CHAPTER 1

INTRODUCTION

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 the statesmanship 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 (Java virtual machine) 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. The *Java platform* is usually associated with the *Java virtual machine* and the *Java core libraries* .

The Java virtual machine (JVM) is a software implementation of a computer that executes programs like a real machine .The Java virtual machine is written specifically for a specific operating system, e.g., for Linux a special implementation is required as well as for Windows.

1.1 Course Objectives

1. Should be able to write Java application programs with proper program structuring using OOP principles – polymorphism and inheritance.
2. Should be able to implement error handling techniques using exception handling.

3. Should be able to build a swing application or any other advanced GUI for front end and apply JDBC concepts to access databases.
4. Should be able to use various I/O manipulation operations through I/O streams.
5. To be able to work efficiently with the platform independent language JAVA.
6. To be able to define and derive classes by efficiently grouping them under packages and importing when necessary.
7. To be able to implement various OOP concepts like inheritance, abstraction , polymorphism and others as required.
8. Implementing exception handling throughout the classes for better handling of runtime errors.
9. To be able to design an effective and interactive GUI to suit the requirements.
10. To use a backend like database connectivity for storing information and processed data.
11. To be able to build paths to explore more features to make the code efficient and interface effective.

1.2 Problem Statement

The Objective of this Project is to develop a portal for E-wallets and Payment, which allows the user to enter the login details. It is also used to manage details of Username, Password, 4 digit MTPN, name, age, address and phone number to sign up. Only users can be able to access it in order to enter the signing up details of the Payment application. The maintenance of these details comes into the task of the administrator where he is responsible to maintain the details like Username, Password, 4-digit MTPN, name, age, address and phone number. All these things will be maintained by the system. The system has functions based on the Recharge, Pay bills, Book Travels, Money Transfer, Pay for Movies via Entertainment Tab, Order Food etc... E-wallets and Payments should have good user interfaces with menus and toolbars. The main goal of this project is designing and developing an interactive online payment application. These details are routinely maintained depending on the numerous inputs and the changes made in the system database. The alterations can happen spontaneously like performing calculations on the direct inputs. This system works on input and output entry mechanisms. It helps in achieving better transactions and more customer satisfaction. This portal is based on a concept to maintain and generate all the transaction records of all cards. From this system, the user can easily maintain each and every payment record. The user can change account details and card information anytime. Transactions can be searched according to their used years or price and payment type which makes it easy to search for the customers. There's no chance of data misuse or loss & it's not time-consuming. The whole project is developed in JAVA, different variables and strings have been used for the development of this project.

It's easy to operate and understand by the users.

1.3 Outcomes of Project

E-wallets and Payment is an application that computerizes the payment procedure which we are aware of. This is a windows based application. This helps in managing data related to buyers and sellers of any product with the payment. Business reports can also be generated and viewed.

Existing system

In the existing E-wallets and Payment Portal, all the transactions are done in offline mode. The response is very slow and it is tough to retrieve particular data. Online data capture and modification is unavailable. The records that are maintained in MS Excel cannot be shared easily in a multi-user environment. There is a lack of security grants access to anyone which may lead to data misuse. These are the limitations of the existing system.

Proposed System

The E-wallets and Payment System application completely automates the existing system. It handles the new and used payments inventory, Front end is developed. Every detail of a transaction like name, transaction amount, type(in case of particular payment type), price, etc are recorded. This window based application is designed with user friendly computerized system with which a particular data can be retrieved with ease.

Car Sales System Modules

This application is divided into following modules

1. Addition of payments and its details

- Viewing the transactions

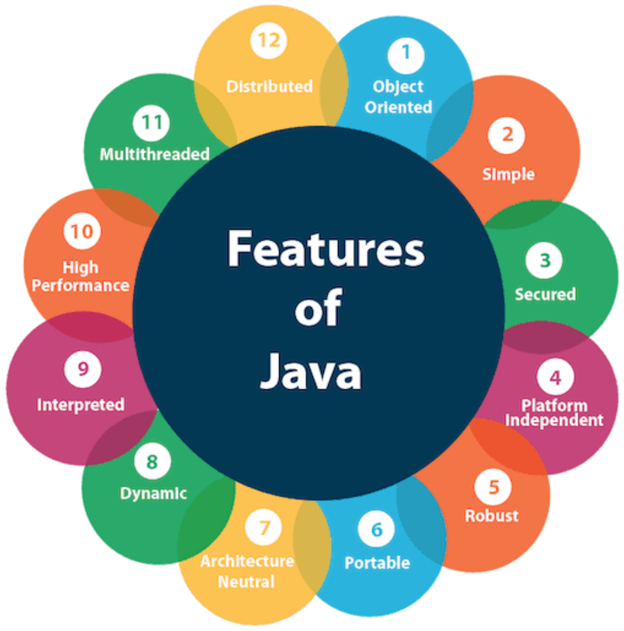
  Searching for a transaction based on type and price and name of the payment.

CHAPTER2

JAVA FEATURES AND OOPs CONCEPTS

2.1 Features of Java

The prime reason behind creation of Java was to bring portability and security features into a computer language. Beside these two major features, there were many other features that played an important role in moulding out the final form of this outstanding language. Those features are:



2.1 Features of Java

Simple

Java is easy to learn and its syntax is quite simple, clean and easy to understand.The confusing and ambiguous concepts of C++ are either left out in Java or they have been re-implemented in a cleaner way.

Eg *:* Pointers and Operator Overloading are not there in java but are an important part of C++.

Object Oriented

In java everything is Object which has some data and behaviour. Java can be easily extended as it is based on the Object Model.

Robust

Java makes an effort to eliminate error prone codes by emphasizing mainly on compile time error checking and runtime checking. But the main areas which Java improved were Memory

Management and mishandled Exceptions by introducing automatic **Garbage Collector** and **Exception Handling**.

Platform Independent

Unlike other programming languages such as C, C++ etc. which are compiled into platform specific machines. Java is guaranteed to be a write-once, run-anywhere language.

The Java program is compiled into bytecode. This bytecode is platform independent and can be run on any machine, plus this bytecode format also provides security. Any machine with Java Runtime Environment can run Java Programs.

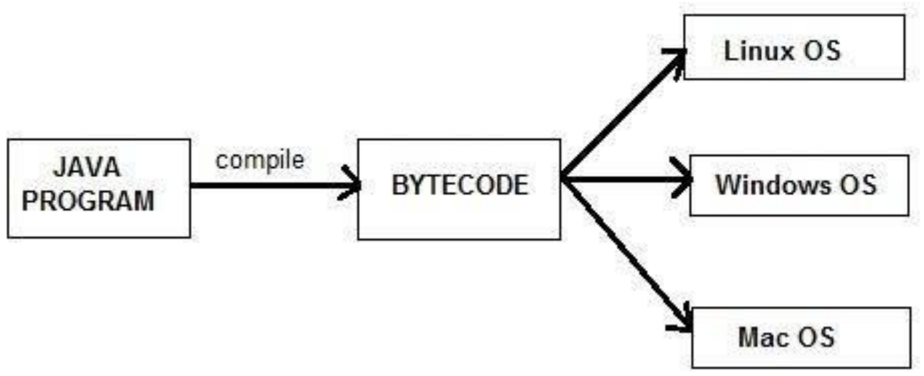


Fig 2.2 Platform Independent

Secure

When it comes to security, Java is always the first choice. With Java secure features it enables us to develop virus free, tamper free systems. Java programs always run in a Java runtime environment with almost null interaction with system OS, hence it is more secure.

Dynamic

We dynamically load classes when required. The memory management with garbage collection capability is another reason for its dynamicity.

Interpreted

Java code is compiled to convert it to Byte-code. This code is later interpreted by the interpreter which makes this language platform independent.

Multi Threading

Java multithreading makes it possible to write a program that can do many tasks simultaneously. Benefit of multithreading is that it utilizes the same memory and other resources to execute multiple threads at the same time, like While typing, grammatical errors are checked along.

Architectural Neutral

Compilers generate bytecodes, which have nothing to do with a particular computer architecture, hence a Java program is easy to interpret on any machine.

Portable

Java Byte code can be carried to any platform. No implementation dependent features.
Everything related to storage is predefined, example: size of primitive data types

High Performance

Java is an interpreted language, so it will never be as fast as a compiled language like C or C++.
But, Java enables high performance with the use of just-in-time compilers.

Distributed

Java has the ability to create distributed applications by means of RMI and EJB. This property of Java enables the user to demand any file by calling any method on any system.

2.2 OOPs (Object-Oriented Programming System) :

Object means a real-world entity such as a pen, chair, table, computer, watch, etc. **Object-Oriented Programming** is a methodology or paradigm to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts:

1. Object
2. Class
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation

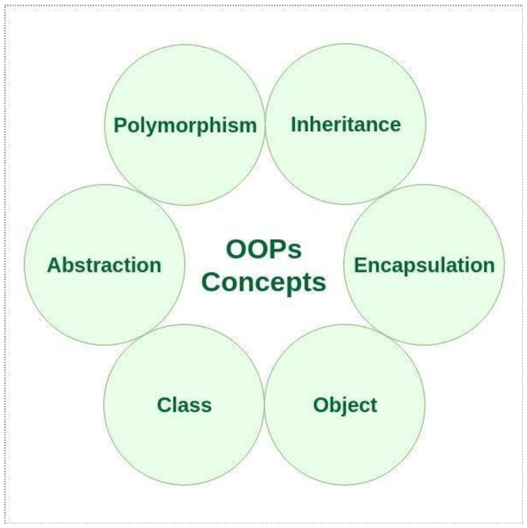


Fig 2.3 OOPs Concepts

Object

Any entity that has state and behaviour is known as an object. For example a chair, pen, table, keyboard, bike, etc. It can be physical or logical.

