# Contents

**List of figures:**

# Chapter 1: Introduction

## 1.1 Background

This is the money transfer application used in e-banking and online payment system. The report provides the basic knowledge of java technology (Swing) for application development along other different technologies such as MySql as database. The report is organized in four chapters' including introduction, requirements, development and testing, and conclusion. Introduction chapter describes about existing system and developed system. Chapter two describes requirements that are fulfilled by system and how those requirements are fulfilled. Third chapter insights about development stages and testing work. Final chapter is conclusion which concludes the report.

The goal of this project is to create an application that allows the consumer to directly payment the product from the seller over the internet using this application and it also help to send or receive money from other resource like e-payment. This goal is achieved by using java technology like Swing and MYSQL as database.

It is a way to send and receive money online without needing your credit or debit card details. More people now use e-payment because they can make it easier to transfer money and make purchases.

## 1.2 Issue/ Problem with the report

Some of the problems of this report are:

i. How are the users benefitted by this service?
ii. How secure and reliable is the data that are saved in the server?
iii. What are the effects of power failure on the app?

## 1.3 Objective of the report

Application will be made using JAVA programming language, eclipse-IDE and MySQL database will be used to store the data.

- Simple easy to understand interface.
- User deposit the money in their account.
- User can see their transaction history.
- User can recover their password easily.

## 1.4 Tools and technology used

- UI Designing using Pencil app
- Front-end: java swing, awt
- Back-end: java with MYSQL connector
- Code editor: eclipse and visual studio

# Chapter 2: Tasks and activities performed

## 2.1 Analysis of tasks

### 2.1.1 Functional requirements

The functional requirement describes what the system should do.

- System can only access valid user.
- User can see their transaction history.
- User can recover their password.
- User sends money through email.

#### 2.1.1.1 Admin requirements:

1. System access to only authorized administrator.
2. To be able to insert, update and delete the Transaction in database.
3. To check and keep Transaction record in database.
4. Able to manage history.

