# CHAPTER 1

# INTRODUCTION

Post office application will help in automating functions of the administration department it help in reducing the time spent in record keeping and work can be carried out easily. The old records can be easily accessed in future.

The project is a web based application for post office management and their customers. It handles all types of transaction details of the post office and their project will reduce the work and most of the work done by computers. It maintains all the old record for later reference and it have provision for automatic update as per the status are the main objectives of this project , post office also help in saving money of customer and do perform the withdraw and e_bill operations through their saving account so that transfer of money is digitalized even we also provided the insurance account and claimed money is passed to the saving accounts

## 1.1 PRODUCT PERSPECTIVE AND OBJECTIVE

This designed product in a follow-on member of an existing database that helps to enhance the existing database and provide wider scope for the database to be used by all the post offices. The burden on the human effort is reduced due to the implementation of this database.

## 1.2 Product Function:

- Store and retrieve the data easily and efficiently.
- Complete details of the customer.
- Securing the data present in the database.
- Handling the data efficiently and accurately
- Retrieving the entire entry using a keyword.

## 1.3 AIM AND PERSPECTIVE

To maintains all the old record for later reference and it can be updated or deleted at any point of time by only through the admins.

# CHAPTER 2

## SYSTEM REQUIRMENTS SPECIFICATION

System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems. The former may prevent a device or application from getting installed, whereas the latter may cause a product to malfunction or perform below expectation or even to hang or crash.

## 2.1 HARDWARE REQUIREMENTS

| | | |
|---|---|---|
| PROCESSOR | : | intel® Core(TM) i5 |
| RAM | : | 2GB minimum |
| ROM | : | 300MB minimum |

## 2.2 SOFTWARE REQUIREMENTS

| | | |
|---|---|---|
| Operating system | : | WINDOWS 10 |
| Programming language | : | JAVA |
| Frontend | : | JAVA frames |
| Back end | : | mysql MariaDB |

## PRIMARY KEY

Such type of candidate key which is chosen as a primary key for table is known as primary key. Primary keys are used to identify table there is only one primary key per table.

## FOREIGN KEY

Foreign key are those keys which is used to define relationship between two tables. When we want to implement relationship between two tables then we use concept of foreign key**.**

## MariaDB

MariaDB is developed as open source software and as a relational database it provides an SQL interface for accessing data. It is based on the structure query language (SQL), which is used for adding, removing, and modifying information in the database. Standard SQL commands, such as ADD, DROP, INSERT, UPDATE used in MariaDB.

In addition to web usage, MariaDB can be used for stand-alone applications ranging from enterprise transactional and analytics systems down to mobile devices, embedded with other software. MariaDB works in the cloud or on premise.

Many features contribute to MariaDB's standing as a database system. Its speed is one of its most prominent features. MariaDB is remarkably scalable, and is able to handle tens of thousands of tables and billions of rows of data. It can also manage small amounts of data quickly and smoothly, making it convenient for small business or personal projects.

### JAVA AND JFRAME

Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and handled devices, but later became a popular choice for creating web applications.

The Java syntax is similar to C++, but is strictly an object-oriented programming language. For example, most Java programs contain classes, which are used to define objects, and methods, which are assigned to individual classes. Java is also known for being more strict than C++, meaning variables and functions must be explicitly defined. This means Java source code may produce errors or "exceptions" more easily than other languages, but it also limits other types of errors that may be caused by undefined variables or unassigned types.

JFrame is a class in Java and has its own methods and constructors. Methods are functions that impact the JFrame, such as setting the size or visibility. Constructors are run when the instance is created: One constructor can create a blank JFrame, while another can create it with a default title.

# CHAPTER 3

## DESIGN AND IMPLEMENTATION

Database design is the process of producing a detailed data model of database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. The database implementation or deployment is the process **os** installation of database software, configuration and customization, running, testing, integrating with applications, and training the users.

## 3.1 ER DIAGRAM

An entity -relationship model is a systematic way of describing and defining a business process. An ER model id typically implemented as a database. The main component of E-R model are: entity set and relationship set. The ER diagram is show in **fig 3.1**

## 3.2 RELATION SCHEMA

It formulate all the constraints that are to be applied on the data. A database schema defines its entities and the relationship among them.it contains a descriptive detail of the database. It shows how data will be stored in a secondary storage.The RELATION SCHEMA is show in **fig 3.2**

