BITS Pilani presentation

BITS Pilani
Pilani Campus

Dr. Nagesh BS

innovate achieve lead

BITS Pilani
Pilani Campus

SE ZG501
Software Quality Assurance and Testing
Lecture No. 6

Test Plan

1. **Objectives** – **Defines the purpose of testing and what it aims to achieve**.
   - *Example:* Ensuring that the **login functionality works correctly** for all user roles.
2. **Scope** – **Specifies what will be tested and what will not be covered.**
   - *Example:* Testing only the web version of an application, excluding mobile apps.
3. **Test Items** – **Identifies the components, modules, or features to be tested**.
   - *Example:* Testing the **checkout process in an e-commerce application**.
4. **Test Environment** – **Defines the hardware, software, and network configurations required for testing**.
   - *Example:* A test server running **Windows 11 with a MySQL database**.
5. **Testing Strategy** – **Describes the approach, techniques, and levels of testing to be used**.
   - *Example:* Using automated regression testing and manual exploratory testing.

Test Plan

6. **Test Schedule** – **Specifies timelines, milestones, and deadlines for test execution.**

- *Example:* Unit testing from March 1–5, system testing from March 6–10.

7. **Resource Requirements** – Lists the **personnel, tools, and infrastructure needed for testing.**

- *Example:* Requiring 3 testers, Selenium automation tool, and a dedicated test server.

8. **Risk Management** – **Identifies potential risks that could impact testing and mitigation strategies**.

- *Example:* Risk: Delayed development could shorten testing time. Mitigation: Prioritize critical test cases.

Test Plan

9. **Test Deliverables** – **Defines the documents, reports, and output produced during testing.**

- *Example:* Test plan document, defect report, test summary report.

STP Template

1. Scope of the Tests

Defines what will be tested, including:

- The software package (name, version, and revision).
- Relevant documents supporting the test plan.
- **2. Testing Environment**
- Specifies the setup needed for testing:
- Testing locations/sites.
- Hardware and firmware requirements.
- Involved organizations and manpower needs.
- Training and preparation for the test team.
- **3. Test Details (for each test)**
- Includes specific details of each test case:
- **Test identification & objective** – Defines the purpose of the test.
- **Reference documents** – Links to design and requirement documents.

STP Template

- **Test class & level** – Identifies whether it's unit, integration, or system testing.
- **Test case requirements** – Lists conditions for executing test cases.
- **Special requirements** – Factors like response time, security, or performance metrics.
- **Data recording** – Information to be logged during tests.
- **4. Test Schedule**
- Provides estimated time for various testing phases:
- **Preparation** – Setting up resources and test environments.
- **Testing** – Executing test cases.
- **Error correction** – Debugging and fixing detected issues.
- **Regression testing** – Retesting to ensure fixes do not introduce new defects.
- This structured approach ensures a **systematic and well-documented** testing process for software projects.

Software Test Descriptions

1. **Scope of the Tests**

- Defines the **software package** to be tested (name, version, revision).
- Lists the **documents** that form the basis of the test design.
- **2. Test Environment (for each test)**
- Identifies the **test case details** (linked to the Software Test Plan - STP).
- Describes the **operating system and hardware configuration** required.
- Provides **instructions for software loading**.
- **3. Testing Process**
- Step-by-step **input instructions** for the test.
- Specifies **data to be recorded** during the test execution.
- **4. Test Cases (for each case)**
- Includes **test case identification details**.
- Lists **input data and system settings**.
- Defines **expected intermediate results** (if applicable).
- Specifies **expected final results** (numerical, messages, system behavior, etc.).

Software Test Descriptions

- **5. Actions to Be Taken in Case of Program Failure/Cessation**
- Defines contingency actions when failures occur.

  6. Procedures to Be Applied According to the Test Results Summary
- Describes the **post-test procedures**, including result evaluation and reporting.

This template helps ensure **consistency and completeness** in software testing documentation.

Types of Reviews

The diagram illustrates the **types of reviews used during the software development cycle**, ensuring quality assurance at each stage. Below is a breakdown of the process:

1. Stages of Software Production

- **System Specification and Design:** Initial phase where system requirements and design
- are specified.
- **Requirements Specification:** Defines what the system should do.
- **Preliminary Design:** Outlines the system architecture and basic design.
- **Programming Phase:** Involves detailed design, coding, unit testing, and integration.
- **Validation Phase:** Ensures that the developed software meets the specified requirements.
- **System Integration and Validation:** Final stage where all components are tested together.

Types of Reviews

- **2. Types of Reviews at DiCerent Phases**
- **Project Launch Review:** Conducted at the beginning to ensure a proper start.
- **End of Phase Reviews:** At key milestones (requirements, design, coding, validation).
- **End of Project Review:** Final review after project completion.
- **3. Quality Control & Assurance Mechanisms**
- **SQAP (System Quality Assurance Plan)** writing is the process of creating a structured
- document that defines **quality assurance (QA) activities** for a software project
- **Document Reviews:** Verifying correctness and completeness of written documents.
- **Inspections:** Detailed examination of design/code to detect defects.
- **Metrology:** Ensuring software measurements and quality metrics are maintained.

Types of Reviews

- **Audits:** Formal evaluations of the processes followed.
- **Project Reviews:** Ongoing evaluation of project status, risks, and issues.
- This structured approach helps ensure **software quality, early defect detection, and compliance** with standards throughout the development lifecycle.

THANK YOU