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

Example: A dog is an object because it has states like color, name, breed, etc. as well as behaviours like wagging the tail, barking, eating, etc.

Class

Collection of objects is called class. It is a logical entity.A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

Eg: class date

```
	{
		public: int
		day; int
		month;
		int year;
	};
```

Inheritance

When one object acquires all the properties and behaviours of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism

Types of inheritance :Single inheritance

Multiple inheritance

Multilevel inheritance

Hybrid inheritance

Hierarchical inheritance .

Polymorphism

If *one task is performed in different ways*, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.In Java, we use method overloading and method overriding to achieve polymorphism.

There are 2 types of polymorphism: 1.Function overloading

2.Operator overloading

Function overloading – It allows programmers to define or use two or more functions with the same name and in the same scope

Operator overloading – It allows you to redefine the way any operator works for user defined types only.It cannot be used   for   built in types.

Abstraction

Hiding internal details and showing functionality is known as abstraction. For example phone calls, we don't know the internal processing.

In Java, we use abstract class and interface to achieve abstraction.

Encapsulation

Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, it is wrapped with different medicines.

A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

CHAPTER 3

REQUIREMENTS AND DESIGN

3.1 Hardware Specification

SL.NO	COMPONENT	REQUIREMENT
1	Processor	Intel core i5
2	Speed	1.8 GHz
3	RAM	256MB (min)
4	Hard disk	10GB

3.2 Software Specification

SL.NO	COMPONENT	REQUIREMENT
1	Operating system	Windows 10
2	Programming language	JAVA
3	IDE	Eclipse neon
4	Front-end	Java swing
5	Back-end	MySQL database

3.3 Class Diagram

Class diagrams are the most popular UML models. They show members of a class and its associativity with other classes in a well-structured manner. Aggregation and Composition shown in the class diagram help analyse its functional behaviour with other classes.

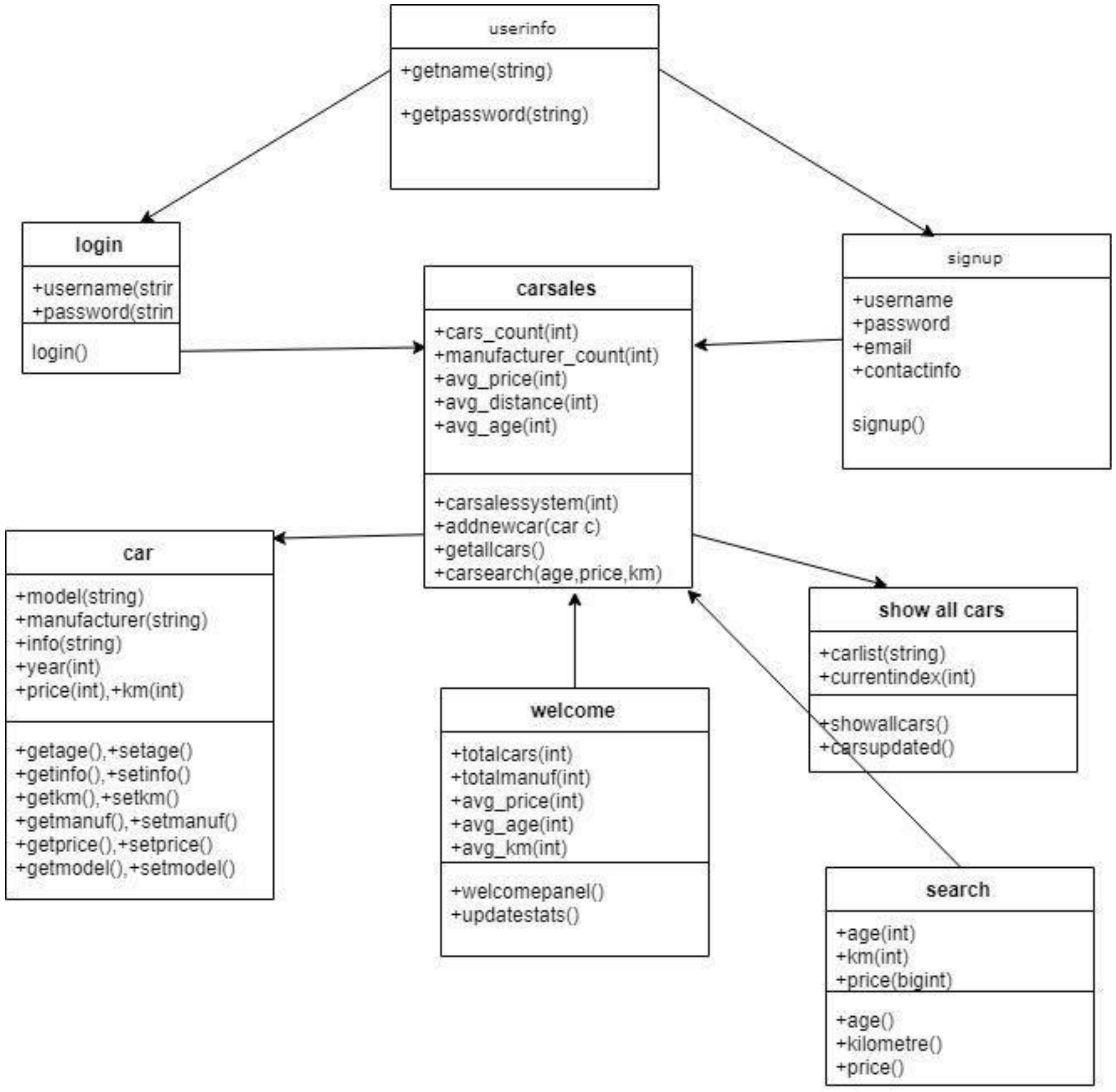


Fig 3.1 Class Diagram

3.4 Data Flow Diagram

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

This diagram helps represent the dataflow of the functionality. A well-designed data flow diagrams can help understand requirements more accurately in a graphical way.

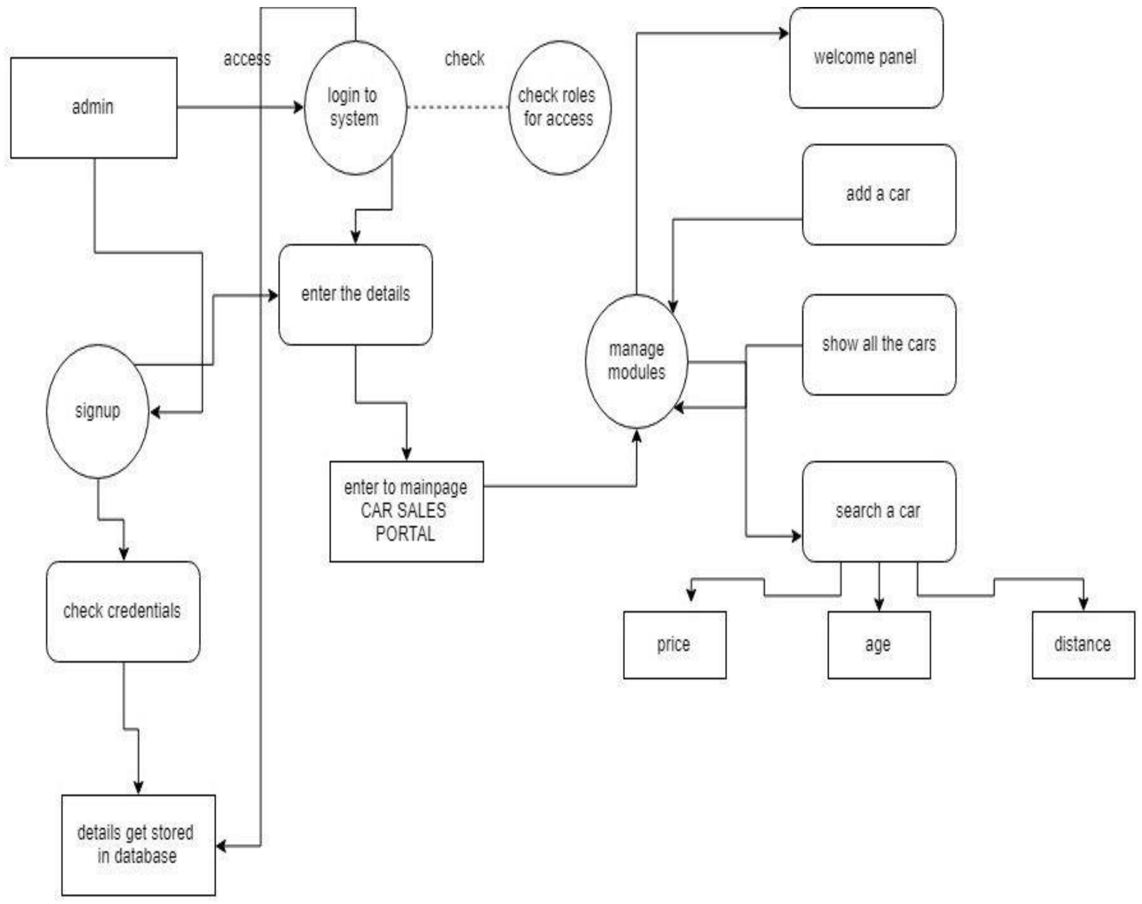


Fig 3.2 Data Flow Diagram

3.5 Use Case Diagram

This representation is the simplest of all UML design models. The model's main focus is to depict the interactive relationship between the user and the system. The stick figures in the sides are the different kinds of users possible with minimum or almost interaction with the system. The in between ovals represent different use cases. The cases are written in circles.

