Credit Card Payment Gateway

UML REPORT

Academic Year 2022-23 ODD SEMESTER

Department with Specialization: Computer Science and Engineering with Specialisation in Artificial. Intelligence & Machine Learning
Semester : III
Course Code : 18CSC202J
Course Title : OBJECT ORIENTED DESIGN AND PROGRAMMING

Submitted By

Shashank Venkatesha (Reg No: RA2111026010269)
Tharun Raj B (Reg No: RA2111026010243)

Under the Guidance of
Dr. M. Uma
Associate Professor

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR- 603 203

NOVEMBER 2022

BONAFIDE

This is to certify that 18CSC202J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY Project Report titled "B" is the Bonafide work of

Shashank Venkatesha (RA2111026010269)
Tharun Raj (RA2111026010243)

who undertook the task of completing the project within the allotted time.

Signature of the Guide

Dr. M. Uma
Associate Professor
Department of CINTEL
SRM Institute of Science and Technology

Signature of the II Year Academic Advisor

Professor and Head
Department of CINTEL
SRM Institute of Science and Technology

About the course:-

18CSC202J/ 8AIC203J - Object Oriented Design and Programming are 4 credit courses with L T P C as 3-0-2-4 (Tutorial modified as Practical from 2018 Curriculum onwards)

Objectives:

The student should be made to:
• Learn the basics of OOP concepts in C++
• Learn the basics of OOP analysis and design skills. • Be exposed to the UML design diagrams.
• Be familiar with the various testing techniques

Course Learning Rationale (CLR): The purpose of learning this course is to:

1. Utilize class and build domain model for real-time programs
2. Utilize method overloading and operator overloading for real-time application development programs
3. Utilize inline, friend and virtual functions and create application development programs
4. Utilize exceptional handling and collections for real-time object- oriented programming applications
5. Construct UML component diagram and deployment diagram for design of applications
6. Create programs using object-oriented approach and design methodologies for real-time application development

Course Learning Outcomes (CLO): At the end of this course, learners will be able to:

1. Identify the class and build domain model
2. Construct programs using method overloading and operator overloading
3. Create programs using inline, friend and virtual functions, construct programs using standard templates
4. Construct programs using exceptional handling and collections
5. Create UML component diagram and deployment diagram
6. Create programs using object oriented approach and design methodologies

Table 1: Rubrics for Laboratory Exercises

(Internal Mark Splitup:- As per Curriculum)

CLAP-1	5=(2(E-lab Completion) + 2(Simple Exercises)(from CodeZinger, and any other coding platform) + 1(HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-2	7.5=(2.0(E-lab Completion)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-3	7.5=(2.0(E-lab Completion(80 Pgms)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	**2 Mark -** E-lab Completion **80 Program** Completion from 10 Session (Each session min 8 program) **2 Mark -** Code to UML conversion GCR Exercises **3.5 Mark - Hacker Rank** Coding challenge completion
CLAP-4	5= 3 (Model Practical) + 2(Oral Viva)	• **3 Mark** – Model Test • **2 Mark** – Oral Viva

Total 25

COURSE ASSESSMENT PLAN FOR OODP LAB

S.No	List of Experiments	Course Learning Outcomes (CLO)	Blooms Level	PI	No of Programs in each session
1.	Implementation of I/O Operations in C++	CLO-1	Understand	2.8.1	10
2.	Implementation of Classes and Objects in C++	CLO-1	Apply	2.6.1	10
3,	To develop a problem statement. 1. From the problem statement, Identify Use Cases and develop the Use Case model. 2. From the problem statement, Identify the conceptual classes and develop a domain model with a UML Class diagram.	CLO-1	Analysis	4.6.1	Mini Project Given
4.	Implementation of Constructor Overloading and Method Overloading in C++	CLO-2	Apply	2.6.1	10
5.	Implementation of Operator Overloading in C++	CLO-2	Apply	2.6.1	10
6.	Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams and Collaboration diagrams	CLO-2	Analysis	4.6.1	Mini Project Given
7.	Implementation of Inheritance concepts in C++	CLO-3	Apply	2.6.1	10
8.	Implementation of Virtual function & interface concepts in C++	CLO-3	Apply	2.6.1	10
9.	Using the identified scenarios in your project, draw relevant state charts and activity diagrams.	CLO-3	Analysis	4.6.1	Mini Project Given
10.	Implementation of Templates in C++	CLO-3	Apply	2.6.1	10
11.	Implementation of Exception of Handling in C++	CLO-4	Apply	2.6.1	10

12.	Identify the User Interface, Domain objects, and Technical Services. Draw the partial layered, logical architecture diagram with UML package diagram notation such as Component Diagram, Deployment Diagram.	CLO-5	Analysis	4.6.1	Mini Project Given
13.	Implementation of STL Containers in C++	CLO-6	Apply	2.6.1	10
14.	Implementation of STL associate containers and algorithms in C++	CLO-6	Apply	2.6.1	10
15.	Implementation of Streams and File Handling in C++	CLO-6	Apply	2.6.1	10

LIST OF EXPERIMNENTS FOR UML DESIGN AND MODELLING:

To develop a mini project by following the exercises listed below.

