

Guarded Remittance System Employing WANET for Catastrophe Region

A PROJECT REPORT

Submitted by

YOKESHWARAN.T	(211414205124)
MANOJ.S	(211414205048)
NISHANTH.M	(211414205066)
VENUGOPAL.S	(211414205118)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

PANIMALAR ENGINEERING COLLEGE, POONAMALLEE

ANNA UNIVERSITY : CHENNAI 600 025

APRIL 2018

BONAFIDE CERTIFICATE

Certified that this project report **“Guarded Remittance System Employing WANET for Catastrophe Region”** is the bonafide work of **“YOKESHWARAN.T (211411405124), MANOJ.S (211414205048), NISHANTH.M (211414205066), and VENUGOPAL.S (211414205118)”** who carried out the project under my supervision.

SIGNATURE

**Dr.M.HELDA MERCY,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT**

Department of Information Technology
Panimalar Engineering College,
Poonamallee, Chennai - 600 123

SIGNATURE

**Mr.N.BALA SUNDARA GANAPATHY M.E
SUPERVISOR
ASSISTANT PROFESSOR**

Department of Information Technology
Panimalar Engineering College,
Poonamallee, Chennai - 600 123

Submitted for the project and viva-voce examination held on _____

SIGNATURE

INTERNAL EXAMINER

SIGNATURE

EXTERNAL EXAMINER

DECLARATION

I hereby declare that the project report entitled “**GUARDED REMITTANCE SYSTEM EMPLOYING WANET FOR CATASTROPHE REGION**” which is being submitted in partial fulfilment of the requirement of the course leading to the award of the ‘Bachelor of Technology in Information Technology’ in **Panimalar Engineering College, Affiliated to Anna University-Chennai** is the result of the project carried out by me under the guidance and supervision of **Mr. N.Bala Sundara Ganapathy, M.E., Assistant Professor in the Department of Information Technology**. I further declared that I or any other person has not previously submitted this project report to any other institution/university for any other degree/ diploma or any other person.

Date :

Place : Chennai

Signature of batch members

(Yokeshwaran.T)

(Manoj.S)

(Nishanth.M)

(Venugopal.S)

It is certified that this project has been prepared and submitted under my guidance.

Date : **(Mr. N. BALA SUNDARA GANAPATHI M.E.,)**

Place : Chennai

(Assistant Professor / IT)

ACKNOWLEDGEMENT

A project of this magnitude and nature requires kind co-operation and support from many, for successful completion. We wish to express our sincere thanks to all those who were involved in the completion of this project.

Our sincere thanks to **Honorable Founder Dr. JEPPIAAR, M.A., Ph.D.**, for his sincere endeavor in educating us in his premier institution.

We would like to express our deep gratitude to **Our Beloved Secretary and Correspondent, Dr. P. CHINNADURAI, M.A., Ph.D.**, for his kind words and enthusiastic motivation which inspired us a lot in completing this project.

We also express our sincere thanks to **Our Dynamic Directors Mrs. C. VIJAYA RAJESHWARI and Mr. C. SAKTHI KUMAR, M.E.**, for providing us with the necessary facilities for completion of this project.

We also express our appreciation and gratefulness to **Our Principal Dr. K. MANI, M.E., Ph.D.**, who helped us in the completion of the project. We wish to convey our thanks and gratitude to our Head of the Department, **Dr. M. HELDA MERCY, M.E., Ph.D.**, Department of Information Technology, for her support and by providing us ample time to complete our project.

We express our indebtedness and gratitude to our staff in charge, **Mr. N. BALA SUNDARA GANAPATHY, M.E.**, Assistant Professor, Department of Information Technology for his guidance throughout the course of our project. Last, we thank our parents and friends for providing their extensive moral support and encouragement during the course of the project.

ABSTRACT

Mobile payment system in a disaster area have the potential to provide electronic transactions for people to purchase items like foodstuffs, clothes, and medicine. The current payment system requires a fixed infrastructure network such as cellular towers, or wired networks to enable transactions. In Disaster areas, these infrastructures may not be available and impossible to get cash from banks or ATM. To overcome this issue we propose an offline mobile payment system for catastrophe region utilizing WANET (Wireless Adhoc Network). WANET provide an infrastructure less wireless network for communication between bank and user. To make this possible we introduce a mobile payment application which enables us to make payment offline. We use multilevel endorsement mechanism to ensure guarantee payment for merchant. To reduce communication overheads which may occurs during transaction we use lightweight scheme based on Bloom filter and Merkle tree. In order to make transaction easier and secured, QR codes are used with digital signatures which restricts an attacker from double spending. In addition with payment system, regional situation update is provided to upload their current situation such as roadblocks, floods etc.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	ix
	LIST OF TABLES	xi
	LIST OF ABBREVIATIONS	xii
1.	INTRODUCTION	
	1.1 Overview of the Project	1
	1.2 Need for the Project	1
	1.3 Objective of the Project	1
	1.4 Scope of the Project	2
	1.5 Existing System	2
	1.6 Proposed System	3
2.	LITERATURE SURVEY	
	2.1 An Endorsement-based Mobile Payment System for a Disaster Area	4
	2.2 Implication of Secure Micropayment System Using Process Oriented Structural Design by Hash chaining in Mobile Network	5
	2.3 An efficient and secured mobile payment protocol for restricted connectivity scenarios in vehicular ad hoc network	6
	2.4 Off-line Micro-payment Protocol for Multiple Vendors in Mobile Commerce	7
	2.5 Wireless mesh networks: a survey	8

2.6 Feasibility Study	9
3. SYSTEM DESIGN	
3.1 Introduction	11
3.2 Architecture Diagram	12
3.3 UML Diagrams	13
3.3.1 Use Case Diagram	14
3.3.2 Class Diagram	15
3.3.3 Sequence Diagram	16
3.3.4 Activity Diagram	18
3.3.5 Collaboration Diagram	19
4. REQUIREMENTS SPECIFICATION	
4.1 Introduction	21
4.2 Hardware and Software specification	21
4.2.1 Hardware Requirements	22
4.2.2 Software Requirements	22
4.3 Software Technologies	
4.3.1 Java	22
4.3.2 JDBC	25
4.3.3 JSP	27
4.3.4 Android	30
4.3.5 Apache Tomcat	31
4.3.6 MYSQL	34
4.4 Software Tools	
4.4.1 Net Beans IDE	36

4.4.2	MYSQL Front	37
5.	MODULE DESCRIPTION	
5.1	Introduction	38
5.2	List of Module	38
5.2.1	Bank account creation	38
5.2.2	Customer Payment Request	41
5.2.3	Merchant Process	43
5.2.4	Region Situation Update	44
6.	IMPLEMENTAION	
6.1	Sample Code	45
6.2	Sample Screen Shots	54
6.3	Sample Database Tables	65
7.	TESTING AND MAINTENANCE	
7.1	Testing	66
7.2	Test Cases	70
7.3	Maintenance	71
8.	CONCLUSION AND FUTURE ENHANCEMENT	
8.1	Conclusion	73
8.2	Future Enhancement	73
9.	REFERENCES	74
10.	APPENDIX	76

LIST OF FIGURES

Fig No.	Name of the Figure	Page No.
3.2	Architecture Diagram	12
3.3.1	Use case Diagram	14
3.3.2	Class Diagram	15
3.3.3	Sequence Diagram	17
3.3.4	Activity Diagram	18
3.3.5	Collaboration Diagram	20
4.3.1 (a)	Java Features	23
4.3.1 (b)	Working of Java	24
4.3.3	JSP Model Architecture	28
5.2.1.1 (a)	Bank Interface	39
5.2.1.1 (b)	User Database	39
5.2.1.2	Bank Account Creation	40
5.2.1.3	Registration Page	41
5.2.2	Bank Generated QR Code	43
6.2.1	Navigation Page	54
6.2.2	Registration Page	54
6.2.3	Customer Login Page	55
6.2.4	Merchant Login Page	55
6.2.5	Welcome Page	56

6.2.6	Customer Navigation Page	56
6.2.7	Merchant Navigation Page	57
6.2.8	Customer Profile Page	57
6.2.9	Merchant Profile Page	58
6.2.10	Merchant QR Code	58
6.2.11	Customer Payment Page	59
6.2.12	Data Transfer	59
6.2.13	Data Traveling through Virtual Mobiles	60
6.2.14	Data Reached Bank and Verified	60
6.2.15	Received QR code	61
6.2.16	Customer Purchase QR code	61
6.2.17	Merchant Scans the QR code	62
6.2.18	Acknowledge Message to Merchant	62
6.2.19	Merchant Transaction History	63
6.2.20	Wallet Page	63
6.2.21	Disaster Information Page	64
6.2.22	Disaster Update Page	64

LIST OF TABLES

Table No.	Title	Page No.
4.1	Software Technologies	37
6.3.1	Disaster Table	65
6.3.2	Merchant Detail Table	65
6.3.3	Customer Detail Table	65

LIST OF ABBREVIATIONS

JSP	Java Server Pages
IP	Internet Protocol
HTTP	Hyper Text Transfer Protocol
JDK	Java Development Kit
WANET	Wireless Ad hoc Network
SQL	Structured Query Language
IDE	Integrated Development Environment
JDBC	Java Data Base Connectivity
URL	Uniform Resource Locator
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
JVM	Java Virtual Machine
API	Application Programming Interface

CHAPTER 1

INTRODUCTION

1.1 Overview of the Project

The main aim of this project is to provide infrastructure less Wireless Ad hoc Networks (WANET) to enable transactions that permit customers to shop in disaster areas. We developed an application in Android which makes this payment possible. In order to transfer the payment information from customers to bank and vice versa, we use transceivers and mobile phones as intermediates. The payment information is first encoded with DES algorithm and then converted into text like format using Base64 algorithm. We also use QR code for easy transaction and security purpose.

1.2 Need for the Project

Current payment system allow the customer to purchase only during availability of the cellular network or internet connection. But it does not support during network failure. During disaster time there will not be any cellular or landline connectivity and so current payment systems will not be working. At that time, although real cash is considered to be the easiest means for carrying out a payment, cash may not be available since access to a bank is restricted both physically and electronically even though he has cash in his bank account. So the user need a payment system which helps to access the bank without the availability of network and make payment.

1.3 Objective of the Project

The main goal of the project is to provide an infrastructure less Wireless Ad hoc Network (WANET) to make Mobile payment possible. So we introduce Multilevel Endorsement (MLE) mechanism to provide payment guarantees for a

customer-to-merchant transaction with a lightweight scheme based on Bloom filter and Merkle tree to reduce communication overheads. Our mobile payment system achieves secure transaction by adopting various schemes such as Location-based Mutual Monitoring scheme and Blind Signature, while our newly introduced event chain mechanism prevents double spending attacks.

1.4 Scope of the Project

The scope of this project is to enhance the Wireless Adhoc Network (WANET) by installing mobile application to both customer and merchant. The communication between customer and merchant makes the money to transfer in disaster period where the network is down. So it will enable the online shopping system even in disaster periods. Our mobile payment also have security features which prevents the attacks from third party user by double spending the QR code. The transfer of data between the customer and merchant will be completed over a period of time which ensures reliability and sustainability in the disaster periods. This project uses the transceiver to connect the node to find the path for communication which can be updated with more nodes in future.

1.5 Existing System

Mobile payment system in a disaster area have the potential to provide electronic transactions for people purchasing goods like foodstuffs, clothes, and medicine. In existing mobile payment system, payment can be made only when communication infrastructures (such as wired networks and cellular networks) is available. But in disaster areas, communication networks may not be available and so transaction cannot be completed. Also, it is impossible to get physical cash, which a customer needs to pay for the item being purchased since access to the bank is restricted due to road blocks and there will not be direct connection to the bank for transaction. Hence, a customer cannot make a direct transaction with the merchant.

1.6 Proposed System

We proposed a payment system that will provide the infrastructure less network between user and bank during disaster time using WANET network. We use transceivers for the communications between mobiles and banks in order to cover a wider range of communication. The Customer mobiles passes the payment messages consisting of merchant information and his payment amount across the mobiles using Transceiver as a QR code which contains the payment info. We use QR code transaction with a digitally signed photograph which is used for authentication and to restrict an attacker from carrying duplicate QR code transaction. This information bounces to other mobiles until it reaches the Bank and the Bank response message reaches the customer using same strategy.

In order to provide payment guarantees for merchant we use Multilevel Endorsement Mechanism (MLE) for customer-to-merchant. To reduce communication overheads we used lightweight scheme based on Bloom filter and Merkle tree. The merchant scans the QR-code and sends information back to the bank and the finally the amount is credited. At last we provide regional situation updates for the users to broadcast any information such as Roadblocks, Bridge Collapse and flood affected areas.

CHAPTER 2

LITERATURE SURVEY

2.1 An Endorsement-based Mobile Payment System for a Disaster Area

Author:

Babatunde Ojetunde, Naoki Shibata, Juntao Gao, Minoru Ito.

Abstract:

A payment system in a disaster area is essential for people to buy necessities such as groceries, clothing, and medical supplies. However, existing payment systems require the needed communication infrastructures (like wired networks and cellular networks) to enable transactions, so that these systems cannot be relied on in disaster areas, where these communication infrastructures may be destroyed. In this paper, we propose a mobile payment system, adopting infrastructure less Mobile Ad hoc Networks (MANETs), which allow users to shop in disaster areas while providing secure transactions. Specifically, we propose an endorsement-based scheme to guarantee each transaction and a scheme to provide monitoring based on location information, and thus achieve transaction validity and reliability. Our mobile payment system can also prevent collusion between two parties and reset and recover attacks by any user. Security is ensured by using location-based mutual monitoring by nearby users, avoiding thereby double spending in the system [1].

Date of Publish: 30th April 2015

Pros : Endorsement-based mechanism provides guaranteed payment .

Cons: Uses only MANET network for communication which works only when all users uses the application.

2.2 Implication of Secure Micropayment System Using Process Oriented Structural Design by Hash chaining in Mobile Network

Author:

Chitra Kiran N, Dr. G. Narendra Kumar

Abstract:

The proposed system presents a novel approach of designing a highly secured and robust process oriented architecture for micropayment system in wireless adhoc network. Deployment of any confidential transaction over dynamic nature of wireless adhoc network will strike a high amount of security challenges which is very difficult to identify which poses a great difficulty in designing and effective countermeasures. The current work designs the security process using hash chain and Simple Public Key Infrastructure to be implemented on newly designed digital agreement of broker along with paving new secure routing for secure m-transaction as an efficient alternative for digital coin. The system stimulates the intermediate nodes to cooperate for facilitating secure and reliable transaction from source to destination nodes. The system consists of high end encryption using hash function is also independent of any Trusted Third Party when the network topology frequency changes, thereby it is flexible, lightweight, and reliable for secure micropayment systems. The analysis result shows the system is highly robust and secure ensuring anonymity, privacy, nonrepudiation offline payment system over wireless adhoc network.

Date of Publish: 30th January 2012

Pros : Uses WANET for data transfer which is be flexible and reliable.

Cons: Only Hash algorithm is used for security purpose.

2.3 An efficient and secure mobile payment protocol for restricted connectivity scenarios in Vehicular Ad hoc Network

Author:

Wenmin Li, Qiaoyan Wen, Qi Su, Zhengping Jin

Abstract:

Value-added applications in vehicular ad hoc network (VANET) come with the emergence of electronic trading. The restricted connectivity scenario in VANET, where the vehicle cannot communicate directly with the bank for authentication due to the lack of internet access, opens up new security challenges. Hence a secure payment protocol, which meets the additional requirements associated with VANET, is a must. In this paper, we propose an efficient and secure payment protocol that aims at the restricted connectivity scenario in VANET. The protocol applies self-certified key agreement to establish symmetric keys, which can be integrated with the payment phase. Thus both the computational cost and communication cost can be reduced. Moreover, the protocol can achieve fair exchange, user anonymity and payment security [3].

Date of Publish: 10th September 2011

Pros : Introduced an new protocol for data transfer in VANET network which increases security and efficiency.

Cons : Utilizes only VANET network for communication. VANET network can be created only using vehicles.

2.4 Off-line Micro-payment Protocol for Multiple Vendors in Mobile Commerce

Author:

Xiaoling Dai, Oluwatomi Ayoade and John Grundy.

Abstract:

Micro-payment systems have the potential to provide non-intrusive, high-volume and low-cost pay-as-you-use services for a wide variety of web-based applications. Previously we have developed NetPay, an off-line micro-payment protocol for E-commerce. In this paper we describe a set of extensions, Mobile-NetPay, optimised for mobile E-commerce. Mobile-NetPay provides high performance and security using one-way hashing functions for e-coin encryption. Each mobile user's transaction does not involve any broker and double spending is detected during the redeeming transaction. We describe the motivation for Mobile-NetPay and describe three transactions of the Mobile-NetPay protocol in detail to illustrate the approach. We then discuss future research on this protocol [4].

Date of Publish: 19 December 2006

Pros : The mesh users utilize heterogeneous wireless devices like laptop, mobile phone etc. and the mesh routers may need not require internet.

Cons : Wireless Mesh Networks stores routing information for data delivery to reach destination. If nodes are in constant movement, the mesh routers will spend more time updating routing table than delivering data.

2.5 Wireless Mesh Networks: A Survey

Author:

Ian F. Akyildiz, Xudong Wang, Weilin Wang.