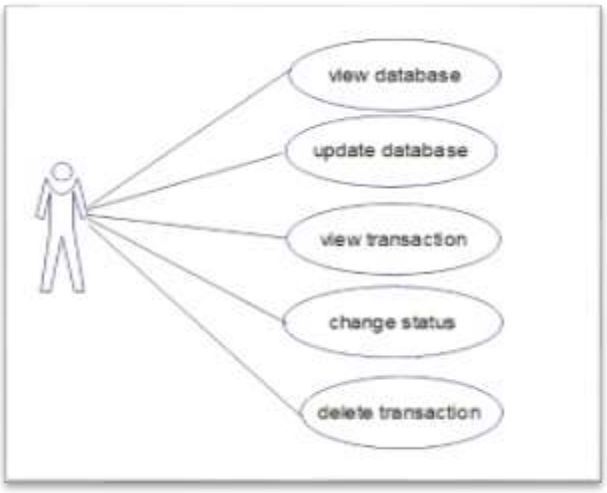



**Figure 1: Use case diagram involving the role of administrator**

## 2.2 Activities performed:

### 2.2.1 Database design

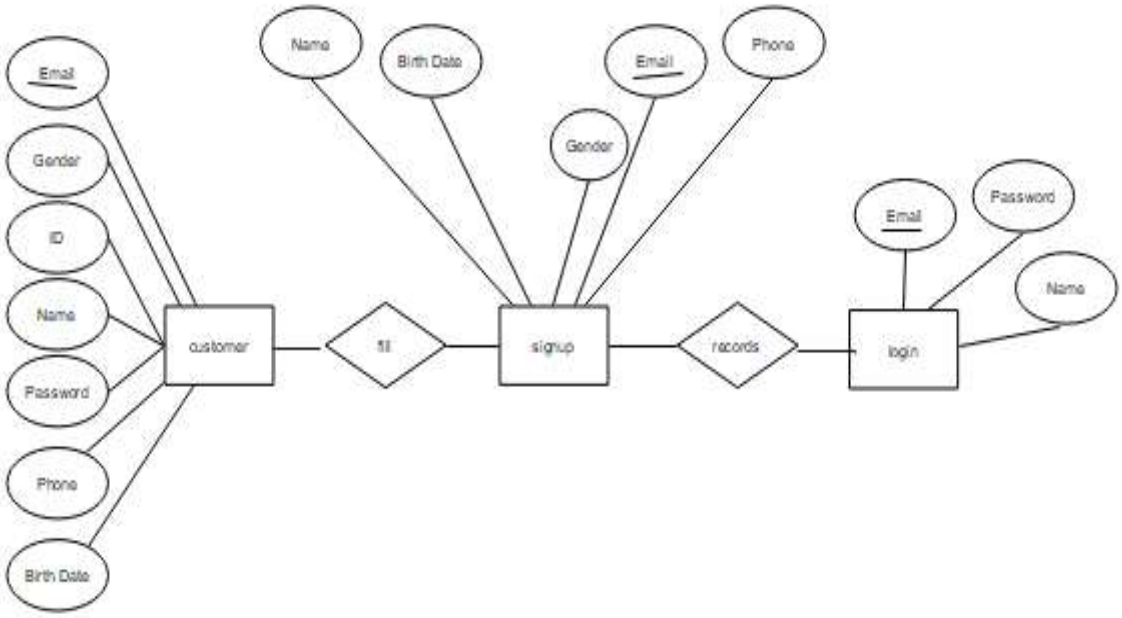
FOR SIGNUP

| # | Name | Type | Collation | Attributes | Null | Default | Comm |
|---|---|---|---|---|---|---|---|
| 1 | email | varchar(50) | utf8mb4_general_ci | | No | None | |
| 2 | password | varchar(10) | utf8mb4_general_ci | | No | None | |
| 3 | name | varchar(20) | utf8mb4_general_ci | | No | None | |
| 4 | dob | date | | | No | None | |
| 5 | gender | varchar(10) | utf8mb4_general_ci | | No | None | |
| 6 | money | varchar(100) | utf8mb4_general_ci | | No | None | |

