Experiment – 02

Credit Card Processing

2.1 PROBLEM STATEMENT:

Credit card processing through offline involves the merchant collecting order information storing this in a database on your site, and entering it using their on-site merchant credit card processing system. Takes time to manually enter credit card information for each order.

This solution creates following cons: ·

Insecure – there is a possibility that a skilled hacker could break into the database and steal an entire list of credit card numbers, thereby damaging the merchant's reputation with current client.

- There is a higher risk of customer charge backs with no signature
- Higher risk of fraud for using stolen credit cards
- Many discerning online shoppers will not give their credit card to an "untrusted" online.

So there is a need of online and trusted credit card processing.

2.2 SOFTWARE REQUIREMENT SPECIFICATION:

2.2.1 INTRODUCTION

The Credit Card Processing System is designed to facilitate the processing of credit card transactions securely and efficiently. This system will be used by merchants and financial institutions to process credit card transactions in real-time. This document outlines the requirements and specifications for the Credit Card Processing System.

PURPOSE OF THE DOCUMENT

When customers complete their shopping cart, their credit card is preauthorized and the order is entered into Sales Order. Credit Card Processing dials out and obtains a credit card payment. Within five minutes the customer receives an e-mail receipt.

SCOPE OF THE DOCUMENT

- Automatically connects to your financial network for credit card authorizations and settlements
- Integrates with Sales Order, Accounts Receivable, and e-Business Manager
- Support for dial-up (modem) connections or secure Internet connections through TCP/IP and SSL
- Compliant with Visa and MasterCard Electronic Commerce Indicator (ECI) regulations
- Multiple address verification options available

OVERVIEW

Credit card processing through offline involves the merchant collecting order information, storing this in a database on your site, and entering it using their on-site merchant credit card processing system. Takes time to manually enter credit card information for each order. This solution creates following cons: · Insecure – there is a possibility that a skilled hacker could break into the database and steal an entire list of credit card numbers, thereby damaging the merchant's reputation with current client. · There is a higher risk of customer charge backs with no signature · Higher risk of fraud for using stolen credit cards · Many discerning online shoppers will not give their credit card to an "untrusted" online merchant

2.2.2 GENERAL DESCRIPTION

A credit card processing system is a software-based platform that facilitates the electronic authorization, verification, and settlement of credit card transactions. It enables merchants to process credit card payments securely and efficiently from customers using their credit or debit cards. The credit card processing system works by securely transmitting customer payment information to the card issuer or the card network, which then approves or declines the transaction based on factors such as available credit, fraud risk, and other security checks. Once approved, the payment is settled and funds are transferred from the customer's account to the merchant's account.

2.2.3 FUNCTIONAL REQUIREMENTS

1. Payment Processing: The system should be able to process credit card payments and verify that the card is valid and has sufficient funds for the transaction. The system should be able to handle transactions in multiple currencies.

2. User Authentication: The system should require customers to authenticate themselves before making a transaction. The system should be able to verify the customer's identity and the authenticity of the credit card being used.

3. Transaction Monitoring: The system should provide real-time transaction monitoring and reporting to both customers and merchants. The system should be able to identify suspicious activity and flag potential fraud.

4. Integration with E-commerce Platforms: The system should be easy to integrate with various e-commerce platforms such as Shopify, WooCommerce, and Magento. The integration should be seamless and require minimal effort from the merchant.

5. Secure Transactions: The system should be designed to ensure the security of all transactions. The system should encrypt all data transmitted between the customer, merchant, and the payment gateway. The system should comply with industry-standard security protocols such as PCI DSS.

6. Refund Management: The system should allow merchants to process refunds for transactions. The system should be able to process partial and full refunds and update the customer's account accordingly.

2.2.4 INTERFACE REQUIREMENTS

Interface Requirements: The credit card processing system shall be designed with a user-friendly interface to facilitate easy and efficient use by the end-user. The following are the interface requirements:

1. Login Interface: The login interface shall provide a secure and reliable means of access to the system. Users shall be required to provide their unique login credentials before accessing the system.

2. Dashboard Interface: The dashboard interface shall display a summary of the user's account, including available credit, transaction history, and pending payments.