Abstract:

Wireless mesh networks (WMNs) consist of mesh routers and mesh clients, where mesh routers have minimal mobility and form the backbone of WMNs. They provide network access for both mesh and conventional clients. The integration of WMNs with other networks such as the Internet, cellular, IEEE 802.11, IEEE 802.15, IEEE 802.16, sensor networks, etc., can be accomplished through the gateway and bridging functions in the mesh routers. Mesh clients can be either stationary or mobile, and can form a client mesh network among themselves and with mesh routers. WMNs are anticipated to resolve the limitations and to significantly improve the performance of ad hoc networks, wireless local area networks (WLANs), wireless personal area networks (WPANs), and wireless metropolitan area networks (WMANs). They are undergoing rapid progress and inspiring numerous deployments. WMNs will deliver wireless services for a large variety of applications in personal, local, campus, and metropolitan areas. Despite recent advances in wireless mesh networking, many research challenges remain in all protocol layers. This paper presents a detailed study on recent advances and open research issues in WMNs. System architectures and applications of WMNs are described, followed by discussing the critical factors influencing protocol design. Theoretical network capacity and the state of the art protocols for WMNs are explored with an objective to point out a number of open research issues. Finally, testbeds, industrial practice and current standard activities related to WMNs are highlighted.

Date of Publish: 1st January 2005

Pros : Off-line micro-payment protocol called Mobile-NetPay provides high performance and security in wireless network.

Cons : This mobile applications need a fixed infrastructure network for mobile payments. So these mobile applications can't be used in infrastructure less areas.

2.6 Feasibility Study

Feasibility study is the initial design stage of any project, which brings together the elements of knowledge that indicate if a project is possible or not. The types of feasibility made in our project are

2.6.1 Technical Feasibility

2.6.2 Operational Feasibility

2.6.3 Resource Feasibility

2.6.4 Financial Feasibility

2.6.1 Technical Feasibility

The technical requirements are essential to implement the proposed system. In this project, we need client-server environment where the clients are mobile applicants and the servers are bank used for transaction. The language used in client side applications is android and it is an open source. The language used is Java J2ME. The server used is Apache Tomcat server which is also an open source application. We use MYSQL as back-end and tool used is MYSQL-Front. MYSQL-Front is open source software. NETBEANS IDE is used to create a virtual devices in the network. DES and Base64 algorithms are used for security purpose. So these technologies are feasible to implement in our project.

2.6.2 Operational Feasibility

The proposed system describes the offline mobile payment system using WANET. To implement WANET, we use transceivers and mobile and form the network. These transceivers is similar to router which transfer the data to other wireless devices. The mobile nodes are deployed virtually to transfer data. The bank transaction is performed creating virtual bank using tomcat server. The application is create using android which is an open source. So the proposed system can be developed using the above mentioned tools and technologies.

2.6.3 Resource Feasibility

For the completion of the project man power and knowledge is required. Since the application development is done using the open source software and having prior knowledge in the technology we can able to complete the project on time and budget. During deployment of the project, original transceiver may not be used since its cost is not affordable, we can use routers to perform its function. The application is developed in android platform and so any android users can download and use the application. Therefore the project can be developed with the resources available.

2.6.4 Financial Feasibility

In our project all the software tools are available as open source and so the cost for the development and deployment of the project is feasible. Hardware components such as transceiver, mobile phones, and computer are also feasible. The bar code reader application used is also as an open source. Therefore financially the project is feasible for development.

CHAPTER 3

SYSTEM DESIGN

3.1 Introduction

We proposed a payment system that will provide the infrastructure less network between User and Bank during disaster time using WANET network. We use Transceivers for the communications between Mobiles and Banks in order to cover a wider range of communication. The Customer mobiles passes the payment messages consisting of merchant information and his payment amount across the mobiles using Transceiver as a QR code which contains the payment information. We use QR code transaction with a digitally signed photograph which is used for authentication and to restrict an attacker from carrying duplicate QR code transaction. This information bounces to other mobiles until it reaches the Bank and the Bank response message reaches the customer using same strategy.

In order to provide payment guarantees we use Endorsement-based Mechanism for a customer-to-merchant transaction and a Multilevel Endorsement (MLE) mechanism to reduce communications overheads. The Merchant scans the QR-code and sends information back to the Bank and the finally the amount is credited. At last we propose Surrounding Monitoring Panel for the users to broadcast any disturbing information such as Roadblocks, Bridge Collapse and flood affected areas.

3.2 Architecture Diagram

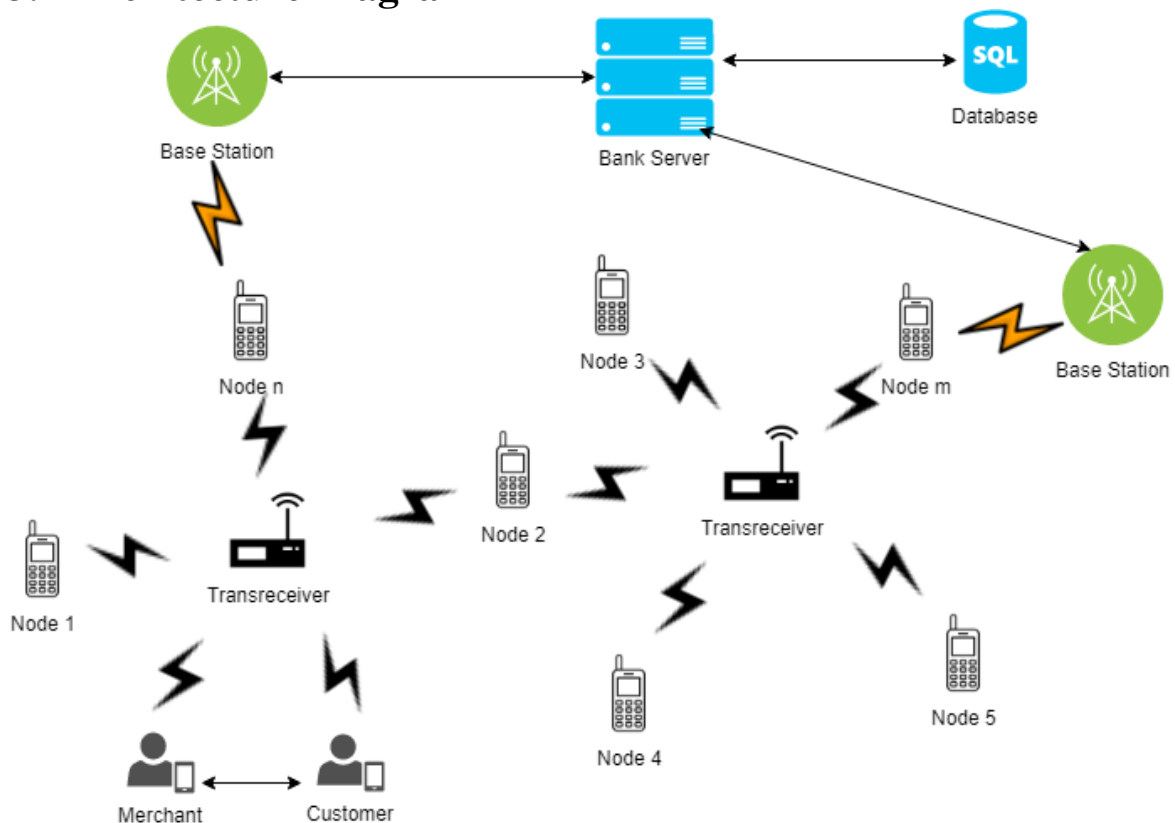


Fig 3.2 Architecture Diagram

3.2.1 Architecture Description

The mobile phones are initially connected to the nearest wireless devices i.e. transceivers. Some mobiles are present in the range of two or more transceivers (overlapping region) and some other mobiles will have internet connectivity. So, this network created using wireless devices is called WANET. Whenever merchant or customer proceed to payment, first the data travel to the nearest transceiver which will pass the message to other mobiles connected in transceiver. The mobiles which are present in the overlapping area of transceivers will transfer the message to another transceiver. This process will be repeated until it reaches the mobile (destination) which has internet facility. By using the internet, transaction data reaches the bank server. Bank server verifies the user details and it transfers to user again by using mobile nodes which are connected to the transceiver. Finally, the user details are verified for mobile payment.

3.3 UML Diagrams

The Unified Modeling Language (UML) was created to forge a common, semantically and syntactically rich visual modeling language for the architecture, design, and implementation of complex software systems both structurally and behaviorally. UML has applications beyond software development, such as process flow in manufacturing.

It is analogous to the blueprints used in other fields, and consists of different types of diagrams. In the aggregate, UML diagrams describe the boundary, structure, and the behavior of the system and the objects within it.

UML is not a programming language but there are tools that can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design.

Purpose of UML Diagrams

Providing system architects, software engineers, and software developers with tools for analysis, design, and implementation of software-based systems as well as for modelling business and similar processes.

Advancing the state of the industry by enabling object visual modelling tool interoperability. However, to enable meaningful exchange of model information between tools, agreement on semantics and notation is required.

3.3.1 Use case Diagram

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

Use case diagrams are used to specify:

- **Requirements**, required usages of a system under design or analysis (subject) - to capture what the system is supposed to do;
- **Functionality** offered by a subject – what the system can do;
- Requirements the specified subject poses on its **environment** - by defining how environment should interact with the subject

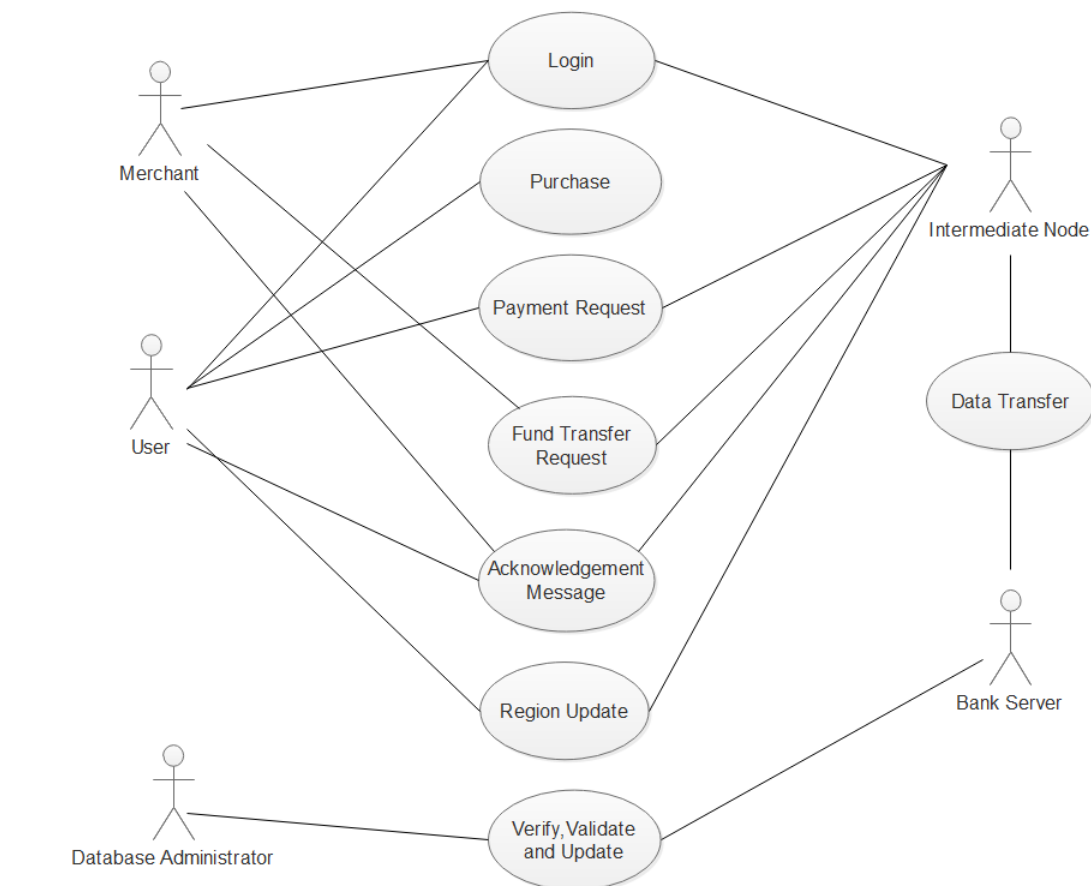


Fig 3.3.1 Use case Diagram

3.3.2 Class Diagram

Class diagram is a static diagram. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

The purpose of the class diagram can be summarized as –

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

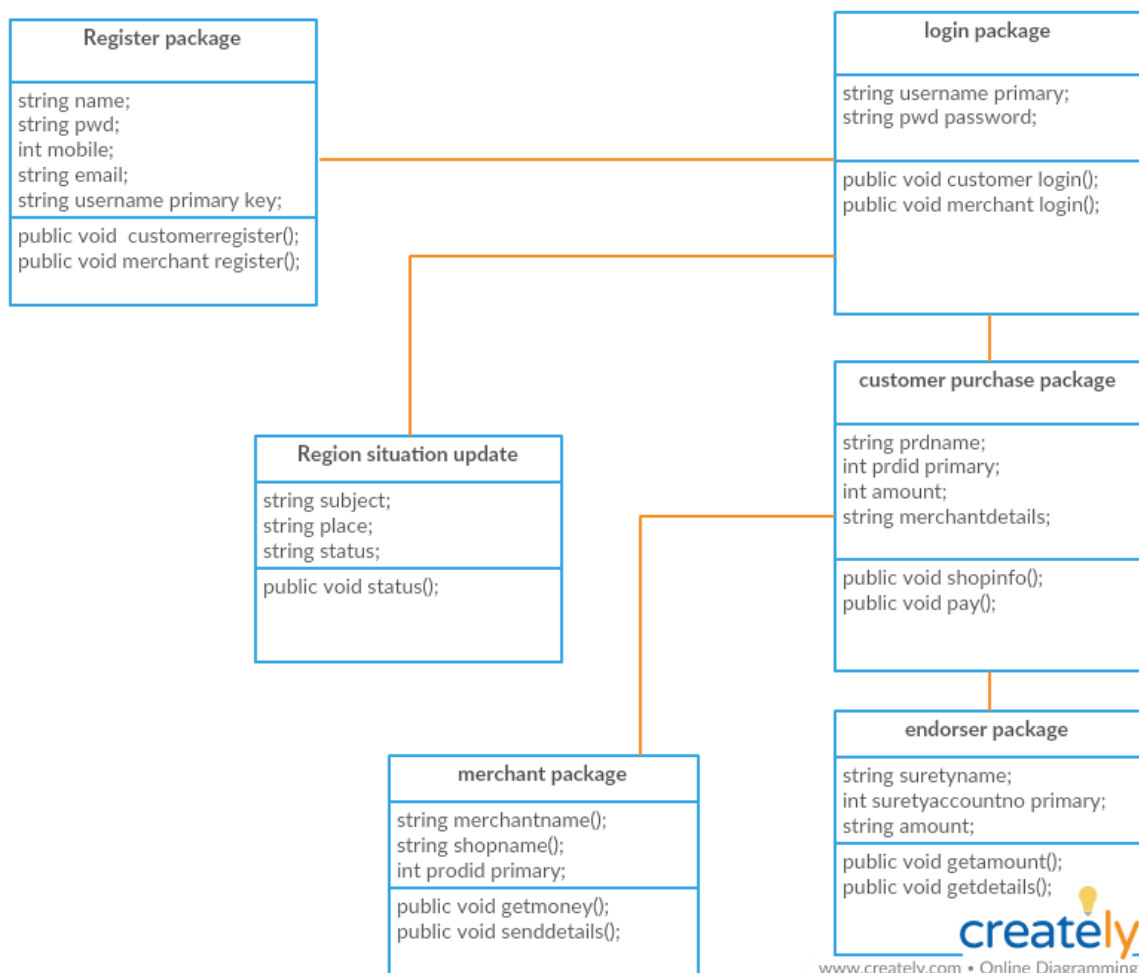


Fig 3.3.2 Class Diagram

3.3.3 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

Purpose of Sequence Diagram

- Model high-level interaction between active objects in a system
- Model the interaction between object instances within a collaboration that realizes a use case
- Model the interaction between objects within a collaboration that realizes an operation
- Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

