A PRELIMINARY REPORT ON

“Banking Application”

SUBMITTED TO THE EDUBRIDGE INDIA PRIVATE LIMITED

SUBMITTED BY

K.SHALINI **BATCH NO: EON-5755**

Under The Guidance Of

Amruta Deore

ACKNOWLEDGEMENT

It gives all of us great pleasure in presenting the preliminary project report on **“Banking Application”.** With due respect and gratitude we would like to take this opportunity to thank internal guide of project **Mrs.Amruta Deore** for giving us all help and guidance we needed. We are really grateful for her kind support. She always encouraged us and given us the motivation to move ahead.
She has put in a lot of time and effort in this project along with us and given us a lot of confidence. Also we wish to thank all the other people who had helped us in the successful completion of this project.

K.SHALINI

ABSTRACT

The main aim of Banking System Project is to maintain database of customers. In this project developed in Java programming language using mysql database.
This project will help to insert the every details of the customers that will stored in database. This project will help to update records of customers If customers made any change in their records. This project helps banking system to find the particular record of the customer and also they can see their balance of customers.

Chapter 1

INTRODUCTION

1.1 INTRODUCTION

The objective of Banking system is to allow the administrator of any organization to edit and find out the personal details of a customer and allows the customer to keep up to date his/her profile. It will also facilitate keeping all the records of the customers such as their name, account number, branch, phone no and balance . So all the information about an customer will be available within a few seconds. Overall it will make the banking application an easier job for the administrator and the customers. The main purpose of banking application is to illustrate the requirements of the customer to help for the bank to maintain and manage the customers's personal data.

1.2 SCOPE

. Without a banking application, managing and maintaining the details of the customer is a difficult job for banks.Banking application System will store all the details of the customer such ,name, account no, branch,phone no, balance of the cutomers.

1.3 SYSTEM REQUIREMENTS:

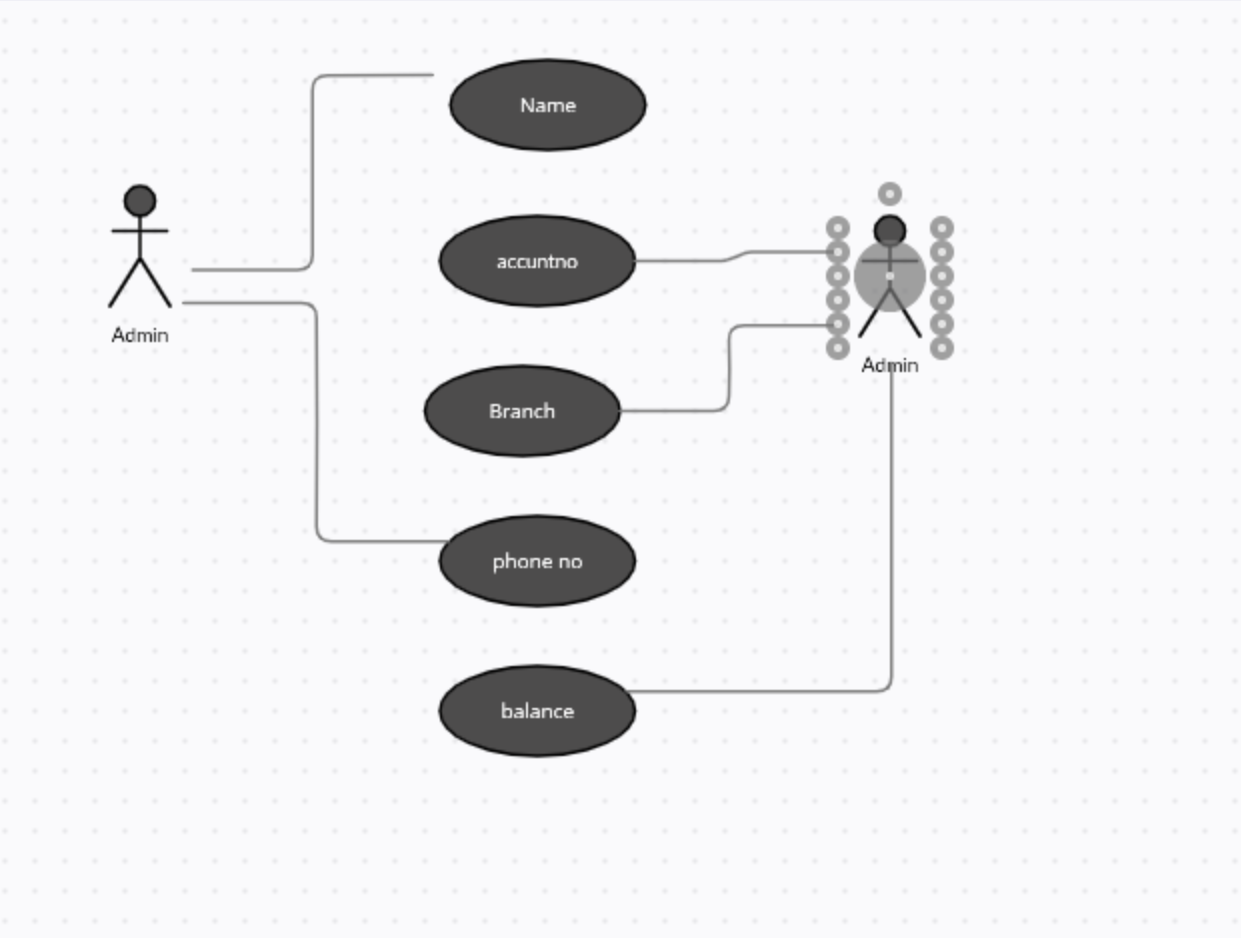
1.3.1 Software Requirements:

