Full Stack Testing
Notes By
SURESH

Mr. K. Suresh – Profile

- Currently working as a Consultant with one of the Top MNC and has 18 + years of genuine experience in Software Testing.
- Hands-on experience with Selenium since 2009 and has 15 years of in-house Corporate Training experience.
- Successfully trained multiple batches on Software Testing
- Expert in Selenium Automation using Selenium RC and WebDriver. Heavy Programming using Java, TestNG and worked with QTP and Ruby with Watir as well.
- Proficient in Planning, Designing, Building and maintaining complex automation frameworks (Keyword, Modular, Hybrid and POM).
- Pro-active Team Leader and Manager with strong focus on documentation and process.
- Rich industry experience as worked with **SymphonyServices, Accenture, IGATE and Capgemini.**

Specializations:

- Expertise in creating Test Automation Frameworks for open source automation tools.
- Competence in open source automation APIs like Selenium RC and Web Driver.
- Proficiency in conducting Classroom, Corporate and Online Training on Selenium.

Manual Testing Course Content

Software Testing Fundamentals

- Introduction to Software Testing
- Software Development Process
- SDLC Real Time process Steps
- What is Software & Software Testing?
- Define QA Process
- History of Software testing
- Objective of Testing
- Why Testing required?
- When to start Testing
- Testing importance
- Definition, Basics & Types
- Software Testing as a Career Path (Skills, Salary, Growth)
- Why software has Defects
- Services based vs Product based Companies

Testing Roles and Responsibilities

- Software Test Engineer
- Real Time Job Role of Tester
- Senior Software Test Engineer
- Test Lead
- Test Manager

Software Testing Methods

- White Box Testing
- Black Box Testing
- Gray Box Testing
- Difference of Whitebox & Blackbox Testing

Software Development Life Cycle - SDLC

- What is SDLC?
- SDLC Phases
- SDLC Models
- Waterfall model
- V model
- Verification & Validation
- Agile Model

Agile Process Concepts

- What is agile?
- Why Agile is important
- Agile Testing principles
- What is mean by scrum master
- Roles of Scrum Master
- Sprint Planning
- Sprint Release
- Product Backlog
- What is Epic
- Concept of User Stories

- Defect Backlog
- Standup meeting
- Status meeting
- Scrum meeting

Software Testing Life Cycle - STLC

Understanding the requirements

- Requirements Specification
- Business requirement specification
- Software requirement specification
- Functional requirement specification

Test Plan Preparation

- Overview of Test Plan
- Entry and Exit criteria
- Test Plan template

Test Engineer Responsibilities

- LAB Checklist

Creation and working with Folder Structure

Test Scenarios

- Test Scenario Entry and Exit Criteria
- Test Scenario Template
- Test Scenarios Identification
- Writing Test Scenarios for application

Test Cases

- Test cases Entry and Exit Criteria
- Test cases Template
- Test cases Identification
- Test Design Guidelines
- Writing Test cases for application
- Good Test Case design steps
- Test Data Preparation

Test Case Design Techniques

- Equivalence Class Partitioning
- Boundary Value Analysis
- State Transition
- Decision Table
- White box Testing Techniques

Software Testing Types

- Smoke Testing
- Sanity Testing
- Re-Testing
- Regression Testing
- Static Testing
- Dynamic Testing
- Ad-hoc Testing
- Functionality Testing
- Usability Testing
- Compatibility Testing
- Data Base Testing

- Interface Testing
- Performance Testing
- Security Testing

User Acceptance Testing

- Alpha Testing
- Beta Testing
- UAT Testing

Test Execution

- When to start Test Execution
- Process to start test execution
- What is build
- Build Release process
- Executing Test Cases on multiple builds
- Test Cases Execution Status

Bug/Defect management

- Defect/Bug Life Cycle
- Defects Reporting
- Defects Reporting Template
- Defects Reporting & Re-Testing
- Defects Closing
- Severity and Priority
- Defect /Bug/Error/Failure
- Defects in Real Time application
- How to find more bugs

Test/Project management Tool: JIRA

- Introduction to Jira
- Features of Jira
- Test Case Design in Jira
- Creating Test Cycle
- Test Execution in Jira
- Bug Reporting using Jira
- Jira Dashboard

Status Reports Process

- Daily Status Report
- Daily Defect Report
- Weekly Status Report
- Monthly Status Report

Test Closure

- Criteria for Test Closure
- Test Closure process
- Test Summary Reports
- When testing need to be stopped

What is Software Testing

"Software testing is a process of executing the application with the intent of finding the defects by
comparing the output behavior of the application with expected behavior (requirement)."

In other words it's comparing the actual behavior of an application with expected behavior.

Why Software Testing

Humans make mistakes all the time!!

"Software testing is really required to point out the defects and errors that were made during
the development phases".

We humans can't identify our mistakes in a work done by us. We should get someone else to check our
work because another person may identify the mistakes done by us. In the same way software
developers may not identify the mismatches in a program or application implemented by them which
can be identify by the another department called Software Test Engineer.

Benefits of Software Testing

"Software testing helps in finalizing the software application against business requirements."

Software testing makes sure that the testing is being done properly and hence the system is ready for
the customers to use.

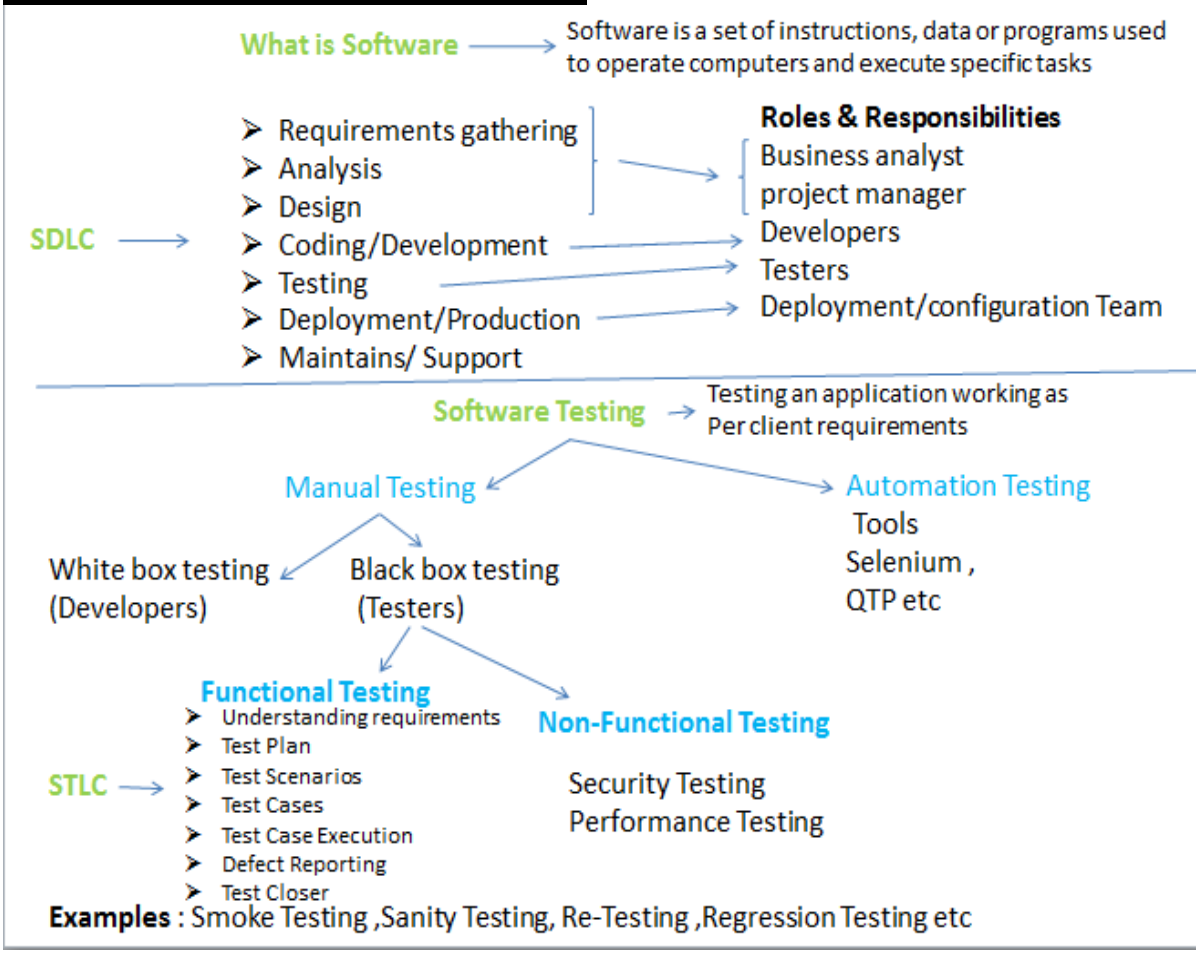
Below are few benefits of software testing.

- Finding the defects before delivery
- Gaines the confidence about quality
- To Prevent defects
- Ensure the requirements are delivered to client

What is Quality

"Software quality is nothing but delivering a bug free application and delivered on time withall requirements."

Overview of Software Testing



Why Testing Required

- To identify the defects in development phases
- To ensure Quality of the product
- Saves Money as defect identified in earlier stages
- To build customer confidence and business

Why Testing Job

- Software Testers Are Made for Challenging Work Environments
- You Can Enjoy Every Day of Work
- Flexible and Fun Work Environment
- It's Creative
- It Is a Secure Career Path
- There Is Attractive Remuneration and Room for Growth
- An Academic Background Isn't a Necessity

Testing Roles & Responsibilities

- Software Test Engineer(STE) – (0-4yrs)
- Sr.Software Test Engineer(SSTE) – (4-6yrs)
- Test Lead (TL) – (6-8yrs)
- Test Manager(TM) – (8-10yrs)
- Sr.Test Manager(STM) – (10+yrs)
- Assoc.Dir. – Software Testing
- VP – Software Testing

Software Test Engineer Responsibilities

- Understating the requirements of the application
- Identifying required Test Scenarios of the project
- Designing and preparing Test cases to validate application
- Execute test cases to validate application
- Logs Test Results(How many Test cases passes/failed)
- Defect Report and Tracking
- Retest fixed defects of previous builds
- Performed various Types of testing assigned by Test Lead(Sanity ,Functionality , Usability, Compatibility , etc)
- Preparing and Sending of status Reports to Lead on assigned tasks
- Participated in regular meetings , team meetings by lead & Manager
- Creating automation scripts for Regression testing

Sr. Software Test Engineer Responsibilities

- Same as test engineer responsibilities

+

- Participates in Review of Test Scenarios ,Test cases and defects
- Some Times involved in Test Plan preparation also.
- If required Leading the team when Team Lead is on Vacation

Test Lead Responsibilities

- Task Preparation and allocation to Team members
- Training Team members
- Team Management
- Test Scenarios & Test Cases Reviews
- Bug Reviews
- Preparation of Build summary report
- Conducting meetings with Team members
- MOM Preparation
- Test Plan preparation

Test Manager Responsibilities

- Project plan and Review of Test Plan
- Effort estimates
- Project Management
- Training Plan – Identify training need based on Resource skills
- Preparing monthly reports
- Client Communications
- MOM
- Provide regular status updates to core team
- Scheduling meetings with Development and Testing Team

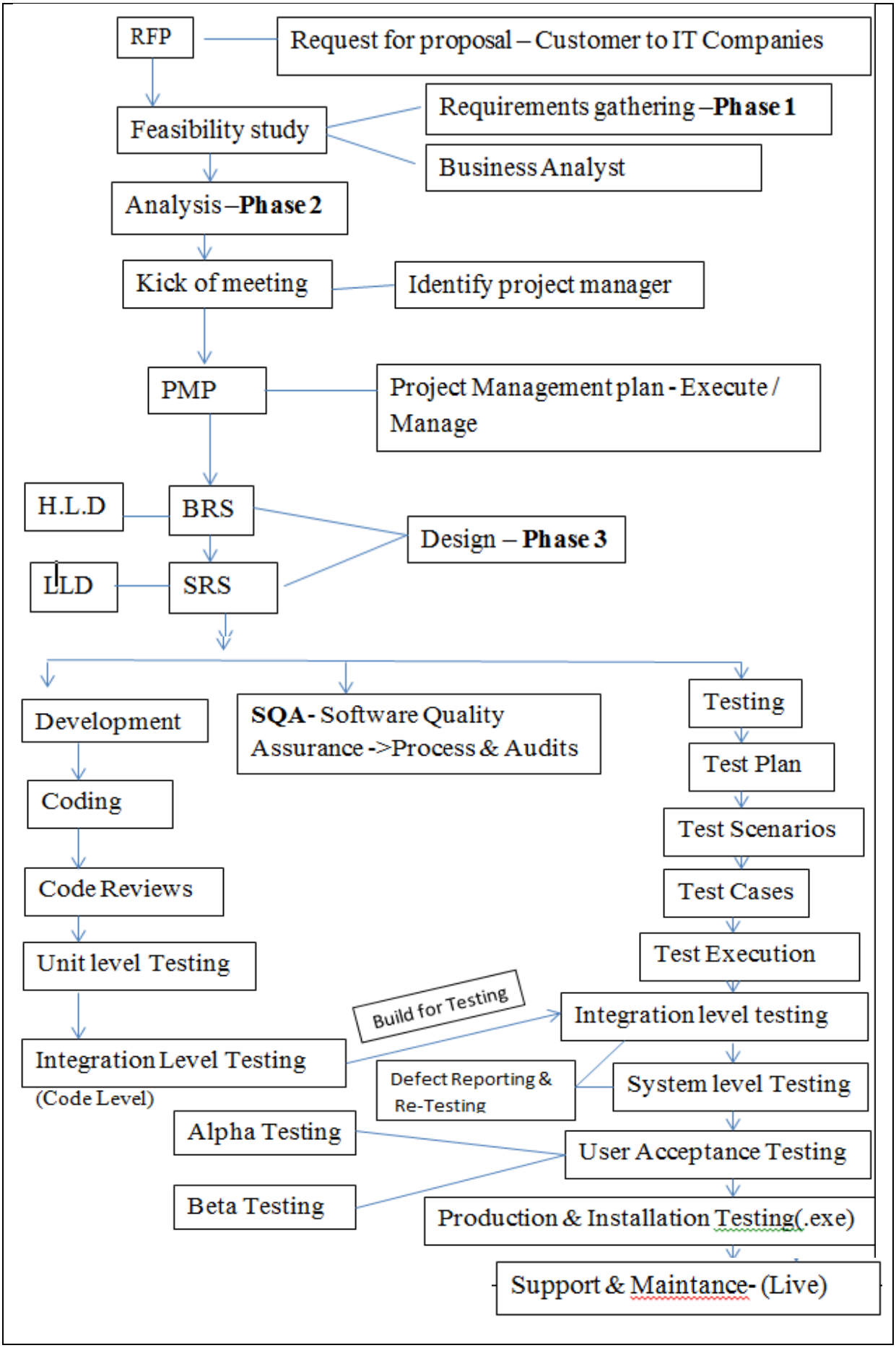
SDLC–Software Development Life Cycle

SDLC is a process followed for a software project, within a software organization. It consists of adetailed plan describing how to develop, maintain, replace and alter or enhance specific software.

Phases in SDLC :

- Requirements gathering
- Analysis
- Design
- Coding/Development
- Testing
- Deployment/Production
- Maintains/Support

SDLC Real Time Process Implementation :

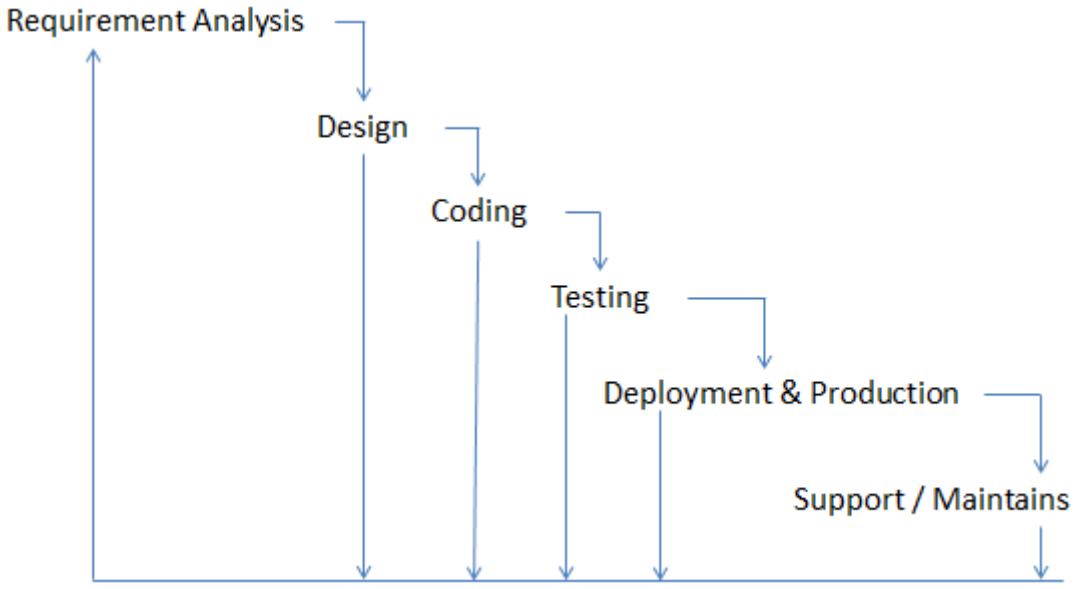

RFP
Request for proposal – Customer to IT Companies
Feasibility study
Requirements gathering –Phase 1
Business Analyst
Analysis –Phase 2
Kick of meeting
Identify project manager
PMP
Project Management plan - Execute / Manage
H.L.D
BRS
Design – Phase 3
LLD
SRS
Development
SQA- Software Quality Assurance ->Process & Audits
Testing
Test Plan
Coding
Test Scenarios
Code Reviews
Test Cases
Unit level Testing
Test Execution
Build for Testing
Integration level testing
Integration Level Testing
(Code Level)
Defect Reporting & Re-Testing
System level Testing
Alpha Testing
User Acceptance Testing
Beta Testing
Production & Installation Testing(.exe)
Support & Maintance- (Live)

Types of SDLC Models

- Waterfall model **or** Life Cycle Model **or** Linear Sequential Model
- V- Model **or** Verification & Validation Model
- Agile …etc

Waterfall model

- This model suggests a systematic and sequential approach to software development that begins at requirements analysis and progress through all life cycle phases sequentially
- Suitable for projects where requirements are clearly defined
- Small and medium term duration
- Having Domain knowledge



Advantages :

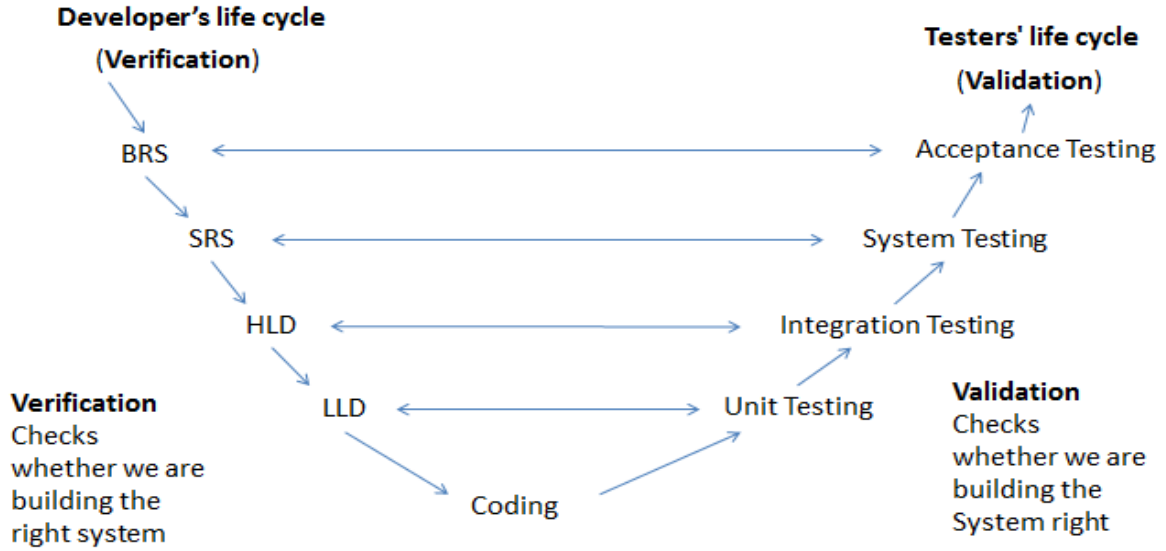
- Project under Control
- Pre-defined outputs at every phase
- Tracking changes is easy

Disadvantages:

- Not suitable for requirements changes
- Does not support going back to previous phase
- If any defect found need to go back to the originating phase

V - model

- V model means verification and validation model, the V shaped life cycle is a sequential path of execution process.



Advantages :

- Simple and easy to use
- Testing activities like planning and test design will be done before coding
- Testing planned parallel to development
- Bug detection in early phases

Disadvantages:

- If any changes happen in midway, then the test document along with requirements documents has to be updated.
- Not Suitable for large and complex projects
- The client sees the only final project ,not intermediate modules

Difference between Verification &Validation :

Verification	Validation
check whether we are developing the right product or not.	check whether the developed product is right.
Verification includes different methods like Inspections, Reviews, and Walkthroughs	In the validation testing, we can find those bugs, which are not caught in the verification process.
In verification testing, we can find the bugs early in the development phase of the product.	Validation includes testing like functional testing, system testing, integration, and User acceptance testing.
The goal of verification is application and software architecture and specification.	The goal of validation is an actual product.

Agile - Process

Overview of Agile Process

- Customer satisfaction by rapid delivery of useful software.
- Welcome changes in requirements ,even late in development
- Working software is delivered frequently (weeks rather than months)
- Close ,daily cooperation between business professionals and developers
- Continuous attention to technical excellence and good design

Project Name : ONLINE BANKING

***Product Backlog 1**– **LOGIN** : High Level Requirement
Epic1 - Login into application : Requirement
User Story1 - login into application as **employer** : Low Level Requirement

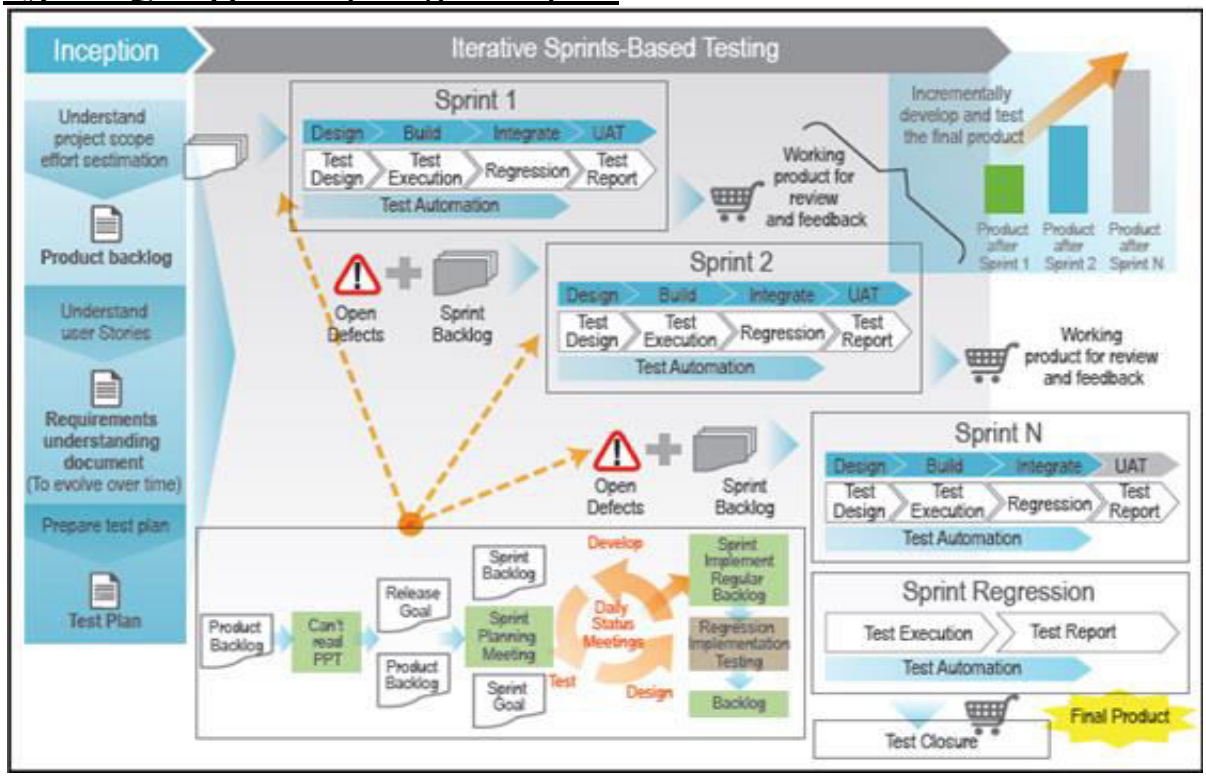
- Enter empid
- Enter password
- Click on login button
- Emp home page should be displayed

User Story2 - Login into application as **customer**

- Enter user id
- Enter password
- Click on login button
- Customer home page should be displayed

***Product backlog 2**- **Profile**
Epic2 - Profile Management
User Story3- Create new profile
User Story4- Update existing profile
***Product Backlog 3** - **Transfer**
Epic3 - Same bank transfer
User Story5 - enter customer account details and transfer amount
Epic4 -Other bank transfer
User Story5 - Add beneficiary account details
User Story6 -Approve beneficiary details
User Story7-Transfer amount

Typical QA Approach for Agile Projects



Agile - Terms

- **User Story** – A shorthand requirements document.
- **Product Backlog** -- A prioritized list of stories that are waiting to be worked on.
- **Product Owner** -- person whom holds the vision for the product.
- **Scrum Master Role**The Scrum Master is a facilitator for the team and product owner.
- **Sprint** -- A development process
- **Stand-up Meeting**– a short (15 minutes or less) daily meeting during which team members report on what they have accomplished since the last meeting, what they plan to accomplish today and report any impediments or blockers to making progress.
- **Scrum Meeting** – To discuss on Sprint planning, development and Review.

