**INDEX**

* Introduction To MobiTrueValue
* Background of Study
* Significance of Study
* Objective of project
* Requirements
* Problem Statement
* Software Requirement
* Hardware Requirements
* Front-End & Back-End
* Data Flow Diagram
* ER Diagram
* Database Architecture
* Database Tables
* Database Queries
* Conclusion
* Code of Project

**Abstract**

The word of “MobiTrueValue” means that mobi means Mobile and TrueValue means actual price of Mobile. It is a desktop application. This application help to customers selling and buying old mobiles . It is used to the calculate values of old mobile functionality working. This application is the user friendly and working on easily to calculate the old mobile values.

This application helps to the shopkeeper and shopkeeper easy to calculate values of the old mobile and selling buying. This is the big problem in market today because each shops not see the mobile functionality and mind to calculate the price of old mobile so this application help the calculate of old mobiles actual price found.

**Introduction**

MobiTrueValue means that Mobi means Mobile and TruValue means Actual price of Mobile . It is a desktop application. It will help to user or a shopkeeper to calculate the actual price of customer’s old mobile. This application first register the Admin and generated the user id and password then after admin create the register the user and create user id and password the user. Registration is mainly done by the system administrator for security reasons. The system Administrator register the user on a special method .

**Background Study**

This application is working on different fields like there are two types of customer first is the buyer and second is the seller. The users only working on user page in user page first register the buyer or seller after registration calculate actual prices of old mobile of customer. This is application is the user friendly because every shopkeeper easily understand the user interface of MobiTureValue.

**Significance of Study**

Every Shopkeeper easy to handle the this project and if Amin work in any field in this project. Admin is main role of project like admin change the password the user ,delete user of application ,admin search all details of user ,buyer and seller . We are all user handle the project easily so like user register the buyer and seller details in this project

And calculate actual values of old mobile functionality working bases.

User don’t go through the user page user work easy to maintain the buyer and seller details, maintain the calculation of old mobiles selling and buying .

**Objective of Project**

The purpose of the project is to create a system of desktop application to access by the mobile shop to find out the actual price of selling and Buying the old mobile. All customers buyer and seller buying and selling old mobiles to the shopkeeper ,Shopkeeper is not gives the actual price of mobiles to buyer and seller which customer has not satisfied by the given price of shopkeeper therefore this is the big problem in the market.

This problem solve to the my application MobiTrueValue . MobiTrueValue is a desktop application . This application check the old mobiles functionality working or not then after all process done then calculate actual price of old mobiles.

**Requirements**

* This is the two way of registration first is the Admin and User Registration after you get the unique user id and Password then go through the login Page.
* If you are Admin go through the Admin page or if you are user go through user Page
* In the database information of every customer is stored.
* Database show the information for every user.

**Problem Statement**

* Mobile Shop buying a mobile without function check and it’s not gives actual value of mobile so customer is not satisfied so customer confuse this is the big problem by the mobile shopkeeper .
* This problem solving by the my desktop application which found the actual price of Mobile.

**Software Requirements**

**MYSQL DBMS :-** It allows combination , extraction , manipulation and organization of data in the users and customers. It platform independent and therefore can implemented and used across several such as Windows, Linux server and is compatible with various hardware, mainframes. It is fast in performance , stable and provides business values at a low cost.

**NetBeans IDE 12.5 :-** The NetBeans IDE is an award -wining integrated development environment available for windows ,Mac ,Linux and Solaris. The NetBeans project consists of an open-source IDE and an application platform that enable developers to rapidly create web , enterprise, desktop and mobile application using the java platform as well as PHP, JavaScript and Ajax, Groovy and Grails and C/C++.

The NetBeans project is supported by a vibrant developer community and offers extensible documentation and training resources as well as diverse selection of third party plugins.