1. Operating System - Windows 10.
2. Platform - Eclipse, mysqlyog.
3. Language - Core Java, My SQL.
4. Files - Chrome.

1.4 UML DIAGRAMS

1.4.1 Use Case Diagram

The following UML use case diagram shows the working of the Banking Application System. Moreover it has five cases that show the particular functionality of the student.The seven cases are name, account no, branch,phnone no and the balance for each customer. The user can view the details of them by entering the account number and can view the name,branch,phone no and the balance for each customer. The administrator can check the name,branch,phone no and the balance for each customer whether the customer entered the information is correct or not. The interactions of the customer and the administrator are what the Banking application use case diagram example.



1.5 MODULE

1.5.1 Insert student record:

The insert record module is used to inserting the record of the customer.Like name, branch,phone number of the that will be stored in database.

Program:

```
private void insertRecord() throws SQLException
{
        String sql="insert into bank (Name,branch,phnone_no,balance) values
(?,?,?,?)";
        PreparedStatement
preparedStatement=connection.prepareStatement(sql);
        System.out.println("Enter name");
        scanner.nextLine();
        preparedStatement.setString(1,scanner.nextLine());

        System.out.println("Enter branch");
        preparedStatement.setString(2,scanner.nextLine());

        System.out.println("Enter phoneno");
        preparedStatement.setInt(3,scanner.nextInt());

        System.out.println("Enter  balance");
        preparedStatement.setInt(4,scanner.nextInt());

int rows=preparedStatement.executeUpdate();


if(rows>0 )
{
        System.out.println("Record inserted successfully");
}


}
```

1.5.2 SelectStudentRecord:

The select record method is used to select the information of a particular customer by entering account numberBy entering the roll number the customer can view their details of name, account number,branch,phone number, balance.

Program:

```
private void selectRecord() throws SQLException
        {
                System.out.println("Enter account no to find result");

               int number= scanner.nextInt();
               String sql="select * from bank where account_no=  " + number;
      Statement statement = connection  .createStatement();


      ResultSet result= statement.executeQuery(sql);
      if(result.next())
      {

        String name=result.getString("NAME");
        int accountNumber=result.getInt("account_no");
        String branch=result.getString("branch");
        int phoneNo=result.getInt("phnone_no");
        int balance=result.getInt("balance");

        System.out.println("Name : " + name);
        System.out.println("Account number : "+ accountNumber);
        System.out.println("Branch : " + branch);
        System.out.println("Phone number : " + phoneNo);
        System.out.println("Balance : " + balance);


      }else {


        System.out.println("***************NO record found***************");
      }

       }
```