Stand-up Call Template

	SURESH IT ACADEMY LEARN TO LEAD			Daily Stand Up Call	
		Project name		Project Lead	
		Project manager		ScrumMaster	
Date	Resource Name	Yesterday Work	Today Work	Any Blocker	
26-Dec	Naveen	Completed Design of Test Scenarios for Search module	Started Design of TestCases		
	Suresh	In-Progress of Desiging Test Scenarios	completing design of Test Scenarios		
	Hari	On Leave	Started Test Execution	Test URL is down for 2 hours	
27-Dec	Naveen	In-Progress of Desiging Test Cases	Working on Desiging Test Cases		
	Suresh	In-Progress of Desiging Test Cases	Working on Desiging Test Cases		
	Hari	In-Progress Test Execution	Working on Bug Reporting		

Agile Testing

Agile Process / Model

Agile is an iterative development methodology that enables teams to deliver value to their customers faster. Whereas requirements and solutions evolve through the collaborative efforts of software teams and their customers or end users.

Agile Testing

Agile testing is a software testing approach that follows the principles and rules of agile software development. Agile testing and coding are done incrementally and interactively, building up each feature until it provides enough value to release to production.

Agile Testing Principles

- **Continuous testing**: agile teams perform tests regularly to make certain that the product is continuously progressing. Testing is done in conjunction with development.
- **Continuous feedback**: testers provide continuous feedback to team members. Members regularly receive feedback regarding quality rather than requirements.
- **Involving the whole team**: testers, developers and business analysts all test
  the software.
- **Quick feedback**: the business team participates in each iteration, ongoing feedback reduces the time it takes to get feedback on development work.
- **High-level software quality**: teams test the software to ensure the code is clean and tight. Through regular testing of the software, issues can be easily detected and fixed in the same iteration as they are developed.
- **Less documentation**: teams use a reusable checklist. Agile development focuses on current customer needs rather than comprehensive, documented requirements and instructions.
- **Customer satisfaction**: customers are exposed to their product during development. They can adapt and update requirements as development progresses. Tests can be modified to updated requirements.
- **Respond to change**: Agile testers should also be flexible and adaptable. They should learn how to adapt to the changing needs of the product and the marketplace.
- **Simplified & clean code:** All the defects which are raised by the agile team are fixed within the same iteration and it helps in keeping the code clean and simplified.
- **Everyone Tests:** In conventional SDLC, only test team tests while in agile including developers and BA's test the application.

Agile Testing Life Cycle:

- Planning Sprints
- Daily Scrums
- Designing Test Cases
- Quality Verification and Validation
- Review Product Stability
- Regression and Deployment

Planning Sprints: Sprints are the core of Agile methodologies, an approach that takes large, complex projects and fragments them into smaller manageable pieces. It is the pre-agreed time period in which the team has to work upon the set of requirements and conclude within the duration.

Daily Scrums: This phase includes the everyday morning meetings so that we can check on the status of testing and set the goals for the day. all the team member discuss their status and set their goal for a whole day in this phase.

Designing Test Cases: It is important for the testing team to maintain a cadence with the development team. The testing team designs the test cases as per the requirements provided in the requirements documents and project design documents.

Quality Verification and Validation: This process is to check the quality of the developed software. As soon as the development gets completed and made available for QA, the testing process starts. The team works together to execute testing in an agile environment.

Review Product Stability: Usually, a customer wants to add or remove some of the features, change their product, or even change their complete idea. Agile is equated to iterative development, that is, New requirements can be accommodated at any stage of the development process without adding to the complexities. **Regression and Deployment:** To check the impact on the existing functionality, the manual and automated test cases are run after new user stories are added to the agile development process in order to deploy the quality product.

Agile Methodologies/Frameworks

- Kanban
- Scrum
- Extreme Programming (XP)
- Lean Software Development

Kanban : Originating from the Japanese language, the translation of the word 'Kanban' is "visual board or signboard" and is connected to the concept of "just in time" .This method uses visual methods for developing and managing projects.

Scrum : One of the most popular agile methodology examples is the agile scrum development methodology, which is depicted by various cycles of

development. Similar to Kanban, Scrum breaks down the development phases into stages or cycles called 'sprints'. The development time for each sprint is maximized and dedicated, thereby managing only one sprint at a time.

Extreme Programming (XP) : Extreme Programming (XP) is a methodology that emphasizes teamwork, communication, and feedback. It focuses on constant development and customer satisfaction. Similar to scrum, this method also uses sprints or short development cycles. This is developed by a team to create a productive and highly efficient environment.

Lean Software Development : Lean product development focuses on producing only what the product actually needs. It optimizes time and resources and reduces unnecessary activities.

The team works with Minimum Viable Products (MVPs) that are released to the customers as early as possible. Customer feedback is then collected and included in future software iterations.

Agile Terms

- **Burn-down chart:** A chart that demonstrates how quickly you and your teammates are burning quickly through your customer's user stories.
- **story point:** A story point in Scrum is the unit for the estimation of total efforts that are required to perform or complete a particular task
- **Velocity:** It is a simple method to measure the rate(hours) at which scrum development groups constantly deliver business value.
- **Scrumban :** A Scrumban is a management framework and Kanban-based model for software development. It is used in software projects that require continuous maintenance, problem-solving, error fixation, etc. This model is chosen to complete a project within minimum time.
- **Release candidate:** A release candidate is a version of a software application that is ready for release, but has not yet been approved by the powers that be. In order to become a release candidate, a software application must go through a rigorous testing process to ensure that it meets all the necessary criteria for release.
- **Sprint Zero :** Sprint Zero is a step to be taken ahead of any sprint. Hiring people, setting development environment, preparing backlog, are some of the activities that fall under Sprint Zero.
- **Planning poker:** Planning poker, also known as Scrum Poker, is a card-based agile technique that is used for planning and estimation with point of 0,1,2,3,4,5 etc. To start a session of planning poker technique, the agile user story is read by the product owner.
- **Build Breaker:** A Build Breaker is usually a bug in the software created because of any execution fail. It may stop the build process, cause unacceptable warnings, and even lead to failures in the automated test environment.

- **Agile manifesto:** The Agile Manifesto is a document that sums up the 12 agile principles that guide the Agile framework. Every Agile methodology strictly follows the principles and practices outlined in the Agile Manifesto.
- **Retrospective:** a session where the team and scrum master reflect on the process and make commitments to improve.
- **Sprint Retrospective:** A meeting to reflect on the sprint and identify areas for improvement.
- **Product Owner** : The product owner is a role on a product development team responsible for managing the product backlog in order to achieve the desired outcome that a product development team seeks to accomplish.

Agile Advantages :

- **Customer Satisfaction:** Regular delivery and adaptability to changing requirements keep customers happy.
- **Flexibility:** Agile embraces change, making it easy to adjust project direction.
- **Faster Time to Market:** Short sprints lead to quicker releases and early value delivery.
- **Improved Quality:** Continuous testing and iterative development improve product quality.
- **Better Collaboration:** Agile fosters teamwork between cross-functional teams and stakeholders.
- **Transparency:** Regular updates and reviews ensure clear visibility into progress.
- **Risk Management:** Early feedback helps identify and address issues sooner.
- **Employee Satisfaction:** Empowered teams and manageable workloads increase morale.
- **Cost Control:** Focus on high-value features and regular checkpoints help manage costs.
- **Continuous Improvement:** Regular retrospectives promote process improvement.
- **High-Quality Products:** Focus on delivering value and refining features continuously.
- **Competitive Advantage:** Agile teams can quickly respond to market changes and customer needs.

Disadvantages :

- **Requires High Commitment:** Agile demands continuous involvement from both the team and the customer, which can be time-consuming.
- **Requires Skilled Teams:** Agile requires teams with a high level of skill and experience to work efficiently, which might be a challenge for inexperienced teams.
- **Time-Consuming Meetings:** Frequent meetings like daily stand-ups, sprint reviews, and retrospectives can become time-consuming, affecting productivity.
- **Customer Availability:** Agile requires regular input from customers or stakeholders, which may not always be feasible.
- **Inconsistent Results:** The flexibility of Agile means that outcomes may vary depending on the team's ability to adapt and prioritize correctly.
- **Short-Term Focus:** Agile's emphasis on immediate delivery may sometimes lead to neglecting long-term vision and architecture.
- **Difficult for New Teams:** Agile can difficult for new teams that are not yet familiar with its principles, practices, and roles.

Software Testing Methodologies

- Block Box Testing
- White box testing

Black Box Testing:

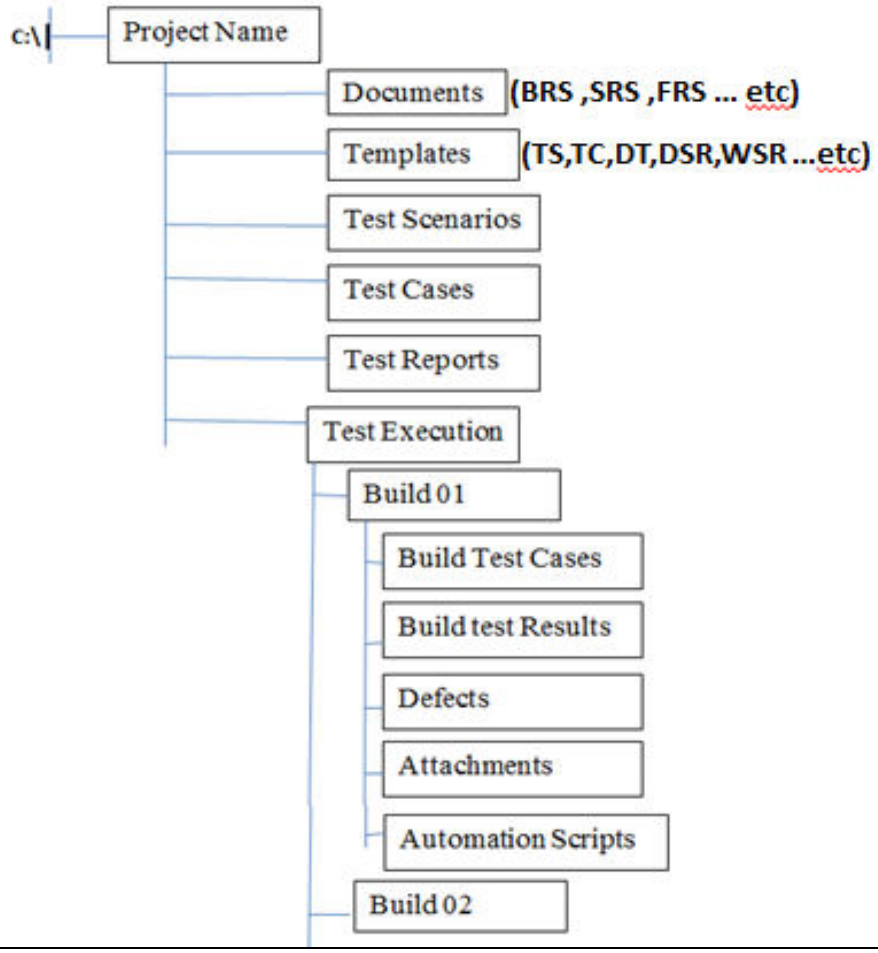
Black box testing involves testing a system with no prior knowledge of its internal workings. A tester provides an input, and observes the output generated by the system under test. This makes it possible to identify how the system responds to expected and unexpected user actions, its response time, usability issues and reliability issues.

Black box testing is a powerful testing technique because it exercises a system end-to-end. Just like end-users "don't care" how a system is coded or architected, and expect to receive an appropriate response to their requests, a tester can simulate user activity and see if the system delivers on its promises. Along the way, a black box test evaluates all relevant subsystems, including UI/UX, web server or application server, database, dependencies, and integrated systems.

White Box Testing

White box testing involves testing an application with detailed inside information of its source code, architecture and configuration. It can expose issues like security vulnerabilities, broken paths or data flow issues, which black box testing cannot test comprehensively or at all.

Real Time Folder Structure To implement STLC



STLC – Software Testing Life Cycle

Software Testing Life Cycle (STLC) is a process used to test software and ensure that quality standards are met. Tests are carried out systematically over several phases. During product development, phases of the STLC may be performed multiple times until a product is deemed suitable for release.

- STLC is an part of Software Development Life Cycle (SDLC).STLC deals only with the testing phases.
- STLC starts as soon as requirements are defined.
- STLC provides a step-by-step process to ensure quality software.

STLC Phases

STLC has the following different phases but it is not mandatory to follow all phases. Phases are dependent on the software or the product, time and resources allocated for the testing and the model of SDLC that is to be followed.

- Requirements understanding(BRS, SRS, FRS, Mock-ups)
- Test Plan
- Test Scenarios
- Test Cases
- Test Execution
- Bug Reporting & Re-Testing
- Test Closer

Note:
BRS – Business Requirement Specification
SRS – Software Requirement Specification
FRS – Functional Requirement Specification
Mock-Ups : Screenshots

Requirements understanding:
When the requirement docs are ready and shared with testing team,then testers starts understanding the requirements to design testscenarios & testcases.
Test Plan:
Test Lead plans the test strategy & approach which is outlined in a test plan document. This strategy includes tools needed, testing steps, and roles and responsibilities. Part of determining this strategy is a risk and cost analysis and an estimated timeline for testing.
Test Scenarios:
(What need to tested)-Design the test scenarios based on scope and criteria's.
Test Cases:
(How to be tested)test cases are created. Each case defines test inputs, procedures, execution conditions, and anticipated results. Test cases should be transparent, efficient, and adaptable. Once all test cases are created, test coverage should be 100%.
Test Execution:
features are tested in the deployed environment, using the established test cases. Expected test results are compared to actual and results are gathered to report back to development teams.
Bug Reporting & Re-Testing : Reporting the bugs using bug template
Test Closer:
This is the last phase of the STLC, during which a test result report is prepared. This report should summarize the entire testing process and provide comparisons between expected results and actual. These comparisons include objectives met, time taken, total costs, test coverage, and any defects found.

Test Plan

Test Plan – A document describing the scope, approach, resources & schedule of testing activities. It identifies test items ,the features to be tested, the testing tasks who will be doing and any risks …etc

Entry Criteria to prepare Test Plan :

- Approved PMP
- Approved SRS
- Test Plan guidelines
- Test Plan template

Exit Criteria of preparing Test Plan :

- Test Plan should be reviewed and approved.

Contents in Test Plan

1. Introduction
  1.1 Test plan objectives
2.Scope of this document
3.Test Strategy
  3.1 Smoke Testing
  3.2 Sanity Testing
  3.3 Functional Testing
  3.4 Database Testing
  3.5 Cross browser Testing
  3.6 Automation Testing
4.Environment Requirements
5.Test Schedule
6.Control Procedures
  6.1 Reviews
  6.2 Bug Reviews
7.Functions To be tested
8.Functions not to be tested
9. Resources & Responsibilities
10. Deliverables and mile stones
11. Defect Management
12. Dependencies
    12.1 personal dependencies
    12.2 Software dependencies
    12.3 Hardware dependencies
    12.4 Test Data & Database
13. Risks
14. Tools
15. Documentation
16. Approvals

17. Entry / Exit for each Testing activity
18. Test Suspension
19. Test Resumption
20. Test completion

Test Scenarios

Identify all possible areas to be tested **or** What is to be tested.
A Test Scenario is a statement describing the functionality of the application to be tested. It is used for end-to-end testing of a feature and is generally derived from the requirement document.
Test scenarios can serve as the basis for lower-level test case creation. A single test scenario can cover one or more test cases. Therefore a test scenario has a one-to-many relationship with the test cases.

How to create a Test Scenario

- Carefully study the Requirement Document – Business Requirement Specification (BRS), Software Requirement Specification (SRS), Functional Requirement Specification (FRS) pertaining to the System Under Test (SUT).
- Isolate every requirement, and identify what possible user actions need to be tested for it. Figure out the technical issues associated with the requirement. Also, remember to analyze and frame possible system abuse scenarios by evaluating the software with a hacker's eyes.
- Enumerate test scenarios that cover every possible feature of the software. Ensure that these scenarios cover every user flow and business flow involved in the operation of the website or app.
- After listing the test scenarios,Get the scenarios reviewed by a team.

Entry Criteria to Identify Test Scenarios :

- Approved Test plan
- Approved SRS
- Test Scenarios Guidelines
- Test Scenario template

Exit Criteria for Test Scenarios :

- Test Scenarios should be reviewed &approved(mapping test scenarios with requirements)

Test Scenarios Template

Company Logo			Proiect ID		Reviewed By	
			Project Name		Date Reviewed	
			Identified by		Approved By	
			Date Identified		Date Approved	
TS#	Req#	Main Functionality	TestScenario Description	Test Case Name/TC ID	Document ID/Reference	Comments

Sample Example :
Company Logo : Logo of the company
Project ID : ID of the Project
Project Name : Name of the Project
Identified by : Name of the tester
Date Identified : Day of the started writing the Test Scenarios
Reviewed By : Test Lead Name
Date Reviewed : Future Date
Approved By : Test Lead Name
Date Approved : Future Date
TS# : TestScenario ID
Req:# : Requirement ID
Main Functionality : PageName
TestScenario Description : What need to be Tested
Testcasename /ID : TestCase ID for corresponding TestScenario
Document ID / Reference : Which document we are refering to design the Scenarios
Comments : In case of any speficdescription need to be mention.

TS#	Req#	Main Functionality	TestScenario Description	Test Case Name/TC ID	Document ID/Reference	Comments
TS_001	3.2	Login Page	Verify Login functionality		SRS	
TS_002	3.2	Login Page	Verify forget password functionality		SRS	
TS_003	3.2	Login Page	Verify new user register functionality		SRS	

Test Case

What is Test case?

A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

It is an in-details document that contains all possible inputs (positive as well as negative) and the navigation steps, which are used for the test execution process. Writing of test cases is a one-time attempt that can be used in the future at the time of regression testing.
Test case gives detailed information about testing strategy, testing process, preconditions, and expected output. These are executed during the testing process to check whether the software application is performing the task for that it was developed or not.

Test case helps the tester in defect reporting by linking defect with test case ID. Detailed test case documentation works as a full proof guard for the testing team because if developer missed something, then it can be caught during execution of these full-proof test cases.

The benefits of an effective test case include:

- Guaranteed good test coverage.
- Reduced maintenance and software support costs.
- Reusable test cases.
- Confirmation that the software satisfies end-user requirements.
- Improved quality of software and user experience.
- Higher quality products lead to more satisfied customers.
- More satisfied customers will increase company profits.

Entry Criteria to Identify Test Cases :

- Approved Test plan
- Approved SRS
- Approved FRS
- Approved Test Scenarios
- Test Case Guidelines
- Test Case Template

Exit Criteria for Test Cases :

- Test Cases should be reviewed &approved(mapping test scenarios with requirements)

Test Case Template

Company Logo		Project History				Test Cases				
		Project ID				Total No. of TCs				
		Project Name				Passed				
						Failed				
		Test Case History				Not Executable				
		Created By		Date Created		Defects Reported				
		Reviewed By		Date Reviewed						
		Approved By		Date Approved						
		Test Information								
		Test Executed By		Date Executed						
		Version		Build						
TC #	TS#	Test Design/ Steps	Input Data	Expected Result	Actual Result	Pass	Fail	Not Executable	Comments	Defect Id

--Sample Example :
Company Logo : Logo of the company
Project ID :ID of the project
ProjectName : Name of the Project
Created By : Tester name
Review By : TestLead name
Approved By : TestLead name
Date Created : Day of the Started writing TestCases
Date Reviewed : Future Date
Date Approved : Future Date
Test Executed By : Tester name
Version : Version of the TestCase Document
Date Executed : Form Which Date TestCase Execution need to start
Build :build number
Total No of TestCases : The Total number of TestCases Created for whole application.
Passed: Total number of TestCases passed
Failed: Total number of TestCase Failed
Not Executable: not executed test case count
Defects Reported: Bugs reports for the Whole application.
TC# : Test Case ID
TS# : Test Scenario ID mentions in Test Scenario Document
Test Design Steps : Steps To be Followed for Testing particular Functionality
Input Date : Test Data need to be used to test particular Functionality
Expected Result : Result Excepting Based on Requirement Document
Actual Result : Result displayed in Application
PASS : Expected Result & Actual Result both are matching consider as PASS
FAIL : Expected Result&Actual Result both are not matching consider as FAIL
Not Executable : Test Case Which are not executed for that current Release
Comments : In case of any speficdescription need to be mention.
Defect ID : Bug ID for failed test Cases

TC #	TS#	Test Design/ Steps	Input Data	Expected Result	Actual Result	Pass	Fail	Not Executable	Comments	Defect Id
TC_001	TS_001	1.Launch browser 2.Enter URL 3.Enter Valid UserName 4.Enter Valid Password 5.Click on Login button	URL : Username : Password :	1.Login should be successful 2.Search hotel page need to display						
TC_002	TS_001	1.Launch browser 2.Enter URL 3.Enter Invalid UserName 4.Enter Valid Password 5.Click on Login button	URL : Username : Password :	Error message need to displayed as "Invalid UserName"						

Test Case Design Techniques

Test Case: Design techniques can broadly split in to 2 categories.

Black box techniques

White box techniques

Black box techniques

- Equivalence Class Partitioning(ECP)
- Boundary Value Analysis(BVA)
- State Transition
- Decision Table / Cause Effect Table

White box techniques

- Statement Testing
- Branch/Decision testing
- Data flow Testing
- Branch condition testing

Equivalence Class Partitioning(ECP) :

Divide a set of test conditions into groups which is having similar behavior to reduce number of test cases.

Age [ ] (accepts 1 to 60)

Invalid	Valid	Invalid
0	1 to 60	>=61

Boundary Value Analysis(BVA):

BVA is based on testing the boundaries of condition

Formula : Minimum ,maximum ,Min-1 ,Max+1

Valid Boundaries : Minimum , Maximum

Invalid Boundaries : Min-1 ,Max-1

Name [ ] (accepts 5 to 10 characters)

Invalid (Min-1)	Valid (Min,Max)	Invalid (Min+1)
4	5 , 10	11

State Transition Table:

Application provides different output for same input based on previous stage

Username	Password	Result
Correct	Wrong	Invalid Password
Correct	Wrong	Invalid Password
Correct	Wrong	Invalid Password
Correct	Wrong	Account Locked

Decision Table :

Testing with different combination of inputs which produce different results.

Username	Password	Result
Correct	Correct	Login Successful
Correct	Wrong	Invalid Password
Wrong	Correct	Invalid Username
Wrong	Wrong	Invalid Username

Test Execution - Bug or Defect Management

Bug / Defect Template

Defect ID		Assigned To				
Status		Browser				
Severity		Found in Version				
Priority		Found in Build				
Module		Fixed in version				
Reported By		Fixed in build				
Title						

Description

Steps To Re-Produce :

1.

2.

3.

