VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELGAUM-590018, KARNATAKA

Project report on

ATM DATABASE SYSTEM

Submitted by

ARPITA HEGDE	4BD17CS018
ARPITA T M	4BD17CS019

Under the Guidance of

Prof. Abdul Razak M S

Prof. Shravani

Assistant Professor

Department of Computer Science and Engineering

BAPUJI INSTITUTE OF ENGINEERING AND TECHNOLOGY

Department of Computer Science and Engineering

Davanagere 577005, Karnataka

ATM Database System

Objective:

This system allows users to make money transactions from one account to another. The users have to enter their account number and then they can access the main menu from where they can perform the action they desire.

Various functionalities including sending and receiving money, changing password, inquire about account balance. It is a flexible application and user-friendly. The front end has been designed using Java swing and database connectivity connects it to a database in MS Access. Due to the various advantages of Java like portability, security, robustness, this system has been designed in Java. This allows users to perform various ATM transactions online and even change their passwords in case of any emergencies. The various controls have been placed on different panels thus making it easy for user to make the best use of the software.

Existing System:

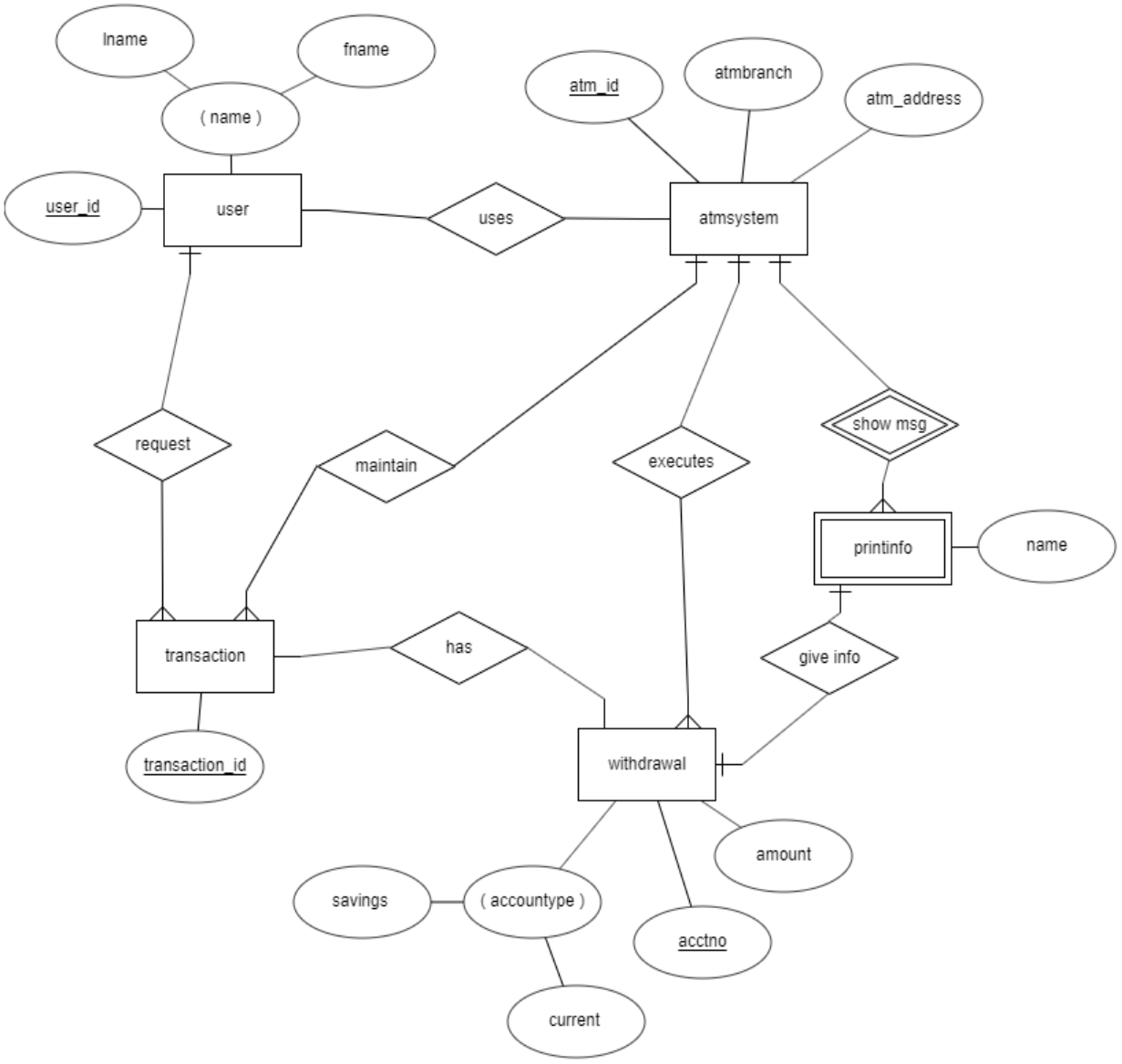
Manually performing transactions and other ATM related actions in the banks is not always possible. There may be cases of emergencies or public holidays which makes it impossible to gain access to these facilities at every time. Also it is time taking and exhaustive in case there is a long queue of customers at the bank or even at the ATM kiosk that are generally provided by banks at various places. Sometimes these kiosks are also out of order making it difficult for the customers to get their job done and also these cause waste of time.