1. To develop a problem statement.
2. Identify Use Cases and develop the Use Case model.
3. Identify the conceptual classes and develop a domain model with UML Class diagram.
4. Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams.
5. Draw relevant state charts and activity diagrams.
6. Identify the User Interface, Domain objects, and technical services. Draw the partial layered, logical architecture diagram with UML package diagram notation.

Suggested Software Tools for UML:
StarUML, Rational Suite, Argo UML (or) equivalent, Eclipse IDE and Junit

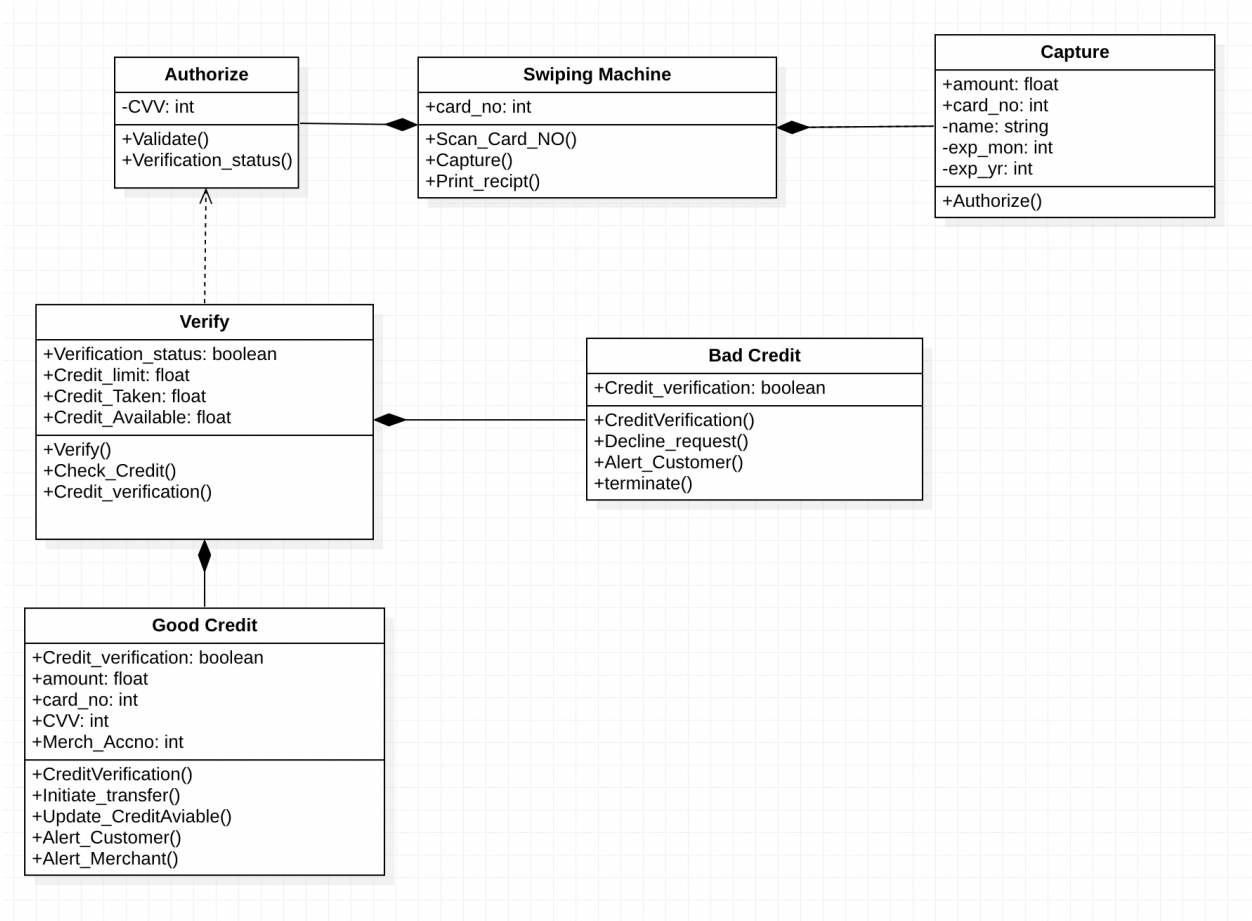
Table of Contents

Abstract:

Payment gateway: A payment gateway is a software that transmits transaction information from a payment portal such as an e-commerce website to bank's processor and authorisation responses from the credit card issuing banks to the payment portal. It usually facilitates communication between the banks.