Expected Result :

Actual Result :

Sample Example :
Defect ID :ID of the new Defect
Status : Status of the Bug
Severity : Need to provide based on Functionality
Priority : Need to provide based on Client Expectation
Module : Page Name
Reported By : Tester Name
Assigned to : Developer or Test Lead Name
Browser Name : Name of the Browser
Found in Build: in which build bug had found
Found in Version: number of the version
Fixed in Version : Developer will provide this
Fixed in Build : Developer will provide this

Bug / Defect Template			
Defect ID	DEF_001	**Assigned To**	DEV
Status	NEW	**Browser**	Chrome
Severity		**Found in Version**	1
Priority		**Found in Build**	1
Module	Search Hotel Page	**Fixed in version**	
Reported By		**Fixed in build**	
Title	Validation is not working for check in date (accepting previous date)		

Description : Steps To Re-Produce :
1.Launch Browser .2.Enter URL
3.Enter valid Username
4.Enter valid password
5.Click on login
6.Select all required feilds
7.Enter check in date as previous date
8.Click on search
Expected Result :
Error message should be displayed as "check in date should be current date or feature date"
Actual Result :
Search hotel page is displayed

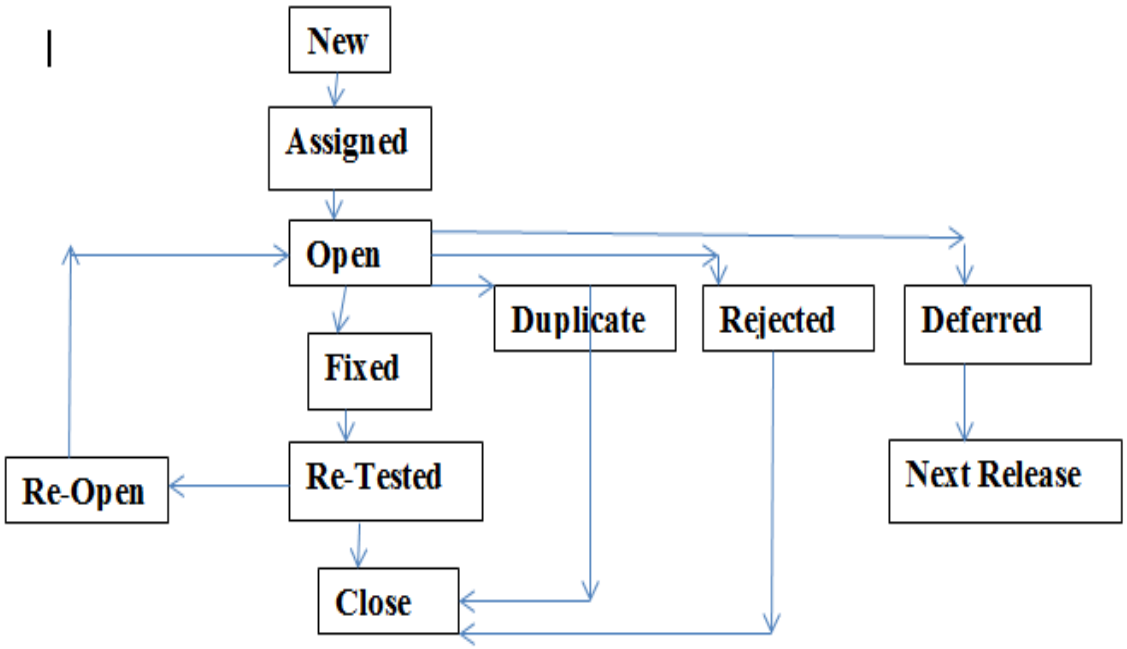
Defect Status : New ,Open ,Fixed ,closed ,Re-Open ,Not a bug, Duplicate ,Need More information, Can't reproduce .
Severity : Importance of defect with respective to functional point of view…means criticalness of defect with respective to application.
Severity classification could be : S1-Urgent ,S2-High ,S3-Medium ,S4-Low , CRITICAL , HIGH ,MEDIUM ,LOW
Priority : Importance of defect with respective to client point of view … means how soon it should be fix.
Priority Classification could be : P1-Urgent ,P2-High ,P3-Medium ,P4-Low, CRITICAL , HIGH ,MEDIUM ,LOW

Sample Examples :
High Priority & High Severity: An error which occurs on the basic functionality of the application and will not allow the user to use the system. (Eg. A site maintaining the student details, on saving record if it, doesn't allow to save the record then this is high priority and high severity bug.)
High Priority & Low Severity: The spelling mistakes that happens on the cover page or heading or title of an application.
High Severity & Low Priority: An error which occurs on the functionality of the application (for which there is no workaround) and will not allow the user to use the system but on click of link which is rarely used by the end user.
Low Priority and Low Severity: Any cosmetic or spelling issues which is within a paragraph or in the report (Not on cover page, heading, title).

Bug or Defect Life Cycle

What is a defect/bug lifecycle in software testing?
A defect lifecycle, or bug lifecycle, is a specific set of states that a software bug goes through from discovery to fixation.

The lifecycle may vary from organization to organization depending on factors like company policy, software developmental model (e.g., Agile, Waterfall, etc.), and project timeline. However, the actual extensive defect lifecycle is as below:



New: When a defect is logged and posted for the first time, its state is set as "new".
Assigned: After the defect has been posted and verified as a bug by the testing

team, it is assigned to the corresponding developer team.
Open: At this stage, the developer team has begun work on fixing the defect.
Fixed: After the necessary code changes are completed by the developers, the defect's state is set to "fixed".
Retest: At this stage, the testing team retests the code given to them by the developers.
Closed: Once the bug has been verified as fixed, the testing team closes the issue.
Reopened: If the bug still exists, its state is set back to Open and the lifecycle restarts.
Duplicate: If the defect is repeated twice or if the defect corresponds to the same concept as the bug, the status is changed to "duplicate."
Rejected: If the developer team feels that the defect is not a genuine defect, they will change the defect's state to "rejected."
Deferred: If the defect is not of a high priority and is expected to get fixed in the next release, then the defect is deferred.

Difference between Defect /Bug/Error/Failure

Defect : Problem which is identified on Developer machine at development phase
Bug : Problem which is identified on Testers machine at testing phase
Error: Problem which is related to coding.
Failure : Problem which is identified By end users at production phase

JIRA Tool

Jira installation Steps [OR] Jira SetUp
1. Open Jira URL -https://id.atlassian.com/login
2. Complete all the setup using gmail id
3. Install **zephyr** tool in JIRA for documenting Test Cases & Doing Test Execution

Steps to Implement Agile Process in JIRA
1. Create new project
2. Create epic
3. Create User Stories
4. Add Created User Stories to Sprint by doing drag and Drop
5. Add Created User Stories to epic
6. Click on Start Sprint and Select Sprint duration
7. Continue in documenting Test Cases & Test Execution

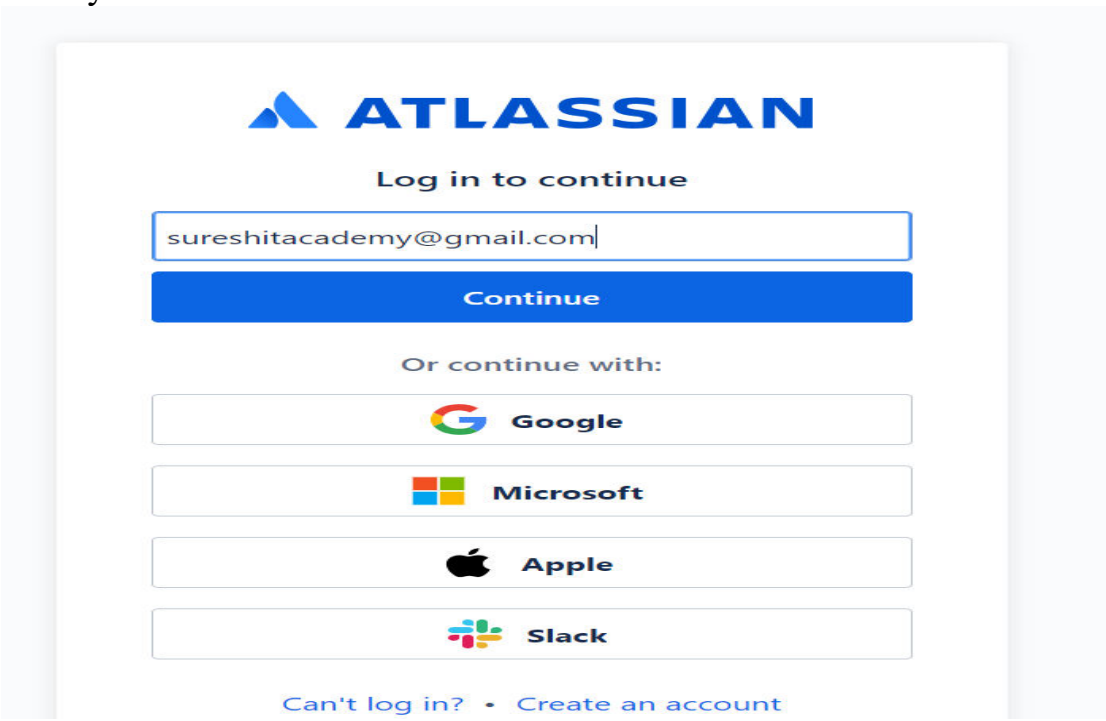
Note: Make sure all the permissions enable in **Zephyr** tool

JIRA Setup

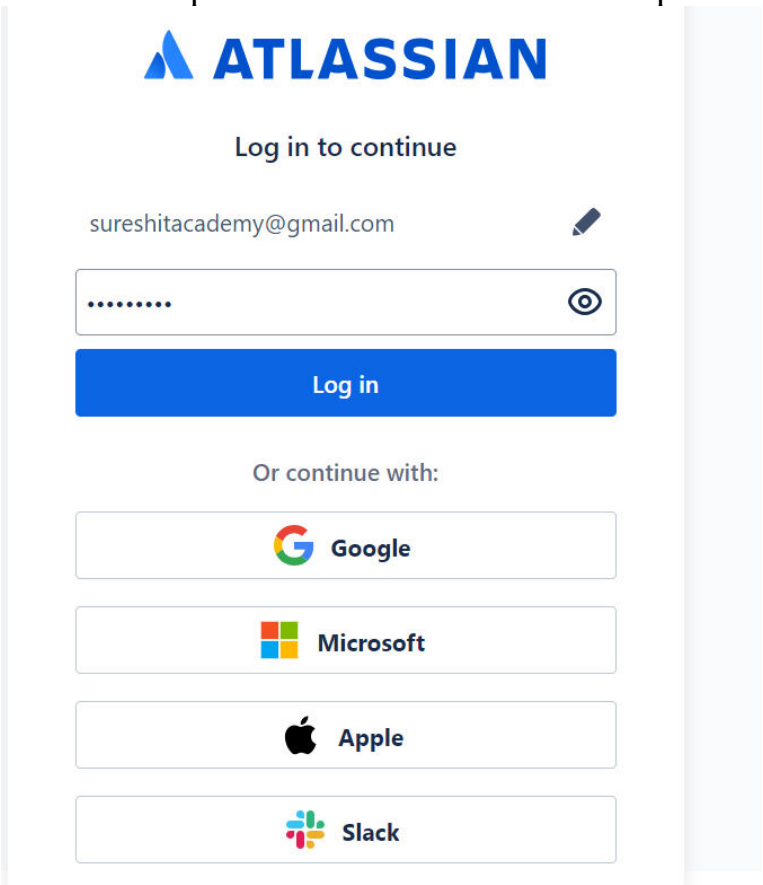
Open Jira URL -**https://id.atlassian.com/login**

Continue with Google option

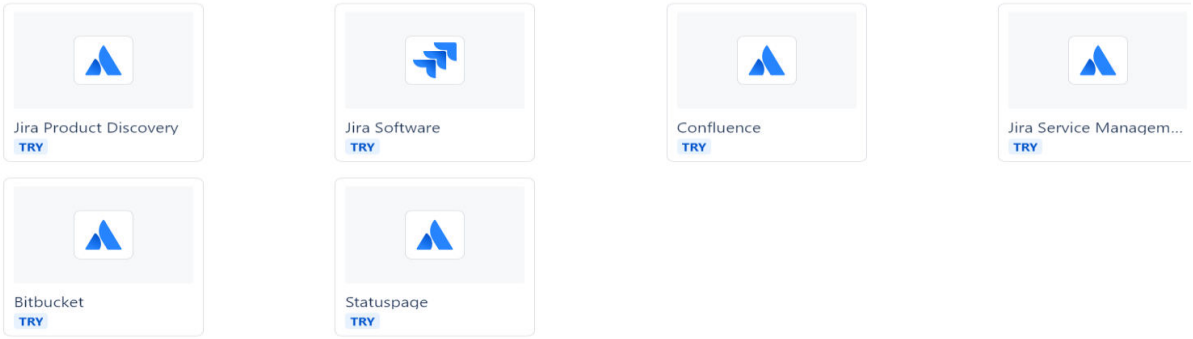
Enter your Gmail id & Click on Continue button



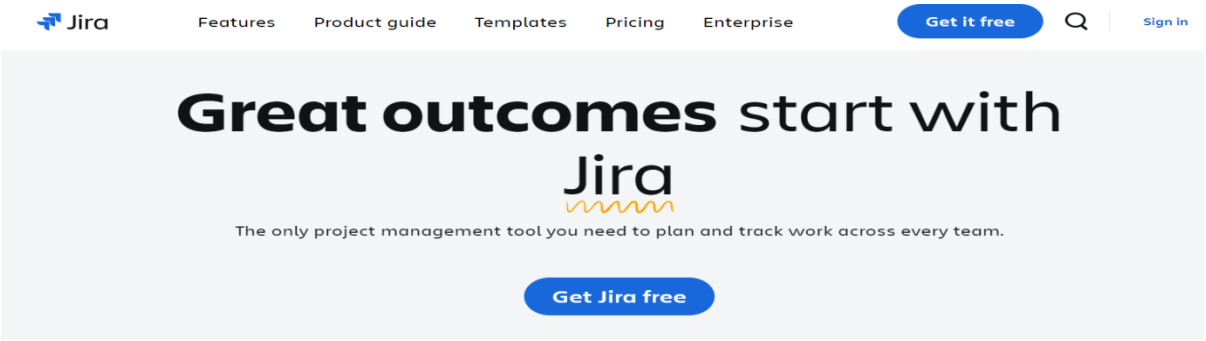
Enter Gmail password and continue with required steps.



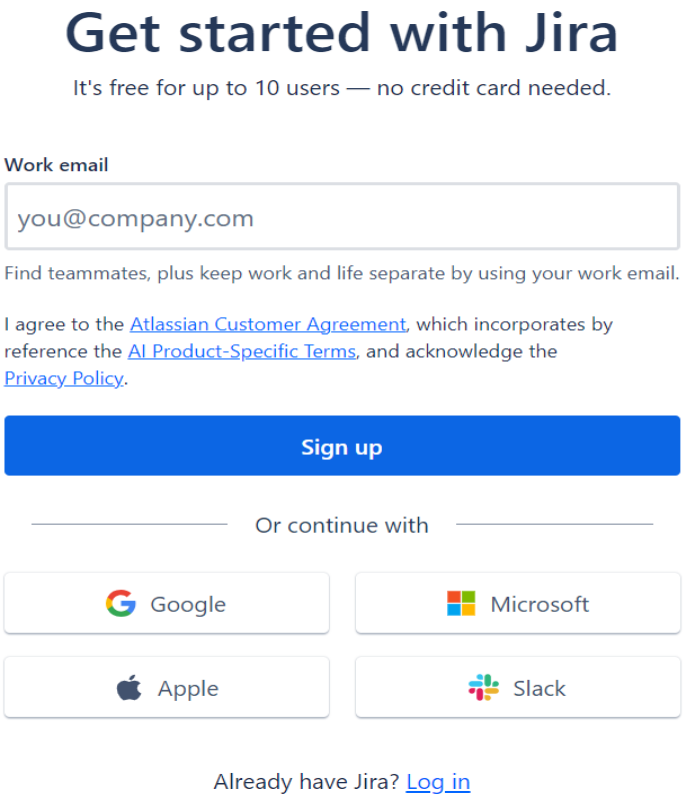
After Login into Jira click on **Jira Software** Option



Click on **Get It Free** button

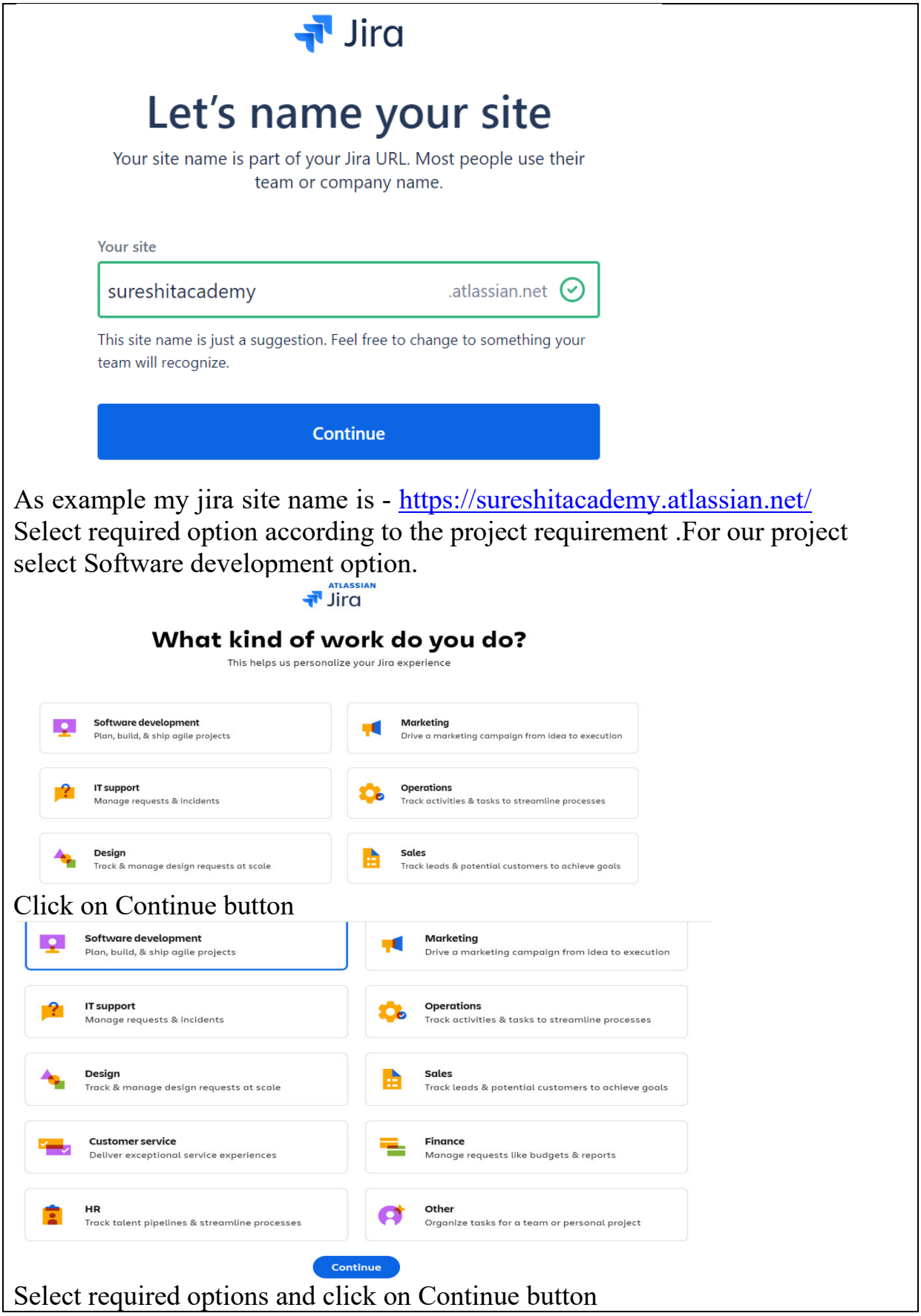


Continue with Google option



Enter any site name (consider as your **Jira website url** for next time login) and click on Continue button

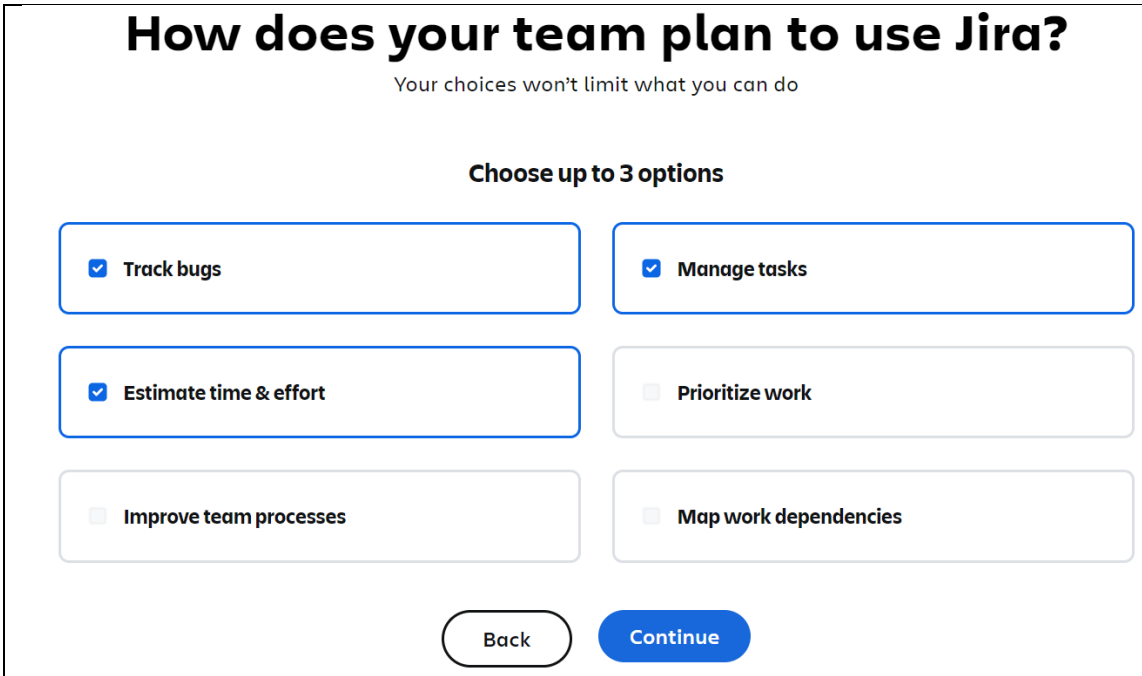
Note : The site you entered will be your JIRA url , use same site name for next time login of jira



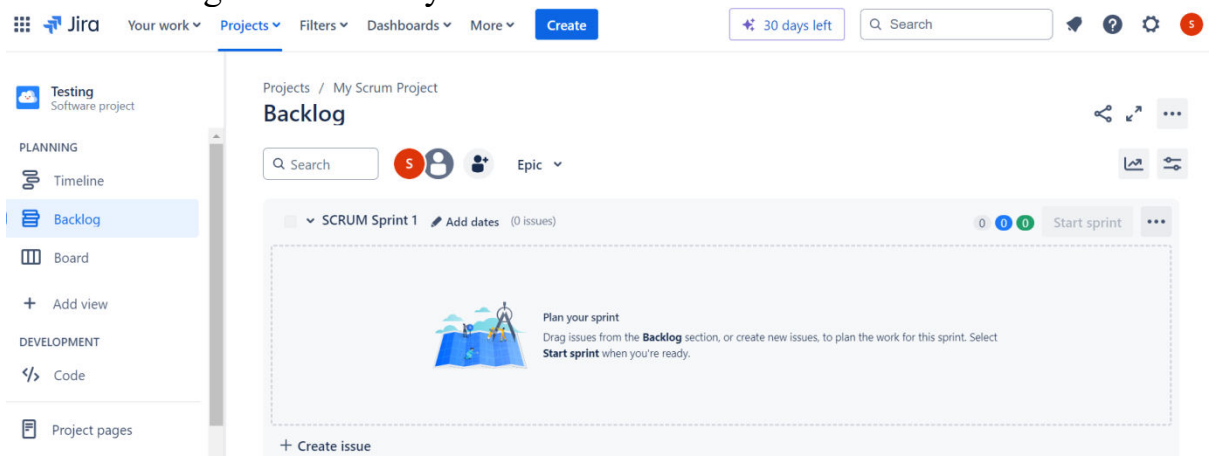
As example my jira site name is - https://sureshitacademy.atlassian.net/
Select required option according to the project requirement .For our project select Software development option.