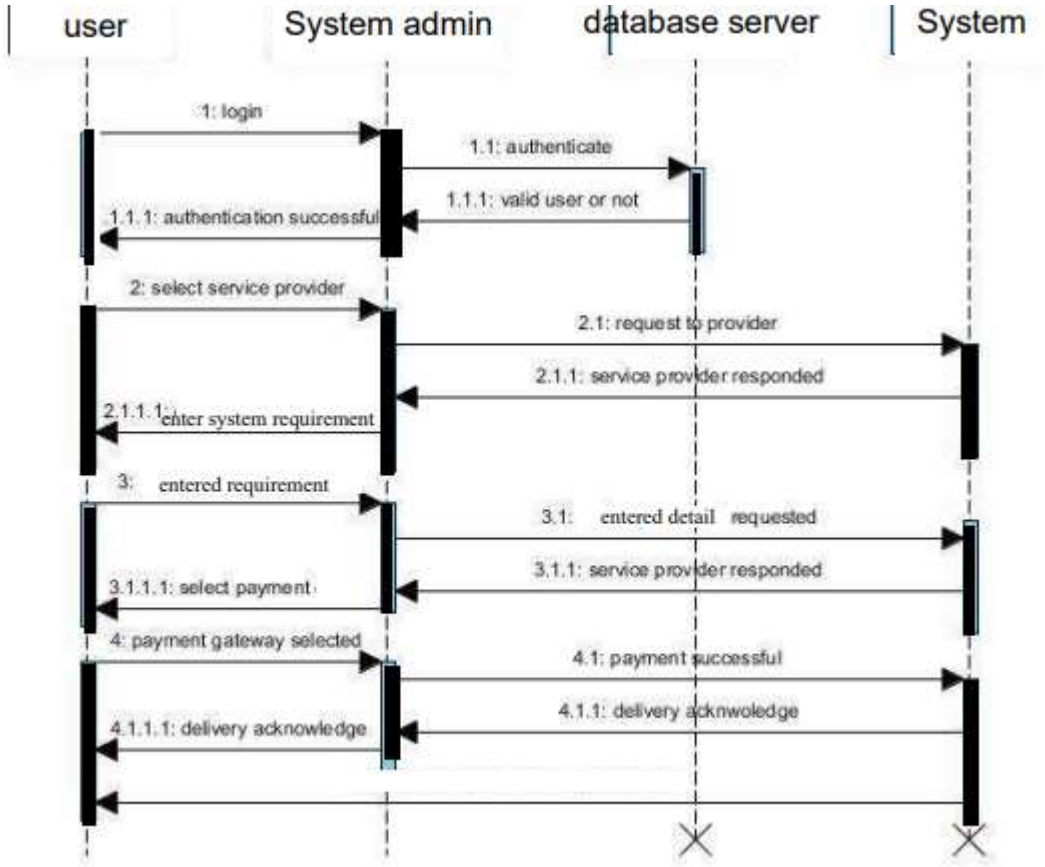
FOR TRANSACTION

| # | Name | Type | Collation | Attributes | Null | Default | Comme |
|---|---|---|---|---|---|---|---|
| 1 | customer_id | int(20) | | | No | None | |
| 2 | name | varchar(20) | utf8mb4_general_ci | | No | None | |
| 3 | date | date | | | No | None | |
| 4 | amount | varchar(10) | utf8mb4_general_ci | | No | None | |
| 5 | email | varchar(50) | utf8mb4_general_ci | | No | None | |

2.2.1.1 ER-Diagram



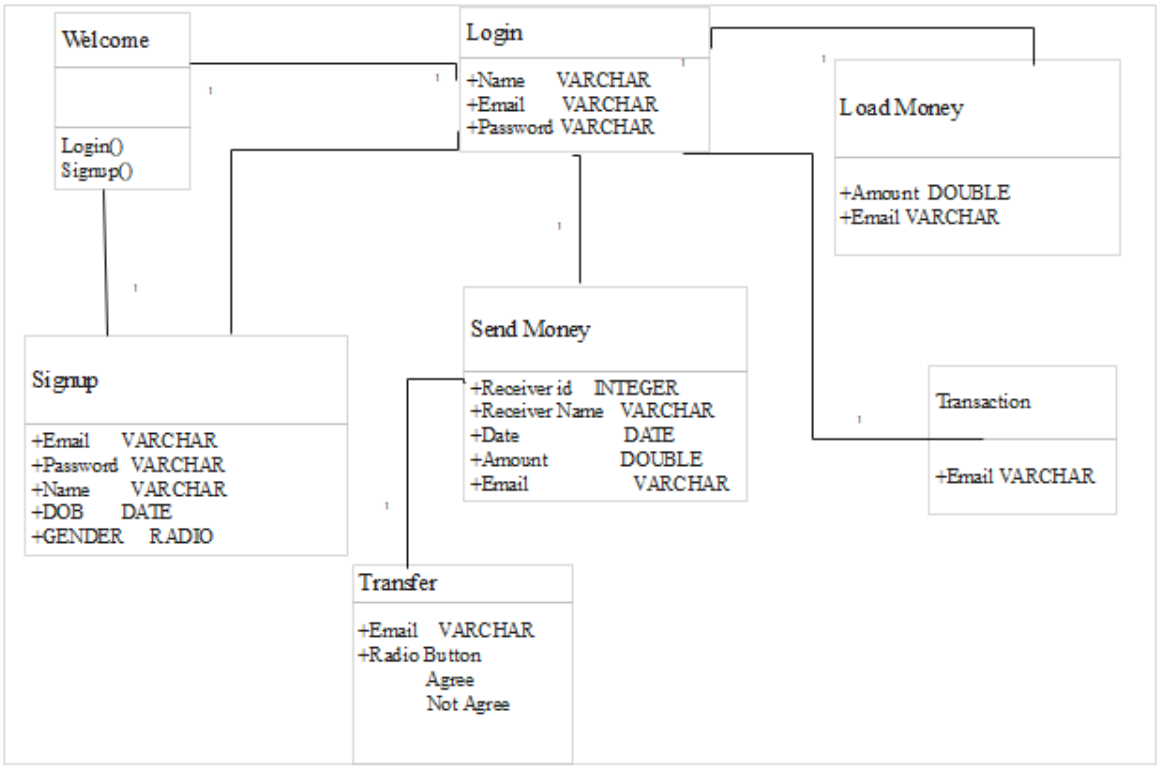
2.2.1.2 Sequence diagram



### 2.2.1.3 Class diagram

A class is a blueprint that is used to create Object. The class defines what the object can do. it gives an overview of a software system by displaying classes, attributes, operations and their relationship.

