Day-2

Date: 06/08/2024 **Introduction to Software Development Lifecycle**

Agile Manifesto

Agile Values:

The four values of the Agile Manifesto

The Agile Manifesto consists of four key values:

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

Agile Principles:

The following 12 Principles are based on the Agile Manifesto.

1. Our highest priority is to satisfy the customer through the early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity–the art of maximizing the amount of work not done–is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.

12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Azile 5 Ceremonies:

The five scrum ceremonies are typically carried out in the following sequence:

- Sprint planning.
- Daily scrum.
- Sprint review.
- Sprint retrospective.
- Backlog refinement.

JIRA

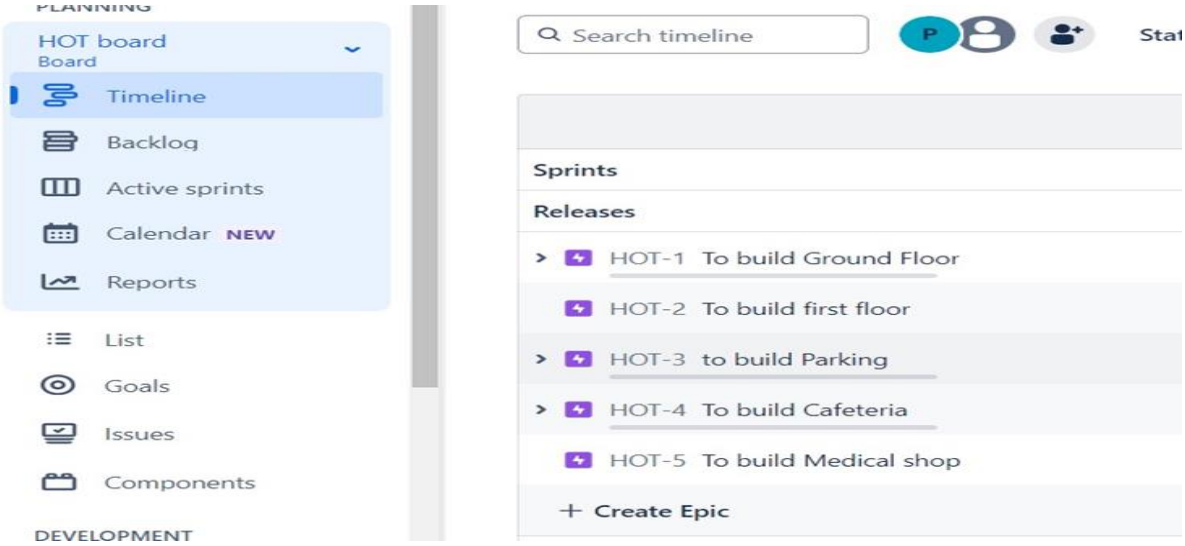