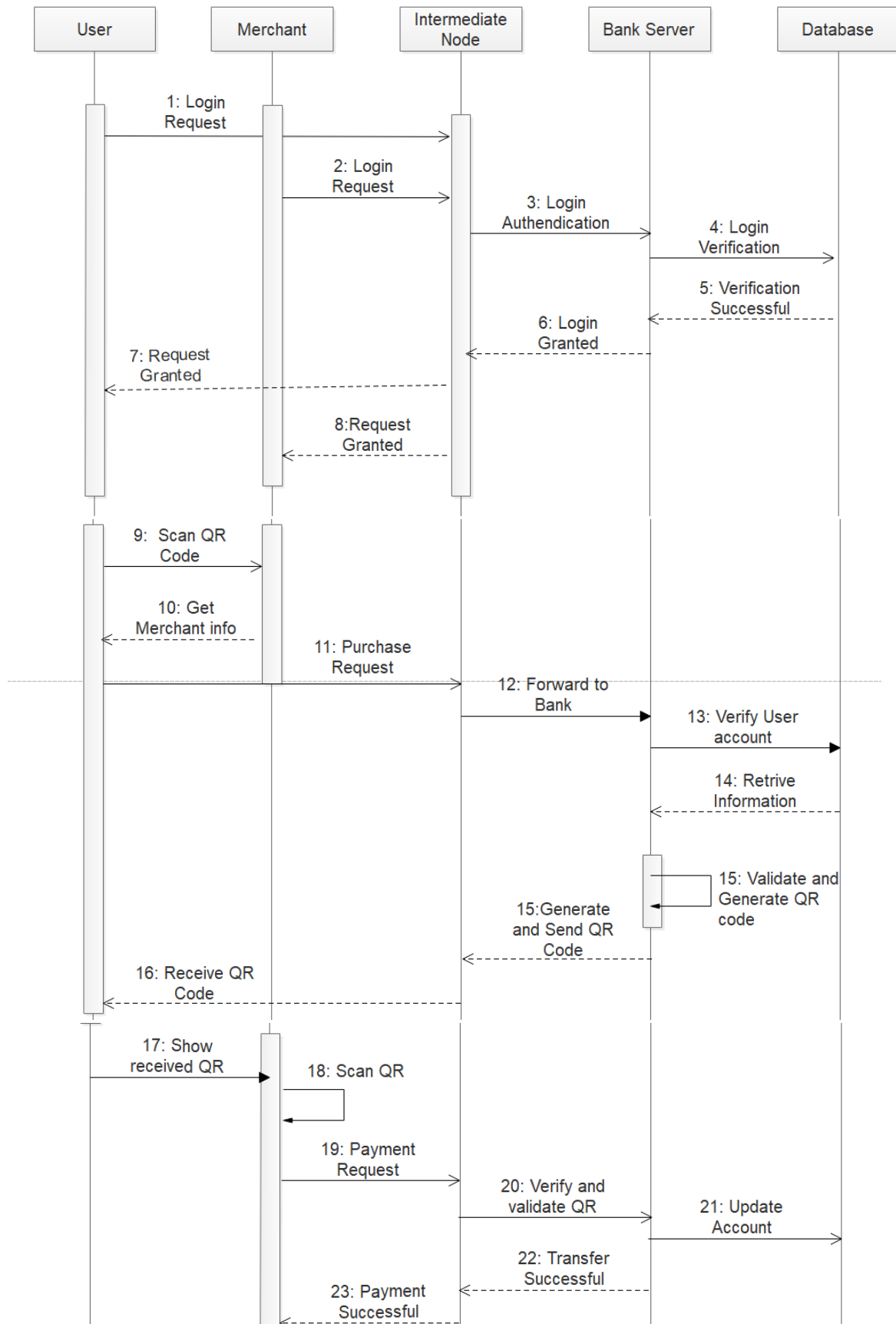


Fig 3.3.3 Sequence Diagram

3.3.4 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

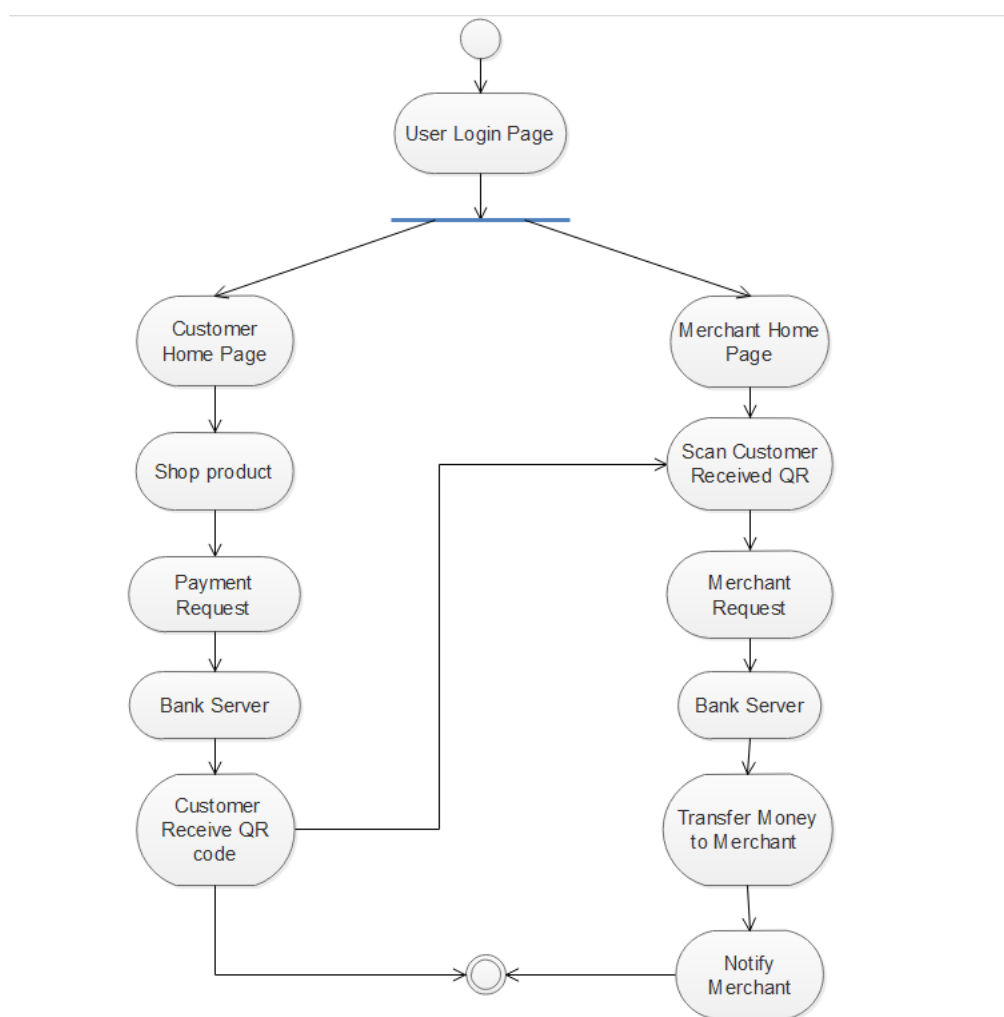


Fig 3.3.4 Activity Diagram

3.3.5 Collaboration Diagram

A collaboration diagram is a type of visual presentation that shows how various software objects interact with each other within an overall IT architecture and how users can benefit from this collaboration. A collaboration diagram often comes in the form of a visual chart that resembles a flowchart. It can show, at a glance, how a single piece of software complements other parts of a greater system. A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML).

Collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. Objects are shown as rectangles with naming labels inside. These labels are preceded by colons and may be underlined. The relationships between the objects are shown as lines connecting the rectangles. The messages between objects are shown as arrows connecting the relevant rectangles along with labels that define the message sequencing.

The purpose of interaction diagram is

- To capture the dynamic behavior of a system.
- To describe the message flow in the system.
- To describe the structural organization of the objects.
- To describe the interaction among objects.

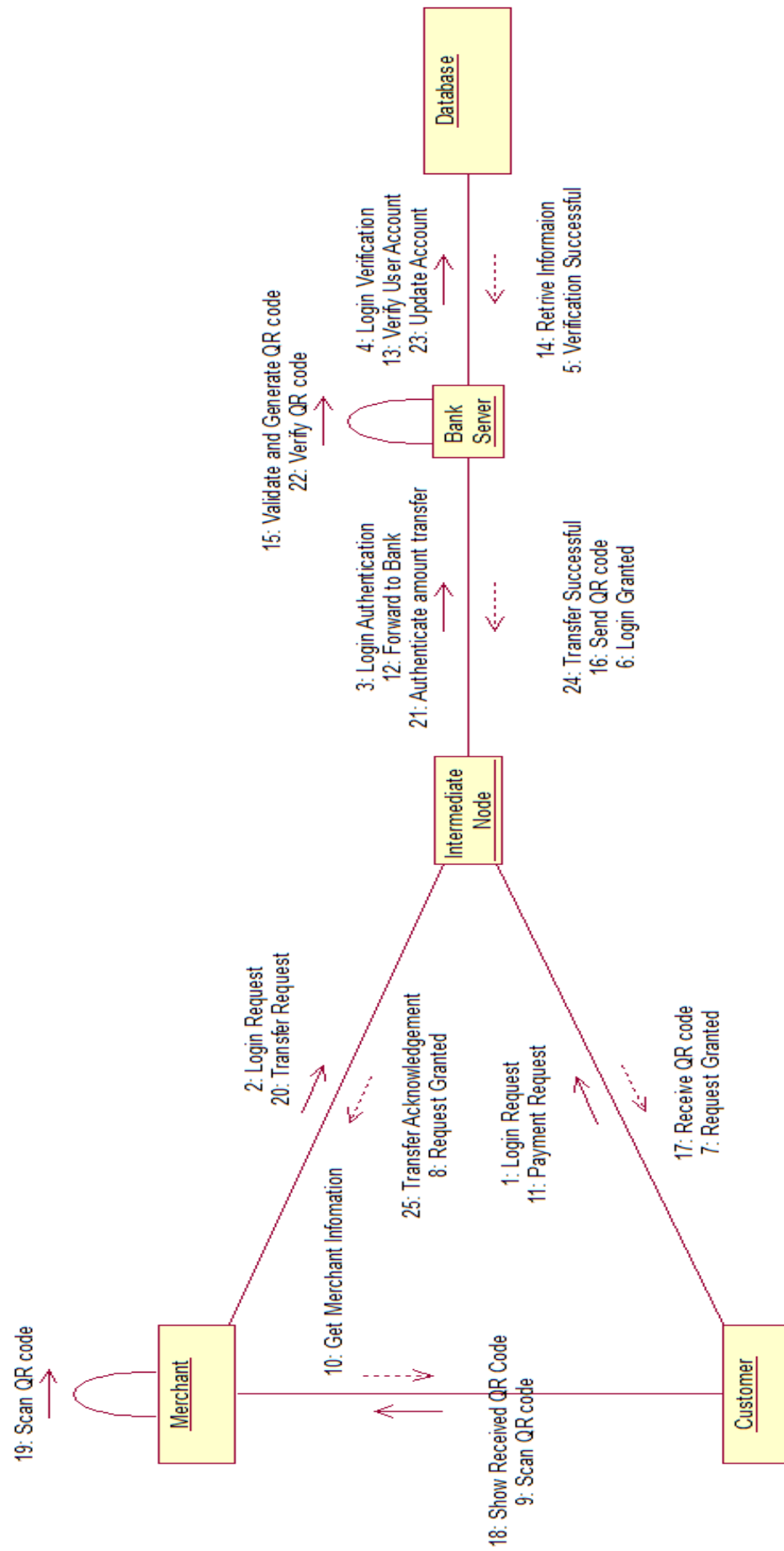


Fig 3.3.5 Collaboration Diagram

CHAPTER 4

REQUIREMENT SPECIFICATIONS

4.1 Introduction

The requirements specification is a technical specification of requirements for the software products. It is the first step in the requirements analysis process and it lists the requirements of a particular software system including functional, performance and security requirements. The requirements also provide usage scenarios from a user, an operational and an administrative perspective. The purpose of software requirements specification is to provide a detailed overview of the software project, its parameters and goals. This describes the project target audience and its user interface, hardware and software requirements. It defines how the client, team and audience see the project and its functionality.

4.2 Hardware and Software Specification

4.2.1 Hardware Requirements

Windows:

- | | |
|--------------|--------------------|
| 1. Hard disk | : 500 GB and above |
| 2. Processor | : i3 and above |
| 3. RAM | : 4GB and above |

Smart Phone:

- | | |
|--------------|-----------------------|
| 1. Memory | : 4GB and above |
| 2. Processor | : Dual core and above |
| 3. RAM | : 1GB and above |

4.2.2 Software Requirements

Windows:

1. Operating System : Windows 7 and above (64-bit)
2. Java version : JDK 1.7
3. Web server : Tomcat 6.20 and Tomcat 7.0.11
4. Database : MySQL

Smart Phone:

1. Operating System : Gingerbread and above
2. No. of Device : 2(at least)

4.3 Software Technologies

4.3.1 JAVA

In this project, JSP is used for the creation of bank web page. This acts as a bank interface for users to add details to bank database. Then Android applications are developed using java programming language in Android Eclipse. Java program is also used to create virtual mobile nodes in the system to show the data transfer,

Introduction

Java is a platform independent language which is write once and run anywhere. Java is a programming language created by James Gosling from Sun Microsystems (Sun) in 1991. The target of Java is to write a program once and then run this program on multiple operating systems. The first publicly available version of Java (Java 1.0) was released in 1995. Sun Microsystems was acquired by the Oracle Corporation in 2010. Oracle has now the steermanship for Java. In 2006 Sun started to make Java available under the GNU General Public License

(GPL). Oracle continues this project called OpenJDK. Over time new enhanced versions of Java have been released. The current version of Java is Java 1.8 which is also known as Java 8. Java is defined by a specification and consists of a programming language, a compiler, core libraries and a runtime (JVM). The Java runtime allows software developers to write program code in other languages than the Java programming language which still runs on the Java virtual machine.

Features of Java

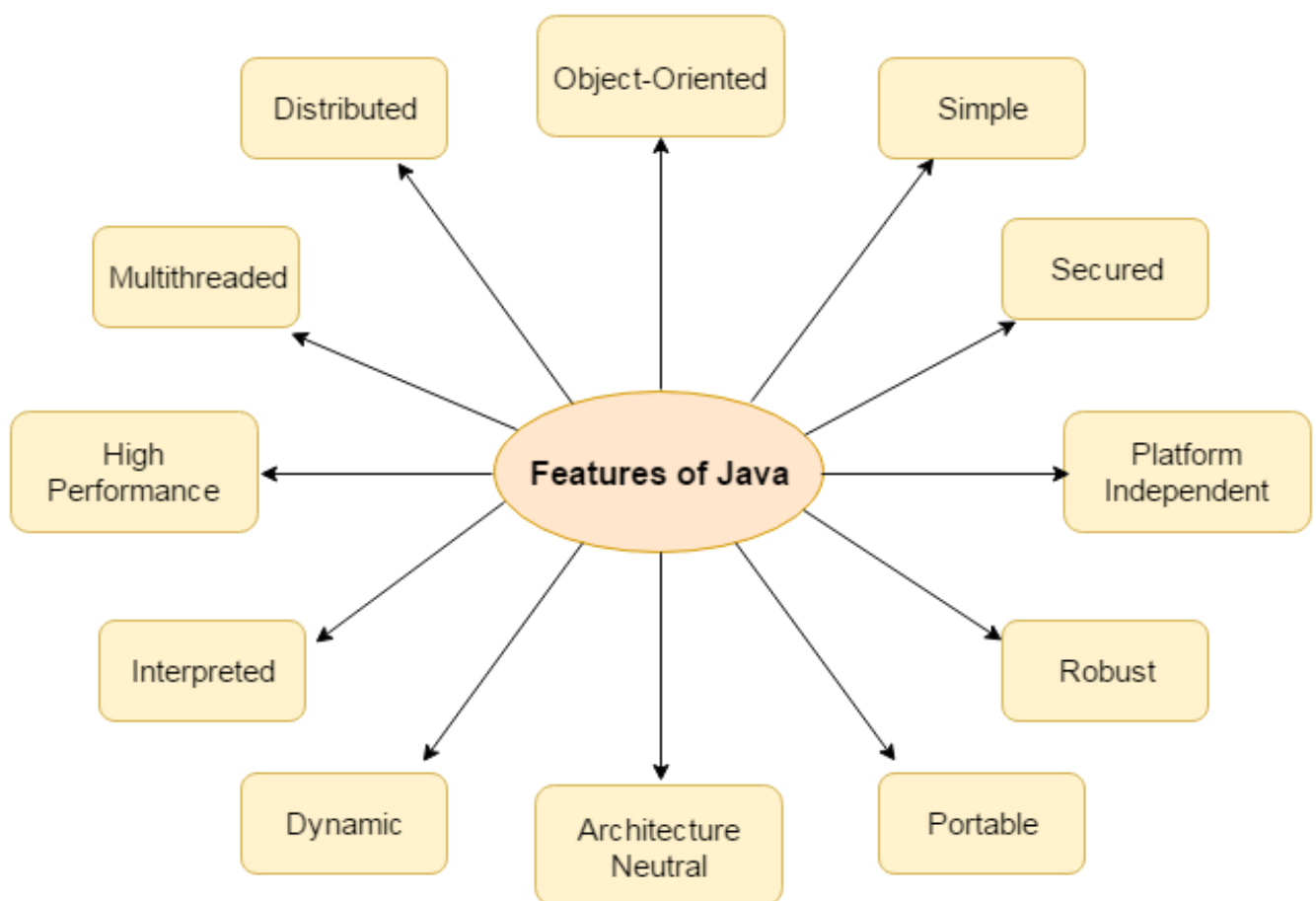


Fig 4.3.1 (a) Java Features

Java as Programming Language

Java is a high-level programming language. Java program is both compiled and interpreted. First the Java Source code is saved as .java file. With the help of a compiler, we translate the .java file into .class file. Then interpreter converts the class file into an intermediate language called Java byte codes. This Java byte code can be executed in computer using JVM (Java Virtual Machine).

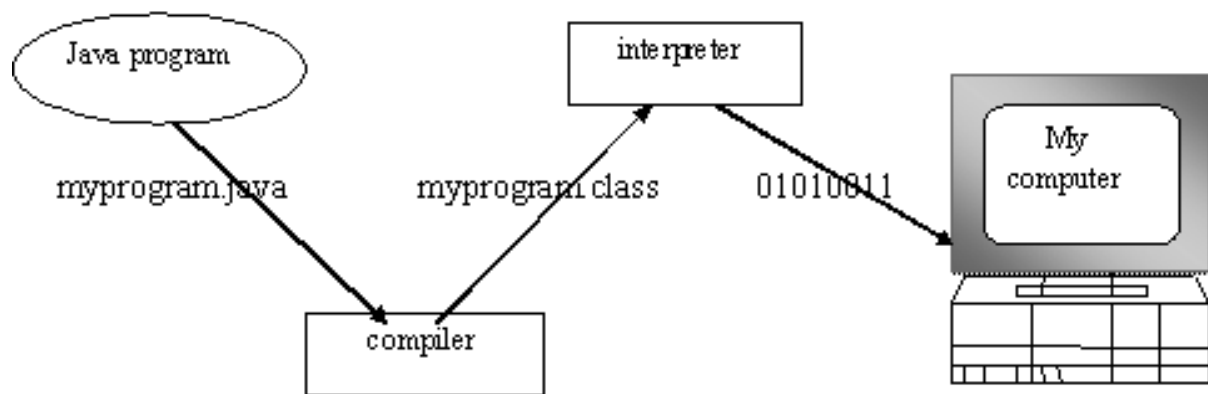


Fig 4.3.1 (b) Working of Java

Java as a Platform

A platform is the hardware or software environment in which a program runs. The Java platform is a software based platform that runs on top of other hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components:

- The Java Virtual Machine (JVM)
- The Java Application Programming Interface (Java API)

Java Applications

- Standalone Application
- Web Application
- Enterprise Application
- Mobile Application

4.3.2 JDBC

JDBC is used to connect java application with the database. It acts as an interface between the application and database. In this project JDBC is used to connect the MySQL Front database with the application. The database holds the information about the user bank details and disaster details.