Proposed System:

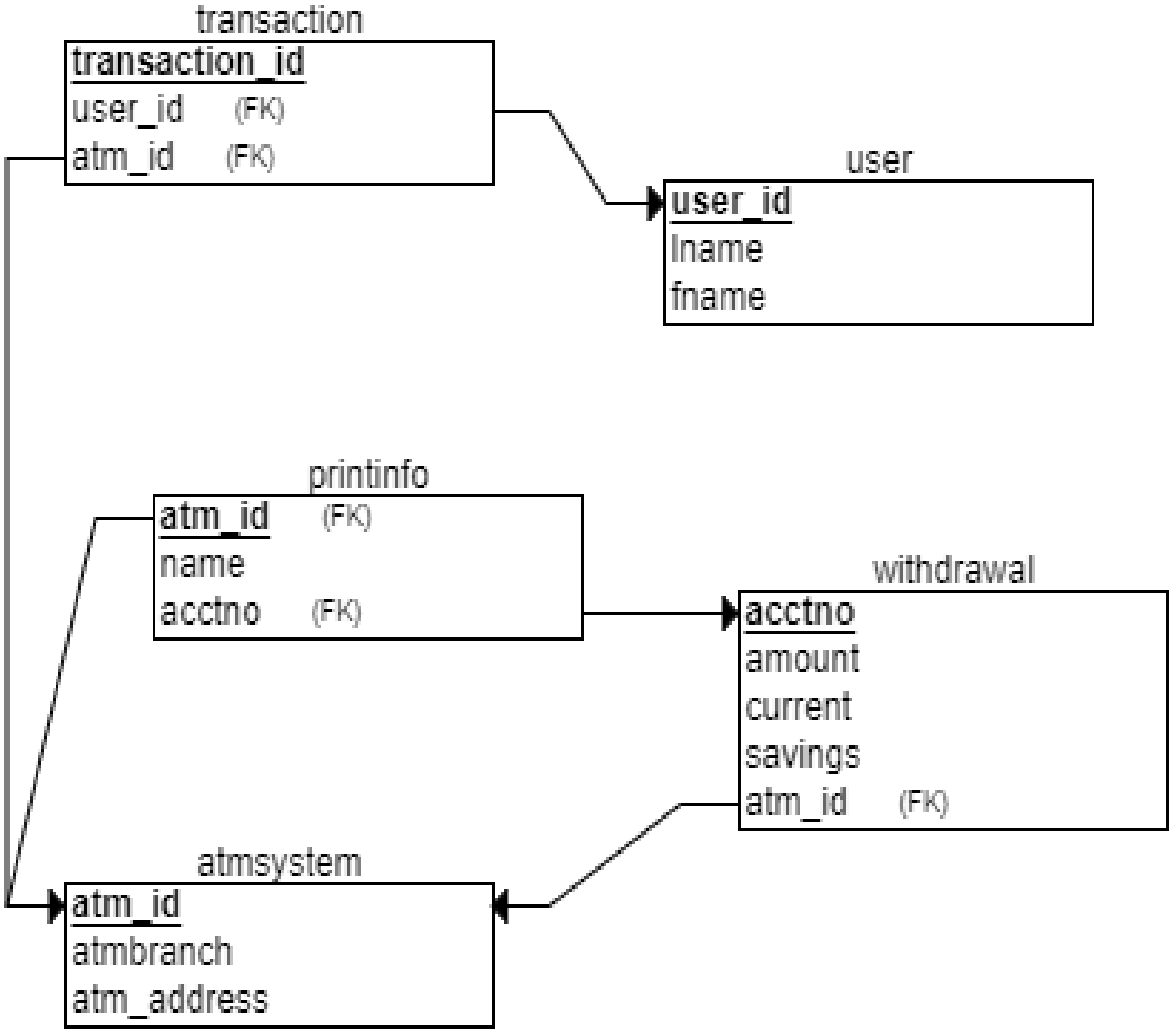
ATM database system provides user the option to access information about their accounts and make cash transactions and even change their passwords online. One does not have to be physically present at the bank or the kiosk to get their jobs done. This efficient and secure software lets users to sit in the comfort of their homes and access the various functionalities of the system. One can also check the balance in their account through this system and also make self transactions.