1.5.3 Update Student Record:

The Update Record method is used for updating the name, branch, phone number of customer if they want to update their details

Program: (for update record)

```
private void updateRecord() throws SQLException
{


        System.out.println("Enter account no to find result");

       int number= scanner.nextInt();
       String sql="select * from bank where account_no=  " + number;
  Statement statement = connection  .createStatement();


  ResultSet result= statement.executeQuery(sql);
  if(result.next())
  {

        String name=result.getString("NAME");
        int accountNumber=result.getInt("account_no");
        String branch=result.getString("branch");
        int phoneNo=result.getInt("phnone_no");
        int balance=result.getInt("balance");

        System.out.println("Name is" + name);
        System.out.println("Account number is"+ accountNumber);
        System.out.println("Branch is" + branch);
        System.out.println("Phone number is" + phoneNo);
        System.out.println("Balance is" + balance);

        System.out.println(" What do you want upadte");
        System.out.println(" 1.Name");
        System.out.println(" 2.Branch");
        System.out.println(" 3.Phone no");
         int choice=scanner.nextInt();
         String sqlQuery="update bank set";
  switch(choice)
  {
  case 1:
        System.out.println("Enter new name");
        scanner.nextLine();
        String newName=scanner.nextLine();
        sqlQuery = sqlQuery + " Name = ? where account_no= "+accountNumber;
     PreparedStatement  preaparedStatement=         connection.prepareStatement(sqlQuery);
     preaparedStatement.setString(1, newName);
      int rows=preaparedStatement.executeUpdate();
      if(rows > 0)
```

```
    {
       System.out.println("record updated successfully");
    }

      break;

case 2:
      System.out.println("Enter new branch");
      scanner.nextLine();
      String newBranch=scanner.nextLine();
      sqlQuery = sqlQuery + " branch = ? where account_no= "+accountNumber;
   PreparedStatement  preaparedStatement1=       connection.prepareStatement(sqlQuery);
   preaparedStatement1.setString(1, newBranch);
    int rows1=preaparedStatement1.executeUpdate();
    if(rows1 > 0)
    {
       System.out.println("record updated successfully");
    }



      break;

case 3:

      System.out.println("Enter new Phone no");
      //scanner.nextInt();
      int newPhoneno=scanner.nextInt();
      sqlQuery = sqlQuery + " phnone_no = ? where account_no= "+accountNumber;
   PreparedStatement  preaparedStatement2=       connection.prepareStatement(sqlQuery);
   preaparedStatement2.setInt(1, newPhoneno);
    int rows2=preaparedStatement2.executeUpdate();
    if(rows2 > 0)
    {
       System.out.println("record updated successfully");
    }




       break;

default:
              break;
 }

}else {


      System.out.println("***************NO record found***************");
}
```

1.5.4 Delete Student Record:

The Delete Record method is used to delete the details of the customer record. By entering the account number we can delete the particular record of the customer if needed.

Program:

```
void deleteRecord() throws SQLException

{
        System.out.println("Enter account number to delete");
        int accountNumber=scanner.nextInt();
        String sql="delete from bank where account_no =  " + accountNumber;
        PreparedStatement ps= connection.prepareStatement(sql);
        int rows= ps.executeUpdate();

        if(rows>0)
        {
                System.out.println("********Record  deleted successfully********");
        }
```

1.5.5 DEPOSIT AMOUNT

The Deposit Record method is used to deposit the amount in the customers account. By using the account number to deposit money in the particular account. It will show the current balance after deposit it will show available balance.

program

```
private void depositAmount() throws SQLException
{
        System.out.println("Enter account number to be deposit");

       int anumber= scanner.nextInt();
       String sql="select balance from bank where account_no=  " + anumber;
   Statement statement = connection  .createStatement();


   ResultSet result = statement.executeQuery(sql);
   //hint accountNumber=result.getInt("account_no");
   int deposit;
   if(result.next())
   {

        int balance1=result.getInt("balance");


        System.out.println("Current balance is "+ balance1);



       System.out.println("enter  depoisit amount");
        deposit=scanner.nextInt();
        balance1  =balance1+deposit;
        System.out.println("Available Balance " + balance1);
       String sql1= "update bank set balance = ? where account_no =  "+anumber;
       PreparedStatement ps=connection.prepareStatement(sql1);
       ps.setInt(1, balance1);
       ps.executeUpdate();
       System.out.println("*****Amount deposited  successfully*********");
       }}
```

