TOPIC: BANKING SYSTEM USING JAVA

GROUP MEMBERS:

NAME COLLEGE REGISTRATION NO:

1. Subhrangshu Chaudhuri JIS COLLEGE OF Engineering 15489
2. Subhajit Ganguly JIS COLLEGE OF Engineering 15452
3. Sukanta Sharma JIS COLLEGE OF Engineering 15499
4. Subhro Dutta JIS COLLEGE OF Engineering 15502
5. Satyabrata Sarkar JIS COLLEGE OF Engineering 15501

**INDEX**

PAGE NO

ITEM Preface……………………………………………………………………………………………………………………..1

Letter of Transmittal ………………………………………………………………………………………………..2

Acknowledgement …………………………………………………………………………………………………..3

Table of Contents …………………………………………………………………………………………………….4

List of illustrations ……………………………………………………………………………………………………5

Abstract …………………………………………………………………………………………………………………..6

Introduction …………………………………………………………………………………………………………….9

Methodology …………………………………………………………………………………………………………..10

**Discussion**

*Origin of Games ……………………………………………………………………………………………………. 11*

*History of Olympia …………………………………………………………………………………………………12*

*Culture ………………………………………………………………………………………………………………….13*

TITLE PAGE…………………………………………………………………………………………………….1

INDEX………………………………………………………………………………………………...............2

ACKNOWLEDGEMENT……………………………………………………………………………………3

REQUIREMENT SPECIFICATION…………………………………………………………………….4-6

TABLE DESCRIPTION………………………………………………………………………………….7-10

DATA FLOW DIAGRAM…………………………………………………………………………………11

SCREEN SHOTS…………………………………………………………………………………………12-43

FUTURE SCOPE OF IMPROVEMENT ……………………………………………………………….44

CODE……………………………………………………………………………………………………...45-122

PROJECT CERTIFICATES…………….…………………………………………………………..123-127

**ACKNOWLEDGEMENT**

First of all we would like to thank God as finally we were able to finish our project that was given to us. This task had been done with all effort by our group members.

We were able to adapt properly and wisely. Besides that, big thank we address to our course lecturer because without his guidance our project could not be done properly like this.

Finally, thank to our beloved friends that always stick together and also work hard to produce a good project with all afford and responsibility. Hope that all the determination will give a lot of benefits to us and also to our group project. Million thanks also we wish to our entire classmates because they also help us in doing our group. They always gave us ideas and comments on our project so that we can improve our project in many ways.

This project has been a great learning curve for us which will surely hold us in good stead in the upcoming years of our life.

Finally we would like to extend our thanks to the whole Globsyn Skills fraternity for this truly memorable one month of our life.

**REQUIREMENT SPECIFICATION**

DOMAIN DESCRIPTION:

Q. What is the banking system? Who are the main players/actor?

Banking system in our project refers to the manual data entry-file type system, without the use of the online type, where there are mainly two actors, the OPERATOR & ADMINISTRATOR . In the bank , here the user provides the information to the operator about the job the user wants and conveys that to the operator. The operator on behalf of the user fills the necessary tabloid for that application of the user to be processed.

Here in our project is the main player is the ADMIN. The Admin creates an operator who works on behalf of the Admin.

PROBLEM DEFINITION:

Q. Why should anyone take our project? What are the advantages of our project?

The Banking System of our project can be a real handful , as our project completely evades out the manual records , and also on the other hand reduces the dependence on the internet thus reducing costs. In our system , first an user fills up a form and gives to an appointed operator who can create an account for the user , delete an account , can issue loan , locker , open a savings or a fixed deposit account , also can issue Demand drafts , ATM cards , Cheque book among various other functions . The Administrator of the bank appoints the various operators of the bank and the Admin has access to all these details apart from the operators , so there are basically two main actors in our project the Admin and the operator . Although the ADMIN is the real head in the bank but all the jobs of the user are done by the Operator on behalf of the Admin.

The admin on the other hand can create the operator , delete an operator , view all details of an user , search an User or an Operator. The Admin can also view information on the Locker details , fixed deposit details , loans issued etc.

Moreover our project can store all the records of the user in the Files that are created in our project.

FUNCTIONAL REQUIREMENTS:

ADMINISTRATOR: The main functions of the Administrator in our project are as given below:

1. Creating or adding a new Operator for the bank.
2. The admin can search an Operator about the whereabouts of the Operator.
3. The Admin can also delete an Operator.
4. Viewing all the Operators together, their details in the form of a list.
5. The Admin can also search an User of the bank.
6. The Admin can also view all the users at once, besides the Admin can also delete a User without asking from the Operator, such is the power of the Administrator.
7. Viewing the Loan details and the Locker details.
8. A big factor is that when a User wants to take Loan or a Locker then without the permission or the consent of the Admin the loan or the Locker cannot be sanctioned by the Operator itself.
9. Besides the Admin can also view the fixed deposit details as well.
10. The Admin has an Approve function created from where the consent for taking the loan is forwarded to the Operator.
11. The Admin can check his own Login History or that of any other operator.

OPERATOR: The main functions of the Operator in our project are as given below:

1. Adding a new account for the User.
2. Deleting an existing account of the user.
3. Adding a withdraw function for the money transactions.
4. Adding a deposit function for transactions in the account of the user.
5. Adding a money transfer function for the transfer of funds from account of one user to the other account.
6. The Operator can check the Individual Passbook of each account holder in the bank just by searching the account number or the username.
7. Adding a function for viewing the details of an account holder in the bank.
8. Adding a function for issuing a cheque book, a Demand Draft, ATM card.
9. Adding a function for applying for a loan to be formally approved only by the Admin.
10. Adding a function for a fixed deposit account for the user.
11. Adding a function for applying of a locker which will also be approved by the Admin.

HARDWARE/SOFTWARE REQUIREMENTS:

1. CPU :- Intel i3
2. RAM :- 512MB
3. Operating system :- Windows 7 or Higher version
4. Platform :- Eclipse Juno (x86)
5. JavaSE vesion 7 update 21

DATABASE DESIGN

TABLE DESCRIPTION

CLASS NAME: Create

FILE NAME: Regis.dat

|  |  |  |  |
| --- | --- | --- | --- |
| SL NO | DATA TYPE | VARIABLE NAME | VARIABLE DESCRIPTION |
| 1 | String | name | Stores the user first name. |
| 2 | String | name1 | Stores the user last name. |
| 3 | String | address | Stores the user address. |
| 4 | String | email | Stores the user email id. |
| 5 | String | nationality | Store the user nationality. |
| 6 | String | acctype | Store the user account type. |
| 7 | String | city | Stores the user city name. |
| 8 | String | gender | Stores the sex of the user |
| 9 | String | dob | Stores the date of birth of the user |
| 10 | String | date | Stores the date of the user registration |
| 11 | String | time | Stores the time of the user registration |
| 12 | String | Identity | Stores the Identity of the user |
| 13 | String | profession | Stores the profession of the user |
| 14 | String | initialamnt | Stores the initial amount of the user |

CLASS NAME: OperatorRegData

FILE NAME: Reg.dat

|  |  |  |  |
| --- | --- | --- | --- |
| SL NO | DATA TYPE | VARIABLE | VARIABLE DESCRIPTION |
| 1 | String | name | Stores first name of operator |
| 2 | String | name1 | Stores last name of operator |
| 3 | String | password | Stores password of operator |
| 4 | String | address | Stores address of operator |
| 5 | String | city | Stores the city name of operator |
| 6 | String | gender | Stores the sex of the operator |
| 7 | String | salary | Stores the salary of the operator |
| 8 | String | dob | Stores the date of birth of the operator |
| 9 | String | date | Stores the system date of registration |
| 10 | String | time | Stores the system time of registration |

CLASS NAME: Actotal

FILE NAME: actotal.dat

|  |  |  |  |
| --- | --- | --- | --- |
| **SL NO** | **DATA TYPE** | **VARIABLE** | **VARIABLE DESCRIPTION** |
| 1 | int | total | Stores the amount present in user account |
| 2 | int | account | Stores user account number |
| 3 | int | depo | Stores the deposit made by the user |
| 4 | int | withdraw1 | Stores the withdrawal made by the user |
| 5 | String | date | Stores the date of user transactions |
| 6 | String | time | Stores the time of user transactions |

CLASS NAME: Depo

FILE NAME: fixed.dat

|  |  |  |  |
| --- | --- | --- | --- |
| **SL NO** | **DATA TYPE** | **VARIABLE NAME** | **VARIABLE DESCRIPTION** |
| 1 | String | savingsacc | Stores user savings account number |
| 2 | String | fixeddepoacc | Stores user fixed deposit account number. |
| 3 | String | savingsaccbal | Stores user savings account balance |
| 4 | String | fixedamnt | Stores user fixed deposit amount |
| 5 | String | terms | Stores terms of fixed deposit |
| 6 | String | rate | Stores rate of fixed deposit of user |
| 7 | String | date | Stores system date |
| 8 | String | time | Stores system time |

CLASS NAME: LoanData

FILE NAME: loan.dat

|  |  |  |  |
| --- | --- | --- | --- |
| SL No | DATA TYPE | VARIABLE NAME | VARIABLE DESCRIPTION |
| 1 | String | bankacc | Stores account number of user |
| 2 | String | acc | Stores loan account number of user |
| 3 | String | amnt | Stores loan amount of user |
| 4 | String | acctype | Stores loan account type of user |
| 5 | String | terms | Stores loan terms of user |
| 6 | String | Rate | Stores loan rate of user |
| 7 | String | date | Stores system date |
| 8 | String | time | Stores system time |
| 9 | String | approve | Stores approval which is default set False for all users. |

CLASS NAME: LockerData

FILE NAME: locker.dat

|  |  |  |  |
| --- | --- | --- | --- |
| SL NO | DATA TYPE | VARIABLE NAME | VARIABLE DESCRIPTION |
| 1 | String | acc | Stores account number of user |
| 2 | String | lockno | Stores locker number of user |
| 3 | String | bal | Stores balance present in savings account of user |
| 4 | String | terms | Stores locker terms of user |
| 5 | String | date | Stores System date |
| 6 | String | time | Stores System time |
| 7 | String | decision | Stores decision which is default set False for all users. |

CLASS NAME: AdminLoginHistoryData

FILE NAME: AdminLoginData.dat

|  |  |  |  |
| --- | --- | --- | --- |
| SL NO | DATA TYPE | VARIABLE NAME | VARIABLE DESCRIPTION |
| 1 | String | ALoginDate | Stores login date of admin |
| 2 | String | ALoginTime | Stores login time of admin |
| 3 | String | ALogoutDate | Stores logout date of admin |
| 4 | String | ALogoutTime | Stores logout time of admin |

CLASS NAME: OperatorLoginHistoryData

FILE NAME: OperatorLoginData.dat

|  |  |  |  |
| --- | --- | --- | --- |
| SL NO | DATA TYPE | VARIABLE NAME | VARIABLE DESCRIPTION |
| 1 | String | OperatorFirstName | Stores first name of Operator |
| 2 | String | OLoginDate | Stores login date of Operator |
| 3 | String | OLoginTime | Stores login time of Operator |
| 4 | String | OLogoutDate | Stores logout date of Operator |
| 5 | String | OLogoutTime | Stores logout time of Operator |

PAGE FLOW DIAGRAM

ADMINISTRATOR

OPERATOR

LOGIN PAGE:

LOGIN AS

LOG OUT

LOGIN HISTORY

CHANGE PASSWORD

LOAN DETAILS

APPLY FOR LOAN

FIXED DEPOSIT DETAILS

LOCKER DETAILS

FIXED DEPOSIT

APPLY FOR LOCKER

REQUEST FOR DD/CHEQUE BOOK/LOAN

VIEW DETAILS OF AN ACCOUNT

VIEW ACCOUNT PASSBOOK

TRANSFER MONEY

WITHDRAW / DEPOSIT

REMOVE ACCOUNT

CREATE NEW ACCOUNT

*LOG OUT*

*LOAN APPROVAL*

*LOCKER APPROVAL*

*FIXED DEPOSIT DETAILS*

*LOCKER DETAILS*

*SHOW LOAN DETAILS*

*DELETE AN USER*

*SHOW ALL USERS*

*USER SEARCH*

*DELETE AN OPERATOR*

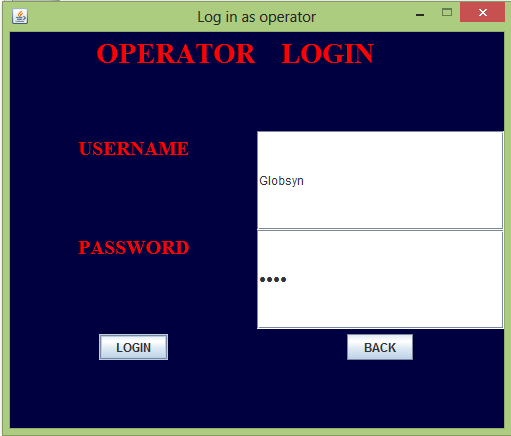
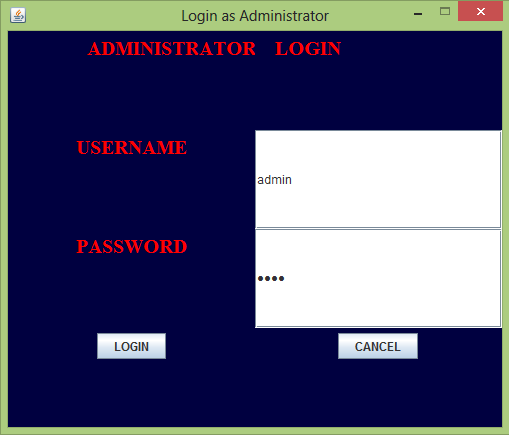
*SHOW ALL OPERATOR*

