College of Engineering & Management, Kolaghat

Online Fees Payment System

Software Requirements Specification

Team Members:-

Sombit Pramanik
Rohit Pathk
Nabin Maity
Rahul Jana

Table of Content

1.

1. Introduction

1.1 Purpose

As the need for a more secure and advanced solution arises, we aim to design this software to be system-independent, ensuring broad compatibility while prioritizing security and efficiency. This software will be developed with these core requirements in mind.

1.2 Scope

Ease of Use: The interface is intuitive and simple, requiring minimal training.

User-Friendly Experience: Designed with non-technical users in mind, making it accessible for all.

Speed and Efficiency: Transactions are faster compared to traditional systems, minimizing delays.

Enhanced Security: Advanced security measures, including cryptographic protections, are implemented to safeguard sensitive financial data.

1.3 References

IEEE Standard 830-1998; IEEE Recommended Practice for Software Requirements Specifications.

1.4 Overview

This SRS document is organized to provide a comprehensive overview of the system's purpose, functionality, and design. The document begins with high-level sections, offering an introduction and overview of the system.

2. Overall Description

2.1 Product Function & Perspective

The Online Fees Payment System facilitates secure, UPI-based online payments for educational institutions. It allows administrators to manage fee collections and users to conveniently make payments via a web interface. The system supports user registration, payment processing, transaction tracking, and

reporting. With a focus on ease of use and speed, the software ensures a seamless experience for both users and admins.

2.2 User Characteristics

The primary users of the system are individuals as well as corporate who want to receive online payments.

2.3 Constraints

The system must be accessible via modern web browsers and optimized for both desktop and mobile devices.

2.4 Assumptions and Dependencies

It is assumed that users will have internet access and a modern web browser. The system depends on third-party services for authentication and messaging.

3. Functional Requirements

3.1 User Registration and Authentication

The system shall allow users to register using an email address and password. The system shall provide secure authentication using Cryptography Hashing.

3.2 Payment Processing

This system will intend to work as an add-on Fronted by leveraging the power of High speed UPI payment processing

3.3 Security Algorithm

we will implement the Cartographic public key and private key base recognition of users

4. Non-Functional Requirements

4.1 Performance Requirements

The system shall handle up to 10,000 concurrent users.

4.2 Security Requirements

The system shall use encryption to protect user data in transit and at rest. The system shall implement secure authentication and authorization mechanisms.

4.3 Usability Requirements

The system shall be intuitive and easy to navigate for users with basic computer skills. The interface shall be responsive and accessible on mobile devices.

4.4 Reliability Requirements

The system shall have an uptime of 99.9%. The system shall provide error handling and recovery mechanisms.

4.5 Maintainability Requirements

The system shall be modular and well-documented to facilitate future maintenance and updates. The code shall follow standard coding practices.

4.6 Scalability Requirements

The system shall be designed to scale horizontally to accommodate increased user traffic.

5. System Architecture

5.1 System Component

The system consists of a front-end (HTML,CSS,JavaScript), a back-end (Flask written in Python), and a database (MySQL). The front-end communicates with the back-end via RESTful API.

5.2 Database Design

The database shall store user information,Transaction preferences, and Payment data. The schema shall include tables for users, Payment preferences, messages and many more.

6. External Interface Requirements

6.1 User Interfaces

The user interface shall be designed to be intuitive and easy to use, with a focus on accessibility.

6.2 Hardware Interfaces

The system shall be accessible via any device with a web browser, including desktops, tablets, and smartphones.

6. 3 Software Interfaces

The system shall use third-party APIs for authentication (Cartographic public-private authentication) and messaging (Custom Made Email API).

6.4 Communication Interfaces

The system shall use HTTPS for secure communication between the front-end and back-end. Real-time messaging shall be handled via WebSocket.

7. Other Requirements

7.1 Legal and Regulatory Requirements

The system shall comply with data protection regulations such as GDPR. User data shall not be shared with third parties without consent.

7.2 Ethical Requirements

The system shall ensure user privacy and provide transparency regarding data usage. Users shall have control over their personal information.

7.3 Development Environment

The development environment shall include Git for version control, VS Code for development, and Docker for containerization.

8. Appendices

8.1 Glossary

JWT: JSON Web Token, REST: Representational State Transfer, API: Application Programming Interface."

8.2 Use Cases

Use case diagrams illustrating the interaction between users and the system.

8.3 Reference

IEEE Software Engineering Standards, Project Management Documentation.