Image-1: Create a project with TimeLine

Image-2: Build backlog

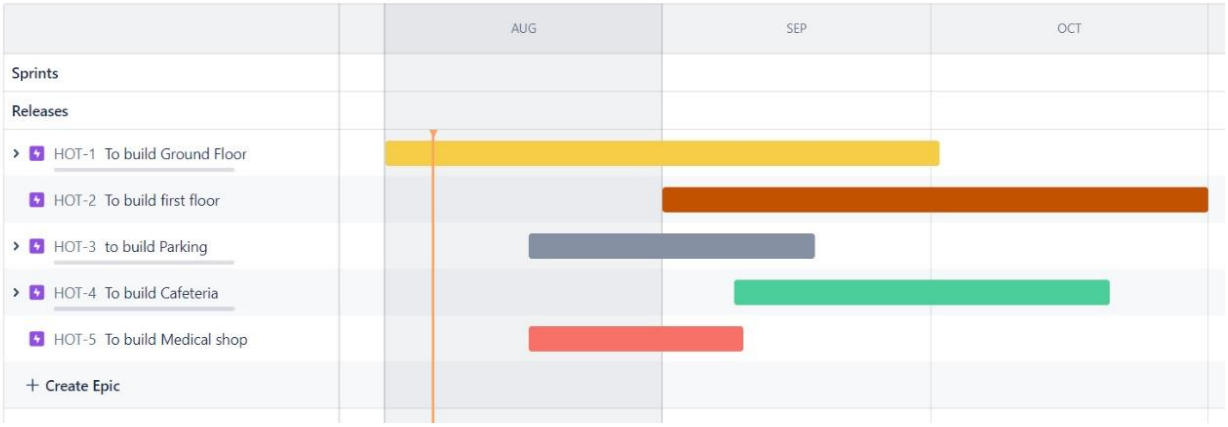


Image-3: Set Time Line

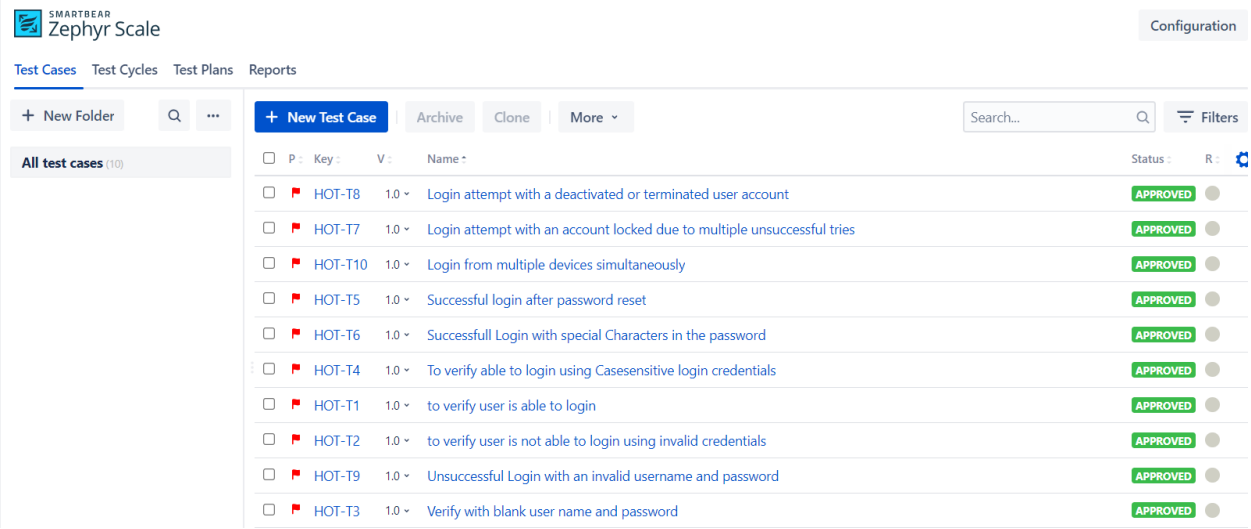


Image-4: Create 10 Test Cases

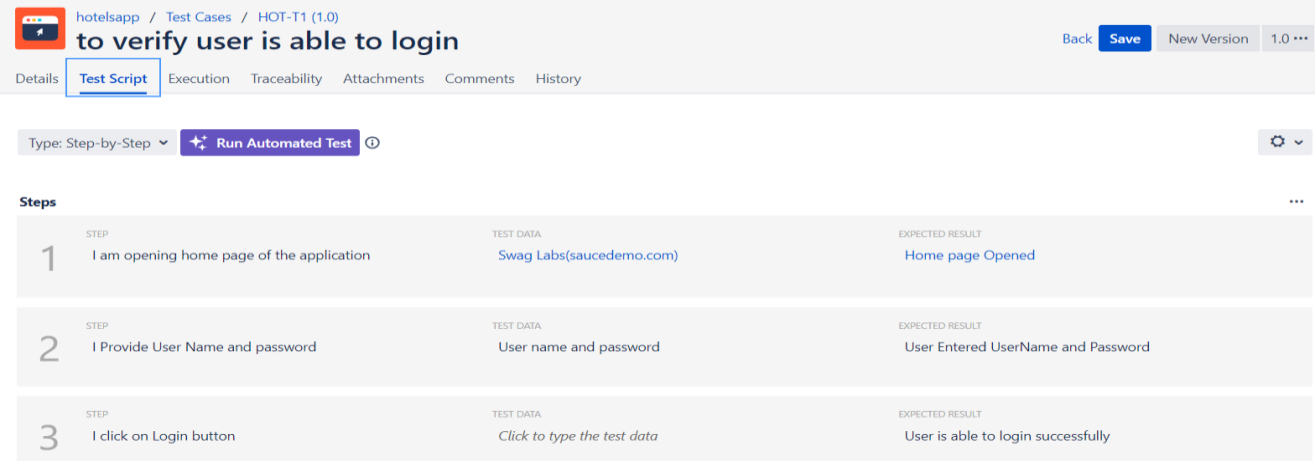


Image-5: Test Script

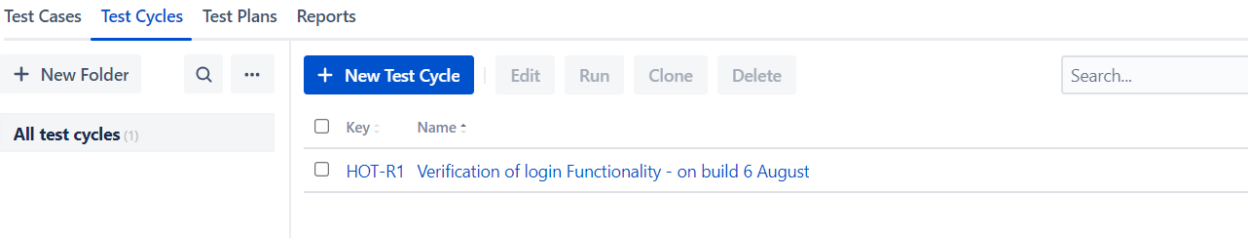


Image-6: Create New Test Cycle

Image-7: Add test cases to test Cycle

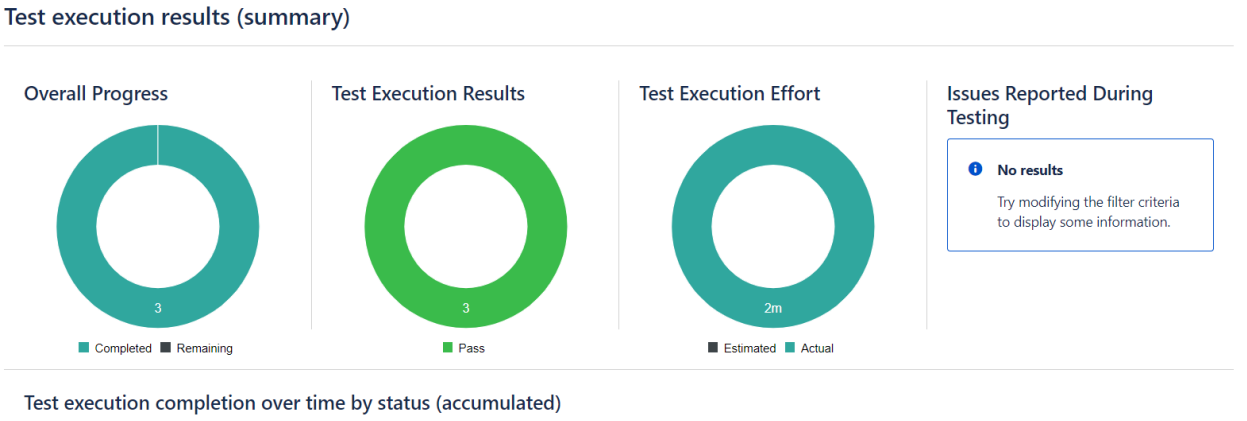


Image-8: Summary Results

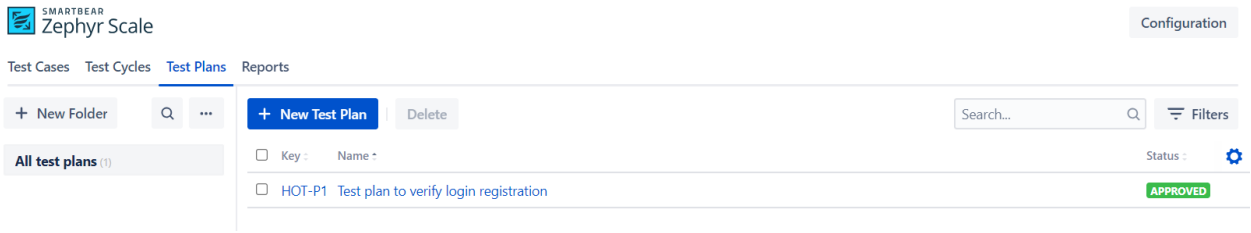


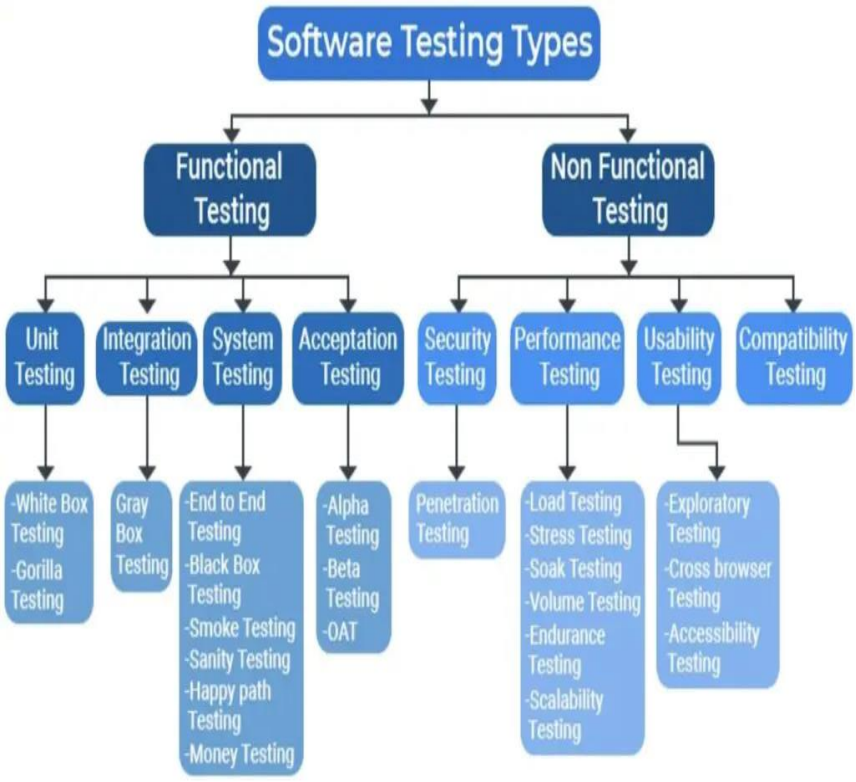
Image-9: Creating a TestPlan

Testing:

Software testing is the process of evaluating and verifying that a software product or application does what it's supposed to do.

The benefits of good testing include preventing bugs and improving performance. Verify and validate application quality to ensure it meets user requirements.