The payment gateway process flow
1. At checkout, merchant's Card Swiping Machine sends encrypted payment details to payment gateway.
2. Payment gateway sends request to customer's bank for authorisation
3. If payment was authorised, the customer's bank pays the merchant

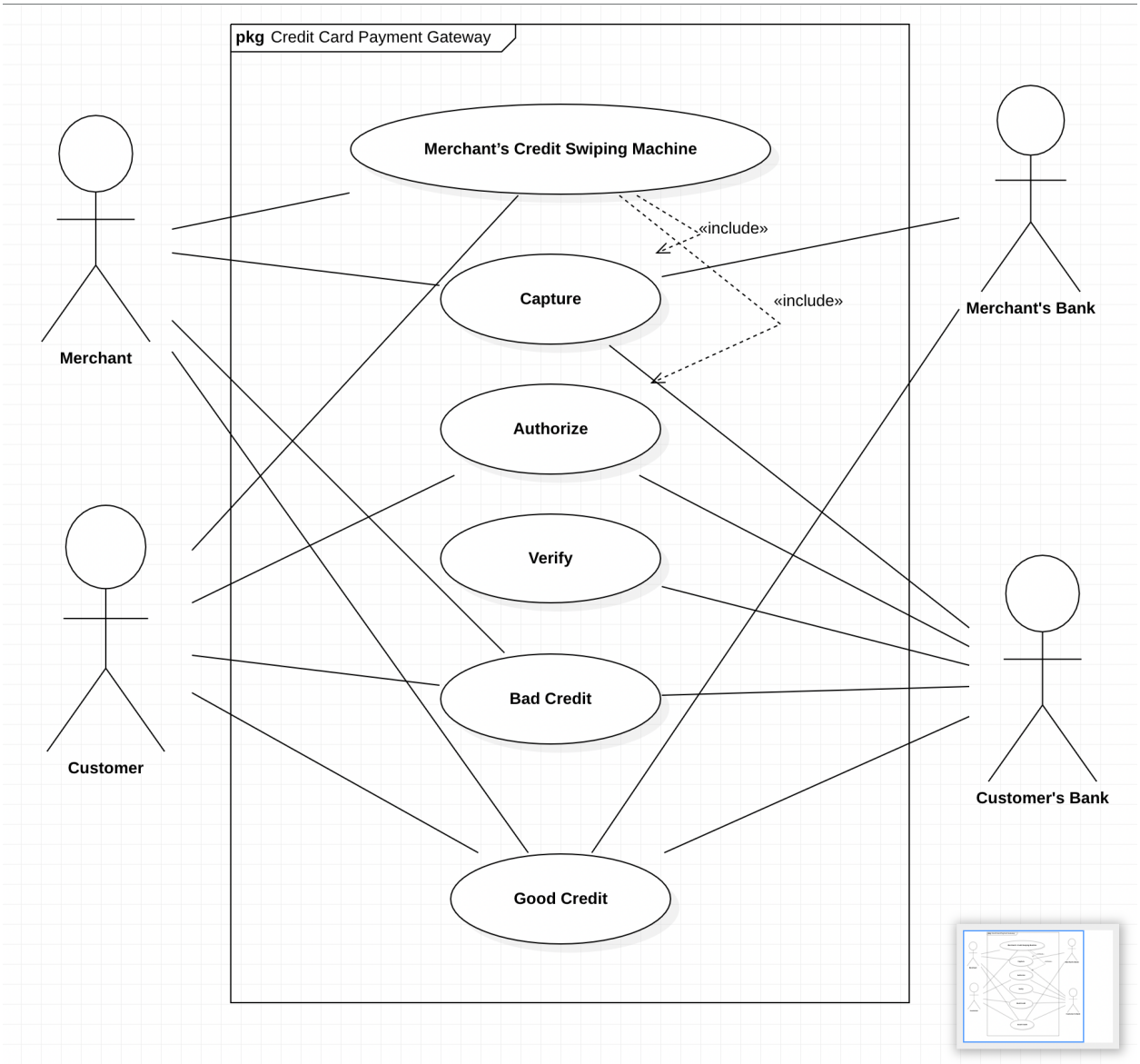
Class Diagram:



Relationship Among Classes:

Class Authorize is Base class with 2 derived classes Swiping Machine and Verify for security reasons to run program with authentication. Again Class Verify has 2 dependency classes and derived classes Good Credit and Bad Credit to verify the credentials so that no fraud occurs. Class Swiping Machine also has a dependent derived class Capture for getting the details and sending it to the bank server.