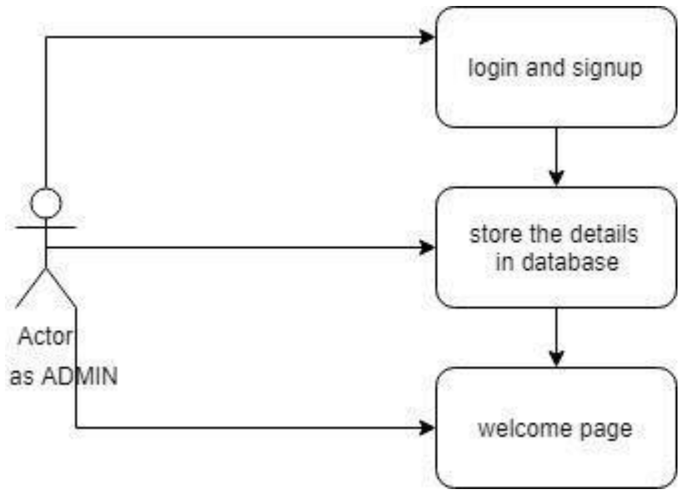


Fig 3.3 Use Case Diagram-1

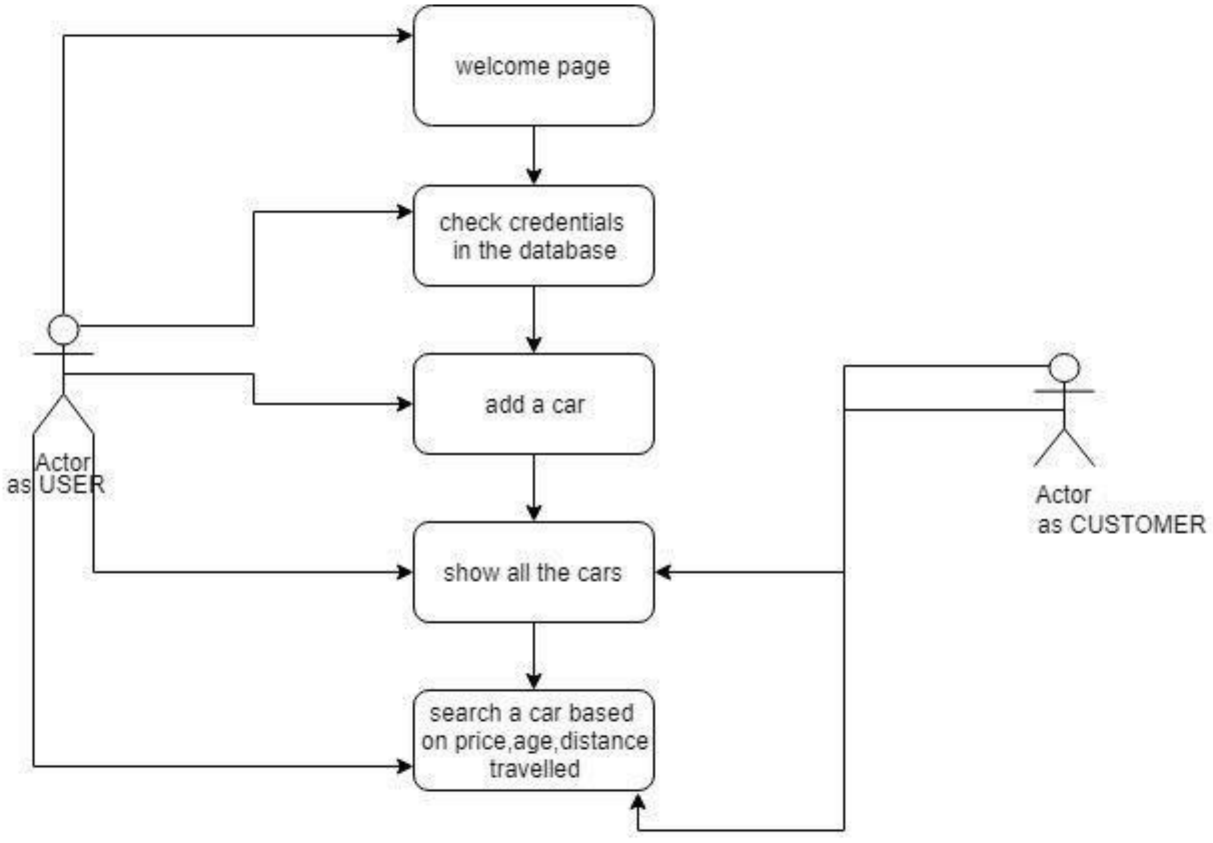

welcome page
check credentials in the database
add a car
show all the cars
search a car based on price,age,distance travelled
Actor as USER
Actor as CUSTOMER

Fig 3.4 Use Case Diagram -2

3.6 Sequence diagram

A **sequence diagram** describes an interaction among a set of objects participated in a collaboration (or scenario), arranged in a chronological order; it shows the objects participating in the interaction by their "lifelines" and the messages that they send to each other.

