ Business)
* 2 types of UAT

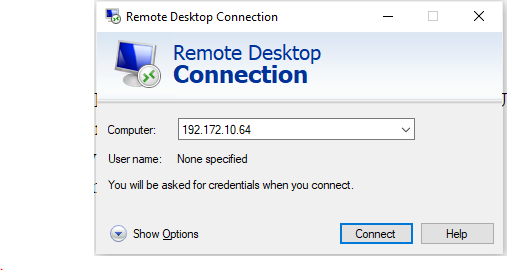
1. Alpha Testing
2. Beta Testing

|  |  |
| --- | --- |
| **Alpha Testing** | **Beta Testing** |
| Alpha Testing will be performed in **service based application**  Ex. Flipkard, Paytm, ICIC bank, etc | Beta Testing will be performed in **product/desktop based application**  Ex. GoldGYM, D-mart payment, etc |
| In alpha Testing, **Developer & tester** are present | In Beta Testing, **Developer & tester** are not present |
| If any **defect (production defect) has been found** then it is **fixed immediate** | If **defect (production defect) has been found** then it is **fixed in next version of application/ build** |
| **Client is interacting** more to developer & tester | **End user** **is interacting** more to Client (Play store, Company side comments, help & Support, Mail, etc) |

* **In my project**, we have done **Alpha Testing**
* If **UAT tester has found defect** in Testing, UAT tester will Crete defeat in JIRA/ Azure DevOps tool & inform to developer & SIT tester
* SIT tester will re-produce (check defect present) the defect in SIT environments,

1. If defect has been **present is SIT environments (missed in testing)**, SIT Tester immediately inform to developer & tell fix the defect ASAP. After fixing SIT tester will, check defect and Developer will deployed to UAT
2. If Defect has **not been** **present is SIT environments,** SIT tester will sent the test proof documents to UAT tester (Saying these defect not present in SIT). These defeats has been occurred due deployment problem

* In my project, I have done UAT Testing for 4 to 6 month. My UAT tester is on leave (Medical reason) then my PM told to do the UAT Testing.
* Tester will never work in both environments (SIT and UAT).
* UAT environments we will access throw **“Remote desktop”**



# Deployment / delivery -

* We are working in Scrum Agile process.
* In Scrum Agile process, deployment / delivery will, be done with 2 week
* Deployment / delivery done on **Saturday** (early)/ Monday

# Production Issue-

* When build is present in production environments/ End user using application/ feature
* If **issue/ defect has been occurred in production environments** then these issue are called as **production issue**

1. If these **production issue occurred from company side** then these arc called **“Hot fix”.** If hot fixed occurred then client sent **“Escalation mail”** to company /project team.
2. If we get Escalation mail (Hot fix) then SIT tester will be reproduce defect (Check defect present in SIT) in SIT environments. If **defect present in SIT environments** (We have **missed functionality in testing**) then SIT tester will inform to PM, Designer, Test lead & developer. PM will tell developer fix these issues ASAP and sent for testing. Developer will fix these issue sent to Testing. Once testing will, be done then these build deployed to production. (PM asked to SIT Tester to give the apology mail)
3. If we get Escalation mail (Hot fix) then SIT tester will be reproduce defect (Check defect present in SIT) in SIT environments. If **defect is not present in SIT environments** then SIT tester will inform to PM, Designer, Test lead & developer. PM will tell developer will check deployment process **(deployment issue/problem)** & redeployed all these changes. (PM asked to developer to give the apology mail)

* If these **production issues occurred from client side (**requirement not completed) then these arc called **“CR (Change request)”.** If any CR comes from client side, these **CR will be accepted at any point of time** but we will check **impact of these CR** on current development, current Testing & application present on production.

1. If impact of CR is more 🡪 PM will inform to client (Current Sprint Or Next Sprint)
2. If impact of CR is less 🡪 These CR will be completed with Current Sprint

# Testing terminology-

* **Error-** If developer some mistake in coding then it is called as Error.
* **Defect –** If tester found wrong functionality then it is called as Defect.
* **Bug-** If this defect developer has been accepted, then it is called Bug.
* **Issue –** If these bug is very critical (testing activity stop) these is bug are called issue.

# Testing approaches/ Scenarios based testing\*\*-

**Monkey Testing approaches -**

* If we have **more test cases to execution (100 to 200TC)** and we have **less time for execution** (1 to 2hr) **OR** If we have move the build from SIT (60 to 80TC) to UAT/ Prod with 1hr **OR** if we have got the **build at last moment** (last day of sprint - Friday at 6pm) and we have to deployed within 1hr then we will **performed Monkey Testing approaches**
* In monkey testing approaches, we will execute **only high priority test cases/ Main / core functionality related test cases**