Click on Continue button

Select required options and click on Continue button

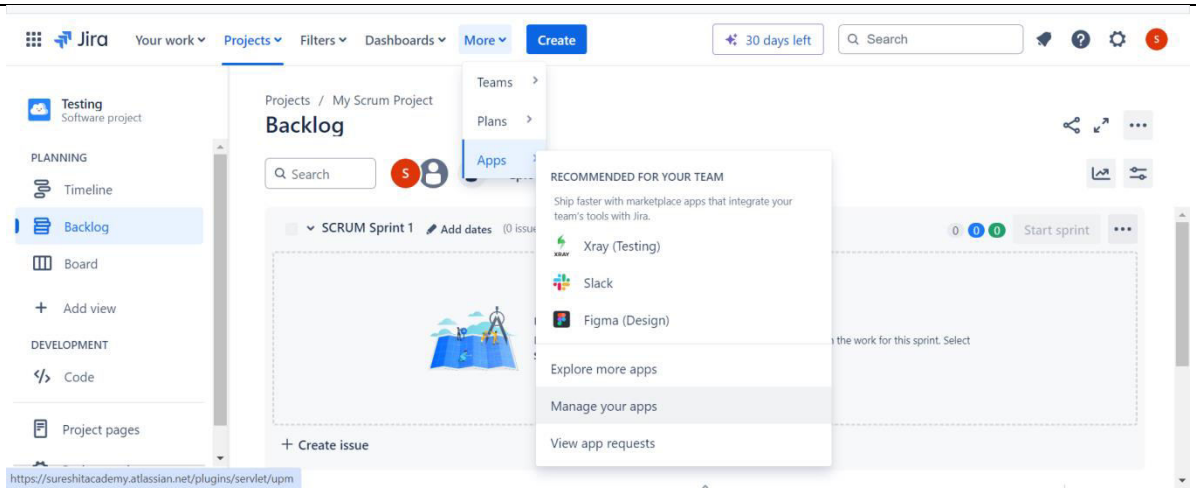


Jira Home Page will be ready

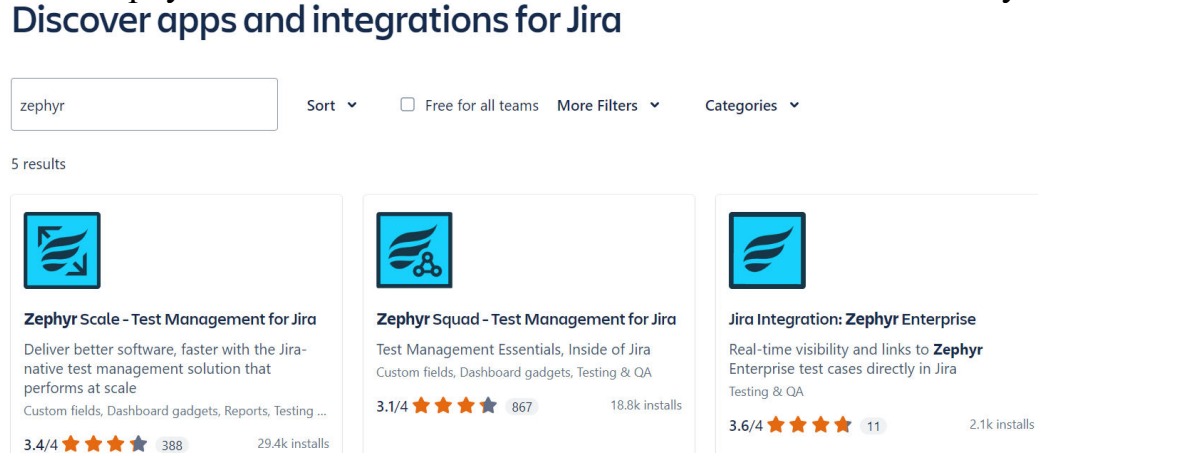


Provide project name and click on **Create project** button
Click on **Done** button

Now install **Zephyr** Scale Tool in JIRA to write and execute test cases.
Steps to install Zephyr Scale
Navigate Apps main menu and click on Explore more Apps option

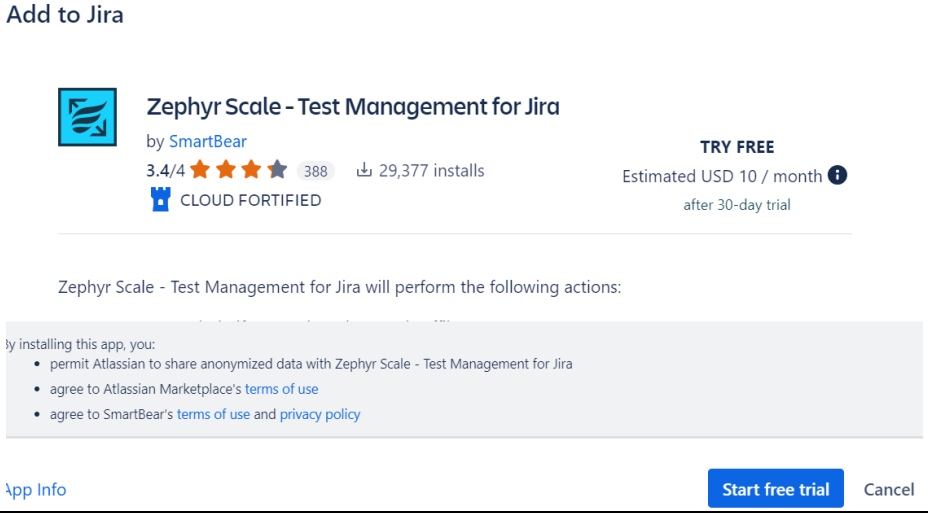


Enter Zephyr scale in textbox and click on **ENTER** button from keyboard



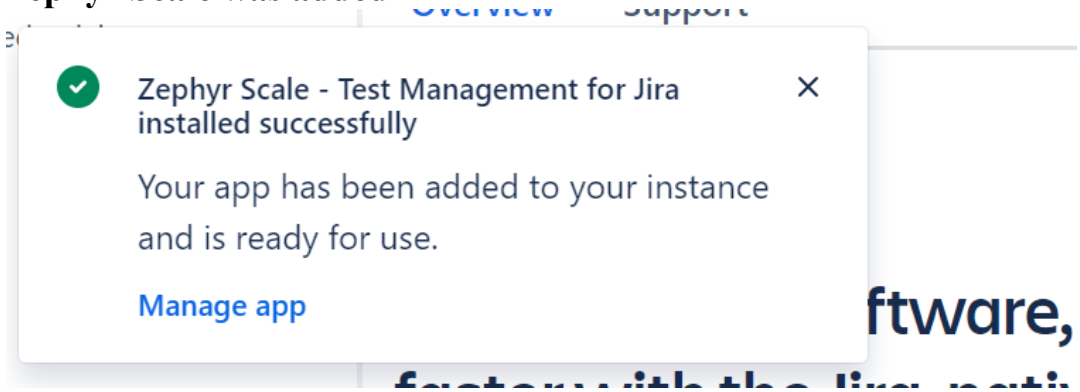
Click on **Zephyr Scale – Test management for Jira** Option

Click on **Try it Free** button

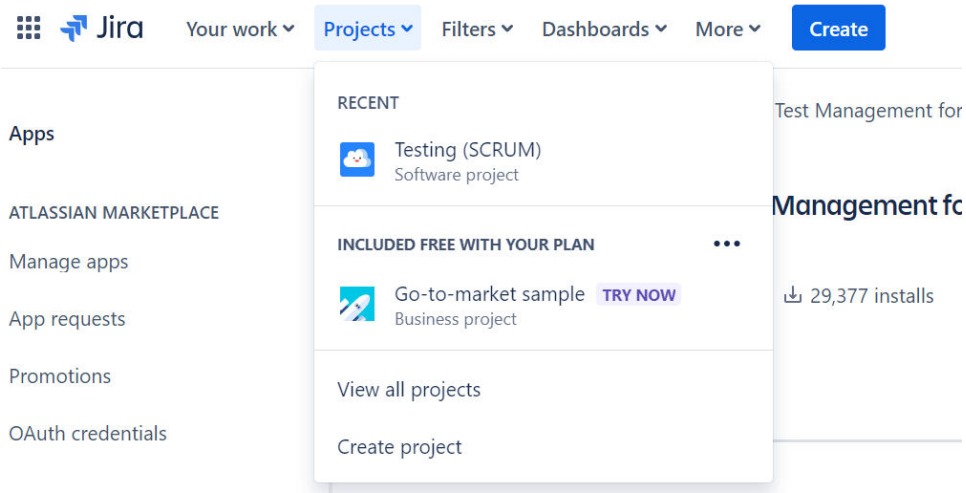


Click on **start free trial** button

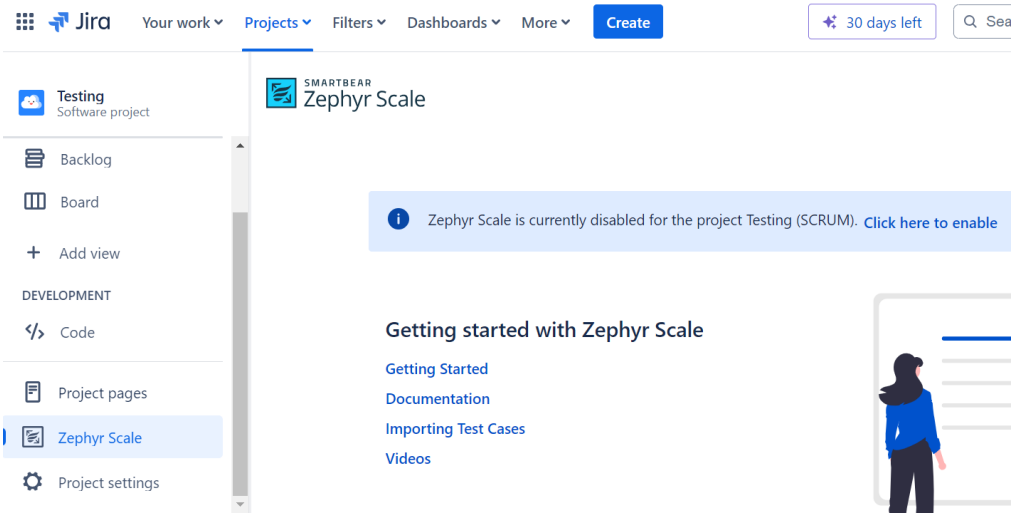
After completion of Zephyr scale installation you will see message - **Success & Zephyr Scale was added**



Navigate to Projects Main menu and click on **Your Created project**

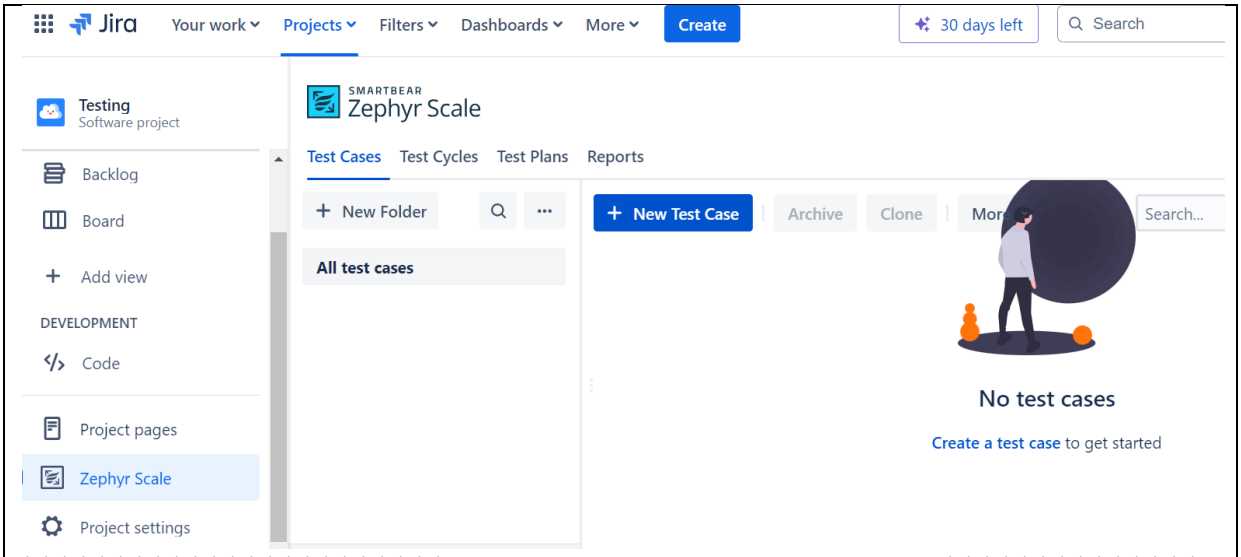


Click on Zephyr Scale option in left Side panel , after that click on **Click here to enable** option



Note : based on your internet speed this page will take some time to Load , if not loading click on Browser refresh button

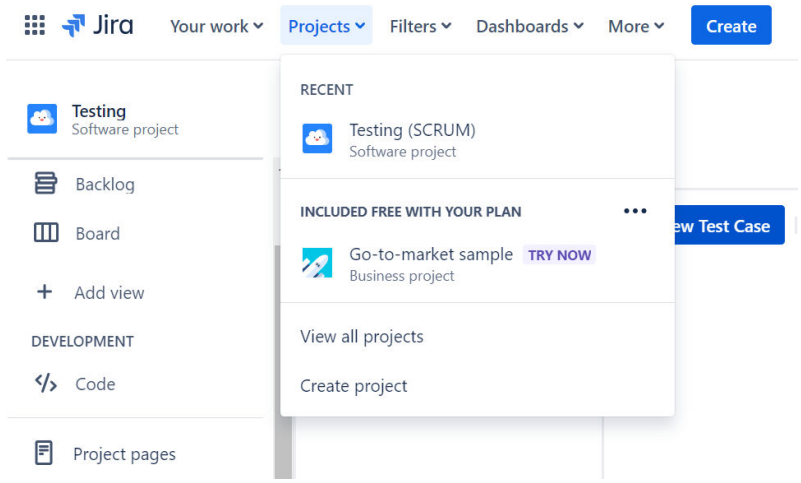
Now you need enable all the option to continue in writing Test Cases & Doing execution



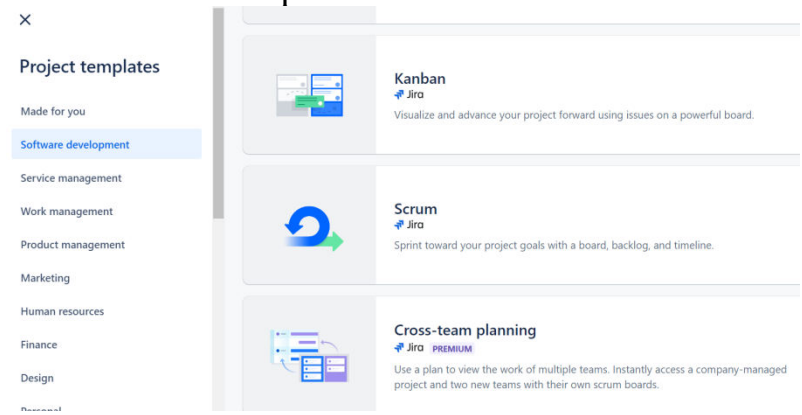
**********By this Jira Setup is completed**********

Agile Process in JIRA Tool

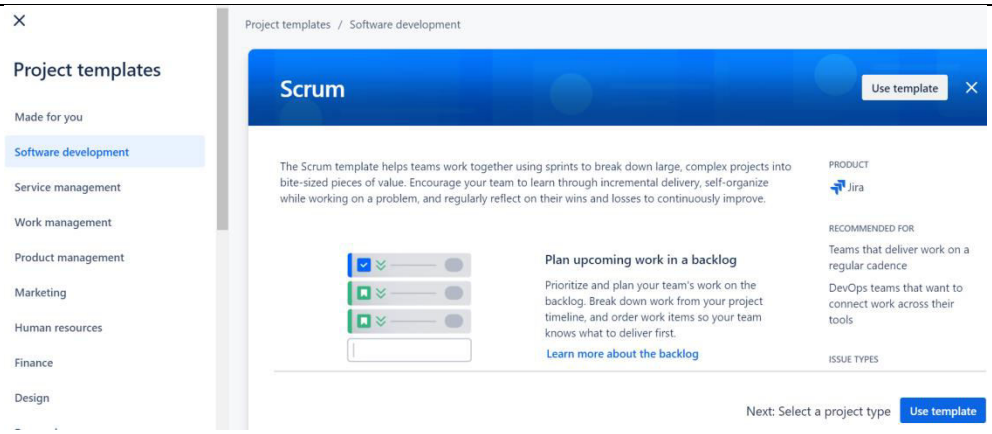
Create new project



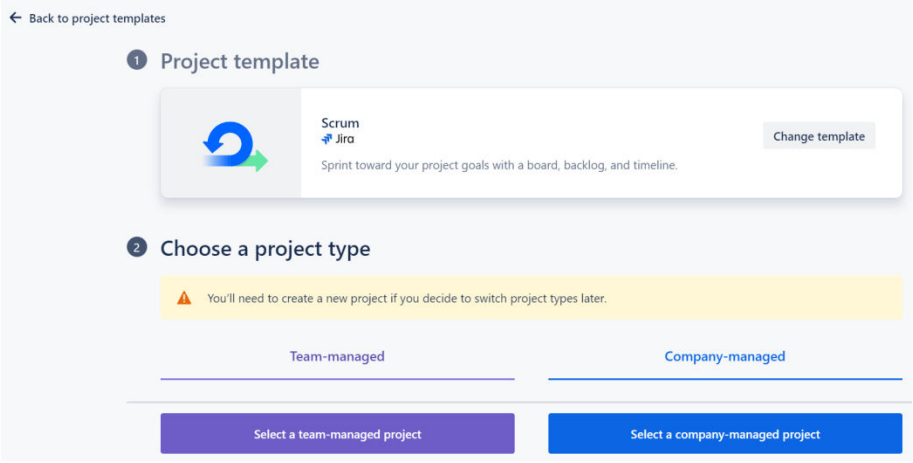
Click on **Scrum** Option



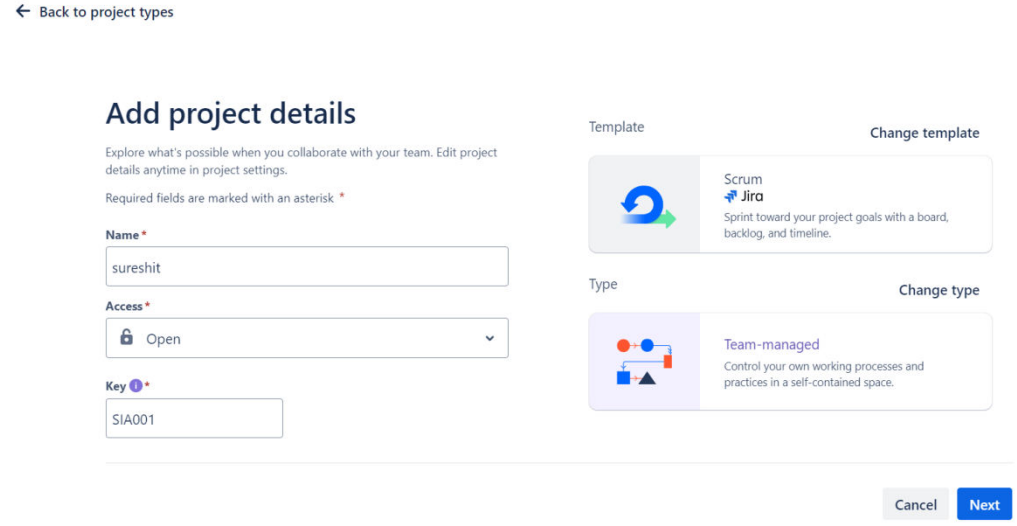
Click on **Use Template** Option



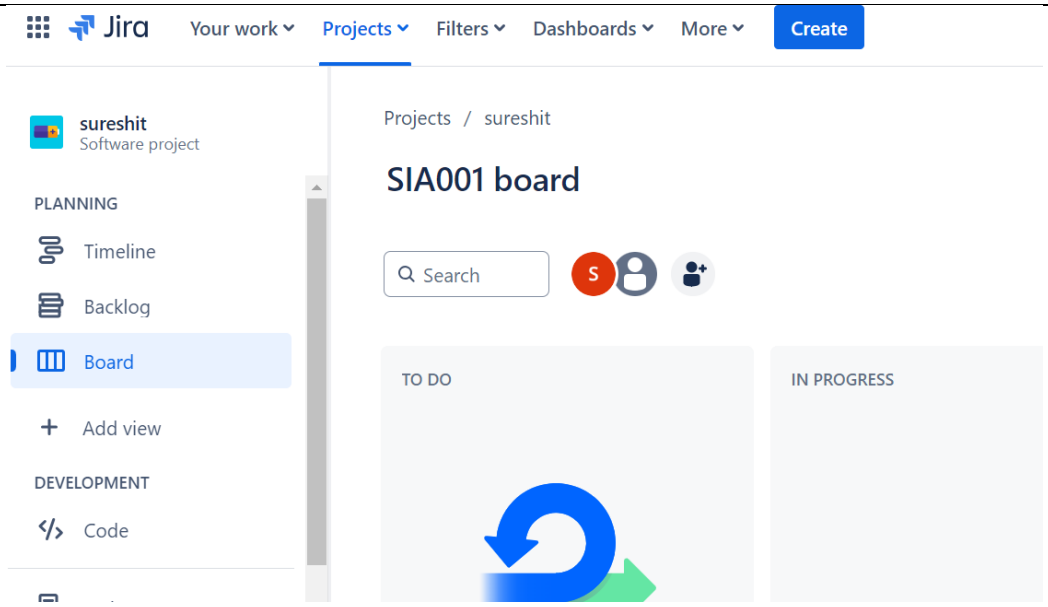
Click on Select **Team Managed** project



Provide Project Name and click on **Create project** button

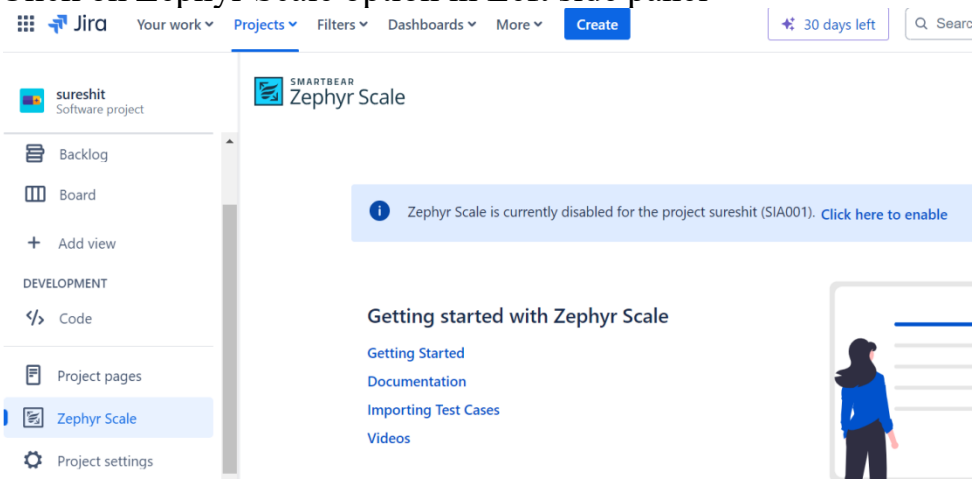


Project will be created

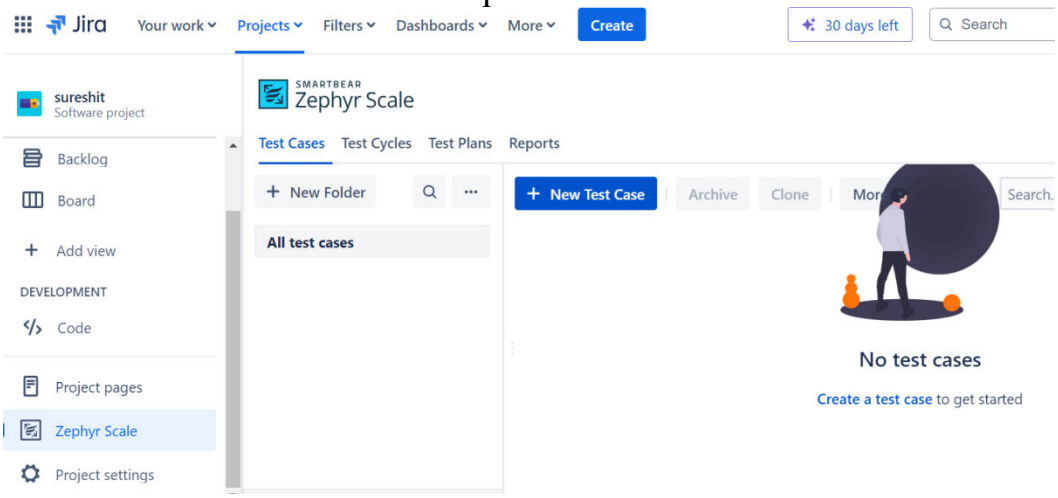


Steps to Create Test Cases :

