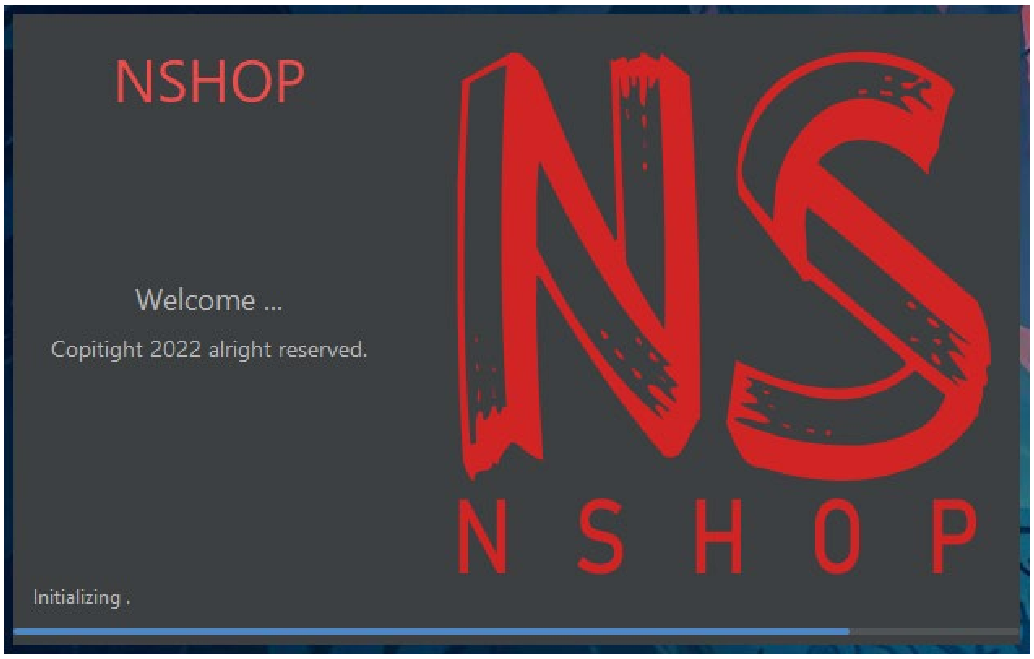
Software Documentation

Splash screen

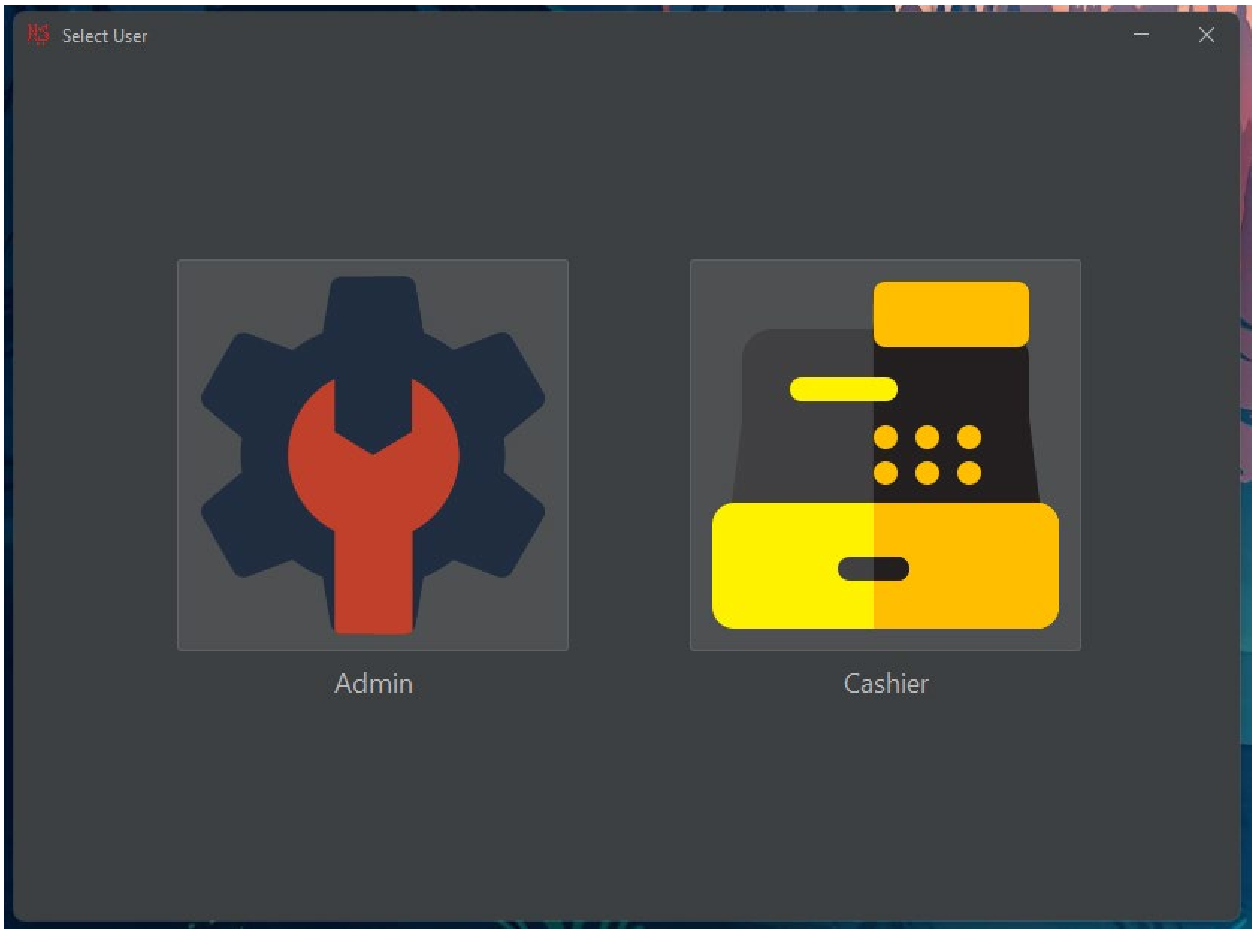


```
public static void main(String args[]) {

    com.formdev.flatlaf.FlatDarkLaf.setup();

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

        public void run() {

            Splash s = new Splash();

            s.setVisible(true);

            Runnable runnable = new Runnable() {

                public void run() {

                    for (int i = 0; i < 101; i++) {

                        s.jProgressBar1.setValue(i);

                        try {
```

```
                Thread.sleep(20);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        SelectSignin ss = new SelectSignin();
        ss.setVisible(true);
        s.dispose();
    }
};
Thread t = new Thread(runnable);
t.start();
Runnable runnable1 = new Runnable() {
    public void run() {
        for (int i = 0; i < 6; i++) {
            if (i % 2 == 0) {
                s.jLabel4.setText("Initializing ..");
            } else if (i % 3 == 0) {
                s.jLabel4.setText("Initializing ...");
            } else {
                s.jLabel4.setText("Initializing .");
            }
```

```
                    try {
                        Thread.sleep(333);
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            }
        };
        Thread t1 = new Thread(runnable1);
        t1.start();
    }
});
}
```

