CREDIT CARD PROCESSING SYSTEM

SOFTWARE REQUIREMENTS SPECIFICATION

SOFTWARE REQUIREMENTS SPECIFICATION

FOR A CREDIT CARD PROCESSING SYSTEM

1. Introduction

1.1 Purpose of this Document : This document defines the requirements for a software system's functionality, performance expectation, and design constraints, serving as a comprehensive guide for the development team.

1.2 Scope of this Document : This document encompasses the functional and non-functional requirements for the credit card processing system, including I/O specifications, transaction processing details, security protocols, and interface requirements for interacting with external systems like payment gateways and banks.

1.3 Overview : The credit card processing system will be robust and secure software solution enabling businesses to accept credit card payments from customers. It will manage transaction authorization, settlement, and reconciliation processes while adhering to Industry security standards, particularly the payment card Industry Data Security Standard (PCI DSS)

2. General Description :

2.1 User Objectives : The system should enable merchants to quickly and securely process credit card transactions, manage transaction records, generate reports, and integrate seamlessly with existing business systems.

2.2 User Characteristics : The system will be used by cashiers, store managers, and accounting personnel, each with varying levels of techical proficiency. Therefore the system should offer a user-friendly interface with appropriate access controls.

2.3 Features :

- Transaction Processing : Process credit card authorizations, captures, refunds, and voids.
- Security : Implement robust security measure to protect sensitive cardholder data and ensure PCI DSS compliance.
- Reporting and Reconciliation : Generate detailed transaction reports and facilitate easy reconciliation with bank statements.
- Integration : Provide APIs or Integration capabilities for seamless connections with other business systems, such as POS terminals and accounting software.
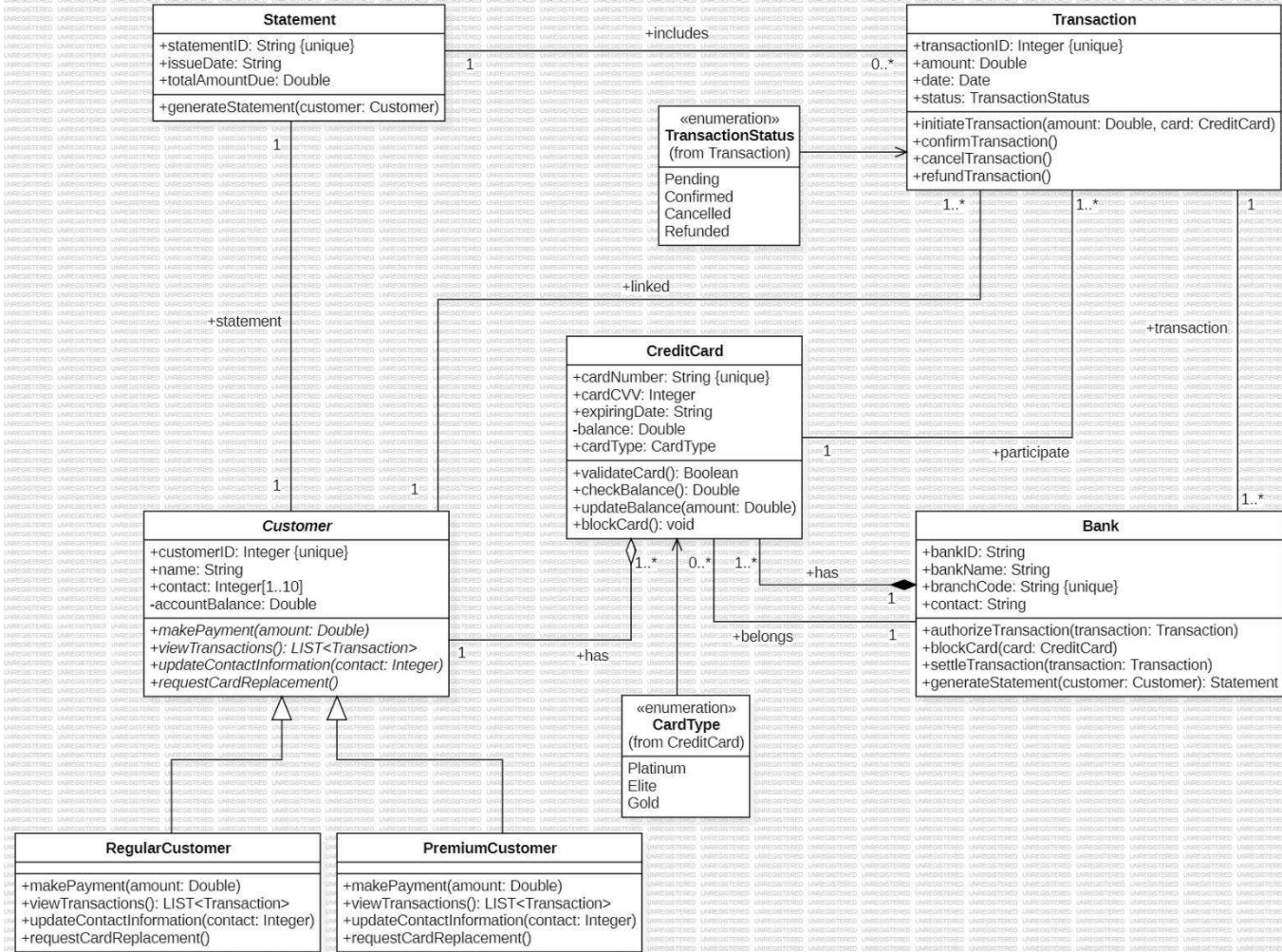
2.4 Benefits : The system will streamline the payment process, reduce processing costs, improve cash flow, minimize the risk of fraudulent transactions, and enhance customer trust.