*SEARCH AN OPERATOR*

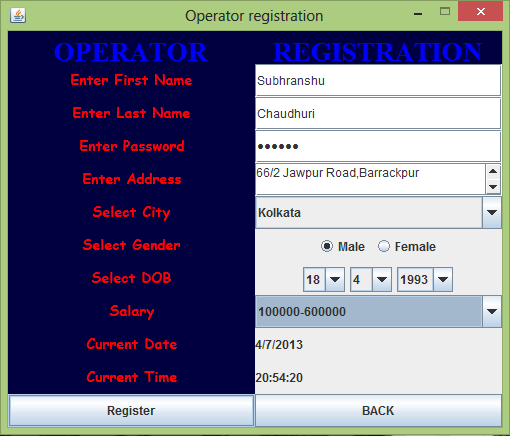
*ADD AN OPERATOR*

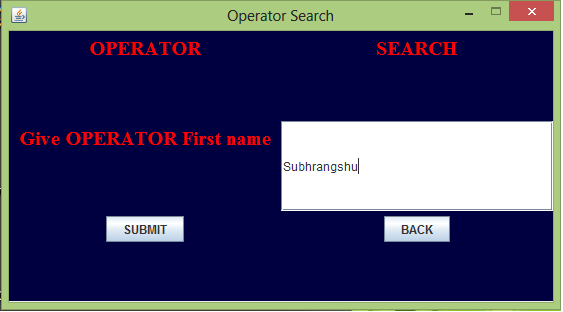
SCREEN SHOTS

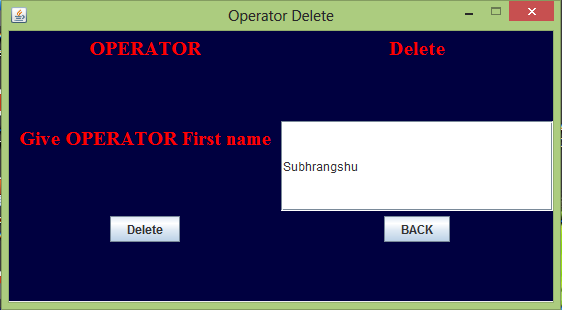


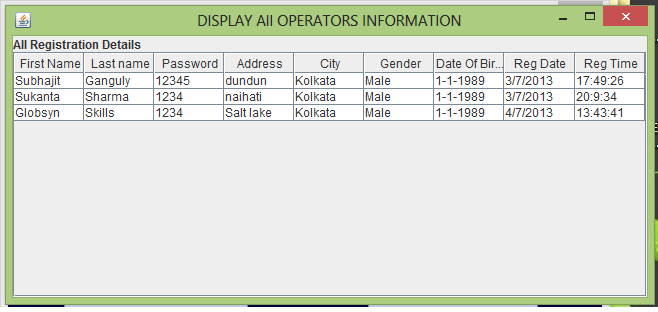


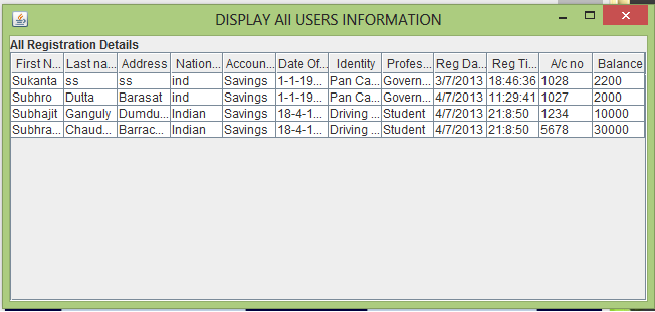




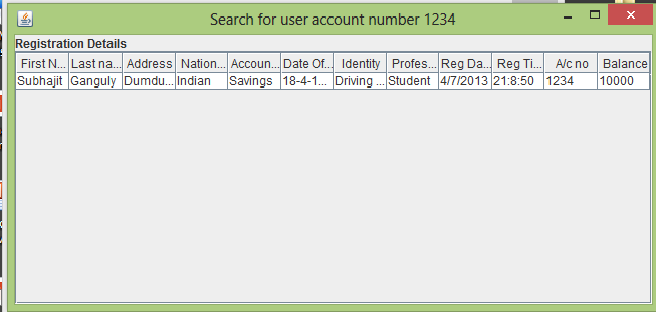


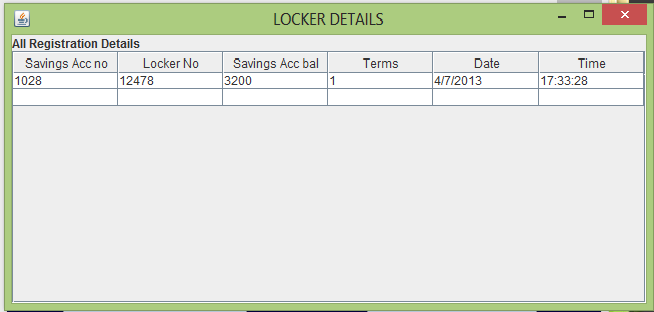


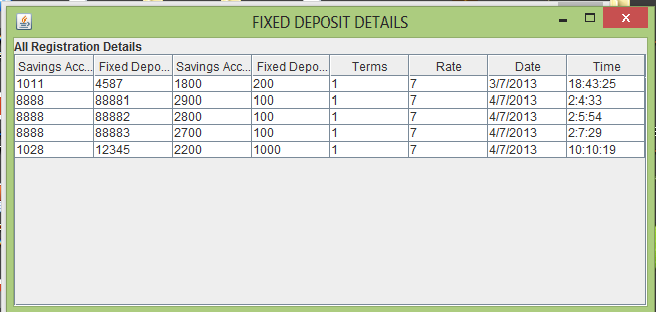


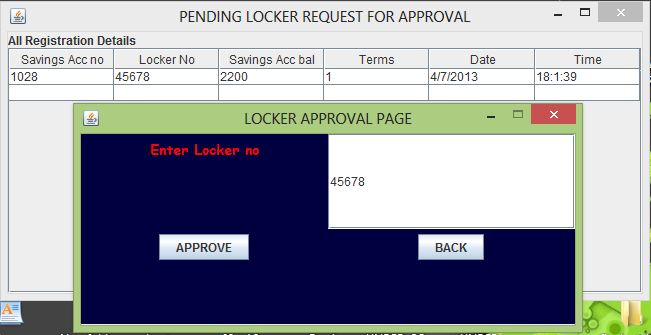


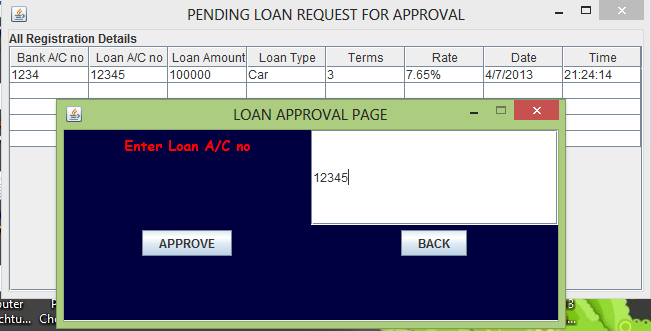


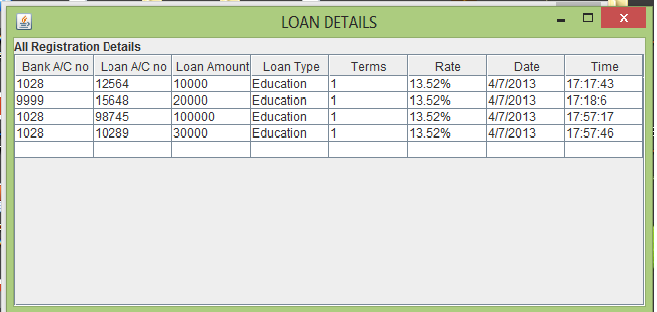




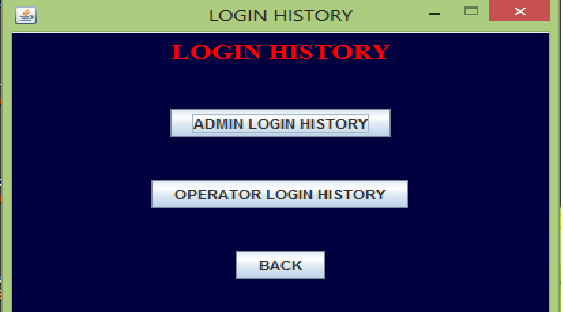


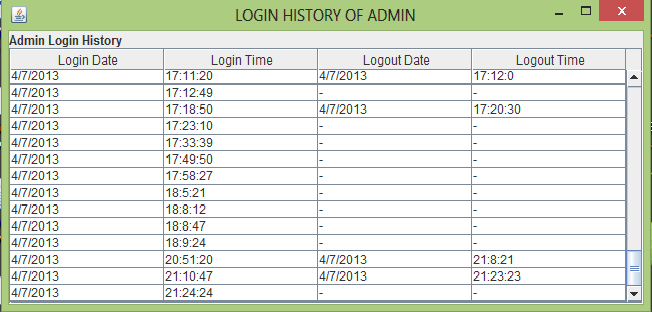


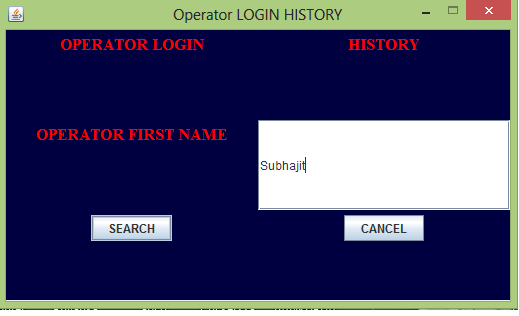


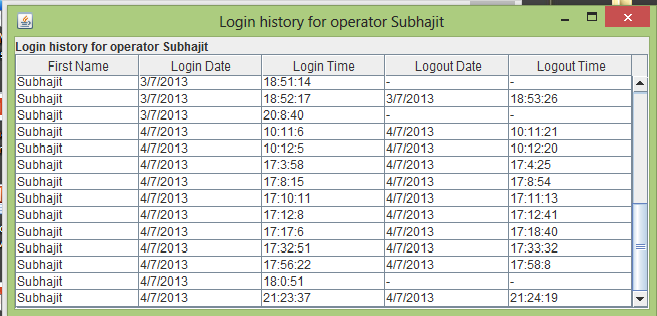


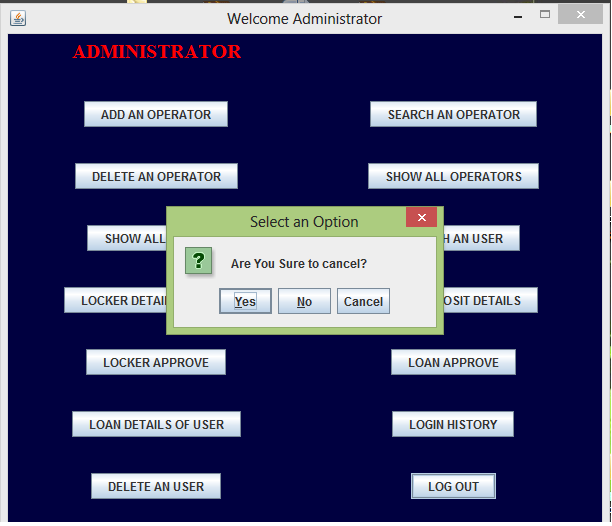








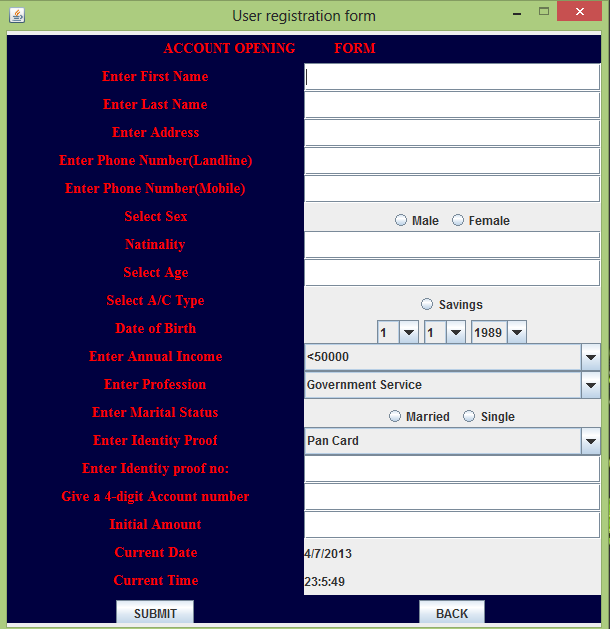




LOGIN AS OPERATOR

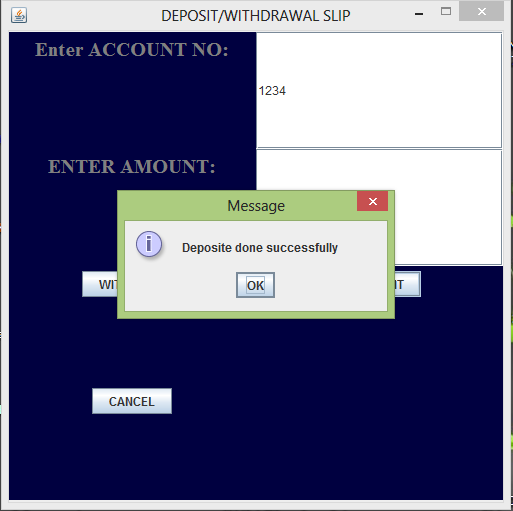


CREATE NEW USER

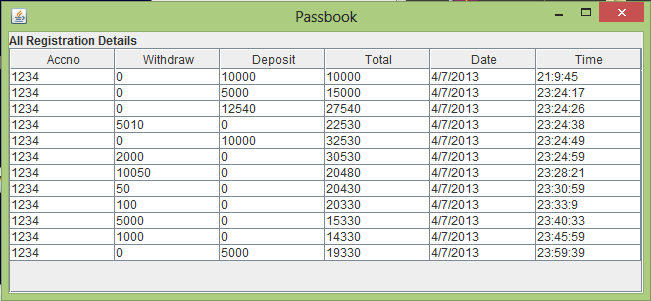


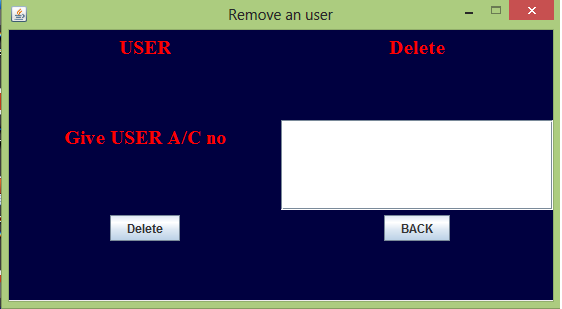
DEPOSIT/WITHDRAWAL

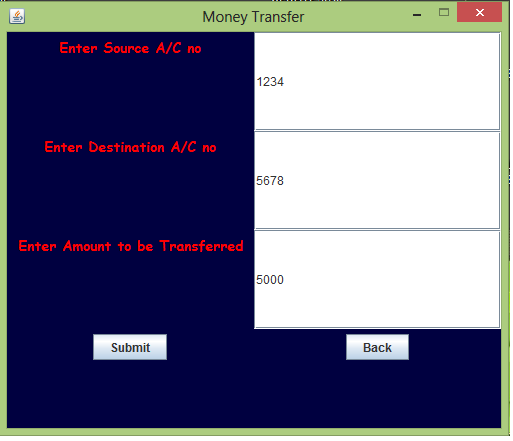


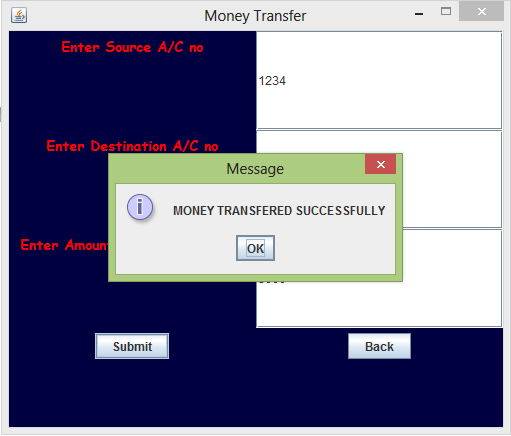


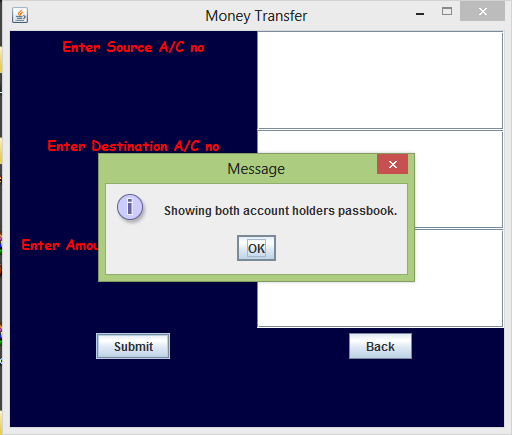
VIEWING PASSBOOK AFTER DEPOSIT

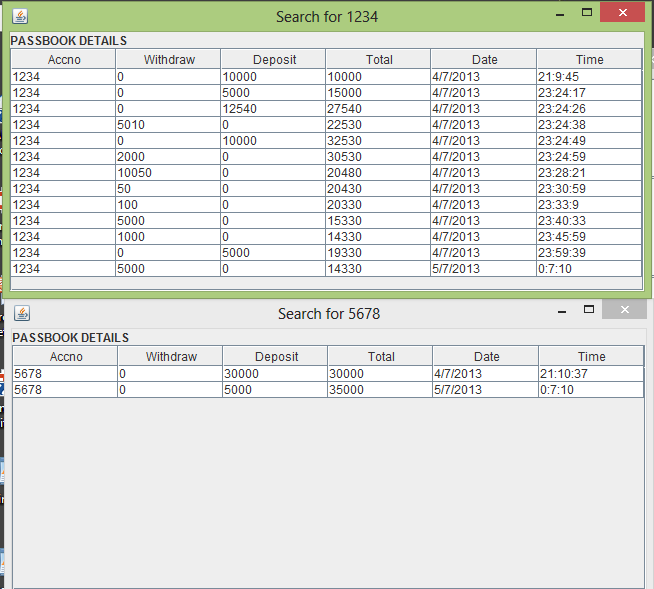


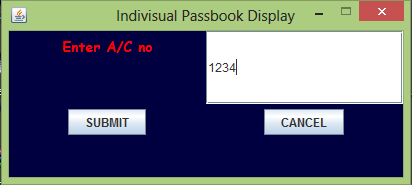


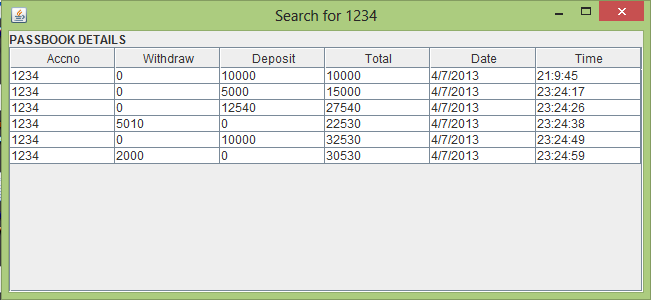


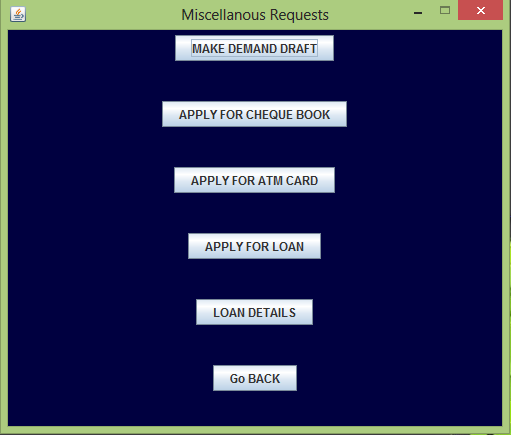


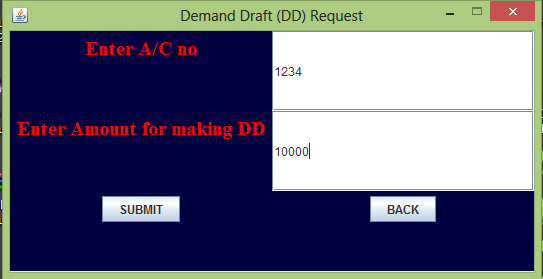


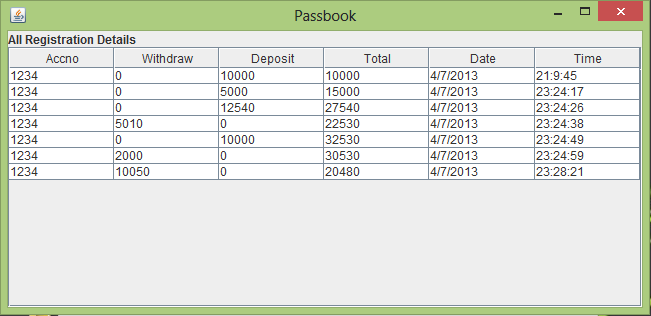


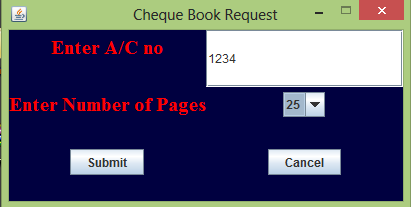


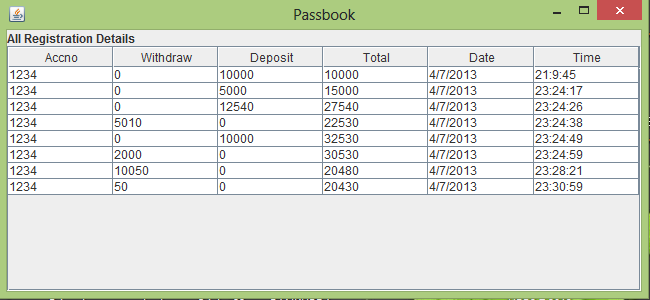


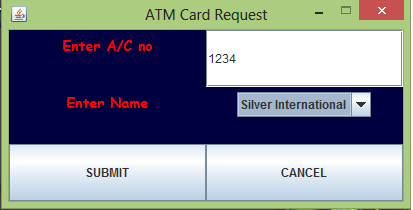


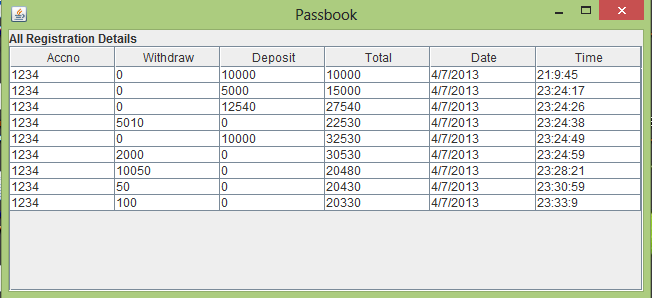