DESIGN

E-R DIADRAM



SCHEMA DIAGRAM



SEVEN STEPS FOR ER TO SCHEMA CONVERSION

Step 1: Mapping of Regular Entity Types

For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Includes only the simple component attributes of a composite attribute. Choose one of the key attributes of E as the primary key for R. If the chosen key of E is a composite, then the set of simple attributes that form it will together form the primary key of R. If multiple keys were identified for E during the conceptual design, the information describing the attributes that form each additional key is kept in order to specify secondary (unique)keys of relation R. knowledge about keys is also kept for indexing purposes and other types of analyses.

Step 2: Mapping of Weak Entity Types.

For each weak entity type W in the ER schema with owner entity type E, create a relation R and include all simple attributes (or simple components of composite attributes) of W as attributes of R. In addition, include as foreign key attributes of R, the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s); this takes care of mapping the identifying relationship type of W. The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any. If there is a weak entity type E2 whose owner is also a weak entity type E1, then E1 should be mapped before E2 to determine its primary key first.

Mapping of Binary 1:1 Relational Types.

1. The foreign key approach.
2. The merged relationship approach, and
3. The cross-reference or relationship relation approach.

The first approach is useful and should be followed unless special conditions exist, as

1. **Foreign key approach:**

Choose one of the relations S, say and include as a foreign key in S the primary key of T. It is better to choose an entity with total participation in R in the role of S. Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S.

2. **Merged relation approach:**

An alternative mapping of a 1:1 relationship type is to merge the two entity types and the relationship into a single relation. This is possible when both participations are total, as this would indicate that the two tables will have the exact same number of tuples at all times.

3. **Cross-reference or relationship relation approach:**

The third option is to set up a third relation R for the purpose of cross referencing the primary keys of the two relations S and T representing the entity types. As we will see, this approach is required for M:N relationships. The relation R is called a relationship relation (or sometimes a lookup table), because each tuple in R represents a relationship instance that relates one tuple from S with one tuple from T. The relation R will include the primary key attributes od S and T as foreign keys to S and T. The primary key of R will be one of the two foreign keys, and the other foreign key will be a unique key of R. The drawback is having an extra relation, and requiring an extra join operation when combining related tuples from the tables

Step 4: Mapping of Binary 1:N Relationship Type.

For each regular binary 1:N relationship type R, identify the relation S that represents the participating entity types at the N-side of the relationship type. Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R; we do this because each entity instance on the N-side is related to at most one entity instance on the 1 side of the relationship type. Include any simple attributes (or simple components of composite attributes) of the 1:N relationship type as attribute of S.

Step 5: Mapping of Binary M:N Relationship Types.

For each binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attributes in S the primary key of the relations that represent the participating entity types; their combination will form the primary key of S. Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S. Notice that we cannot represent an M:N relationship type by single foreign key attribute in one of the participating relations (as we did for1:1 or 1:N relationship types) because of the M:N cardinality ratio; we must create a separate relationship relation S.

Step 6: Mapping of Multivalued Attributes.

For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K—as a foreign key in R—of the relation that represents the entity type or relationship type that has A as a multivalued attribute. The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

Step 7: Mapping of N-array Relationship Types.

For each n-array relationship type R, where n>2, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. Also include any attributes of the n-array relationship type (or simple components of composite attributes) as attributes of S. The primary key of S is usually a combination of all the foreign keys that references the relations representing the participating entity types. However, if the cardinality constraints on any of the entity types E participating in R is 1, then the primary key of S should not include the foreign key attribute.

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements

- Processor : dual core or any heigher versions
- Disc Space : 2GB
- Ram : 1GB

Software Requirements

- Front end development Tool : Net beans version-8.2(IDE)
- Database : My SQL Essential
- Other : My SQL connector.jar

IMPLEMENTATION

LOGIN CODE

```
try{
    String uname=jTextField1.getText();
    char [] pwd=jPasswordField1.getPassword()   ;
    String password=new String(pwd);
    if(uname.equals("BIET")&&password.equals("CSE"))
    {
        this.setVisible(false);
         new user1().setVisible(true);
           }
    else
        JOptionPane.showMessageDialog(null, "invalid user");
}
catch(Exception e)
{
    System.out.println(e.getMessage());
}
```

USER CODE (INSERT)

```
try{
 String sql="INSERT INTO user1 “+"(userid,fname.lname)”+"VALUES(?,?,?)";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE0
18");
   pst=connect.prepareStatement(sql);
   pst.setString(1,jTextField1.getText());
    pst.setString(2,jTextField2.getText());
```

```
    pst.setString(3,jTextField3.getText());
    pst.executeUpdate();
    JOptionPane.showMessageDialog(null, "Success!");
    dispose();
    new atmsystem1().setVisible(true);
}
catch(HeadlessException | SQLException ex)
{
   JOptionPane.showMessageDialog(null, ex);
}
```

USER CODE (DELETE)

```
        try{
 String sql="delete from user1 where userid =?";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE
018");
   pst=connect.prepareStatement(sql);
   pst.setString(1,jTextField1.getText());
  pst.executeUpdate();
   JOptionPane.showMessageDialog(null, " deleted Successfully!");
  }
catch(HeadlessException | SQLException ex)
{
JOptionPane.showMessageDialog(null, ex);
}
```

