MEKELLE UNIVERITY

EITM-School of computing

Software engineering

Software Testing and Quality Assurance – Test Planning and Control

Name – Fseha Hagos

Id – UGR/170040/12

Table of Contents

[Software Testing and Quality Assurance – Test Planning and Control 1](#_Toc187933904)

[1. Testing Planning and Control 1](#_Toc187933905)

[1.1 Test Planning 1](#_Toc187933906)

[1.1.1 Objective: 1](#_Toc187933907)

[1.1.2 Components of a Test Plan 1](#_Toc187933908)

[Introduction: 1](#_Toc187933909)

[Scope of Testing: 1](#_Toc187933910)

[Test Objectives: 1](#_Toc187933911)

[Test Strategy: 2](#_Toc187933912)

[Test Deliverables: 2](#_Toc187933913)

[Test Data 3](#_Toc187933914)

[Test Reports 3](#_Toc187933915)

[Defect Logs 3](#_Toc187933916)

[Entry and Exit Criteria: 3](#_Toc187933917)

[Resources: 3](#_Toc187933918)

[Schedule: 3](#_Toc187933919)

[Risk Management: 3](#_Toc187933920)

[Approval: 3](#_Toc187933921)

[1.1.3 Tools for Test Planning: 4](#_Toc187933922)

[1.2 Test Control 4](#_Toc187933923)

[1.2.1 Objectives: 4](#_Toc187933924)

[1.2.2 Activities: 4](#_Toc187933925)

[1.2.3 Tools for Test Control: 4](#_Toc187933926)

[1.2.4 Deliverables: 4](#_Toc187933927)

[2. Testing Analysis and Design 5](#_Toc187933928)

[2.1 Test Analysis 5](#_Toc187933929)

[2.1.1 Objectives: 5](#_Toc187933930)

[2.1.2 Activities: 5](#_Toc187933931)

[2.2 Test Design 5](#_Toc187933932)

[2.2.1 Objectives: 5](#_Toc187933933)

[2.2.2 Activities: 5](#_Toc187933934)

[2.2.3 Deliverables: 5](#_Toc187933935)

[2.2.4 Tools for Analysis and Design: 6](#_Toc187933936)

[Final Deliverables: 6](#_Toc187933937)

# Software Testing and Quality Assurance – Test Planning and Control

## 1. Testing Planning and Control

## 1.1 Test Planning

## 1.1.1 Objective:

The objective is to ensure the quality and reliability of the Student Registration System, define the scope of testing, and allocate resources efficiently.

## 1.1.2 Components of a Test Plan

## Introduction:

- The system aims to streamline student registration with an emphasis on ease of use and efficiency.

- Testing focuses on functionality, reliability, and usability, identifying potential defects.

- Stakeholders include:

- Developers: Responsible for system development.

- Testers: Tasked with defect detection.

## Scope of Testing:

- Key areas: user registration, Chapa integration (success/failure), admin dashboard functionality, authentication, and data security.

- Non-functional aspects: response time and load handling.

- Exclusions: Testing beyond Chapa integration, as third-party APIs are independently verified.

## Test Objectives:

- Verify proper functioning of the user registration process.

- Ensure secure and reliable payment processing.

## Test Strategy:

- Testing Levels:

- Unit Testing: Validations, API calls, individual functions.

- Integration Testing: Interaction between Django views and Chapa module.

- System Testing: Registration-to-payment workflow.

- Acceptance Testing: Compliance with user requirements.

- Testing Types:

- Manual Testing: Scenarios requiring critical analysis.

- Automated Testing: Selenium (UI), Pytest (backend), load testing for concurrent processes.

- Test Environment Requirements:

- Staging environment replicating production, including database and payment API.