Click on Zephyr Scale option in Left side panel

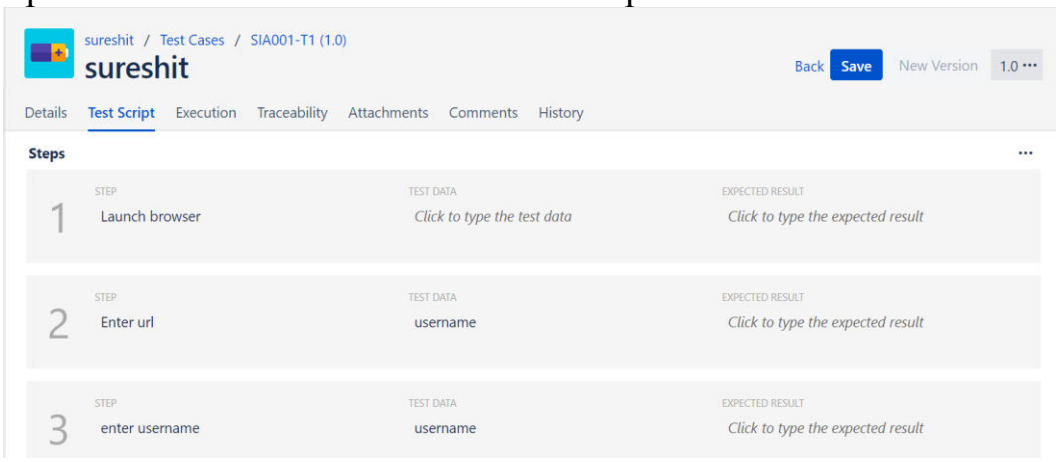


Click on **Click here to enable** option

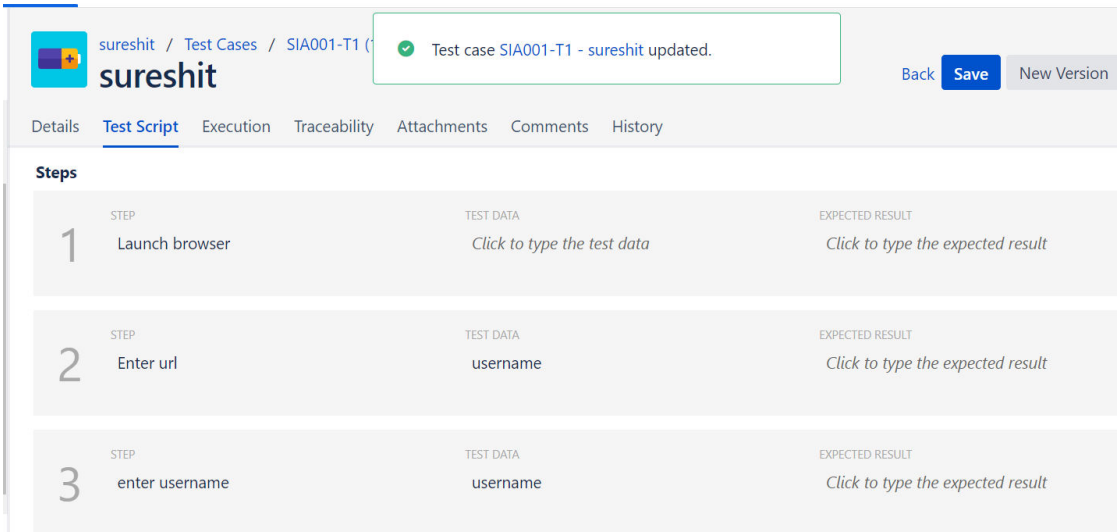


Click on New Test Case button & provide Name , Owner details and then click on create button

Open the Test case and Click on Test Script Tab



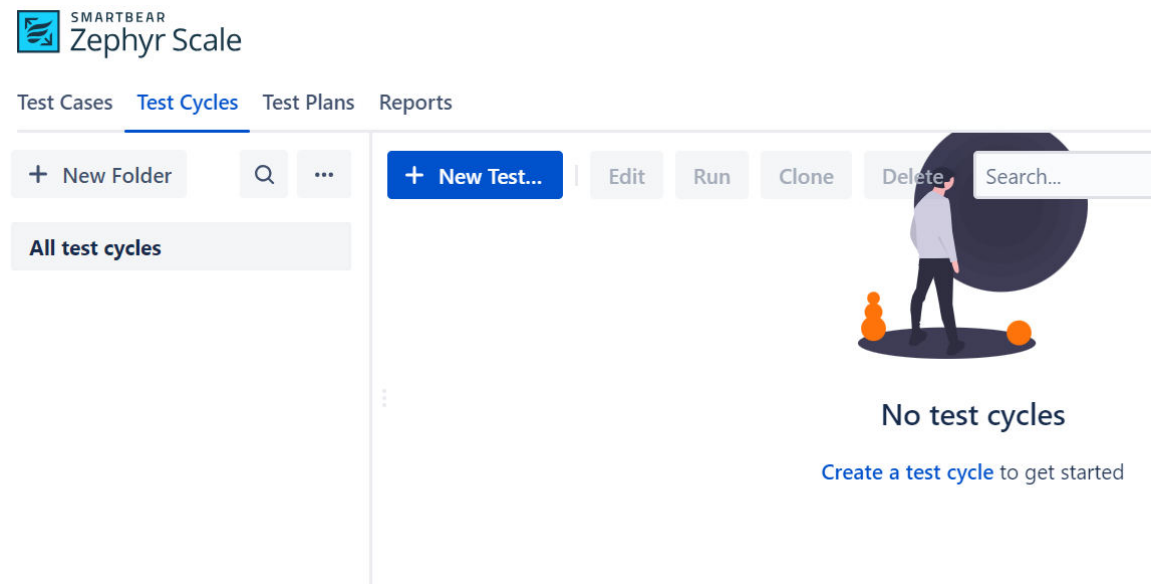
Now document all the Test Steps and click on Save button.



Repeat the same steps to write multiple test cases.

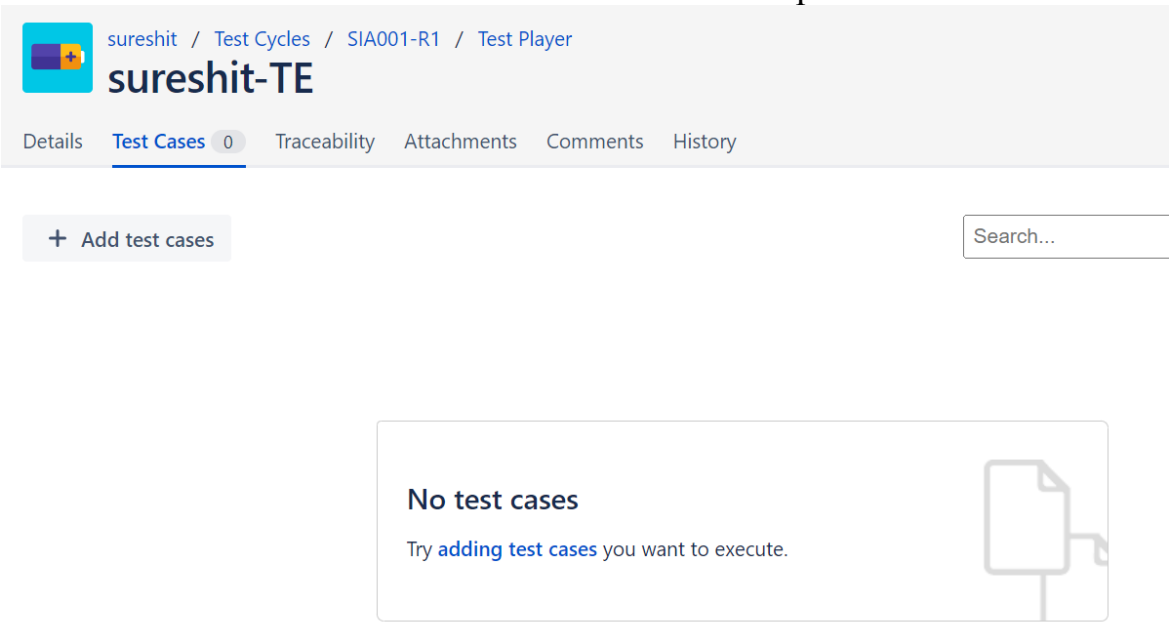
Test Case Execution :

For Doing Execution Create **Test Cycle** by clicking on Test Cycles Tab

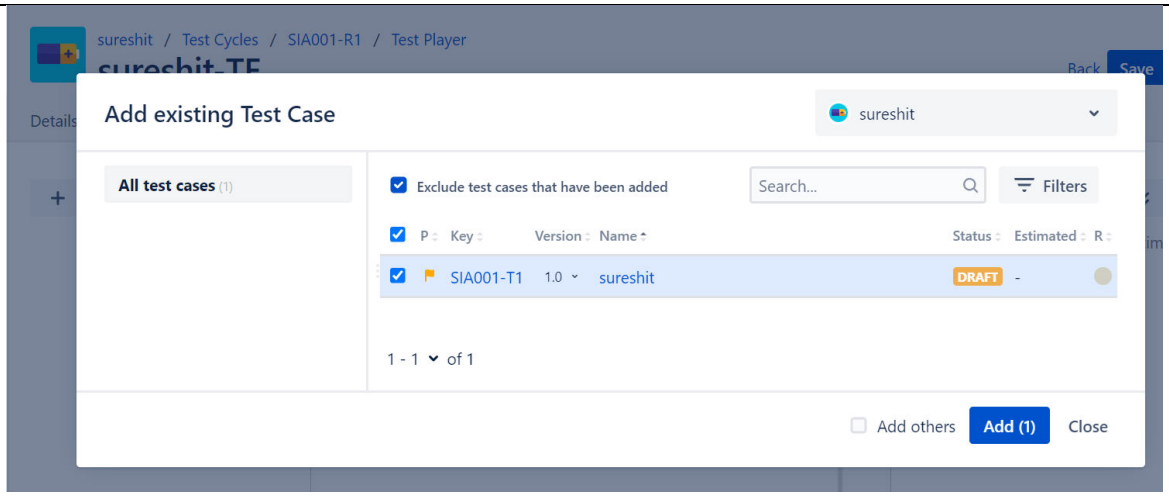


Provide name, owner , start date and end date details and then click on create button

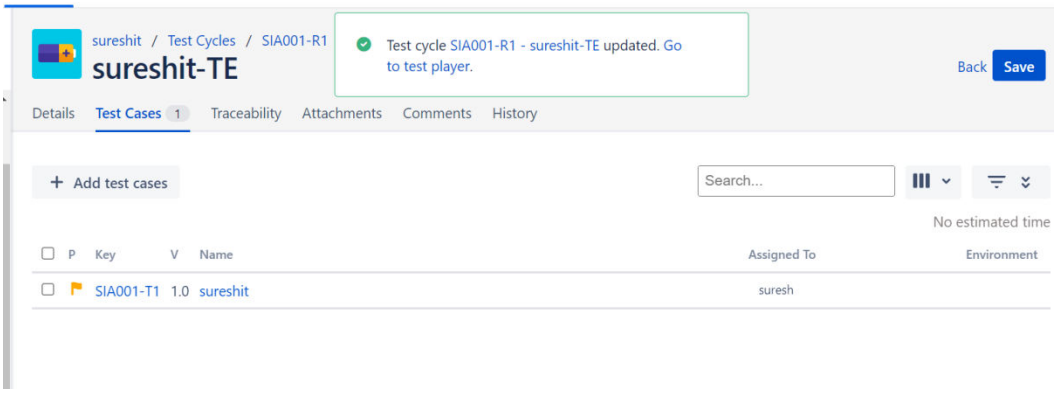
Click on Test Cases tab & click on Add Test Cases Option



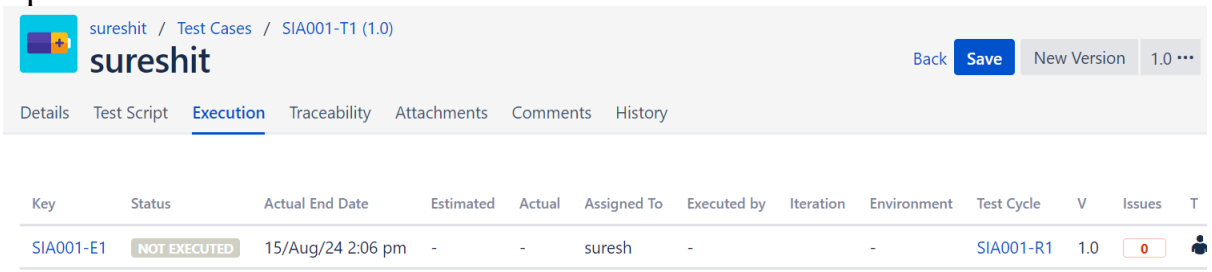
Select the test cases to execute and click on Add button



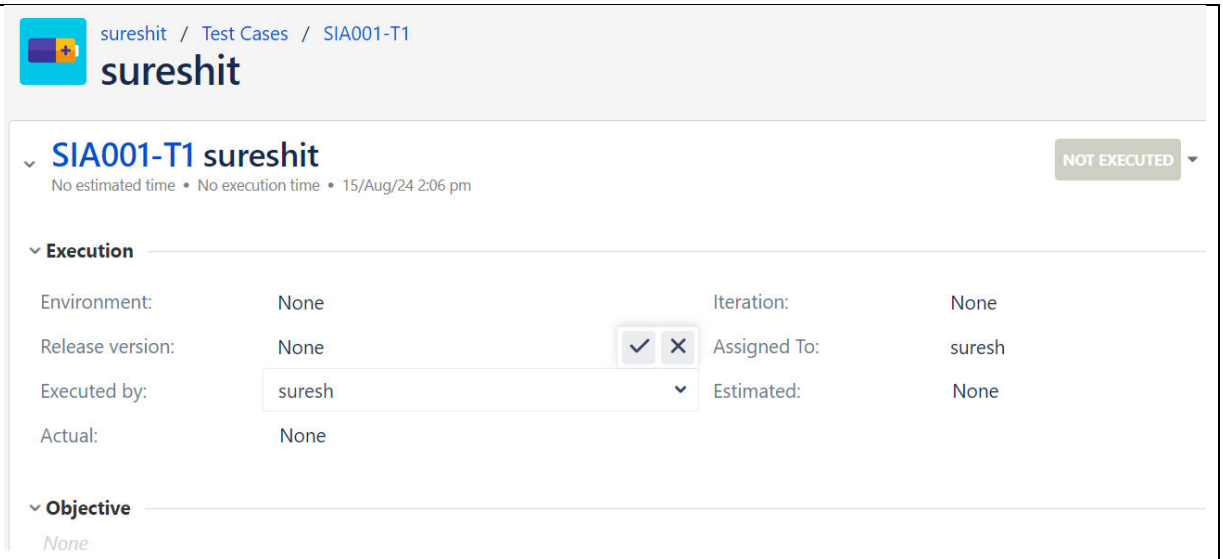
Click on Save button



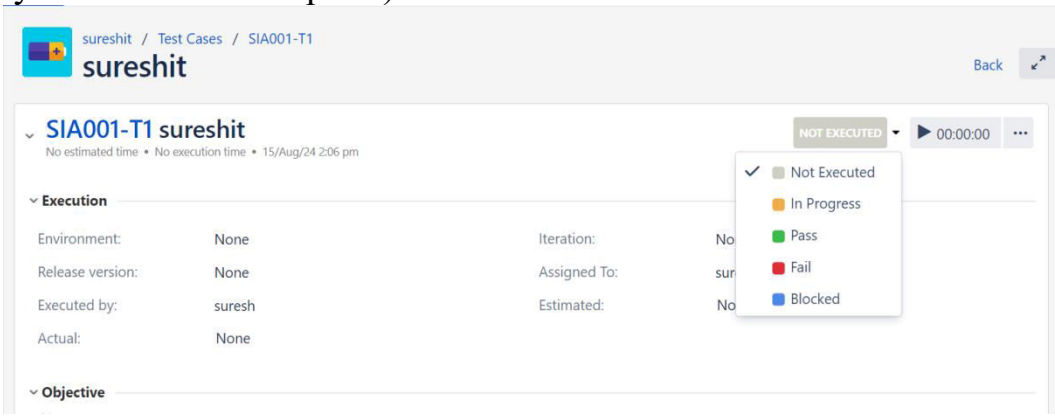
Open individual test cases to Execute and click on Execution Tab



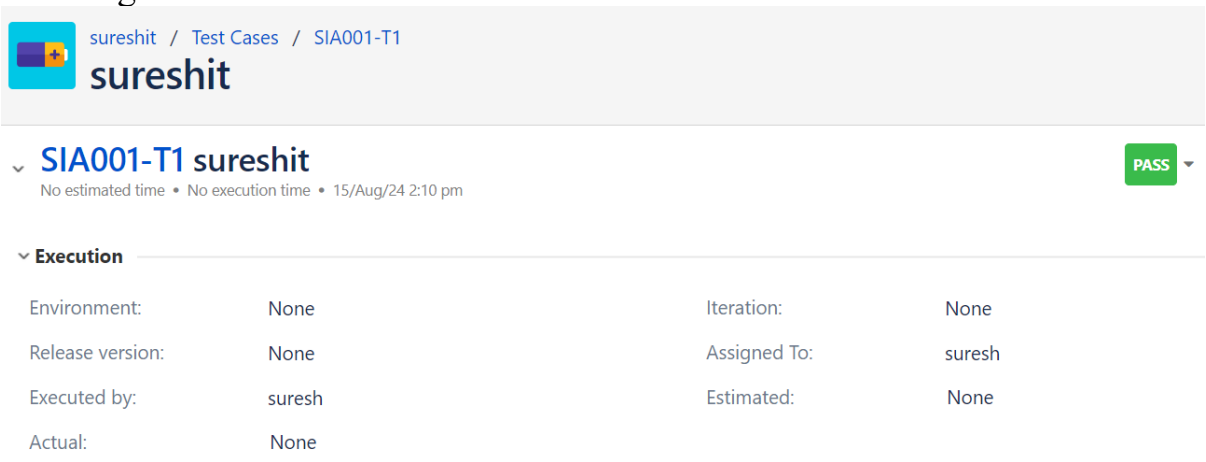
Click on Key of each and every Test Case and provide executed by and expected Results



Now Top of the test case select Execution status
Select required status and click on Save (click on doted symbol to see save option)



Selecting the test case status as PASS



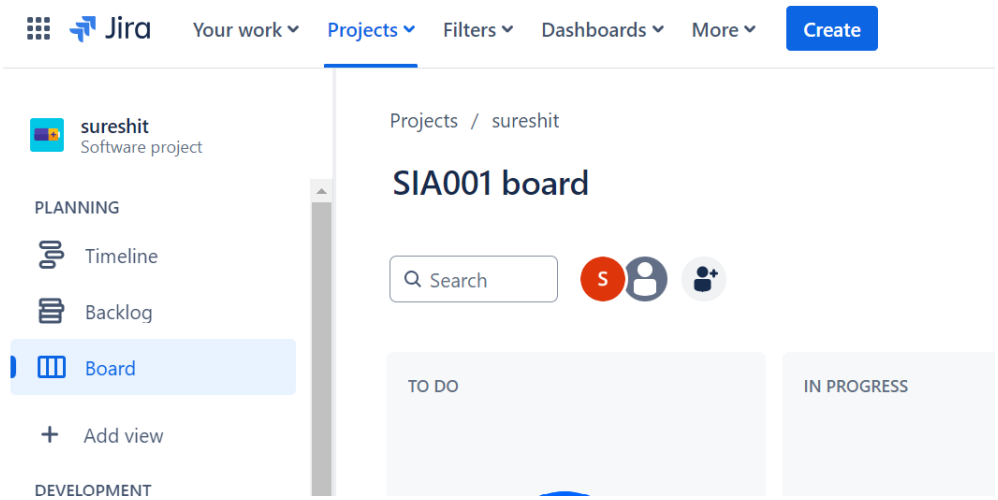
Repeat the same steps to execution all the test cases. In case any test cases failed we can report the bug.

Bug Reporting in JIRA

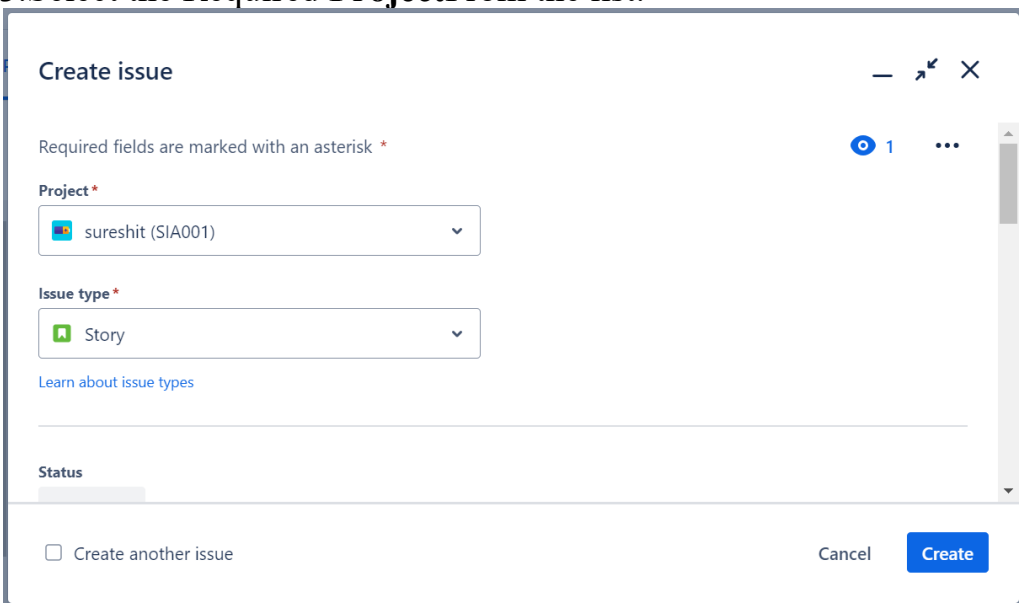
Steps to Report Bugs in JIRA Tool

1.Login to JIRA tool

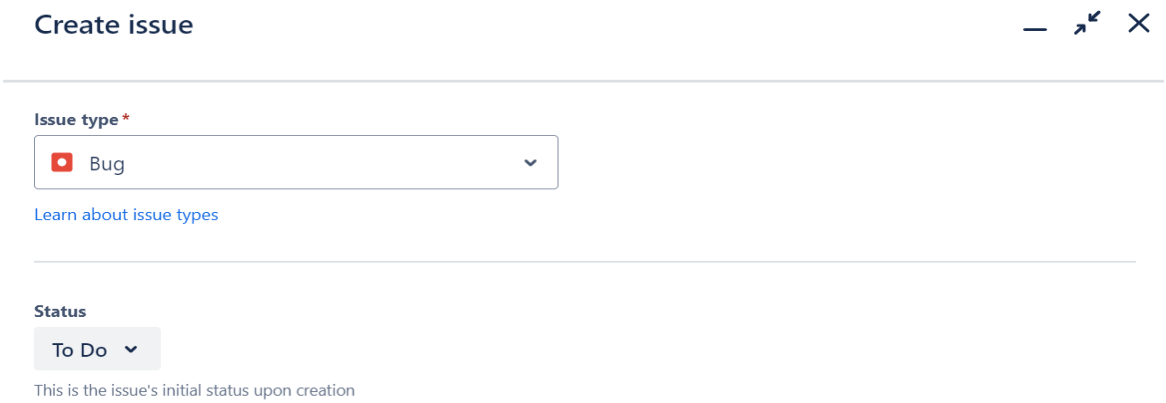
2. Click on **Create** button



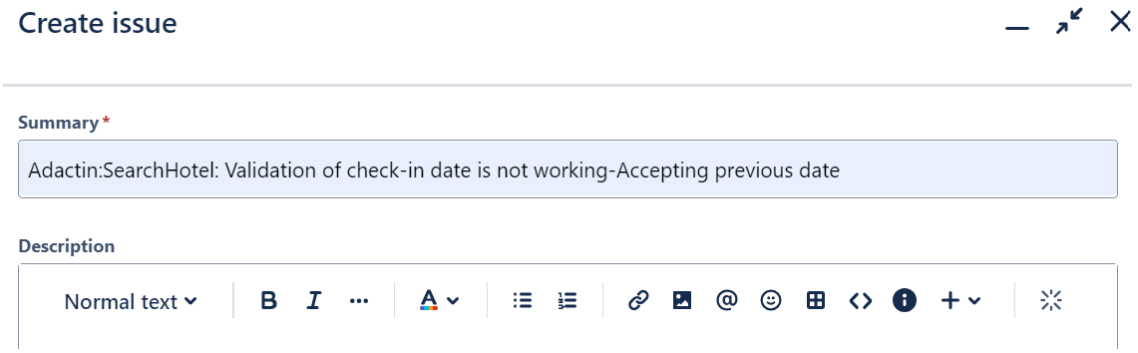
3.Select the Required **Project**From the list.