ATMSYSTEM CODE

```
try{
String sql="INSERT INTO atmsystem1”+"(atmid,atmbranch,atmadddress)”
+"VALUES(?,?,?)";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE0
18");
    pst=connect.prepareStatement(sql);
    pst.setString(1,jTextField1.getText());
     pst.setString(2,jTextField2.getText());
      pst.setString(3,jTextField3.getText());
      pst.executeUpdate();
    JOptionPane.showMessageDialog(null, "Success!");
   }
catch(HeadlessException | SQLException ex)
{
    JOptionPane.showMessageDialog(null, ex);
}
     try{
    String atmid=jTextField1.getText();
    String atmbranch=jTextField2.getText();
    String atmaddress=jTextField3.getText();
    if(atmid.isEmpty()||atmbranch.isEmpty()||atmaddress.isEmpty())
    {
        JOptionPane.showMessageDialog(null,"please enter the information");


    }else
    {
```

```
    this.setVisible(false);
   new Transaction1().setVisible(true);     }
     }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
```

TRANSCATION CODE

```
try{
 String sql="INSERT INTO Transaction1”
+"(transactionid,userid,atmid)”+"VALUES(?,?,?)";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE
018");
 pst=connect.prepareStatement(sql);
 pst.setString(1,jTextField1.getText());
 pst.setString(2,jTextField2.getText());
 pst.setString(3,jTextField3.getText());
  pst.executeUpdate();
  JOptionPane.showMessageDialog(null, "Success!");
  dispose();
  new choose().setVisible(true);
  }
catch(HeadlessException | SQLException ex)
{
    JOptionPane.showMessageDialog(null, ex);
```

```
} }
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        try{
 String sql="delete from Transaction1 where transactionid =?";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE
018");
   pst=connect.prepareStatement(sql);
   pst.setString(1,jTextField1.getText());
  pst.executeUpdate();
   JOptionPane.showMessageDialog(null, " deleted Successfully!");
}
catch(HeadlessException | SQLException ex){
   JOptionPane.showMessageDialog(null, ex);
}
```

WITHDRAWAL CODE

```
try{
 String sql="INSERT INTO withdraw" +"(accntno,amount,atmid)"
+"VALUES(?,?,?)";

connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE0
18");
    pst=connect.prepareStatement(sql);
    pst.setString(1,jTextField1.getText());
     pst.setString(2,jTextField2.getText());
      pst.setString(3,jTextField3.getText());
     pst.executeUpdate();
```

```
    JOptionPane.showMessageDialog(null, "Success!");
    dispose();
    this.setVisible(false);
    new message().setVisible(true);
  }
catch(HeadlessException | SQLException ex){
    JOptionPane.showMessageDialog(null, ex);
}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
            try{
 String sql="delete from withdraw where accntno =?";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE0
18");
    pst=connect.prepareStatement(sql);
    pst.setString(1,jTextField1.getText());
   pst.executeUpdate();
    JOptionPane.showMessageDialog(null, " deleted Successfully!");


}
catch(HeadlessException | SQLException ex){
    JOptionPane.showMessageDialog(null, ex);
}
```

PRINT INFO

```
try{
  String sql="INSERT INTO printinfo" +"(name,atmid,accntno)”
+"VALUES(?,?,?)";
connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE0
18");
    pst=connect.prepareStatement(sql);
    pst.setString(1,jTextField1.getText());
     pst.setString(2,jTextField2.getText());
      pst.setString(3,jTextField3.getText());
    pst.executeUpdate();
    JOptionPane.showMessageDialog(null, "Success!");
    dispose();
        new message1().setVisible(true);
}
catch(HeadlessException | SQLException ex)
{
    JOptionPane.showMessageDialog(null, ex);
}
    }
    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
            try{
  String sql="delete from printinfo where name=?";

connect=DriverManager.getConnection("jdbc:mysql://localhost/atm","root","CSE0
18");
```

```
    pst=connect.prepareStatement(sql);
    pst.setString(1,jTextField1.getText());
  pst.executeUpdate();
    JOptionPane.showMessageDialog(null, " deleted Successfully!");
   }
catch(HeadlessException | SQLException ex)
{
    JOptionPane.showMessageDialog(null, ex);
}
    }
```

CHECK BALANCE(WITHDRAWAL)