**Java Coding :-** This is for advanced user who find PHP codes easy to work with .

**Hardware Requirements**

* Processor i3 or above (if intel)
* Processor ryzen3 or above (if Ryzen)
* RAM 4GB or Above
* 20 GB Space required

**Front-End**

* Java Swing (GUI)

**Back-End**

* Database used (MY SQL Server)

**ER Diagram**

Registration

Admin

user

Type

Login

Admin Page

User page

**Data Flow Diagram**

User login

User registration

Admin login

Admin registration

Admin Page

Buyer registration

Seller registration

User page

Old mobile price calculation

Show Product Details

Show All user

Show buyer

Show Seller

Search All user

**Use Case Diagram**

**Actor**

**Tables**

**Registration Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data type | Description |
| id | Integer | Primary key (auto increment) |
| Name | Varchar | Not null |
| Mobile | Varchar | Not null |
| Email id | Varchar | Not null |
| Username | Varchar | Not null |
| Password | Varchar | Not null |
| Confirm Password | Varchar | Not null |
| Type | Varchar | Not null |

**Buyer Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Date type** | **Description** |
| Id | Int | Primary key(auto increment) |
| name | Varchar | Not Null |
| Mobile No. | Varchar | Not Null |
| Email Id | Varchar | Not Null |
| Address | Varchar | Not Null |
| Type | Varchar | Not Null |

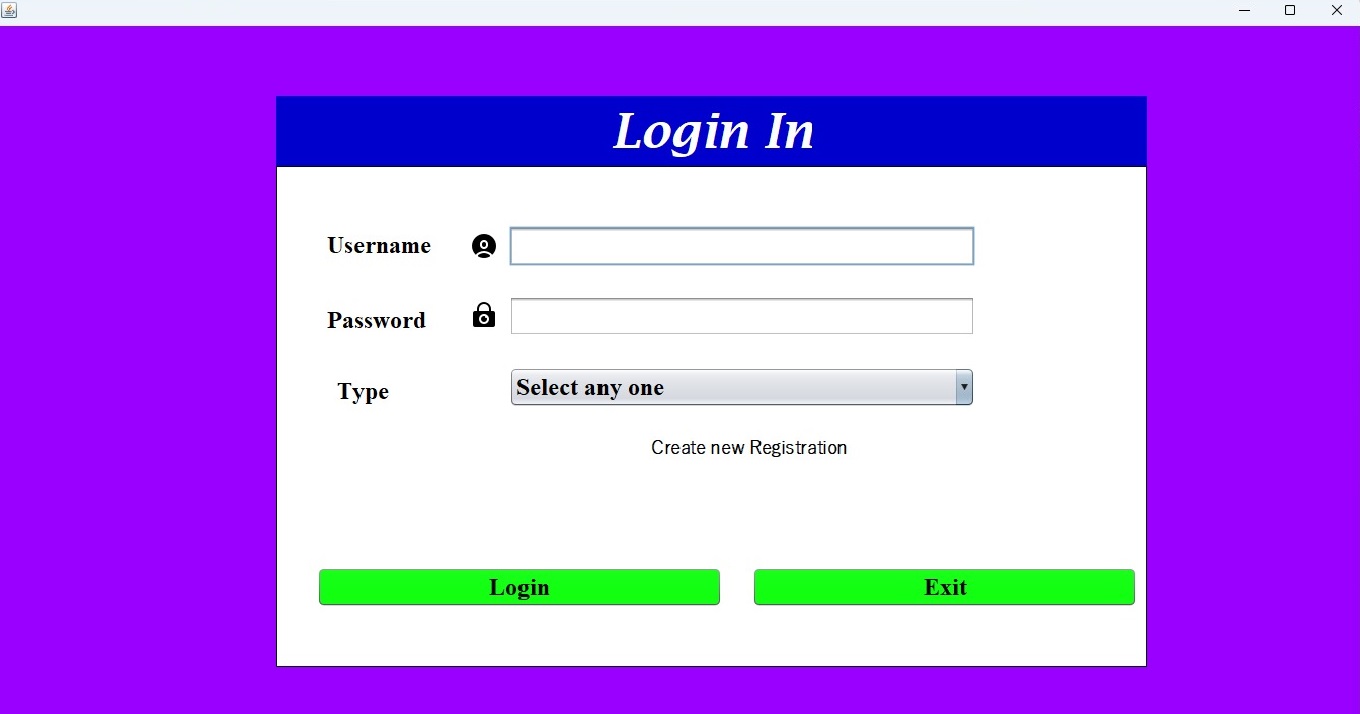
**Product Table**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Description** |
| Product id | Int | Primary key auto increment |
| Customer id | Int | Not Null |
| Bill no. | Varchar | Not Null |
| Product Name | Varchar | Not Null |
| Buying year | Varchar | Not Null |
| Selling year | Varchar | Not Null |
| Buying date | Varchar | Not Null |
| Original Price | Varchar | Not Null |
| Pure Price | Varchar | Not Null |
| Status | Varchar | Not Null |