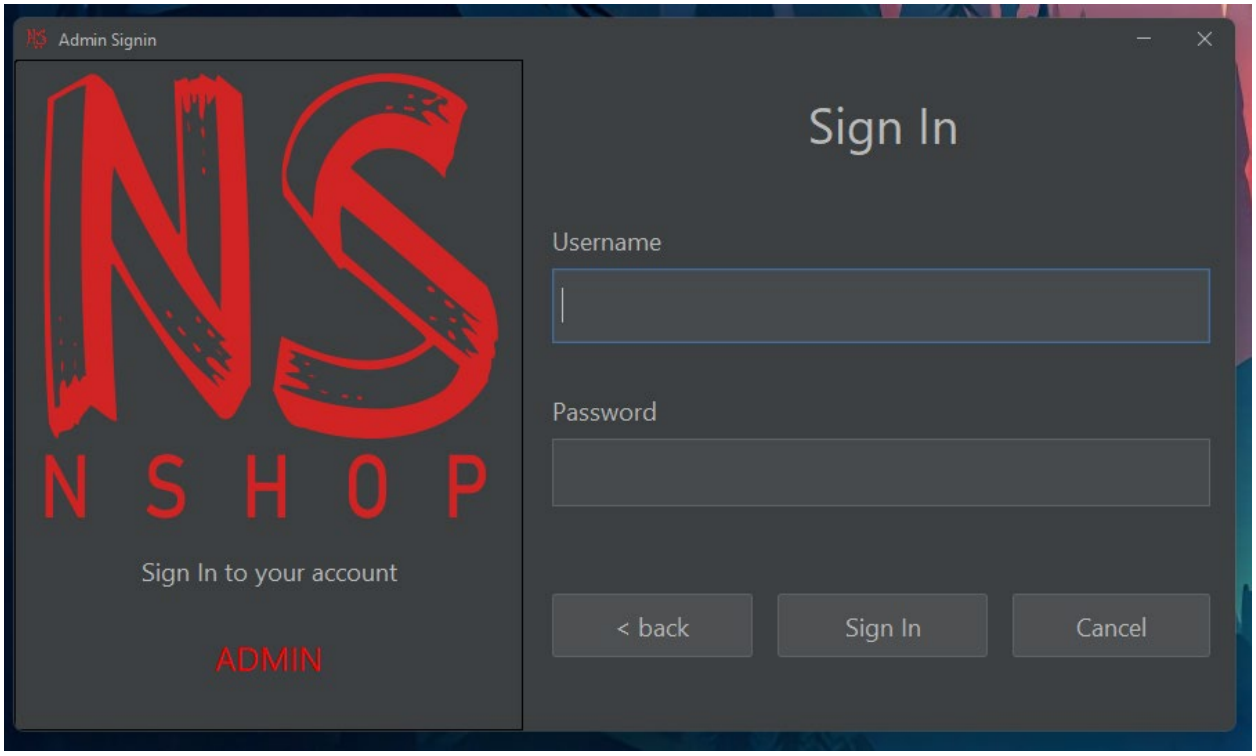
User Select



```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    as = new AdminSignIn();
    as.setVisible(true);
    this.dispose();
}


private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    cs = new CashierSignIn1();
    cs.setVisible(true);
```

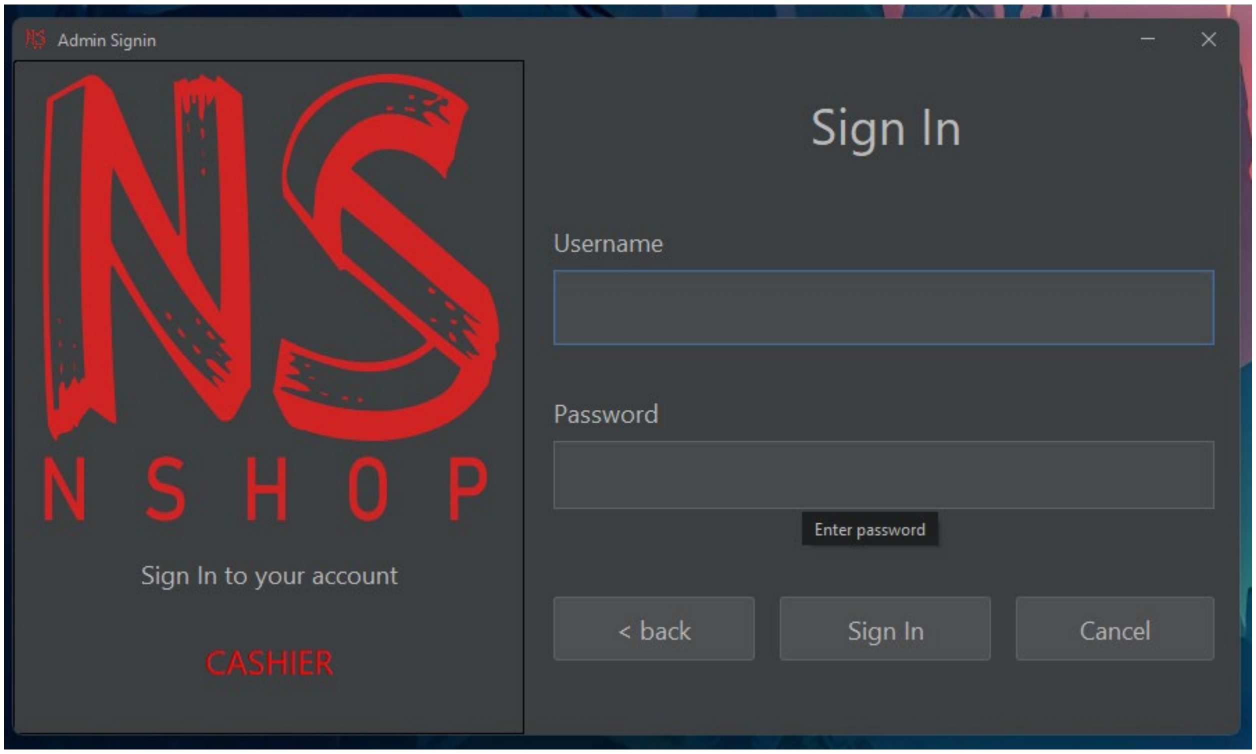
```
        this.dispose();
    }
```

Admin Sign in



```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        SelectSignin ss = new SelectSignin();
        ss.setVisible(true);
        this.dispose();
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        this.dispose();
    }


    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        String un = jTextField1.getText();
        char[] pw = jPasswordField1.getPassword();
        String password = String.valueOf(pw);

        if (un.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Username", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (password.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Password", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            try {
                ResultSet rs = MySQL.search("SELECT * FROM `user` WHERE `username` = '" + un + "'
AND `password` = '" + password + "' AND `user_type_id` IN (1,2,3);");
                if (rs.next()) {
                    if (rs.getString("ststus_id").equals("1")) {
                        AdminId = rs.getString("id");

                        Success s = new Success(as, false);
            s.setVisible(true);

            Runnable runnable = new Runnable() {

                @Override

                public void run() {

                    try {

                        Thread.sleep(1200);



                    } catch (Exception e) {

                        e.printStackTrace();

                    }

                    s.dispose();

                    AdminHome ah;

                    try {

                        ah = new AdminHome(rs.getString("fname") + " " + rs.getString("lname"),
rs.getString("user_type_id"));

                        ah.setVisible(true);

                    } catch (SQLException ex) {

                        ex.printStackTrace();

                    }



                    SelectSignin.as.dispose();

                    System.gc();
                }


            };

            Thread t = new Thread(runnable);

            t.start();

          } else {

            JOptionPane.showMessageDialog(this, "You are currently Blocked please contact
an admin.", "Warning", JOptionPane.ERROR_MESSAGE);

          }


        } else {

          JOptionPane.showMessageDialog(this, "Invalid Username Or Password", "Warning",
JOptionPane.WARNING_MESSAGE);

        }

      } catch (Exception e) {

        e.printStackTrace();

      }

    }

}


private void jTextField1KeyPressed(java.awt.event.KeyEvent evt) {

    if (evt.getKeyCode() == 10) {
      jPasswordField1.grabFocus();
    }
}

private void jPasswordField1KeyPressed(java.awt.event.KeyEvent evt) {
    if (evt.getKeyCode() == 10) {
      jButton2ActionPerformed(null);
    }
}
```