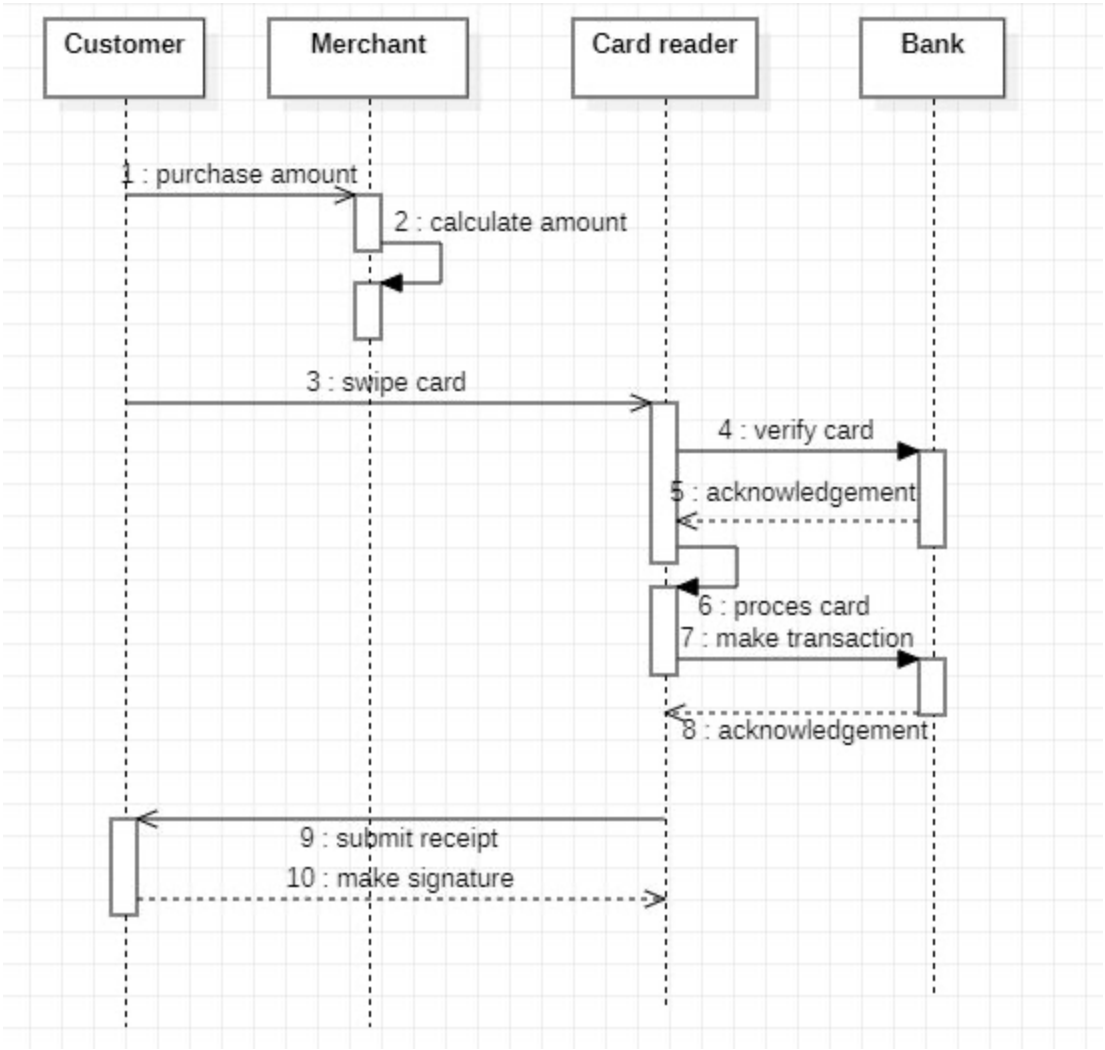
Use Case Diagram



This use case diagram consists of four actors Merchant, Customer and Bank (System). The Merchant requests money from the swiping machine. The card is entered and verified and then asks for the pin. The pin is entered by the costumer and will be verified by the payment gate way. If verified the payment is success and receipt is given out. If not the payment is rejected due to incorrect pin or insufficient credit limit.