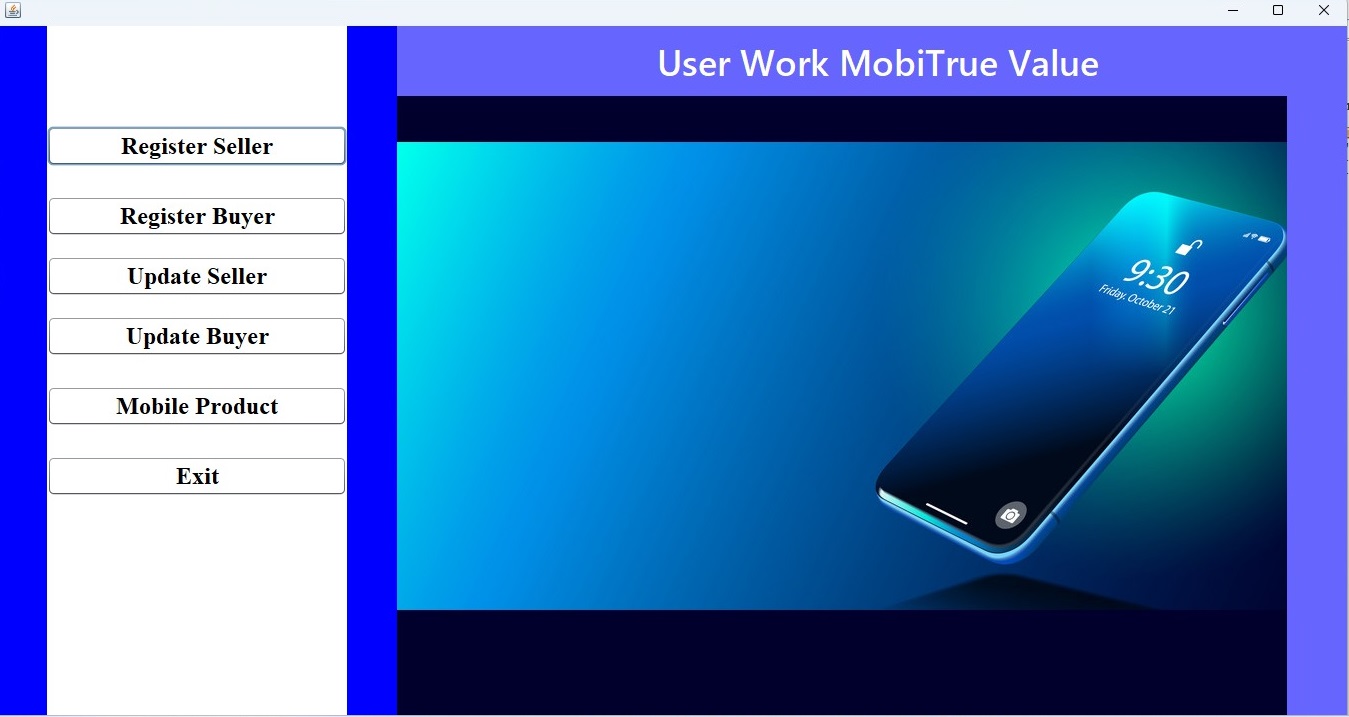
**Figure 1:**

****

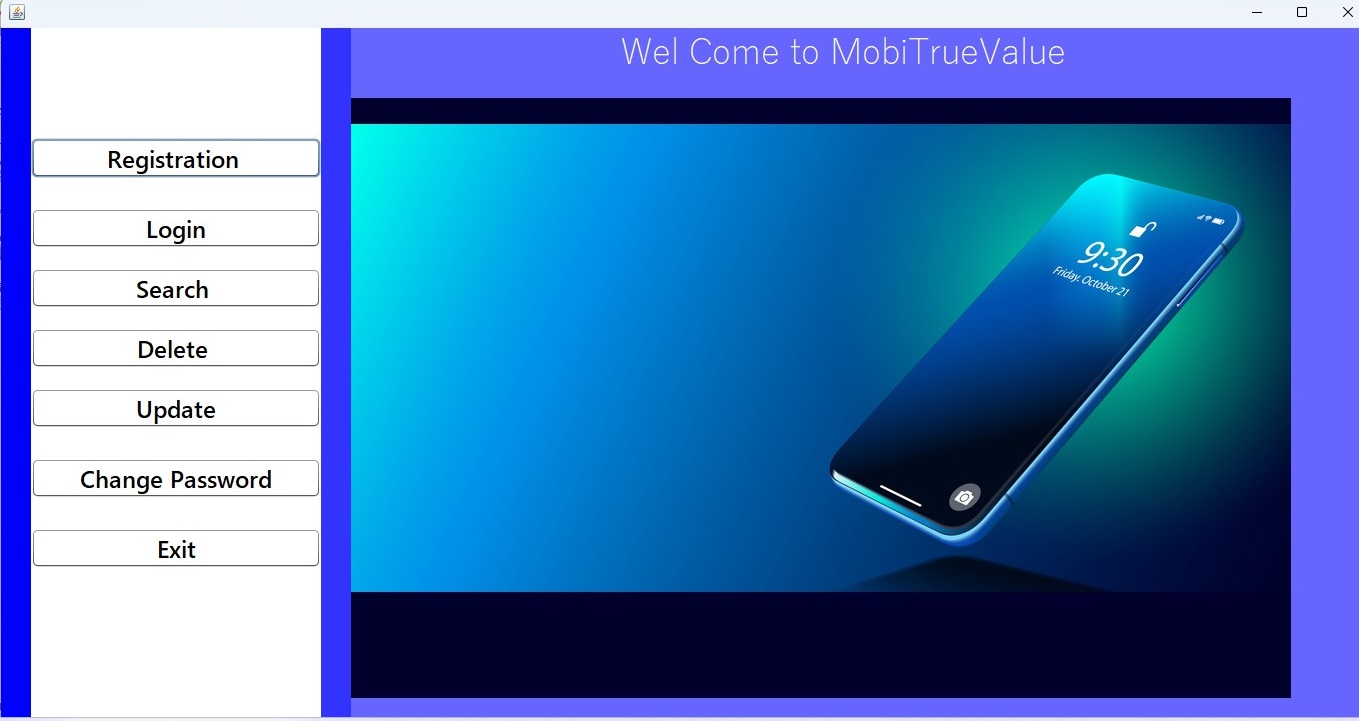
**Figure 2**

****

**Figure 3**

****

**Figure 4**

****

**Conclusion**

* Mobitrue value will help to store the user data and to calculate the actual price of the old mobile phone of user. it provide an User Interface which will make the process of calculating the price easier.
* It can we handled by the shop owner and by the shop employee so that they can easily calculate the actual price of the old mobile phone.
* Admin add to the user registration first time and give the id and Password for user then after user working on the user page Interface
* There are two type of registration process for the customer first is the Buyer and second is the seller.

**Code of Project**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

package Admin;

import java.sql.\*;

import javax.swing.JOptionPane;

import java.util.regex.\*;

/\*\*

\*

\* @author anish8540

\*/

public class Registration extends javax.swing.JFrame {

/\*\*

\* Creates new form Registration

\*/

public Registration() {

initComponents();

this.setSize(1366, 768);

this.setBounds(0, 0, 1366, 768);

}

public boolean validation(){

boolean b=false;

if(nameTxt.getText().equals("")){

b=false;

JOptionPane.showMessageDialog(null, "name is required");

nameTxt.requestFocus();

}

else

b=true;

return b;

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jPanel1 = new javax.swing.JPanel();

jPanel2 = new javax.swing.JPanel();

nametTxt = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jLabel6 = new javax.swing.JLabel();

jLabel1 = new javax.swing.JLabel();

nameTxt = new javax.swing.JTextField();

mobileTxt = new javax.swing.JTextField();

emailTxt = new javax.swing.JTextField();

usernameTxt = new javax.swing.JTextField();

typeComBox = new javax.swing.JComboBox<>();

exitBtn = new javax.swing.JButton();

registerBtn = new javax.swing.JButton();

passwordTxt = new javax.swing.JPasswordField();

confirmTxt = new javax.swing.JPasswordField();

nameL = new javax.swing.JLabel();

nameL1 = new javax.swing.JLabel();

mobileL = new javax.swing.JLabel();

jPanel3 = new javax.swing.JPanel();