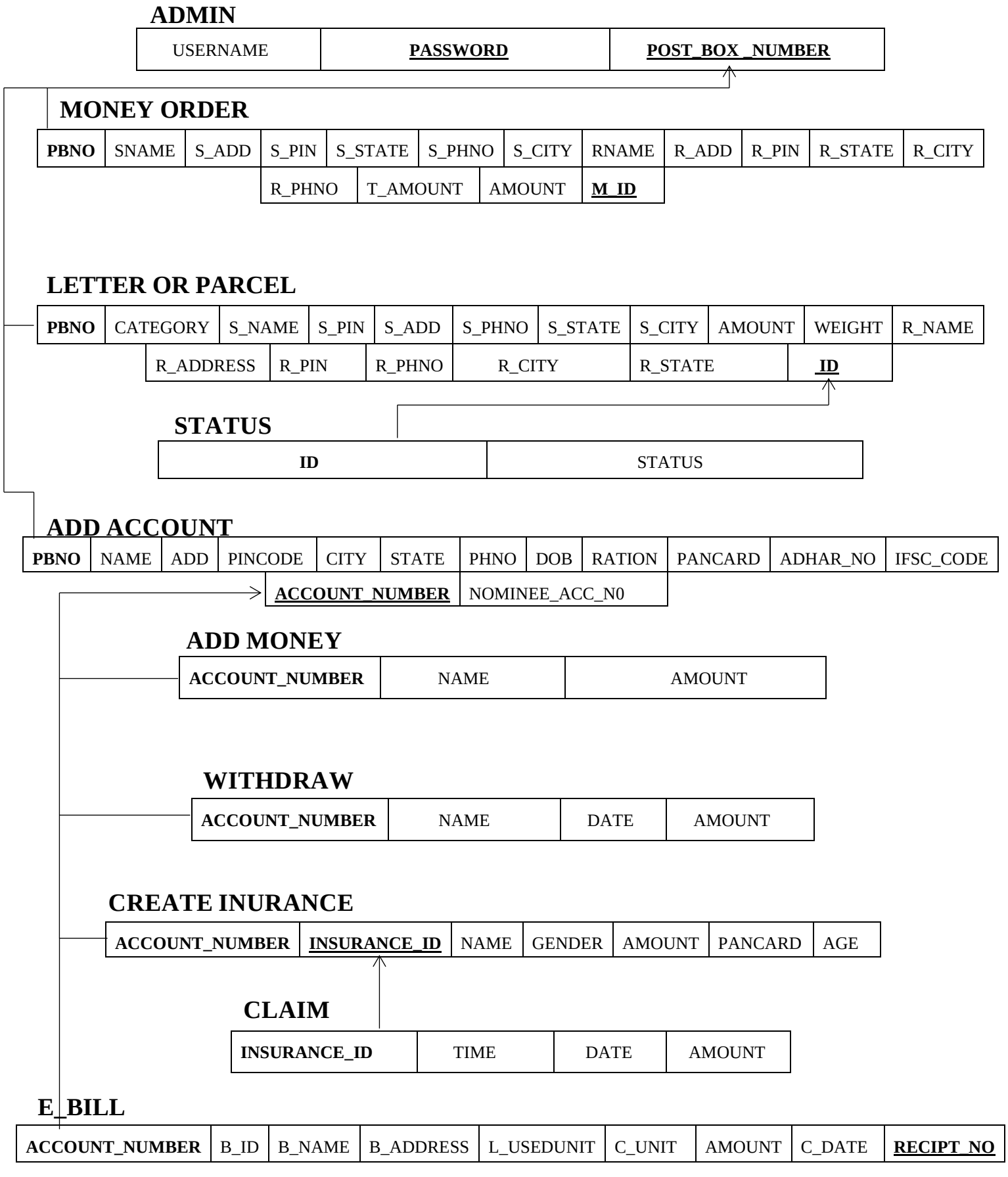

ADMIN
USERNAME
PASSWORD
POST_BOX _NUMBER
MONEY ORDER
PBNO
SNAME
S_ADD
S_PIN
S_STATE
S_PHNO
S_CITY
RNAME
R_ADD
R_PIN
R_STATE
R_CITY
R_PHNO
T_AMOUNT
AMOUNT
M_ID
LETTER OR PARCEL
PBNO
CATEGORY
S_NAME
S_PIN
S_ADD
S_PHNO
S_STATE
S_CITY
AMOUNT
WEIGHT
R_NAME
R_ADDRESS
R_PIN
R_PHNO
R_CITY
R_STATE
ID
STATUS
ID
STATUS
ADD ACCOUNT
PBNO
NAME
ADD
PINCODE
CITY
STATE
PHNO
DOB
RATION
PANCARD
ADHAR_NO
IFSC_CODE
ACCOUNT_NUMBER
NOMINEE_ACC_N0
ADD MONEY
ACCOUNT_NUMBER
NAME
AMOUNT
WITHDRAW
ACCOUNT_NUMBER
NAME
DATE
AMOUNT
CREATE INURANCE
ACCOUNT_NUMBER
INSURANCE_ID
NAME
GENDER
AMOUNT
PANCARD
AGE
CLAIM
INSURANCE_ID
TIME
DATE
AMOUNT
E_BILL
ACCOUNT_NUMBER
B_ID
B_NAME
B_ADDRESS
L_USEDUNIT
C_UNIT
AMOUNT
C_DATE
RECIPT_NO