Introduction

Java Database Connectivity (JDBC) is an Application Programming Interface (API) for the programming language which defines how a client may access a database. It is developed by Oracle Corporation and Sun microsystems released JDBC as part of Java Development Kit 1.1 on February 19, 1997. It is Java based data access technology and used for java database connectivity.

Functionality

JDBC allows multiple implementations to exist and be used by the same application. The API provides a mechanism for dynamically loading the java packages and registering them with the JDBC driver manager. The driver manager is used as a connection factory for creating JDBC connections. JDBC connections support creating and executing statements. This may include SQL statement such as CREATE, UPDATE, INSERT, DELETE and SELECT.

Types of JDBC Drivers

TYPE-1:

TYPE-1 calls native code of the locally available ODBC driver.

TYPE-2:

TYPE-2 calls database vendor native library on a client side. This code then talks to database over the network.

TYPE-3:

TYPE-3, the pure-java driver that talks with the server side middleware that then talks to the database.

TYPE-4:

TYPE-4, the pure-java driver that uses database native protocol.

Steps to Connect Database in Java

1. Register the driver class

- The `forName` method of `Class` class is used to register the driver class. This method is used to dynamically load the driver class.
- **Syntax:** `Public static void forName(String className)` throws `ClassNotFoundException`

2. Create the connection object

- The `getConnection` method of `DriverManager` class is used to establish connection with the database.
- **Syntax:** `public static Connection getConnection(String url)` throws `SQLException`

3. Create the statement object

- The createStatement method of connection interface is used to create statement. The object of statement is responsible to execute queries with the database.
- **Syntax:** public Statement createStatement() throws SQLException

4. Execute query

- The executeQuery method of statement interface is used to execute queries to the database.
- **Syntax:** public ResultSet executeQuery(string sql) throws SQLException

5. Close the connection object

- By closing connection object statement and resultset will be closed automatically. The close method of connection interface is used to close the connection.
- **Syntax:** public void close() throws SQLException

4.3.3 JSP

In this project JSP is used to create the bank server page. Since original bank web sites cannot be used, we create a duplicate web pages to acts as bank pages create account and enter user information.

Introduction

Java server pages is a technology that helps software developers create a dynamically generated web pages based on HTML, XML or other document types. It is released in 1999 by Sun microsystems with .jsp as a filename extension. JSP is similar to PHP and ASP, but it uses the java programming

language. To deploy and run java server pages, a compatible web server with a servlet container, such as Apache Tomcat is required.

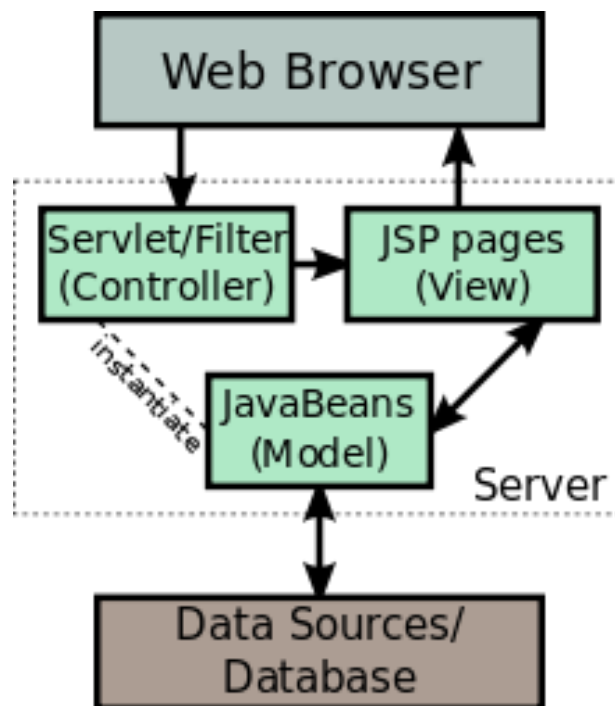


Fig 4.3.3 JSP Model Architecture

Overview of JSP

JSP may be viewed as a high-level abstraction of java servlets. JSPs are translated into servlets at run time, therefore JSP is a servlet and JSP is reused until the original JSP is modified. JSP allows java code and certain predefined actions to be interleaved with static web markup contents, such as HTML, with page being compiled. The compiled pages, as well as any dependent java libraries, contain java byte code rather than machine code. They must be executed within a Java Virtual Machine (JVM).

Syntax:

The elements of JSP have been described below:

The Scriptlet

- A scriptlet may contain any number of JAVA language statements, variables or method declarations or expressions that are valid.
- Syntax : **<jsp:scriptlet> code fragment </jsp:scriptlet>**

JSP Declarations

- A declaration declares one or more variables or methods that we can use in java code later in the JSP file.
- Syntax : **<jsp:declaration>code fragment</jsp:declaration>**

JSP Expression

- A JSP expression element contains a scripting language expression that is evaluated, converted to a string, and inserted where the expression appears in the JSP file.
- Syntax: **<jsp:expression>expression</jsp:expression>**

JSP Comments

- JSP comments mark text or statements that JSP container should ignore. It is useful when we want to hide a part of your JSP page.
- Syntax : **< %--comment-- %>**

JSP directives

- JSP directive affects all the overall structure of the servlet class.
- Syntax: **<% @page... %>**

Lifecycle of a JSP Page

1. Translation of JSP page
2. Compilation of JSP page
3. Class loading
4. Instantiation
5. Initialization
6. Request processing
7. Destroy

Advantages of JSP

1. Less code than servlet
2. Easy to maintain
3. Fast development

4.3.4 Android

In this project, application is developed in android platform since it is used by most of the people. Android is more popular than any other mobile operating system and user friendly. Moreover it is an open source and all the required tools are available as open source.

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open source software and designed primarily for touchscreen mobile devices such as smartphones and tablets. In addition, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics.

Interface

Android's default user interface is mainly based on direct manipulation, using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, along with a virtual keyboard. Game controllers and full-size physical keyboards are supported via Bluetooth or USB. The response to user input is designed to be immediate and provides a fluid touch interface, often using the vibration capabilities of the device to provide haptic feedback to the user. Internal hardware, such as accelerometers, gyroscopes and proximity sensors are used by some applications to respond to additional user actions, for example adjusting the screen from portrait to landscape depending on how the device is oriented, or allowing the user to steer a vehicle in a racing game by rotating the device, simulating control of a steering wheel.

4.3.5 Apache Tomcat

Apache Tomcat acts as the local bank server in this project. Real time bank servers cannot be used for project demo since it is tedious process, we used it as a server local server and interact with the application.

Introduction

Tomcat, developed by Apache, is a standard reference implementation for Java servlets and JSP. It can be used standalone as a web server or be plugged into a web server like Apache, Netscape Enterprise Server, or Microsoft Internet Information Server. To start Tomcat, it has to be first set the JAVA_HOME environment variable to the JDK home directory using the following command. The JDK home directory is where your JDK is stored. On my computer, it is c:\Program Files\jdk1.6.0_24.

The apache tomcat server is an open source, java-based web application container that was created to servlets and java server page (JSP) web applications.

It was created under the apache-Jakarta subproject; however, due to its popularity, it is now hosted as separate apache project, where it is supported and enhanced by a group of volunteers from the open source java community.

Tomcat Manager Web Application

The Tomcat manager web application is packaged with the tomcat server. It is installed in the context path of manager and provides the basic functionality to manage web applications running in the tomcat server from any web browser. Some of the provided functionality includes the ability to install, start, stop, remove, and report on web applications.

- **The Server**

The first container element referenced in this snippet is the `<Server>` element. It represents the entire Catalina servlet engine and is used as a top-level element for a single Tomcat instance. The `<Server>` element may contain one or more `<Service>` containers.

- **The Service**

The next container element is the `<Service>` element, which holds a collection of one or more `<Connector>` elements that share a single `<Engine>` element. N-number of `<Service>` elements may be nested inside a single `<Server>` element.

- **The Connector**

The next type of element is the `<Connector>` element, which defines the class that does the actual Handling requests and responses to and from a calling client application.

- **The Engine**

The third container element is the `<Engine>` element. Each defined `<Service>` can have only one `<Engine>` element and this single `<Engine>`

component handles all requests received by all of the defined `<Connector>` components defined by a parent service.

- **The Host**

The `<Host>` element defines the virtual hosts that are contained in each instance of a Catalina `<Engine>`. Each `<Host>` can be a parent to one or more web applications, with each being represented by a `<Context>` component.

- **The Context**

The `<Context>` element is the most commonly used container in a Tomcat instance. Each `<Context>` element represents an individual web application that is running within a defined `<Host>`. There is no limit to the number of contexts that can be defined within a `<Host>`.

Java Web Applications

The main function of the Tomcat server is to act as a container for Java web applications. Therefore, before we can begin our Tomcat-specific discussion, a brief introduction as to exactly what web applications are is in order. The concept of a web application was introduced with the release of the Java servlets specification. According to this specification, “a web application is a collection of servlets, html pages, classes, and other resources that can be bundled and run on multiple containers from multiple vendors.” What this really means is that a web application is a container that can hold any combination of the following list of objects:

- Servlets
- Java Server Pages (JSPs)
- Utility classes
- Static documents, including HTML, images, JavaScript libraries, cascading style sheets, and so on
- Client-side classes

One of the main characteristics of a web application is its relationship to the Servlets Context. Each web application has one and only one Servlets Context. This relationship is controlled by the servlets container and guarantees that no two web applications will clash when accessing objects in the Servlets Context.

4.3.6 MYSQL

MYSQL is used as database to store the user bank information and application related information. MYSQL is easy to handle and JDBC can be used easily. MYSQL acts as back end in the project.

Introduction

MySQL is an open source Relational DataBase Management System (RDBMS). Its name is the combination of “My” the name of co-founder Michael Widenius and SQL is the abbreviation of Structured Query Language. The MySQL development project has made its source code available under the GNU general public license. It has developed by Oracle Corporation. MySQL is fast, easy to use. MySQL is initially released on 23 May 1995(22 years ago).

Overview

MySQL is written in C and C++. Its SQL parser is written in YACC but it uses a lexical analyzer. MySQL works on many system platforms including LINUX, MACOS, MICROSOFT WINDOWS, OPEN SOLARIS NETBSD. The MySQL server software itself and the client libraries use dual-licensing distribution. It has also been tested to be a “fast, stable and true multi-user, multi-threaded SQL database server”. MySQL is a central component of the LAMP open source web application. LAMP is an acronym for LINUX, APACHE, MYSQL, PERL/PHP/PYTHON. Application that uses the MySQL includes

wordpress, Drupal, MyBB. MySQL is also used in large scale websites such as Google, Facebook, Youtube.

Features

MySQL is offered under two editions: open source MySQL community server and the enterprise server. Some of the major features in MySQL are:

- Cross platform support
- Stored procedures
- Cursors
- Information schema
- Query caching

Advantages

- Highly secure and scalable
- High performance
- High flexibility
- High productivity

Disadvantages

- MySQL version less than 5.0 does not support role ,commit and stored procedures
- MySQL does not support a very large database efficiently
- MySQL does not handle transactions very efficiently and it is prone to data corruption.

4.4 Software Tools

4.4.1 NetBeans IDE

NetBeans is an IDE tool used to run java program. To create virtual mobile nodes in the project we use NetBeans. Since it is not feasible to provide many mobile in real time perform data transfer, we create virtual mobile nodes using java program. NetBeans allows applications to be developed from a set of modular software components called modules. In addition to Java development, it has extensions for other languages like PHP, C, C++ and HTML5, Javadoc and Javascript. Applications based on NetBeans, including the NetBeans IDE, can be extended by third party developers.

Integrated Development Environment

An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. An IDE normally consists of a source code editor, build automation tools, and a debugger. Most modern IDEs have intelligent code completion. Some IDEs, such as NetBeans and Eclipse, contain a compiler, interpreter, or both but others, such as Sharp Develop and Lazarus, do not. Sometimes a version control system, or various tools to simplify the construction of a Graphical User Interface (GUI), are integrated.

Features:

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

- User interface management (e.g. menus and toolbars)
- User settings management

- Storage management (saving and loading any kind of data)
- Window management
- Wizard framework (supports step-by-step dialogs)
- NetBeans Visual Library
- Integrated development tools.

4.4.2 MYSQL Front

MYSQL Front is a tool used in the project for the storage of data. MySQL-Front is a Windows front end program for the MySQL database server. The database structure and data can be handled via dialogs or SQL commands. Import and Export in standard file formats is supported. The database server can be connected directly or via HTTP tunneling. These MySQL Front tools actually helps to work fast and easy with the MySQL database. We use MYSQL Front as the database to store the bank information, login information, and disaster situation. This act as the back end of the project. This uses the MYSQL queries to interact with the data base.

Table 4.1 Software Technologies

S. No	Software	Programming Language	Usage in Project
1.	MyEclipse	JSP(Java Server Page)	Used to create Bank Interface
2.	Tom Cat Server	Java	Used as local server
3.	Android Eclipse	Android Java	Used to create Android Application
4.	MYSQL Front	My SQL	Used as Database (Back End)

CHAPTER 5

MODULE DESCRIPTION

5.1 Introduction

A module is a software component or part of a program that contains one or more routines. One or more independently developed modules make up a program. Modules make a programmer's job easy by allowing the programmer to focus on only one area of the functionality of the software application. Modules are typically incorporated into the program (software) through interfaces.

5.2 List of Modules

The modules which are used in this project are

- 5.2.1 Bank account creation
- 5.2.2 Customer Payment Request
- 5.2.3 Merchant process
- 5.2.4 Region Situation Update

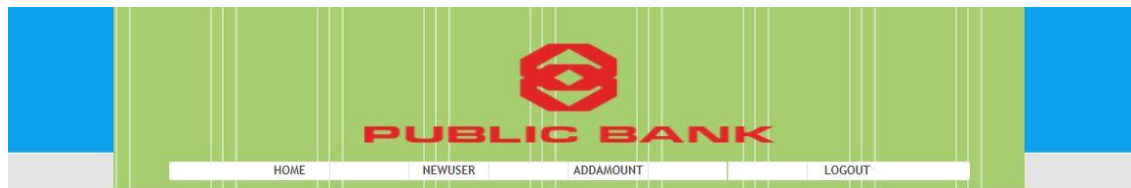
5.2.1 Bank account creation

In this module customer and merchant will create a bank account and sign up in the application. Our application need the users to hold a bank account for transaction. Additionally customers has to give their endorser name (surety). This will help the customer to withdraw amount from their endorser account if customer account has minimum balance.

5.2.1.1 Implementation of Bank Interface:

The implementation of the bank server is performed using the tool Apache Tomcat server which acts as a Local Server. The bank interface is created using JSP (Java Servlet Page). The back end is SQL and the tool used is MYSQL Front. JDBC (Java Data Base Connectivity) is used as API (Application Programming

Interface) to communicate with the database. All the bank data and application data are stored in the MYSQL.



Account Information

Account No:

Account Holder:

Balance Amount:

Mobile Number:

E Mail:

Address:

City:

PinCode:

State:

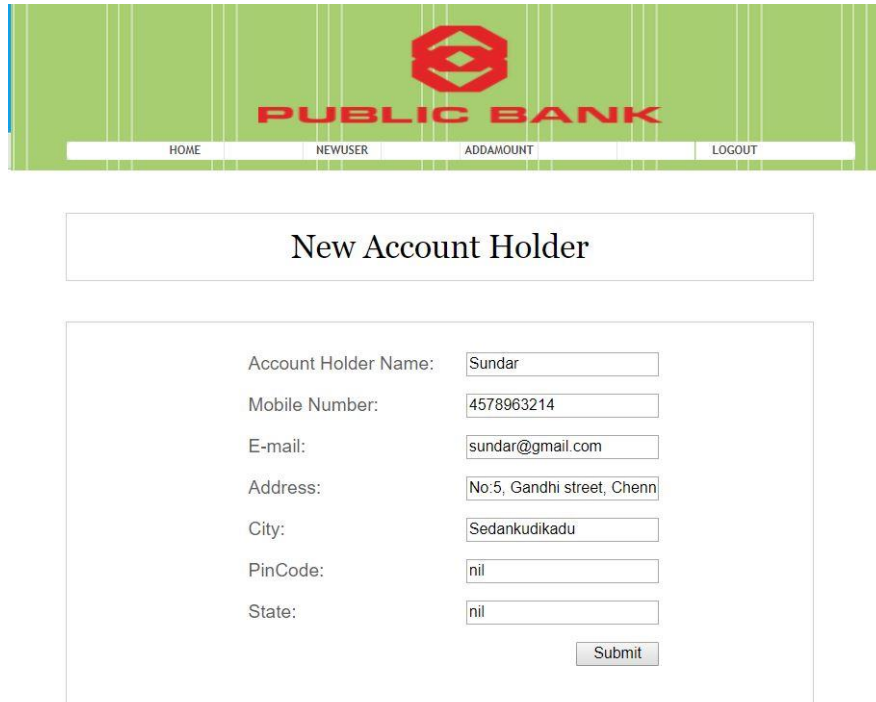
5.2.1.1 (a) Bank Interface

accountno	accountholderna...	mobilenumber	mailid	Address	city	pincode	state
01459440	vig	7895214630	dasf@gmail.com	dafa	Saidapet West(Mer1	600015	TAMIL NADU
37260311	venkat	7200375772	venkat@gmail.com	saidapet	Chennai G.P.O.	600001	TAMIL NADU
47919893	yokesh	7894561230	q@gmail.com	as	Asaveerankudikadu	621719	TAMIL NADU
52044396	venu	9874563210	venu@gmail.com	a	Lloyds Estate	600014	TAMIL NADU
56599860	Mani	9876543210	mani@gmail.com	saidapet	Chennai Race Cours	600032	TAMIL NADU
57781118	Ramesh	9876543210	ramesh@gmail.com	saidapet	Chennai Race Cours	600032	TAMIL NADU
60892966	mano	9876543210	mano@gmail.com	tambaram	Chennai G.P.O.	600001	TAMIL NADU
63766599	siva	9876543210	siva@gmail.com	saidapet	Chennai Race Cours	600032	TAMIL NADU
64897701	Aravindh	9876543210	aravindh@gmail.com	saidapet	Chennai Race Cours	600032	TAMIL NADU
75193127	praveen	9876543210	praveen@gmail.com	saidapet	Chennai G.P.O.	600001	TAMIL NADU
77266724	ram	9876543210	ram@gmail.com	saidapet	Chennai G.P.O.	600001	TAMIL NADU
82981049	maddy	9876543210	maddy@gmail.com	saidapet	Chennai G.P.O.	600001	TAMIL NADU
92479205	nish	7896541230	n@gmail.com	bss	Asaveerankudikadu	621719	TAMIL NADU
92860918	manoj	1234567890	ma@gmail.com	qwe	Karuppur Senapathy	621707	TAMIL NADU

5.2.1.1 (b) User Database

5.2.1.2 Creation of bank account:

To open an account in the bank, user has to give their name, phone number e-mail id, address, city, pin code and state. Then he has to deposit money in his account. After that he can give at most two bank account holder names as the endorsers.