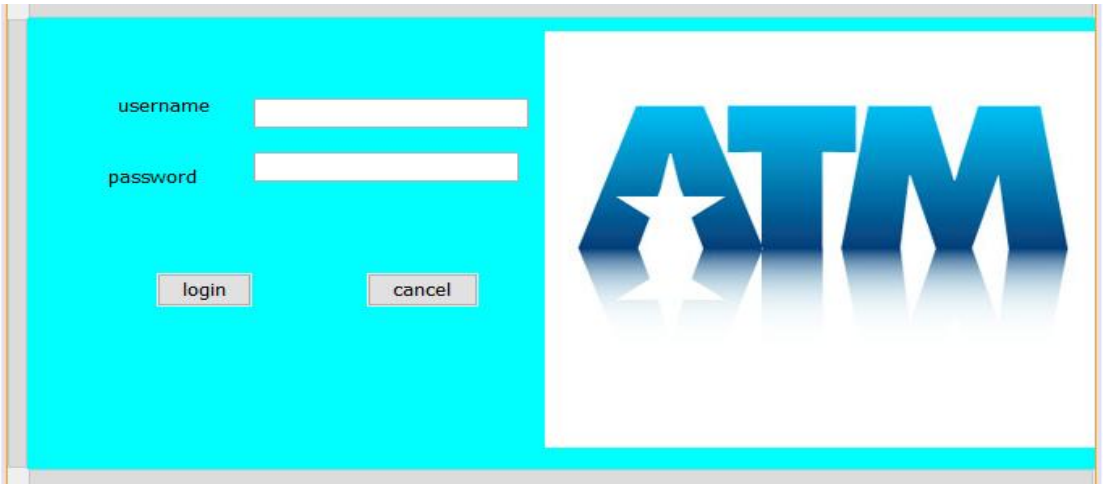
```
try {
            double ta = Double.parseDouble(tFirst.getText());
            double wa = Double.parseDouble(tSecond.getText());
            double bal;
            bal = (ta-wa);
            String result = String.format("%.0f", bal);
            tResult.setText(result);
 }
 catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "enter valid number", "Just
Numbers", JOptionPane.INFORMATION_MESSAGE);
        }
     }
```

BALANCE ENQUIRY FOR CASH DEPOSIT

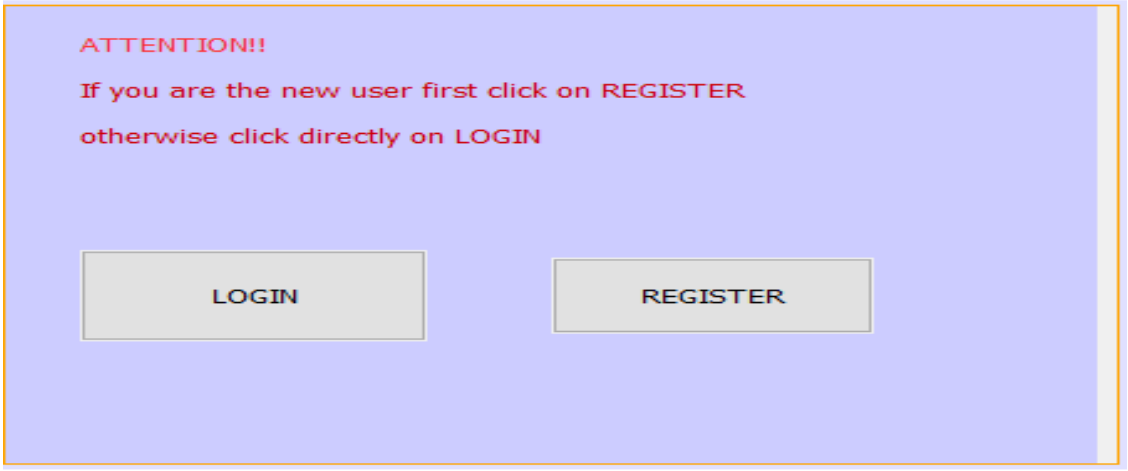
```
 try {
        double ad = Double.parseDouble(txtFirst.getText());
        double ab = Double.parseDouble(txtSecond.getText());
        double total;
        total = (ad + ab);
        String result = String.format("%.0f", total);
        txtResult.setText(result);
 }
      catch (NumberFormatException e) {

 JOptionPane.showMessageDialog(null, "enter valid number", "Just Numbers",
JOptionPane.INFORMATION_MESSAGE);
      }
   }
```

SNAPSHOTS

Login:



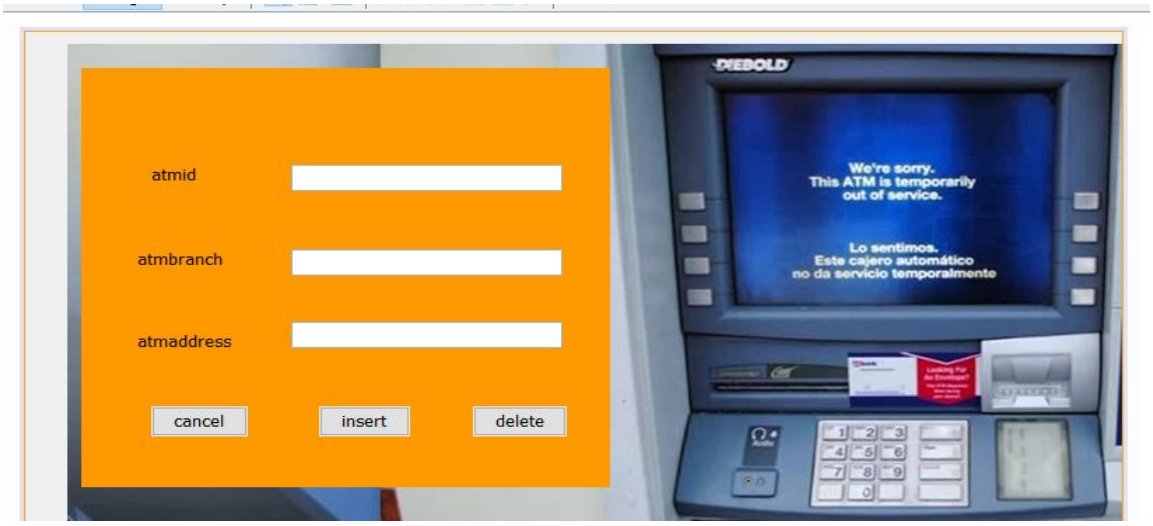
Type of user



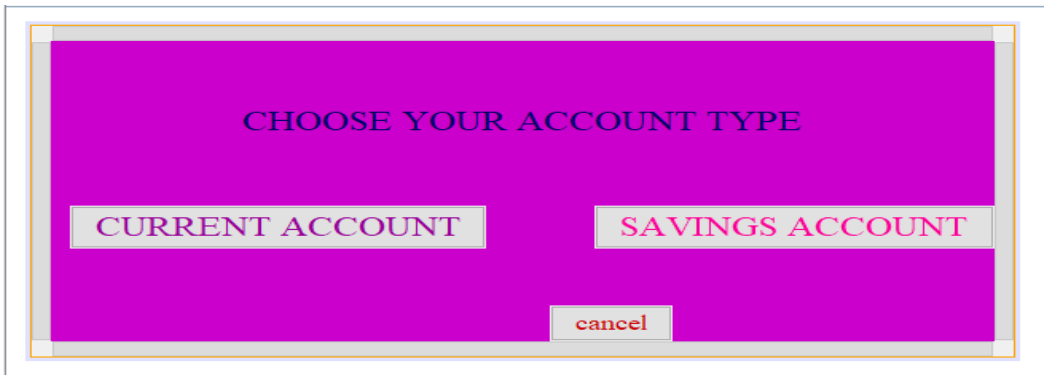
Old user

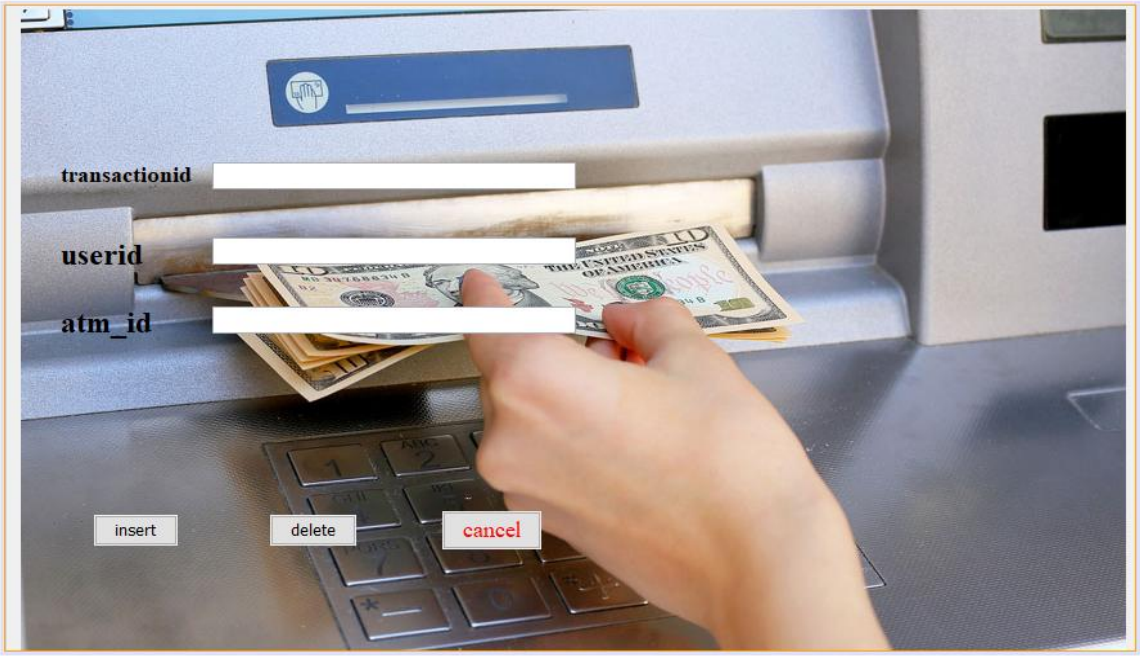
New user