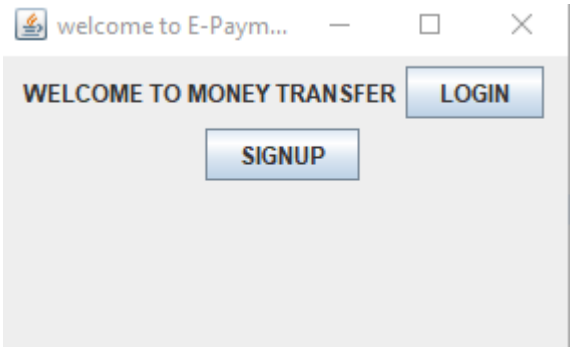
The class diagram of the application is shown in figure below.



## 2.3 `Software component

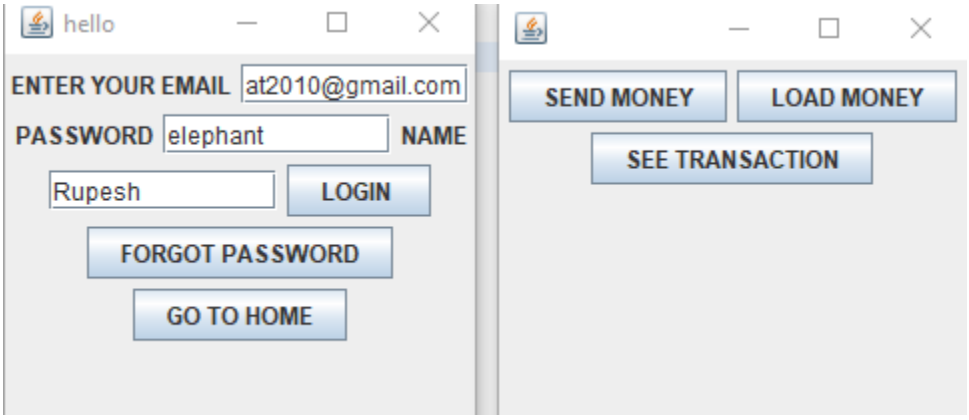
### 2.3.1 Customer UI

In this user can use two button. If the customer is new she/he will use signup button and if user is already registered then she/he can use login button.



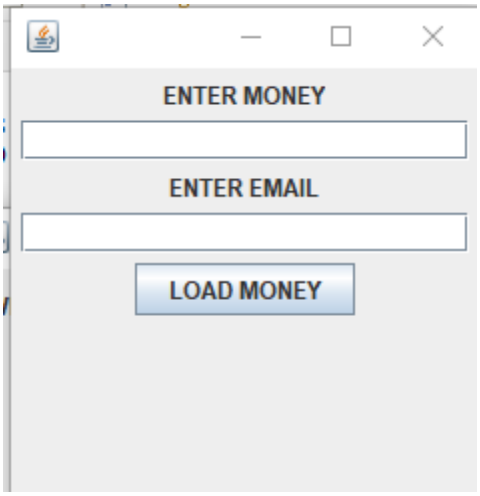

### 2.3.2 Login

After the login is completed then the system verify the user is valid or not. If the user is valid then the system opens the API. In that API there are three button. If customer choose send money if they want to send amount to other. Customer choose load money to deposit amount in their account. See transaction is used to see their old history made by the users.



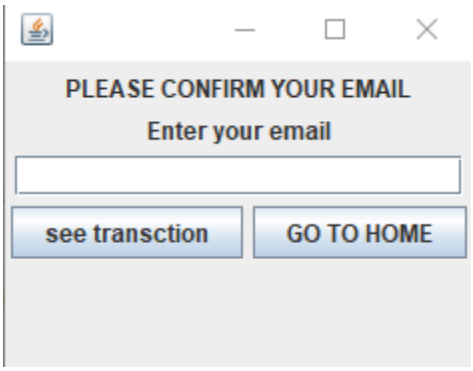
### 2.3.3 Deposit

In deposit button , the new API is opened and customer see two JTextField  one for money and another for email . money is used to deposit the amount but email is used to verify the user details .