Cashier Sign in

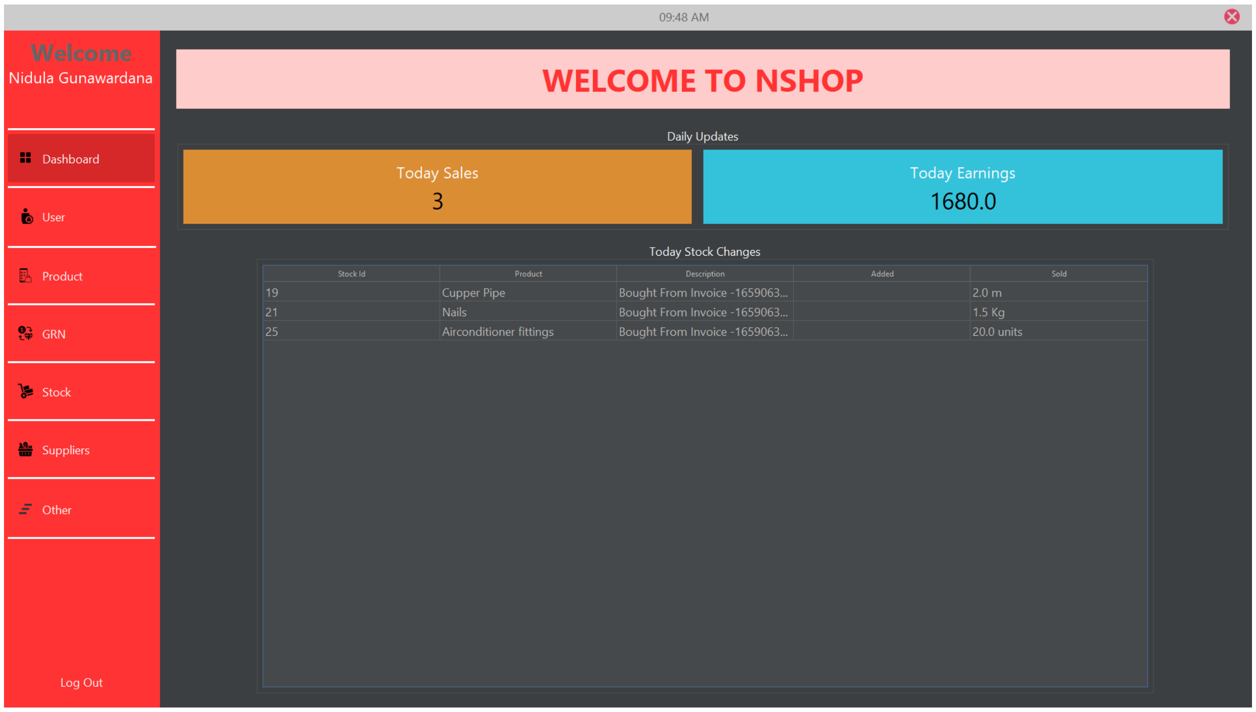


```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        SelectSignin ss = new SelectSignin();
        ss.setVisible(true);

        this.dispose();

    }


    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        this.dispose();

    }


    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        String un = jTextField1.getText();

        char[] pw = jPasswordField1.getPassword();

        String password = String.valueOf(pw);


        if (un.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter Username", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (password.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter Password", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else {

            try {
```

```
        ResultSet rs = MySQL.search("SELECT * FROM `user` WHERE `username` = '" + un + "'
AND `password` = '" + password + "' AND `user_type_id` = 4;");

        if (rs.next()) {

            if (rs.getString("ststus_id").equals("1")) {

                cashierId = rs.getString("id");

                Success s = new Success(cs, false);

                s.setVisible(true);

                Runnable runnable = new Runnable() {

                    @Override

                    public void run() {

                        try {

                            Thread.sleep(1200);

                        } catch (Exception e) {

                            e.printStackTrace();

                        }

                        s.dispose();

                        CashierHome ch = new CashierHome();

                        ch.setVisible(true);

                        SelectSignin.cs.dispose();

                        System.gc();

                    }

                };
```

```
                    Thread t = new Thread(runnable);
                    t.start();
                } else {
                    JOptionPane.showMessageDialog(this, "You are currently blocked. Please Contact
an admin.", "Warning", JOptionPane.ERROR_MESSAGE);
                }
            } else {
                JOptionPane.showMessageDialog(this, "Invalid Username Or Password", "Warning",
JOptionPane.WARNING_MESSAGE);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
private void jTextField1KeyPressed(java.awt.event.KeyEvent evt) {
    if (evt.getKeyCode() == 10) {
        jPasswordField1.grabFocus();
    }
}

private void jPasswordField1KeyPressed(java.awt.event.KeyEvent evt) {
    if (evt.getKeyCode() == 10) {

        jButton2ActionPerformed(null);

    }

}
```

Admin Home



WELCOME TO NSHOP

Daily Updates

Today Sales 3

Today Earnings 1680.0

Today Stock Changes

Stock Id	Product	Description	Added	Sold
19	Cupper Pipe	Bought From Invoice -1659063...		2.0 m
21	Nails	Bought From Invoice -1659063...		1.5 Kg
25	Airconditioner fittings	Bought From Invoice -1659063...		20.0 units

```
public AdminHome() {

        initComponents();

        viewTime();

        jButton2ActionPerformed(null);



    }
```

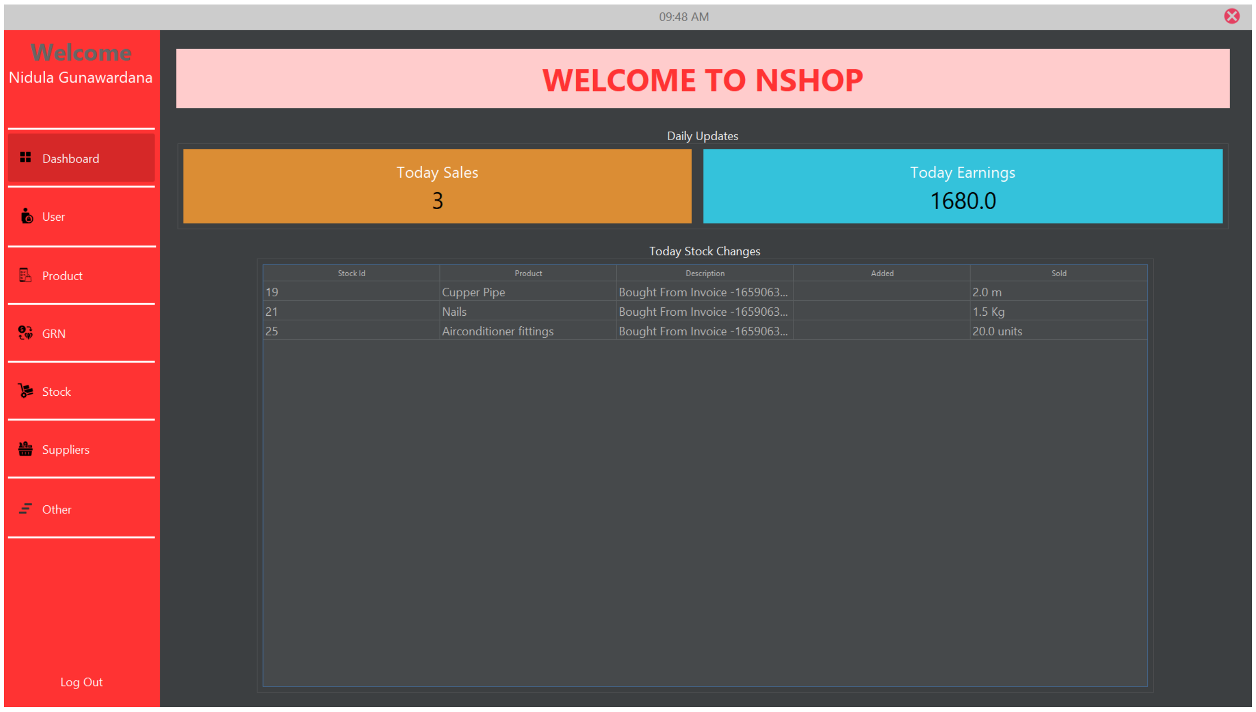
```
public AdminHome(String username, String adminType) {

    initComponents();

    ImageIcon i = new ImageIcon("src/resources/logomain.png");

    Image x = i.getImage();

    setIconImage(x);

    viewTime();

    jButton2ActionPerformed(null);

    jLabel4.setText(username);

    this.adminType = adminType;

    if (adminType.equals("2")) {

        jButton3.setVisible(false);

        jLabel7.setVisible(false);

        jButton8.setVisible(false);

        jLabel17.setVisible(false);

        jButton6.setVisible(false);

        jLabel14.setVisible(false);

    } else if (adminType.equals("3")) {

        jButton3.setVisible(false);

        jLabel7.setVisible(false);

        jButton8.setVisible(false);

        jLabel17.setVisible(false);

        jButton6.setVisible(false);
        jLabel14.setVisible(false);

        jButton7.setVisible(false);

        jLabel16.setVisible(false);

    }

}


public void viewTime() {

    DateTimeFormatter sdf = DateTimeFormatter.ofPattern("hh:mm a");



    Runnable runnable = new Runnable() {

        @Override

        public void run() {

            while (true) {

                jLabel19.setText(sdf.format(LocalTime.now()));

                try {

                    Thread.sleep(10000);

                } catch (Exception e) {

                    e.printStackTrace();

                }

            }



        }
        };


        timet = new Thread(runnable);

        timet.start();

    }


    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        timet.stop();

        timet = null;

        this.dispose();


    }


    private void jLabel15MouseClicked(java.awt.event.MouseEvent evt) {

        int option = JOptionPane.showConfirmDialog(this, "Do you want to log out?",
"Confirmation", JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);

        if (option == JOptionPane.YES_OPTION) {

            timet.stop();

            new SelectSignin().setVisible(true);

            this.dispose();

        }

    }
```

```
Color enteredC = new Color(214, 40, 40);

Color exitC = new Color(255, 51, 51);

Color pressedC = new Color(226, 85, 85);

Color releasedC = new Color(240, 114, 114);


private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

   jPanel4.removeAll();

   dashboard d = new dashboard();

   jPanel4.add(d);

   jPanel4.validate();

   jPanel4.repaint();


   jButton2.setBackground(enteredC);

   jButton3.setBackground(exitC);

   jButton4.setBackground(exitC);

   jButton5.setBackground(exitC);

   jButton6.setBackground(exitC);

   jButton7.setBackground(exitC);

   jButton8.setBackground(exitC);

   System.gc();

}
```