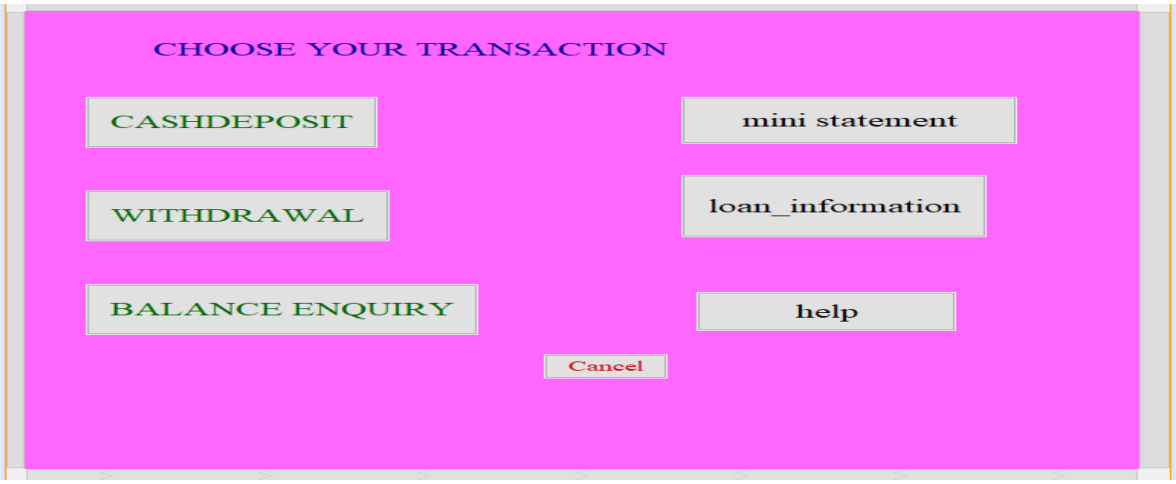
Atm System



Transaction

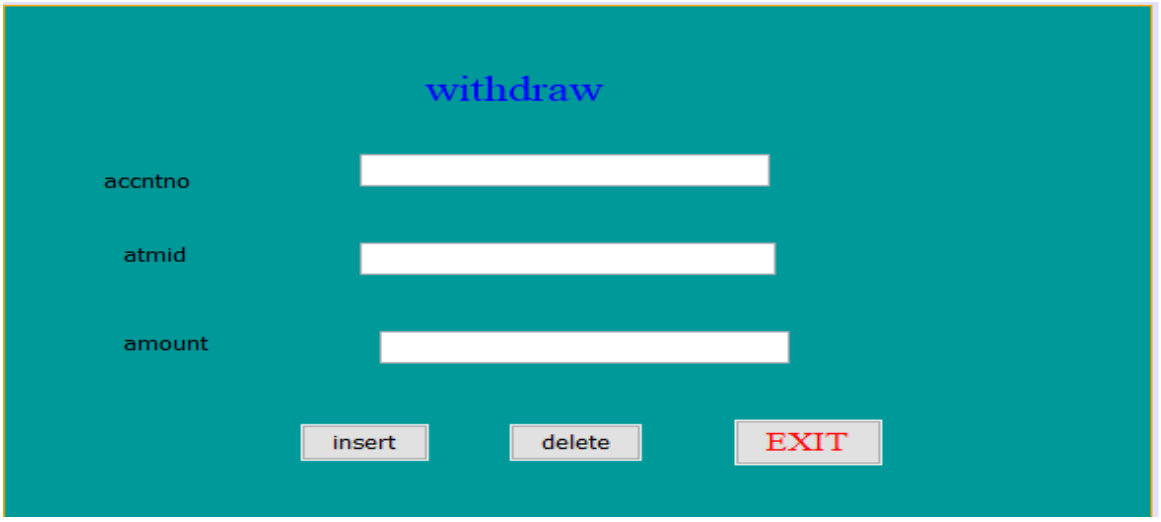




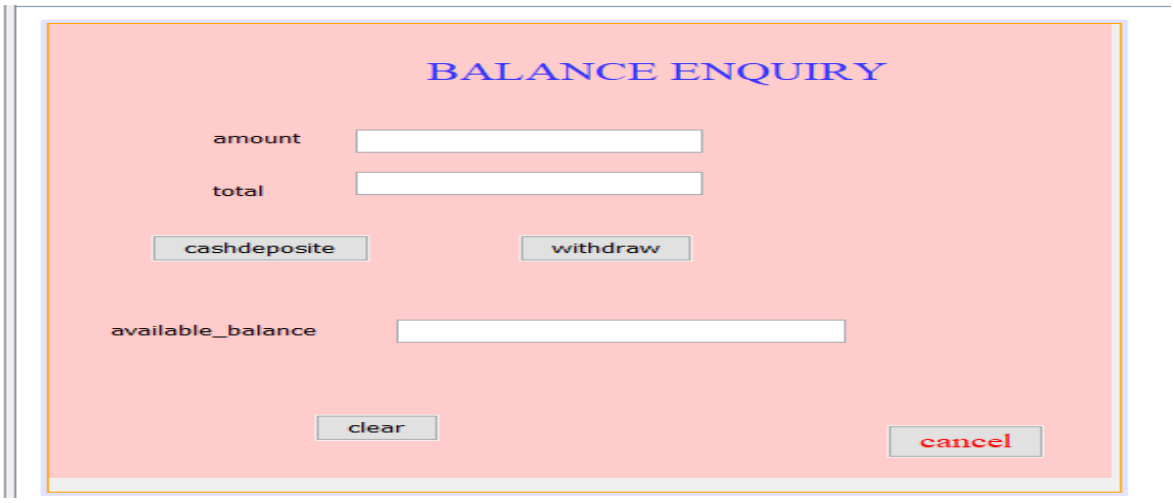