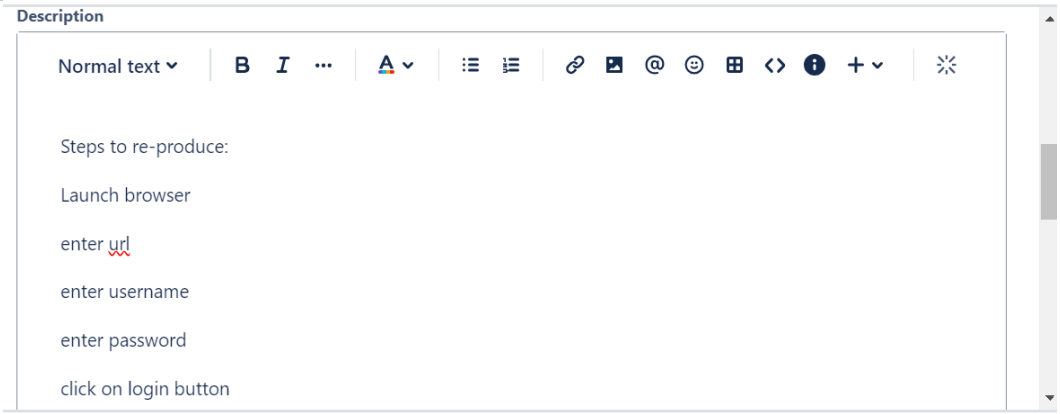
4.Select the **Bug** option from Issue Type dropdown and status as To-Do option



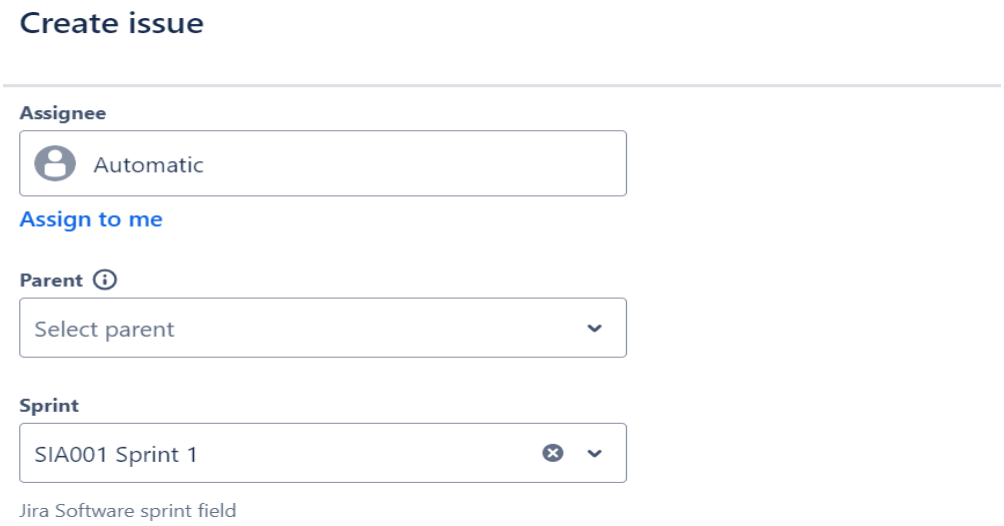
5.Provide **Summary** of the bug



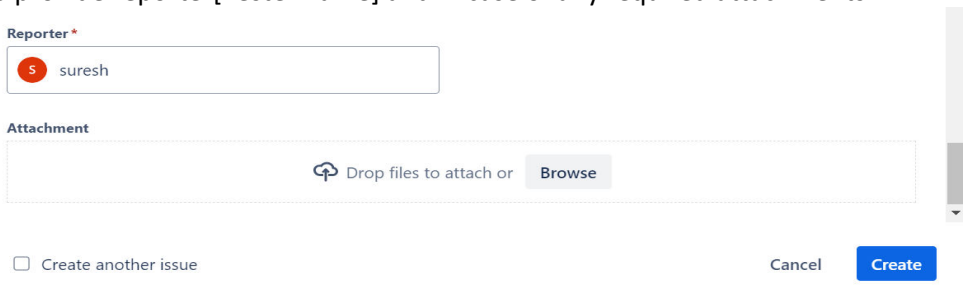
6.Provide the **Bug Description** which includes – Steps to re-produce , Expected Result and Actual Result



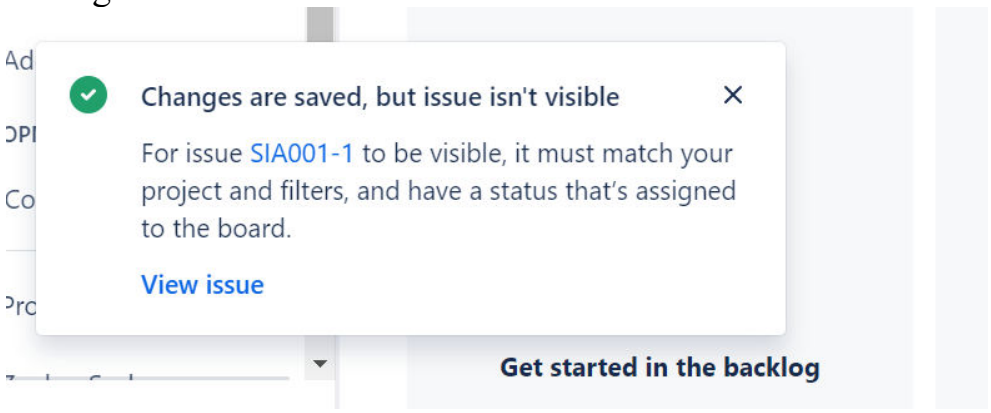
7.Provide Assignee[developer] name & select working sprint



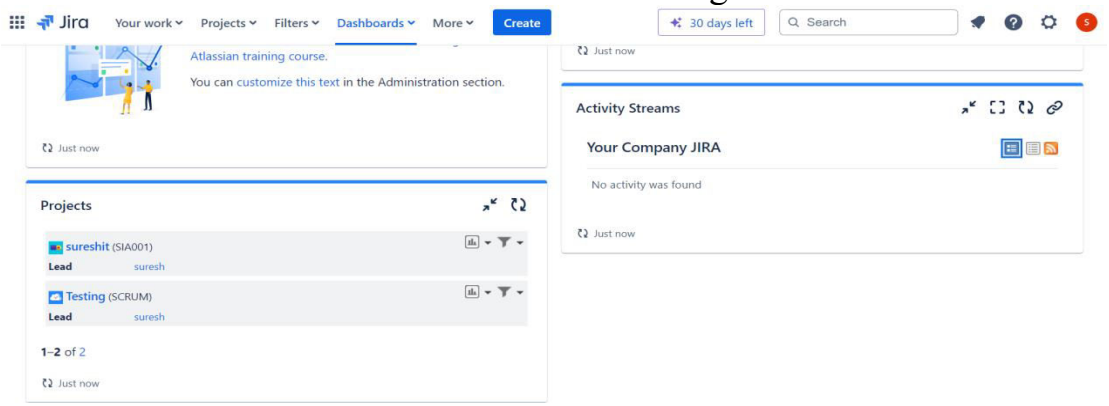
8.provide reporter[Tester name] and in case of any required attachments



9.Click on Create button
10.Bug will be created.



11.Refer Default dashboard to check created bugs



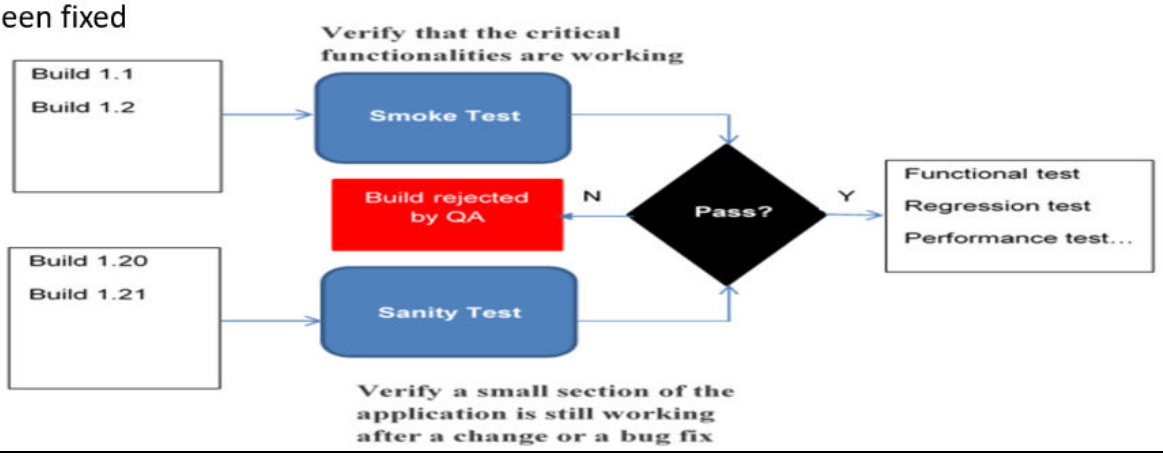
Types of Testing

- **Smoke Testing** or **BVT-Build Verification Testing** or **TAT – Testers Acceptance Testing**

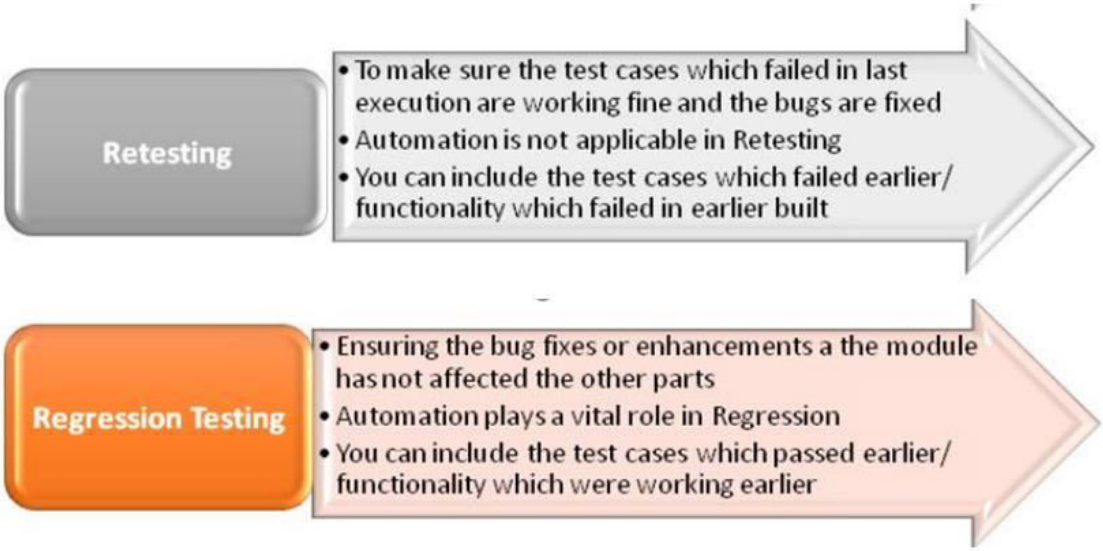
It is first level of testing on any newly released build to check main functionality of the application.

- **Sanity Testing :**

Sanity Testing is done during the release phase to check for the main functionalities of the application.Sanity Testing is done to check the new functionality/bugs have been fixed



- **Re-Testing :** Testing defects were fixed or not in the current build
- **Regression Testing:** To check existing functionality is un affected whenever the new change is added



- **Static Testing:** Testing an application with out performing any action.

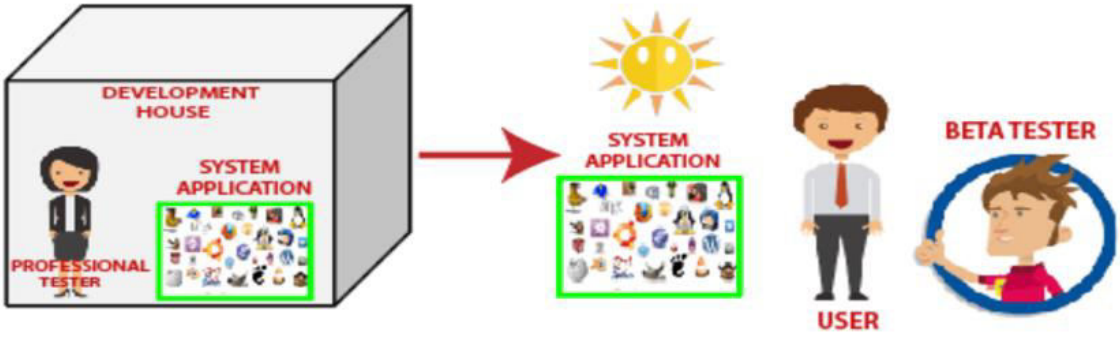
Eg: GUI Testing , colors ,spelling ,alignment … etc.

- **Dynamic Testing:** Testing an application by performing required actions.

Eg: Functionality Testing ,Textbox, button … etc

- **Ad-hoc Testing:** Testing the application with out any proper planning

- **Alpha Testing :**Final Testing on the application doing with in the development company
- **Beta Testing:** Testing doing in customer environment ,This testing will be done by the customer or third party test engineers.



BETA TESTING OF THE PRODUCT IN REAL WORLD ENVIRONMENT

Functionality Testing - Test for – all the links in web pages, database connection, forms used for submitting or getting information from the user in the web pages, Cookie testing etc.

Usability testing - Usability testing is nothing but the User-friendliness check. In Usability testing, the application flow is tested so that a new user can understand the application easily. Basically, system navigation is checked in Usability testing.

Compatibility testing - Compatibility testing is used to determine if your software is compatible with other elements of a system with which it should operate, e.g. Browsers, Operating Systems, or hardware.

Database Testing – Database connection and user entered Data in application is saving into respective database tables



Interface testing-Three areas to be tested here are - Application, Web and Database Server

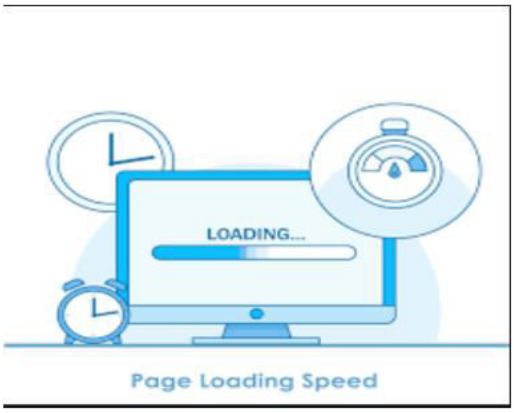
01. Application: Test requests are sent correctly to the Database and output at the client side is displayed correctly.

02. Web Server: Test Web server is handling all application requests without any service denial.

03. Database Server: Make sure queries sent to the database give expected results.

Performance testing – Testing page ,data, images load time

Security testing -Security Testing involves the test to identify any flaws and gaps from a security point of view.



Test Closure :

Test Closure is a document that is prepared prior to formally completing the testing process. This memo contains a report of test cases executed,passed,failed and number of defects found,fixed,re-tested,closed.

We will provide the path of all the required documents to client like testscenarios,testcases,testexecution and bugreporting,based on this report it will decide to stop or continue the testing.

When to stop Testing?

This can be difficult to determine. Most modern software applications are so complex, and run in such an interdependent environment, that complete testing can never be done. Common factors in deciding when to stop are:

- Deadlines (release deadlines, testing deadlines, etc.)
- Test cases completed with certain percentage passed
- Test budget depleted
- Coverage of code/functionality/requirements reaches a specified point
- Bug rate falls below a certain level
- Alpha & Beta testing period ends

Banking - Domain Testing

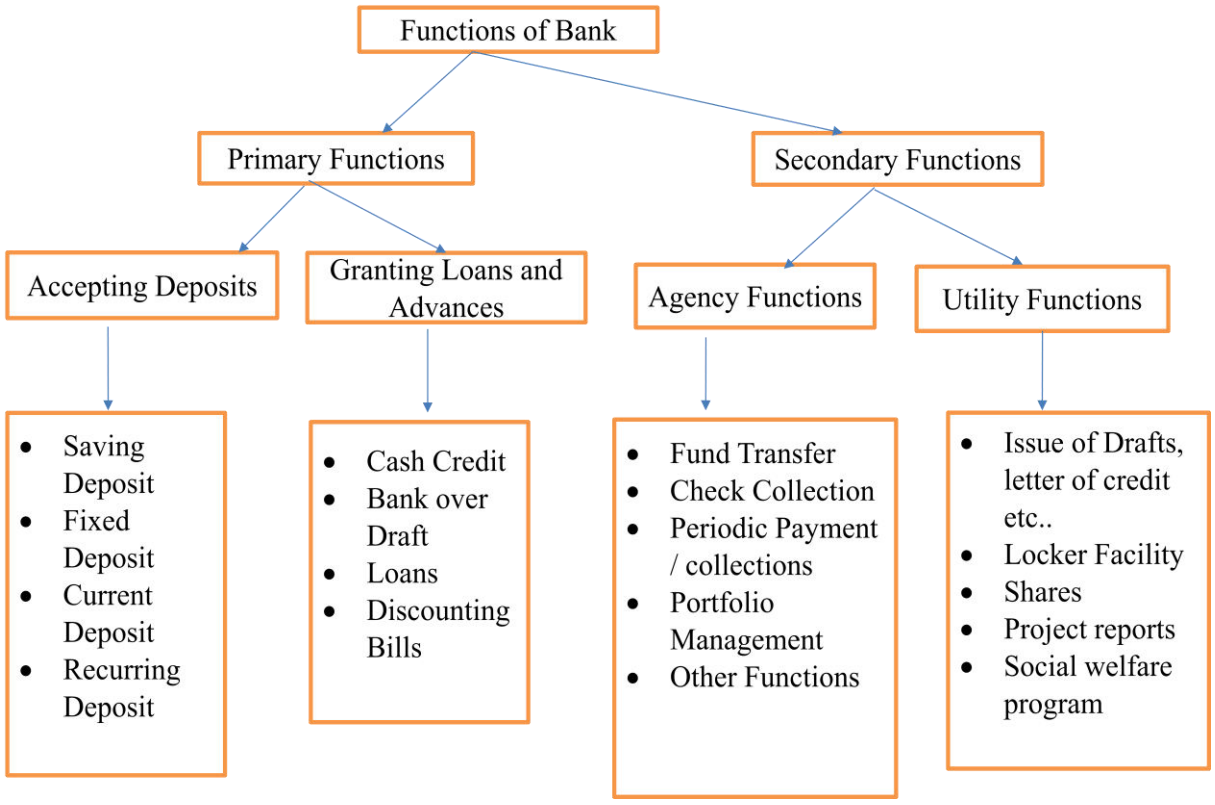
What is mean by Domain in Software Industry ? : A domain is a field of study that defines a set of common requirements, terminology, and functionality for any software program

Types of Domains:

- Banking
- Insurance
- Finance
- HRMS
- ERP
- Telecom
- HealthCare
- Retail Market
- E- Commerce
- Games
- Travel ….etc

Definition of Banking?

Banking can be defined as the business activity of accepting and safeguarding money owned by other individuals and entities, and then lending out this money in order to earn a profit.



Types of Banking Software Applications

- Internet banking system
- ATM banking (Automated Teller Machine)
- Loan management system
- Core banking system
- Mobile Banking
- Credit management system
- Investment management system
- Stock market management system
- Financial management system

Internet banking system: Internet banking allows the customers and financial institution to conduct final transaction using banks or financial institute website.
ATM banking (Automated Teller Machine) : It is an electronic banking outlet, which allows customers to complete basic transaction.
Loan management system: The database collects all the information and keeps the track about the customers who borrows the money.
Core banking system: Core banking is a service provided by a networked bank branches. With this, customer can withdraw money from any branch.
Mobile Banking : Mobile Banking is a service provided through Mobile App
Credit management system: Credit management system is a system for handling credit accounts, assessing risks and determining how much credit to offer to the customer.
Investment management system : It is a process of managing money, including investments, banking, budgeting and taxes.
Stock market management system : The stock market management is a system that manages financial portfolio like securities and bonds.
Financial management system : Financial management system is used to govern and keep a record of its income, expense and assets and to keep the accountability of its profit.

Characteristics of banking Application

- Simple and secure sign-in
- Multi-tier functionality to support thousands of concurrent user sessions
- Banking application should integrate with other numerous applications like trading ,bill pay utility ,credit card etc…
- Real-Time and Batch processing
- Bank Applications should handle Complex Business workflows
- The high rate of Transactions per seconds
- Robust Reporting section to keep track of day to day transactions
- Strong Auditing to troubleshoot customer issues
- Massive storage system
- Support multiple service sectors (loan , retail banking etc.)

- Disaster / Recovery Management.
- Chabot for customer support
- Should support fast and Secure Transactions
- Should support for multiple Browsers and Platforms
- Support users on various payment systems(visa , MasterCard ,Amex)

Common Terms of Banking :

1. Account Types

- **Savings Account:** A bank account that earns interest over time and is generally used for saving money.
- **Current Account:** A type of account for businesses or individuals that allows for numerous transactions daily without earning interest.
- **Fixed Deposit (FD):** An investment account where money is deposited for a fixed period at a fixed interest rate.
- **Recurring Deposit (RD):** A type of term deposit where the customer deposits a fixed amount regularly and earns interest on it.

2. Transactions

- **Deposit:** The act of adding money to a bank account.
- **Withdrawal:** The act of taking money out of a bank account.
- **Transfer:** Moving money from one account to another, either within the same bank or to a different bank.
- **ACH (Automated Clearing House):** An electronic system for processing transactions such as direct deposits, bill payments, and other electronic funds transfers.
- **NEFT (National Electronic Funds Transfer):** A nation-wide payment system facilitating one-to-one funds transfer in India.
- **RTGS (Real Time Gross Settlement):** A system where funds transfer occurs in real-time and on a gross basis.

3. Banking Operations

- **Core Banking System (CBS):** The back-end system that processes banking transactions across various branches of a bank in real-time.
- **Clearing:** The process of settling transactions between banks, especially in the context of checks.
- **Reconciliation:** The process of matching and comparing transactions in the banking records with those of the customer's records.
- **KYC (Know Your Customer):** A process where a bank verifies the identity of its customers, typically by collecting personal information and documents.
- **AML (Anti-Money Laundering):** Procedures and regulations designed to prevent money laundering and other financial crimes.

4. Cards and Payments

- **Credit Card:** A card issued by a financial institution that allows the holder to borrow funds for purchases, repayable with interest.
- **Debit Card:** A card issued by a bank allowing the holder to transfer money electronically from their bank account when making a purchase.
- **EMV (Europay, MasterCard, Visa):** A global standard for credit and debit cards equipped with computer chips to authenticate transactions.
- **SWIFT (Society for Worldwide Interbank Financial Telecommunication):** A network that enables secure and standardized financial transactions between banks worldwide.
- **Payment Gateway:** A service that authorizes credit card or direct payments for online purchases.

5. Loans and Mortgages

- **Home Loan:** A loan taken out by an individual to purchase a house or property.
- **Personal Loan:** A loan provided for personal use, such as debt consolidation, vacations, or medical expenses, typically unsecured.
- **Auto Loan:** A loan taken out to purchase a vehicle, usually secured against the vehicle being purchased.
- **Mortgage:** A loan specifically for purchasing real estate, where the property itself serves as collateral.
- **EMI (Equated Monthly Installment):** A fixed payment amount made by a borrower to a lender at a specified date each calendar month.

6. Interest and Rates

- **Interest Rate:** The percentage charged by a lender to a borrower for the use of assets.
- **Fixed Interest Rate:** An interest rate that remains constant throughout the term of the loan or deposit.
- **Floating Interest Rate:** An interest rate that can change periodically based on the prevailing market conditions.
- **APY (Annual Percentage Yield):** The rate of return earned on a deposit over a year, considering the effect of compounding interest.

7. Compliance and Regulation

- **PCI DSS (Payment Card Industry Data Security Standard):** A set of security standards designed to protect card information during and after a financial transaction.
- **SOX (Sarbanes-Oxley Act):** A U.S. federal law aimed at protecting investors by improving the accuracy and reliability of corporate disclosures.
- **Basel III:** An international regulatory framework designed to strengthen regulation, supervision, and risk management within the banking sector.

8. Financial Instruments

- **Cheque:** A written, dated, and signed instrument that directs a bank to pay a specific sum of money to the bearer or the order of a specific person.
- **Demand Draft:** A prepaid negotiable instrument used for transferring money from one location to another.
- **Bank Guarantee:** A guarantee provided by a bank that if a borrower defaults on a loan, the bank will cover the loss.
- **Letter of Credit (LC):** A document from a bank guaranteeing that a seller will receive payment from the buyer as agreed.

9. Digital Banking

- **Internet Banking:** A service that allows customers to conduct financial transactions online through a bank's website.
- **Mobile Banking:** The use of a mobile device to conduct banking activities, such as checking account balances, transferring funds, or paying bills.
- **UPI (Unified Payments Interface):** An instant real-time payment system that facilitates inter-bank transactions in India through mobile devices.
- **e-Wallet:** A digital wallet used to store payment information and make electronic transactions.

10. Risk and Fraud

- **Credit Risk:** The risk of loss due to a borrower's failure to repay a loan.
- **Market Risk:** The risk of losses in investments due to market fluctuations.
- **Operational Risk:** The risk of loss due to failed internal processes, people, systems, or external events.
- **Fraud Detection:** The process of identifying fraudulent activities in banking transactions using various tools and techniques.
- Understanding these terms is crucial for navigating and working within the banking domain, especially when involved in software development and testing.

Project Task :

Complete Domain testing E-Banking project , The URL will be provided in the LAB Session

Insurance – Domain Testing

Definition of Insurance?