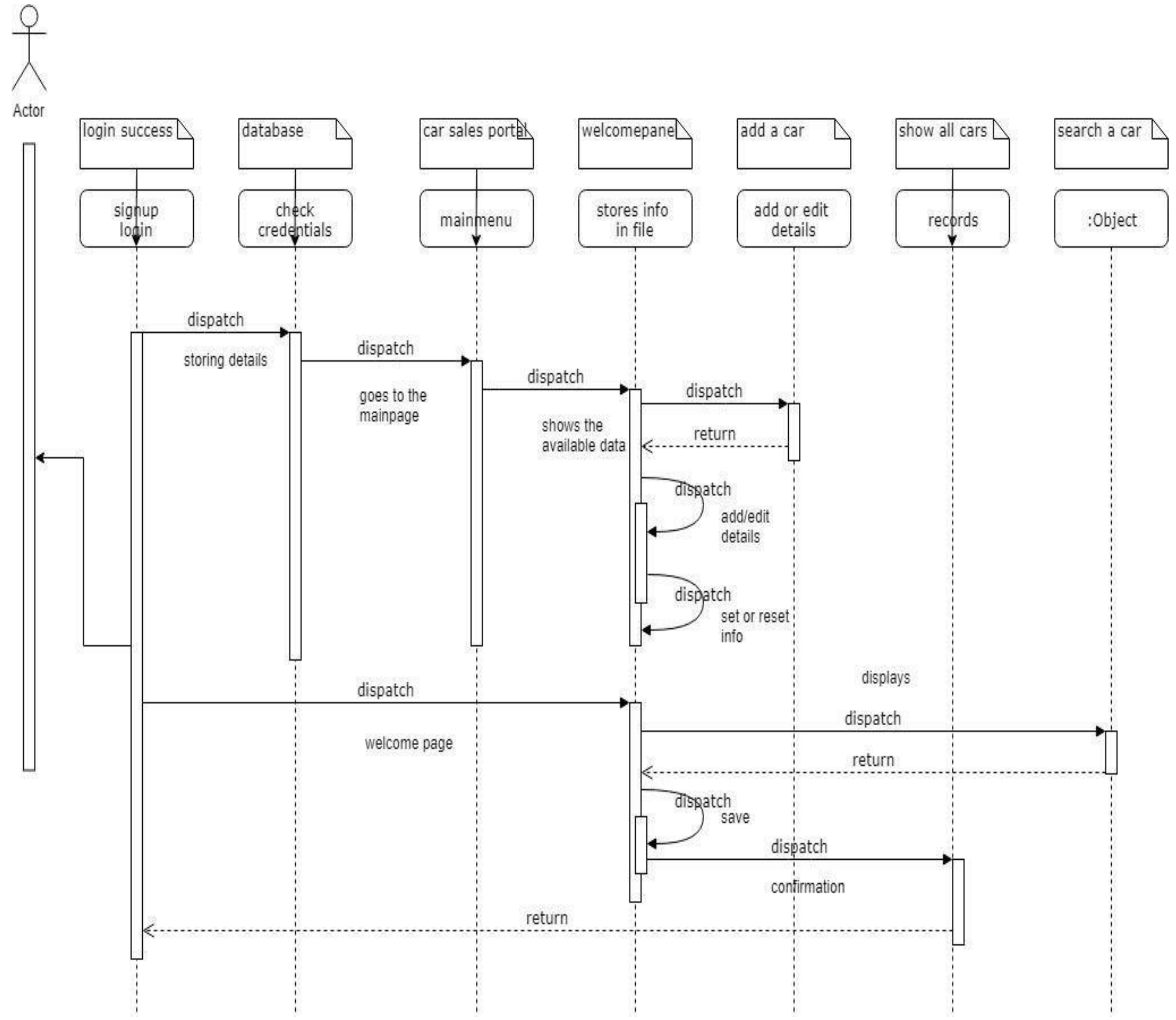


Fig 3.5 Sequence Diagram

CHAPTER 4

IMPLEMENTATION

The project implementation has three basic goals

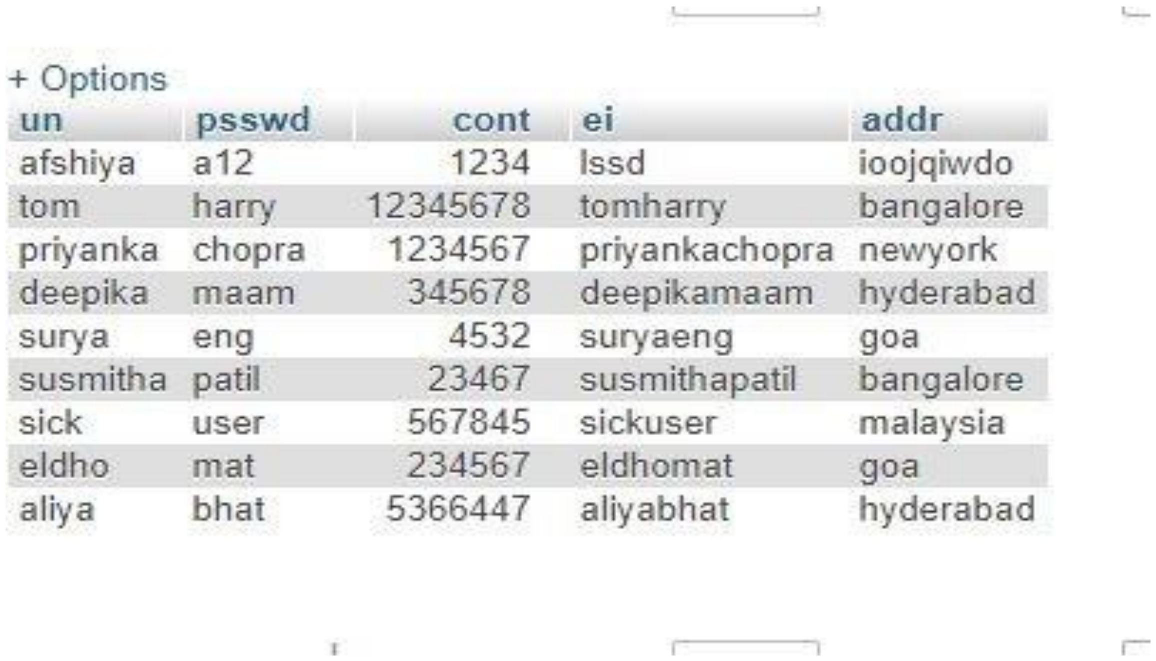
- To implement required back-end
- Implementation of the problem statement
- To design an effective front-end

4.1 Back-End

A structured database is used to store the details such as login information, and sign-up information . MySQL is the database management system used for the implementation of these tables of data.

The login information consists of the user name and password fields which are retrieved during login to check user authentication.

The signup information contains the username ,password ,contact info, and address which are retrieved in the database named login and therefore inside the login database a table named info is created where all the login and sign-up information is stored.



+ Options

un	psswd	cont	ei	addr
afshiya	a12	1234	lssd	ioojqiwdo
tom	harry	12345678	tomharry	bangalore
priyanka	chopra	1234567	priyankachopra	newyork
deepika	maam	345678	deepikamaam	hyderabad
surya	eng	4532	suryaeng	goa
susmitha	patil	23467	susmithapatil	bangalore
sick	user	567845	sickuser	malaysia
eldho	mat	234567	eldhomat	goa
aliya	bhat	5366447	aliyabhat	hyderabad

Fig 4.1 Database for login and signup

4.2 Implementation of Problem Statement

After implementation of login and signup page we enter into a new panel named payments. This page is named as E-wallets and Payment. Here, you have many panels named Receive, Transactions, Saved cards, Help and FAQs, Your account etc. In the first panel, we can first add our card with entering the details - be it Debit or Credit or even Mastro/Master cards. The next panel is based on checking the details of our account and also gets an initial debit of Rs.1 which is initially always present in the wallet before we add any further amount. Then we can finally make payments from our application. We can Recharge, Pay bills, Book Travels, Money Transfer, Pay for Movies via Entertainment Tab, Order Food etc. . Based on this information you show all the transactions stored in the application. You are searching a transaction based on name, type and amount when you also can set and reset the information by using the set and reset buttons.

4.3 Front-End

For the GUI part of the project, we have used java swing. Swing is a part of JFC (java foundation classes). It's a platform independent API particularly created to assist Graphical User Interface in Java.

Unlike AWT (Abstract Window Toolkit), which is platform dependent, swing is often called lightweight as they do not require a lot of allocation in OS's windowing toolkits.

The basic swing hierarchy is as follows,

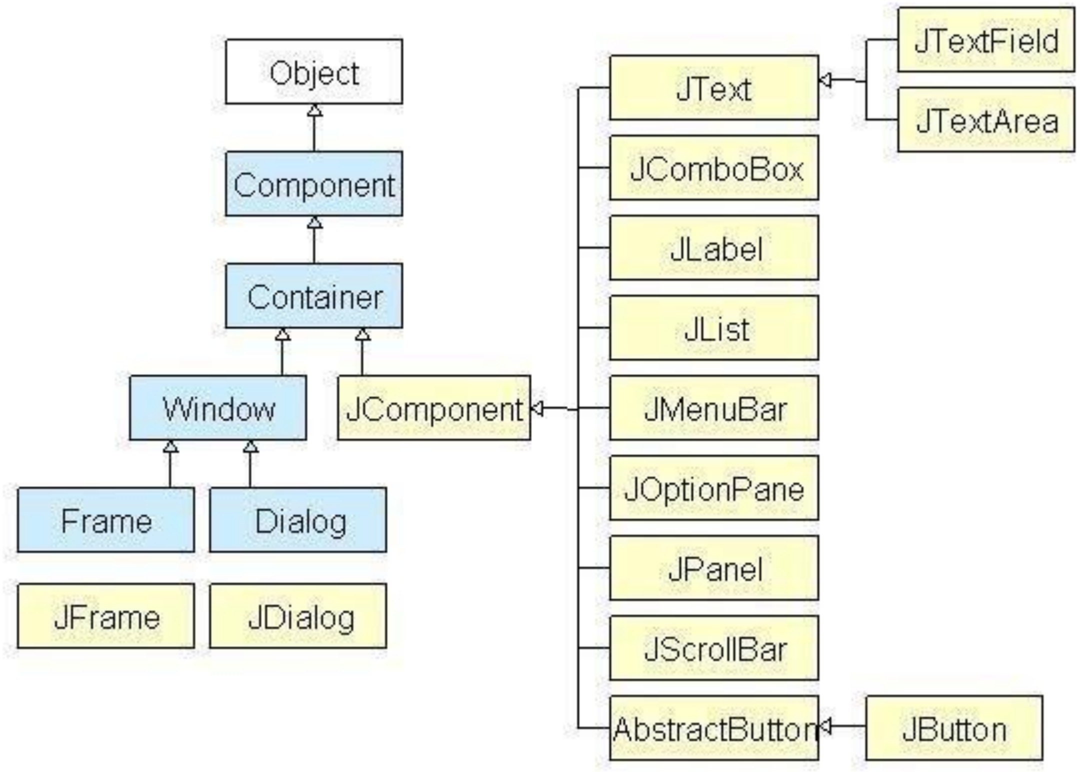


Fig. 4.3 Java Swing Hierarchy

A component is an individual control item, such as a button or table. A container usually holds a group of such components. Thus, a container can also be called a component since you can add many of them on the existing one just like adding components. Furthermore, for any component to be visible, it has to be held within a container. Thus, by default all Swing based GUIs will have a minimum one container. This in Swing is defined as **containment hierarchy** . Utilising such components an effective interface for users to interact with the software has been created.