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    if (this.adminType.equals("1")) {
        jPanel4.removeAll();
        Users u = new Users(ah);
        jPanel4.add(u);
        jPanel4.validate();
        jPanel4.repaint();

        jButton2.setBackground(exitC);
        jButton3.setBackground(enteredC);
        jButton4.setBackground(exitC);
        jButton5.setBackground(exitC);
        jButton6.setBackground(exitC);
        jButton7.setBackground(exitC);
        jButton8.setBackground(exitC);
        System.gc();
    }

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    jPanel4.removeAll();

    Product p = new Product(ah);

    jPanel4.add(p);

    jPanel4.validate();

    jPanel4.repaint();


    jButton2.setBackground(exitC);

    jButton3.setBackground(exitC);

    jButton4.setBackground(enteredC);

    jButton5.setBackground(exitC);

    jButton6.setBackground(exitC);

    jButton7.setBackground(exitC);

    jButton8.setBackground(exitC);

    System.gc();

}


private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

    jPanel4.removeAll();

    GRN grn = new GRN(ah);

    jPanel4.add(grn);

    jPanel4.validate();

    jPanel4.repaint();
    jButton2.setBackground(exitC);

    jButton3.setBackground(exitC);

    jButton4.setBackground(exitC);

    jButton5.setBackground(enteredC);

    jButton6.setBackground(exitC);

    jButton7.setBackground(exitC);

    jButton8.setBackground(exitC);

    System.gc();

}


private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

    jPanel4.removeAll();

    Stock st = new Stock(ah);

    jPanel4.add(st);

    jPanel4.validate();

    jPanel4.repaint();



    jButton2.setBackground(exitC);

    jButton3.setBackground(exitC);

    jButton4.setBackground(exitC);

    jButton5.setBackground(exitC);
    jButton6.setBackground(enteredC);

    jButton7.setBackground(exitC);

    jButton8.setBackground(exitC);

    System.gc();

}


private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

    jPanel4.removeAll();

    Supplier su = new Supplier(ah);

    jPanel4.add(su);

    jPanel4.validate();

    jPanel4.repaint();


    jButton2.setBackground(exitC);

    jButton3.setBackground(exitC);

    jButton4.setBackground(exitC);

    jButton5.setBackground(exitC);

    jButton6.setBackground(exitC);

    jButton7.setBackground(enteredC);

    jButton8.setBackground(exitC);

    System.gc();

}
```

```
private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

    jPanel4.removeAll();

    Other ot = new Other(ah);

    jPanel4.add(ot);

    jPanel4.validate();

    jPanel4.repaint();


    jButton2.setBackground(exitC);

    jButton3.setBackground(exitC);

    jButton4.setBackground(exitC);

    jButton5.setBackground(exitC);

    jButton6.setBackground(exitC);

    jButton7.setBackground(exitC);

    jButton8.setBackground(enteredC);


    System.gc();

}
```

Dashboard



WELCOME TO NSHOP

Daily Updates

Today Sales
3

Today Earnings
1680.0

Today Stock Changes

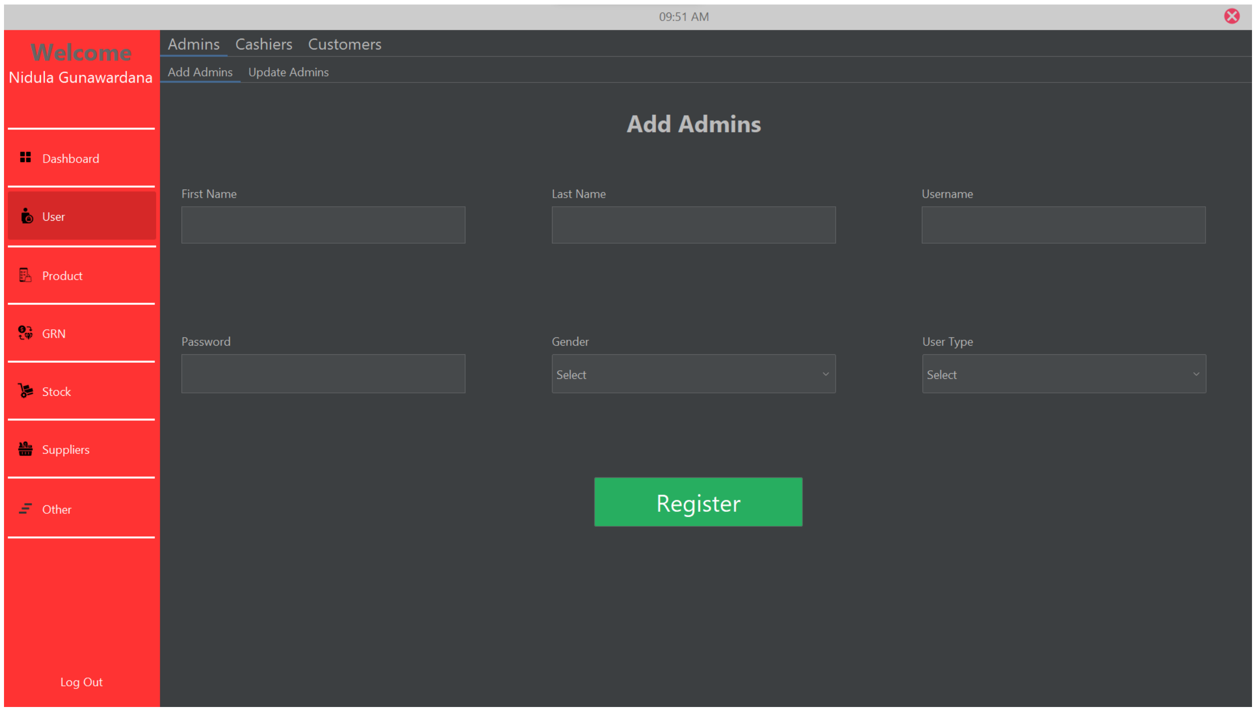
Stock Id	Product	Description	Added	Sold
19	Cupper Pipe	Bought From Invoice -1659063...		2.0 m
21	Nails	Bought From Invoice -1659063...		1.5 Kg
25	Airconditioner fittings	Bought From Invoice -1659063...		20.0 units

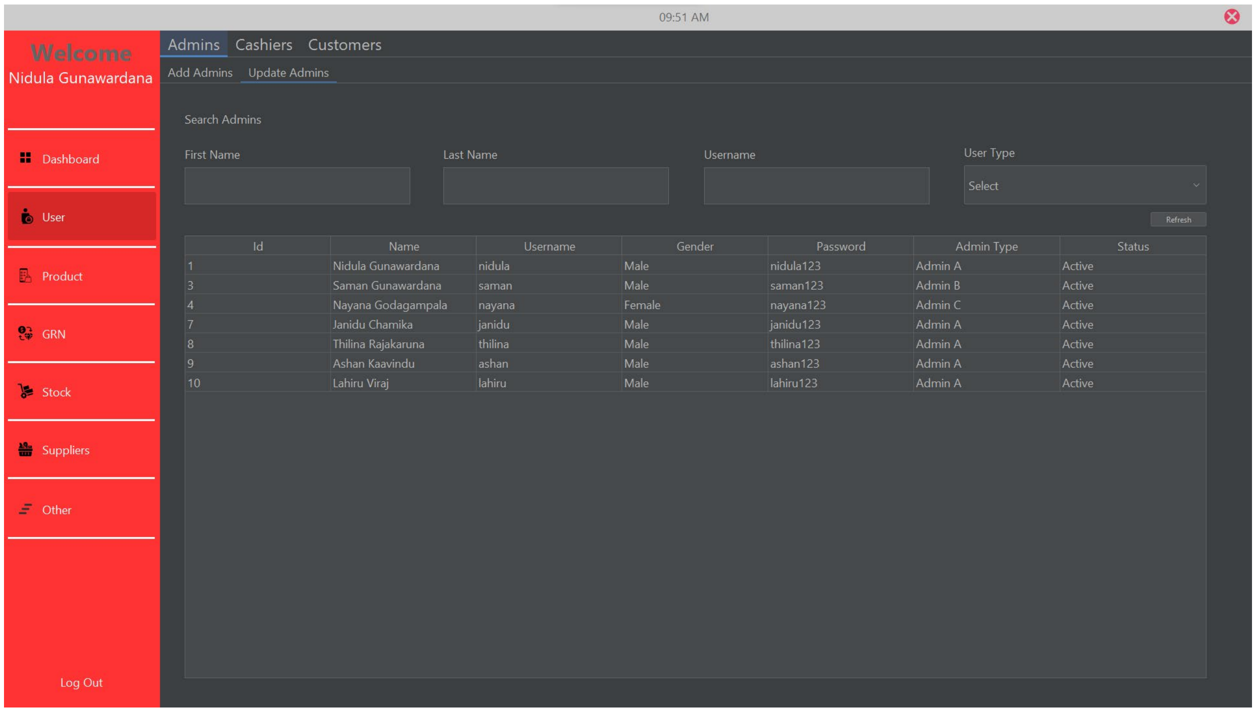
```
public dashboard() {