**fig 3.2**

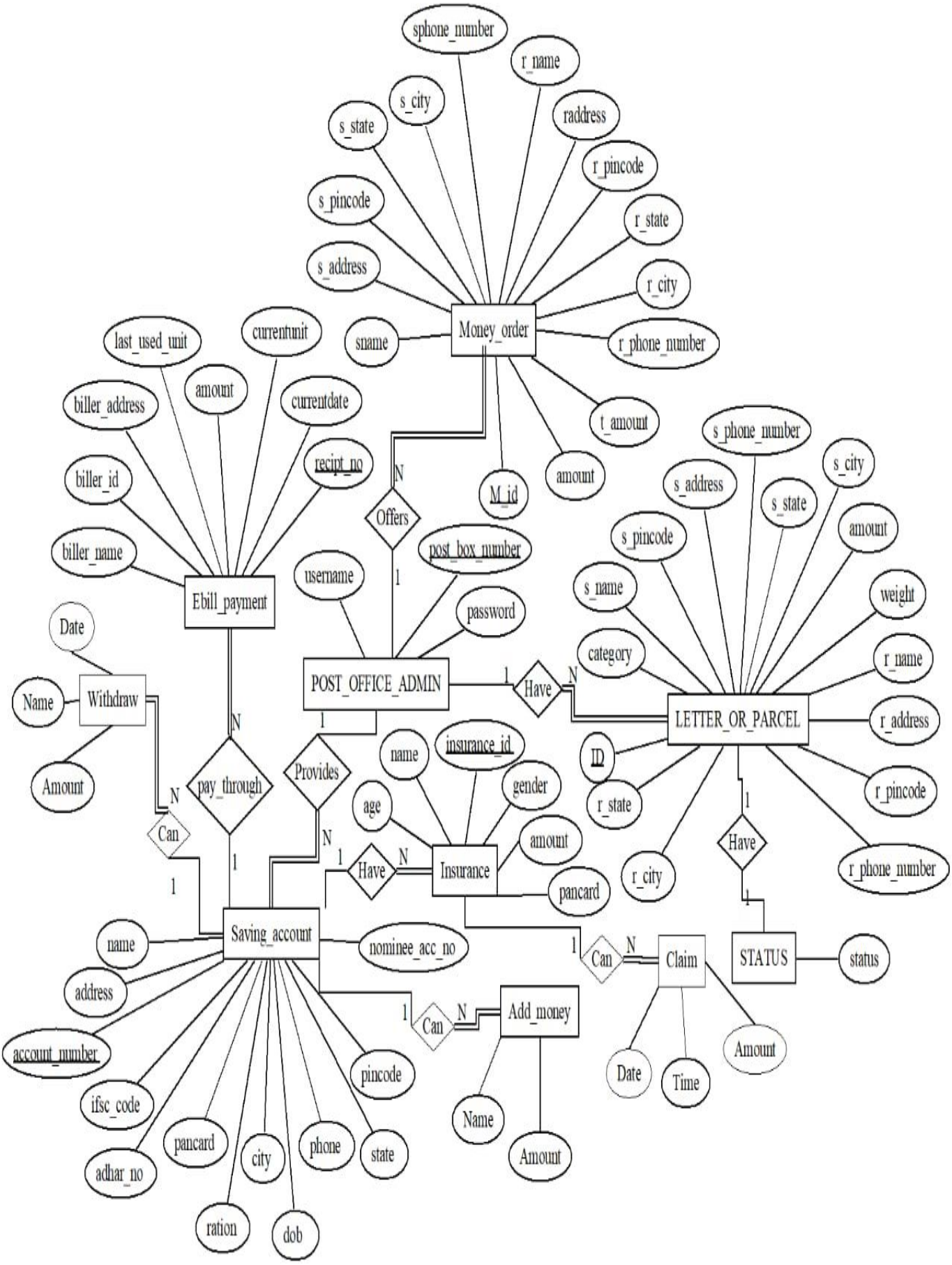

sphone_number
r_name
s_city
raddress
s_state
r_pincode
s_pincode
r_state
s_address
r_city
Money_order
sname
r_phone_number
last_used_unit
currentunit
amount
biller_address
currentdate
t_amount
recipt_no
biller_id
M_id
amount
N
Offers
post_box_number
biller_name
Ebill_payment
username
1
password
Date
POST_OFFICE_ADMIN
1
Have
N
Name
Withdraw
N
pay_through
Provides
1
Amount
N
Can
1
1
N
1
Have
N
Insurance
name
insurance_id
age
gender
amount
pancard
Saving_account
nominee_acc_no
1
Can
N
Claim
STATUS
status
name
address
account_number
1
Can
N
Add_money
pincode
ifsc_code
pancard
city
phone
state
adhar_no
ration
dob
Name
Amount
Date
Time
Amount
s_phone_number
s_city
s_address
s_state
amount
s_pincode
s_name
weight
category
r_name
LETTER_OR_PARCEL
r_address
ID
r_pincode
r_state
1
Have
r_city
r_phone_number
1

**fig 3.1**

## 3.3 Table 1:Relationships in the relational schema

| Entity 1 | Entity 2 | Relationships | Description | Cardinality ratio |
|---|---|---|---|---|
| Post_office | Letter_or_Parcel | Have | 1 postoffice maintain N letter or parcel | 1:N |
| Post_office | Money_order | offers | 1 postoffice offere N money_order | 1:N |
| Post_office | Saving_account | provide | 1 postoffice provide N Saving account | 1:N |
| Letter_or_Parcel | Status | Have | 1 letter or parcel have 1 status | 1:1 |
| Saving_account | Insurance | Have | 1 saving_account have N Insurance | 1:N |
| Saving_account | ebill_payment | Pay_through | 1 saving_account pays N ebill_payment | 1:N |
| Saving_account | Withdraw | Can | 1 saving_account Can have N Withdraw | 1:N |
| Saving_account | Add_money | Can | 1 saving_account Can have N Add_money | 1:N |
| Insurance | Claim | Can | 1 Insurance Can have N Claim | 1:N |

## 3.4 IMPLEMENTATION

**Here we used MariaDB (MySQL) at the front end to store the data and information and Java for designing front end**

## 3.5 TABLE CREATED

create table addaccount(account_number varchar(20) primary key,name varchar(20),address varchar(20),ifsc_code varchar(20),adhar_no varchar(20),pancard varchar(20),ration varchar(20),DOB varchar(20),phone varchar(20),state varchar(20),city varchar(20),pincode varchar(20),nominee_acc_no varchar(30));

create table addmoney(name varchar(20), account_number references addaccount (account_number) on delete set null,amount varchar(20));

create table withdrawmoney(name varchar(20),account_number references on delete set null, date varchar(20),amount varchar(20));

create table createinsurance(account_number varchar(20),insurance_id varchar(20) primary key,name varchar(20),age varchar(20),gender varchar(20), amount varchar(20),pancard varchar(20),foreign key(account_number) references addaccont(account_number) on delete cascade);

create table insuranceclaim(insurance_id references createinsurance(insurance_id) on delete set null,time varchar(10),date varchar(20),amount varchar(20));

create table e_bill(biller_name varchar(20),biller_id varchar(20) ,biller_address varchar(40),last_used_unit varchar(20),currentunit varchar(20),amount varchar(20),current_date varchar(20),recipt_no varchar(20),primary key(recipt_no));

create table moneyorder( mid varchar(20),sname varchar(20),address varchar(20),pincode varchar(20),state varchar(20),city varchar(20),s_phone_number varchar(20),amount int,t_amount int,rname varchar(20),raddress varchar(20),rpincode varchar(20),rstate varchar(20),rcity varchar(20),rphone_number varchar(20));

create table lorp(category varchar(20),id varchar(20) primary key,s_name varchar(20),s_address varchar(20),s_pincode varchar(20),s_state varchar(20),s_city varchar(20),s_phone_number varchar(20),weight varchar(20),amount varchar(20),r_name varchar(20),r_address varchar(20),r_pincode varchar(20),r_state varchar(20),r_city varchar(20),r_phone_number varchar(20));

create table status(id varchar(20),status varchar(20),foreign key(id) references lorp(id) on delete cascade);