CHAPTER 5

OUTPUT SNAPSHOTS

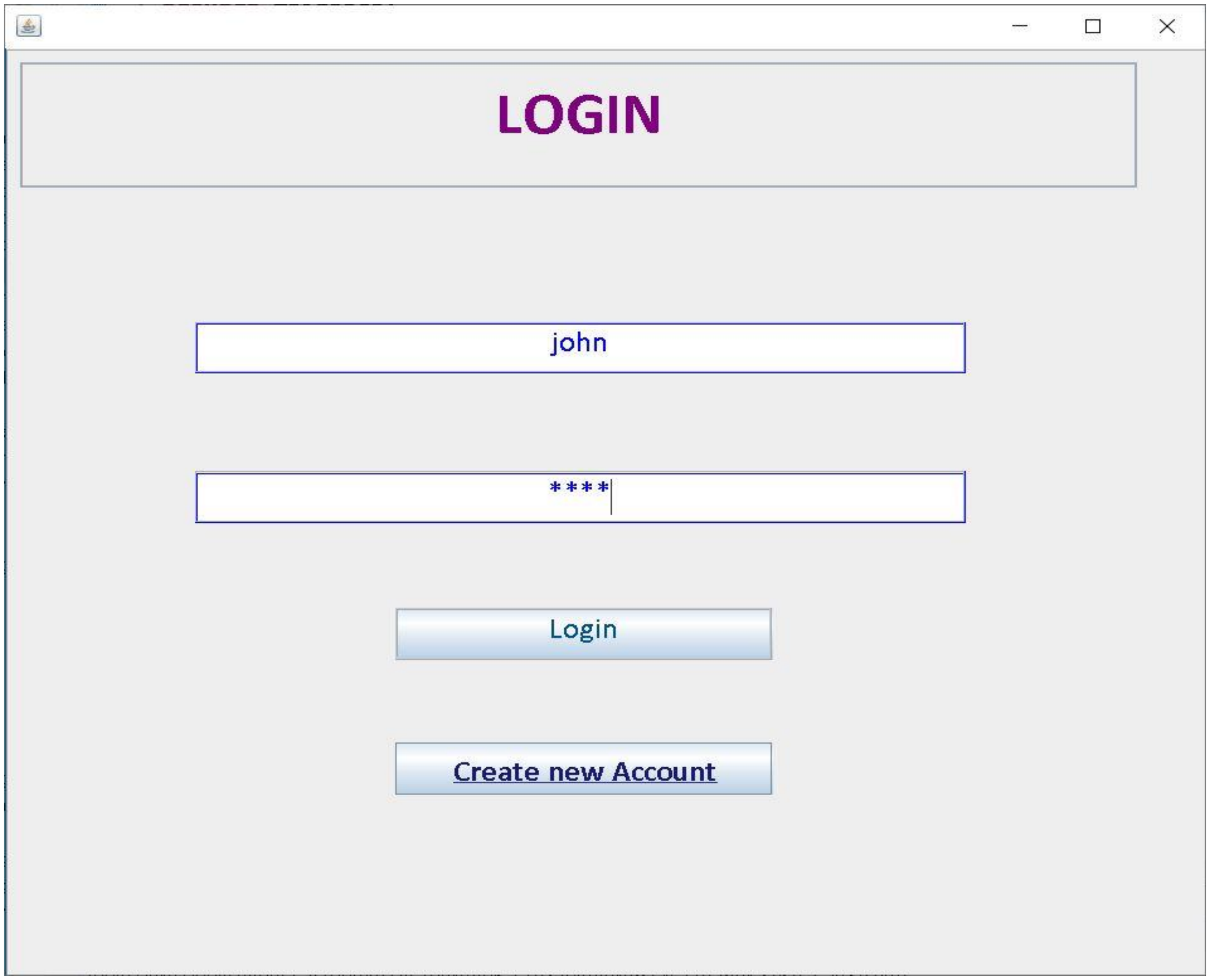
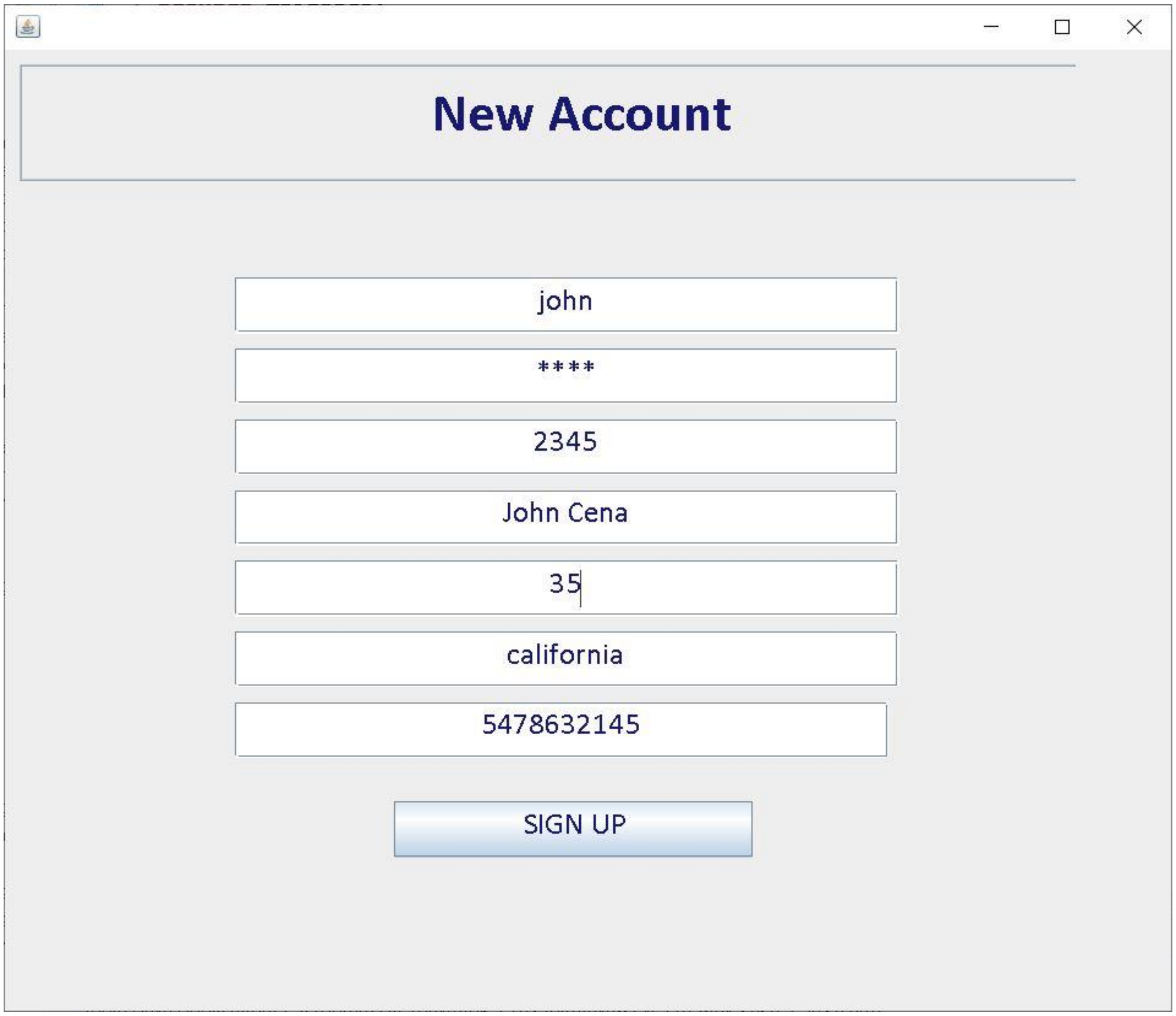