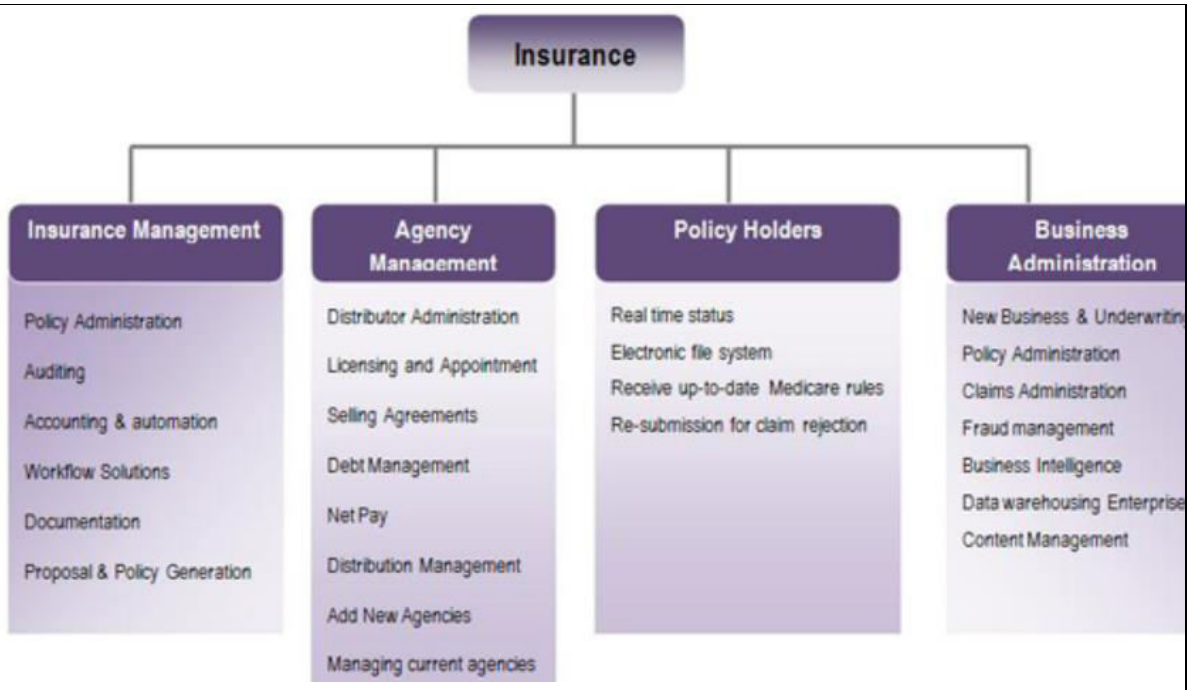
Insurance is a financial arrangement in which an individual or entity (the policyholder) pays regular premiums to an insurance company in exchange for protection against specific risks or financial losses. In the event of a covered loss or event, the insurance company provides financial compensation or reimbursement, helping to mitigate the impact of the loss on the policyholder.

Purpose of Insurance:

The primary purpose of insurance is to provide financial security and peace of mind by transferring the risk of significant financial loss from the policyholder to the insurer. It helps individuals and businesses manage potential risks and recover from unforeseen events without bearing the full financial burden.

Types of Insurance:

- **Life Insurance:** Provides financial protection to beneficiaries in the event of the policyholder's death.
- **Health Insurance:** Covers medical expenses, including doctor visits, hospital stays, surgeries, and prescription drugs.
- **Auto Insurance:** Covers losses or damages related to vehicles, including accidents, theft, and liability for injuries caused to others.
- **Homeowners Insurance:** Protects against damages to a home and its contents, as well as liability for accidents that occur on the property.
- **Travel Insurance:** Provides coverage for unexpected events during travel, such as trip cancellations, medical emergencies, and lost luggage.
- **Business Insurance:** Covers various risks faced by businesses, including property damage, liability, and employee-related risks.



Characteristics of Insurance Application

1. Policy Management :
2. Claims Processing
3. Underwriting
4. Customer Management
5. Regulatory Compliance
6. Financial Management
7. Security
8. Integration with Other Systems
9. Customization and Flexibility
10. Reporting and Analytics
11. Scalability
12.User experience

1. Policy Management :

Policy Creation and Management: The application allows for the creation, issuance, and management of various types of insurance policies (e.g., life, health, auto, property). It should handle policy life cycles from issuance to renewal, amendment, and cancellation.

Coverage Details: The system maintains detailed records of policy coverages, limits, deductibles, and exclusions.

Premium Calculation: The application calculates premiums based on various factors, such as the type of coverage, risk factors, and policyholder information.

2. Claims Processing

Claims Submission: The system allows policyholders or agents to submit claims, providing necessary details such as the nature of the loss, supporting documents, and incident reports.

Claims Adjudication: The application should support the review and adjudication of claims, determining the validity of the claim and the amount to be paid out.

Claims Payment: It manages the disbursement of claim payments, including tracking payments made to claimants, repair services, and healthcare providers.

3. Underwriting

Risk Assessment: The system supports underwriting by assessing the risk associated with insuring an individual or entity. This involves evaluating factors like health status, driving history, property location, and more.

Decision Making: Based on the risk assessment, the system helps underwriters make decisions about whether to accept or reject an insurance application and at what premium.

4. Customer Management

Customer Information: The application stores detailed customer profiles, including personal information, contact details, and policy history.

Interaction History: It tracks all interactions with the customer, including policy inquiries, claims filed, and premium payments.

Self-Service Portal: Many insurance applications provide a self-service portal where customers can manage their policies, make payments, and file claims.

5. Regulatory Compliance

Compliance with Laws: The application ensures that all insurance products and processes comply with local, national, and international regulations (e.g., GDPR, HIPAA, Solvency II).

Reporting: It generates reports required by regulatory bodies, such as financial disclosures, customer privacy reports, and anti-money laundering (AML) compliance reports.

6. Financial Management

Billing and Payments: The system handles the billing of premiums, including generating invoices, processing payments, and managing overdue accounts.

Financial Reporting: It provides financial reports related to the performance of insurance products, including profit margins, loss ratios, and reserve calculations.

7. Security

Data Encryption: Given the sensitivity of personal and financial information, the application must use strong encryption to protect data at rest and in transit.

Access Control: The system must enforce strict access control measures to ensure that only authorized users can access sensitive information and perform critical operations.

Fraud Detection: The application should include mechanisms to detect and prevent fraudulent activities, such as false claims or identity theft.

8. Integration with Other Systems

Third-Party Integrations: Insurance applications often integrate with third-party systems, such as payment gateways, credit bureaus, healthcare providers, and government databases.
APIs: The system should provide APIs to allow for integration with other software applications, enabling data sharing and process automation across different platforms.

9. Customization and Flexibility

Product Customization: The application should allow insurers to customize insurance products to meet the needs of different market segments.
Workflow Management: It should support the customization of workflows to align with the insurer's specific business processes, including underwriting, claims processing, and customer service.

10. Reporting and Analytics

Data Analytics: The system should provide robust analytics capabilities to help insurers analyze risk, customer behavior, claims trends, and financial performance.
Reporting Tools: It should offer customizable reporting tools that allow users to generate various types of reports, such as sales performance, claims ratios, and customer demographics.

11. Scalability

Handling Volume: The application should be scalable to handle large volumes of policies, claims, and customer interactions, especially for large insurance companies with millions of clients.
Cloud-Based Solutions: Many modern insurance applications are cloud-based, allowing them to scale resources up or down based on demand.

12. User Experience

Intuitive Interface: The application should have a user-friendly interface that is easy to navigate for both internal users (e.g., agents, underwriters) and customers.
Multi-Platform Support: The system should be accessible on multiple platforms, including web browsers, mobile apps, and tablets, to accommodate different user preferences.

Common Terms of Insurance :

1.Policy-Related Terms

- **Policyholder:** The individual or entity that owns the insurance policy and pays the premiums.
- **Insurer:** The insurance company that provides coverage and pays claims under the terms of the insurance policy.
- **Premium:** The amount of money paid by the policyholder to the insurer, usually on a regular basis (e.g., monthly, annually), in exchange for

coverage.

- **Policy:** A contract between the insurer and the policyholder that outlines the terms, conditions, coverage limits, and exclusions of the insurance coverage.
- **Coverage:** The specific risks, events, or circumstances that the insurance policy protects against (e.g., health issues, property damage, death).
- **Claim:** A formal request made by the policyholder to the insurance company for payment or compensation when a covered event occurs.
- **Deductible:** The amount of money the policyholder is required to pay out-of-pocket before the insurance company begins to cover a claim.
- **Beneficiary:** The person or entity designated by the policyholder to receive the insurance payout in the event of the policyholder's death or other specified events.
- **Policy Term:** The duration for which the insurance policy is in effect, typically defined by a start date and an end date.

- **Endorsement:** A modification or addition to an existing insurance policy that changes the terms or coverage.
- **Renewal:** The process of extending the duration of an insurance policy beyond its original term, typically involving the payment of additional premiums.

2. Claim-Related Terms

- **Claim:** A formal request made by the policyholder to the insurance company for payment or compensation when a covered event occurs.
- **Claimant:** The person or entity making a claim under an insurance policy, typically the policyholder or a beneficiary.
- **Deductible:** The amount of money the policyholder must pay out-of-pocket before the insurance company begins to cover the costs of a claim.
- **Settlement:** The payment made by the insurance company to the claimant as compensation for a covered loss.
- **Loss Adjuster:** A professional employed by the insurer to assess the validity and value of a claim.
- **Subrogation:** The process by which an insurance company seeks reimbursement from a third party that is responsible for the loss after compensating the policyholder.

3. Risk and Underwriting

- **Risk:** The potential for loss or damage, which is assessed by the insurer to determine the likelihood of a claim and the appropriate premium.
- **Underwriting:** The process by which an insurer evaluates the risk of

insuring a person or asset and determines the terms and pricing of the insurance policy.

- **Actuary:** A professional who analyzes statistical data to assess risk and determine premium rates for insurance policies.
- **Reinsurance:** Insurance purchased by an insurance company from another insurer to protect against large claims or catastrophic losses.
- **Risk Pooling:** The practice of spreading risk among a large number of policyholders, which helps to stabilize premium rates and ensure that funds are available to pay claims.

4. Beneficiary and Payout

- **Beneficiary:** The person or entity designated by the policyholder to receive the insurance payout in the event of the policyholder's death or other specified events.
- **Face Value:** The amount of money the insurer agrees to pay the beneficiary upon the occurrence of the insured event, such as the death of the policyholder in life insurance.
- **Surrender Value:** The amount the policyholder receives if they choose to terminate the policy before its maturity or before a claim is made.
- **Annuity:** A financial product that provides a stream of income, typically used for retirement, where payments are made at regular intervals, usually for the life of the annuitant.

5. Legal and Regulatory Terms

- **Insurable Interest:** A legal requirement that the policyholder must have a financial interest in the insured person or property, meaning they would suffer a loss if the insured event occurs.
- **Grace Period:** The additional time allowed for the policyholder to pay a premium after the due date without losing coverage.
- **Indemnity:** A principle in insurance where the insured is compensated for the actual loss incurred, restoring them to their financial position before the loss, without allowing them to profit.
- **Insurance Fraud:** The act of falsifying or exaggerating claims to receive payouts from an insurance policy, which is illegal and punishable by law.
- **Compliance:** Adherence to laws, regulations, and guidelines relevant to the insurance industry, including those set by regulatory bodies like the Insurance Regulatory and Development Authority (IRDA) in India or the National Association of Insurance Commissioners (NAIC) in the U.S.

6. Miscellaneous Terms

- **Grace Period:** The period after the premium due date during which the policy remains in force without penalty, even if the premium has not been paid.
- **Broker:** An intermediary who represents the policyholder and helps them

find the best insurance policy by comparing different insurers.

- **Agent:** An individual or entity authorized to sell insurance policies on behalf of an insurance company, representing the insurer rather than the policyholder.
- **Endowment Policy:** A life insurance policy that pays out a lump sum either on a specific date or upon the policyholder's death, whichever comes first.
- **Umbrella Policy:** A type of insurance policy that provides additional liability coverage beyond the limits of the policyholder's other insurance policies, such as auto or home insurance.

Real Time Regular Interview Questions

1. Introduce yourself?

- Your Name
- Trainings in case of fresher
- Roles in Manual testing
- Roles in Automation testing
- Projects you have done
- Academic details
- Working company details in case of experience.

2. Explain STLC?

Software Testing Life Cycle (STLC) is a process used to test software and ensure that quality standards are met. Tests are carried out systematically over several phases. During product development, phases of the STLC may be performed multiple times until a product is deemed suitable for release.
STLC is an part of Software Development Life Cycle (SDLC).STLC deals only with the testing phases.

3.Explain Your role in the project?

- Understating the requirements of the application
- Identifying required Test Scenarios of the project
- Designing and preparing Test cases to validate application
- Execute test cases to validate application
- Logs Test Results(How many Test cases passes/failed)
- Defect Report and Tracking
- Retest fixed defects of previous builds
- Performed various Types of testing assigned by Test Lead(Sanity
- ,Functionality , Usability, Compatibility , etc)
- Preparing and Sending of status Reports to Lead on assigned tasks
- Participated in regular meetings , team meetings by lead & Manager
- Creating automation scripts for Regression testing

4.What is manual testing and automation testing?

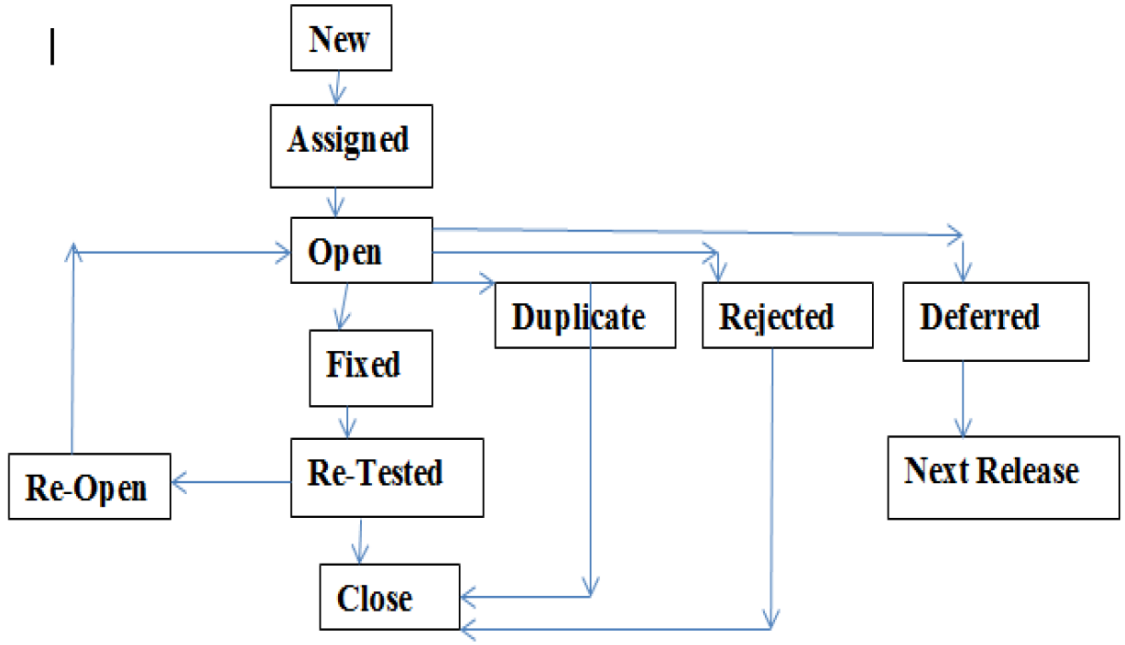
Manual Testing: It is a type of software testing in which test cases are executed manually by a tester without using any automated tools.

Automation Testing: This software testing method uses scripted sequences that are executed by automation testing tools.

5.Explain bug life cycle?

The defect life cycle is also known as bug life cycle. It is a process in which bug goes through different stages in its entire life.

Consider the bug as a lively object that has different stages throughout its life right from the state it's opened by tester to the state of getting closed after getting fixed.



6.What is RTM?

RTM stands for "Requirements Traceability Matrix".A traceability matrix is a mapping document between requirement documents and Test Cases . it helps the testing team to understand the level of testing that is done for a given product.

7.Explain agile process?

Overview:

- Customer satisfaction by fast delivery.
- Requirement changes are accepted.
- Software is delivered frequently.
- The ability to move quickly ,easily and respond to change rapidly.
- Agile terminologies

userstory,productbacklog,productowner,scrummaster,sprint,stand-up meeting,scrummeeting,status meeting.

Advantages:

- Face-to-Face conversation is the best form of communication.
- close co-operation between business people and developers.

- customerssatisfication by rapid,continuous delivery of useful software.

Disadvantages:

- In agile methodology ,dependency on documentation.
- for complex projects,the resource requirement and effort are difficult to estimate.
- projects can be become ever-lasting because there's no clear end.

8. Explain retesting & regression testing?

Retesting: Testing of a particular bug after it has been fixed to confirm that bug has been fixed correctly are not.

Regression testing: It is a type of testing where you can verify that the changes made in the codebase do not impact the existing software functionality.

9. Difference between QA & Software Testing?

Software testing focuses on the functionality of the software and if there are any bugs in it. Quality assurance is a management approach for ensuring the successful implementation of the company's quality objectives.

10. What is Test Plan?

A test plan is a document detailing the objectives,resources,and processes for a specific test for a software or hardware product.The plan typically contains a detailed understanding of the eventual workflow.

11.Difference between Test Plan & Test Cases?

Test Plan	Test Case
A large detailed document that covers management aspects and testing aspects in the entire testing project.	Specific and precise document for a particular testing feature that covers only testing aspects.
Testers,testleaders,managers,stakeholders,and other departments need to be updated about the testing process.	Only testing teams and test leaders.
Both testing & project managing aspects like schedule,scope,potential risks,staff responsibilities,bugs reporting,and more.	Only testing aspects such as test steps,testdata,testenvironment,intended test results,real test results,teststatus,etc.,
Till the end of the whole testing project.	Till the end of the particular testing process.

12.What is Data Driven Testing?

Data-driven Testing(DDT) is also known as "Table-driven testing".DDT is data that is external to your functional tests,and is loaded and used to extend your automated test cases.

You can take the same test case and run it with as many different inputs as you

like,thus getting better coverage from a single test.

13.What is System Testing?

System testing also referred to as system-level tests or system-integration testing,is the process in which a quality assurance(QA) team evaluates how the various components of an application interact together in the full,integrated system or application.

14.Who will Take care about System Testing?

System testing includes functional & non-functional testing and is performed by the testers. **[OR]**

In general,System- integration testing,especially the end-to-end test,is the responsibility of the testers.

15.What is GUI?

Graphical User Interface(GUI), is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation,insteadot text-based UIs,typed command labels or text navigation.

16.What is Smoke Testing & Sanity Testing?

Smoke test is done to make sure that the critical functionalities of the program are working fine, whereas sanity testing is done to check that newly added functionalities, bugs, etc., have been fixed. The software build may be either stable or unstable during smoke testing.

17.What is Test Stategy?

A Test Strategy is a set of instructions, guidelines or principles that determine the test design and how the testing process will be carried out. It sets the testing process standards.

It can comprise of the principles used to detail the scope and overview of the testing process, testing methodology, specifications for the testing environment, tools used for testing, release control, risk analysis and mitigation, review and approval of a product before its official release.

18.What is Test Case Design Techinques?

Test Case: Design techniques can broadly split in to 2 categories.

- Black box techniques
- White box techniques

Black box techniques:

- Equivalence Class Partitioning(ECP)
- Boundary Value Analysis(BVA)
- State Transition
- Decision Table / Cause Effect Table

White box techniques:

- Statement Testing
- Branch/Decision testing
- Data flow Testing
- Branch condition testing

19.What is concrete testing?

concrete testing is - how data moves through the testing workflow – and, in the process, is better able to structure and secure the data forever.

concrete testing an interface for accessing and communicating with that database, and a way to integrate everything with your concrete testing machine.

20. What is build testing?

Build Verification Testing is basic testing performed on every new build to make sure that it conforms to what is required of it before being passed to the testing or UAT team for further testing. This procedure can also called Build Acceptance Testing or Smoke Testing and is a measure of stability of the software build.

21.What is Test scenarios,test cases?

Test scenarios:

A Test Scenario is a statement describing the functionality of the application to be tested. It is used for end-to-end testing of a feature and is generally derived from the requirement document.

Test scenarios can serve as the basis for lower-level test case creation. A single test scenario can cover one or more test cases. Therefore a test scenario has a one-to-many relationship with the test cases.

Test cases:

A test case is a document, which has a set of test data, preconditions, expected results and post conditions, developed for a particular test scenario in order to verify compliance against a specific requirement. Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution post condition.

It is an in-details document that contains all possible inputs (positive as well as negative) and the navigation steps, which are used for the test execution process. Writing of test cases is a one-time attempt that can be used in the future at the time of regression testing.

22.What are the test steps?

Test Steps **describe the execution steps and expected results that are documented against each one of those steps**. Each step is marked pass or fail based on the comparison result between the expected and actual outcome.

23.What is bug testing?

A malfunction in the software/system is **an error that may cause components or the system to fail to perform its required functions**. In other words, if an error is encountered during the test it can cause malfunction. For example, incorrect data description, statements, input data, design, etc.

24. Difference between validation and verification?

Verification	Validation
check whether we are developing the right product or not.	check whether the developed product is right.

Verification includes different methods like Inspections, Reviews, and Walkthroughs	In the validation testing, we can find those bugs, which are not caught in the verification process.
In verification testing, we can find the bugs early in the development phase of the product.	Validation includes testing like functional testing, system testing, integration, and User acceptance testing.
The goal of verification is application and software architecture and specification.	The goal of validation is an actual product.

25.what is testing?

Testing is the actual method for finding issues, or bugs, with the quality of the application. The testing process ensures the software meets standards and user requirements.

26. why testing required?

- To identify the defects in development phases
- To ensure Quality of the product
- Saves Money as defect identified in earlier stages
- To build customer confidence and business

27.how manual testing help in real time?

When user is required to carry out every activity related to testing manually, we say it is a Manual testing process. It helps in

- To identify the defects in development phases
- To ensure Quality of the product
- Saves Money as defect identified in earlier stages
- To build customer confidence and business

28. Shall we use tools in manual testing

Yes, a test case management tool helps to organize, measure, report, and collaborate on manual testing process. Testing usually involves a lot of documentation, and if the right tools and processes are not in place, things can quickly become disorganized. A good test case management tool can make life a lot easier when it comes to keeping those assets organized and up to date. Suggested tools like Jira , QC , TestLodge,Zephyr,Testlink …

29. when you will do retesting?

Once the development team releases the new build, then the test team has to test the already posted bugs to make sure that the bugs are fixed or not.

30.what is SDLC?

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software.

31.what is smoke testing?

Smoke Testing is done whenever the new functionalities of software are

developed and integrated with existing build that is deployed in QA/staging environment. It ensures that all critical functionalities are working correctly or not.

32.Is smoke testing done by developer (or) tester?

It is carried out by both the testers and developers because of its ease and less time commitment. It is part of the rigorous testing process and uses test cases to check all important components of the build. Smoke testing is performed when the developers provide a fresh build to the Quality Assurance teams.

33.what is BVA?

Boundary-value analysis is a software testing technique in which tests are designed to include representatives of boundary values in a range of using formula as min , max,min-1,max+1

34.can we write BVA FOR A BUTTON?

Test cases/scenarios For Radio Buttons
Check for selecting a radio button
Check for only one radio button is getting selected at a time
Check for the Radio button label description
Check for size and color of button
Check for the alignment of the Radio Buttons

35.explain about the BA?

A **Business Analyst** is a person who analyses an organization, designs processes, and systems, and assesses the business models so that they can be integrated with technology.

36.what is test scenarios?

Identify all possible areas to be tested or What is to be tested.
A Test Scenario is a statement describing the functionality of the application to be tested. It is used for end-to-end testing of a feature and is generally derived from the requirement document. Test scenarios can serve as the basis for lower-level test case creation. A single test scenario can cover one or more test cases. Therefore a test scenario has a one-to-many relationship with the test cases.

37.how can you write test cases &test scenarios?

How to create a Test Scenario:

Carefully study the Requirement Document – Business Requirement Specification (BRS), Software Requirement Specification (SRS), Functional Requirement Specification (FRS) pertaining to the System Under Test (SUT).

- Isolate every requirement, and identify what possible user actions need to be tested for it. Figure out the technical issues associated with the requirement. Also, remember to analyze and frame possible system abuse scenarios by evaluating the software with a hacker's eyes.
- Enumerate test scenarios that cover every possible feature of the software. Ensure that these scenarios cover every user flow and business flow

involved in the operation of the website or app.

- After listing the test scenarios, Get the scenarios reviewed by a team

38.The build is delay how to convince the client?

Be open and honest about all the obstacles, difficulties, and mishaps that emerged. Show them proof and try to explain in a simple way why the delay has originally happened and what needs to be done in order for your team to overcome the issues. The majority of clients are really understanding when you approach them with honesty and willingness to keep them on board.

39.what is bug report?

A bug report is a specific report that outlines information about what is wrong and needs fixing with software or on a website.The report lists reasons,or seen errors,to point out exactly what is viewed as wrong,and includes a request and or details for how to address each issue.

40.what is pesticide paradox in testing?

The pesticide paradox says that if the same tests are repeated over and over again, eventually, the same set of test cases will no longer identify any new bugs in the system. To overcome this 'pesticide paradox,' the test cases need to be regularly reviewed and revised.

41.what is concurrent testing?

Concurrency Testing is defined as a testing technique to detect the defects in an application when multiple users are logged in. In other words monitoring the effect while multiple users perform the same action at the same time.

42.what is software testing?

“Software testing is a process of executing the application with the intent of finding the defects by comparing the output behavior of the application with expected behaviour requirement.

In other words it’s comparing the actual behavior of an application with expected behavior.

43.what is manual testing?

Manual Testing is a type of software testing in which test cases are executed manually by a tester without using any automated tools. The purpose of Manual Testing is to identify the bugs, issues, and defects in the software application. Manual software testing is the most primitive technique of all testing types and it helps to find critical bugs in the software application.

