MEKELLE UNIVERSITY

ETHIOPIAN INSTITUTE OF TECHNOLOGY - MEKELLE

SCHOOL OF COMPUTING

DEPARTMENT OF SOFTWARE ENGINEERING

Software Testing and Quality Assurance:

Assignment on : Preparing SRS

Name	ID
MIKYAS HAFTAMU	UGR/170241/12

SUBMITTED TO: Inst. Mesele

SUBMITION DATE: Jan 16 2025

Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

The purpose of this document is to outline the functional and non-functional requirements for the Payment Form, which allows users to input their payment information and submit it for processing.

1.2 Scope

This Payment Form will be part of a web application built using Laravel. It will collect user payment details, including first name, last name, email, and amount, and submit this data to a payment processing route.

1.3 Definitions, Acronyms, and Abbreviations

- SRS: Software Requirements Specification
- UI: User Interface
- HTML: Hypertext Markup Language
- CSS: Cascading Style Sheets
- Laravel: A PHP web application framework

2. Overall Description

2.1 Product Perspective

The Payment Form is a standalone UI component within a Laravel web application. It will interact with the backend for payment processing.

2.2 Product Functions

- ❖ Collect user’s first name, last name, email, and payment amount.
- ❖ Validate user input.
- ❖ Display success or error messages based on the payment processing outcome.
- ❖ Submit payment data to the designated route for processing.

2.3 User Classes and Characteristics

- End Users: Individuals making a payment. They should be familiar with basic web forms.
- System Administrators: Users who manage the payment processing backend.

2.4 Operating Environment

- The application will run in web browsers (Chrome, Firefox, Safari) on various devices (desktops, tablets, mobile devices).
- The backend will be powered by a Laravel PHP framework.

2.5 Design and Implementation Constraints

- The form must comply with web accessibility standards.
- All data must be transmitted securely using HTTPS.

2.6 Assumptions and Dependencies

- Users have internet access.
- The backend payment processing system (e.g., Chapa) is operational and accessible.

3. Functional Requirements

3.1 User Input Requirements

- First Name:

Type: Text

Required: Yes

Validation: Must contain alphabetic characters only.

- Last Name:

Type: Text

Required: Yes

Validation: Must contain alphabetic characters only.

- Email:

Type: Email

Required: Yes

Validation: Must be a valid email format.

- Amount:

Type: Number

Required: Yes

Validation: Must be a positive number (greater than 0) and lessthan 100,000.

3.2 Payment Submission

- Action: The form submits a POST request to the pay route.
- CSRF Protection: The form must include a CSRF token for security.

3.3 Success and Error Messages

- Upon successful payment processing, a success message should be displayed.
- If an error occurs during processing, an error message should be displayed.

4. Non-Functional Requirements

4.1 Performance Requirements

- The form should load within 2 seconds on a standard internet connection.

4.2 Security Requirements

- Data must be transmitted securely using HTTPS.
- Input validation must be implemented to prevent XSS and SQL injection attacks.

4.3 Usability Requirements

- The form should be intuitive and easy to navigate.
- Placeholder text should guide the user on the expected input format.

4.4 Compatibility Requirements

- The application should be compatible with modern web browsers and responsive on various screen sizes.

5. User Interface Requirements

5.1 Layout

- The form should be centered on the page with a clean, simple design using Tailwind CSS.
- The form should include labels for each input field.

5.2 Input Elements

- Each input field should have appropriate styling (padding, borders, focus effects).

5.3 Buttons

- The submit button should be clearly labeled "Pay" and styled to be visually distinct.

6. Acceptance Criteria

- The form must validate inputs correctly and provide appropriate feedback.
- The form must successfully submit data to the backend and handle responses accurately.
- All success and error messages must be displayed to the user.