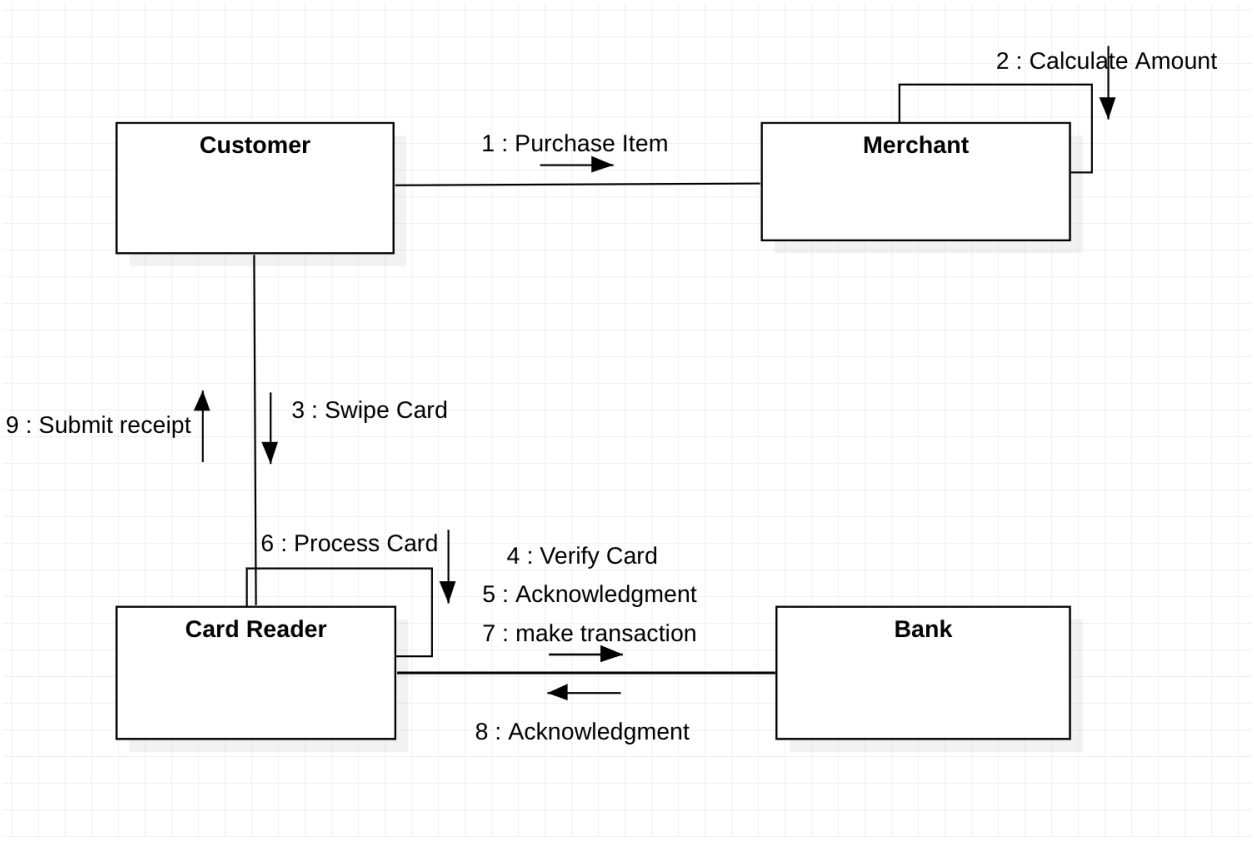
Interaction Diagram:

Sequence Diagram:



In this sequence diagram 4 lifelines are created: Customer, Bank, Card Reader, and Merchant. Following operations take place in the sequence diagram. Firstly, The Merchant requests money from the swiping machine. Customer swipes the card. The card is entered and verified and then asks for the pin. The pin is entered by the costumer and will be verified by the payment gate way and send signals. If verified the payment will be done to the merchant's bank successfully and receipt is given out. If not the payment is rejected due to incorrect pin or insufficient credit limit.

Collaboration Diagram



This collaboration diagram focuses on the relationship of objects- how they associate and connect through messages in a sequence rather than interactions. In this collaboration diagram, the interaction between various lifelines is shown such as Bank, Customer, Card Reader and Merchant.

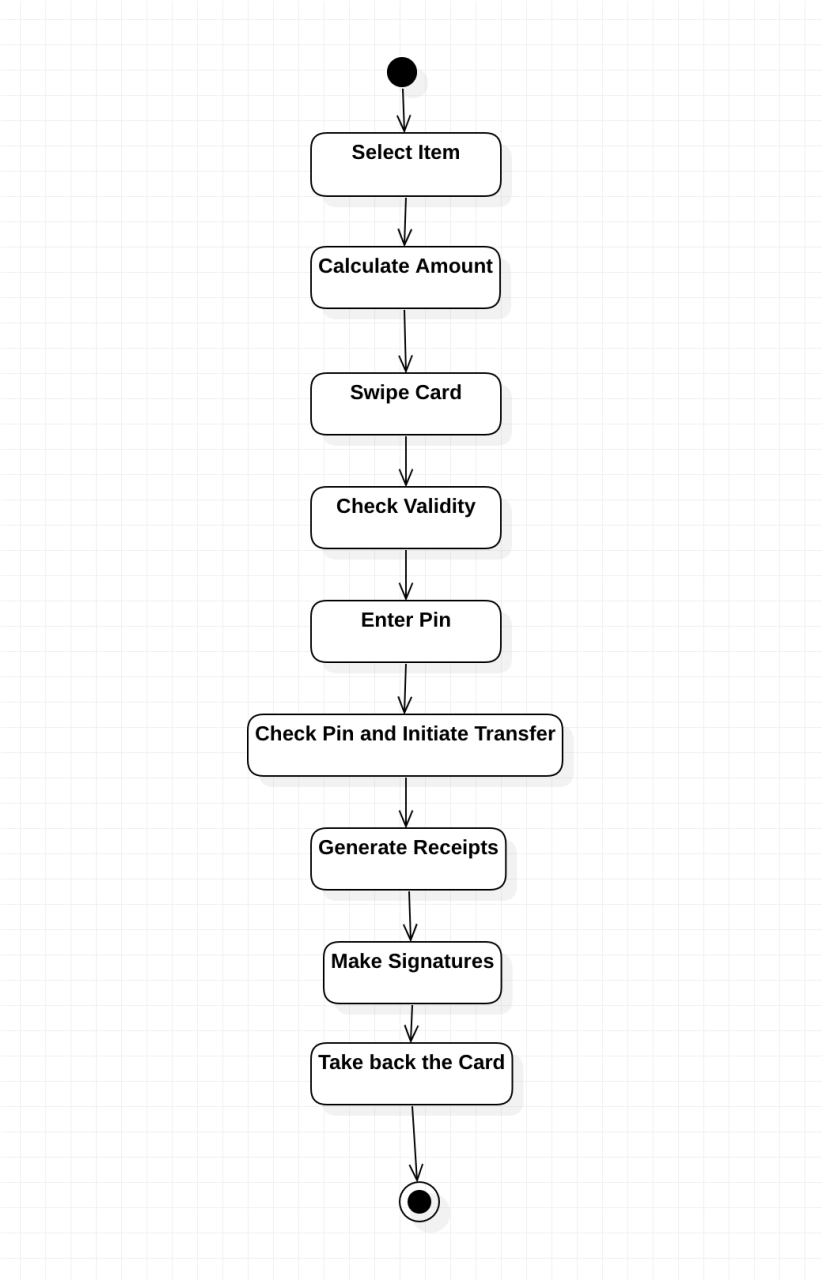
The following operations are shown in this collaboration diagram.