## 3.6 CODE FOR TRIGGER

```
delimiter //
create trigger amount
before insert on lorp
for each row
begin
if(new.weight>0&&new.weight<=10) then
set new.amount='10';
else if(new.weight>11&&new.weight<=20) then
set new.amount='15';
else if(new.weight>=21&&new.weight<=30) then
set new.amount='25';
else if(new.weight>=31&&new.weight<=40) then
set new.amount='35';
else if(new.weight>=41&&new.weight<=50) then
set new.amount='45';
else if(new.weight>=51&&new.weight<=60) then
set new.amount='60';
else if(new.weight>=61&&new.weight<=70) then
set new.amount='75';
else set new.amount='100';
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end; //
delimiter ;
```

…….

## 3.7 STORED PROCEDURE CODE

```
DELIMITER$$
CREATE PROCEDURE getdeatils()
BEGIN
SELECT   l.category, s.id, l.s_name,l.s_pincode,l.s_phone_number,l.r_name,l.r_address,
l.r_pincode, s.status   FROM status S,lorp L where S.id=L.id;
END$$
DELIMITER ;
```

## 3.6 JAVA CODE

```
//:Open a connection
Class .forName(“com.mysql.jdbc.Driver”);
Connection  conn = DriverManager.getConnection("jdbc:mysql://localhost:7777/postoffice",
"root", "root");
System.out.println("Connected database successfully...");
```

# CHAPTER 4

## TESTING AND ANALYSIS

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

## 4.1 UNIT TESTING

Unit testing is undertaken when a module has been created and successfully reviewed. In order to test a single module, we need to provide a complete environment besides the module we would require

- The procedures belonging to other modules that the module under test calls. Non-local data structures that module accesses

- A procedure to call the functions of the module under test with appropriate parameters Unit testing was done on each and every module
- Testing admin login form-This form is used for log in of administrator of the system. In this we enter the username and password if both are correct administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask for username and password.
- Admin- Admin can enter the additional soil detail that he encounters with.

## 4.2 INTEGRATION TESTING

In this type of testing we test various integration of the project module by providing the input. The primary objective is to test the module interfaces in order to ensure that no errors are occurring when one module invokes the other module.

We have checked all the modules by giving various type of inputs, when input does not match with stored data, it will be rejected by showing the error.

# CHAPTER 5

## SCREENSHOTS

### DATABASE

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
Enter password: ****
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 485
Server version: 10.4.8-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use LG;
Database changed
MariaDB [LG]> show tables;
+--------------------+
| Tables_in_lg       |
+--------------------+
| addaccount         |
| addaccounts        |
| addmoney           |
| createinsurance    |
| createlifeinsurance |
| e_bill             |
| getdetails         |
| insuranceclaim     |
| lorp               |
| moneyorder         |
| register           |
| status             |
| view               |
| withdrawmoney      |
+--------------------+
14 rows in set (0.001 sec)
```

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
MariaDB [LG]>
MariaDB [LG]> desc addaccounts;
+---------------+------------+------+-----+---------+-------+
| Field         | Type       | Null | Key | Default | Extra |
+---------------+------------+------+-----+---------+-------+
| account_number | varchar(20) | NO   | PRI | NULL    |       |
| name          | varchar(20) | YES  |     | NULL    |       |
| address       | varchar(20) | YES  |     | NULL    |       |
| ifsc_code     | varchar(20) | YES  |     | NULL    |       |
| adhar_no      | varchar(20) | YES  |     | NULL    |       |
| pancard       | varchar(20) | YES  |     | NULL    |       |
| ration        | varchar(20) | YES  |     | NULL    |       |
| dob           | varchar(20) | YES  |     | NULL    |       |
| phone         | varchar(20) | YES  |     | NULL    |       |
| state         | varchar(20) | YES  |     | NULL    |       |
| city          | varchar(20) | YES  |     | NULL    |       |
| pincode       | varchar(20) | YES  |     | NULL    |       |
| nominee_acc_no | varchar(20) | YES  |     | NULL    |       |
+---------------+------------+------+-----+---------+-------+
13 rows in set (0.028 sec)
```

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
MariaDB [LG]>
MariaDB [LG]> desc createlifeinsurance;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| account_number | varchar(20) | NO   | MUL | NULL    |       |
| insurance_id   | varchar(20) | NO   | PRI | NULL    |       |
| name           | varchar(20) | YES  |     | NULL    |       |
| age            | varchar(20) | YES  |     | NULL    |       |
| gender         | varchar(20) | YES  |     | NULL    |       |
| amount         | varchar(20) | YES  |     | NULL    |       |
| pancard        | varchar(20) | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
7 rows in set (0.028 sec)
```

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
MariaDB [LG]>  desc  e_bill;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| account_number | varchar(20) | YES  |     | NULL    |       |
| biller_name    | varchar(20) | YES  |     | NULL    |       |
| biller_id      | varchar(20) | YES  |     | NULL    |       |
| biller_address | varchar(20) | YES  |     | NULL    |       |
| last_used_unit | varchar(20) | YES  |     | NULL    |       |
| currentunit    | varchar(20) | YES  |     | NULL    |       |
| amount         | varchar(20) | YES  |     | NULL    |       |
| currentdate    | varchar(20) | YES  |     | NULL    |       |
| recipt_no      | varchar(20) | NO   | PRI | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
9 rows in set (0.034 sec)
```