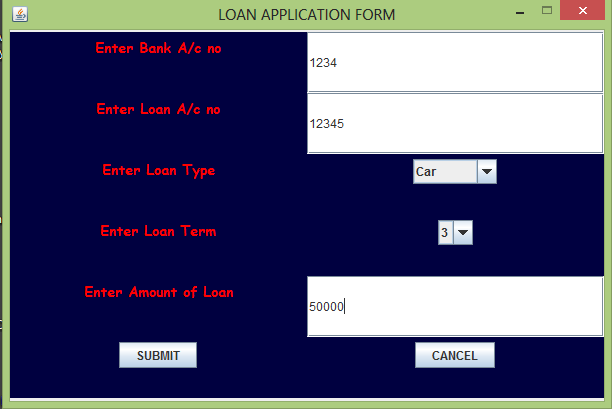


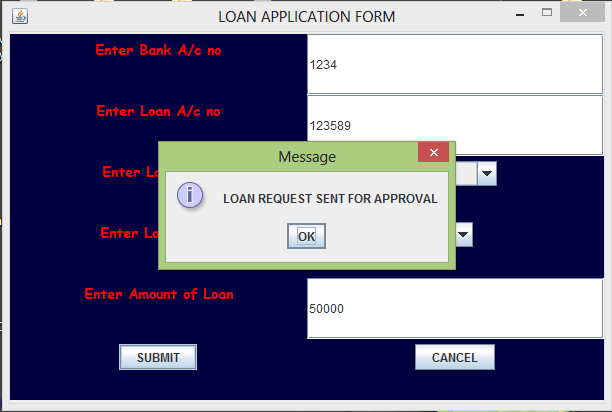


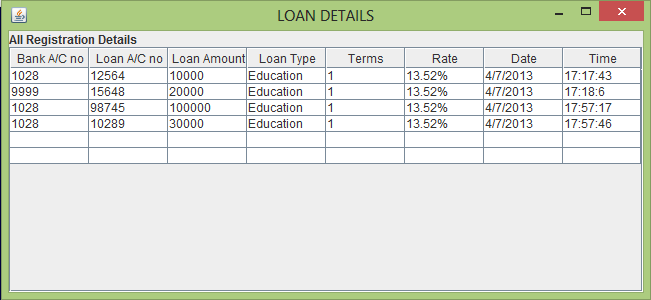


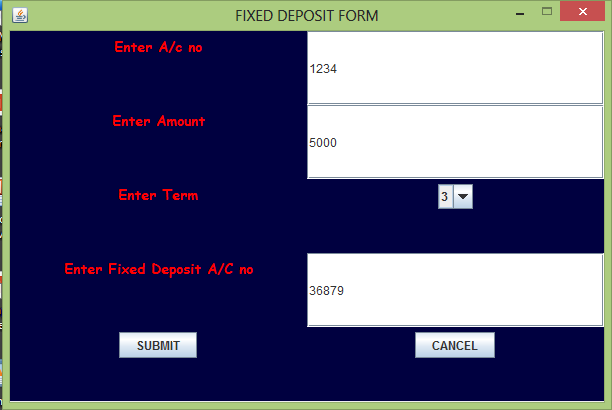


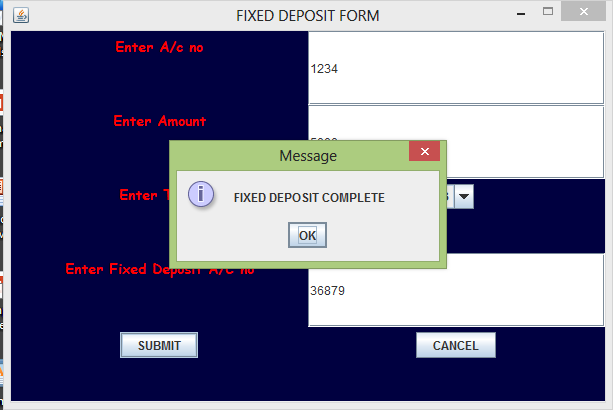


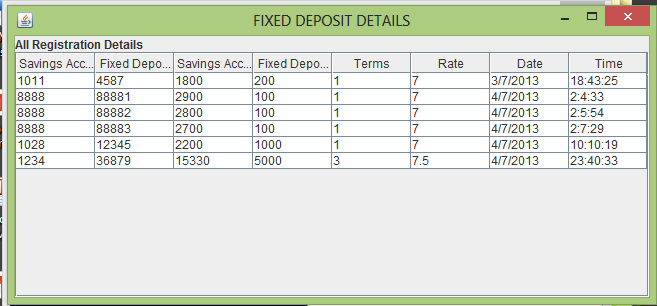






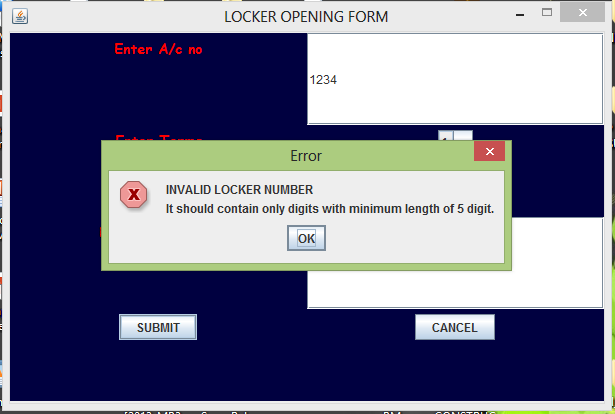


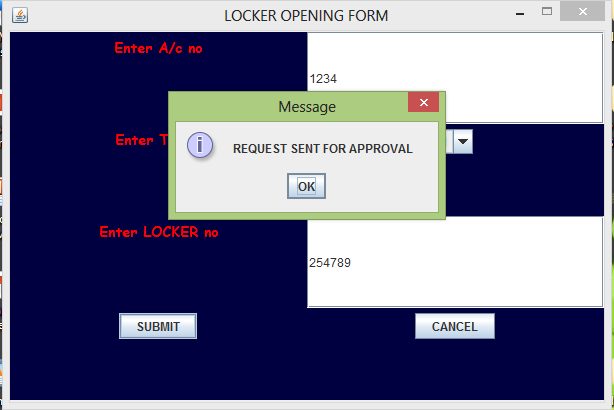


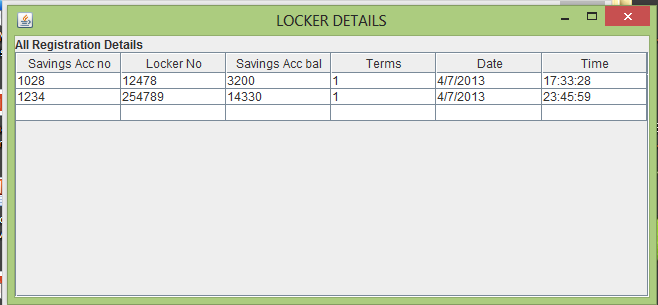




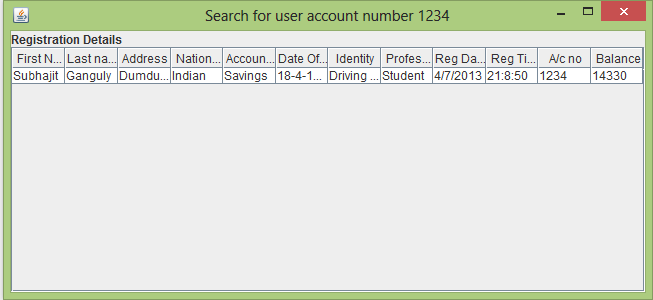


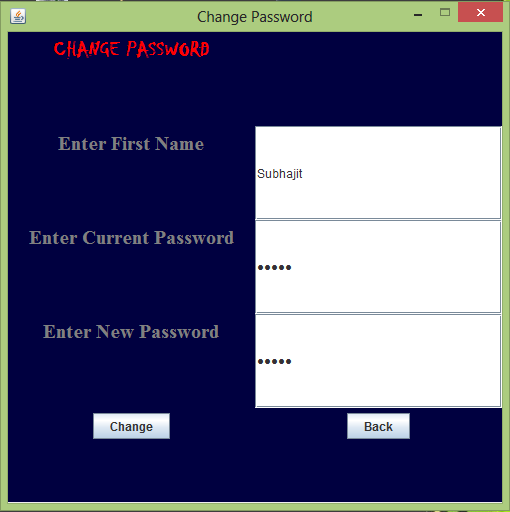


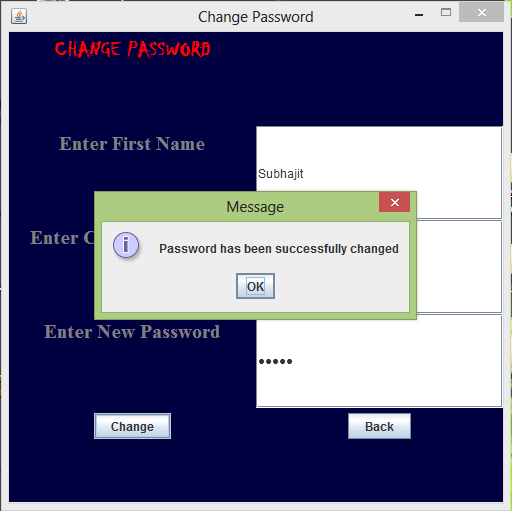


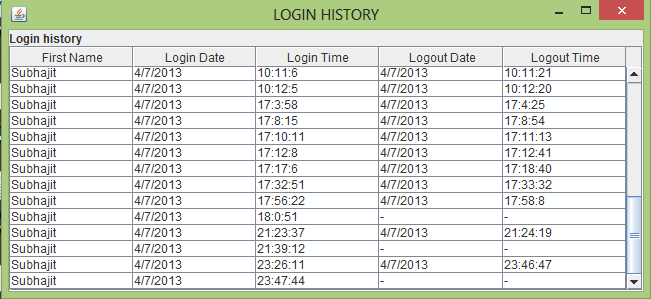












FUTURE SCOPE OF IMPROVEMENT

First of all we would like to thank God as finally we were able to finish our project that was given to us. This task had been done with all effort by our group members.

If at all we would like to improve then we would have loved to add certain new functionalities as well provided we had the time to do it. But as the time span for doing our project was limited we had to be satisfied with this much.

We would have loved features like editing profile, approval of the fixed deposit. More importantly we would have added some graphical features if we had the time to do it.In our present version of our project we haven’t added any graphics but we would have loved to do it.

With the use of JAVA which is object-oriented programming approach, has been very useful in doing our project but we couldn’t avail of the online facility such as online transactions. This type of online transactions could have been done if we had used used other platforms but that’s the utmost.