1. Customers makes a purchase
2. The Merchant requests money from the swiping machine after Calculating the amount.
3. Customer swipes the card.
4. The card is entered and verified and then asks for the pin.
5. The pin is entered by the costumer
6. and will be verified by the payment gate way and send signals.
7. If verified the payment will be done to the merchant's bank successfully. If not the payment is rejected due to incorrect pin or insufficient credit limit.
8. Acknowledgment is given after transaction.

9. and receipt is given out to the customer.

Behavioural Diagram:

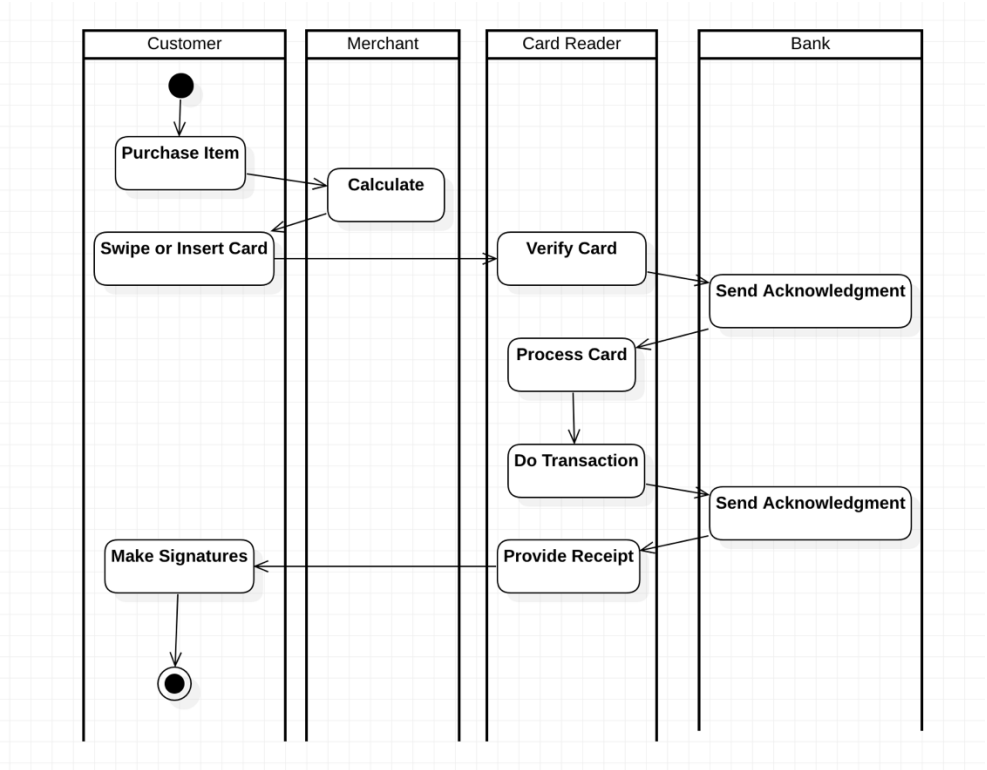
State Chart Diagram



States of object are represented as rectangle with round corner, the transaction between the different states. A transition is a relationship between two state that indicates that when an event occur the object moves from the prior state to the subsequent.