The screenshot shows the Public Bank website header with a green background and a red logo. Below the header is a navigation bar with links: HOME, NEWUSER, ADDAMOUNT, and LOGOUT. The main content area is titled "New Account Holder" and contains a form with the following fields:

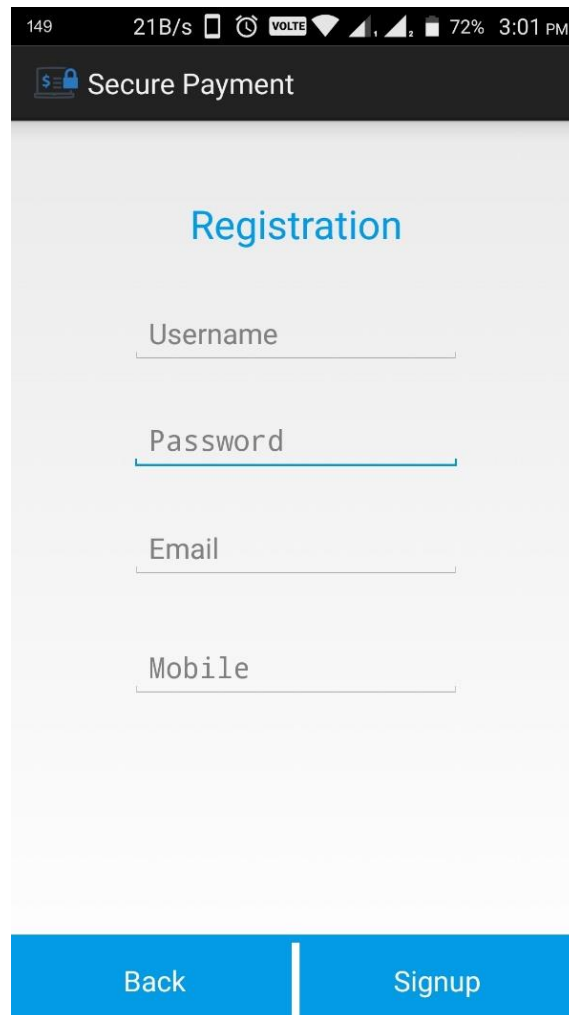
Account Holder Name:	Sundar
Mobile Number:	4578963214
E-mail:	sundar@gmail.com
Address:	No.5, Gandhi street, Chenn
City:	Sedankudikadu
PinCode:	nil
State:	nil

At the bottom right of the form is a "Submit" button.

5.2.1.2 Bank Account Creation

5.2.1.3 Signing Up in application:

Then the uses has to sign up in the application only after the creation of bank account. Separate signup pages are available for customer and merchant. To create an account we have to provide some basic information such as name, address, phone no, email id, username and password. This information are stored in the bank database for authentication of users.

A screenshot of a mobile application's registration page. The status bar at the top shows '149', '21B/s', 'VOLTE', and '72% 3:01 PM'. Below the status bar is a dark header with a lock icon and the text 'Secure Payment'. The main content area is light gray and features the title 'Registration' in blue. There are four input fields: 'Username', 'Password', 'Email', and 'Mobile', each with a blue underline. At the bottom, there is a blue bar with two buttons: 'Back' and 'Signup', separated by a vertical white line.

5.2.1.3 Registration Page

5.2.2 Customer Payment Request

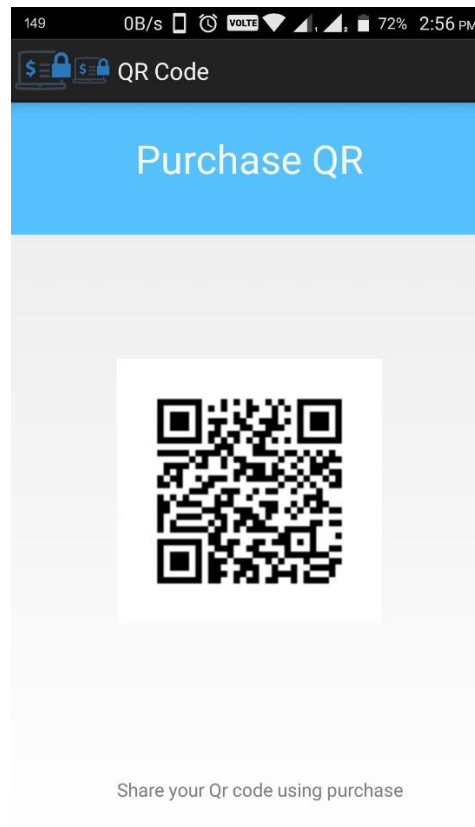
In this module the customer will request the bank server for the amount which he has to pay the merchant. After the customer has done shopping he has to login into the application with the information such as login id and password. Login will be authenticated by the bank server. After logged in the customer will be shown the welcome page. We can toggle to other pages by clicking the icon on the top left corner. We can move to seven different pages and they are Profile page, Wallet page, Deposit page, Purchase page, QR code, Disaster page and About us page. The profile page will show the Customer details such as name,

mail id and phone number. Wallet page shows the money which he has in his account. Disaster page will be discussed later. About us page will give information about the application and this acts as the welcome page. Purchase page is used to make purchase request to bank. QR code page will hold the received QR code from the bank.

To make payment request the customer should select the purchase button. Then the customer should scan the merchant QR code using the phone camera which contains merchant information such as merchant name, id and mobile number. We use the bar code reader to scan the QR code. After getting the merchant information, customer should fill the payment details (product name and its price). Then the customer should click the send money button to transfer the information to the bank. The data are encrypted before transmission to provide security. We use DES algorithm for encryption and decryption which keys cannot be easily hacked. Then the encrypted data is encoded with Base64 algorithm. Base64 encoding is used to convert binary data into a text-like format that allows it to be transported in environments that can handle only text safely.

Finally the Base64 encrypted data is send through WANET. Here WANET consist of mobile phones and transceiver. All mobiles using this application will be connected to the transceiver. The data is send from customer mobile to the nearest transceiver and from the transceiver to other connected mobiles and from that mobiles to other transceivers. This process will be repeated until the data reaches the mobile which has the internet facility. Then the data will reach the bank through internet banking facility. This data will be decrypted and will be verified and validated by the bank server. If the information are correct and the customer has sufficient balance in his account then the server creates a QR code. This QR has a digital signature which avoids the users from double spending it. This QR is send back to the customer using the same way (WANET). If balance is insufficient in the customer account then the bank checks for the endorsers

account to withdraw money. If there is unavailability of balance in any account it send the message to the customer. This received QR code can be view in the customer QR code page. If the merchant scans the QR code money will be credited to his account.



5.2.2 Bank Generated QR Code

5.2.3 Merchant process

In this module the merchant request to transfer the amount from the customer to his account. Merchant can do this process by scanning the received QR code from the customer. First the merchant should login into his application in the same manner as the customer did. Then he will have the same welcome page as the customer. Here he has the same pages like the customer such as My Wallet page, Disaster page, About us and Merchant profile. This pages purpose is described in the above module. Other than these pages we have My QR page,

Mini statement page and Scan QR. My QR page consist of QR code which holds the Merchant information which is already described in previous module. Mini statement has the history of the transaction that has been made till present.

Scan QR is used to scan the customer received QR code and proceed to transaction phase. When the QR code is scanned the merchant gets the information for the transaction. Then he has to enter the product name as password and click send money button. This data is encrypted and travels using WANET and reaches bank. Bank server verify the digital signature and if correct it transfers the amount to the merchant. After the transaction the QR code becomes invalid and acknowledgement message is send to the merchant about the successful transaction. If QR code is invalid then it send message to merchant about the failure transaction. During the communication process between the bank and the user the data is encoded and decoded using DES and Base64 algorithm.

5.2.4 Region Situation Update

In this final module we will be providing a page where the user can upload their surrounding situation. During disaster time the area we live may be ruined. This information should be shared to others for safety. This helps the travelers who travel by road. Since all the user are connected to WANET, messages can be easily transferred to all users. We created a page where a user can view and update the surrounding destructive or disturbing information. The information will be broadcasted and will reach all users who are connected to WANET. The Situation update will be broadcasted using three fields-subject, place and status. The message can be heavy rain or flood or bridge collapse. This can also help to broadcast emergency message and call for rescue which will be useful for the victims in disaster area.

CHAPTER 6

IMPLEMENTAION

6.1 Sample Code:

Securepayment.java

```
package securepayment;

import java.io.IOException;
import javafx.application.Application;
import static javafx.application.Application.launch;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class SecurePayment extends Application {

    @Override
    public void start(Stage primaryStage) throws IOException {
        Parent
        root=FXMLLoader.load(getClass().getResource("indexpage.fxml"));
        Scene scene=new Scene(root);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Cloud");
        primaryStage.show();
    }
}
```

```

        public static void main(String[] args) {
            launch(args);
        }
    }
}

```

Transfercall.java

```

package securepayment;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.URL;
import java.net.URLEncoder;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.HashMap;
import java.util.TreeMap;
import javax.swing.JOptionPane;

public class TransferCall extends Thread
{
    HashMap< String, String> nodeport = new HashMap<String,
String>();
    HashMap< String, String> nodeadd = new HashMap<String, String>();

```

```

    HashMap< String, String> nodemem = new HashMap<String,
String>();
    HashMap< String, String> nodedis = new HashMap<String, String>();
    HashMap< String, String> nodename = new HashMap<String,
String>();
    TreeMap< String, String> nodeusername = new TreeMap<String,
String>();
    TreeMap< String, String> nodesystem = new TreeMap<String,
String>();
    TreeMap< String, String> noderan = new TreeMap<String, String>();
    TreeMap< String, String> noderating = new TreeMap<String,
String>();
    TreeMap< String, String> activetables = new TreeMap<String,
String>();
    String node, port, sys, range, dist, publickey, node1, certify, address,
DistRange, broadcast, discovery;
    NodecreationController cs;
    String cptxt,pass,times;
    public TransferCall(NodecreationController cs, String port, String
nodename) {
        start();
        this.cs = cs;
        this.port = port;
    }

    @Override
    public void run() {
        try {
            ServerSocket ss = new ServerSocket(Integer.parseInt(port));

```

```

while (true) {
    System.out.println("connection open" + port);
    Socket s = ss.accept(); //establishes connection
    System.out.println("connection established");
    ObjectInputStream dis = new
ObjectInputStream(s.getInputStream());
    String temp = dis.readObject().toString();
    if (temp.equalsIgnoreCase("Userverification")) {
        String unames=dis.readObject().toString().trim();
        String mname = dis.readObject().toString().trim();
        String mid = dis.readObject().toString().trim();
        String pname = dis.readObject().toString().trim();
        String amont = dis.readObject().toString().trim();
        String mmob = dis.readObject().toString().trim();
        String internode = dis.readObject().toString().trim();
        String interport = dis.readObject().toString().trim();
        String usernode = dis.readObject().toString().trim();
        String userport = dis.readObject().toString().trim();
        String userip = dis.readObject().toString().trim();
        String useremail = dis.readObject().toString().trim();
        String usermob = dis.readObject().toString().trim();

        Socket scs = ss.accept(); //establishes connection

        String cps = dis.readObject().toString();
        if (cps.equalsIgnoreCase("Cptext"))
        {
            cptxt = dis.readObject().toString();
            pass=dis.readObject().toString().trim();

```

```

    }

    JOptionPane.showMessageDialog(null, "Data Reached Bank
and Transaction is Verified");

    Decrypt d=new Decrypt();
    String data= d.decrypt(cptxt, pass);
    String st[]=data.split("@");
    {
        String name=st[0];
        String ema=st[1];
        String mob=st[2];
        String amount=st[3];

        URL u = new
        URL("http://localhost:9999/BankAdmin/UserRequestFromApp?name="+
name+"&email="+ema+"&mobile="+mob+"&amount="+amount);

        BufferedReader in = new BufferedReader(
            new InputStreamReader(u.openStream()));
        String str = in.readLine();
        if(name.equalsIgnoreCase("success"))
        {

            try
            {
                DateFormat dateFormat = new
SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
                Calendar cal = Calendar.getInstance();
                String currentdate=dateFormat.format(cal.getTime());

```

```

        System.out.println(currentdate);
        Socket sss = new
Socket(userip,Integer.parseInt(userport));
        ObjectOutputStream oos= new
ObjectOutputStream(sss.getOutputStream());
        oos.writeObject("Returnprocess");
        oos.writeObject("success");
        oos.writeObject(pname);
        oos.writeObject(amont);
        oos.writeObject(mname);
        oos.writeObject(currentdate);
        oos.close();
        s.close();
    }
    catch(Exception e)
    {

    }
}

else if (temp.equalsIgnoreCase("Merchantverification")) {
    String unames=dis.readObject().toString().trim();
    String mname = dis.readObject().toString().trim();
    String mid = dis.readObject().toString().trim();
    String pname = dis.readObject().toString().trim();
    String amont = dis.readObject().toString().trim();
    String mmob = dis.readObject().toString().trim();
    String internode = dis.readObject().toString().trim();

```

```

        String interport = dis.readObject().toString().trim();
        String usernode = dis.readObject().toString().trim();
        String userport = dis.readObject().toString().trim();
        String userip = dis.readObject().toString().trim();
        String useremail = dis.readObject().toString().trim();
        String usermob = dis.readObject().toString().trim();

        Socket scs = ss.accept(); //establishes connection

        String cps = dis.readObject().toString();
        if (cps.equalsIgnoreCase("MerchantCheck"))
        {
            cptxt = dis.readObject().toString();
            pass=dis.readObject().toString().trim();
            times=dis.readObject().toString().trim();
        }
        JOptionPane.showMessageDialog(null, "Data Reached Bank
and Transaction verified");

        Decrypt d=new Decrypt();
        String data= d.decrypt(cptxt, pass);
        String st[]=data.split("@");
        {
            String name=st[0];
            String ema=st[1];
            String mob=st[2];
            String amount=st[3];

        String anothertemp=URLEncoder.encode(data);

```

```

        URL u = new
URL("http://localhost:9999/BankAdmin/MerchantRequest?acc="+name+
"&temp="+anothertemp);
        BufferedReader in = new BufferedReader(
            new InputStreamReader(u.openStream()));
        String str = in.readLine();
        if(name.equalsIgnoreCase("success"))
        {
            try
            {
                DateFormat dateFormat = new
SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
                Calendar cal = Calendar.getInstance();
                String currentdate=dateFormat.format(cal.getTime());
                System.out.println(currentdate);
                Socket sss = new
Socket(userip,Integer.parseInt(userport));
                ObjectOutputStream oos= new
ObjectOutputStream(sss.getOutputStream());
                oos.writeObject("Returnprocess");
                oos.writeObject("success");
                oos.writeObject(pname);
                oos.writeObject(amont);
                oos.writeObject(mname);
                oos.writeObject(currentdate);
                oos.close();
                s.close();
            }
            catch(Exception e)

```



```

        {
        }

        }

    }

    if(cps.equalsIgnoreCase("MerchantCheck"))
    {
        cptxt = dis.readObject().toString();
        pass=dis.readObject().toString().trim();
        times=dis.readObject().toString().trim();
    }
}

} catch (Exception e)
{
}

}

}