Types of Testing:



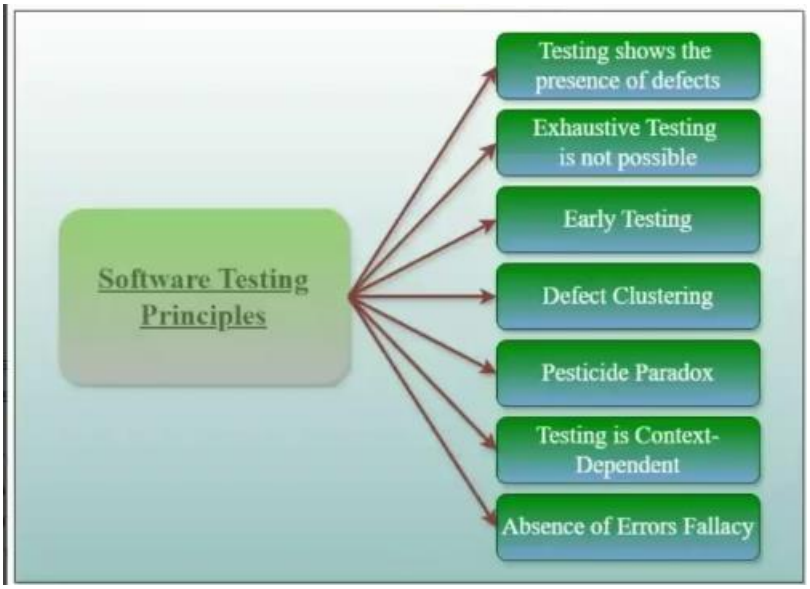


Fig: Seven Software Testing Principles

Assignement:

1.What is the difference between Software test planning and software test strategy?

Ans.

**→A Test Plan outlines the approach, scope, objectives, resources, and schedule for testing a specific project or product.

→A Test Strategy defines the high-level approach to testing for an organization or a project, guiding the overall testing process.

**→A test plan can be defined as a document for a software project that defines the approach, scope, and intensity of the effort of software testing.

→The test strategy is a set of instructions or protocols that explain the test design and determine how the test should be performed.

**→A test plan can be changed.

→While the test strategy can't be changed.

**→The test plan happens independently.

→While test strategy is often found as a part of a test plan.

**→A test plan has the essential objective of how to test when to test, and who will confirm it.

→While test strategy has the essential objectives of what approach to pursue and which module to check.

2.What are the different levels of Software Testing available?

Ans. In general, mainly four levels of testing in software testing:

Unit Testing,

System Testing,

Integration Testing and

Acceptance Testing.

3.What is the difference between webapp and website?

Ans. A website is a series of webpages that are typically accessed through a web browser and are hosted on a web server.

A web application is a software program that is accessed through a web browser and runs on a web server.

The main difference is that a website is static and a web application is dynamic.