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
MariaDB [LG]> desc lorp;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| category      | varchar(20) | YES  |     | NULL    |       |
| id            | varchar(20) | NO   | PRI | NULL    |       |
| s_name        | varchar(20) | YES  |     | NULL    |       |
| s_address     | varchar(20) | YES  |     | NULL    |       |
| s_pincode     | varchar(20) | YES  |     | NULL    |       |
| s_state       | varchar(20) | YES  |     | NULL    |       |
| s_city        | varchar(20) | YES  |     | NULL    |       |
| s_phone_number| varchar(20) | YES  |     | NULL    |       |
| weight        | varchar(20) | YES  |     | NULL    |       |
| amount        | varchar(20) | YES  |     | NULL    |       |
| r_name        | varchar(20) | YES  |     | NULL    |       |
| r_address     | varchar(20) | YES  |     | NULL    |       |
| r_pincode     | varchar(20) | YES  |     | NULL    |       |
| r_state       | varchar(20) | YES  |     | NULL    |       |
| r_city        | varchar(20) | YES  |     | NULL    |       |
| r_phone_number| varchar(20) | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
16 rows in set (0.131 sec)
```

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
MariaDB [LG]>
MariaDB [LG]>
MariaDB [LG]> desc moneyorder;
+--------------+------------+------+-----+---------+-------+
| Field        | Type       | Null | Key | Default | Extra |
+--------------+------------+------+-----+---------+-------+
| sname        | varchar(20) | YES  |     | NULL    |       |
| address      | varchar(20) | YES  |     | NULL    |       |
| pincode      | varchar(20) | YES  |     | NULL    |       |
| state        | varchar(20) | YES  |     | NULL    |       |
| city         | varchar(20) | YES  |     | NULL    |       |
| sphone_number | varchar(20) | YES  |     | NULL    |       |
| amount       | int(11)     | YES  |     | NULL    |       |
| t_amount     | int(11)     | YES  |     | NULL    |       |
| rname        | varchar(20) | YES  |     | NULL    |       |
| raddress     | varchar(20) | YES  |     | NULL    |       |
| rpincode     | varchar(20) | YES  |     | NULL    |       |
| rstate       | varchar(20) | YES  |     | NULL    |       |
| rcity        | varchar(20) | YES  |     | NULL    |       |
| rphone_number | varchar(20) | YES  |     | NULL    |       |
+--------------+------------+------+-----+---------+-------+
14 rows in set (0.032 sec)
```

MySQL Client (MariaDB 10.4 (x64)) - "C:\Program Files\MariaDB 10.4\bin\mysql.exe" "--defaults-file=C:\Program Files\MariaDB 10.4\data\my.ini" -uroot -p

```
MariaDB [LG]>
MariaDB [LG]>
MariaDB [LG]>
MariaDB [LG]>
MariaDB [LG]>  desc  status;
+--------+------------+------+-----+---------+-------+
| Field  | Type       | Null | Key | Default | Extra |
+--------+------------+------+-----+---------+-------+
| id     | varchar(20) | YES  | MUL | NULL    |       |
| status | varchar(20) | NO   |     | NULL    |       |
+--------+------------+------+-----+---------+-------+
2 rows in set (0.030 sec)

MariaDB [LG]>
```

```
MariaDB [LG]>  desc  e_bill;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| account_number  | varchar(20) | YES  |     | NULL    |       |
| biller_name     | varchar(20) | YES  |     | NULL    |       |
| biller_id       | varchar(20) | YES  |     | NULL    |       |
| biller_address  | varchar(20) | YES  |     | NULL    |       |
| last_used_unit  | varchar(20) | YES  |     | NULL    |       |
| currentunit     | varchar(20) | YES  |     | NULL    |       |
| amount          | varchar(20) | YES  |     | NULL    |       |
| currentdate     | varchar(20) | YES  |     | NULL    |       |
| recipt_no       | varchar(20) | NO   | PRI | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
9 rows in set (0.034 sec)
```