```

6.2 Sample Screen Shot:

6.2.1 Navigation Page

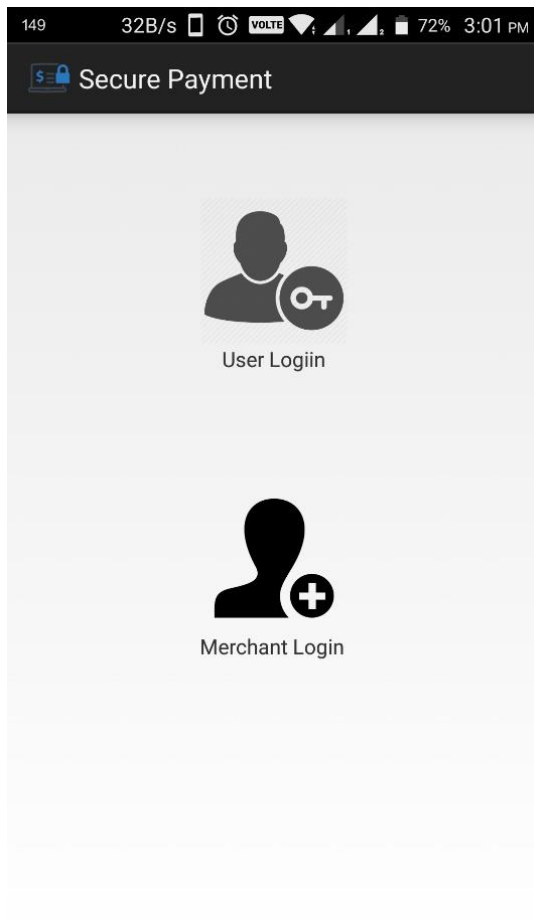


Fig 6.2.1 Navigation Page

6.2.2 Customer & Merchant Registration page

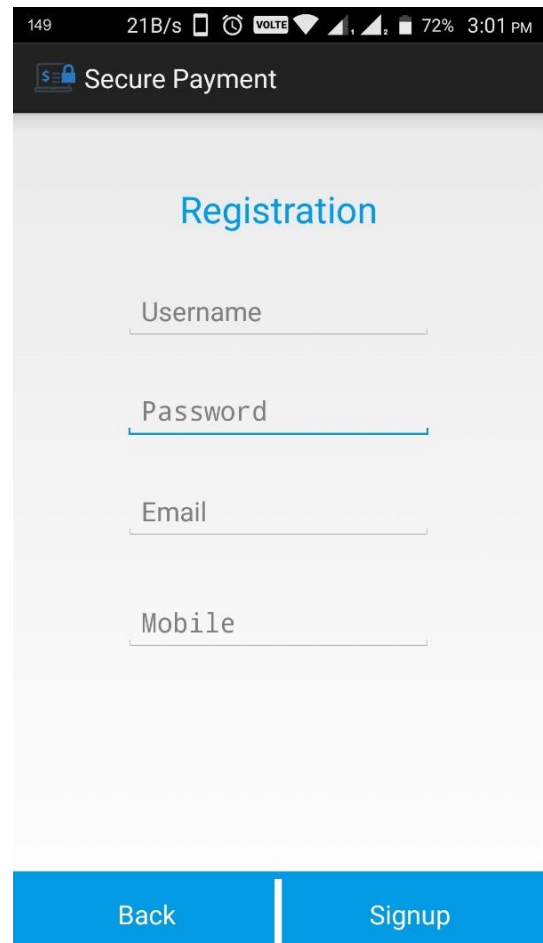
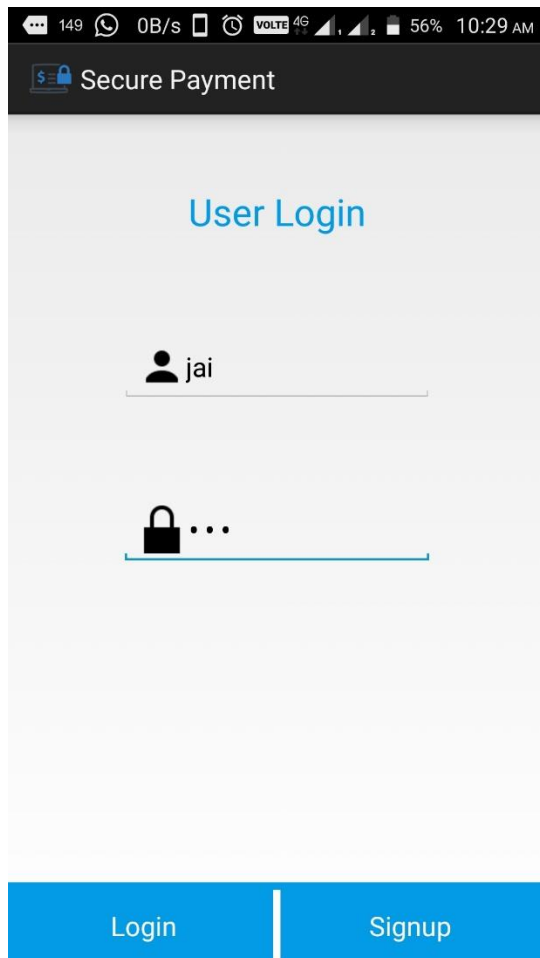


Fig 6.2.2 Registration Page

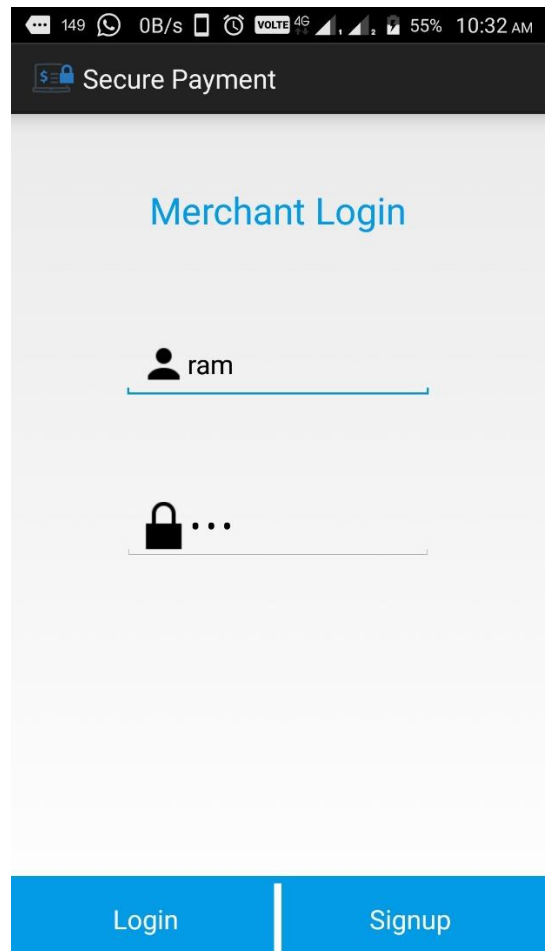
6.2.3 Customer Login Page



The screenshot shows a mobile application interface for a customer login. At the top, a dark status bar displays the time as 10:29 AM and battery level at 56%. Below this, a dark header bar contains a lock icon and the text "Secure Payment". The main content area has a light gray background with the title "User Login" in blue. There are two input fields: the first contains the username "jai" with a person icon, and the second contains a masked password "..." with a lock icon. At the bottom, a blue bar features two buttons: "Login" and "Signup", separated by a vertical white line.

Fig 6.2.3 Customer Login Page

6.2.4 Merchant Login Page



The screenshot shows a mobile application interface for a merchant login. At the top, a dark status bar displays the time as 10:32 AM and battery level at 55%. Below this, a dark header bar contains a lock icon and the text "Secure Payment". The main content area has a light gray background with the title "Merchant Login" in blue. There are two input fields: the first contains the username "ram" with a person icon, and the second contains a masked password "..." with a lock icon. At the bottom, a blue bar features two buttons: "Login" and "Signup", separated by a vertical white line.