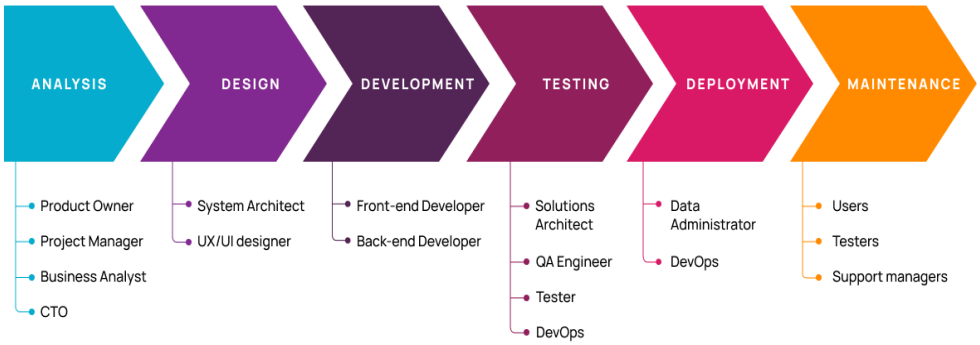
Assignment 1:

SDLC overview: Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment, highlighting the importance of each phase and how they interconnect.

1. **What is Software Development Life Cycle (SDLC)?**

SDLC is a process followed for software building within a software organization. SDLC consists of a precise plan that describes how to develop, maintain, replace, and enhance specific software. The life cycle defines a method for improving the quality of software and the all-around development process.

6 Phases of the Software Development Life Cycle



Stage 1: Requirements Analysis

- To understand and document what stakeholders need from the software. This includes functional and non-functional requirements.

Key Activities:

- **Stakeholder Interviews**: Engaging with end-users and clients to gather detailed needs.
- **Requirement Analysis**: Evaluating and prioritizing requirements to ensure they align with business goals.
- **Documentation**: Creating detailed requirement specifications and use cases.

Importance:

- Sets clear expectations and objectives for the project. Accurate requirements reduce the risk of project failure and scope creep.

Stage-2: Defining Requirements

- To clearly define and document what stakeholders need from the software. This phase ensures all requirements are thoroughly understood and agreed upon before moving forward.

Key Activities:

- Requirement Elicitation: Gathering initial requirements from stakeholders through interviews, surveys, and observation.
- Requirement Analysis: Breaking down and evaluating the gathered requirements to ensure they are complete, feasible, and aligned with business goals.
- Requirement Specification: Creating detailed documentation that includes functional, non-functional, business, and technical requirements.
- Validation: Reviewing and validating the documented requirements with stakeholders to confirm accuracy and completeness.

Importance:

- Sets a clear foundation for all subsequent phases. Well-defined requirements reduce the risk of scope creep and project failures by ensuring that all stakeholders have a shared understanding of what the project will deliver.

Satge-3: Design

- To develop the actual code and integrate components based on the design specifications.

Key Activities:

- **Coding**: Writing code according to design documents and coding standards.
- **Integration**: Combining various software components and ensuring they work together as intended.
- **Code Review**: Regular reviews of the codebase to ensure adherence to standards and practices.

Importance:

- This phase brings the software to life. Effective coding and integration are crucial for creating functional and high-quality software.

Stage-4: Implementation

- To develop the actual code and integrate components based on the design specifications.

Key Activities:

- **Coding**: Writing code according to design documents and coding standards.
- **Integration**: Combining various software components and ensuring they work together as intended.
- **Code Review**: Regular reviews of the codebase to ensure adherence to standards and practices.

Importance:

- This phase brings the software to life. Effective coding and integration are crucial for creating functional and high-quality software.

Stage-5: Testing

- To verify that the software meets the specified requirements and is free of defects.

Key Activities:

- **Unit Testing**: Testing individual components for correctness.
- **Integration Testing**: Ensuring that different components work together properly.
- **System Testing**: Verifying the entire system's functionality and performance.
- **User Acceptance Testing (UAT)**: Confirming that the software meets end-user needs and requirements.

Importance:

- Identifies and fixes defects before deployment, ensuring the software is reliable and meets quality standards.

Stage-5: Deployment

- To release the completed software to users and ensure it is fully operational in the production environment.

Key Activities:

- **Deployment Preparation**: Finalizing deployment procedures, documentation, and training materials.
- **Go-Live**: Installing and configuring the software in the production environment.

- **Post-Deployment Support**: Addressing any issues that arise after deployment and providing user support.

Importance:

- Marks the official release of the software to users. Proper deployment ensures smooth transition and operational stability.

How They Interconnect

- **Defining Requirements → Design**: Clear requirements guide the design phase.
- **Design → Implementation**: The design provides a blueprint for development.
- **Implementation → Testing**: The developed software is tested to ensure it meets the design and requirements.
- **Testing → Deployment**: Validated software is deployed to users.
- **Deployment → Feedback → Defining Requirements**: User feedback can refine or introduce new requirements, starting a new cycle.