3. Functional Requirements :

3.1 Transaction Processing : The system shall support various card types, shall perform real-time authorization requests to appropriate card networks.

3.2 Security : The system shall encrypt all sensitive cardholder data at rest and in transit, shall use tokenization to replace actual card number with unique tokens during processing

3.3 Reporting and Reconciliation : The system shall generate detailed transaction reports, including date, time, amount, card type and authorization status.

3.4 Integration : The system shall offer APIs or SDKs for integration with other business systems, ensuring seamless operation.

4. Interface Requirements :

4.1 User Interface : The system shall provide a web-based UI accessible through both desktop and mobile devices.

4.2 External System Interfaces :

- Payment Gateways : For secure communication and transaction routing.
- Acquiring Banks : To facilitate the receipt of funds for processed transactions.
- Card networks : For transactions authorization and settlement.

5. Performance Requirements

5.1 Transaction Processing Speed : The system should process authorization requests within a few seconds to ensure a smooth checkout experience for customers.

5.2 System Availability : The system should be highly available, targeting an uptime of 99.99% to minimize disruptions in credit card processing.

5.3 Scalability : The system should be scalable to handle increasing transaction volumes as the business grows.

6. Design Constraints :

6.1 Security Compliance : The system must comply with the payment card Industry Data Security Standard (PCI DSS) to ensure the secure handling of credit card information.

6.2 Legal and Regulatory Requirements : The system must adhere to all relevant legal and regulatory requirements related to credit card processing in the regions where it operates.

7. Non-Functional Attributes :

7.1 Security

7.2 Reliability

7.3 Usability

7.4 Maintainability

8. Preliminary Schedule and Budget

8.1 Schedule : The project is estimated to take 6 months, broken into key phases:

- Requirements gathering (2 weeks)
- Design Phase (1 month)
- Development Phase (3 month)
- Testing Phase (1 month)
- Deployment and training (2 weeks)
- Post-deployment support (2 weeks).

8.2 Budget : The total estimated budget is $150,000, allocated as follows:

- Requirements gathering : $15000
- Design Phase : $25,000
- Development Phase : $80,000
- Testing Phase : $20,000
- Deployment and training : $7,500
- Post-deployment support : $2,500.

UML DIAGRAMS

CLASS DIAGRAM

CREDIT CARD PROCESSING SYSTEM - CLASS DIAGRAM

Statement

+statementID: String {unique}
+issueDate: String
+totalAmountDue: Double

+generateStatement(customer: Customer)

Transaction

+transactionID: Integer {unique}
+amount: Double
+date: Date
+status: TransactionStatus

+initiateTransaction(amount: Double, card: CreditCard)
+confirmTransaction()
+cancelTransaction()
+refundTransaction()

+includes

1   0..*

«enumeration»
TransactionStatus
(from Transaction)

Pending
Confirmed
Cancelled
Refunded

1

1..*   1..*   1

+statement

+linked

+transaction

CreditCard

+cardNumber: String {unique}
+cardCVV: Integer
+expiringDate: String
-balance: Double
+cardType: CardType

+validateCard(): Boolean
+checkBalance(): Double
+updateBalance(amount: Double)
+blockCard(): void

1   +participate

Customer

+customerID: Integer {unique}
+name: String
+contact: Integer[1..10]
-accountBalance: Double

+makePayment(amount: Double)
+viewTransactions(): LIST<Transaction>
+updateContactInformation(contact: Integer)
+requestCardReplacement()

Bank

+bankID: String
+bankName: String
+branchCode: String {unique}
+contact: String

+authorizeTransaction(transaction: Transaction)
+blockCard(card: CreditCard)
+settleTransaction(transaction: Transaction)
+generateStatement(customer: Customer): Statement

1   1

1..*   0..*   1..*   +has

1

+belongs   1

1   +has

«enumeration»
CardType
(from CreditCard)

Platinum
Elite
Gold

RegularCustomer

+makePayment(amount: Double)
+viewTransactions(): LIST<Transaction>
+updateContactInformation(contact: Integer)
+requestCardReplacement()

PremiumCustomer

+makePayment(amount: Double)
+viewTransactions(): LIST<Transaction>
+updateContactInformation(contact: Integer)
+requestCardReplacement()

2. CREDIT CARD PROCESSING

<<enumerations>>
Status
Pending
Confirmed
Cancelled
Refunded

CreditCard

CreditCard		
CardNumber : String		
card CVV : Int		
expiry Date : String		
Balance : Float		
validate Card ()		
checkBalance ()		
updateBalance()		
block Card ()		

1 .... participate .... 1..*

Transaction

Transaction	
Transaction ID : Int	
amount : Float	
date : String	
Status : Bool	
Initiate Transaction ()	
confirm Transaction ()	
cancel Transaction ()	
refund Transaction ()	

has

1..*   belongs

Transaction

1..*

Customer

Customer	
customerId : Int	
Name : String	
contact : Int	
accBalance : Float	
make Payment ()	
view Transaction ()	
update Contact Info ()	
request Card Replacement ()	

Bank

Bank	
BankId : Int	
bank Name : String	
branch Code : String	
contact : Int	
authorize Transaction ()	
block Card ()	
settle Transaction ()	
generate Statement ()	

Statement   1

includes

Statement

Statement	
ID : String	
IssuDate : String	
total Amount Due : Double	
generate Statement ()	

1