jLabel8 = new javax.swing.JLabel();

jLabel7 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setBackground(new java.awt.Color(255, 255, 255));

getContentPane().setLayout(null);

jPanel1.setBackground(new java.awt.Color(153, 0, 255));

jPanel1.setAutoscrolls(true);

jPanel1.setLayout(null);

jPanel2.setBackground(new java.awt.Color(255, 255, 255));

jPanel2.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0), 2));

jPanel2.setLayout(null);

nametTxt.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

nametTxt.setText("Name");

jPanel2.add(nametTxt);

nametTxt.setBounds(20, 40, 70, 30);

jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jLabel2.setText("Mobile No");

jPanel2.add(jLabel2);

jLabel2.setBounds(20, 110, 120, 29);

jLabel3.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jLabel3.setText("Email Id");

jPanel2.add(jLabel3);

jLabel3.setBounds(20, 180, 110, 29);

jLabel4.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jLabel4.setText("Username");

jPanel2.add(jLabel4);

jLabel4.setBounds(20, 240, 110, 29);

jLabel5.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jLabel5.setText("Password");

jPanel2.add(jLabel5);

jLabel5.setBounds(20, 310, 100, 29);

jLabel6.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jLabel6.setText("Confirm Password");

jPanel2.add(jLabel6);

jLabel6.setBounds(10, 380, 200, 29);

jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

jLabel1.setText("Type");

jPanel2.add(jLabel1);

jLabel1.setBounds(20, 460, 60, 29);

nameTxt.addKeyListener(new java.awt.event.KeyAdapter() {

public void keyReleased(java.awt.event.KeyEvent evt) {

nameTxtKeyReleased(evt);

}

});

jPanel2.add(nameTxt);

nameTxt.setBounds(280, 30, 510, 40);

mobileTxt.addKeyListener(new java.awt.event.KeyAdapter() {

public void keyReleased(java.awt.event.KeyEvent evt) {

mobileTxtKeyReleased(evt);

}

});

jPanel2.add(mobileTxt);

mobileTxt.setBounds(280, 100, 510, 40);

jPanel2.add(emailTxt);

emailTxt.setBounds(280, 170, 510, 40);

jPanel2.add(usernameTxt);

usernameTxt.setBounds(280, 230, 510, 40);

typeComBox.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N

typeComBox.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Select any one", "Admin", "User" }));

jPanel2.add(typeComBox);

typeComBox.setBounds(280, 450, 520, 40);

exitBtn.setBackground(java.awt.Color.green);

exitBtn.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

exitBtn.setText("Back");

exitBtn.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

exitBtnActionPerformed(evt);

}

});

jPanel2.add(exitBtn);

exitBtn.setBounds(670, 530, 400, 40);

registerBtn.setBackground(java.awt.Color.green);

registerBtn.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N

registerBtn.setText("Register");

registerBtn.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

registerBtnActionPerformed(evt);

}

});

jPanel2.add(registerBtn);

registerBtn.setBounds(200, 530, 410, 40);

jPanel2.add(passwordTxt);

passwordTxt.setBounds(280, 300, 510, 40);

jPanel2.add(confirmTxt);

confirmTxt.setBounds(280, 370, 510, 40);

nameL.setFont(new java.awt.Font("Times New Roman", 1, 12)); // NOI18N

nameL.setText(" ");

jPanel2.add(nameL);

nameL.setBounds(540, 40, 110, 20);

nameL1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jPanel2.add(nameL1);

nameL1.setBounds(820, 40, 210, 20);

mobileL.setFont(new java.awt.Font("Yu Gothic UI Light", 1, 18)); // NOI18N

jPanel2.add(mobileL);

mobileL.setBounds(840, 110, 190, 0);

jPanel1.add(jPanel2);

jPanel2.setBounds(120, 80, 1090, 590);

jPanel3.setBackground(new java.awt.Color(0, 0, 204));

jPanel3.setForeground(new java.awt.Color(255, 255, 255));

jLabel8.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Admin/icons8-add-user-male-48.png"))); // NOI18N

jPanel3.add(jLabel8);

jLabel7.setFont(new java.awt.Font("Times New Roman", 3, 48)); // NOI18N

jLabel7.setForeground(new java.awt.Color(255, 255, 255));

jLabel7.setText("Registration");

jPanel3.add(jLabel7);

jPanel1.add(jPanel3);

jPanel3.setBounds(120, 20, 1090, 60);

getContentPane().add(jPanel1);

jPanel1.setBounds(0, 0, 1370, 770);

pack();