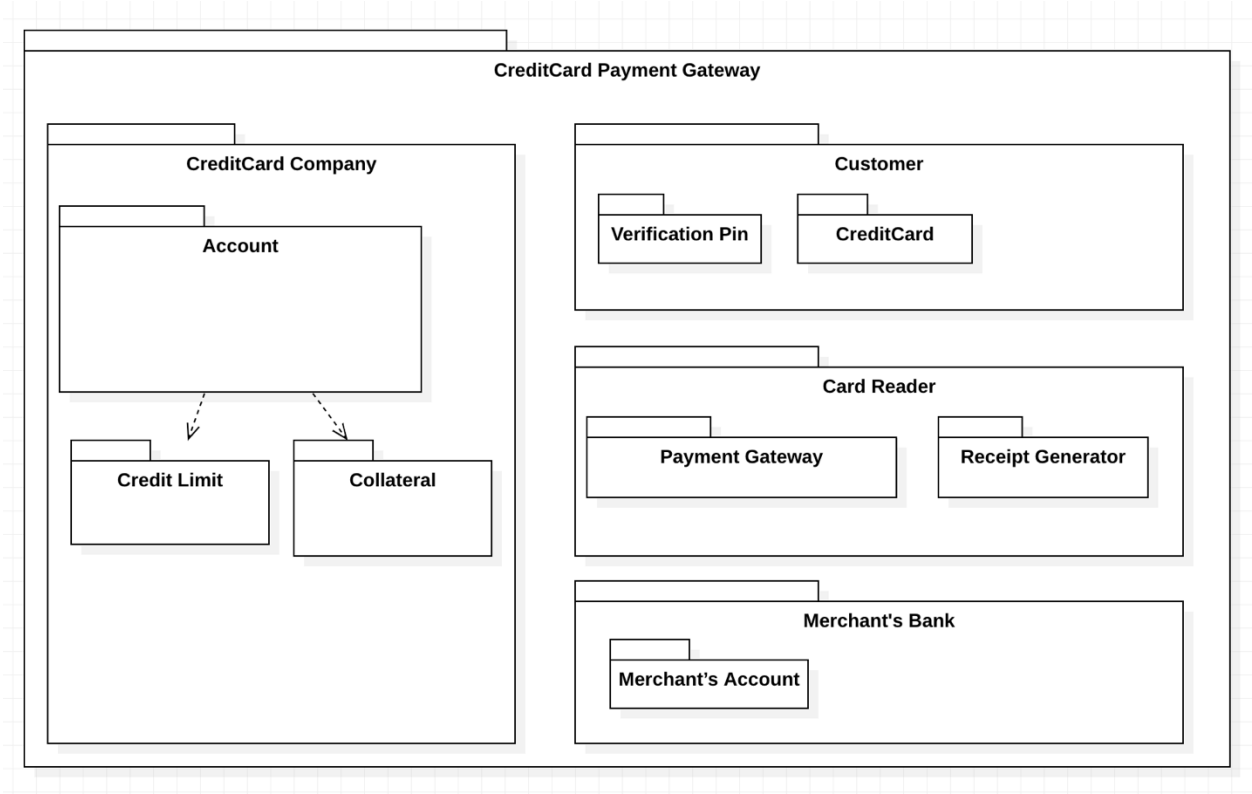
1. Customers makes a purchase
2. The Merchant requests money from the swiping machine after calculating the amount.
3. Customer swipes the card.
4. The card is entered and verified and
5. then asks for the pin and is entered by the costumer
6. and will be verified by the payment gate way and send signals. If verified the payment will be done to the merchant's bank successfully. If not the payment is rejected due to incorrect pin or insufficient credit limit.
7. and receipt is given out to the customer.
8. Acknowledgment is given after transaction and make signature in receipt.
9. Take back the card

Activity Diagram



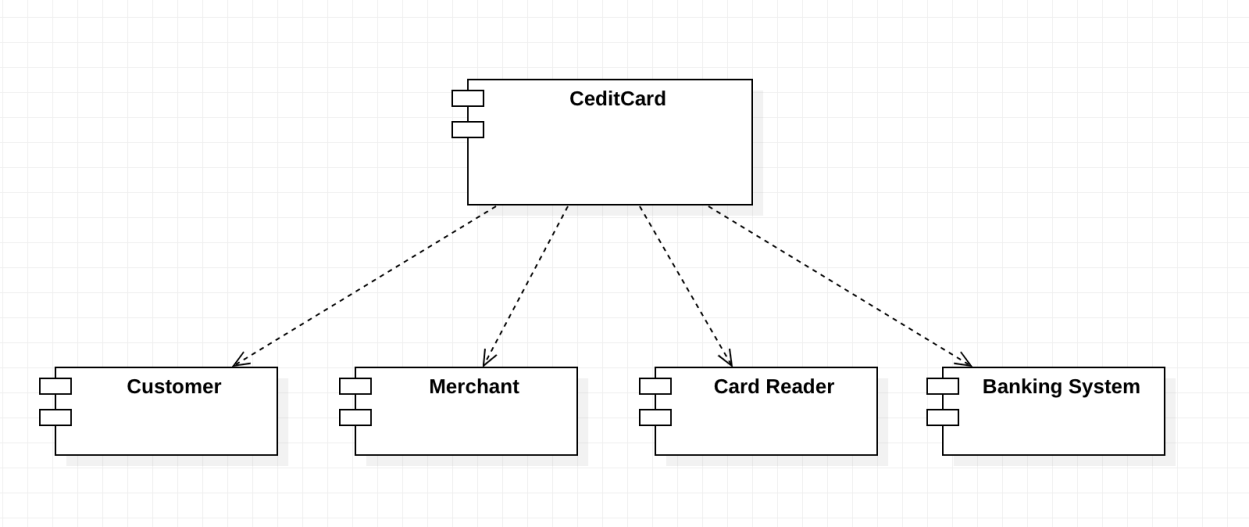
In this Activity Diagram, firstly, The Merchant requests money from the swiping machine. Customer swipes the card. The card is entered and verified and then asks for the pin. The pin is entered by the costumer and will be verified by the payment gate way and send signals. If verified the payment will be done to the merchant's bank successfully and receipt is given out. If not the payment is rejected due to incorrect pin or insufficient credit limit.