Cash deposit

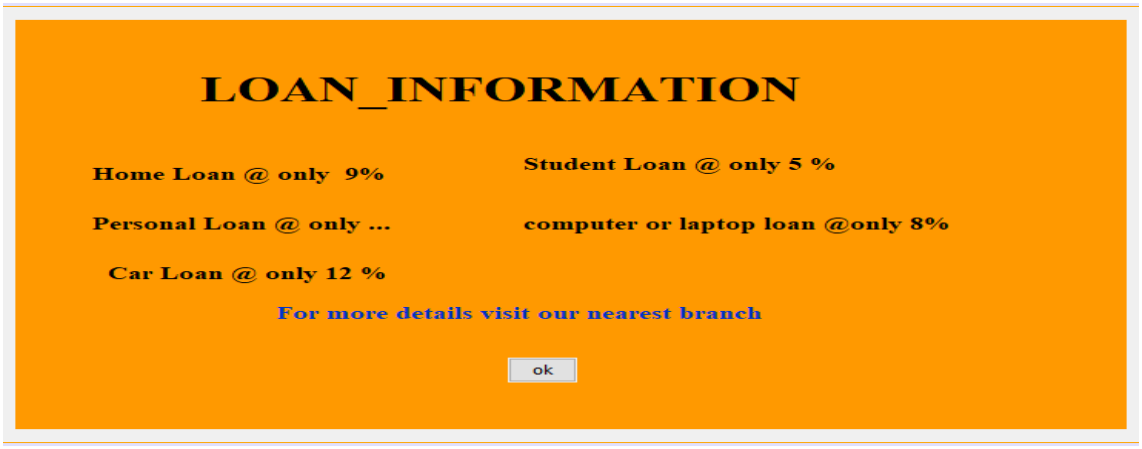
Withdrawal



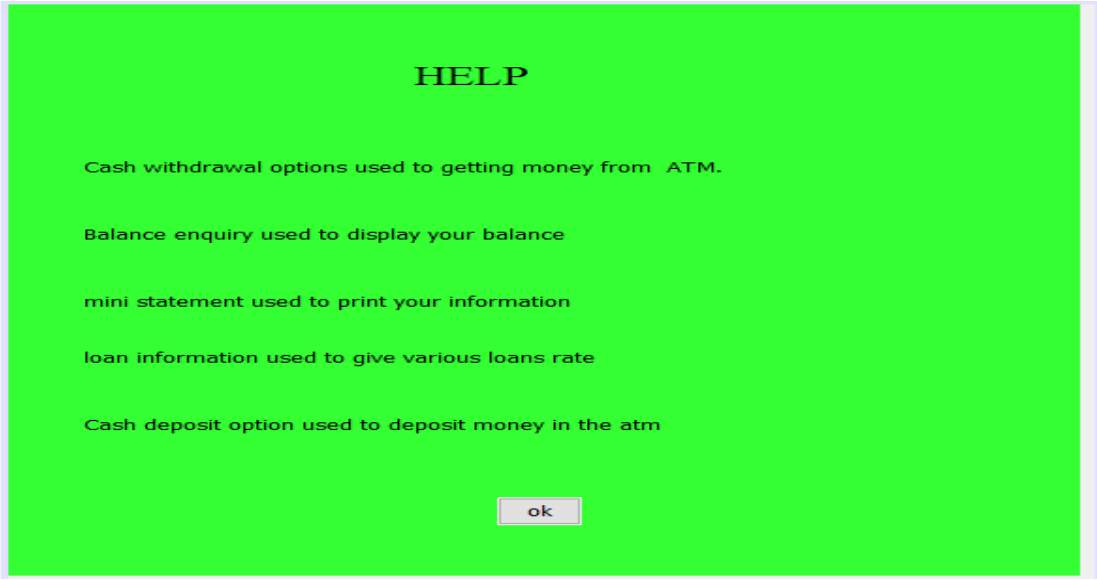
Balance Enquiry



Loan information



Help



Print info