**Mekelle University**



EIT-M

School Of Computing

Software Architecture and design

Project

Name: Id.No

1. Dawit G/egziabiher Ugr/172121/12

**Test Plan for Student Registration System**

**1. Test Plan ID**

**STP-01**

**2. Introduction**

The purpose of this document is to outline the test plan for the Student Registration System, including test objectives, scope, testing approach, resources, schedule, and deliverables. The system allows users to manage students, courses, registrations, schedules, progress tracking, payments, notifications, and reports. The frontend is built using React, and the backend uses Express.js with MongoDB.

**3. Test Objectives**

The primary objective of this test plan is to verify the end-to-end functionality of the **student registration system** by ensuring that all requests made from the React frontend are correctly processed and handled by the backend, with appropriate responses returned for every scenario. This includes testing CRUD operations for each backend module, validating role-based access and authentication, and checking the correct interaction between the frontend and backend components.

**4. Scope of Testing**

**4.1 Features to Be Tested**

* **User Management**: Create, update, delete, and retrieve user profiles.
* **Course Management**: CRUD operations for courses.
* **Registration Management**: Register users for courses.
* **Payment Management**: Record and track payments.
* **Progress Management**: Record and view user progress.
* **Notifications**: CRUD operations for notifications.
* **Reports**: Generate, update, and delete reports.
* **Schedules**: Manage course schedules.
* **Authentication**: Login functionality and token validation.

**4.2 Features Not to Be Tested**

* External libraries and frameworks, unless customization is involved.
* Low-priority UI features like animations or minor style inconsistencies.

**5. Testing Approach**

**5.1 Types of Testing**

1. **Unit Testing**:
   * Validate individual modules (controllers and routes).
   * Tools: Mocha, Chai.
2. **Integration Testing**:
   * Test API endpoints and ensure frontend-backend communication works.
   * Tools: Postman, Jest.
3. **User Acceptance Testing (UAT)**:
   * Validate the system meets user expectations.
4. **Regression Testing**:
   * Ensure new changes don’t break existing functionality.

**6. Testing Deliverables**

* **Test Plan**: This document.
* **Test Cases**: Documented in a test management tool or spreadsheet.
* **Bug Reports**: Created in the issue-tracking system (e.g., Jira).
* **Test Summary Report**: A report summarizing testing activities and results.

**7. Test Environment**

**7.1 Hardware Requirements**

* **Frontend Testing**: Devices (desktop, mobile, tablets) with various screen sizes.
* **Backend Testing**: Server with Node.js runtime and MongoDB.

**7.2 Software Requirements**

* Operating System: Windows 10/11, macOS, Ubuntu.
* Browsers: Chrome, Firefox, Edge, Safari.
* Tools: Postman, Mocha, Chai, Selenium, JMeter.

**7.3 Test Data**

* User profiles, course details, payment records, progress data, and reports.

**8. Testing Schedule**

| **Phase** | **Start Date** | **End Date** | **Tasks** |
| --- | --- | --- | --- |
| Test Plan Creation | Day 1 | Day 3 | Prepare the test plan and strategy. |
| Unit Testing | Day 4 | Day 10 | Write and execute unit tests. |
| Integration Testing | Day 11 | Day 15 | Test API integration and frontend-backend. |
| Functional Testing | Day 16 | Day 20 | Validate user flows and CRUD operations. |
| Performance Testing | Day 21 | Day 23 | Test system scalability and load handling. |
| Security Testing | Day 24 | Day 26 | Check vulnerabilities. |
| Regression Testing | Day 27 | Day 28 | Ensure no new bugs are introduced. |
| UAT | Day 29 | Day 30 | Perform user acceptance testing. |

**9. Entry and Exit Criteria**

**9.1 Entry Criteria**

* Requirements are finalized.
* Development is complete and code is stable.
* Test environment is set up.

**9.2 Exit Criteria**

* All test cases executed.
* Critical and high-priority defects are resolved.
* Test summary report approved by stakeholders.

**10. Risks and Mitigation**

| **Risk** | **Likelihood** | **Impact** | **Mitigation** |
| --- | --- | --- | --- |
| Unavailability of test environment | Medium | High | Use cloud-based environments or local setups. |
| Delayed delivery of features | Medium | Medium | Regular communication with the development team. |
| Limited time for regression testing | High | High | Prioritize test cases for critical features. |
| Incomplete test data | Low | Medium | Collaborate with developers to create comprehensive datasets. |

**11. Roles and Responsibilities**

| **Role** | **Responsibility** |
| --- | --- |
| QA Lead(Self) | Prepare and manage the test plan. |
| Test Engineers(Self) | Write, execute, and report test cases. |
| Developers(Self) | Fix reported bugs and provide testable builds. |
| Product Owner(self) | Approve UAT and provide feedback on testing results. |

**12. Tools**

* **Testing Frameworks**: Mocha, Chai
* **API Testing**: Postman.

**13. Test Automation**

* **Scope**: Automate regression tests, API tests, and repetitive functional tests.
* **Tools**: Cypress for UI automation, Mocha for backend tests.
* **Strategy**: Develop reusable test scripts for all critical features.

**14. Approval**

| **Stakeholder** | **Signature** | **Date** |
| --- | --- | --- |
| QA Lead |  |  |
| Product Owner |  |  |
| Project Manager |  |  |