    initComponents();

    searchStock();

    todaySales();

    todayEarnings();

  }



  public void todaySales() {

    try {

      int tot = 0;

      String dNow = new SimpleDateFormat("yyyy-MM-dd").format(new Date());
```

```
            ResultSet rs = MySQL.search("SELECT * FROM `stock_changes` WHERE
`stock_changes`.`date_time` >= '" + dNow + " 00:00:00' AND `stock_changes`.`date_time` <= '" +
dNow + " 23:59:59' ORDER BY `stock_changes`.`date_time` DESC;");
            while (rs.next()) {
                if (Double.parseDouble(rs.getString("qty")) < 0) {
                    tot++;
                }
            }
            jLabel2.setText(String.valueOf(tot));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void todayEarnings() {
        try {
            double tot = 0;
            String dNow = new SimpleDateFormat("yyyy-MM-dd").format(new Date());
            ResultSet rs = MySQL.search("SELECT * FROM `invoice` INNER JOIN `invoice_payments`
ON `invoice_payments`.`invoice_id` = `invoice`.`id` WHERE `invoice`.`date_time` >= '" + dNow +
" 00:00:00' AND `invoice`.`date_time` <= '" + dNow + " 23:59:59' ORDER BY
`invoice`.`date_time` DESC;");
            while (rs.next()) {
                tot = tot + Double.parseDouble(rs.getString("invoice_payments.payment")) +
Double.parseDouble(rs.getString("invoice_payments.balance"));
            }
            jLabel4.setText(String.valueOf(tot));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void searchStock() {

        try {

            String dNow = new SimpleDateFormat("yyyy-MM-dd").format(new Date());
            ResultSet rs = MySQL.search("SELECT * FROM `stock_changes` INNER JOIN `stock` ON
`stock`.`id` = `stock_changes`.`stock_id` INNER JOIN `qty_units` ON `qty_units`.`id` =
`stock_changes`.`qty_units_id` INNER JOIN `product` ON `product`.`id` = `stock`.`product_id`
WHERE `stock_changes`.`date_time` >= '" + dNow + " 00:00:00' AND
`stock_changes`.`date_time` <= '" + dNow + " 23:59:59' ORDER BY `stock_changes`.`date_time`
DESC;");

            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            while (rs.next()) {

                Vector v = new Vector();

                if (Double.parseDouble(rs.getString("stock_changes.qty")) < 0) {
                    //Qty unit Id
                    ResultSet quirs = MySQL.search("SELECT * FROM `qty_units` WHERE `id` = '" +
rs.getString("stock_changes.qty_units_id") + "';");
                    quirs.next();
                    String mul = quirs.getString("multiplication");
                    //Qty Unit Id
                    v.add(rs.getString("stock.id"));
                    v.add(rs.getString("product.name"));
                    v.add(rs.getString("stock_changes.description"));
                    v.add("");
                    v.add(String.valueOf(-1 * Double.parseDouble(rs.getString("stock_changes.qty"))) +
" " + rs.getString("qty_units.name"));

                } else if (Double.parseDouble(rs.getString("stock_changes.qty")) > 0) {
                    //Qty unit Id
                    ResultSet quirs = MySQL.search("SELECT * FROM `qty_units` WHERE `id` = '" +
rs.getString("stock_changes.qty_units_id") + "';");
```

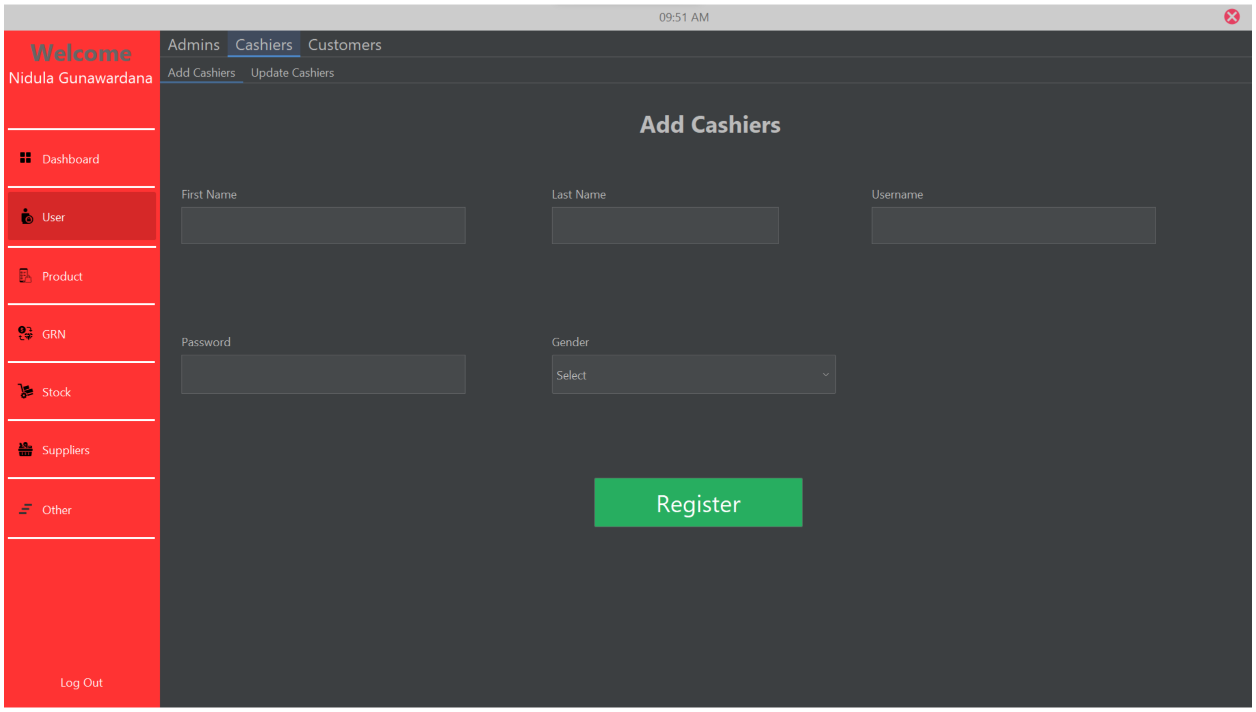
```
                quirs.next();

                String mul = quirs.getString("multiplication");

                //Qty Unit Id

                v.add(rs.getString("stock.id"));

                v.add(rs.getString("product.name"));

                v.add(rs.getString("stock_changes.description"));

                v.add(rs.getString("stock_changes.qty") + " " + rs.getString("qty_units.name"));

                v.add("");