Fig 6.2.4 Merchant Login Page

6.2.5 Customer & Merchant Welcome Page

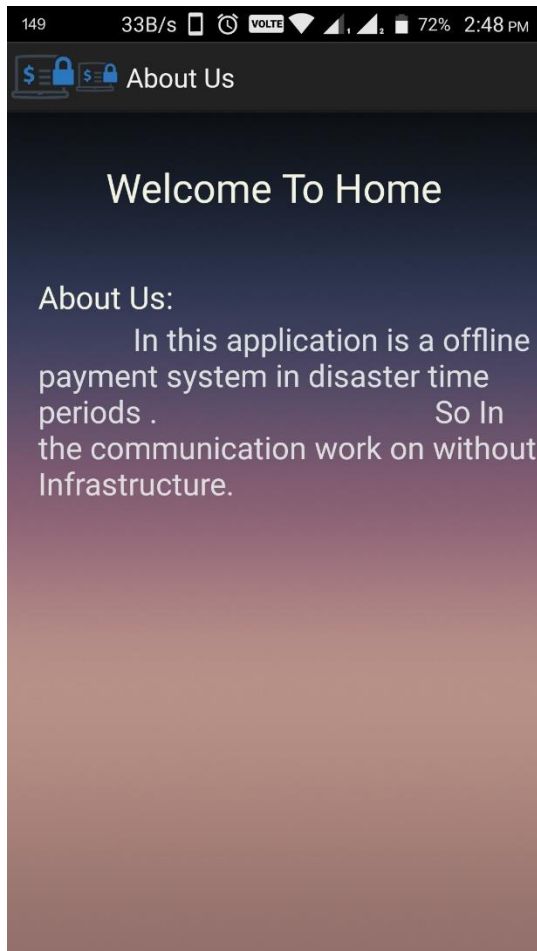


Fig 6.2.5 Welcome Page

6.2.6 Customer Navigation Page

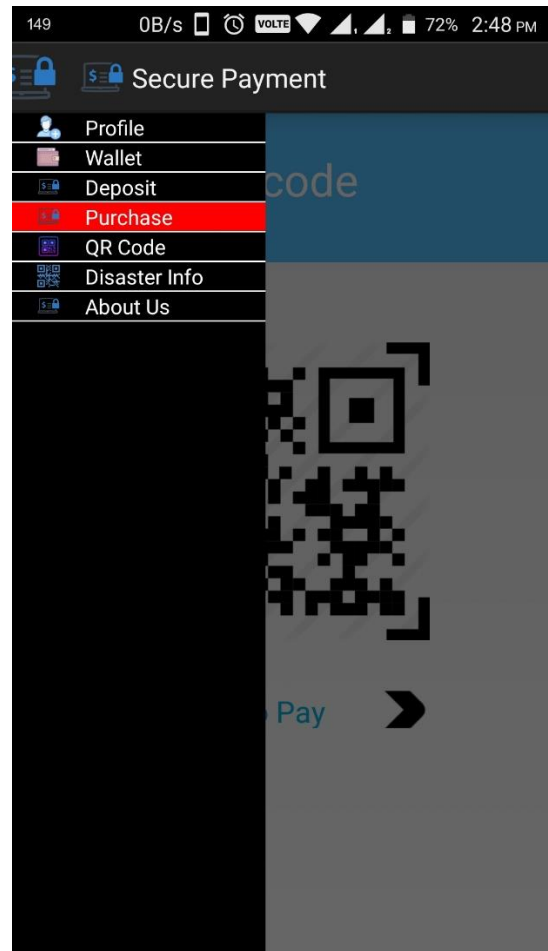


Fig 6.2.6 Customer Navigation Page

6.2.7 Merchant Navigation Page

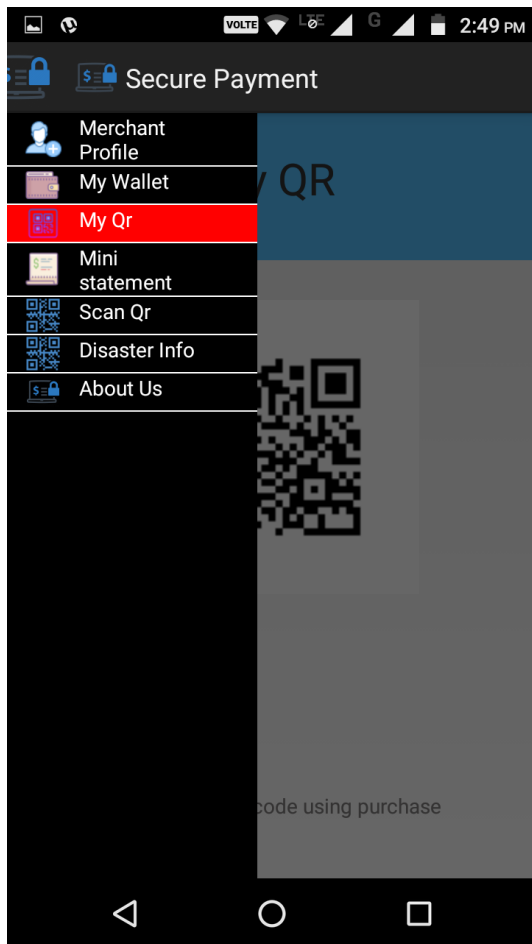


Fig 6.2.7 Merchant Navigation Page

6.2.8 Customer Profile Page

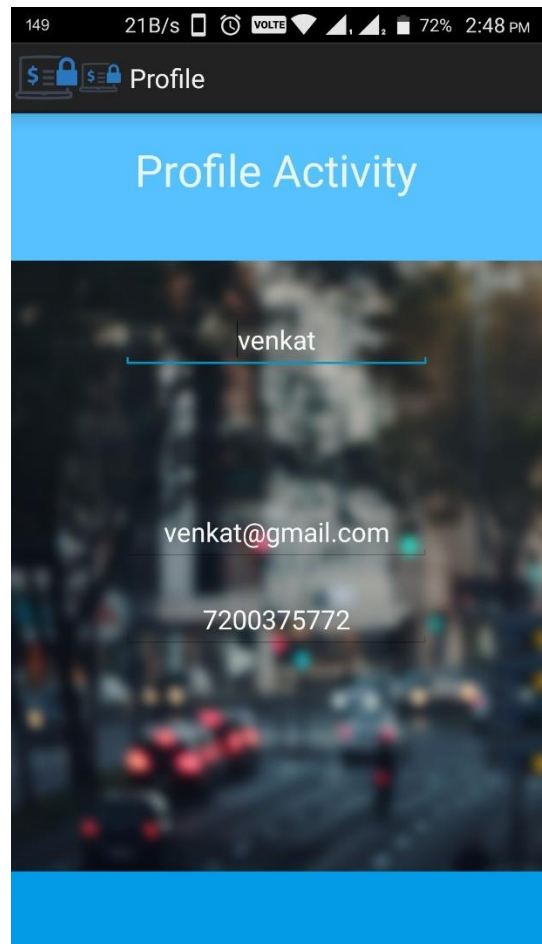


Fig 6.2.8 Customer Profile Page

6.2.9 Merchant Profile Page

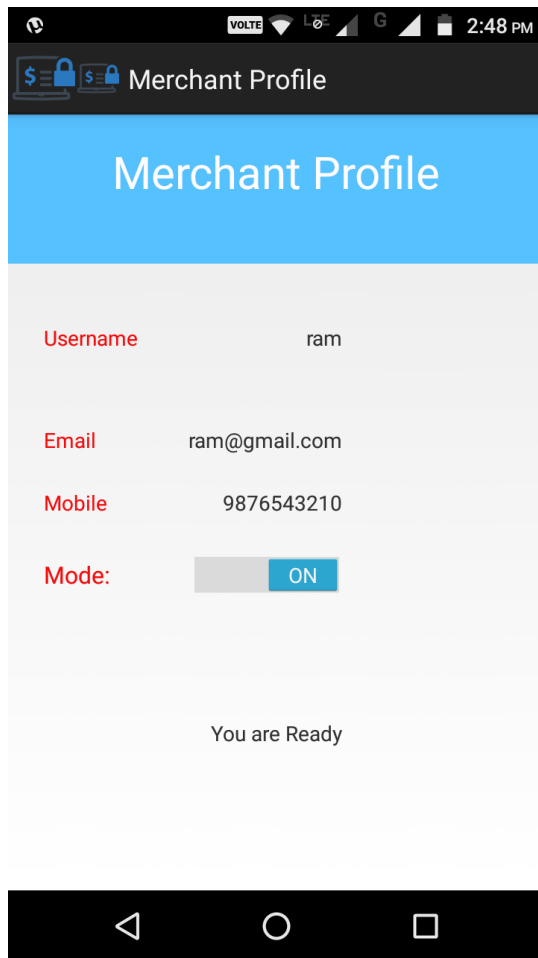


Fig 6.2.9 Merchant Profile Page

6.2.10 Merchant QR Code

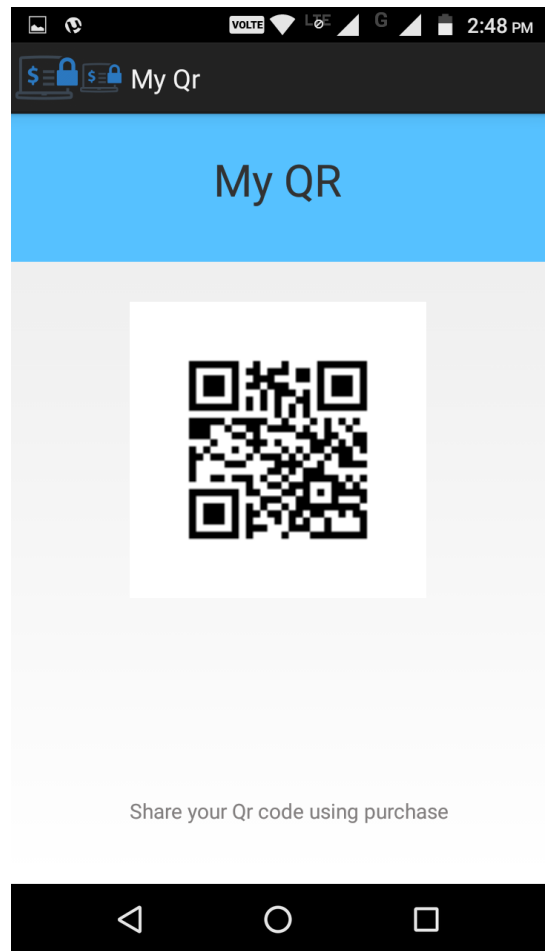


Fig 6.2.10 Merchant QR Code

6.2.11 Customer Payment Page

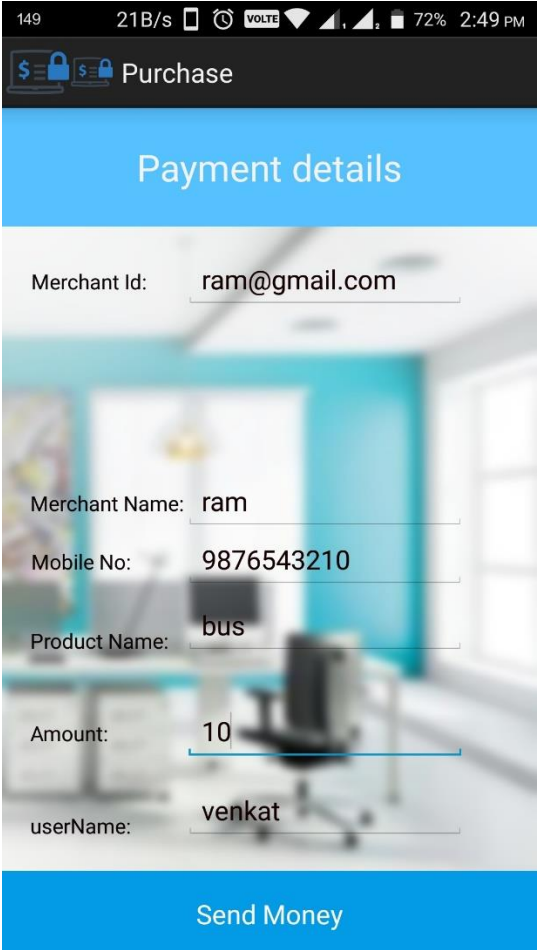


Fig 6.2.11 Customer Payment Page

6.2.12 Data Traveling Through Other Mobile

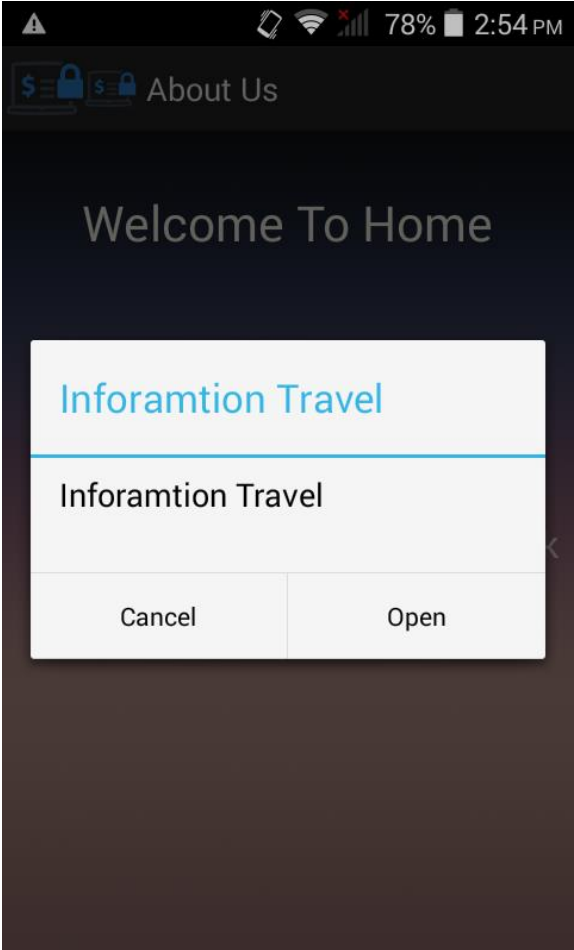


Fig 6.2.9 Data Transfer

6.2.13 Data Traveling Through Virtual Mobile

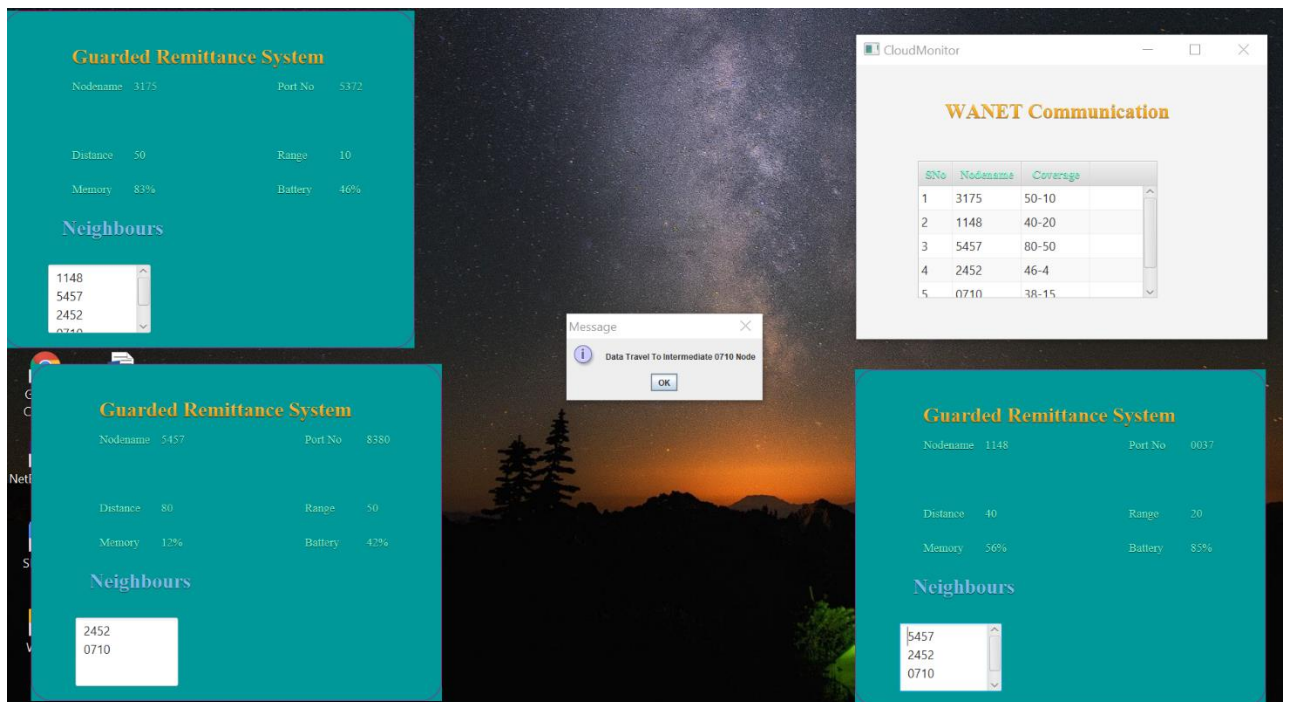


Fig 6.2.13 Data Traveling Through Virtual Mobiles

6.2.14 Data Reached Bank and Verified

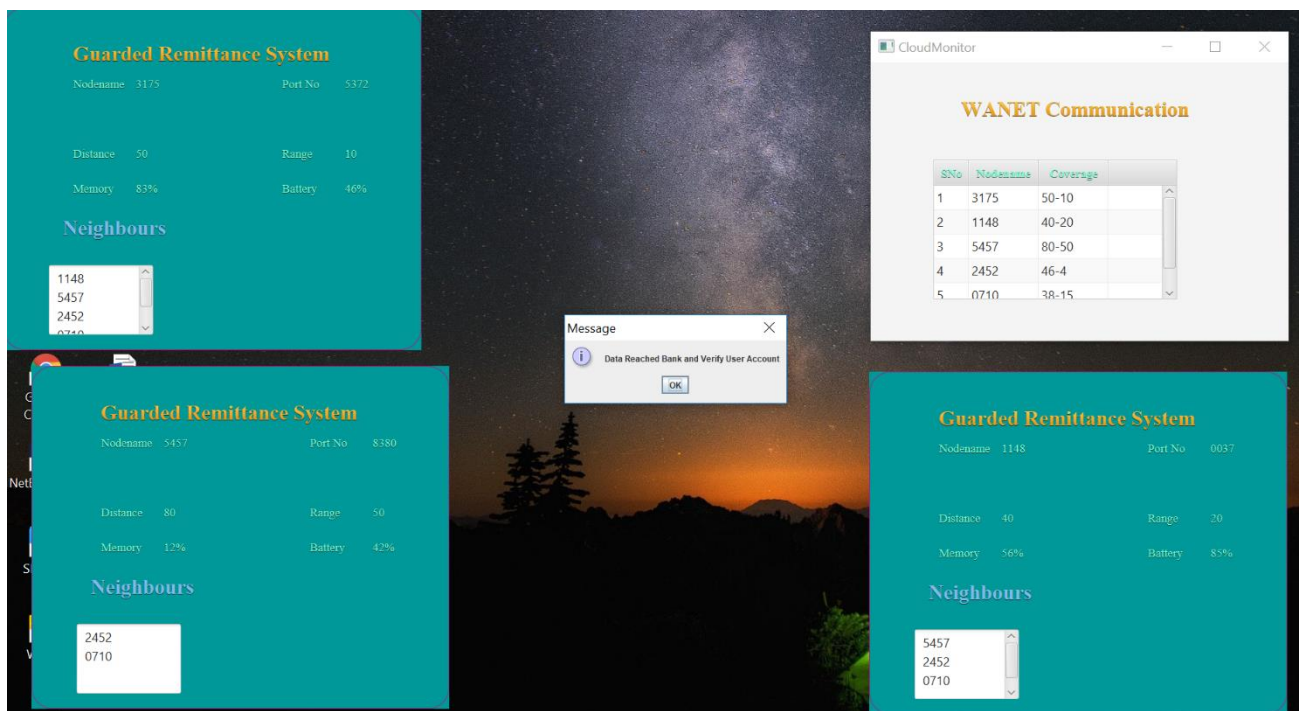


Fig 6.2.14 Data Reached Bank and Verified

6.2.15 Customer Received QR Code from Bank

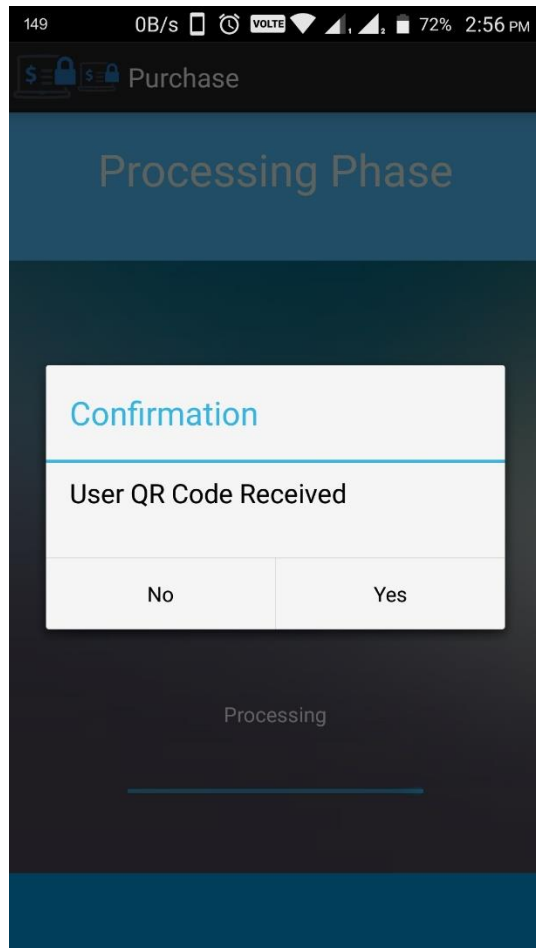


Fig 6.2.15 Received QR code

6.2.16 Customer Purchase QR Code

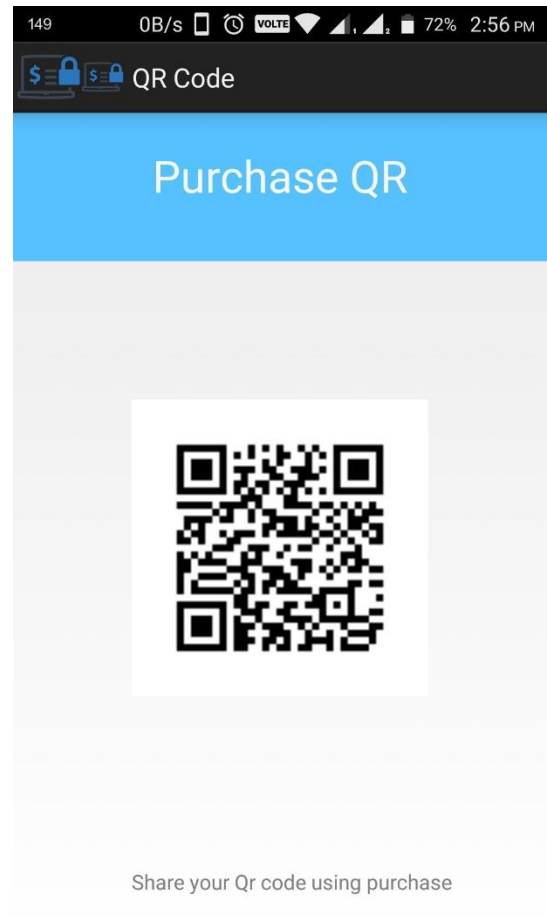
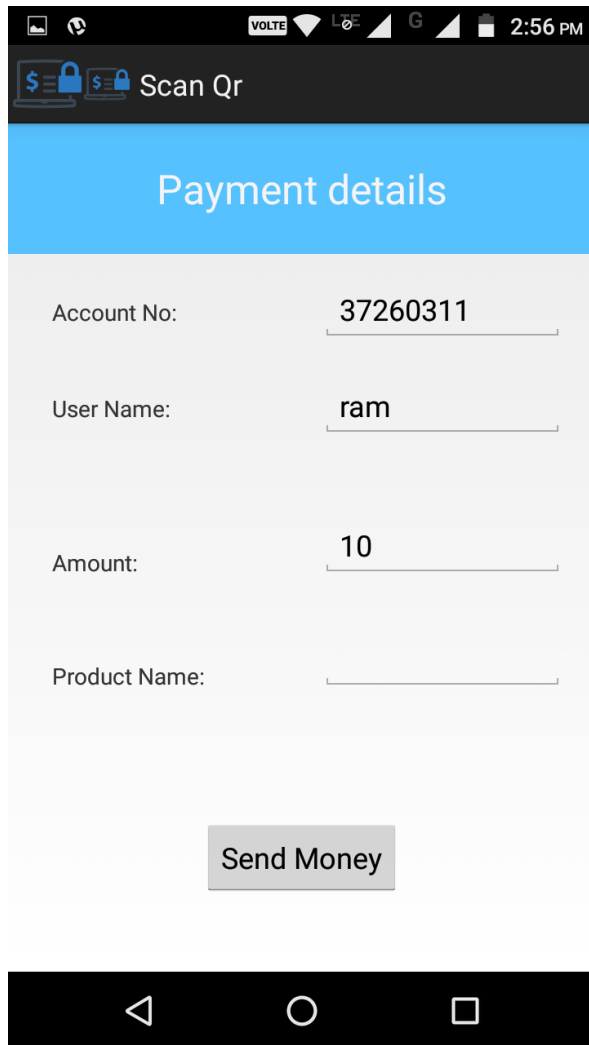


Fig 6.2.16 Customer Purchase QR code

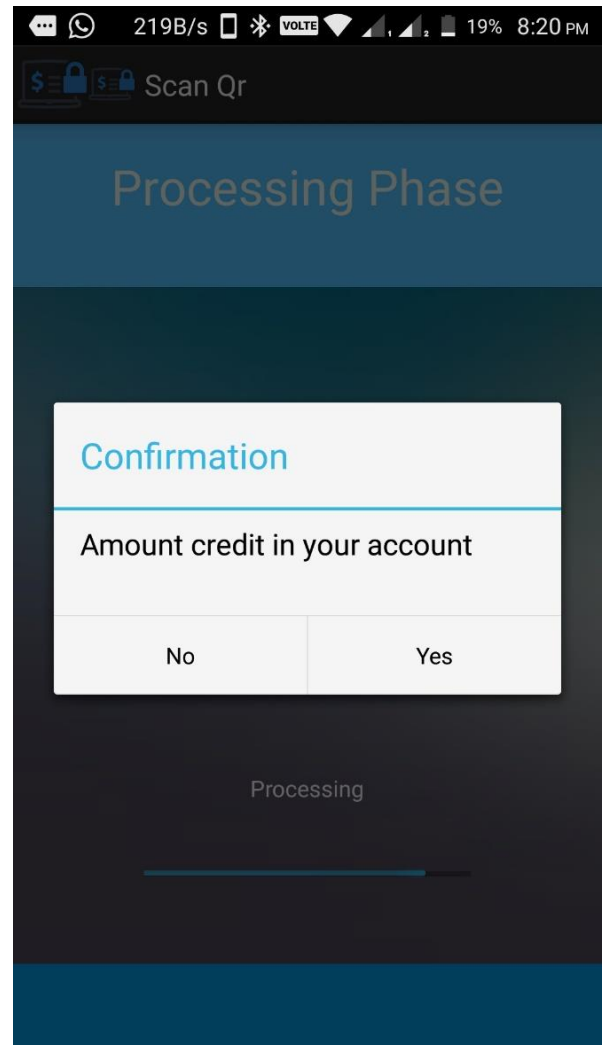
6.2.17 Merchant Scans the QR Code



The screenshot shows a mobile application interface titled "Scan Qr". Below the title bar is a blue header with the text "Payment details". The main area contains four input fields: "Account No:" with the value "37260311", "User Name:" with the value "ram", "Amount:" with the value "10", and "Product Name:" which is empty. At the bottom of the form is a grey button labeled "Send Money". The status bar at the top shows the time as 2:56 PM and various connectivity icons.