Package Diagram:



The Credit Card Payment Gateway package has 4 sub packages namely CreditCard Company, Customer, Card Reader and Merchant's Bank. CreditCard Company package has 3 sub packages Account, Credit Limit and Collateral, where in Credit Limit and Collateral package will be dependent on the Account Package. Customer package has 2 sub packages Verification Pin and CreditCard. Card Reader has Payment Gateway and Receipt Generator packages. Merchant's Bank Package has Merchant's Account.

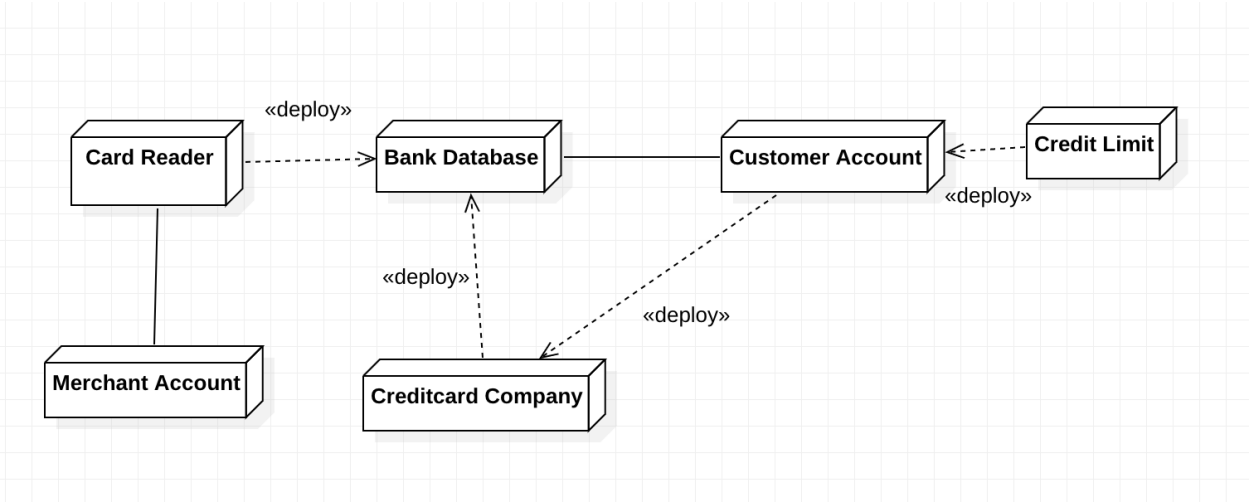
Component Diagram:



Component diagrams are used to visualize the organization and relationships among components in a system.

The Customer, Merchant , Card Reader and the Banking System are all Components dependent on the CerditCard.

DEPLOYMENT DIAGRAM:



Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

The deployment diagram consists of Bank Database node. Card Reader, CreditCard Company is deployed to bank database. Card Reader is linked to Merchant's Account. Credit Limit is deployed to Customer Account. Customer account is Linked to CreditCard Company. The Card Reader and Credit Card are the hardware components whereas the Merchant Account, Bank Database, Credit Card Company, Customer Account, Credit Limit are the software components. Customer interacts with the Card Reader. The Card Reader accesses the bank database to enable transaction physically the web page deploys to bank database to enable online transaction by transfer of data from Card Reader.

Conclusion:

Here are the UML Diagrams that composes a CREDITCARD PAYMENT GATEWAY System. Each of the UML Diagrams has a major role in achieving a well-developed and functional CREDITCARD PAYMENT GATEWAY System.

The UML Diagrams works together to achieve the most desired functions of the CREDITCARD PAYMENT GATEWAY Project. All of these were designed to guide programmers and beginners about the behavior and structure of CREDITCARD PAYMENT GATEWAY.

By completing all the given Diagrams, the CREDITCARD PAYMENT GATEWAY Project System development would be much easier and attainable. So those UML diagrams were given to teach you and guide you through your project development journey. You can use all the given UML diagrams as your reference or have them for your project development. The ideas presented in UML Diagrams were all based on CREDITCARD PAYMENT GATEWAY System requirements.

https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm
https://www.javatpoint.com/uml-diagrams
https://docs.staruml.io/working-with-uml-diagrams/use-case-diagram