            }

            dtm.addRow(v);

        }

    } catch (Exception e) {

        e.printStackTrace();

    }

}
```

User





Id	Name	Username	Gender	Password	Admin Type	Status
1	Nidula Gunawardana	nidula	Male	nidula123	Admin A	Active
3	Saman Gunawardana	saman	Male	saman123	Admin B	Active
4	Nayana Godagampala	nayana	Female	nayana123	Admin C	Active
7	Janidu Chamika	janidu	Male	janidu123	Admin A	Active
8	Thilina Rajakaruna	thilina	Male	thilina123	Admin A	Active
9	Ashan Kaavindu	ashan	Male	ashan123	Admin A	Active
10	Lahiru Viraj	lahiru	Male	lahiru123	Admin A	Active



Add Cashiers

First Name

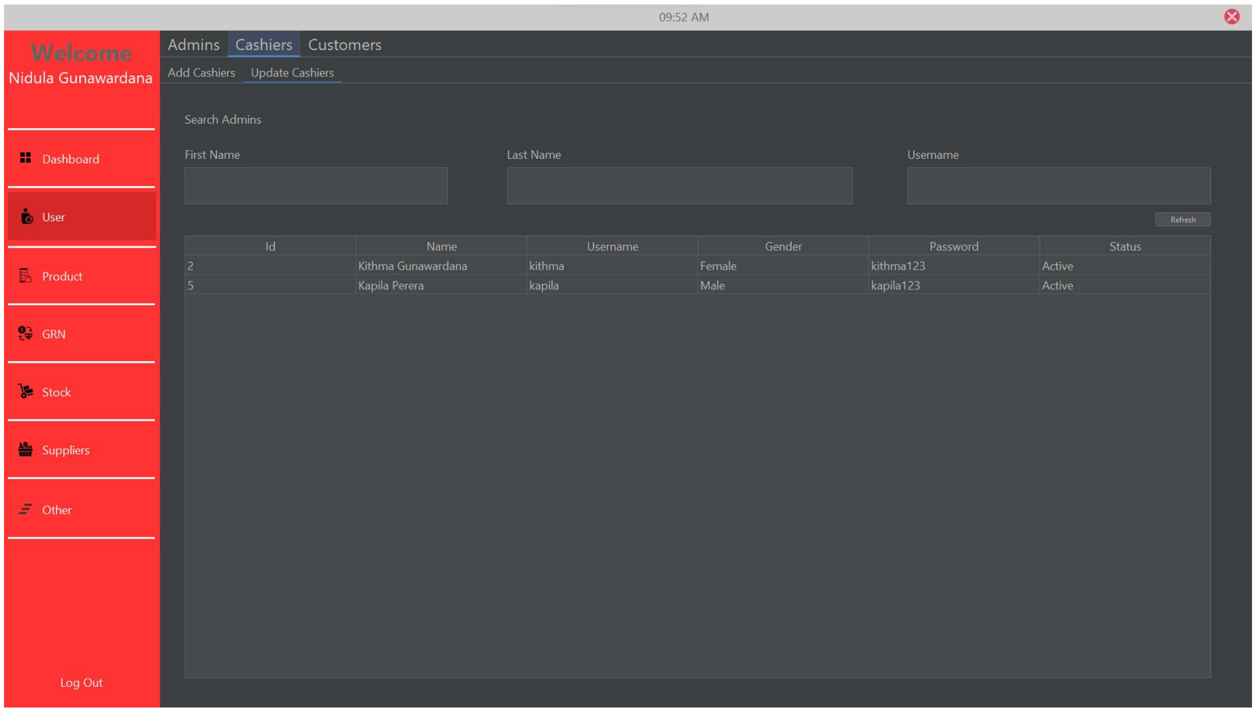
Last Name

Username

Password

Gender

Select



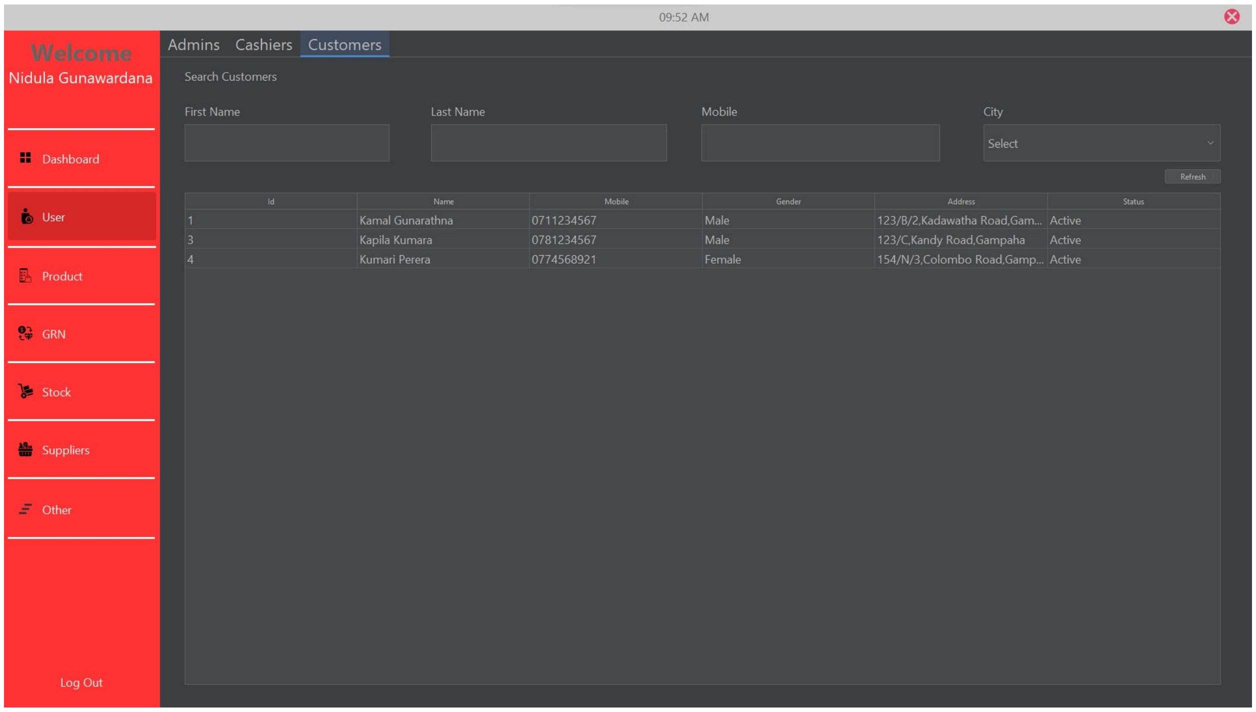
Search Admins

First Name

Last Name

Username

Id	Name	Username	Gender	Password	Status
2	Kithma Gunawardana	kithma	Female	kithma123	Active
5	Kapila Perera	kapila	Male	kapila123	Active



Search Customers

First Name	Last Name	Mobile	City
			Select

Id	Name	Mobile	Gender	Address	Status
1	Kamal Gunarathna	0711234567	Male	123/B/2,Kadawatha Road,Gam...	Active
3	Kapila Kumara	0781234567	Male	123/C,Kandy Road,Gampaha	Active
4	Kumari Perera	0774568921	Female	154/N/3,Colombo Road,Gamp...	Active

```
AdminHome ah;

    int to_updateId = -1;

    int to_updateCashierId = -1;



    /**

     * Creates new form dashboard

     */

    public Users() {

        initComponents();

        loadGender();

        loadUserTypes();

        jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));
    jTable2.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    searchAdmins();

    searchCashiers();

    loadCities();

    searchCustomers();

}


public Users(AdminHome ah) {

    initComponents();

    loadGender();

    loadUserTypes();

    this.ah = ah;

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    jTable2.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    searchAdmins();

    searchCashiers();

    loadCities();

    searchCustomers();

}


public void loadGender() {
    try {

        ResultSet rs = MySQL.search("SELECT * FROM `gender`;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox1.setModel(dcm);


        DefaultComboBoxModel dcm1 = new DefaultComboBoxModel(v);

        jComboBox4.setModel(dcm1);


    } catch (Exception e) {

        e.printStackTrace();

    }


}


public void loadUserTypes() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `user_type` WHERE `id` IN (1,2,3);");
```