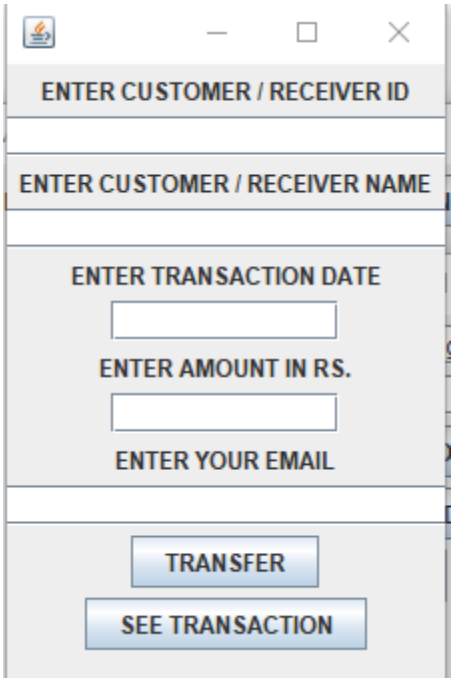

### 2.3.4 Transaction

User can easily view their history and their own details. Only valid users are able to see the details. User put email and only the button is enable to get action and if the history is seen then the user easily go to homeAPI.




### 2.3.5 Payment

After the send money button is clicked then only this API is opened. User can easily transfer money from one to another. All the information is entered and user click the transfer then the details are store in database(transaction).


ENTER CUSTOMER / RECEIVER ID
ENTER CUSTOMER / RECEIVER NAME
ENTER TRANSACTION DATE
ENTER AMOUNT IN RS.
ENTER YOUR EMAIL
TRANSFER
SEE TRANSACTION

## 2.4 Analysis of possible solutions

Some of the possible solutions for the problems existing in an organization can be discussed below.

1. The organization should install powerful server/s as the server can get loaded frequently. For this, the organization can install multiple servers as if a server gets lagged or is turned off for maintenance, the other server could help to operate tasks and activities smoothly and efficiently.
2. In the case of security, frequent checking and updating should be performed. The organization should associate with strong firewall protection from viruses and worms.
3. Power backups like UPS, generator or inverter should be used in case of power failure.
4. Frequent backups should be done for the databases with existing backup tools.

# Chapter 3: Development and testing

## 3.1 Development

Software used to develop this system

1. Designing using Pencil app
2. Front-end: java swing, awt
3. Back-end: java with MYSQL connector
4. Code editor: eclipse and visual studio

## 3.2 Testing the system

1. Test Login
2. Test payment
3. Test adding/editing of data

### 3.2.1 Test Case

| Project name: Money transfer |
|---|
| Test Case |
| Test Title :Users detail |
| Description: Test user details is valid or not for login |
| Pre-condition: users must have registered in signup |
| Post condition: valid user are only use system |
| |

| Step | Test Steps | Test Data | Excepted Result | Status Pass/fail |
|---|---|---|---|---|
| 1 | Navigate to user login page | Username:admin<br>Password:admin<br>Email:admin@gmail.com | Login page should open | pass |
| 2 | Navigate to user login page | Username:-----<br>Password:-----<br>Email:----- | Login page shouldnot open | fail |

# Chapter 4: Discussions and Conclusions

## 4.1 Conclusions

This app developed in simple method. It is understandable so any user can take the benefits of it. The extra skill or anything is not needed to use the app. The system helps to keep record of transaction and information.

# References

Money transfer Book pdf =>https://www.files.ethz.ch/isn/105513/Passing_the_Buck.pdf

MMTS service Book pdf =>

https://www.findevgateway.org/sites/default/files/publications/files/mfg-en-paper-mobile-money-transfer-services-the-next-phase-in-the-evolution-in-person-to-person-payments-aug-2010.pdf