Fig 5.1 :Login



New Account

john

****

2345

John Cena

35

california

5478632145

Fig 5.2: Signup

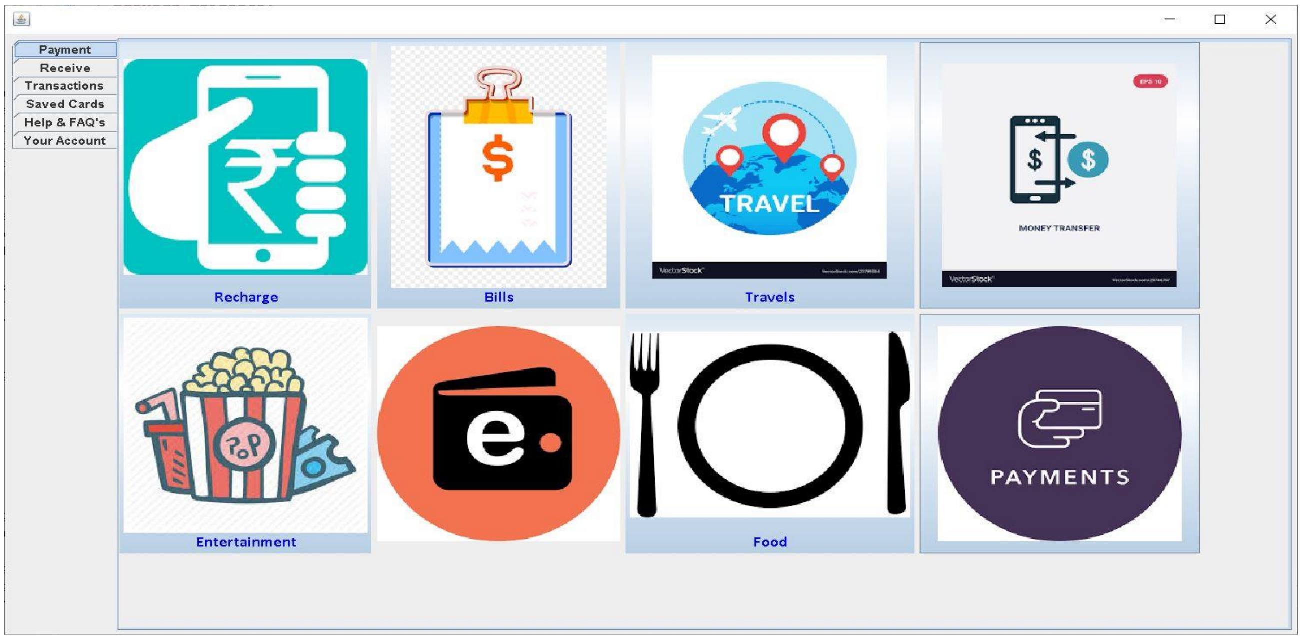


Fig 5.3 : Payment Tab

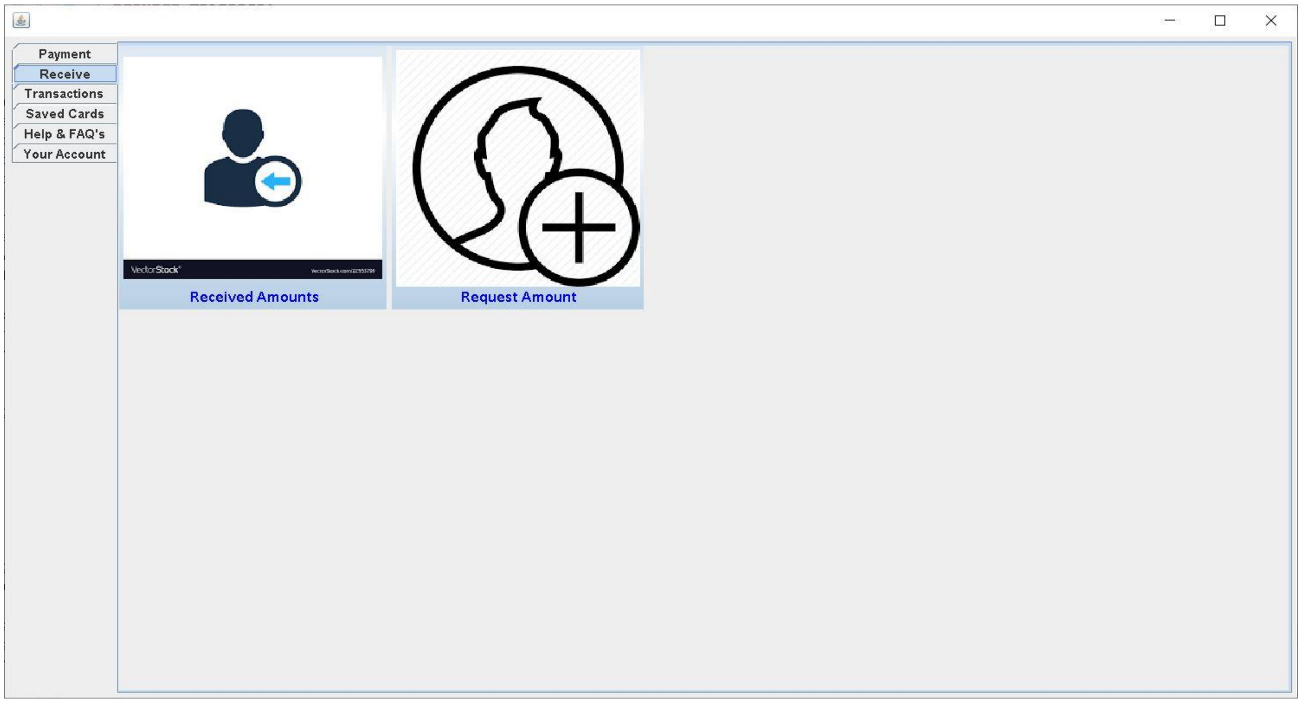
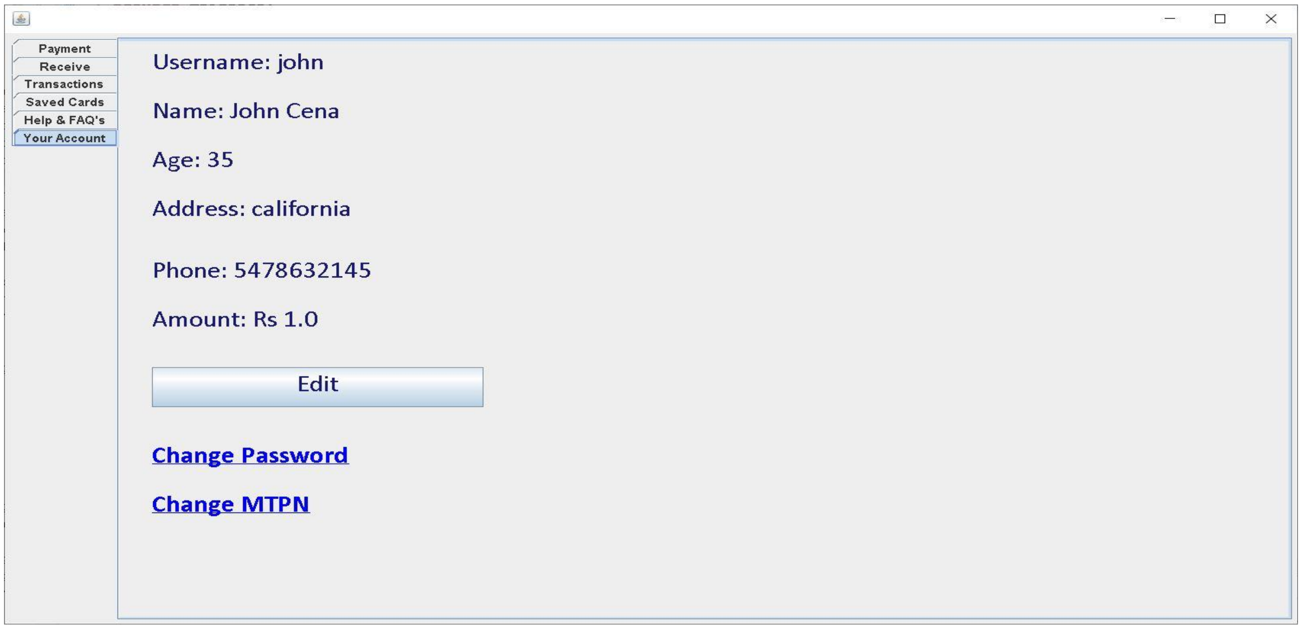