```
            Vector v = new Vector();

            v.add("Select");

            while (rs.next()) {

                v.add(rs.getString("name"));

            }

            DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

            jComboBox2.setModel(dcm);

            DefaultComboBoxModel dcm1 = new DefaultComboBoxModel(v);

            jComboBox3.setModel(dcm1);

        } catch (Exception e) {

            e.printStackTrace();

        }

    }


    public void searchAdmins() {

        String fname = jTextField4.getText();

        String lname = jTextField5.getText();

        String un = jTextField6.getText();

        String at = jComboBox3.getSelectedItem().toString();


        Vector queryv = new Vector();
```

```
if (!fname.isBlank()) {

    queryv.add("`user`.`fname` LIKE '" + fname + "%' ");

}


if (!lname.isBlank()) {

    queryv.add("`user`.`lname` LIKE '" + lname + "%' ");

}

if (!un.isBlank()) {

    queryv.add("`user`.`username` LIKE '" + un + "%' ");

}

if (!at.equals("Select")) {

    queryv.add("`user_type`.`name` = '" + at + "' ");

} else {

    queryv.add("`user`.`user_type_id` IN (1,2,3) ");

}


String query = "";



for (int i = 0; i < queryv.size(); i++) {

    if (i == 0) {

        query += "WHERE ";

    }
```

```
            query += queryv.get(i);

            if (i + 1 != queryv.size()) {

                query += "AND ";

            }


        }

//        System.out.println(query);

        try {

            ResultSet rs = MySQL.search("SELECT * FROM `user` INNER JOIN `user_type` ON
`user_type`.`id` = `user`.`user_type_id` INNER JOIN `ststus` ON `ststus`.`id` = `user`.`ststus_id`
INNER JOIN `gender` ON `gender`.`id` = `user`.`gender_id` " + query + " ORDER BY `user`.`id`;");


            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

            dtm.setRowCount(0);

            while (rs.next()) {

                Vector v = new Vector();

                v.add(rs.getString("user.id"));

                v.add(rs.getString("user.fname") + " " + rs.getString("user.lname"));

                v.add(rs.getString("user.username"));

                v.add(rs.getString("gender.name"));

                v.add(rs.getString("user.password"));
            v.add(rs.getString("user_type.name"));

            v.add(rs.getString("ststus.name"));

            dtm.addRow(v);

        }

    } catch (Exception e) {

        e.printStackTrace();

    }


}


public void clearAddFields() {

    jTextField1.setText("");

    jTextField2.setText("");

    jTextField3.setText("");

    jPasswordField1.setText("");

    jComboBox1.setSelectedIndex(0);

    jComboBox2.setSelectedIndex(0);

    jButton1.setText("Register");

    jLabel1.setText("Add Admins");

    to_updateId = -1;

}
```

```
public void loadForUpdate() {

    if (to_updateId != -1) {

        jTextField1.setText(jTable1.getValueAt(to_updateId, 1).toString().split(" ")[0]);

        jTextField2.setText(jTable1.getValueAt(to_updateId, 1).toString().split(" ")[1]);

        jTextField3.setText(jTable1.getValueAt(to_updateId, 2).toString());

        jPasswordField1.setText(jTable1.getValueAt(to_updateId, 4).toString());

        jComboBox1.setSelectedItem(jTable1.getValueAt(to_updateId, 3).toString());

        jComboBox2.setSelectedItem(jTable1.getValueAt(to_updateId, 5).toString());

        jTabbedPane2.setSelectedIndex(0);

        jButton1.setText("Update");

        jLabel1.setText("Update Admins");

    }

}


public void changeStatusAdmin() {

    statusChangeAdmin aca = new statusChangeAdmin(ah, false, this, to_updateId, 1);

    aca.setVisible(true);

}


public void changeStatusCashier() {

    statusChangeAdmin aca = new statusChangeAdmin(ah, false, this, to_updateCashierId, 2);

    aca.setVisible(true);
```

```
}


public void searchCashiers() {

    String fname = jTextField10.getText();

    String lname = jTextField11.getText();

    String un = jTextField12.getText();



    Vector queryv = new Vector();



    if (!fname.isBlank()) {

        queryv.add("`user`.`fname` LIKE '" + fname + "%' ");

    }



    if (!lname.isBlank()) {

        queryv.add("`user`.`lname` LIKE '" + lname + "%' ");

    }

    if (!un.isBlank()) {

        queryv.add("`user`.`username` LIKE '" + un + "%' ");

    }



    queryv.add("`user`.`user_type_id` = 4 ");
```

```
        String query = "";


        for (int i = 0; i < queryv.size(); i++) {
            if (i == 0) {
                query += "WHERE ";
            }


            query += queryv.get(i);
            if (i + 1 != queryv.size()) {
                query += "AND ";
            }


        }
//        System.out.println(query);
        try {
            ResultSet rs = MySQL.search("SELECT * FROM `user` INNER JOIN `user_type` ON
`user_type`.`id` = `user`.`user_type_id` INNER JOIN `ststus` ON `ststus`.`id` = `user`.`ststus_id`
INNER JOIN `gender` ON `gender`.`id` = `user`.`gender_id` " + query + " ORDER BY `user`.`id`;");


            DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();
            dtm.setRowCount(0);
            while (rs.next()) {
```

```
            Vector v = new Vector();

            v.add(rs.getString("user.id"));

            v.add(rs.getString("user.fname") + " " + rs.getString("user.lname"));

            v.add(rs.getString("user.username"));

            v.add(rs.getString("gender.name"));

            v.add(rs.getString("user.password"));

            v.add(rs.getString("ststus.name"));

            dtm.addRow(v);

        }

    } catch (Exception e) {

        e.printStackTrace();

    }

}


public void clearAddFieldsCashier() {

    jTextField7.setText("");

    jTextField8.setText("");

    jTextField9.setText("");

    jPasswordField2.setText("");

    jComboBox4.setSelectedIndex(0);

    jButton3.setText("Register");

    jLabel13.setText("Add Cashiers");
    to_updateCashierId = -1;

}


public void loadForupdateCashier() {

    if (to_updateCashierId != -1) {

        jTextField7.setText(jTable2.getValueAt(to_updateCashierId, 1).toString().split(" ")[0]);

        jTextField8.setText(jTable2.getValueAt(to_updateCashierId, 1).toString().split(" ")[1]);

        jTextField9.setText(jTable2.getValueAt(to_updateCashierId, 2).toString());

        jPasswordField2.setText(jTable2.getValueAt(to_updateCashierId, 4).toString());

        jComboBox4.setSelectedItem(jTable2.getValueAt(to_updateCashierId, 3).toString());

        jTabbedPane3.setSelectedIndex(0);

        jButton3.setText("Update");

        jLabel13.setText("Update Cashier");

    }

}


public void loadCities() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `city`;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {
            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox5.setModel(dcm);


    } catch (Exception e) {

        e.printStackTrace();

    }

}


public void searchCustomers() {

    String fname = jTextField13.getText();

    String lname = jTextField14.getText();

    String mobile = jTextField15.getText();

    String city = jComboBox5.getSelectedItem().toString();


    Vector queryv = new Vector();


    if (!fname.isBlank()) {

        queryv.add("`customers`.`fname` LIKE '" + fname + "%' ");

    }
```

```
if (!lname.isBlank()) {
    queryv.add("`customers`.`lname` LIKE '" + lname + "%' ");
}
if (!mobile.isBlank()) {
    queryv.add("`customers`.`mobile` LIKE '" + mobile + "%' ");
}

if (!city.equals("Select")) {
    queryv.add("`city`.`name` LIKE '" + city + "%' ");
}

 queryv.add("`customers`.`id` NOT IN ('0') ");

String query = "";

for (int i = 0; i < queryv.size(); i++) {
    if (i == 0) {
        query += "WHERE ";
    }

    query += queryv.get(i);
    if (i + 1 != queryv.size()) {
            query += "AND ";

          }


        }

        try {

          ResultSet rs = MySQL.search("SELECT * FROM `customers` INNER JOIN `ststus` ON
`ststus`.`id` = `customers`.`ststus_id` INNER JOIN `gender` ON `gender`.`id` =
`customers`.`gender_id` INNER JOIN `city` ON `city`.`id` = `customers`.`city_id` " + query + "
ORDER BY `customers`.`id`;");