**Exploratory Testing approaches-**

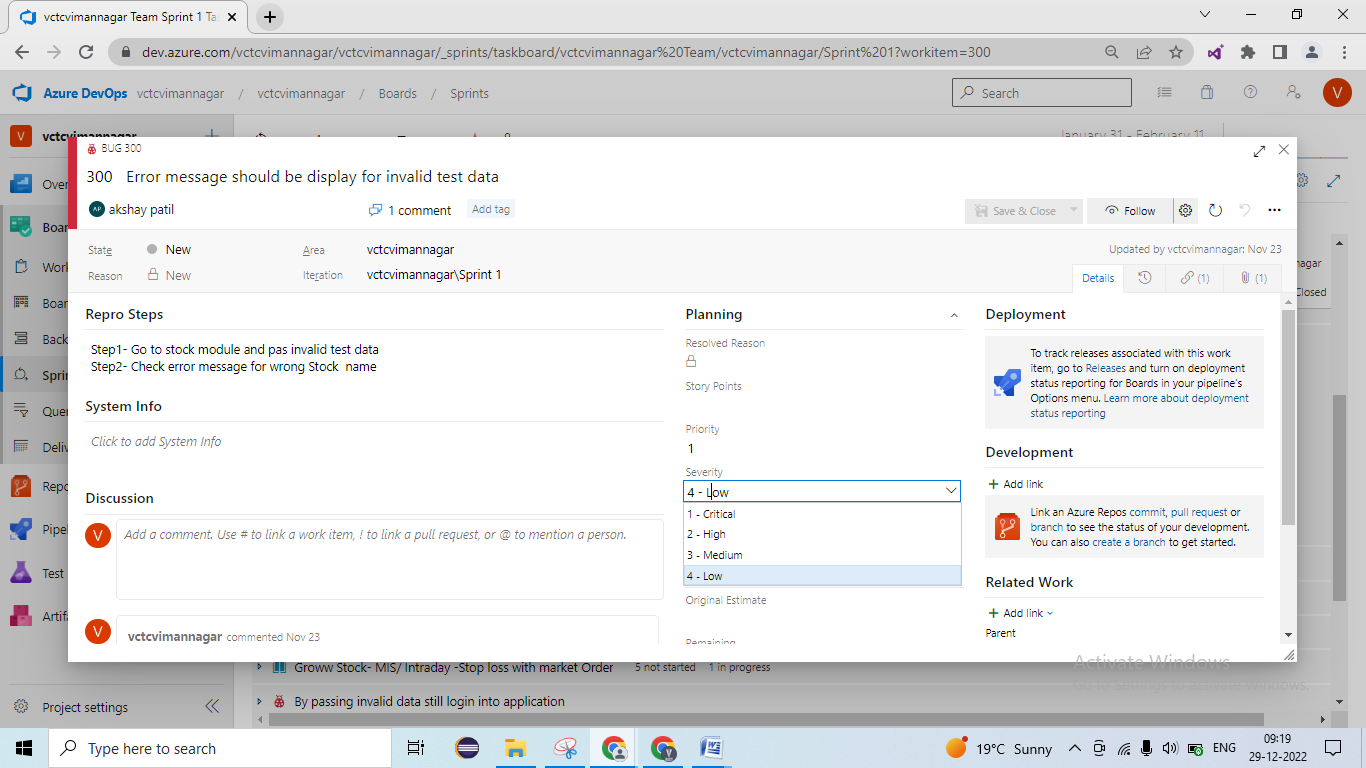
* If we have **more test data** for testing but we have **less knowledge about the functionality** **OR** If your **colleague (Tester) is on leave and you have test these US** then we will performed **Exploratory Testing approaches**
* In Exploratory Testing approaches, We have don’t have knowledge about the functionality then we will explore the knowledge about these functionality

**Ad-hoc Testing approaches-**

* If we have **more knowledge about the domain/ functionality** but we have **less test data** for testing then we will performed **Ad-hoc Testing approaches**
* These Ad-hoc Testing approaches will be **performed by Senior peoples**

# Priority & Severity\*\*-

* Priority & Severity term defines for Defect
* **Priority –** Priority term defines **how impact will occurred to client business.**
* Priority defines terms – Very High, High, Medium, and Low.
* **Severity -** Severity term defines how **impact will occurred to the application/ functionality**
* Severity defines terms – Critical, High, Medium, and Low.

****

1. **High Priority & High Severity-**

* **Ex.** At the time ofOrder place if we get error message (Payment not happing)
* **Ex.** Order place should placed but in database data is not stored

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

* **Ex.** If product prize is miss mated in frond end (original price = 900$ but display = 900rs)
* **Ex.** If client logo is not present in the application

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

* **Ex.** If rarely used functionality not working – Ex. Promo code is not working
* **Ex.** AfterOrder place invoice download button is not working but we got pdf in email id
* **Ex.** If OPT is not auto filled in application/ functionality

1. **Low Priority & Low Severity-**

* **Ex.** Spelling mistakes in thank message = thak message
* **Ex.** Logo color is different
* **Product Backlog** –
* **EPIC**- Module – Recharge Module
* **Feature** – Browser plane, Promo code
* **User Story** - Browser plane = 7 to 8 US
* Sprint Backlog

**Interviews Question-**

1. What is UAT Testing
2. What is your approaches, if defect occurred in UAT
3. When you are doing deployment in your project
4. What is your approaches, if issue occurred in production
5. Have you worked in production issue
6. What is hot fix? Have you handled hot fix?
7. What is your approach, if more test cases to execution (100 to 200TC) and we have less time for execution (1 to 2hr)?
8. What is your approach, if your colleague (Tester) is on leave and you have test these US
9. What is Priority & Severity and Give me example of High Priority & low Severity and Low Priority & High Severity

# Production issue –

**RETAIL PROJECT = WORLD WISE application🡪**

**DB = PRODUCT PRICE = 10 $ (1$ = 90RS)**

**Dollar to Euro convert – Logic is wrong = (1$ = 5 EURO \* USB\_FACTORE \* PROD\_VRB\_FACTOR) = 40EURO**

**INDAI country - FROND END = PRODUCT PRICE = 900RS**

**EUROPE country - FROND END = PRODUCT PRICE = 30 EURO**