Fig 5.4 : Receiver Tab



Username: john

Name: John Cena

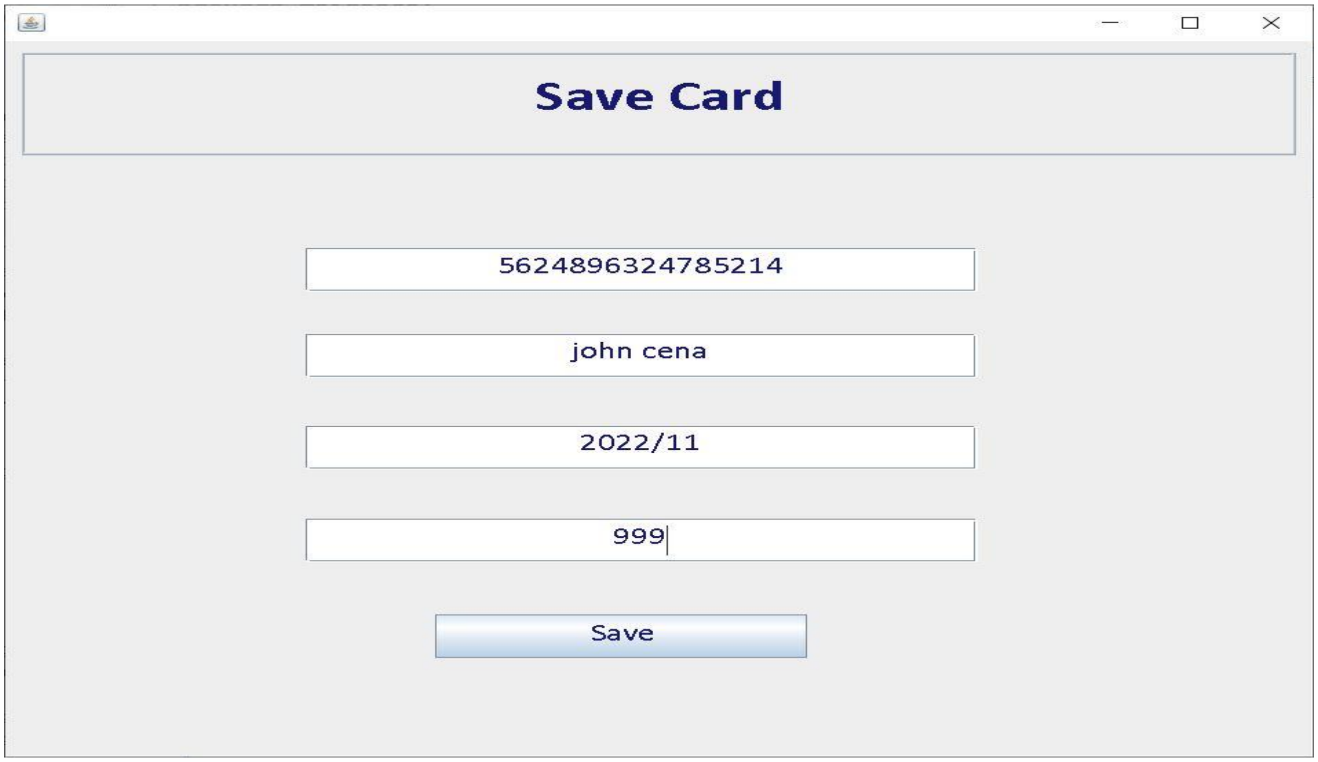
Age: 35

Address: california

Phone: 5478632145

Amount: Rs 1.0

Fig 5.5 : Account Details



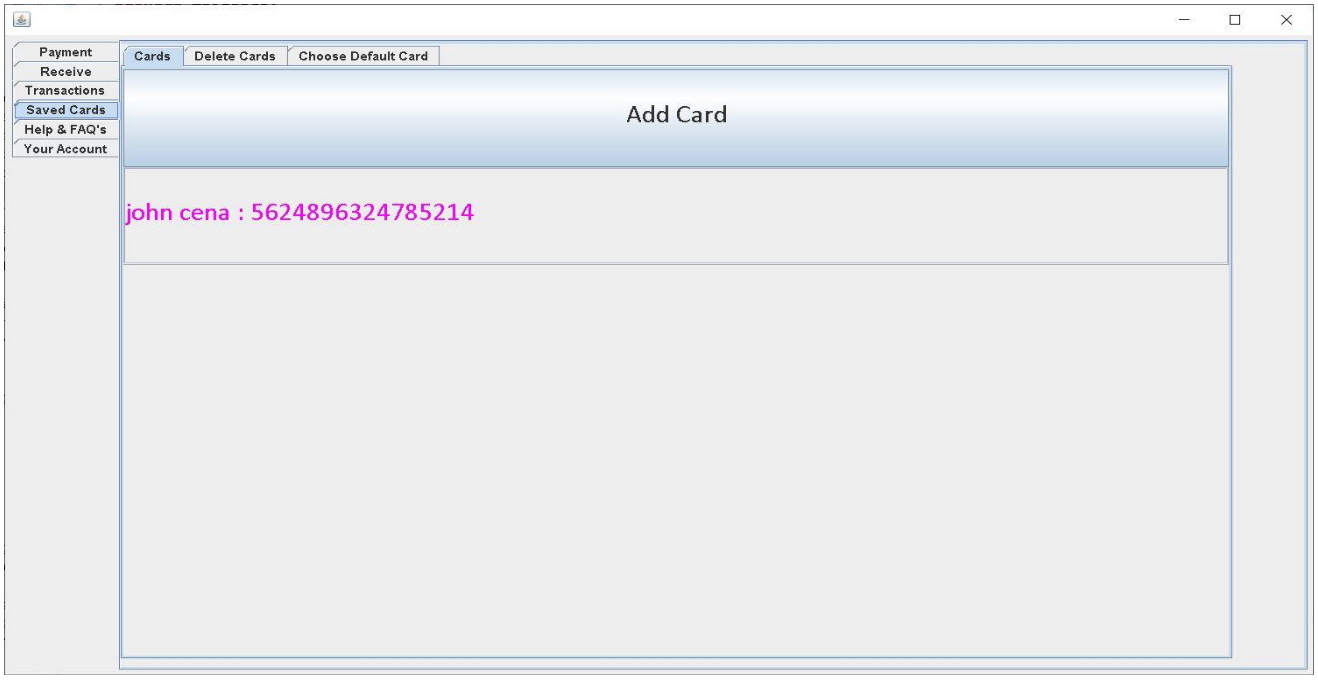
5624896324785214

john cena

2022/11

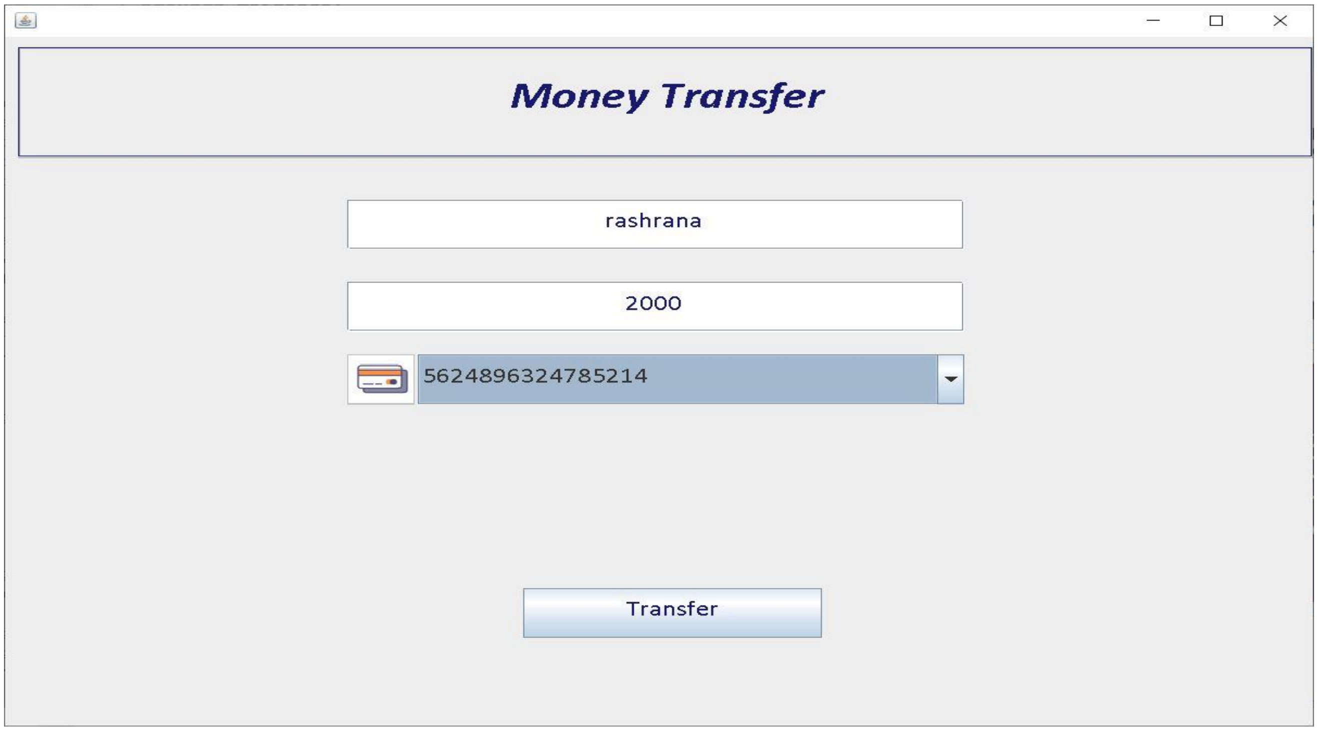
999

Fig 5.6 : Add Cards



john cena : 5624896324785214

Fig 5.7 : Saved Cards

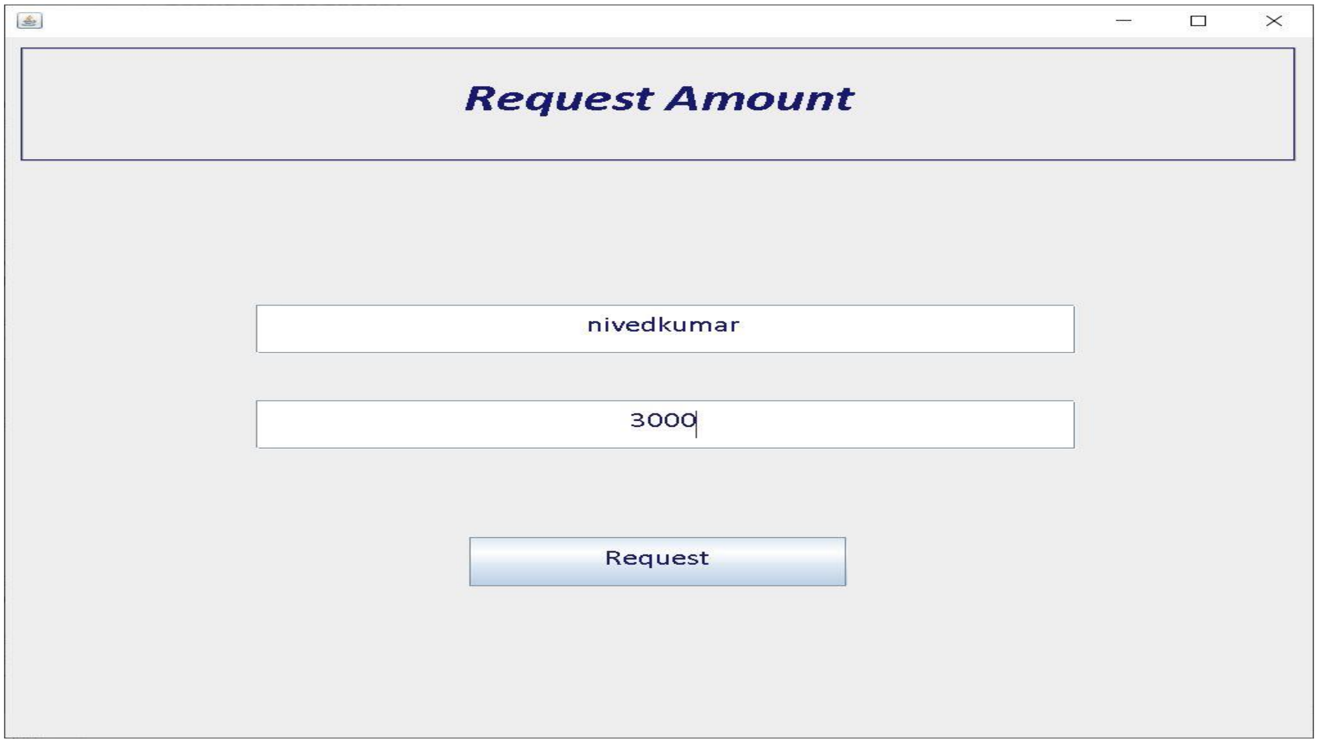


rashrana

2000

5624896324785214

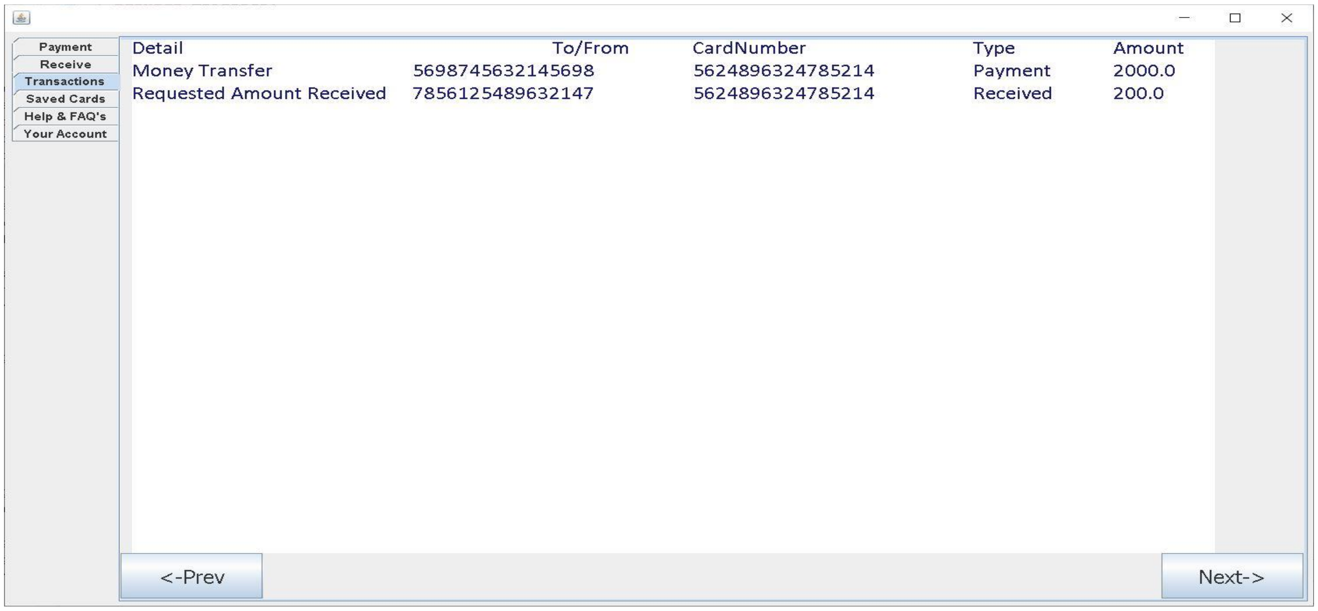
Fig 5.8 : Money Transfer



nivedkumar

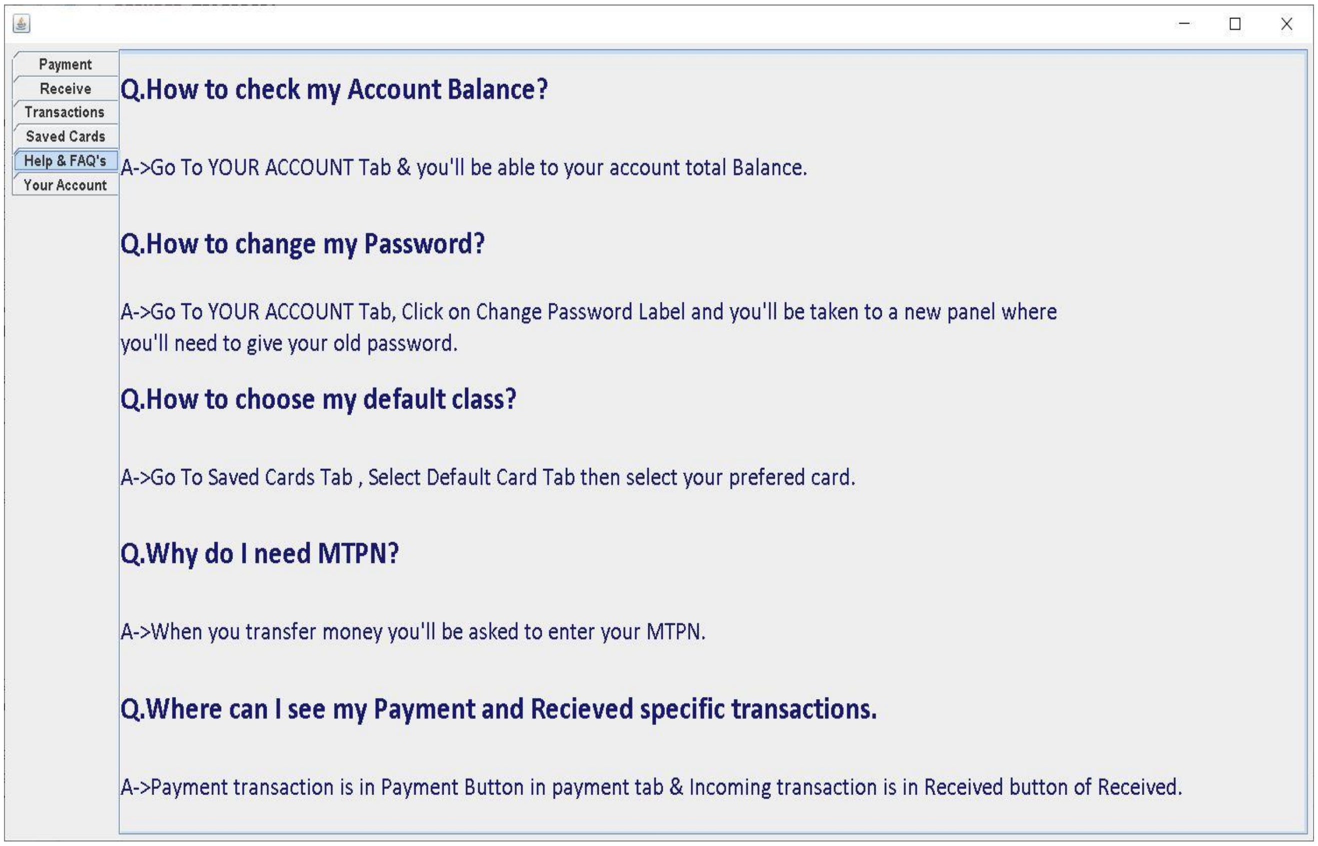
3000

Fig 5.9 Request Money



Detail	To/From	CardNumber	Type	Amount
Money Transfer	5698745632145698	5624896324785214	Payment	2000.0
Requested Amount Received	7856125489632147	5624896324785214	Received	200.0

Fig 5.10 : Transactions



Q.How to check my Account Balance?

A->Go To YOUR ACCOUNT Tab & you'll be able to your account total Balance.

Q.How to change my Password?

A->Go To YOUR ACCOUNT Tab, Click on Change Password Label and you'll be taken to a new panel where you'll need to give your old password.

Q.How to choose my default class?

A->Go To Saved Cards Tab , Select Default Card Tab then select your prefered card.

Q.Why do I need MTPN?

A->When you transfer money you'll be asked to enter your MTPN.

Q.Where can I see my Payment and Recieved specific transactions.

A->Payment transaction is in Payment Button in payment tab & Incoming transaction is in Received button of Received.

Fig 5.11 : FAQs

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

This project satisfies the needs to manage the common people needs thus making it user friendly to the user .At the end of the project I am able to conclude that