}// </editor-fold>

private void registerBtnActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

String name=nameTxt.getText().trim();

String mobile=mobileTxt.getText().trim();

String email=emailTxt.getText().trim();

String username=usernameTxt.getText().trim();

String password=passwordTxt.getText().toString().trim();

String confirm=confirmTxt.getText().toString().trim();

String type=typeComBox.getSelectedItem().toString().trim();

if(name.isEmpty()!=true)

{

if(mobile.isEmpty()!=true){

if(email.isEmpty()!=true){

if(username.isEmpty()!=true){

if(password.isEmpty()!=true){

if(confirm.isEmpty()!=true){

if(typeComBox.getSelectedIndex()!=0){

Connection con=DriverManager.getConnection("jdbc:mysql://localhost/mobitruevaluedb","root","1234");

String sqlry="INSERT INTO registration(name,mobile\_No,Email\_id,Username,Password,Confirm\_password,Type)VALUES(?,?,?,?,?,?,?)";

PreparedStatement pst=con.prepareStatement(sqlry);

pst.setString(1, name);

pst.setString(2, mobile);

pst.setString(3, email);

pst.setString(4, username);

pst.setString(5, password);

pst.setString(6, confirm);

pst.setString(7, type);

int st1=pst.executeUpdate();

if(st1!=0)

{

JOptionPane.showMessageDialog(this, "registration sucessfully");

nameTxt.setText("");

mobileTxt.setText("");

emailTxt.setText("");

passwordTxt.setText("");

confirmTxt.setText("");

usernameTxt.setText("");

typeComBox.setToolTipText("");

}

else{

JOptionPane.showMessageDialog(this, "registration unsuccessfully");

}

}else{

JOptionPane.showMessageDialog(this, "Select your type");

}

}else{

JOptionPane.showMessageDialog(this, "Enter your Confirm password, Confrim password is Empty");

}

}else{

JOptionPane.showMessageDialog(this, "Enter your Password, password is empty");

}

}else{

JOptionPane.showMessageDialog(this, "Enter your username, username is Empty");

}

}else{

JOptionPane.showMessageDialog(this, "Enter your Email id, Email is Empty");

}

}else{

JOptionPane.showMessageDialog(this, "Enter your mobile no, mobile no is Empty");

}

}else{

JOptionPane.showMessageDialog(this, "Enter your name, name is Empty");

}

}

catch(Exception ex)

{

//ex.printStackTrace();

JOptionPane.showMessageDialog(this, "please enter your data");

}

}

private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {

new LoginPage().setVisible(true);

this.setVisible(false);

}

private void nameTxtKeyReleased(java.awt.event.KeyEvent evt) {

String pat="^[a-zA-Z\\s]{0,30}$";

Pattern patr=Pattern.compile(pat, 10);

Matcher match=patr.matcher(nameTxt.getText().trim());

if(!match.matches())

{

nameL1.setText("Naming is Invaild");

}

else{

nameL1.setText(" ");

}

}

private void mobileTxtKeyReleased(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String pat="\\d+";

Pattern patr=Pattern.compile(pat,10);

Matcher match=patr.matcher(mobileTxt.getText().trim());

if(!match.matches())

{

mobileL.setText("invail mobile number");

}

else

mobileL.setText(" ");

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Registration.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Registration.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Registration.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Registration.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

new Registration().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JPasswordField confirmTxt;

private javax.swing.JTextField emailTxt;

private javax.swing.JButton exitBtn;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

private javax.swing.JLabel jLabel7;

private javax.swing.JLabel jLabel8;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel2;

private javax.swing.JPanel jPanel3;

private javax.swing.JLabel mobileL;

private javax.swing.JTextField mobileTxt;

private javax.swing.JLabel nameL;

private javax.swing.JLabel nameL1;

private javax.swing.JTextField nameTxt;

private javax.swing.JLabel nametTxt;

private javax.swing.JPasswordField passwordTxt;

private javax.swing.JButton registerBtn;

private javax.swing.JComboBox<String> typeComBox;

private javax.swing.JTextField usernameTxt;

// End of variables declaration

}