But overall we are all very satisfied with our effort in making this a very successful project. All that we had thought of is executed in our project by the dedication of our group.

CODE

package test;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.util.\*;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import javax.swing.table.DefaultTableModel;

import javax.swing.\*;

class FirstWindow extends JFrame implements ActionListener

{

private JLabel l0,l1,l2,l3;

private JButton admin,user;

public FirstWindow(String title)

{

super("WELCOME TO UNITED BANK OF PARTNERS");

Container c=getContentPane();

c.setLayout(new GridLayout(3,2));

Font f1=new Font("Times New Roman",Font.BOLD,25);

l0=new JLabel("WELCOME TO UNITED");

l0.setFont(f1);

l0.setForeground(Color.WHITE);

JPanel fpanel=new JPanel();

fpanel.add(l0);

fpanel.setBackground(new Color(0,0,64));

l1=new JLabel("BANK OF PARTNERS ");

l1.setFont(f1);

l1.setForeground(Color.WHITE);

JPanel spanel=new JPanel();

spanel.add(l1);

spanel.setBackground(new Color(0,0,64));

l3=new JLabel("AS: ");

l3.setFont(f1);

l3.setForeground(Color.WHITE);

JPanel xpanel=new JPanel();

xpanel.add(l3);

xpanel.setBackground(new Color(0,0,64));

Font f2=new Font("Times New Roman",Font.BOLD,25);

l2=new JLabel(" LOGIN");

l2.setFont(f2);

l2.setForeground(Color.WHITE);

//l1.setBorder(new javax.swing.border.LineBorder(java.awt.Color.GREEN, 4));

JPanel ppanel=new JPanel();

ppanel.add(l2);

ppanel.setBackground(new Color(0,0,64));

admin=new JButton("ADMINISTRATOR");

JPanel gpanel=new JPanel();

gpanel.add(admin);

gpanel.setBackground(new Color(0,0,64));

admin.addActionListener(this);

gpanel.add(new JLabel(""));

user=new JButton("OPERATOR");

JPanel qpanel=new JPanel();

qpanel.add(user);

qpanel.setBackground(new Color(0,0,64));

user.addActionListener(this);

qpanel.add(user);

c.add(fpanel);c.add(spanel);

c.add(ppanel);c.add(xpanel);

c.add(gpanel);c.add(qpanel);

setSize(600,450);

setLocation(200,200);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e)

{

if(e.getSource()==admin)

{

//JOptionPane.showMessageDialog(this, "HELLO....");

new SecondWindow() ;

setVisible(false);

}

if(e.getSource()==user)

{

//JOptionPane.showMessageDialog(this, "HELLO....");

new ThirdWindow() ;

setVisible(false);

}

}

}

public class Main

{

public static void main(String[] args)

{

new FirstWindow("WELCOME TO UNITED BANK OF PARTNERS");

}

}

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.util.\*;

import java.awt.Component;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import javax.swing.\*;

class SecondWindow extends JFrame implements ActionListener

{

private JLabel l0,l1,l2,l3;

private JTextField t1;

private JPasswordField p1;

private JButton login,cancel;

private String sysDate,sysTime;

private ArrayList<AdminLoginHistoryData> alist = new ArrayList<AdminLoginHistoryData>();

//private JLabel lTimeValue;

//private JLabel lDateValue;

//private Component lDate;

//private Component lTime;

private boolean flagID = false;

private boolean flagPass = false;

public SecondWindow()

{

super("Login as Administrator");

Container c=getContentPane();

c.setLayout(new GridLayout(4,2));

Font f1=new Font("Times New Roman",Font.BOLD,20);

l1=new JLabel(" ADMINISTRATOR");

l1.setFont(f1);

l1.setForeground(Color.RED);

JPanel fpanel=new JPanel();

fpanel.add(l1);

fpanel.setBackground(new Color(0,0,64));

l0=new JLabel("LOGIN ");

l0.setFont(f1);

l0.setForeground(Color.RED);

JPanel epanel=new JPanel();

epanel.add(l0);

epanel.setBackground(new Color(0,0,64));

//Font f1=new Font("Times New Roman",Font.BOLD,20);

l2=new JLabel("USERNAME");

l2.setFont(f1);

l2.setForeground(Color.RED);

JPanel apanel=new JPanel();

apanel.add(l2);

apanel.setBackground(new Color(0,0,64));

//Font f1=new Font("Times New Roman",Font.BOLD,20);

l3=new JLabel("PASSWORD");

l3.setFont(f1);

l3.setForeground(Color.RED);

JPanel bpanel=new JPanel();

bpanel.add(l3);

bpanel.setBackground(new Color(0,0,64));

t1=new JTextField();

p1=new JPasswordField();

login=new JButton("LOGIN");

login.addActionListener(this);

JPanel cpanel=new JPanel();

cpanel.add(login);

cpanel.setBackground(new Color(0,0,64));

cancel=new JButton("CANCEL");

cancel.addActionListener(this);

JPanel dpanel=new JPanel();

dpanel.add(cancel);

dpanel.setBackground(new Color(0,0,64));

/\*\*\*\*\*\*\*\*DATE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Calendar cal = Calendar.getInstance();

String cday = ""+cal.get(Calendar.DATE);

int x =cal.get(Calendar.MONTH);

String cmonth = ""+(x+1);

String cyear =""+cal.get(Calendar.YEAR);

sysDate = cday+"/"+cmonth+"/"+cyear;

//lDateValue = new JLabel(sysDate);

String chr = ""+cal.get(Calendar.HOUR\_OF\_DAY);

String cmin = ""+cal.get(Calendar.MINUTE);

String csec =""+cal.get(Calendar.SECOND);

sysTime = chr+":"+cmin+":"+csec;

/\*lTimeValue=new JLabel(sysTime);

JPanel datePanel = new JPanel();

datePanel.add(lDate);

datePanel.add(lDateValue);

JPanel timePanel = new JPanel();

timePanel.add(lTime);

timePanel.add(lTimeValue);\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

c.add(fpanel);c.add(epanel);

c.add(apanel);c.add(t1);

c.add(bpanel);c.add(p1);

c.add(cpanel);c.add(dpanel);

setSize(500,425);

setLocation(200,200);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e)

{

AdminLoginHistoryData obj;

if(e.getSource()==login)

{

/\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*/

String n = t1.getText().trim();

String ps = p1.getText().trim();

/\* user id check

\* \*/

String namepattern = "^[A-Za-z]";

Scanner scan = new Scanner( n ) ;

String matched = scan.findInLine( namepattern ) ;

if ( matched == null )

{

//JOptionPane.showMessageDialog(this, "INVALID USER NAME\nIt should contain only alphabet.");

JOptionPane.showMessageDialog(this, "INVALID USER NAME\nIt should contain only alphabet.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagID = true;

}

String passwordpattern = "^[0-9]{4}" ;

Scanner scan2 = new Scanner( ps ) ;

String matched2 = scan2.findInLine( passwordpattern ) ;

if ( matched2 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

p1.setText("");

}

else

{

flagPass = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*if the both are valid the \*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagID == true )

{

if(flagPass == true)

{

/\*user id and pass word check ------> login

\* \*/

if(t1.getText().equals("admin") && p1.getText().equals("1234"))

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

obj = new AdminLoginHistoryData();

obj.setALoginDate(sysDate);

obj.setALoginTime(sysTime);

obj.setALogoutDate(" - ");

obj.setALogoutTime(" - ");

alist.add(obj);

new AdminLoginHistoryAddInfo(obj);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

JOptionPane.showMessageDialog(this, "Welcome Administrator");

new Admin() ;

setVisible(false);

}

/\* else part password error

\* \*/

else

{

//JOptionPane.showMessageDialog(this, "Wrong username/password...\nTry again");

JOptionPane.showMessageDialog(this, "Wrong username/password...\nTry again", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

p1.setText("");

}

}

}

flagID = false;

flagPass = false;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* NOT USABLE PART \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*else

{

JOptionPane.showMessageDialog(this, "Try Agian");

}\*/

/\*user id and pass word check ------> login

\* \*/

/\*if(t1.getText().equals("admin") && p1.getText().equals("1234"))

{

JOptionPane.showMessageDialog(this, "Welcome Administrator");

new Admin() ;

setVisible(false);

}\*/

/\*\*\*\*\*\*subhranshsu\*\*\*\*\*\*/

/\*if(t1.getText().equals("admin"))

{

char [] pwd=p1.getPassword();

String str=new String(pwd);

if(str.equals("1234"))

{

JOptionPane.showMessageDialog(this, "Welcome Administrator");

new Admin() ;

setVisible(false);

}

}\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* else part password error

\* \*/

/\*else

{

JOptionPane.showMessageDialog(this, "Wrong username/password..");

t1.setText("");

p1.setText("");

}

}\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(e.getSource()==cancel)

{

int con=JOptionPane.showConfirmDialog(this, "Are you sure to cancel?");

if(con==JOptionPane.YES\_OPTION)

{

new FirstWindow("");

setVisible(false);

}

}

}

}

public class Main1

{

public static void main(String[] args)

{

}

}

import java.awt.Container;

import java.awt.GridLayout;

import java.awt.Color;

import java.util.\*;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.event.ActionListener; //to sense mouse click

import java.awt.event.ActionEvent;

import java.io.Serializable;

import java.util.Calendar;

import javax.swing.\*;

public class Actotal extends JFrame implements Serializable

{

private int total;

private int account;

private int depo;

private int withdraw1;

private String date;

private String time;

public String getDate() {

return date;

}

public void setDate(String date) {

this.date = date;

}

public String getTime() {

return time;

}

public void setTime(String time) {

this.time = time;

}

public int getTotal() {

return total;

}

public void setTotal(int total) {

this.total = total;

}

public int getAccount() {

return account;

}

public void setAccount(int account) {

this.account = account;

}

public int getDepo() {

return depo;

}

public void setDepo(int depo) {

this.depo = depo;

}

public int getWithdraw1() {

return withdraw1;

}

public void setWithdraw1(int withdraw1) {

this.withdraw1 = withdraw1;

}

}

import java.awt.Container;

import java.awt.GridLayout;

import java.awt.Color;

import java.util.\*;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.event.ActionListener; //to sense mouse click

import java.awt.event.ActionEvent;

import java.io.Serializable;

import java.util.Calendar;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import javax.swing.\*;

public class Actotalcreate extends JFrame

{

private String sysdate,systime;

//ArrayList<Create> list1;

Actotal sd=new Actotal();

Actotalcreate(int a,int b,int d,int t)

{

Calendar cal = Calendar.getInstance();

String cday = ""+cal.get(Calendar.DATE);

int x =cal.get(Calendar.MONTH);

String cmonth = ""+(x+1);

String cyear =""+cal.get(Calendar.YEAR);

sysdate = cday+"/"+cmonth+"/"+cyear;

String chr = ""+cal.get(Calendar.HOUR\_OF\_DAY);

String cmin = ""+cal.get(Calendar.MINUTE);

String csec =""+cal.get(Calendar.SECOND);

systime = chr+":"+cmin+":"+csec;

sd.setAccount(a);

sd.setWithdraw1(b);

sd.setDepo(d);

sd.setTotal(t);

sd.setDate(sysdate);

sd.setTime(systime);

ArrayList<Actotal> list4;

try

{

FileInputStream fin=new FileInputStream("actotal.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list4=(ArrayList<Actotal>)oin.readObject();

}catch(Exception e)

{

list4=new ArrayList<Actotal>();

}

list4.add(sd);

try

{

FileOutputStream fout=new FileOutputStream("actotal.dat");

ObjectOutputStream oout=new ObjectOutputStream(fout);

oout.writeObject(list4);

}catch(Exception e){}

}

}

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

public class AddInformation1

{

ArrayList<Create> list4;

// ArrayList<Create>list2;

public AddInformation1(Create rg1)

{

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list4=(ArrayList<Create>)oin.readObject();

}catch(Exception e)

{

list4=new ArrayList<Create>();

}

list4.add(rg1);

try

{

FileOutputStream fout=new FileOutputStream("Regis.dat");

ObjectOutputStream oout=new ObjectOutputStream(fout);

oout.writeObject(list4);

}catch(Exception e){}

}

}

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.util.\*;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import javax.swing.\*;

class Admin extends JFrame implements ActionListener

{

private JLabel l1,l2;

private JButton b1,b2,b3,b4,b5,b6,b7,b8;

private JButton b9;

private JButton b10;

private String sysDate;

private String sysTime;

private ArrayList<AdminLoginHistoryData> alist = new ArrayList<AdminLoginHistoryData>();

private JButton b11;

private JButton b12;

private JButton b13;

private JButton b14;

public Admin()

{

/\*\*\*\*\*\*\*\*\*added\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

super("Welcome Administrator");

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*DATE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Calendar cal = Calendar.getInstance();

String cday = ""+cal.get(Calendar.DATE);

int x =cal.get(Calendar.MONTH);

String cmonth = ""+(x+1);

String cyear =""+cal.get(Calendar.YEAR);

sysDate = cday+"/"+cmonth+"/"+cyear;

//lDateValue = new JLabel(sysDate);

String chr = ""+cal.get(Calendar.HOUR\_OF\_DAY);

String cmin = ""+cal.get(Calendar.MINUTE);

String csec =""+cal.get(Calendar.SECOND);

sysTime = chr+":"+cmin+":"+csec;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Container c=getContentPane();

c.setLayout(new GridLayout(8,2));

Font f1=new Font("Times New Roman",Font.BOLD,20);

l1=new JLabel("ADMINISTRATOR");

l1.setFont(f1);

l1.setForeground(Color.RED);

JPanel bpanel=new JPanel();

bpanel.add(l1);

bpanel.setBackground(new Color(0,0,64));

l2=new JLabel("");

l2.setFont(f1);

l2.setForeground(Color.RED);

JPanel cpanel=new JPanel();

cpanel.add(l2);

cpanel.setBackground(new Color(0,0,64));

b1=new JButton("ADD AN OPERATOR");

b1.addActionListener(this);

JPanel apanel=new JPanel();

apanel.add(b1);

apanel.setBackground(new Color(0,0,64));

b2=new JButton("SEARCH AN OPERATOR");

b2.addActionListener(this);

JPanel dpanel=new JPanel();

dpanel.add(b2);

dpanel.setBackground(new Color(0,0,64));

b3=new JButton("DELETE AN OPERATOR");

b3.addActionListener(this);

JPanel epanel=new JPanel();

epanel.add(b3);

epanel.setBackground(new Color(0,0,64));

b4=new JButton("SHOW ALL OPERATORS");

b4.addActionListener(this);

JPanel fpanel=new JPanel();

fpanel.add(b4);

fpanel.setBackground(new Color(0,0,64));

b5=new JButton("SHOW ALL USERS");

b5.addActionListener(this);

JPanel gpanel=new JPanel();

gpanel.add(b5);

gpanel.setBackground(new Color(0,0,64));

b6=new JButton("SEARCH AN USER");

b6.addActionListener(this);

JPanel hpanel=new JPanel();

hpanel.add(b6);

hpanel.setBackground(new Color(0,0,64));

b7=new JButton("DELETE AN USER");

b7.addActionListener(this);

JPanel opanel=new JPanel();

opanel.add(b7);

opanel.setBackground(new Color(0,0,64));

b8=new JButton("LOG OUT");

b8.addActionListener(this);

JPanel qpanel=new JPanel();

qpanel.add(b8);

qpanel.setBackground(new Color(0,0,64));

b9=new JButton("LOAN DETAILS OF USER");

b9.addActionListener(this);

JPanel qopanel=new JPanel();

qopanel.add(b9);

qopanel.setBackground(new Color(0,0,64));

b10=new JButton("LOGIN HISTORY");

b10.addActionListener(this);

JPanel ppanel=new JPanel();

ppanel.add(b10);

ppanel.setBackground(new Color(0,0,64));

b11=new JButton("LOCKER DETAILS OF USER");

b11.addActionListener(this);