**Fig 5.1:DATABASE**

# LOGIN PAGE FOR ADMIN


**Fig 5.2:LOGIN PAGE**

# FRONTPAGE


**Fig 5.3:FRONT PAGE**

# MENU

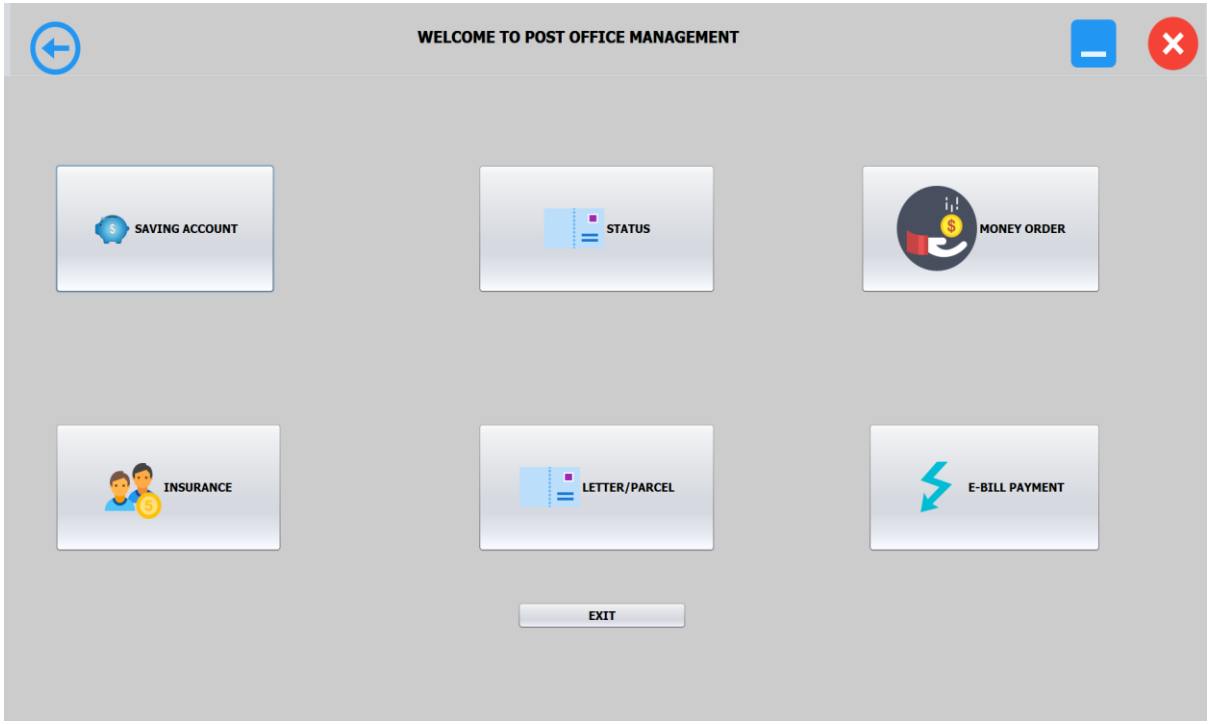



**Fig 5.4 MENU PAGE**

# CREATE ACCOUNT

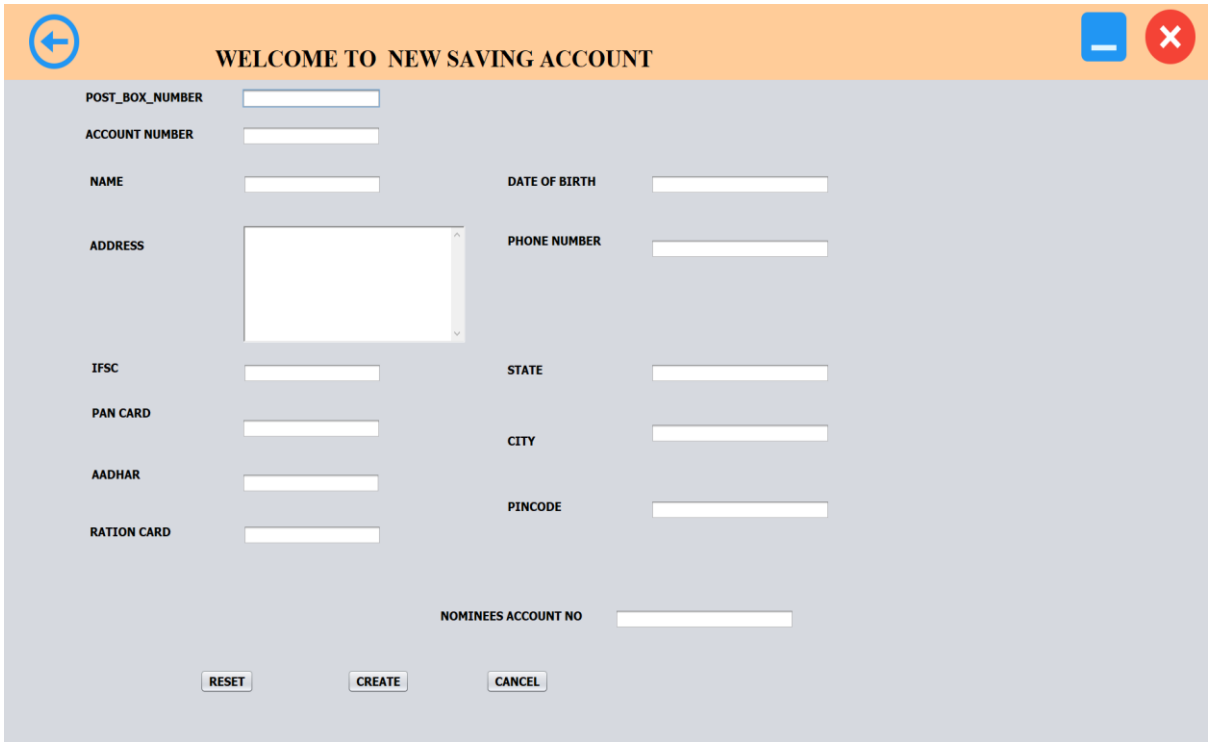



**Fig 5.5:CREATE ACCOUNT**

# MONEY ORDER


**Fig 5.6: MONEY ORDER**

# LIFE INSURANCE

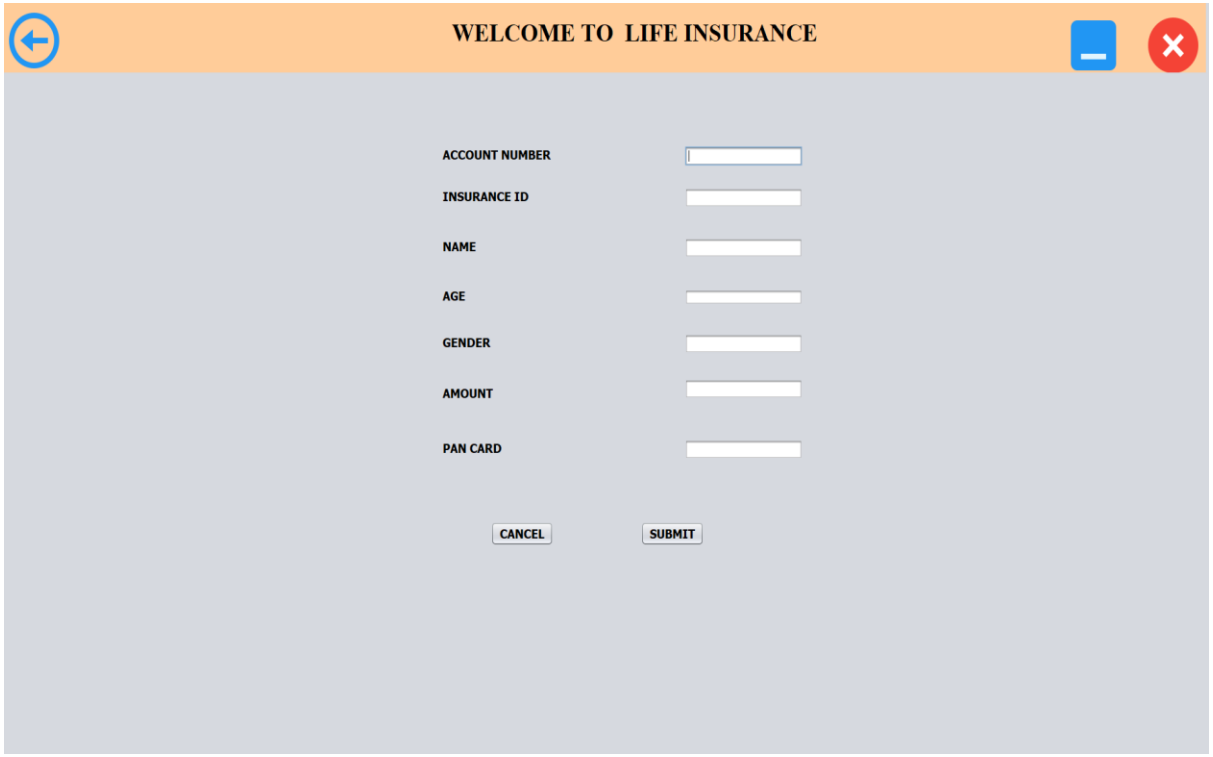



**Fig 5.7: LIFE INSURANCE**

# E_BILL


**Fig 5.8: E_BILL**

# LETTER OR PARCEL

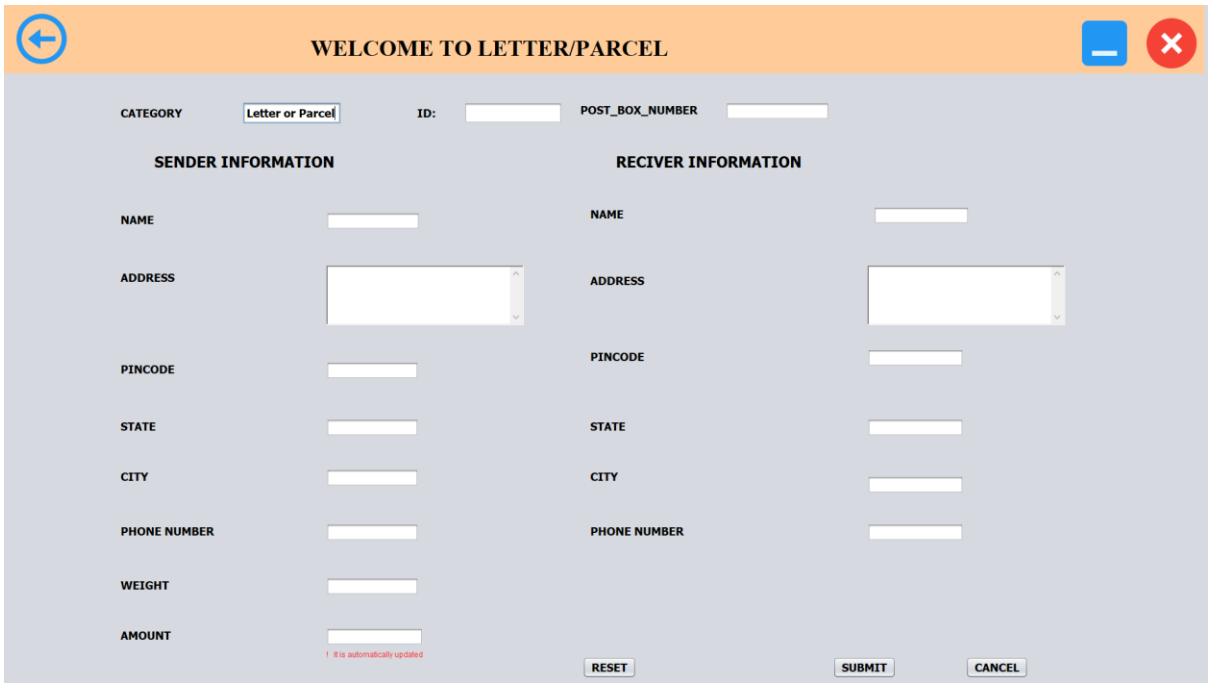



**Fig 5.9 LETTER OR PARCEL**

# VIEW OF TABLES

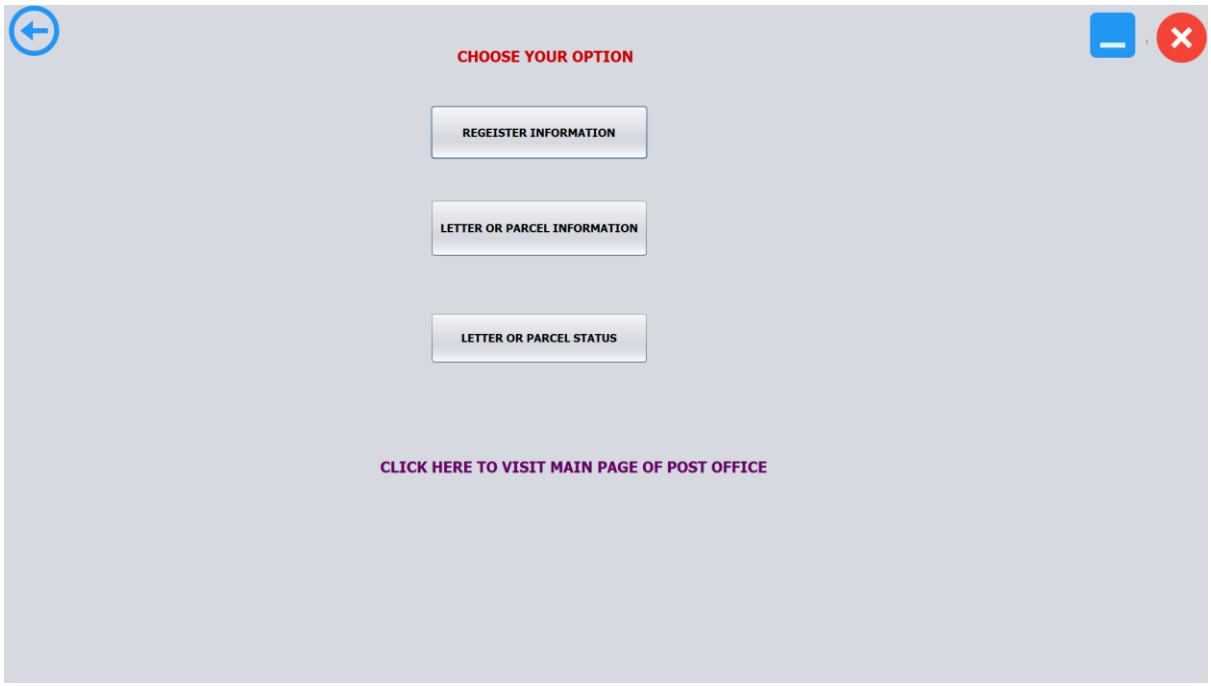
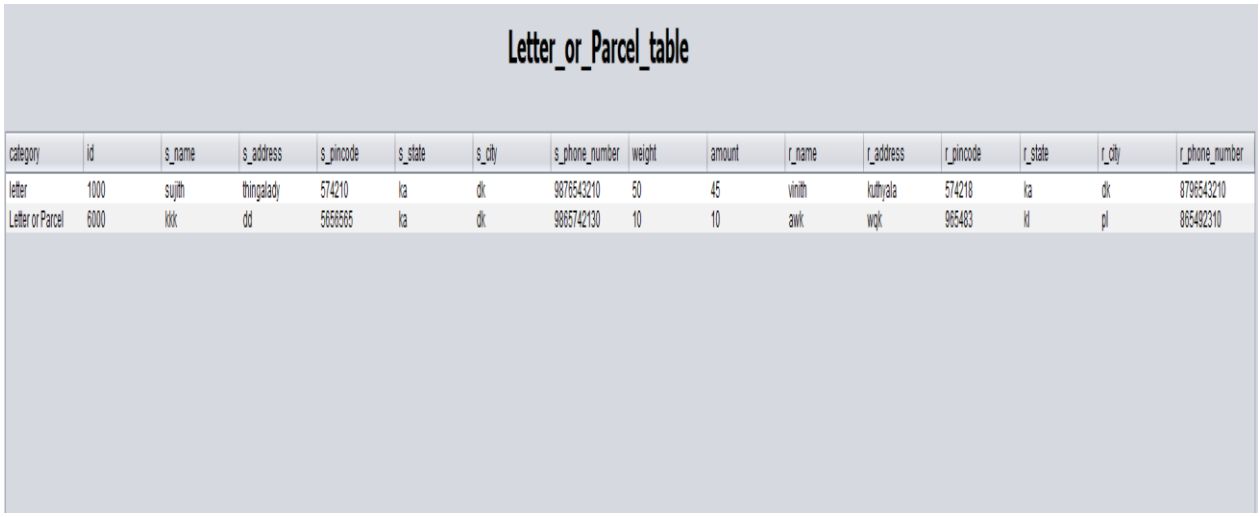



**Fig 5.10 VIEW**

## VIEW LETTER/PARCEL



Letter_or_Parcel_table

| category | id | s_name | s_address | s_pincode | s_state | s_city | s_phone_number | weight | amount | r_name | r_address | r_pincode | r_state | r_city | r_phone_number |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| letter | 1000 | sujith | thingalady | 574210 | ka | dk | 9876543210 | 50 | 45 | vinith | kuthyala | 574218 | ka | dk | 8796543210 |
