MEKELLE UNIVERSITY

Software Engineering Department

SRS for Student Registration System Built in Laravel Framework

Name: Solomon Tsige

ID: Eitm/ur170246/12

Software Requirements Specification (SRS) for Student Registration System

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description of the Student Registration System. It outlines the system's functionality, requirements, and constraints to ensure that all stakeholders have a clear understanding of the system.

1.2 **Scope**

The Student Registration System is a web-based application that allows educational institutions to manage student registrations efficiently. The system will enable users to add, edit, view student records, as well as process payments through PayPal.

1.3 Audience

This document is intended for:

Developers

Testers

Stakeholders

2. Overall Description

2.1 Product Perspective

The Student Registration System will be a standalone web application built using the Laravel framework. It will interact with a MySQL database to store student information and payment records.

2.2 Product Functions

User authentication (registration and login)

Student self management information (add, edit, view)

Payment processing via PayPal

Data validation and error handling

2.3 User Classes and Characteristics

Student: Registers for courses and makes payments.

3. Functional Requirements

3.1 User Authentication

- The system shall allow users to register with email, and password.
- The system shall allow users to log in using their credentials.
- The system shall provide password recovery options.

3.2 Payment Processing

- The system shall integrate with PayPal for payment processing.
- The system shall allow students to make payments for registration.
- The system shall handle payment success and failure scenarios.

3.3 Data Validation

- The system shall validate all input data before processing.
- The system shall provide error messages for invalid input.

4. Non-Functional Requirements

4.1 Performance

- The system shall handle more than 100 or actually depends on the localhost storage and performance degradation.
- The system shall respond to user actions within 10 seconds.

4.2 Security

- The system shall encrypt user passwords.
- The system shall use HTTPS for secure data transmission.

4.3 Usability

- The system shall have an intuitive user interface.

4.4 Reliability

- The system shall have an uptime of 99.5%.
- The system shall back up data daily.

5. Use Cases

5.1 Use Case: User Registration

Actors: Student

- ✓ Preconditions: user is on the registration page.
- ✓ Post-conditions: user account is created.

Main Flow:

- ✓ User enters registration details (email, password).
- ✓ User submits the registration form.
- ✓ System validates the input.
- ✓ System creates a new user account.
- ✓ System redirects the user to the login page.

Main Flow:

- ✓ System displays the student's current details in an editable form.
- ✓ System validates the input.
- ✓ System updates the student record in the database.

System displays a success message and redirects to the student list.

5.5 Use Case: Make Payment

Actors: Student

Preconditions: Student is logged in and has a valid registration.

Postconditions: Payment is processed.

Main Flow:

Student navigates to the payment page.

Student reviews the registration details and amount due.

Student clicks the "Make Payment" button.

System redirects the student to PayPal for payment processing.

Student completes the payment on PayPal.

system displays a success message.

6. System Architecture

The Student Registration System will follow a Model-View-Controller (MVC) architecture. The architecture will consist of the following components:

Model: Represents the data and business logic (e.g., Student, User, Payment models).

View: Represents the user interface (e.g., Blade templates for registration, student list, payment).

Controller: Handles user requests and interacts with the model and view (e.g., StudentController, PaypalController).

7. Assumptions and Dependencies

The system assumes that users have access to a stable internet connection.

The system depends on the Laravel framework and its ecosystem (e.g., Composer, MySQL).

The system relies on the PayPal API for payment processing, which requires valid API credentials.

Conclusion

This Software Requirements Specification (SRS) document outlines the essential features, requirements, and use cases for the Student Registration System. It serves as a guide for developers, testers, and stakeholders to ensure that the system meets the intended goals and provides a seamless experience for users.

MEKELLE UNIVERSITY

Software Engineering Department

Test Planning for Student Registration System Built in Laravel Framework

Name: Solomon Tsige

ID: Eitm/ur170246/12

Test Planning for Student Registration System

- ***Objectives:-*** *The primary objectives of this test plan is to ensure that the system of student registration system function as intended, meet the specified requirement and provide user interface.The testing validate the functionality of the course registration, payment processing and student profile management.*
- ***Scope***

-Functional testing -verification of login and register,payment

-usability interface for user

-performance testing -responsiveness

-security testing-responsiveness

-Regression testing-ensuring the new changes not affect existing function

- ***Test Objectives***

-To verify all functionality

-To identify and fix defects before the system goes to use

-To validate the system can handle expected user load

- ***Test Deliverables***

-Test plan document

-Test case document

-Defect report

- ***Test Strategies***

-Manual testing

-Automated testing

-Performance testing

-Security testing tool

- ***Resource***

Human resource-test manager,developers,analyst

Tools-testing tools eg. Selenium for automation testing

- ***Entry and Exit Criteria***

Entry:

-Test case are created and reviewed

-Test data is prepared and available

Exit:

-All the test case have been executed with pass rate at least 95%.

-Stakeholder approval is obtained.

Risk Management

-Risk : incomplete requirement

-Mitigation: required meeting with stakeholder

Schedule

Test planning - maybe 2weeks

Test case devt-3weeks

Test execution-4weeks

Test reporting-1 weeks