Fig 6.2.17 Merchant Scans the QR code

6.2.18 Acknowledgment Message to Merchant



The screenshot shows the same "Scan Qr" application interface, but now a "Confirmation" dialog box is displayed in the center. The dialog has a title "Confirmation" and a message "Amount credit in your account". Below the message are two buttons: "No" and "Yes". Above the dialog, the text "Processing Phase" is visible in a large, semi-transparent font. Below the dialog, the word "Processing" is displayed above a progress bar. The status bar at the top shows the time as 8:20 PM and a battery level of 19%.

Fig 6.2.18 Acknowledge Message to Merchant

6.2.19 Transaction History

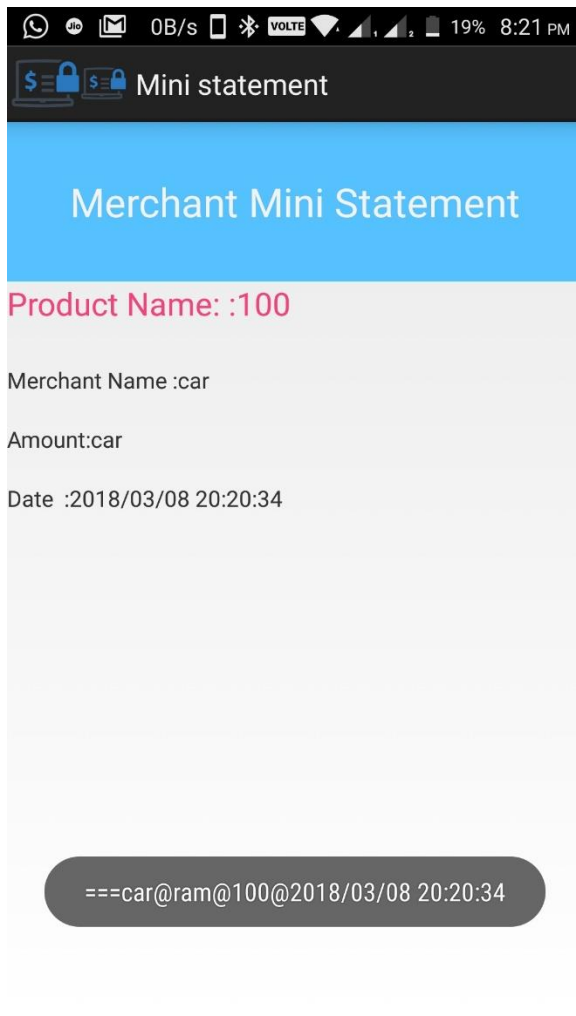


Fig 6.2.19 Merchant Transaction History

6.2.20 Wallet Page

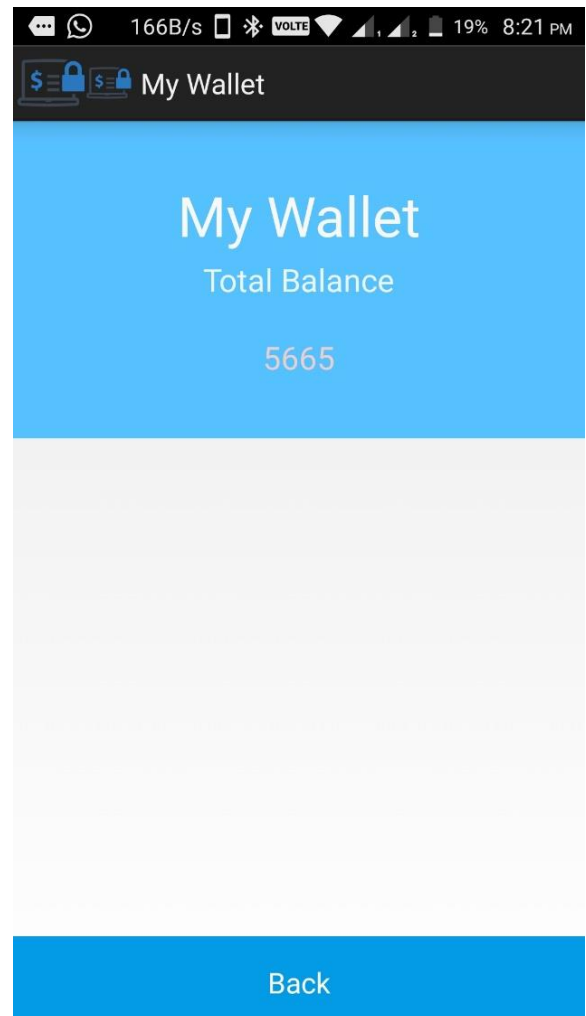


Fig 6.2.20 Wallet Page

6.2.21 Disaster Information

The screenshot shows a mobile application interface for disaster information. At the top, there's a status bar with network speed (166B/s), signal strength, and battery level (19%). Below the status bar is a header with a lock icon and the text 'Disaster Info'. The main content area has a blue header with the text 'Disaster Status'. Below this, there's a list of disaster entries, each with three lines of text: 'Subject :', 'Place :', and 'Status :'. The entries are: 1. Subject : rain fall, Place : saidapet, Status : heavy rain; 2. Subject : sub, Place : tambaram, Status : heavy rain; 3. Subject : rain, Place : saidapet, Status : heavy water; 4. Subject : aaa, Place : bbb, Status : ccc. At the bottom, there's a blue button with the text 'Update Information'.

Subject :	rain fall
Place :	saidapet
Status :	heavy rain
Subject :	sub
Place :	tambaram
Status :	heavy rain
Subject :	rain
Place :	saidapet
Status :	heavy water
Subject :	aaa
Place :	bbb
Status :	ccc

Update Information

Fig 6.2.21 Disaster Information Page

6.2.22 Disaster Update Page

The screenshot shows a mobile application interface for disaster update. At the top, there's a status bar with network speed (0B/s), signal strength, and battery level (72%). Below the status bar is a header with a lock icon and the text 'Disaster Info'. The main content area has a blue header with the text 'Upadet status'. Below this, there's a form with three input fields: 'Earthquake', 'chennai', and 'washed out'. At the bottom, there's a blue button with the text 'Update'.

Earthquake

chennai

washed out

Update

Fig 6.2.22 Disaster Update Page

6.3 Sample Database Table

6.3.1 Disaster Information Table

ID	Subject	Place	Status
1.	Earthquake	Chennai	No power supply
2.	Tsunami	Kanyakumari	Road Block
3.	Heavy Rain	T Nagar	Flood

Table 3.6.1 Disaster Information

6.3.2 Merchant Details Table

ID	User Name	Password	E-mail Address	Mobile No.	Public Key	Private Key
1.	Yokesh	1235	yo@gmail.com	9874563210	77-481	101-481
2.	Venu	Hai	vu@gmail.com	1234567890	13-493	69-493
3.	Nishath	4565	ni@gmail.com	5478963210	68-452	35-426

Table 6.3.2 Merchant Detail

6.3.3 Customer Detail Table

User ID	User Name	Password	E-Mail Address	Mobile No.	Public Key	Private Key
1.	Ramesh	3594	ram@gmail.com	9876543210	817-7031	5713-7031
2.	Hari	7894	hari@gmail.com	8745693210	95-469	371-469
3.	Venkat	8756	venkat@gmail.com	7200375772	131-2231	1451-2231

Table 6.3.3 Customer Detail

CHAPTER 7

TESTING AND MAINTENANCE

7.1 Testing

Software Testing is evaluation of the software against requirements gathered from users and system specifications. Testing is conducted at the phase level in software development life cycle or at module level in program code. Software testing comprises of Validation and Verification.

i) Software Validation

Validation is process of examining whether or not the software satisfies the user requirements. It is carried out at the end of the SDLC. If the software matches requirements for which it was made, it is validated.

- Validation answers the question – "Are we developing the product which attempts all that user needs from this software?"
- Validation emphasizes on user requirements.

ii) Software Verification

Verification is the process of confirming if the software is meeting the business requirements, and is developed adhering to the proper specifications and methodologies.

- Verification ensures the product being developed is according to design specifications.
- Verification answers the question– "Are we developing this product by firmly following all design specifications?"
- Verifications concentrates on the design and system specifications.

7.1.1 Need for Testing

- **Errors** - These are actual coding mistakes made by developers. In addition, there is a difference in output of software and desired output, is considered as an error.
- **Fault** - When error exists fault occurs. A fault, also known as a bug, is a result of an error which can cause system to fail.
- **Failure** - failure is said to be the inability of the system to perform the desired task. Failure occurs when fault exists in the system.

7.1.2 Different Ways of Testing

Testing can either be done manually or using an automated testing tool:

- **Manual Testing:**

i) This testing is performed without taking help of automated testing tools. The software tester prepares test cases for different sections and levels of the code, executes the tests and reports the result to the manager.

ii) Manual testing is time and resource consuming. The tester needs to confirm whether or not right test cases are used. Major portion of testing involves manual testing.

- **Automated Testing:**

i) This testing is a testing procedure done with aid of automated testing tools. The limitations with manual testing can be overcome using automated test tools.

7.1.3 Testing Levels

Software testing is a process of executing a program or application with the intent of finding the software errors. Testing is one of the phase in software development cycle. It is performed to make sure that there are no hidden bugs or issues left in the software. Software is tested on various levels -

1. Unit Testing

Unit testing is a level of software testing where individual units or components of a software are tested. The software developers in our project perform this testing to make sure that individual unit in the project is error free. Unit testing is performed under white-box testing approach. Unit testing helps our project developers to decide whether the individual units of the program are working as per requirement. Login page is one of the individual unit in android application. This page is tested manually and all test cases passed successfully.

2. Integration Testing

After completion of unit testing in our project, the units or modules are integrated and testing is performed which gives raise to integration testing. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated. The modules such as bank account creation and customer payment request are integrated and testing is performed. During the integration of modules, we found an error in signing into the application. After evaluation of codes, we identified a bug which does not allow us to login. Finally the bug was removed.

3. Acceptance Testing

When the software is ready to hand over to the customer, it has to go through last phase of testing where it is tested for user-interaction and response. This is important because even if the software matches all user requirements and

if user does not like the way it appears or works, it may be rejected. This type of testing can be done in two ways. In our project both alpha and beta testing is done.

- **Alpha testing** - The project development team themselves performed the alpha testing by using the system as if it is being used in work environment. They try to find out how user would react to some action in software and how the system should respond to inputs. And they deducted an error where the regional situation update page does not update any data. This error was corrected by the team itself.
- **Beta testing** - After the software is tested internally, it is handed over to the users to use it under their production environment only for testing purpose. In our project the team members acts as the beta testers and bta testing is made.

4. Regression Testing

Whenever a change in a software application is made, it is possible that other areas within the application have been affected by this change. Regression testing is performed to verify that a fixed bug hasn't resulted in another functionality or business rule violation. The intent of regression testing is to ensure that a change, such as a bug fix should not result in another fault being uncovered in the application. We used regression testing by introducing Multilevel Endorsement Mechanism in our application for guarnted payment. If we implement this feature, it should not affect the existing user transcation. So regression tests the typical transcation and multilevel endorser transcation to check the desired transcation.

7.2 Test Cases

Test Case ID	Test Item	Input Specification		Output Specification	Pass/Fail
		Action	Case Description		
TC_01	Customer login page	Onsuccess	On the success event of login	It redirects to customer home page.	Pass
TC_02	Customer login page	OnFailure	When customer login is failure, it must display failure message	Server shows invalid username or password	Pass
TC_03	Merchant login page	Onsuccess	On the success event of login	It redirects to merchant home page.	Pass
TC_04	Merchant login page	OnFailure	When merchant login is failure, it must display failure message	Server shows invalid username or password	Pass
TC_05	Signup Page	OnSuccess	The new user can register to the application	Server toasts Successfully registered message.	Pass
TC_06	Signup page	OnFailure	The new user can't register to the application	Server toasts an error message	Pass
TC_07	QR code Scan	OnSuccess	When the QR code verified and it is valid	It shows the Payment details screen	Pass
TC_08	QR code Scan	Onfailure	When the QR code verified and it is invalid	Ask us to rescan	Pass

TC_09	Customer request for Money	Onsuccess	Customer details will be verified and QR will be generated	Customer will receive QR code from bank	Pass
TC_10	Customer request for Money	OnFailure	The Customer detail may not reaches bank or no intermediates nodes available	Customer cannot request money	Pass
TC_11	Merchant request for Money	OnSuccess	Money is credited to merchant's account	Displays amount credited in your account	Pass
TC_12	Merchant request for Money	OnFailure	The QR code may be already used and invalid	Bank shows invalid QR code message	Pass
TC_13	Disaster Status	On update	The information is updated by the user	Information can be viewed by all app users	Pass

7.3 Maintenance

Software maintenance is the modification of a software product after delivery to correct faults, to improve performance or other attributes. A common perception of maintenance is that it merely involves fixing defects.

The maintenance activities into four classes:

- **Adaptive** – modifying the system to cope with changes in the software environment (DBMS, OS)
- **Perfective** – implementing new or changed user requirements which concern functional enhancements to the software
- **Corrective** – diagnosing and fixing errors, possibly ones found by users

- **Preventive** – increasing software maintainability or reliability to prevent problems in the future

Software Maintenance Processes

This section describes the six software maintenance processes as:

- The implementation process contains software preparation and transition activities, such as the conception and creation of the maintenance plan; the preparation for handling problems identified during development; and the follow-up on product configuration management.
- The problem and modification analysis process, which is executed once the application has become the responsibility of the maintenance group. The maintenance programmer must analyze each request, confirm it (by reproducing the situation) and check its validity, investigate it and propose a solution, document the request and the solution proposal, and finally, obtain all the required authorizations to apply the modifications.
- The process considering the implementation of the modification itself.
- The process acceptance of the modification, by confirming the modified work with the individual who submitted the request in order to make sure the modification provided a solution.
- The migration process (platform migration, for example) is exceptional, and is not part of daily maintenance tasks. If the software must be ported to another platform without any change in functionality, this process will be used and a maintenance project team is likely to be assigned to this task.
- Finally, the last maintenance process, also an event which does not occur on a daily basis, is the retirement of a piece of software.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

Currently available mobile payment system demands fixed infrastructure for communication and transaction. But we designed an android application system which does not require any fixed infrastructure for communication. This allows users to make mobile payment in offline mode utilizing WANET. Instead of searching for network we create WANET using wireless devices such as transceivers and user mobiles and establish communication between bank and users. This will be a great remedy for the people to purchase in disaster area. And to provide guaranteed payment for merchant we use Multilevel Endorser Mechanism. Also we use cipher technique to perform a secured transaction and to make payment process easier we use QR code.

8.2 Future Enhancement

In future, we improve the QoS (Quality of Service) of the network and network architectures of WANET. The proposed system connects the mobile via transceiver to pass messages and data. The future work involves the transfer of messages and data not only through transceivers but also through mobiles with or without the availability of transceiver. This allows the data to travel through the mobile independently. The transmission will have more reliability in future by introducing complex hashing algorithms such as MD5, SHA1, SHA256, etc.

CHAPTER 9

REFERENCES

- [1] B. Ojetunde, N. Shibata, J. Gao, and M. Ito, “An endorsement-based mobile payment system for a disaster area,” in *Proc. 29th IEEE Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, Gwangju, South Korea, Mar. 2015, pp. 482–489.

- [2] A. Mishra and K. M. Nadkarni, “Security in wireless ad hoc networks,” in *The Handbook of Ad Hoc Wireless Networks*. Boca Raton, FL, USA: CRC Press, 2003, ch. 30, pp. 499–549.

- [3] W. Li, Q. Wen, Q. Su, and Z. Jin, “An efficient and secure mobile payment protocol for restricted connectivity scenarios in vehicular ad hoc network,” *Comput. Commun.*, vol. 35, no. 2, pp. 188–195, Jan. 2012.

- [4] X. Dai, O. Ayoade, and J. Grundy, “Off-line micro-payment protocol for multiple vendors in mobile commerce,” in *Proc. 7th Int. Conf. Parallel Distrib. Comput. Appl. Technol. (PDCAT)*, Taipei, Taiwan, 2006, pp. 197–202.

- [5] V. Patil and R. K. Shyamasundar, “An efficient, secure and delegable micro-payment system,” in *Proc. IEEE Int. Conf. e-Technol. e-Commerce e-Service (EEE)*, Taipei, Taiwan, Mar. 2004, pp. 394–404.

- [6] Y.-Y. Chen, J.-K. Jan, and C.-L. Chen, “A novel proxy deposit protocol for e-cash systems,” *Appl. Math. Comput.*, vol. 163, no. 2, pp. 869–877, 2005.
- [7] Y. Meier, J. Xu, O. Atan, and M. van der Schaar, “Personalized grade prediction: A data mining approach,” in *Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE, 2015, pp. 907–912.
- [8] C. G. Brinton and M. Chiang, “Mooc performance prediction via clickstream data and social learning networks,” in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 2299–2307.
- [9] KDD Cup, “Educational data mining challenge,” <https://pslcdatashop:web:cmu.edu/KDDCup/>, 2010.
- [10] Y. Jiang, R. S. Baker, L. Paquette, M. San Pedro, and N. T. Heffernan, “Learning, moment-by-moment and over the long term,” in *International Conference on Artificial Intelligence in Education*. Springer, 2015, pp. 654–657.

CHAPTER 10
APPENDIX
(PUBLICATION DETAILS)

Paper Title	Guarded Remittance System Employing WANET for Catastrophe Region
Authors	Mr.Balasundaraganapathi.N, Yokeswaran.T, Venugopal.S, Nishanth.M, Manoj.S
Journal Name	International Research Journal of Engineering and Technology (IRJET)
Edition	Volume: 05 Issue: 03
Month and Year	April 2018