44.what is positive testing?

Positive Testing is a type of testing which is performed on a software application by providing the valid data sets as an input. It checks whether the software application behaves as expected with positive inputs or not. Positive testing is performed in order to check whether the software application does exactly what it is expected to do.

45.for manual testing can we use any tools in it?

No, first we have to refer to the client which type of tool should be used in this

project.By using POC(proof of concept) .so,that client can get idea on these tool, what is the use of it and why we are using this particular tool.

46.after new bug is fixed by the developer ,first what you test retesting or regression testing?

When ever new bug has been fixed by the developer,first we we have to do retesting,so that we can check whether that bug is working as expected or not.After that we have to go for regression testing.

47.what is smoke testing?

Smoke Testing is a software testing process that determines whether the deployed software build is stable or not. Smoke testing is a confirmation for QA team to proceed with further software testing. It consists of a minimal set of tests run on each build to test software functionalities. Smoke testing is also known as “Build Verification Testing” or “Confidence Testing.”

48.what is BVA? Expalin with example?

BVA:BVA is another **Black Box Test Design Technique**, which is used to find the errors at boundaries of input domain (tests the behavior of a program at the input boundaries) rather than finding those errors in the centre of input.

If (Min,MAX) is the range given for a field validation, then the boundary values come as follows:

Invalid Boundary Check { Min-1 ; Max+1 }

Valid Boundary Check {Min; Min+1 ;Max-1;Max }

Requirement: Validate AGE field, which accepts values from 21-60. We verify the following Boundary Value

Test cases:

TC001: Validate AGE by entering 20 [Min-1]: Invalid Boundary Check

TC002: Validate AGE by entering 21 [Min]: Valid Boundary Check TC003: Validate AGE by entering 22 [Min+1]: Valid Boundary Check TC004: Validate AGE by entering 59 [Max-1]: Valid Boundary Check TC005: Validate AGE by entering 60 [Max-1]: Valid Boundary Check TC006: Validate AGE by entering 61[Max+1]: Invalid Boundary Check

49.can we write BVA for customer ID?it access only 5 to 8 characters?

50.what is functional testing?

Functional testing is **a type of testing that seeks to establish whether each application feature works as per the software requirements**. Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user's expectations.

51.what is unit testing?

A unit test is **a way of testing a unit - the smallest piece of code that can be logically isolated in a system**. In most programming languages, that is a

function, a subroutine, a method or property. The isolated part of the definition is important.

Or

Integration testing is **the second level of the software testing process comes after unit testing**. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units

53.what is test plan?

A test plan is a document detailing the objectives,resources,and processes for a specific test for a software or hardware product.The plan typically contains a detailed understanding of the eventual workflow.

54.what is test scenarios?explain?

ATest Scenario is defined as any functionality that can be tested. It is also called *Test Condition* or *Test Possibility*. As a tester, you should put yourself in the end user's shoes and figure out the real-world scenarios and use cases of the Application Under Test.

Entry Criteria to Identify Test Scenarios :

- Approved Test plan
- Approved SRS

Test Scenarios Guidelines

- Test Scenario template

Exit Criteria for Test Scenarios :

- Test Scenarios should be reviewed & approved(mapping test scenarioswith requirements)

55.what is test cases?expalin?

Test cases define how to test a system, software or an application. A test case is a singular set of actions or instructions for a tester to perform that validates a specific aspect of a product or application functionality. If the test fails, the result might be a software defect that the organization can triage. Entry Criteria to Identify Test Cases :

- Approved Test plan
- Approved SRS
- Approved FRS
- Approved Test Scenarios
- Test Case Guidelines
- Test Case Template

Exit Criteria for Test Cases :

- Test Cases should be reviewed & approved(mapping test scenarios with requirements)

56.Explain your Role in project?

- Understating the requirements of the application

- Identifying required Test Scenarios of the project
- Designing and preparing Test cases to validate application
- Execute test cases to validate application
- Logs Test Results(How many Test cases passes/failed)
- Defect Report and Tracking
- Retest fixed defects of previous builds
- Performed various Types of testing assigned by Test Lead(Sanity
- ,Functionality , Usability, Compatibility , etc)
- Preparing and Sending of status Reports to Lead on assigned tasks
- Participated in regular meetings , team meetings by lead & Manager
- Creating automation scripts for Regression testing

57.WHAT IS EXPLORATORY TESTING

Exploratory Testing is a type of software testing where Test cases are not created in advance but testers check system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a "thinking" activity.

Exploratory Testing is widely used in Agile models and is all about discovery, investigation, and learning. It emphasizes personal freedom and responsibility of the individual tester.

58.TELL ME RISK-BASED TESTING?

Risk-based testing (RBT) is a type of software testing that functions as an organizational principle used to prioritize the tests of features and functions in software, based on the risk of failure, the function of their importance and likelihood or impact of failure.

59.What is your role in project?

- Understating the requirements of the application
- Identifying required Test Scenarios of the project
- Designing and preparing Test cases to validate application
- Execute test cases to validate application
- Logs Test Results(How many Test cases passes/failed)
- Defect Report and Tracking
- Retest fixed defects of previous builds
- Performed various Types of testing assigned by Test Lead(Sanity
- ,Functionality , Usability, Compatibility , etc)
- Preparing and Sending of status Reports to Lead on assigned tasks
- Participated in regular meetings , team meetings by lead & Manager
- Creating automation scripts for Regression testing

60.Types of testing?

There are three types of testing.They are:

□□**White box testing:**White-box testing is a testing technique which checks the internal functioning of the system. In this method, testing is based on coverage of code statements, branches, paths or conditions. White-Box testing is

considered as low-level testing. It is also called glass box, transparent box, clear box or code base testing. The white-box Testing method assumes that the path of the logic in a unit or program is known.

- **Black box testing:**

In Black-box testing, a tester doesn't have any information about the internal working of the software system. Black box testing is a high level of testing that focuses on the behavior of the software. It involves testing from an external or end-user perspective. Black box testing can be applied to virtually every level of software testing: unit, integration, system, and acceptance.

Grey Box Testing:

While white box testing assumes the tester has complete knowledge, and black box testing relies on the user's perspective with no code insight, grey box testing is a compromise. It tests applications and environments with partial knowledge of internal workings. Grey box testing is commonly used for penetration testing, end-to-end system testing, and integration testing.

61.What are the types functional testing?

- Smoke testing.
- Sanity testing.
- Retesting
- Regression testing.
- Static testing
- Dynamic testing
- Adhotesting
- Alpha testing
- Beta testing
- Functionality testing
- Usability testing
- Compatability testing

62.Write test cases for email password Positive & Negative

Requirement is : Password should be {alphanumeric} {6-12 characters}

63.What you done in adactin?

- Understating the requirements of the application
- Identifying required Test Scenarios of the project
- Designing and preparing Test cases to validate application
- Execute test cases to validate application
- Logs Test Results(How many Test cases passes/failed)
- Defect Report and Tracking
- Retest fixed defects of previous builds
- Performed various Types of testing assigned by Test Lead(Sanity
- ,Functionality , Usability, Compatibility , etc)
- Preparing and Sending of status Reports to Lead on assigned tasks
- Participated in regular meetings , team meetings by lead & Manager
- Creating automation scripts for Regression testing

65.Your part of in that HRMS?

- Understating the requirements of the application
- Identifying required Test Scenarios of the project
- Designing and preparing Test cases to validate application
- Execute test cases to validate application
- Logs Test Results(How many Test cases passes/failed)
- Defect Report and Tracking
- Retest fixed defects of previous builds
- Performed various Types of testing assigned by Test Lead(Sanity
- ,Functionality , Usability, Compatibility , etc)
- Preparing and Sending of status Reports to Lead on assigned tasks
- Participated in regular meetings , team meetings by lead & Manager
- Creating automation scripts for Regression testing

66.What is Api testing?

API TESTING is a software testing type that validates Application Programming Interfaces (APIs). The purpose of API Testing is to check the functionality, reliability, performance, and security of the programming interfaces. In API Testing, instead of using standard user inputs(keyboard) and outputs, you use software to send calls to the API, get output, and note down the system's response. API tests are very different from GUI Tests and won't concentrate on the look and feel of an application. It mainly concentrates on the business logic layer of the software

67.Explain black box & white box testing?

White box testing: White-box testing is a testing technique which checks the internal functioning of the system. In this method, testing is based on coverage of code statements, branches, paths or conditions. White-Box testing is considered as low-level testing. It is also called glass box, transparent box, clear box or code base testing. The white-box Testing method assumes that the path of the logic in a unit or program is known.

Black box testing:

In Black-box testing, a tester doesn't have any information about the internal working of the software system. Black box testing is a high level of testing that focuses on the behavior of the software. It involves testing from an external or end-user perspective. Black box testing can be applied to virtually every level of software testing: unit, integration, system, and acceptance.

68.What are the Levels of testing?

There are mainly four **Levels of Testing** in software testing :

Unit Testing : checks if software components are fulfilling functionalities or not.

2. **Integration Testing** : checks the data flow from one module to other modules.
3. **System Testing** : evaluates both functional and non-functional needs for the testing.

4. **Acceptance Testing** : checks the requirements of a specification or contract are met as per its delivery

69.ExplainTestCase Design Techquies?

Test Case: Design techniques can broadly split in to 2 categories.
Black box techniques
White box techniques

Black box techniques:

- Equivalence Class Partitioning(ECP)
- Boundary Value Analysis(BVA)
- State Transition
- Decision Table / Cause Effect Table

White box techniques:

- Statement Testing
- Branch/Decision testing
- Data flow Testing
- Branch condition testing

70.Write Test cases and scenarios for Gmail login btn?

71.What is Functional testing &Non functional testing?

Functional testing:

Functional testing is a type of testing which verifies that each **function** of the software application operates in conformance with the requirement specification. This testing mainly involves black box testing, and it is not concerned about the source code of the application.
Every functionality of the system is tested by providing appropriate input, verifying the output and comparing the actual results with the expected results. This testing involves checking of User Interface, APIs, Database, security, client/ server applications and functionality of the Application Under Test. The testing can be done either manually or using automation.

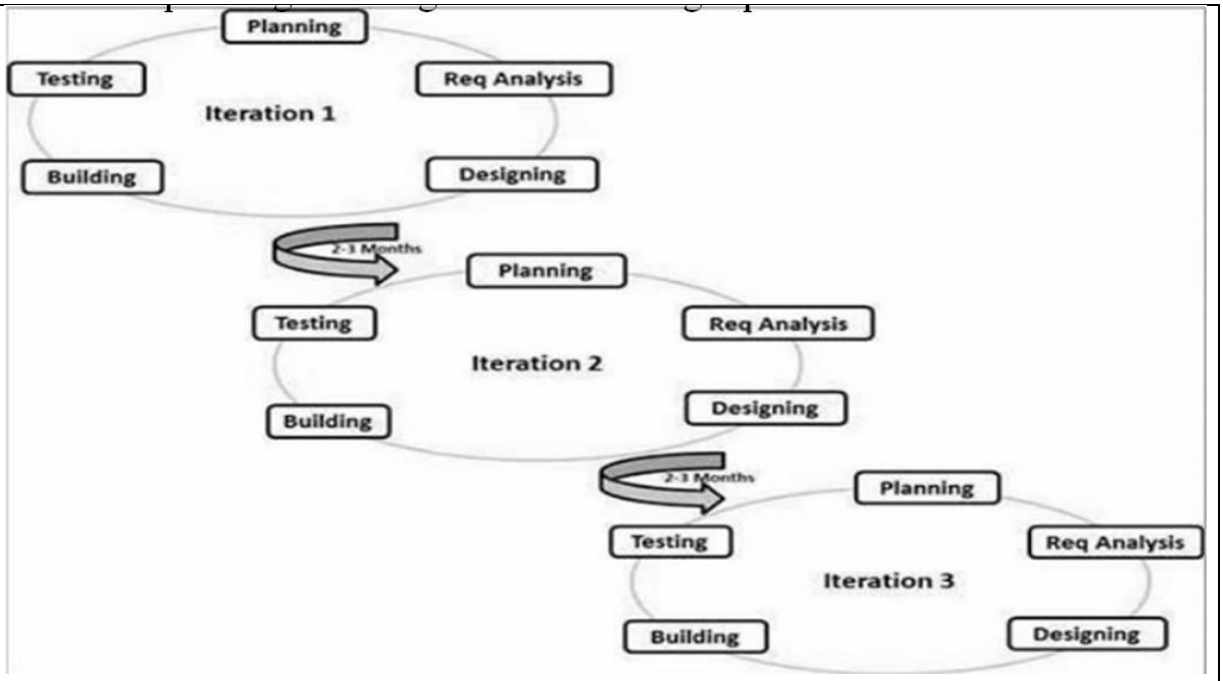
Non functional testing:

Non-functional testing is a type of testing to check non-functional aspects (performance, usability, reliability, etc.) of a software application. It is explicitly designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing.

72.Explain agile methodology?

Agile Methodology meaning a practice that promotes **continuous iteration** of development and testing throughout the software development lifecycle of the project. In the Agile model in software testing, both development and testing activities are concurrent, unlike the Waterfall model.
The agile software development emphasizes on four core values.

1. Individual and team interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan.



Scrum:

SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team-based development environment. Basically, Scrum is derived from activity that occurs during a rugby match. Scrum believes in empowering the development team and advocates working in small teams (say- 7 to 9 members). Agile and Scrum consist of three roles, and their responsibilities are explained as follows:

Product Backlog

This is a repository where requirements are tracked with details on the no of requirements(user stories) to be completed for each release. It should be maintained and prioritized by Product Owner, and it should be distributed to the

scrum team. Team can also request for a new requirement addition or modification or deletion.

73.Why do you choose testing then Development?

Any software or product is incomplete without its testing. Without testing, there could be some bugs in the product and customers will not get satisfied according to the requirements. For testing, all companies need to have software tester for complete and satisfied software or product without any errors or bugs. Software Testers Are Made for Challenging Work Environments

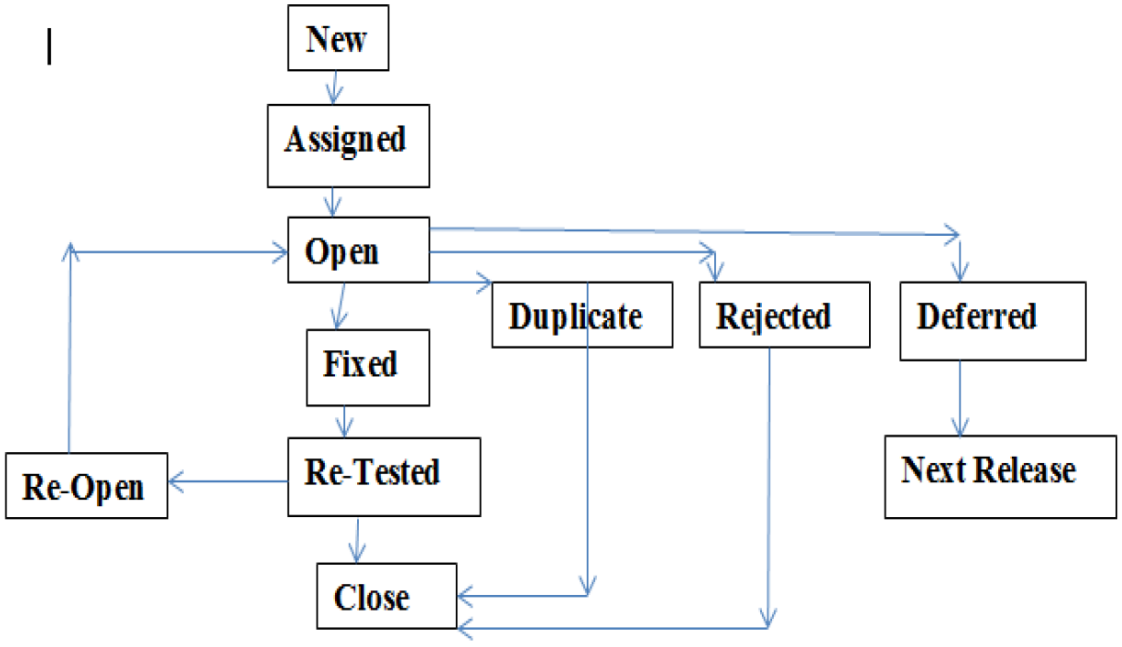
- You Can Enjoy Every Day of Work
- Flexible and Fun Work Environment
- It's Creative
- It Is a Secure Career Path
- There Is Attractive Remuneration and Room for Growth
- An Academic Background Isn't a Necessity

74.What is Agile Model?

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations.

75.What are the phases of bug life cycle?

- New.
- Open
- Fixed
- Pending Retest
- Retest
- Reopen
- Verified



76.Difference between bug and Defect?

We can say that a mistake made by a programmer during coding is called an error, an error found during the unit testing in the development phase is called a defect, an error found during the testing phase is called a bug and when an error is found at an end user's end is called as the failure.

77.What is Dynamic Testing?

Dynamic testing technique is the type of testing that validates the functionality of an application when the **code is executed** / by executing the code. In simple terms dynamic testing is performed by actually using the application and seeing if a functionality works the way it is expected to.

78.What is Static Testing?

Static testing as the name itself suggests is static in nature, which also means there are no changing conditions or parameters. In other words, this is performed **without executing the code**.

79.What is priority and severity?

Severity : Importance of defect with respective to functional point of view…means criticalness of defect with respective to application.

Severity classification could be : S1-Urgent ,S2-High ,S3-Medium ,S4-Low , CRITICAL , HIGH ,MEDIUM ,LOW

Priority : Importance of defect with respective to client point of view … means how soon it should be fix.

Priority Classification could be : P1-Urgent ,P2-High ,P3-Medium ,P4-Low, CRITICAL , HIGH ,MEDIUM ,LOW

80.What is Functional and Non Functional Testing?

Functional testing:

Functional testing is a type of testing which verifies that each **function** of the software application operates in conformance with the requirement specification. This testing mainly involves black box testing, and it is not concerned about the source code of the application.Every functionality of the system is tested by providing appropriate input, verifying the output and comparing the actual results with the expected results. This testing involves checking of User Interface, APIs, Database, security, client/ server applications and functionality of the Application Under Test. The testing can be done either manually or using automation.

Non functional testing:

Non-functional testing is a type of testing to check non-functional aspects (performance, usability, reliability, etc.) of a software application. It is explicitly designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing.

81.What is meant by sprint?

In Agile product development, a sprint is a set period of time during which specific work has to be completed and made ready for review.

82.Explain white box Techniques?

White Box Testing is a testing technique in which software's internal structure, design, and coding are tested to verify input-output flow and improve design, usability, and security. In white box testing, code is visible to testers, so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing.

Following are important WhiteBox Testing Techniques:

- Statement Coverage
- Decision Coverage
- Branch Coverage
- Condition Coverage
- Multiple Condition Coverage
- Finite State Machine Coverage
- Path Coverage
- Control flow testing
- Data flow testing

83.What is Testing ?

In general, testing is **finding out how well something works**. In terms of human beings, testing tells what level of knowledge or skill has been acquired. In computer hardware and software development, testing is used at key checkpoints in the overall process to determine whether objectives are being met.

84.Explain ECP and State Transition Table with one example each?

85.Difference between manual & automation testing?

Aspect of Testing	Manual	Automation
Test Execution	Done manually by QA testers	Done automatically using automation tools and scripts
Test Efficiency	Time-consuming and less efficient	More testing in less time and greater efficiency
Types of Tasks	Entirely manual tasks	Most tasks can be automated, including real user simulations
Test Coverage	Difficult to ensure sufficient test coverage	Easy to ensure greater test coverage

86.When to perform smoke testing, sanity testing, regression testing & retesting?

Sanity testing is usually performed after receiving a fairly stable software build or sometimes when a software build might have undergone minor changes in the code or functionality. It decides if end to end testing of a software product shall be carried out further or not.

87.Can Automation testing replace manual testing completely?

Automation testing will not replace manual testing. You need both manual and automation testing.

Manual testing handles complex test cases, while automated testing handles simpler, more repetitive tests.

So, manual testing is still important. But adding automated testing makes your manual tests more efficient.

88.Advantages of manual testing?

- Uses human intelligence to find errors
- Lets testers focus on complex features and functions
- Tester knowledge of the project
- Detects errors outside of the code
- Provides accurate emulation of user experience
- It helps maintain a testable system.

89.what are the testing techniques used in project?

- Equivalence Partitioning. In equivalence partitioning, the input of a program is divided into classes
- Boundary Value Analysis
- Decision Table Testing
- Exploratory Testing
- Experienced Based Testing
- Use Case Testing
- Check List Based Testing
- Risk-Based Testing.

90.Write Test Scenarios and Test Cases for a Login page with {UN,PW,Submit& Refresh button}?

91.what is Usability Testing?

Usability Testing also known as User Experience (UX) Testing, is a testing method for measuring how easy and user-friendly a software application is. A small set of target end-users, use software application to expose usability defects. Usability testing mainly focuses on user's ease of using application, flexibility of application to handle controls and ability of application to meet its objectives.

This testing is recommended during the initial design phase of SDLC, which gives more visibility on the expectations of the users.

92.What is GUI Testing?

Graphical User Interface Testing (GUI) Testing is the process for ensuring proper functionality of the graphical user interface (GUI) for a specific application.GUI testing generally evaluates a design of elements such as layout, colors and also fonts, font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links, and content. GUI testing processes may be either manual or automatic and are often performed by third-party companies,

rather than developers or end users.

93.what is Agile?

Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches.

The four Agile Manifesto values are:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan.

94.Expalin Tester role in Agile Methodology ?

1. Understanding functionality of feature.
2. work closely with the development staff in all phases of development.
3. Communicate more with developers and end-users
4. Participating in preparing Test Plans.
5. Preparing Test Scenarios.
6. Preparing Test Cases for the product.
7. Preparing Test Data?s for the test cases.
8. Preparing Test Environment to execute the test cases.
9. Analyzing the Test Cases prepared by other team members.
10. Executing the Test Cases.
11. Defect Tracking.
12. Giving mandatory information of a defect to developers in order to fix it.
13. Retesting the fixed bugs to check for existence and to check for its effect.
14. Preparing Suggestion Documents to improve the quality of the application.

95.What is Test Closure?

Test Closure is a document that is prepared prior to formally completing the testing process. This memo contains a report of test cases executed, passed,failed and number of defects found, fixed, re-tested, closed.We will provide the path of all the required documents to client like test scenarios, test cases, test execution and bug reporting, based on this report it will decide to stop or continue the testing

96.Diffirence between waterfall and agile?

Below is the difference between Agile and Waterfall software models in the tabular form for better understanding:

Agile	Waterfall
1.Sprints are used to break down the project into manageable pieces.	In general, the methodology gets separated into several stages.
2.Agile models follow the concept of consistent growth during the project itself so that afterward it reduces the risk of completing the requirements.	The waterfall technique is a sequential design process.
3.The model is known for its versatility.	This methodology is a systematic developing method that may be rather rigorous at times.
4.Agile may be thought to be a compilation of several outlines.	In this, the software gets developed as a specific outline.
5.This method is flexible and allows changes for the progress of the project soon after the first stage of designing gets completed.	Once the project development begins, there is no way to change the specifications if any are required.
Because this method uses an incremental development approach, the phases of design, production, testing, and other project management may appear several times.	All the development phases of the project are accomplished once
7.This method is a software advancement technique in which you can change or develop the demands over time.	Appropriate for specialized requirements and unforeseen changes.
8.Testing occurs with software development simultaneously.	The "Testing" is done after the "Build" step.
9.It gives a production attitude wherein the software product fulfills the demands and adapts to the customer's expectations	It is entirely focused on completing the project.
10.It goes perfectly with Duration and Resources or non-fixed finance.	By obtaining risk agreement at the start of the process, it is possible to reduce risk in firm fixed price contracts.
11.It is preferable to work with small, focused teams with a higher level of cooperation.	A team's capacity to coordinate and synchronize is severely restricted.
12.The product manager and his colleagues prepare a list of lacks every day over the development of a project.	In this, the Business analyst develops thespecifications before the beginning of a project.

97.Explain steps of agile approach?

- Project planning
- Product roadmap creation
- Release planning
- Sprint planning
- Daily stand-ups
- Sprint review and retrospective.

98.What are advantages & drawbacks of agile?

Advantages:

- Testers are involved from the beginning.
- Agile testing allows for early testing.
- Agile testing saves time and money.
- Testers have more time to write test cases and implement the test cases.
- Testing is not able to be skipped.
- Cost of defects are reduced greatly.

Drawbacks:

- Poor resource planning
- Limited documentation
- Fragmented output
- No finite end
- Difficult measurement.

99.how to perform multi browser testing?

if we are using Selenium WebDriver, we can automate test cases using Internet Explorer, FireFox, Chrome, Safari browsers.To execute test cases with different browsers in the same machine at the same time we can integrate Testng framework with Selenium WebDriver.

Cross Browser Testing is a type of functional test to check that your web application works as expected in different browsers.

100.challenges faced in agile?

- Changing Requirements
- Not Enough Information
- Continuous Testing
- Technical Skills
- Frequent Regression Cycles
- Lack of Communication
- No Quality Measurement.