1.5.6 Withdraw Amount

The withdraw amount method is used to withdraw amount from customer account. By using the account number we can deposit amount in the particular account.It shows current balance after withdrawl it shows available balance of the customer

Program

```
private void withdrawAmount() throws SQLException
{
         System.out.println("Enter account number to widthdraw");

        int anumber= scanner.nextInt();
        String sql="select * from bank where account_no=  " + anumber;
    Statement statement = connection  .createStatement();


    ResultSet result = statement.executeQuery(sql);
    //hint accountNumber=result.getInt("account_no");
    //hint withdraw;
    if(result.next())
    {

         int balance1=result.getInt("balance");

         System.out.println("Current balance is "+ result.getInt(5));

        System.out.println("enter  widthdraw  amount");
   int  withdraw=scanner.nextInt();
         balance1  =balance1-withdraw;
         System.out.println("Available Balance " + balance1);
        // sqlQuery = statement.executeUpdate("update bank set balance = 100 where
account_no =1 ",+ balance1);
         String sql1= "update bank set balance = ? where account_no =  "+anumber;
        PreparedStatement ps=connection.prepareStatement(sql1);
        ps.setInt(1, balance1);
        ps.executeUpdate();
        System.out.println("*****Amount withdraw  successfully********");


    }
```

Chapter2

PROJECT IMPLEMENTATION

2.1 SCREENS

2.1.1 Insert Student Record Page:

```
Enter choice
1.Insert record
2.Select record
3.Update record
4.Delete record
5.Deposit amount
6.Widthdraw amount
7.Exit
1
Enter name
dev
Enter branch
chennai
Enter phoneno
76890543
Enter  balance
400
Record inserted successfully
```

2.1.2 Select Student Record:

```
Enter choice
1.Insert record
2.Select record
3.Update record
4.Delete record
5.Deposit amount
6.Widthdraw amount
7.Exit
2
Enter account no to find result
3
Name : Shalini
Account number : 3
Branch : Chennai
Phone number : 1234789052
Balance : 1000
Enter choice
```

2.1.3 Update Student Record:

2.1.3.1 Name

```
Enter choice
1.Insert record
2.Select record
3.Update record
4.Delete record
5.Deposit amount
6.Widthdraw amount
7.Exit
3
Enter account no to find result
2
Name isdivya
Account number is2
Branch isdelhi
Phone number is98734733
Balance is1500
 What do you want upadte
 1.Name
 2.Branch
 3.Phone no
1
Enter new name
arun
record updated successfully
Enter choice
```

2.1.4 Delete Student Record:

```
Enter choice
1.Insert record
2.Select record
3.Update record
4.Delete record
5.Deposit amount
6.Widthdraw amount
7.Exit
4
Enter account number to delete
3
********Record  deleted successfully********
```

2.1.5 Deposit Account:

```
Enter choice
1.Insert record
2.Select record
3.Update record
4.Delete record
5.Deposit amount
6.Widthdraw amount
7.Exit
5
Enter account number to be deposit
5
Current balance is 1000
enter  depoisit amount
2000
Available Balance 3000
*****Amount deposited  successfully********
```

2.1.6 Withdraw account:

```
Enter choice
1.Insert record
2.Select record
3.Update record
4.Delete record
5.Deposit amount
6.Widthdraw amount
7.Exit
6
Enter account number to widthdraw
2
Current balance is 1500
enter  widthdraw  amount
200
Available Balance 1300
*****Amount withdraw  successfully********
Enter choice
```

Chapter 3

CONCLUSIONS

Banking application can be used by the administrators to maintain the customers records easily.The result of this project will be accurate and totally error free. The growing use of internet and computers confirms the good scopes of project.