Ruth Mary Paul (1BM22CS360)

1. Introduction

1.1 Purpose of this Document

The purpose of this document is to provide a detailed outline of the requirements for the Credit Card Processing System. This document serves as a guide for the development team to ensure that the final product meets the needs of both the business and the end-users.

1.2 Scope of this Document

This document covers the functional and non-functional requirements for the Credit Card Processing System. It outlines the features, constraints, and limitations of the system, ensuring the secure and efficient management of credit card transactions.

1.3 Overview

The Credit Card Processing System is a centralized platform designed to manage credit card transactions, including payment processing, statement generation, fraud detection, and credit limit management. Accessible through a web interface, the system integrates with third-party payment gateways and implements advanced security measures to ensure user data protection.

2. General Description

The system will:

- Provide modules for user management, transaction processing, fraud detection, and reporting.
- Be accessible through a secure and intuitive web interface.
- Ensure scalability, reliability, and security to handle high transaction volumes.
- Use the Java programming language and adhere to the MVC architectural pattern.

3. Functional Requirements

3.1 User Management

- Allow merchants to register and log in.
- Enable role-based access control for merchants, administrators, and customers.

3.2 Payment Gateway Integration

- Integrate with various payment gateways for secure transactions.

3.3 Payment Processing

- Support one-time payments, recurring payments, and installment payments.
- Enable transaction processes like authorizations, captures, voids, and refunds.

3.4 Fraud Detection

- Include advanced algorithms for fraud prevention.

3.5 Reporting and Notifications

- Generate detailed transaction reports.
- Notify users of successful, failed, or disputed transactions.

3.6 Security

- Ensure data encryption, tokenization, and PCI DSS compliance.

3.7 Error Handling

- Provide user-friendly error messages and logs.

4. Interface Requirements

4.1 User Interface

- Intuitive and accessible design for merchants and customers.

4.2 Integration Interfaces

- APIs for payment gateway and customer management integrations.

4.3 Reporting Interface

- Allow users to view and export transaction reports.

4.4 Security Interface

- Implement features like encryption and secure authentication.

5. Performance Requirements

5.1 Key Metrics

- Response time: < 2 seconds for payment processing.
- Scalability: Handle up to 10,000 transactions/hour and 1,000 concurrent users.
- Availability: 99.99% uptime.

5.2 Load Testing

- Regular testing to ensure performance under peak conditions.

6. Design Constraints

- Programming Language: Flutter.
- Database: Firebase.
- Minimum system requirements: Windows 10 or above, 1 GHz processor, 512 MB RAM, 4 Mbps internet speed.

7. Non-Functional Attributes

- **Availability:** High uptime.
- **Scalability:** Support for increasing transaction volumes.
- **Reliability:** Consistent performance under varying loads.
- **Security:** Compliance with industry standards.
- **Usability:** User-friendly design for seamless navigation.
- **Extensibility:** Easily extendable for future requirements.

8. Preliminary Schedule and Budget

- **Timeline:** Completion within three months from the start date.
- **Budget:** Covers man-hours only, excluding software costs.