JPanel qoopanel=new JPanel();

qoopanel.add(b11);

qoopanel.setBackground(new Color(0,0,64));

b12=new JButton("FIXED DEPOSIT DETAILS");

b12.addActionListener(this);

JPanel qipanel=new JPanel();

qipanel.add(b12);

qipanel.setBackground(new Color(0,0,64));

b13=new JButton("LOCKER APPROVE");

b13.addActionListener(this);

JPanel q2panel=new JPanel();

q2panel.add(b13);

q2panel.setBackground(new Color(0,0,64));

b14=new JButton("LOAN APPROVE");

b14.addActionListener(this);

JPanel q3panel=new JPanel();

q3panel.add(b14);

q3panel.setBackground(new Color(0,0,64));

c.add(bpanel);c.add(cpanel);

c.add(apanel);c.add(dpanel);

c.add(epanel);c.add(fpanel);

c.add(gpanel);c.add(hpanel);

c.add(qoopanel);c.add(qipanel);

c.add(q2panel);c.add(q3panel);

c.add(qopanel);c.add(ppanel);

c.add(opanel);c.add(qpanel);

setSize(600,525);

setLocation(200,200);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e)

{

AdminLoginHistoryData obj;

if(e.getSource()==b1)

{

new OperatorReg("");

setVisible(false);

}

if(e.getSource()==b2)

{

new OperatorSearch();

setVisible(false);

}

if(e.getSource()==b3)

{

new OperatorDelete();

setVisible(false);

}

if(e.getSource()==b4)

{

new OperatorDisplay();

//setVisible(false);

}

if(e.getSource()==b5)

{

//JOptionPane.showMessageDialog(this, "DISPLAY ALL CURRENT RECORDS");

new DiaplayAll() ;

//setVisible(false);

}

if(e.getSource()==b6)

{

new UserSearch();

setVisible(false);

}

if(e.getSource()==b7)

{

new AdminUserRemove();

setVisible(false);

}

if(e.getSource()==b8)

{

int con=JOptionPane.showConfirmDialog(this, "Are You Sure to cancel?");

if(con==JOptionPane.YES\_OPTION)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

obj = new AdminLoginHistoryData();

obj.setALogoutDate(sysDate);

obj.setALogoutTime(sysTime);

//obj.setALoginDate(" - ");

//obj.setALogoutTime(" - ");

alist.add(obj);

new AdminLogoutHistoryAddInfo(obj);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

new SecondWindow();

//new FirstWindow("");

setVisible(false);

}

//new FirstWindow("");

//setVisible(false);

}

if(e.getSource()==b9)

{

new LoanDisplay();

//setVisible(false);

}

if(e.getSource()==b10)

{

new LoginHistory();

setVisible(false);

}

if(e.getSource()==b11)

{

new LockerDisplay();

//setVisible(false);

}

if(e.getSource()==b12)

{

new DepoDisplay();

//setVisible(false);

}

if(e.getSource()==b13)

{

new LockerShowApprove();

new LockerApprove();

//new LockerDisplay();

setVisible(false);

}

if(e.getSource()==b14)

{

new LoanShowApprove();

new LoanApprove();

//new LoanDisplay();

setVisible(false);

}

}

}

public class AdminLogin

{

public static void main(String[] args)

{

}

}

import java.awt.BorderLayout;

import java.awt.Container;

import java.io.FileInputStream;

import java.io.ObjectInputStream;

import java.util.ArrayList;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

import javax.swing.JTable;

public class AdminLoginDisplay extends JFrame

{

public AdminLoginDisplay()

{

super("LOGIN HISTORY OF ADMIN");

String heading[]={"Login Date","Login Time","Logout Date","Logout Time"};

String data[][];//=new String[20][12];//2-d array of string type of 20 rows,9 col

ArrayList<AdminLoginHistoryData> list;//=new ArrayList<Create>();

try

{

FileInputStream fin=new FileInputStream("AdminLoginData.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list=(ArrayList<AdminLoginHistoryData>)oin.readObject();

data = new String[list.size()][heading.length+1];

int r=0;

for(AdminLoginHistoryData re : list)

{

data[r][0]=re.getALoginDate();

data[r][1]=re.getALoginTime();

data[r][2]=re.getALogoutDate();

data[r][3]=re.getALogoutTime();

/\*

data[r][0]=re.getName();

data[r][1]=re.getName1();

data[r][2]=re.getAddress();

data[r][3]=re.getNationality();

data[r][4]=re.getAcctype();

data[r][5]=re.getDob();

data[r][6]=re.getIdentity();

data[r][7]=re.getProfession();

data[r][8]=re.getDate();

data[r][9]=re.getTime();

data[r][10]=re.getAc();

data[r][11]=re.getInitialamnt();

\*/

r++;

}

Container con=getContentPane();

con.setLayout(new BorderLayout());//refer copy

JTable datatable=new JTable(data, heading);//datatable is the obj of the JTABLE,data is the array

JScrollPane jsp=new JScrollPane(datatable);

con.add(new JLabel("Admin Login History"),BorderLayout.NORTH);

con.add(jsp,BorderLayout.CENTER);//jsp=jscrollpane

setSize(650, 300);

setLocation(200, 200);

setVisible(true);

}catch(Exception e)

{

//e.printStackTrace();

//System.out.println(e.toString());

JOptionPane.showMessageDialog(this, "No file found in data base", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

public class AdminLoginHistoryAddInfo

{

ArrayList<AdminLoginHistoryData> list4;

public AdminLoginHistoryAddInfo(AdminLoginHistoryData rg1)

{

try

{

FileInputStream fin=new FileInputStream("AdminLoginData.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list4=(ArrayList<AdminLoginHistoryData>)oin.readObject();

}catch(Exception e)

{

list4=new ArrayList<AdminLoginHistoryData>();

}

list4.add(rg1);

try

{

FileOutputStream fout=new FileOutputStream("AdminLoginData.dat");

ObjectOutputStream oout=new ObjectOutputStream(fout);

oout.writeObject(list4);

}catch(Exception e){}

}

}

import java.io.Serializable;

import javax.swing.JFrame;

public class AdminLoginHistoryData extends JFrame implements Serializable

{

private String ALoginDate;

private String ALoginTime;

private String ALogoutDate;

private String ALogoutTime;

public String getALoginDate() {

return ALoginDate;

}

public void setALoginDate(String aLoginDate) {

ALoginDate = aLoginDate;

}

public String getALoginTime() {

return ALoginTime;

}

public void setALoginTime(String aLoginTime) {

ALoginTime = aLoginTime;

}

public String getALogoutDate() {

return ALogoutDate;

}

public void setALogoutDate(String aLogoutDate) {

ALogoutDate = aLogoutDate;

}

public String getALogoutTime() {

return ALogoutTime;

}

public void setALogoutTime(String aLogoutTime) {

ALogoutTime = aLogoutTime;

}

}

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

public class AdminLogoutHistoryAddInfo extends JFrame

{

ArrayList<AdminLoginHistoryData> list4;

public AdminLogoutHistoryAddInfo(AdminLoginHistoryData rg1)

{

try

{

FileInputStream fin=new FileInputStream("AdminLoginData.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list4=(ArrayList<AdminLoginHistoryData>)oin.readObject();

}

catch(Exception e)

{

JOptionPane.showMessageDialog(this, "No file found in database", "Error", JOptionPane.ERROR\_MESSAGE);

list4=new ArrayList<AdminLoginHistoryData>();

}

//list4.get(i).setInitialamnt(amt);

list4.get(list4.size()-1).setALogoutDate(rg1.getALogoutDate());

list4.get(list4.size()-1).setALogoutTime(rg1.getALogoutTime());

//list4.add(rg1);

try

{

FileOutputStream fout=new FileOutputStream("AdminLoginData.dat");

ObjectOutputStream oout=new ObjectOutputStream(fout);

oout.writeObject(list4);

}catch(Exception e){}

}

}

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.util.\*;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import javax.swing.\*;

public class AdminUserRemove extends JFrame implements ActionListener

{

private JLabel l1,l2,l3;

private JTextField t1;

private JButton submit,back;

private boolean flagAccNum = false;

public AdminUserRemove()

{

super("Remove an user");

Container c=getContentPane();

c.setLayout(new GridLayout(3,2));

Font f1=new Font("Times New Roman",Font.BOLD,20);

l1=new JLabel("USER");

l1.setFont(f1);

l1.setForeground(Color.RED);

JPanel fpanel=new JPanel();

fpanel.add(l1);

fpanel.setBackground(new Color(0,0,64));

l2=new JLabel("Delete");

l2.setFont(f1);

l2.setForeground(Color.RED);

JPanel apanel=new JPanel();

apanel.add(l2);

apanel.setBackground(new Color(0,0,64));

l3=new JLabel("Give USER A/C no");

l3.setFont(f1);

l3.setForeground(Color.RED);

JPanel bpanel=new JPanel();

bpanel.add(l3);

bpanel.setBackground(new Color(0,0,64));

t1=new JTextField();

submit=new JButton("Delete");

submit.addActionListener(this);

JPanel cpanel=new JPanel();

cpanel.add(submit);

cpanel.setBackground(new Color(0,0,64));

back=new JButton("BACK");

back.addActionListener(this);

JPanel dpanel=new JPanel();

dpanel.add(back);

dpanel.setBackground(new Color(0,0,64));

c.add(fpanel);c.add(apanel);

c.add(bpanel);c.add(t1);

c.add(cpanel);c.add(dpanel);

setSize(550,300);

setLocation(200,200);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == submit)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

String vaccNum = t1.getText();

/\*for account number check

\* \*/

String accNumpattern = "^[0-9]{4}" ;

Scanner scan = new Scanner( vaccNum ) ;

String matched = scan.findInLine( accNumpattern ) ;

if ( matched == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagAccNum = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*IF VALIDATION SUCCESSFUL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagAccNum == true)

{

String sname = t1.getText().trim();

new UserRemoveDisplay(sname);

//setVisible(false);

t1.setText("");

}

flagAccNum = false;

}

if(e.getSource()==back)

{

int con=JOptionPane.showConfirmDialog(this, "Are You Sure to cancel?");

if(con==JOptionPane.YES\_OPTION)

{

new Admin();

setVisible(false);

}

}

}

}

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.\*;

import java.awt.BorderLayout;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.io.\*;

import javax.swing.\*;

public class ATMcard extends JFrame implements ActionListener

{

private JLabel l0,l1;

private JButton bsubmit,bcancel;

private JTextField t1;

private JComboBox type;

ArrayList<Create> list1;

int r,account,amnt,t,p,d=0,k,l;

private boolean flagAccNum = false;

private boolean flagAmountBefore = false;

private boolean flagDomestic = false;

private boolean flagSilverInternational = false;

private boolean flagGoldInternational = false;

private boolean flagSearchAcc = false;

public ATMcard()

{

super("ATM Card Request");

Container c=getContentPane();

c.setLayout(new GridLayout(3,2));

t1=new JTextField();

bsubmit=new JButton("SUBMIT");

bsubmit.addActionListener(this);

JPanel bpanel=new JPanel();

bpanel.add(bsubmit);

bpanel.setBackground(new Color(0,0,64));

bcancel=new JButton("CANCEL");

bcancel.addActionListener(this);

JPanel cpanel=new JPanel();

cpanel.add(bcancel);

cpanel.setBackground(new Color(0,0,64));

Font f1=new Font("comic sans ms",Font.BOLD,14);

l0=new JLabel("Enter A/C no");

l0.setFont(f1);

l0.setForeground(Color.RED);

JPanel dpanel=new JPanel();

dpanel.add(l0);

dpanel.setBackground(new Color(0,0,64));

l1=new JLabel("Enter Name");

l1.setFont(f1);

l1.setForeground(Color.RED);

JPanel epanel=new JPanel();

epanel.add(l1);

epanel.setBackground(new Color(0,0,64));

String cvalue[]={"Domestic","Silver International","Gold International"};

type=new JComboBox(cvalue);

JPanel fpanel=new JPanel();

fpanel.add(type);

fpanel.setBackground(new Color(0,0,64));

c.add(dpanel);c.add(t1);

c.add(epanel);c.add(fpanel);

c.add(bsubmit);c.add(bcancel);

setSize(400,200);

setLocation(100,100);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

r=0;

k=0;

l=0;

if(e.getSource()==bsubmit)

{

String search=t1.getText();

r=0;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

String vAccNum = t1.getText();

/\*for account number check

\* \*/

String accNumpattern = "^[0-9]{4}" ;

Scanner scan1 = new Scanner( vAccNum ) ;

String matched1 = scan1.findInLine( accNumpattern ) ;

if ( matched1 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagAccNum = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*IF VALIDATION SUCCESSFUL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagAccNum == true)

{

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

}

catch(Exception e1) {}

for(Create re : list1)

{

if(re.getAc().equals(search))

{

flagSearchAcc = true;

account=Integer.parseInt(t1.getText());

amnt=Integer.parseInt(re.getInitialamnt());

String w=(String)type.getSelectedItem();

if(amnt>500)

{

flagAmountBefore = true;

if(w.equals("Domestic"))

{

p=50;

t=amnt-p;//t=510-50 = 460

if(t>=500)

{

new Actotalcreate(account,p,d,t);

new TotalUpdate(account,t);

flagDomestic = true;

r++;

}

}

if(w.equals("Silver International"))

{

p=100;

t=amnt-p;

if(t>=500)

{

new Actotalcreate(account,p,d,t);

new TotalUpdate(account,t);

flagSilverInternational = true;

k++;

}

}

if(w.equals("Gold International"))

{

p=500;

t=amnt-p;

if(t>=500)

{

new Actotalcreate(account,p,d,t);

new TotalUpdate(account,t);

flagGoldInternational = true;

l++;

}

}

}

/\*

if((r==1)||(k==1)||(l==1))

{

JOptionPane.showMessageDialog(this, "ATM has been issued");

}

else

{

JOptionPane.showMessageDialog(this, "ATM cannot be issued");

}

\*/

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*MSG DISPLAY\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagSearchAcc == false)

{

JOptionPane.showMessageDialog(this, "Account number doest not exist.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

if((flagSearchAcc == true) && (flagAmountBefore == false))

{

JOptionPane.showMessageDialog(this, "Account balane must be above INR 500 after issuing ATM Card.", "In sufficient balance error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if((flagDomestic == true))

{

JOptionPane.showMessageDialog(this, "ATM Card successfully issued.\nYou have been charged INR 50.");

new Passdisp(t1.getText());

t1.setText("");

}

if((flagSilverInternational == true))

{

JOptionPane.showMessageDialog(this, "ATM Card successfully issued.\nYou have been charged INR 100.");

new Passdisp(t1.getText());

t1.setText("");

}

if((flagGoldInternational == true))

{

JOptionPane.showMessageDialog(this, "ATM Card successfully issued.\nYou have been charged INR 500.");

new Passdisp(t1.getText());

t1.setText("");

}

if((flagSearchAcc == true) && (flagDomestic == false) && (flagSilverInternational == false) && (flagGoldInternational == false))

{

JOptionPane.showMessageDialog(this, "ATM Card cannot be issued for insufficient balance.", "In sufficient balance error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

flagSearchAcc = false;

flagAccNum = false;

flagDomestic = false;

flagSilverInternational = false;

flagGoldInternational = false;

}

flagSearchAcc = false;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PREVIOUS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

}

catch(Exception e1) {}

for(Create re : list1)

{

if(re.getAc().equals(search))

{

account=Integer.parseInt(t1.getText());

amnt=Integer.parseInt(re.getInitialamnt());

String w=(String)type.getSelectedItem();

if(amnt>500)

{

if(w.equals("Domestic"))

{

p=50;

t=amnt-p;

if(t>=500)

{

new Actotalcreate(account,p,d,t);

new TotalUpdate(account,t);

r++;

}

}

if(w.equals("Silver International"))

{

p=100;

t=amnt-p;

if(t>=500)