## Test Deliverables:

- Test cases/scripts

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario** | **Description** | **Preconditions** | **Steps** | **Expected Results** | **Priority** |
| TC\_REG\_001 | Successful Registration | Validate registration with valid inputs. | Registration form is accessible. | 1. Open form. 2. Enter valid data. 3. Submit form. | Registration completes. Confirmation email sent. Redirect to payment page. | High |
| TC\_REG\_002 | Invalid Email Validation | Test handling of invalid email inputs. | Registration form is accessible. | 1. Open form. 2. Enter invalid email. 3. Submit form. | Error: "Enter a valid email." Registration fails. | High |
| TC\_PAY\_001 | Successful Payment via Chapa | Confirm Chapa processes payment successfully. | User logged in. Chapa ready. | 1. Login. 2. Register. 3. Enter valid payment info. 4. Confirm payment. | Payment completes. Receipt email sent. | High |

### Test Data

* Valid and invalid email formats.
* Payment scenarios (valid and invalid inputs).

Test Reports Key statistics such as pass and fail rates will be documented.

Defect Logs Defects will be categorized by severity: high, medium, and low.

## Entry and Exit Criteria:

- Entry: Finalized requirements, completed development, prepared test environment.

- Exit: Resolved critical defects, 95% functional coverage.

## Resources:

- Team: Two me and my classmate.

## Schedule:

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Start Date | End Date | Responsible |
| Test Case Creation | Jan 9, 2025 | Jan 16, 2025 | Developer |
| Environment Setup | Jan 11, 2025 | Jan 17, 2025 | Developer |
| Test Execution | Jan 22, 2025 | Jan 26, 2025 | Team Member |
| Defect Logging/Fixing | Jan 18, 2025 | | Jan 29, 2025 | Developer |
| Final Test Report | Jan 32, 2025 | Jan 32, 2025 | Developer |

## Risk Management:

- Risks: Chapa API downtime, data loss due to network issues.

- Mitigation: API fallback mechanisms, frequent backups.

## Approval:

- Reviewed and approved by Instructor.

## 1.1.3 Tools for Test Planning:

- Documentation: Microsoft Word.

- Test Management: Jira.

## 1.2 Test Control

## 1.2.1 Objectives:

- Ensure testing aligns with the plan.

- Adapt to scope or requirement changes.

- Provide transparency to stakeholders.

## 1.2.2 Activities:

1. Monitoring:

- Track execution progress, defect trends, and adherence to schedule.

2. Reporting:

- Status updates, test coverage metrics, and risk summaries.

3. Issue Management:

- Address delays, reallocate resources, and resolve conflicts.

4. Change Management:

- Revise plans for requirement changes, ensure version control.

5. Defect Tracking:

- Prioritize defects, ensure timely resolutions.

## 1.2.3 Tools for Test Control:

- Test Management Tools: Jira.

- Defect Tracking Tools: Integrated in test management.

## 1.2.4 Deliverables:

- Test plan document.

- Metrics reports.

- Final test report.

## 2. Testing Analysis and Design

## 2.1 Test Analysis

## 2.1.1 Objectives:

- Identify test conditions based on requirements.

- Ensure comprehensive coverage.

- Maintain traceability between requirements and test cases.

## 2.1.2 Activities:

1. Review requirements for ambiguities.

2. Identify and categorize requirements (functional, non-functional).

3. Derive high-level test conditions.

4. Prioritize based on risk and complexity.

## 2.2 Test Design

## 2.2.1 Objectives:

- Create detailed, reproducible test cases.

- Define test data requirements.

## 2.2.2 Activities:

1. Develop test cases using clear templates:

- Test Case ID, Description, Preconditions, Steps, Expected Results, Priority.

2. Design test data for valid, invalid, and edge cases.

3. Set up test environments to replicate production.

4. Automate where applicable (e.g., Selenium, Pytest).

5. Create a Requirements Traceability Matrix (RTM).

## 2.2.3 Deliverables:

- Test scenarios and cases.

- Test data.

- RTM.

## 2.2.4 Tools for Analysis and Design:

- Management: Jira.

- Documentation: Microsoft Word.

- Automation: Selenium, Pytest.

## Final Deliverables:

- Comprehensive test report covering results, challenges, and readiness for deployment.