| Letter or Parcel | 6000 | kkk | dd | 5656565 | ka | dk | 9865742130 | 10 | 10 | awk | wqk | 96483 | kl | pl | 86548230 |

**Fig 5.11 VIEW LETTER/ PARCEL**

## REGISTER TABLE

Register_table

| FirstName | LastName | username | password | Retype_Password | BirthDate | Address |
|---|---|---|---|---|---|---|
| sujith | rai | 4kv17cs062 | 8197740871 | 8197740871 | 21/01/1999 | puttur |
| vineeth | kuthyala | 4kv17cs064 | 9448851464 | 9448851464 | 30/12/1998 | chrikallu |
| kvg | kvg | kvg | cse | cse | 12/03/1986 | sullia |

**Fig 5.12 REGISTER TABLE**

## STATUS OF LETTER/PARCEL

Status of Letter & Parcel

| category | id | s_name | s_pincode | s_phone_number | r_name | r_address | r_pincode | status |
|---|---|---|---|---|---|---|---|---|
| letter | 1000 | sujith | 574210 | 9876543210 | vinith | kuthyala | 574216 | delivered |
| Letter or Parcel | 6000 | kkk | 5656565 | 9865742130 | awk | wqk | 965483 | not_delivered |

**Fig 5.13 DELIVERY STATUS(letter/parcel)**

# CHAPTER 6

## CONCLUSION

The system was mainly designed to reduce the manual work of updating and also make it easier for the employees. All the data's of the accounts, money order, insurance details of letter or parcels are stored more efficiently and it is retrieved whenever it is required from database.

In this project we are mainly targeted on post office for the storing the data and we are given all the basic operations that that performs in the post office. Futher requirements and improvement can be easily done by designing codes. Improvement can be appended by changing the existing modules.

# REFERENCES

[1] Remez Elmasri, Kant B. Navathe, The fundamentals of database systems of 7th edition , 2017, Pearson

[2] Ramakrishnan and Gehrke, Database management system, 3rd edition, 2014, McGraw Hill

[3] Herbert Schildt:Java The complete references,7th/9th edition ,Tata McGraw Hill, 2007.

[4] Jim Keogh:J2EE The complete reference,Mcgraw Hill,2007.