{

new Actotalcreate(account,p,d,t);

new TotalUpdate(account,t);

k++;

}

}

if(w.equals("Gold International"))

{

p=500;

t=amnt-p;

if(t>=500)

{

new Actotalcreate(account,p,d,t);

new TotalUpdate(account,t);

l++;

}

}

}

if((r==1)||(k==1)||(l==1))

{

JOptionPane.showMessageDialog(this, "ATM has been issued");

}

else

{

JOptionPane.showMessageDialog(this, "ATM cannot be issued");

}

}

}

\*/

}

if(e.getSource()==bcancel)

{

int con=JOptionPane.showConfirmDialog(this, "Are you sure to cancel?");

if(con==JOptionPane.YES\_OPTION)

{

new Request();

setVisible(false);

}

}

}

}

import java.awt.Color;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import java.util.Scanner;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

public class ChangePassword extends JFrame implements ActionListener

{

private JLabel lHeading,lFirstName,lCurrentPassword,lNewPassword;

private JTextField tFirstName;

private JPasswordField pCurrentPassword,pNewPassword;

private JButton bBack,bChange;

ArrayList<OperatorRegData> list;

private boolean flagFirstName = false;

private boolean flagNewPassword = false;

private boolean flagCurrentPassword = false;

private boolean flagFirstNameMatch = false;

private boolean flagCurrentPasswordMatch = false;

private JLabel lHeading2;

public ChangePassword()

{

super("Change Password");

Container c=getContentPane();

c.setLayout(new GridLayout(5,2));

Font f1=new Font("Chiller",Font.BOLD,22);

Font f2=new Font("Times New Roman",Font.BOLD,20);

lHeading=new JLabel("CHANGE PASSWORD");

lHeading.setFont(f1);

lHeading.setForeground(Color.RED);

JPanel apanel=new JPanel();

apanel.add(lHeading);

apanel.setBackground(new Color(0,0,64));

lHeading2=new JLabel("");

lHeading2.setFont(f1);

lHeading2.setForeground(Color.RED);

JPanel abpanel=new JPanel();

abpanel.add(lHeading2);

abpanel.setBackground(new Color(0,0,64));

lFirstName=new JLabel("Enter First Name");

lFirstName.setFont(f2);

lFirstName.setForeground(Color.GRAY);

JPanel bpanel=new JPanel();

bpanel.add(lFirstName);

bpanel.setBackground(new Color(0,0,64));

lCurrentPassword=new JLabel("Enter Current Password");

lCurrentPassword.setFont(f2);

lCurrentPassword.setForeground(Color.GRAY);

JPanel cpanel=new JPanel();

cpanel.add(lCurrentPassword);

cpanel.setBackground(new Color(0,0,64));

lNewPassword=new JLabel("Enter New Password");

lNewPassword.setFont(f2);

lNewPassword.setForeground(Color.GRAY);

JPanel dpanel=new JPanel();

dpanel.add(lNewPassword);

dpanel.setBackground(new Color(0,0,64));

tFirstName=new JTextField();

pCurrentPassword=new JPasswordField();

pNewPassword=new JPasswordField();

bChange=new JButton("Change");

bChange.addActionListener(this);

JPanel epanel=new JPanel();

epanel.add(bChange);

epanel.setBackground(new Color(0,0,64));

bBack=new JButton("Back");

bBack.addActionListener(this);

JPanel fpanel=new JPanel();

fpanel.add(bBack);

fpanel.setBackground(new Color(0,0,64));

c.add(apanel);c.add(abpanel);

c.add(bpanel); c.add(tFirstName);

c.add(cpanel);c.add(pCurrentPassword);

c.add(dpanel); c.add(pNewPassword);

c.add(epanel); c.add(fpanel);

setSize(500,500);

setLocation(100,100);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == bChange)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

String vFirstName = tFirstName.getText();

String vCurrentPassword = pCurrentPassword.getText();

String vNewPassword = pNewPassword.getText();

/\*for first name check

\* \*/

String firstNamepattern = "^[a-zA-Z]" ;

Scanner scan1 = new Scanner( vFirstName ) ;

String matched1 = scan1.findInLine( firstNamepattern ) ;

if ( matched1 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabets.", "Error", JOptionPane.ERROR\_MESSAGE);

tFirstName.setText("");

}

else

{

flagFirstName = true;

}

/\*for current password check

\* \*/

String currentPasswordpattern = "^[0-9]{4}" ;

Scanner scan2 = new Scanner( vCurrentPassword ) ;

String matched2 = scan2.findInLine( currentPasswordpattern ) ;

if ( matched2 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

pCurrentPassword.setText("");

}

else

{

flagCurrentPassword = true;

}

/\*for new password check

\* \*/

String newPasswordpattern = "^[0-9]{4]" ;

Scanner scan3 = new Scanner( vNewPassword) ;

String matched3 = scan3.findInLine( currentPasswordpattern ) ;

if ( matched3 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID NEW PASSWORD\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

pNewPassword.setText("");

}

else

{

flagNewPassword = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*IF VALIDATION SUCCESSFUL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagFirstName == true)

{

if(flagCurrentPassword == true)

{

if(flagNewPassword == true)

{

try

{

FileInputStream fin=new FileInputStream("Reg.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list=(ArrayList<OperatorRegData>)oin.readObject();

}

catch(Exception e1)

{

JOptionPane.showMessageDialog(this, "No file found in database", "Error", JOptionPane.ERROR\_MESSAGE);

list=new ArrayList<OperatorRegData>();

}

for(int i=0;i<list.size();i++)//for(OperatorRegData element : list)

{

if(list.get(i).getName().equals(tFirstName.getText()))

{

flagFirstNameMatch = true;

if(list.get(i).getPassword().equals(pCurrentPassword.getText()))

{

flagCurrentPasswordMatch = true;

list.get(i).setPassword(pNewPassword.getText());

}

}

}

try

{

FileOutputStream fout=new FileOutputStream("Reg.dat");

ObjectOutputStream oout=new ObjectOutputStream(fout);

oout.writeObject(list);

}catch(Exception e1){}

if((flagFirstNameMatch == true) && (flagCurrentPasswordMatch == true))

{

JOptionPane.showMessageDialog(this, "Password has been successfully changed");

tFirstName.setText("");

pCurrentPassword.setText("");

pNewPassword.setText("");

}

if((flagFirstNameMatch == true) && (flagCurrentPasswordMatch == false))

{

JOptionPane.showMessageDialog(this, "Password incorrect", "Error", JOptionPane.ERROR\_MESSAGE);

pCurrentPassword.setText("");

}

if((flagFirstNameMatch == false) && (flagCurrentPasswordMatch == false))

{

JOptionPane.showMessageDialog(this, "Operator does not exist.", "Error", JOptionPane.ERROR\_MESSAGE);

tFirstName.setText("");

pCurrentPassword.setText("");

pNewPassword.setText("");

}

}

}

}

flagFirstName = false;

flagCurrentPassword = false;

flagNewPassword = false;

flagFirstNameMatch = false;

flagCurrentPasswordMatch = false;

}

if(e.getSource()==bBack)

{

int rply = JOptionPane.showConfirmDialog(this, "Are you sure to quit?");

if(rply == JOptionPane.YES\_OPTION)

{

new Userwindow("");

setVisible(false);

}

}

}

}

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.\*;

import java.awt.BorderLayout;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.io.\*;

import javax.swing.\*;

public class ChequeBook extends JFrame implements ActionListener

{

private JComboBox pages;

private JTextField t1;

private JLabel l0,l1;

private JButton bsubmit,bcancel;

ArrayList<Create> list1;

private int amnt,account,d,t,r,w1,tot1,tot2,v,c;

private boolean flagAccNum = false;

public ChequeBook()

{

super("Cheque Book Request");

Container c=getContentPane();

t1=new JTextField();

c.setLayout(new GridLayout(3,2));

bsubmit=new JButton("Submit");

bsubmit.addActionListener(this);

JPanel fpanel=new JPanel();

fpanel.add(bsubmit);

fpanel.setBackground(new Color(0,0,64));

bcancel=new JButton("Cancel");

bcancel.addActionListener(this);

JPanel gpanel=new JPanel();

gpanel.add(bcancel);

gpanel.setBackground(new Color(0,0,64));

Font f1=new Font("Times New Roman",Font.BOLD,20);

l0=new JLabel("Enter A/C no");

l0.setFont(f1);

l0.setForeground(Color.RED);

JPanel apanel=new JPanel();

apanel.add(l0);

apanel.setBackground(new Color(0,0,64));

l1=new JLabel("Enter Number of Pages");

l1.setFont(f1);

l1.setForeground(Color.RED);

JPanel bpanel=new JPanel();

bpanel.add(l1);

bpanel.setBackground(new Color(0,0,64));

String cvalue[]={"25","50"};

pages=new JComboBox(cvalue);

JPanel cpanel=new JPanel();

cpanel.add(pages);

cpanel.setBackground(new Color(0,0,64));

c.add(apanel);c.add(t1);

c.add(bpanel);c.add(cpanel);

c.add(fpanel);c.add(gpanel);

setSize(400,200);

setLocation(100,100);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

d=0;

r=0;

c=0;

String search=t1.getText();

//r=0;

if(e.getSource()==bsubmit)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

String vAccNum = t1.getText();

/\*for account number check

\* \*/

String accNumpattern = "^[0-9]{4}" ;

Scanner scan1 = new Scanner( vAccNum ) ;

String matched1 = scan1.findInLine( accNumpattern ) ;

if ( matched1 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagAccNum = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*IF VALIDATION SUCCESSFUL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagAccNum == true)

{

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

}

catch(Exception e1){}

for(Create re : list1)

{

if(re.getAc().equals(search))

{

account=Integer.parseInt(t1.getText());

amnt=Integer.parseInt(re.getInitialamnt());

String w=(String)pages.getSelectedItem();

w1=Integer.parseInt(w);

if(amnt>500)

{

if(w1==25)

{

v=50;

tot1=amnt-v;

if(tot1>=500)

{

new Actotalcreate(account,v,d,tot1);

new TotalUpdate(account,tot1);

r++;

}

}

if(w1==50)

{

v=100;

tot2=amnt-v;

if(tot2>=500)

{

new Actotalcreate(account,v,d,tot2);

new TotalUpdate(account,tot2);

c++;

}

}

}

}

}

if((r==1)||(c==1))

{

JOptionPane.showMessageDialog(this, "Check book has been issued");

new Passdisp(t1.getText());

t1.setText("");

}

else

{

JOptionPane.showMessageDialog(this, "Check cannot be issued");

t1.setText("");

}

}

flagAccNum = false;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PREVIOUS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

}

catch(Exception e1){}

for(Create re : list1)

{

if(re.getAc().equals(search))

{

account=Integer.parseInt(t1.getText());

amnt=Integer.parseInt(re.getInitialamnt());

String w=(String)pages.getSelectedItem();

w1=Integer.parseInt(w);

if(amnt>500)

{

if(w1==25)

{

v=50;

tot1=amnt-v;

if(tot1>=500)

{

new Actotalcreate(account,v,d,tot1);

new TotalUpdate(account,tot1);

r++;

}

}

if(w1==50)

{

v=100;

tot2=amnt-v;

if(tot2>=500)

{

new Actotalcreate(account,v,d,tot2);

new TotalUpdate(account,tot2);

c++;

}

}

}

}

}

if((r==1)||(c==1))

{

JOptionPane.showMessageDialog(this, "Check book has been issued");

}

else

{

JOptionPane.showMessageDialog(this, "Check cannot be issued");

}

\*/

}

if(e.getSource()==bcancel)

{

int con=JOptionPane.showConfirmDialog(this, "Are you sure to cancel?");

if(con==JOptionPane.YES\_OPTION)

{

new Request();

//new UserMain1();

setVisible(false);

}

}

}

}

import java.awt.Container;

import java.awt.GridLayout;

import java.awt.Color;

import java.util.\*;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.event.ActionListener; //to sense mouse click

import java.awt.event.ActionEvent;

import java.io.Serializable;

import java.util.Calendar;

import javax.swing.\*;

public class Create implements Serializable

{

private String ac;

public String getAc() {

return ac;

}

public void setAc(String ac) {

this.ac = ac;

}

private String name;

private String name1;

private String address;

private String email;

private String nationality;

private String acctype;

private String city;

private String gender;

private String dob;

private String date;

private String time;

private String Identity;

private String profession;

private String initialamnt;

public String getInitialamnt() {

return initialamnt;

}

public void setInitialamnt(String initialamnt) {

this.initialamnt = initialamnt;

}

public String getNationality() {

return nationality;

}

public void setNationality(String nationality) {

this.nationality = nationality;

}

public String getAcctype() {

return acctype;

}

public void setAcctype(String acctype) {

this.acctype = acctype;

}

public String getIdentity() {

return Identity;

}

public void setIdentity(String identity) {

Identity = identity;

}

public String getProfession() {

return profession;

}

public void setProfession(String profession) {

this.profession = profession;

}

public String getName1()

{

return name1;

}

public void setName1(String name1)

{

this.name1 = name1;

}

public String getEmail()

{

return email;

}

public void setEmail(String email)

{

this.email = email;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getDob() {

return dob;

}

public void setDob(String dob) {

this.dob = dob;

}

public String getDate() {

return date;

}

public void setDate(String date) {

this.date = date;

}

public String getTime() {

return time;

}

public void setTime(String time) {

this.time = time;

}

}

import java.awt.Color;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.FileInputStream;

import java.io.ObjectInputStream;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Scanner;

import javax.swing.\*;

public class CreateFrame extends JFrame implements ActionListener

{

private JLabel l0,l01,l1,l2,l3,l4,l40,l5,l6,l7,l8,l9,l10,l11,l12,l13,l14,l15,l16,l17,l18,l19,l20,l21;

private JTextField t1,tid1,tid2,tid3,tid4,tid5,tid6,tid7,tid8,tid10;

private JComboBox city,day,month,year,annualincome,identity,profession,acctype,acno;

private JRadioButton rmale,rfemale,rsav,rothers,married,unmarried;

private JButton bsubmit,back;

private String sysdate,systime;

//private ArrayList<Create> alist = new ArrayList<Create>();

private boolean flagName = false;

private boolean flagName1 = false;

private boolean flagAddress = false;

private boolean flagNationality = false;

private ArrayList<Create> alist;

private boolean flagAccountNumber = false;

private int count = 0;

private String ini;

private boolean flagInitialAmount = false;

private boolean flagLandline = false;

private boolean flagMobile = false;

private boolean flagAge = false;

private boolean flagProof = false;

CreateFrame()

{

super("User registration form");

Container c=getContentPane();

c.setLayout(new GridLayout(21,2));

tid1=new JTextField();

tid2=new JTextField();

tid3=new JTextField();

tid4=new JTextField();

tid5=new JTextField();

tid6=new JTextField();

tid7=new JTextField();

tid8=new JTextField();

//tid9=new JTextField();

tid10=new JTextField();

String cvalue[]={"<50000","50000-100000","100000-600000",">600000"};

annualincome=new JComboBox(cvalue);

String d1value[]={"Government Service","Business","Private Sector","Student","Others"};

profession=new JComboBox(d1value);

String evalue[]={"Pan Card","Voter Card","Ration Card","Driving License"};

identity=new JComboBox(evalue);

rsav=new JRadioButton("Savings");

rothers=new JRadioButton("Savings");

ButtonGroup r1group=new ButtonGroup();

r1group.add(rsav);

JPanel kpanel=new JPanel();

kpanel.add(rsav);

rmale=new JRadioButton("Male");

rfemale=new JRadioButton("Female");

ButtonGroup rgroup=new ButtonGroup();

rgroup.add(rmale);

rgroup.add(rfemale);

JPanel gpanel=new JPanel();

married=new JRadioButton("Married");

unmarried=new JRadioButton("Single");

ButtonGroup r2group=new ButtonGroup();

r2group.add(married);

r2group.add(unmarried);