3. Payment Interface: The payment interface shall facilitate secure and reliable credit card payments. Users shall be required to provide their credit card information, including card number, expiry date, and CVV code, to complete the payment.

4. Transaction Interface: The transaction interface shall display a detailed summary of each transaction, including the date and time of the transaction, the amount charged, and the transaction status.
5. Report Interface: The report interface shall provide users with the ability to generate reports on their credit card usage, including spending patterns, transaction history, and account balances.
6. Settings Interface: The settings interface shall allow users to configure their account settings, including personal information, billing preferences, and notification settings.
7. Help and Support Interface: The help and support interface shall provide users with access to documentation, FAQs, and support channels to assist with any issues they may encounter while using the system.

2.2.5 PERFORMANCE REQUIREMENTS

1. Response Time: The system should have a maximum response time of 2 seconds for all transactions. This includes the time taken to verify the user's identity, authenticate the credit card, and process the transaction.
2. Transaction Volume: The system should be capable of handling at least 1000 transactions per second during peak times. This includes processing, authorization, and settlement of transactions.
3. System Availability: The system should have a minimum uptime of 99.99%. This means the system should be available and accessible to users for at least 99.99% of the time.
4. Network Latency: The system should have a maximum network latency of 100 milliseconds for all transactions. This includes the time taken to transmit data between the user's device and the system.
5. Data Throughput: The system should be capable of processing at least 10 GB of data per day. This includes transaction data, user data, and system logs.
6. Concurrent Users: The system should be capable of handling at least 10,000 concurrent users at peak times. This includes users accessing the system via mobile devices, web browsers, and other applications.

7. Scalability: The system should be scalable to handle increasing transaction volumes and user loads. The system should be able to add additional resources, such as servers or processing power, to accommodate increased demand.
8. Error Rates: The system should have a low error rate of less than 0.01%. This includes errors such as failed transactions, timeouts, and system failures.
9. Security: The system should be secure and comply with industry standards such as PCI DSS. This includes protecting user data, preventing fraud, and ensuring system integrity.

2.2.6 NON-FUNCTIONAL REQUIREMENTS

1. Reliability: The system should be designed to ensure high availability and reliability. The system should be able to handle large volumes of transactions without downtime.
2. Performance: The system should be able to process transactions quickly and efficiently. The system should have low response times and be able to handle high levels of traffic.
3. Scalability: The system should be able to scale horizontally and vertically to handle increasing transaction volumes. The system should be able to add additional resources as needed without downtime.
4. User-Friendly: The system should be designed to be user-friendly and easy to use for both customers and merchants. The system should have a simple and intuitive interface that is easy to navigate.

2.2.7 DESIGN CONSTRAINTS

1. Security: The system must comply with Payment Card Industry Data Security Standards (PCI-DSS) and other relevant security regulations. The system must encrypt all sensitive data, including credit card numbers, and store it securely.
2. Scalability: The system must be scalable to handle a large volume of transactions. The system must be designed to handle peak loads and scale up or down as needed.
3. Availability: The system must be highly available and have a reliable failover mechanism in place to ensure that transactions can be processed even in the event of a failure.

4. Compatibility: The system must be compatible with different types of credit cards and payment gateways. The system must also be compatible with different types of merchant accounts.
5. Performance: The system must be designed for optimal performance, with response times that meet or exceed industry standards.
6. User Interface: The user interface must be intuitive and easy to use, with clear navigation and easy-to-understand instructions. The system must also be accessible to users with disabilities.

2.2.8 PRELIMINARY SCHEDULE AND BUDGET

Schedule:

Requirements gathering and analysis - 2 weeks

Design and architecture - 4 weeks

Development and coding - 12 weeks

Testing and quality assurance - 4 weeks

Deployment and training - 2 weeks

Total time estimate: 24 weeks or 6 months

Budget:

Salaries for project team (developers, testers, designers, project manager) - $400,000

Hardware and software infrastructure - $50,000

Third-party payment gateway integration - $20,000

Miscellaneous expenses (travel, training, etc.) - $30,000

Total budget estimate: $500,000