Date __/__/__
Page

SRS for a Credit Card Processing System

1. INTRODUCTION

1.1 Purpose:

The purpose of this document is to define the functional and non-functional requirements of a Credit Card Processing System (CCPS). This system will serve as a secure, reliable, and scalable platform for processing credit and debit card transactions. The document provides a comprehensive blueprint for development, ensuring all project stakeholders are aligned on the system's objectives and specifications.

1.2 Scope:

This SRS addresses the core functionalities of the CCPS, including transaction authorization, capture, voiding and refunding. It also covers the management of merchant accounts and the provision of transaction reporting. The scope does not include advanced features such as fraud detection algorithms, recurring billing, or direct integration with third-party marketplaces.

1.3 Overview:

The CCPS is a service intended for integration by ~~business clients to process~~ card-based payments from ~~it~~ It will be a robust, backend-oriented service that exposes a well-defined API for integration by merchant and e-commerce platforms. The system's architecture will be designed for high availability and strict security, ensuring compliance with industry standards.

Date __/__/__
Page _____

2. GENERAL DESCRIPTION

The ccPS is a service intended for integration by business clients to process card-based payments from their customers. The primary users will be application programming interfaces (APIs) acting on behalf of merchant administrators will access a web-based portal to manage accounts, view transaction history, and generate reports. The system must operate within a secure, cloud-based environment with high-speed internet connectivity.

3. FUNCTIONAL REQUIREMENTS

The system shall perform the following functions:

- Transaction Processing: Process credit and debit card ~~tra~~ransactions, including authorization, capture, void, and refund operations. Each transaction will be assigned a unique identifier.

- Merchant Account Management: Allow administrators to create, modify, and deactivate merchant accounts, each with unique API keys and configuration settings.

- Transaction History: Maintain a searchable and ~~fi~~reable database of all transactions, providing details such as amount, date, time, status, and associated merchant.

- Reporting: Generate detailed transaction reports for merchants, including daily summaries, settlement reports, and custom data-range analyses.

- Tokenization: Securely tokenize cardholder data to minimize the storage of sensitive information on merchant systems.

4. INTERFACE REQUIREMENTS

- Software Interface: The system shall expose a RESTful API for seamless integration. The API will be well-documented and provide endpoints for all core transaction and mangement functions.

- User Interface: A web-based administrative portal shall be provided for merchant administrators to manage their accounts and access reports. The UI must be secure and require multi-factor authentication.

- Communication Interface: All communication with the CCPS API shall be secured using TLS 1.2 or higher to ensure data privacy and integrity.

5. Performance Requirements

- Response Time: Transaction authorization requests shall be processed within 500 milliseconds, with a 99% success rate.

- Concurrency: The system shall be capable of handling a peak load of 500 concurrent transactions per second without performance degradation.

Date __/__/__
Page _____

- Availability: The system must an uptime of 99.99% to ensure continuous service for merchant operations.

6. DESIGN CONSTRAINTS

- Security Compliance: The system must adhere strictly to the Payment card Industry Data security standard (PCI DSS) to ensure the secure handling of cardholder data.

- Technology stack: Development shall utilize a fault-toler. -ant and scalable cloud architecture. The database must be ~~inherently scalable to support a rapid increase in transaction~~ a highly-available, distributed system to support transactional integrity.

- Scalability: The architecture must be inherently scalable to support a rapid increase in transaction volume and the onboarding of new merchants without a complete redesign.

7. NON-FUNCTIONAL REQUIREMENTS

- Security: The system shall implement robust authentication and authorization mechanisms for both API and UI access. All sensitive data will be encrypted at rest and in transit.

Date __/__/__
Page

- Reliability: The system shall be designed with redundancy and automated failover capabilities to prevent a single point of failure. All events and errors must be logged for auditing and analysis.

- Maintainability: The codebase shall be designed with follow best practices for modularity and documentation facilitating future ~~enhancing~~ enhancements and trouble-shooting.

8. PRELIMINARY SCHEDULE AND BUDGET

- Schedule: The estimated timeline for core development, including security and compliance testing, is 9-12 months. This is followed by a two-month pilot phase with select merchants.

- Budget: The preliminary budget for development, based on a specialized team of 5-6 engineers, is estimated in the range of $150,000 to $250,000, excluding ongoing infrastructure and maintenance costs.