JPanel gopanel=new JPanel();

gopanel.add(married);

gopanel.add(unmarried);

gpanel.add(rmale);

gpanel.add(rfemale);

t1=new JTextField();

String dvalue[]=new String[31];

for(int i=0;i<=30;i++)

{

dvalue[i]=String.valueOf(i+1);

}

day=new JComboBox(dvalue);

String mvalue[]=new String[12];

for(int i=0;i<=11;i++)

{

mvalue[i]=String.valueOf(i+1);

}

month=new JComboBox(mvalue);

String yvalue[]=new String[25];

int cnt=0;

for(int i=1989;i<=2013;i++)

{

yvalue[cnt]=String.valueOf(i);

cnt++;

}

year=new JComboBox(yvalue);

JPanel cpanel=new JPanel();

cpanel.add(day);

cpanel.add(month);

cpanel.add(year);

bsubmit=new JButton("Register");

bsubmit.addActionListener(this);

Font f1=new Font("Times New Roman",Font.BOLD,14);

l01=new JLabel(" ACCOUNT OPENING");

l01.setFont(f1);

l01.setForeground(Color.RED);

JPanel fpanel=new JPanel();

fpanel.add(l01);

fpanel.setBackground(new Color(0,0,64));

l0=new JLabel("FORM ");

l0.setFont(f1);

l0.setForeground(Color.RED);

JPanel apanel=new JPanel();

apanel.add(l0);

apanel.setBackground(new Color(0,0,64));

//Font f1=new Font("Times New Roman",Font.BOLD,14);

l1=new JLabel("Enter First Name");

l1.setFont(f1);

l1.setForeground(Color.RED);

JPanel bpanel=new JPanel();

bpanel.add(l1);

bpanel.setBackground(new Color(0,0,64));

l2=new JLabel("Enter Last Name");

l2.setFont(f1);

l2.setForeground(Color.RED);

JPanel copanel=new JPanel();

copanel.add(l2);

copanel.setBackground(new Color(0,0,64));

l3=new JLabel("Enter Address");

l3.setFont(f1);

l3.setForeground(Color.RED);

JPanel dpanel=new JPanel();

dpanel.add(l3);

dpanel.setBackground(new Color(0,0,64));

l4=new JLabel("Enter Phone Number(Landline)");

l4.setFont(f1);

l4.setForeground(Color.RED);

JPanel epanel=new JPanel();

epanel.add(l4);

epanel.setBackground(new Color(0,0,64));

l40=new JLabel("Enter Phone Number(Mobile)");

l40.setFont(f1);

l40.setForeground(Color.RED);

JPanel vpanel=new JPanel();

vpanel.add(l40);

vpanel.setBackground(new Color(0,0,64));

l5=new JLabel("Select Sex");

l5.setFont(f1);

l5.setForeground(Color.RED);

JPanel qpanel=new JPanel();

qpanel.add(l5);

qpanel.setBackground(new Color(0,0,64));

l6=new JLabel("Natinality");

l6.setFont(f1);

l6.setForeground(Color.RED);

JPanel wpanel=new JPanel();

wpanel.add(l6);

wpanel.setBackground(new Color(0,0,64));

l7=new JLabel("Select Age");

l7.setFont(f1);

l7.setForeground(Color.RED);

JPanel opanel=new JPanel();

opanel.add(l7);

opanel.setBackground(new Color(0,0,64));

l8=new JLabel("Select A/C Type");

l8.setFont(f1);

l8.setForeground(Color.RED);

JPanel zpanel=new JPanel();

zpanel.add(l8);

zpanel.setBackground(new Color(0,0,64));

l9=new JLabel("Date of Birth");

l9.setFont(f1);

l9.setForeground(Color.RED);

JPanel xpanel=new JPanel();

xpanel.add(l9);

xpanel.setBackground(new Color(0,0,64));

l10=new JLabel("Enter Annual Income");

l10.setFont(f1);

l10.setForeground(Color.RED);

JPanel vxpanel=new JPanel();

vxpanel.add(l10);

vxpanel.setBackground(new Color(0,0,64));

l11=new JLabel("Enter Profession");

l11.setFont(f1);

l11.setForeground(Color.RED);

JPanel vppanel=new JPanel();

vppanel.add(l11);

vppanel.setBackground(new Color(0,0,64));

l12=new JLabel("Enter Marital Status");

l12.setFont(f1);

l12.setForeground(Color.RED);

JPanel npanel=new JPanel();

npanel.add(l12);

npanel.setBackground(new Color(0,0,64));

l13=new JLabel("Enter Identity Proof");

l13.setFont(f1);

l13.setForeground(Color.RED);

JPanel mpanel=new JPanel();

mpanel.add(l13);

mpanel.setBackground(new Color(0,0,64));

l14=new JLabel("Enter Identity proof no:");

l14.setFont(f1);

l14.setForeground(Color.RED);

JPanel jpanel=new JPanel();

jpanel.add(l14);

jpanel.setBackground(new Color(0,0,64));

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*extra\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*l15=new JLabel("Enter Initial Amount");

l15.setFont(f1);

l15.setForeground(Color.RED);

JPanel bcpanel=new JPanel();

bcpanel.add(l15);

bcpanel.setBackground(new Color(0,0,64));\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Calendar cal = Calendar.getInstance();

String cday = ""+cal.get(Calendar.DATE);

int x =cal.get(Calendar.MONTH);

String cmonth = ""+(x+1);

String cyear =""+cal.get(Calendar.YEAR);

sysdate = cday+"/"+cmonth+"/"+cyear;

l16=new JLabel(sysdate);

String chr = ""+cal.get(Calendar.HOUR\_OF\_DAY);

String cmin = ""+cal.get(Calendar.MINUTE);

String csec =""+cal.get(Calendar.SECOND);

systime = chr+":"+cmin+":"+csec;

l17=new JLabel(systime);

l18=new JLabel("Current Time");

l18.setFont(f1);

l18.setForeground(Color.RED);

JPanel vopanel=new JPanel();

vopanel.add(l18);

vopanel.setBackground(new Color(0,0,64));

l19=new JLabel("Current Date");

l19.setFont(f1);

l19.setForeground(Color.RED);

JPanel vipanel=new JPanel();

vipanel.add(l19);

vipanel.setBackground(new Color(0,0,64));

l20=new JLabel("Give a 4-digit Account number");

l20.setFont(f1);

l20.setForeground(Color.RED);

JPanel poppanel=new JPanel();

poppanel.add(l20);

poppanel.setBackground(new Color(0,0,64));

l20=new JLabel("Initial Amount");

l20.setFont(f1);

l20.setForeground(Color.RED);

JPanel kop=new JPanel();

kop.add(l20);

kop.setBackground(new Color(0,0,64));

bsubmit=new JButton("SUBMIT");

bsubmit.addActionListener(this);

JPanel dopanel=new JPanel();

dopanel.add(bsubmit);

dopanel.setBackground(new Color(0,0,64));

back=new JButton("BACK");

back.addActionListener(this);

JPanel dipanel=new JPanel();

dipanel.add(back);

dipanel.setBackground(new Color(0,0,64));

c.add(fpanel);c.add(apanel);

c.add(bpanel);c.add(tid1);

c.add(copanel);c.add(tid2);

c.add(dpanel);c.add(tid3);

c.add(epanel);c.add(tid4);

c.add(vpanel);c.add(tid5);

c.add(qpanel);c.add(gpanel);

c.add(wpanel);c.add(tid6);

c.add(opanel);c.add(tid7);

c.add(zpanel);c.add(kpanel);

c.add(xpanel);c.add(cpanel);

c.add(vxpanel);c.add(annualincome);

c.add(vppanel);c.add(profession);

c.add(npanel);c.add(gopanel);

c.add(mpanel);c.add(identity);

c.add(jpanel);c.add(tid8);

//c.add(bcpanel);c.add(tid9);

c.add(poppanel);c.add(t1);

c.add(kop);c.add(tid10);

c.add(vipanel);c.add(l16);

c.add(vopanel);c.add(l17);

c.add(dopanel);c.add(dipanel);

setSize(600,625);

setLocation(200,200);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

String name;//first name

String name1;//last name

String address;

String nationality;

String acc;//account type

String gn;

String dob;

String date;

String time;

String Iden;

String prof;

String inti;

String ac1;

String land;

String mob;

String age;

String proof;

Create reg;

if(e.getSource()==bsubmit)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*VALIDATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*/

name=tid1.getText();

name1=tid2.getText();

address=tid3.getText();

nationality=tid6.getText();

ac1=t1.getText();

ini = tid10.getText();

land = tid4.getText();

mob = tid5.getText();

age = tid7.getText();

proof = tid8.getText();

/\* user first name check

\* \*/

String namepattern = "^[A-Za-z]";

Scanner scan = new Scanner( name ) ;

String matched = scan.findInLine( namepattern ) ;

if ( matched == null )

{

//JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabet.");

JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabet.", "Error", JOptionPane.ERROR\_MESSAGE);

tid1.setText("");

}

else

{

flagName = true;

}

/\* user last name check

\* \*/

String namepattern1 = "^[A-Za-z]";

Scanner scan1 = new Scanner( name1 ) ;

String matched1 = scan1.findInLine( namepattern1 ) ;

if ( matched1 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabet.");

JOptionPane.showMessageDialog(this, "INVALID LAST NAME\nIt should contain only alphabet.", "Error", JOptionPane.ERROR\_MESSAGE);

tid2.setText("");

}

else

{

flagName1 = true;

}

/\* user address check

\* \*/

String namepattern2 = "^[A-Za-z0-9]{1}";

Scanner scan2 = new Scanner( address ) ;

String matched2 = scan2.findInLine( namepattern2 ) ;

if ( matched2 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabet.");

JOptionPane.showMessageDialog(this, "INVALID ADDRESS\nIT SHOLUD NOT BLANK.", "Error", JOptionPane.ERROR\_MESSAGE);

tid3.setText("");

}

else

{

flagAddress = true;

}

/\*for landline numbercheck

\* \*/

String landlinepattern = "^[0-9]{8}" ;

Scanner scan6 = new Scanner( land ) ;

String matched6 = scan6.findInLine( landlinepattern ) ;

if ( matched6 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID LANDLINE NUMBER\nIt should contain only digits with minimum length of 8 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

tid4.setText("");

}

else

{

flagLandline = true;

}

/\*for mobile numbercheck

\* \*/

String mobpattern = "^[0-9]{10}" ;

Scanner scan7 = new Scanner( mob ) ;

String matched7 = scan7.findInLine( mobpattern ) ;

if ( matched7 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID MOBILE NUMBER\nIt should contain only digits with minimum length of 10 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

tid5.setText("");

}

else

{

flagMobile = true;

}

/\* user nationality check

\* \*/

String namepattern3 = "^[A-Za-z]";

Scanner scan3 = new Scanner( nationality ) ;

String matched3 = scan3.findInLine( namepattern3 ) ;

if ( matched3 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabet.");

JOptionPane.showMessageDialog(this, "INVALID NATIONALITY\nIt should contain only alphabet.", "Error", JOptionPane.ERROR\_MESSAGE);

tid6.setText("");

}

else

{

flagNationality = true;

}

/\*for agecheck

\* \*/

String agepattern = "^[0-9]{2}" ;

Scanner scan8 = new Scanner( age ) ;

String matched8 = scan8.findInLine( agepattern ) ;

if ( matched8 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID AGE\nAge must be minimum 10 years & it should contain only digit.", "Error", JOptionPane.ERROR\_MESSAGE);

tid7.setText("");

}

else

{

flagAge = true;

}

/\* user proof check

\* \*/

String proofpattern2 = "^[A-Za-z0-9]{1}";

Scanner scan9 = new Scanner( proof ) ;

String matched9 = scan9.findInLine( proofpattern2 ) ;

if ( matched9 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID FIRST NAME\nIt should contain only alphabet.");

JOptionPane.showMessageDialog(this, "INVALID IDENTITY PROOF NUMBER\nIT SHOLUD NOT BLANK.", "Error", JOptionPane.ERROR\_MESSAGE);

tid8.setText("");

}

else

{

flagProof = true;

}

/\*for account number check

\* \*/

String accpattern = "^[0-9]{4}" ;

Scanner scan4 = new Scanner( ac1 ) ;

String matched4 = scan4.findInLine( accpattern ) ;

if ( matched4 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagAccountNumber = true;

}

/\*for initial amount check

\* \*/

String inipattern = "^[0-9]{3}" ;

Scanner scan5 = new Scanner( ini ) ;

String matched5 = scan5.findInLine( inipattern ) ;

if ( matched5 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID INITIAL AMOUNT\nIt should contain only digits with minimum length of 3 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

tid10.setText("");

}

else

{

flagInitialAmount = true;

}

if(flagName == true)

{

if(flagName1 == true)

{

if(flagAddress == true)

{

if(flagNationality == true)

{

if(flagAccountNumber == true)

{

if(flagInitialAmount == true)

{

if(flagLandline == true)

{

if(flagMobile == true)

{

if(flagAge == true)

{

if(flagProof == true)

{

/\*\*\*\*\*\*\*\*\*\*read from file and store in alist\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

alist=(ArrayList<Create>)oin.readObject();

}

catch(Exception ex)

{

alist=new ArrayList<Create>();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*operation\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//it checks account numebr is already present or not

for(Create element : alist)

{

if(element.getAc().equals(t1.getText()))

{

count++;

}

}

if(count > 0)//if present

{

JOptionPane.showMessageDialog(this, "Account numner is already registered.\nTry another one", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else//if not present

{

/\*\*\*\*\*\*\*\*\*\*\*\*variable setting\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*check the minimum initial amount\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(Integer.parseInt(ini) >= 500)

{

inti=tid10.getText();

name=tid1.getText();

name1=tid2.getText();

address=tid3.getText();

nationality=tid6.getText();

ac1=t1.getText();

Iden=(String)identity.getSelectedItem();

prof=(String)profession.getSelectedItem();

acc="";

if(rsav.isSelected())

{

acc="Savings";

}

else if(rothers.isSelected())

{

acc="Others";

}

gn="";

if(rmale.isSelected())

{

gn="Male";

}

else if(rfemale.isSelected())

{

gn="Female";

}

String d=(String)day.getSelectedItem();

String m=(String)month.getSelectedItem();

String y=(String)year.getSelectedItem();

dob=d + "-" + m + "-" + y;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*packing variable into object\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

reg=new Create();

reg.setName(name);

reg.setName1(name1);

reg.setAddress(address);

reg.setNationality(nationality);

reg.setAcctype(acc);

reg.setGender(gn);

reg.setDob(dob);

reg.setIdentity(Iden);

reg.setProfession(prof);

reg.setAc(ac1);

reg.setInitialamnt(inti);

reg.setDate(l16.getText());

reg.setTime(l17.getText());

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int con=JOptionPane.showConfirmDialog(this, "Are You Sure to Register?");

if(con==JOptionPane.YES\_OPTION)

{

new AddInformation1(reg);

new Actotalcreate(Integer.parseInt(t1.getText()),0,Integer.parseInt(tid10.getText()),Integer.parseInt(tid10.getText()));//acc ---> Integer.parseInt(t1.getText()), withdrw -----> 0,deposite --------> Integer.parseInt(tid10.getText()),total --------> Integer.parseInt(tid10.getText()) (all are int)

new TotalUpdate(Integer.parseInt(t1.getText()),Integer.parseInt(tid10.getText()));

//setVisible(false);

JOptionPane.showMessageDialog(this, "Successfully registered");

new Individualpassbook(ac1);

tid10.setText("");

tid1.setText("");

tid2.setText("");

tid3.setText("");

tid6.setText("");

t1.setText("");

tid4.setText("");

tid5.setText("");

tid7.setText("");

tid8.setText("");