- A description of the background and context of the project and its relation to work is already done in the area
- Made statement for the aims and objectives of the project
- The description about the sources, scope and all of those is describe efficiently
- We define the problem on which we are working in the project
- We understand the domain that describes the operations being implemented in the program
- We designed user interface and security issues related to the system
- Finally the program is implemented and tested according to the test cases related to today's scenarios.

.

6.2 FUTURE SCOPE OF THE PROJECT

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

- We can add every PC and mobile transaction in future to see all that the user has made his/her payments to.
- We can give more advanced software for this project by including more facilities.
- We will host the platform on online servers to make it accessible worldwide.
- We integrate multiple load balancers to distribute the loads of the system.

- Create a master and slave database structure to reduce the overload of the database queries.
- Implement the backup mechanism for taking backup of codebase and database on a regular basis on different servers.
- We should make sure the file documentation is done from time to time so that each and every information related to the E-wallets and Payment portal is stored.

CHAPTER 7

REFERENCES

[1] https://www.scribd.com/document/17842767/DBMS-car-sales

[2] www.itprojectz.com/itprojectz_new/c-projects-on-carsalessystem

[3] https://www.slideshare.net/mobile/nirdhishwar/carsales

[4] www.programmingwithbasics.com/2016/03/database

[5] https://www.javatpoint.com/java-swing

[6] https://www.geeksforgeeks.org/carsales/

[7] https://www.tutorialspoint.com/jfreechart/