}

}

else

{

JOptionPane.showMessageDialog(this, "Minimum initial amount should be 500", "Error", JOptionPane.ERROR\_MESSAGE);

tid10.setText("");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

inti=tid10.getText();

name=tid1.getText();

name1=tid2.getText();

address=tid3.getText();

nationality=tid6.getText();

ac1=t1.getText();

Iden=(String)identity.getSelectedItem();

prof=(String)profession.getSelectedItem();

acc="";

if(rsav.isSelected())

{

acc="Savings";

}

else if(rothers.isSelected())

{

acc="Others";

}

gn="";

if(rmale.isSelected())

{

gn="Male";

}

else if(rfemale.isSelected())

{

gn="Female";

}

String d=(String)day.getSelectedItem();

String m=(String)month.getSelectedItem();

String y=(String)year.getSelectedItem();

dob=d + "-" + m + "-" + y;

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*packing variable into object\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

reg=new Create();

reg.setName(name);

reg.setName1(name1);

reg.setAddress(address);

reg.setNationality(nationality);

reg.setAcctype(acc);

reg.setGender(gn);

reg.setDob(dob);

reg.setIdentity(Iden);

reg.setProfession(prof);

reg.setAc(ac1);

reg.setInitialamnt(inti);

reg.setDate(l16.getText());

reg.setTime(l17.getText());

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

int con=JOptionPane.showConfirmDialog(this, "Are You Sure to Register?");

if(con==JOptionPane.YES\_OPTION)

{

new AddInformation1(reg);

//setVisible(false);

JOptionPane.showMessageDialog(this, "Successfully registered");

tid10.setText("");

tid1.setText("");

tid2.setText("");

tid3.setText("");

tid6.setText("");

t1.setText("");

}

\*/

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

}

}

}

}

}

}

}

}

}

flagName = false;

flagName1 = false;

flagAddress = false;

flagNationality = false;

flagAccountNumber = false;

flagInitialAmount = false;

count =0;

flagLandline = false;

flagMobile = false;

flagAge = false;

flagProof = false;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*previous\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*inti=tid10.getText();

name=tid1.getText();

name1=tid2.getText();

address=tid3.getText();

nationality=tid6.getText();

ac1=t1.getText();

Iden=(String)identity.getSelectedItem();

prof=(String)profession.getSelectedItem();

acc="";

if(rsav.isSelected())

{

acc="Savings";

}

else if(rothers.isSelected())

{

acc="Others";

}

gn="";

if(rmale.isSelected())

{

gn="Male";

}

else if(rfemale.isSelected())

{

gn="Female";

}

String d=(String)day.getSelectedItem();

String m=(String)month.getSelectedItem();

String y=(String)year.getSelectedItem();

dob=d + "-" + m + "-" + y;

reg=new Create();

reg.setName(name);

reg.setName1(name1);

reg.setAddress(address);

reg.setNationality(nationality);

reg.setAcctype(acc);

reg.setGender(gn);

reg.setDob(dob);

reg.setIdentity(Iden);

reg.setProfession(prof);

reg.setAc(ac1);

reg.setInitialamnt(inti);

reg.setDate(l16.getText());

reg.setTime(l17.getText());

int con=JOptionPane.showConfirmDialog(this, "Are You Sure to Register?");

if(con==JOptionPane.YES\_OPTION)

{

new AddInformation1(reg);

setVisible(false);

}\*/

}

if(e.getSource()==back)

{

int rply = JOptionPane.showConfirmDialog(this, "Are you sure to quit?");

if(rply == JOptionPane.YES\_OPTION)

{

new Userwindow("");

setVisible(false);

}

}

}

}

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.Color;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.\*;

import java.awt.BorderLayout;

import java.awt.Container;

import java.awt.Font;

import java.awt.GridLayout;

import java.io.\*;

import javax.swing.\*;

public class Withdraw extends JFrame implements ActionListener

{

private JLabel l1,l2,l3,l4;

private JTextField t1,t2;

private JButton withdraw,cancel,deposit;

private int x,amnt,t,w,account,d,r=0;

ArrayList<Create> list1;

private boolean flagWithdraw = false;

private boolean flagAccNum = false;

private boolean flagYes = false;

private boolean flagAccMatch = false;

private boolean flagNo = false;

public Withdraw()

{

super("DEPOSIT/WITHDRAWAL SLIP");

Container c=getContentPane();

c.setLayout(new GridLayout(4,2));

Font f1=new Font("Times New Roman",Font.BOLD,20);

/\*l1=new JLabel(" WITHDRAWAL");

l1.setFont(f1);

l1.setForeground(Color.GRAY);

JPanel fpanel=new JPanel();

fpanel.add(l1);

fpanel.setBackground(new Color(0,0,64));

l1=new JLabel("SLIP ");

l1.setFont(f1);

l1.setForeground(Color.GRAY);

JPanel apanel=new JPanel();

apanel.add(l1);

apanel.setBackground(new Color(0,0,64));\*/

l1=new JLabel("Enter ACCOUNT NO:");

l1.setFont(f1);

l1.setForeground(Color.GRAY);

JPanel bpanel=new JPanel();

bpanel.add(l1);

bpanel.setBackground(new Color(0,0,64));

l2=new JLabel("ENTER AMOUNT:");

l2.setFont(f1);

l2.setForeground(Color.GRAY);

JPanel b1panel=new JPanel();

b1panel.add(l2);

b1panel.setBackground(new Color(0,0,64));

l3=new JLabel("DEPOSIT:");

l3.setFont(f1);

l3.setForeground(Color.GRAY);

JPanel b2panel=new JPanel();

b2panel.add(l3);

b2panel.setBackground(new Color(0,0,64));

t1=new JTextField();

t2=new JTextField();

withdraw=new JButton("WITHDRAW");

withdraw.addActionListener(this);

JPanel dpanel=new JPanel();

dpanel.add(withdraw);

dpanel.setBackground(new Color(0,0,64));

deposit=new JButton("DEPOSIT");

deposit.addActionListener(this);

JPanel d1panel=new JPanel();

d1panel.add(deposit);

d1panel.setBackground(new Color(0,0,64));

cancel=new JButton("CANCEL");

cancel.addActionListener(this);

JPanel epanel=new JPanel();

epanel.add(cancel);

epanel.setBackground(new Color(0,0,64));

l4=new JLabel("");

l4.setFont(f1);

l4.setForeground(Color.GRAY);

JPanel b3panel=new JPanel();

b3panel.add(l4);

b3panel.setBackground(new Color(0,0,64));

c.add(bpanel);c.add(t1);

c.add(b1panel);c.add(t2);

c.add(dpanel);c.add(d1panel);

c.add(epanel);c.add(b3panel);

setSize(500,500);

setLocation(100,100);

setResizable(false);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

//Create rg1=new Create();

//String data[][];

//String heading[]={"First Name","Last name","Address","Nationality","Account Type","Date Of Birth","Identity","Profession","Reg Date","Reg Time","A/c no","Initial amnt"};

String search=t1.getText();//search ------->account number

if(e.getSource()==withdraw)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

String vaccNum = t1.getText();

String vwithdrw = t2.getText();

/\*for account number check

\* \*/

String accNumpattern = "^[0-9]{4}" ;

Scanner scan1 = new Scanner( vaccNum ) ;

String matched1 = scan1.findInLine( accNumpattern ) ;

if ( matched1 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagAccNum = true;

}

/\*for withdraw amount check

\* \*/

String withdrawpattern = "^[0-9]{3}" ;

Scanner scan = new Scanner( vwithdrw ) ;

String matched = scan.findInLine( withdrawpattern ) ;

if ( matched == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID WITHDRAW AMOUNT\nIt should contain only digits with minimum withdraw amount of INR 100.", "Error", JOptionPane.ERROR\_MESSAGE);

t2.setText("");

}

else

{

flagWithdraw = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*IF VALIDATION SUCCESSFUL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagAccNum == true)

{

if(flagWithdraw == true)

{

try//read from file

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

//reading all account withdraw info

}

catch(Exception e1)// if file is not present in first time

{

JOptionPane.showMessageDialog(this, "No file found in data base", "Error", JOptionPane.ERROR\_MESSAGE);

}

/\*\*\*sukanta\*\*\*\*\*\*\*\*\*\*\*\*/

//x = list1.size();

for(Create re : list1)

{

if(re.getAc().equals(search))

{

flagAccMatch = true;

w=Integer.parseInt(t2.getText());

amnt=Integer.parseInt(re.getInitialamnt());

if(amnt>=500)

{

flagNo = true;

t = amnt - w;

if(t >= 500)

{

account=Integer.parseInt(t1.getText());

d=0;

//r++;

flagYes = true;

}

/\*

if(amnt>w)

{

t=(amnt-w);

account=Integer.parseInt(t1.getText());

d=0;

r++;

}

else

{

r=0;

JOptionPane.showMessageDialog(this, "Insufficient Balance", "Error", JOptionPane.ERROR\_MESSAGE);

}

\*/

}

}

}

/\*\*\*\*\*\*\*\*msg display and argu pass\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if((flagAccMatch == true) && (flagNo == true) && (flagYes == true))

{

new Actotalcreate(account,w,d,t);//acc ---> account, withdrw -----> w,deposite --------> d,total --------> t (all are int)

new TotalUpdate(account,t);

JOptionPane.showMessageDialog(this, "Withdraw done successfully");

new Passdisp(t1.getText());

t1.setText("");

t2.setText("");

}

if((flagAccMatch == true) && (flagNo == true) && (flagYes == false))

{

JOptionPane.showMessageDialog(this, "Insufficient balance.\nAccount balance must be 500 after withdraw.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

t2.setText("");

}

if((flagAccMatch == true) && (flagNo == false) && (flagYes == false))

{

JOptionPane.showMessageDialog(this, "Account number has not the minimum balance of INR 500.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

t2.setText("");

}

if((flagAccMatch == false) && (flagNo == false) && (flagYes == false))

{

JOptionPane.showMessageDialog(this, "Account number does not exists", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

t2.setText("");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

flagYes = false;

flagAccMatch = false;

flagNo = false;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*mmmmmmmmmyyyyyyyyyyyyyyyyyyyy\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

if(flagYes == true)//if(r>0)

{

new Actotalcreate(account,w,d,t);

new TotalUpdate(account,t);

JOptionPane.showMessageDialog(this, "Withdraw done successfully");

new Passdisp(t1.getText());

t1.setText("");

t2.setText("");

}

else

{

//JOptionPane.showMessageDialog(this, "No Data Found");

JOptionPane.showMessageDialog(this, "Failed\nAccount balance must be 500 after withdraw.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

t2.setText("");

}

flagYes = false;

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*mmmmmmmmmyyyyyyyyyyyyyyyyyyyy\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

}

flagAccNum =false;

flagWithdraw = false;

}

if(e.getSource()==cancel)

{

int rply = JOptionPane.showConfirmDialog(this, "Are you sure to quit?");

if(rply == JOptionPane.YES\_OPTION)

{

new Userwindow("");

setVisible(false);

}

}

if(e.getSource()==deposit)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*validation check\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

String vaccNum = t1.getText();

String vwithdrw = t2.getText();

/\*for account number check

\* \*/

String accNumpattern = "^[0-9]{4}" ;

Scanner scan1 = new Scanner( vaccNum ) ;

String matched1 = scan1.findInLine( accNumpattern ) ;

if ( matched1 == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID ACCOUNT NUMBER\nIt should contain only digits with minimum length of 4 digit.", "Error", JOptionPane.ERROR\_MESSAGE);

t1.setText("");

}

else

{

flagAccNum = true;

}

/\*for withdraw amount check

\* \*/

String withdrawpattern = "^[0-9]{3}" ;

Scanner scan = new Scanner( vwithdrw ) ;

String matched = scan.findInLine( withdrawpattern ) ;

if ( matched == null )

{

//JOptionPane.showMessageDialog(this, "INVALID PASSWORD\nIt should contain only digits with minimum password length of 4 digit.");

JOptionPane.showMessageDialog(this, "INVALID WITHDRAW AMOUNT\nIt should contain only digits with minimum withdraw amount of INR 100.", "Error", JOptionPane.ERROR\_MESSAGE);

t2.setText("");

}

else

{

flagWithdraw = true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*IF VALIDATION SUCCESSFUL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(flagAccNum == true)

{

if(flagWithdraw == true)

{

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

}catch(Exception e1){}

//x = list1.size();

for(Create re : list1)

{

if(re.getAc().equals(search))

{

flagAccMatch = true;

amnt=Integer.parseInt(re.getInitialamnt());

w=Integer.parseInt(t2.getText());

t=(amnt+w);

account=Integer.parseInt(t1.getText());

d=0;

r++;

}

}

if((flagAccNum == true) && (flagWithdraw== true) && (flagAccMatch == true))//if(r>0)

{

new Actotalcreate(account,d,w,t);//acc ---> account, withdrw -----> d,deposite --------> w,total --------> t (all are int)

new TotalUpdate(account,t);

JOptionPane.showMessageDialog(this, "Deposite done successfully");

new Passdisp(t1.getText());

t1.setText("");

t2.setText("");

}

if(flagAccMatch == false)

{

JOptionPane.showMessageDialog(this, "Account number does not exists");

t1.setText("");

t2.setText("");

}

flagAccMatch = false;

}

}

flagAccNum =false;

flagWithdraw = false;

/\*

try

{

FileInputStream fin=new FileInputStream("Regis.dat");

ObjectInputStream oin=new ObjectInputStream(fin);

list1=(ArrayList<Create>)oin.readObject();

}catch(Exception e1){}

x = list1.size();

for(Create re : list1)

{

if(re.getAc().equals(search))

{

amnt=Integer.parseInt(re.getInitialamnt());

w=Integer.parseInt(t2.getText());

t=(amnt+w);

account=Integer.parseInt(t1.getText());

d=0;

r++;

}

}

if(r>0)

{

new Actotalcreate(account,d,w,t);

new TotalUpdate(account,t);

new Passdisp(t1.getText());

t1.setText("");

t2.setText("");

}

else

{

JOptionPane.showMessageDialog(this, "No Data Found");

}

\*/

}

}

}

PROJECT CERTIFICATES

**Certificate**

This is to certify that Mr Subhrangshu Chaudhuri of JIS College of Engineering, WBUT registration number: 111230110104 of 2011-12 , has successfully completed a project on Banking System using JAVA under the guidance of Mr Chandan Mukherjee**.**

**-----------------------------------------------**

**MR Chandan Mukherjee**

**GLOBSYN FINISHING SCHOOL**

PROJECT CERTIFICATES

**Certificate**

This is to certify that Mr Subhajit Ganguly of JIS College of Engineering, WBUT registration number: 111230110101 of 2011-12, has successfully completed a project on Banking System using JAVA under the guidance of Mr Chandan Mukherjee**.**

**-----------------------------------------------**

**MR Chandan Mukherjee**

**GLOBSYN FINISHING SCHOOL**

PROJECT CERTIFICATES

**Certificate**

This is to certify that Mr Sukanta Sharma of JIS College of Engineering, WBUT registration number: 111230110109 of 2011-12, has successfully completed a project on Banking System using JAVA under the guidance of Mr Chandan Mukherjee**.**

**-----------------------------------------------**

**MR Chandan Mukherjee**

**GLOBSYN FINISHING SCHOOL**

PROJECT CERTIFICATES

**Certificate**

This is to certify that Mr Subhro Dutta of JIS College of Engineering, WBUT registration number: 111230110105 of 2011-12has successfully completed a project on Banking System using JAVA under the guidance of Mr Chandan Mukherjee**.**

**-----------------------------------------------**

**MR Chandan Mukherjee**

**GLOBSYN FINISHING SCHOOL**

PROJECT CERTIFICATES

**Certificate**

This is to certify that Mr Satyabrata Sarkar of JIS College of Engineering, WBUT registration number: 111230110087 of 2011-12, has successfully completed a project on Banking System using JAVA under the guidance of Mr Chandan Mukherjee**.**

**-----------------------------------------------**

**MR Chandan Mukherjee**

**GLOBSYN FINISHING SCHOOL**