          DefaultTableModel dtm = (DefaultTableModel) jTable3.getModel();

          dtm.setRowCount(0);

          while (rs.next()) {

            Vector v = new Vector();

            v.add(rs.getString("customers.id"));

            v.add(rs.getString("customers.fname") + " " + rs.getString("customers.lname"));

            v.add(rs.getString("customers.mobile"));

            v.add(rs.getString("gender.name"));

            v.add(rs.getString("customers.line1") + "," + rs.getString("customers.line2") + "," +
rs.getString("city.name"));

            v.add(rs.getString("ststus.name"));

            dtm.addRow(v);

          }
        } catch (Exception e) {

            e.printStackTrace();

        }

    }

  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        String fname = jTextField1.getText();

        String lname = jTextField2.getText();

        String un = jTextField3.getText();

        String gender = jComboBox1.getSelectedItem().toString();

        String ut = jComboBox2.getSelectedItem().toString();

        char[] pwar = jPasswordField1.getPassword();

        String password = String.valueOf(pwar);


        if (fname.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter First name", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (lname.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter Last name", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (un.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter Username", "Warning",
JOptionPane.WARNING_MESSAGE);
```

```
        } else if (password.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Password", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (gender.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select Gender", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (ut.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select Admin Type", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {

            try {
                //Gender
                ResultSet grs = MySQL.search("SELECT * FROM `gender` WHERE `name` = '" + gender +
"';");
                grs.next();
                String genderId = grs.getString("id");
                //Gender

                //Admin Type
                ResultSet utrs = MySQL.search("SELECT * FROM `user_type` WHERE `name` = '" + ut +
"';");
                utrs.next();
                String utId = utrs.getString("id");

                //Admin Type

                if (jButton1.getText().equals("Register")) {

                    //Search For user

                    ResultSet ars = MySQL.search("SELECT * FROM `user` WHERE `username` = '" + un +
"' AND `password` = '" + password + "';");

                    //Search For user


                    if (ars.next()) {

                        JOptionPane.showMessageDialog(this, "Username and password amready exists
add diffeent one.", "Warning", JOptionPane.WARNING_MESSAGE);

                    } else {


                        MySQL.iud("INSERT INTO
`user`(`fname`,`lname`,`username`,`gender_id`,`password`,`user_type_id`,`ststus_id`) VALUES('"
+ fname + "','" + lname + "','" + un + "','" + genderId + "','" + password + "','" + utId + "','1');");

                        AddedSuccess as = new AddedSuccess(ah, false);

                        as.setVisible(true);

                        Runnable runnable = new Runnable() {

                            @Override

                            public void run() {

                                try {

                                    Thread.sleep(1500);
                                    as.dispose();

                                } catch (Exception e) {

                                    e.printStackTrace();

                                }

                            }

                        };

                        Thread st = new Thread(runnable);

                        st.start();

                        clearAddFields();

                        searchAdmins();

                    }

                } else {

                    //Search For user

                    ResultSet ars = MySQL.search("SELECT * FROM `user` WHERE `username` = '" + un +
"' AND `password` = '" + password + "' AND `id` NOT IN (" + jTable1.getValueAt(to_updateId,
0).toString() + ");");

                    //Search For user


                    if (ars.next()) {

                        JOptionPane.showMessageDialog(this, "Username and password amready exists
try diffeent one.", "Warning", JOptionPane.WARNING_MESSAGE);

                    } else {
```

```
                        MySQL.iud("UPDATE `user` SET `fname`='" + fname + "',`lname`='" + lname +
"',`username`='" + un + "',`gender_id`='" + genderId + "',`password`='" + password +
"',`user_type_id`='" + utId + "' WHERE `id` = '" + jTable1.getValueAt(to_updateId, 0).toString() +
"';");
                        AddedSuccess as = new AddedSuccess(ah, false, "Updated Successfully");
                        as.setVisible(true);
                        Runnable runnable = new Runnable() {
                            @Override
                            public void run() {
                                try {
                                    Thread.sleep(1500);
                                    as.dispose();
                                } catch (Exception e) {
                                    e.printStackTrace();
                                }
                            }
                        };
                        Thread st = new Thread(runnable);
                        st.start();
                    }
                    clearAddFields();
                    searchAdmins();
```

```
            }

        } catch (Exception e) {
            e.printStackTrace();
        }

    }

}

private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {
    searchAdmins();
}

private void jTextField5KeyReleased(java.awt.event.KeyEvent evt) {
    searchAdmins();
}

private void jTextField6KeyReleased(java.awt.event.KeyEvent evt) {
    searchAdmins();
}
```

```
private void jComboBox3ItemStateChanged(java.awt.event.ItemEvent evt) {
    searchAdmins();
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    jTextField4.setText("");
    jTextField5.setText("");
    jTextField6.setText("");
    jComboBox3.setSelectedIndex(0);
    searchAdmins();
}

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
    int selectedRow = jTable1.getSelectedRow();
    if (evt.getClickCount() == 2) {
        if (selectedRow != -1) {
            updateChange uc = new updateChange(ah, false, this, 1);
            uc.setVisible(true);
            this.to_updateId = selectedRow;
        }

    }
    }


    private void jTabbedPane2StateChanged(javax.swing.event.ChangeEvent evt) {
        if (jTabbedPane2.getSelectedIndex() == 1 && jButton1.getText().equals("Update")) {
            clearAddFields();
        }
    }


    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        String fname = jTextField7.getText();
        String lname = jTextField8.getText();
        String un = jTextField9.getText();
        String gender = jComboBox4.getSelectedItem().toString();
        char[] pwar = jPasswordField2.getPassword();
        String password = String.valueOf(pwar);


        if (fname.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter First name", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (lname.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Last name", "Warning",
JOptionPane.WARNING_MESSAGE);
```

```
        } else if (un.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Username", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (password.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Password", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (gender.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select Gender", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {

            try {
                //Gender
                ResultSet grs = MySQL.search("SELECT * FROM `gender` WHERE `name` = '" + gender +
"';");
                grs.next();
                String genderId = grs.getString("id");
                //Gender

                if (jButton3.getText().equals("Register")) {
                    //Search For user
                    ResultSet ars = MySQL.search("SELECT * FROM `user` WHERE `username` = '" + un +
"' AND `password` = '" + password + "';");
```

```
            //Search For user


            if (ars.next()) {
                JOptionPane.showMessageDialog(this, "Username and password amready exists
add diffeent one.", "Warning", JOptionPane.WARNING_MESSAGE);
            } else {


                MySQL.iud("INSERT INTO
`user`(`fname`,`lname`,`username`,`gender_id`,`password`,`user_type_id`,`ststus_id`) VALUES('"
+ fname + "','" + lname + "','" + un + "','" + genderId + "','" + password + "','4','1');");
                AddedSuccess as = new AddedSuccess(ah, false);
                as.setVisible(true);
                Runnable runnable = new Runnable() {
                    @Override
                    public void run() {
                        try {
                            Thread.sleep(1500);
                            as.dispose();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                };

                Thread st = new Thread(runnable);

                st.start();

                clearAddFieldsCashier();

                searchCashiers();

            }

        } else {

            //Search For user

            ResultSet ars = MySQL.search("SELECT * FROM `user` WHERE `username` = '" + un +
"' AND `password` = '" + password + "' AND `id` NOT IN (" +
jTable1.getValueAt(to_updateCashierId, 0).toString() + ");");

            //Search For user



            if (ars.next()) {

                JOptionPane.showMessageDialog(this, "Username and password amready exists
try diffeent one.", "Warning", JOptionPane.WARNING_MESSAGE);

            } else {

                MySQL.iud("UPDATE `user` SET `fname`='" + fname + "',`lname`='" + lname +
"',`username`='" + un + "',`gender_id`='" + genderId + "',`password`='" + password + "' WHERE
`id` = '" + jTable2.getValueAt(to_updateCashierId, 0).toString() + "';");

                AddedSuccess as = new AddedSuccess(ah, false, "Updated Successfully");

                as.setVisible(true);

                Runnable runnable = new Runnable() {
                    @Override
                    public void run() {
                        try {
                            Thread.sleep(1500);
                            as.dispose();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                };
                Thread st = new Thread(runnable);
                st.start();
            }
            clearAddFieldsCashier();
            searchCashiers();
        }

    } catch (Exception e) {
        e.printStackTrace();
    }

}
```

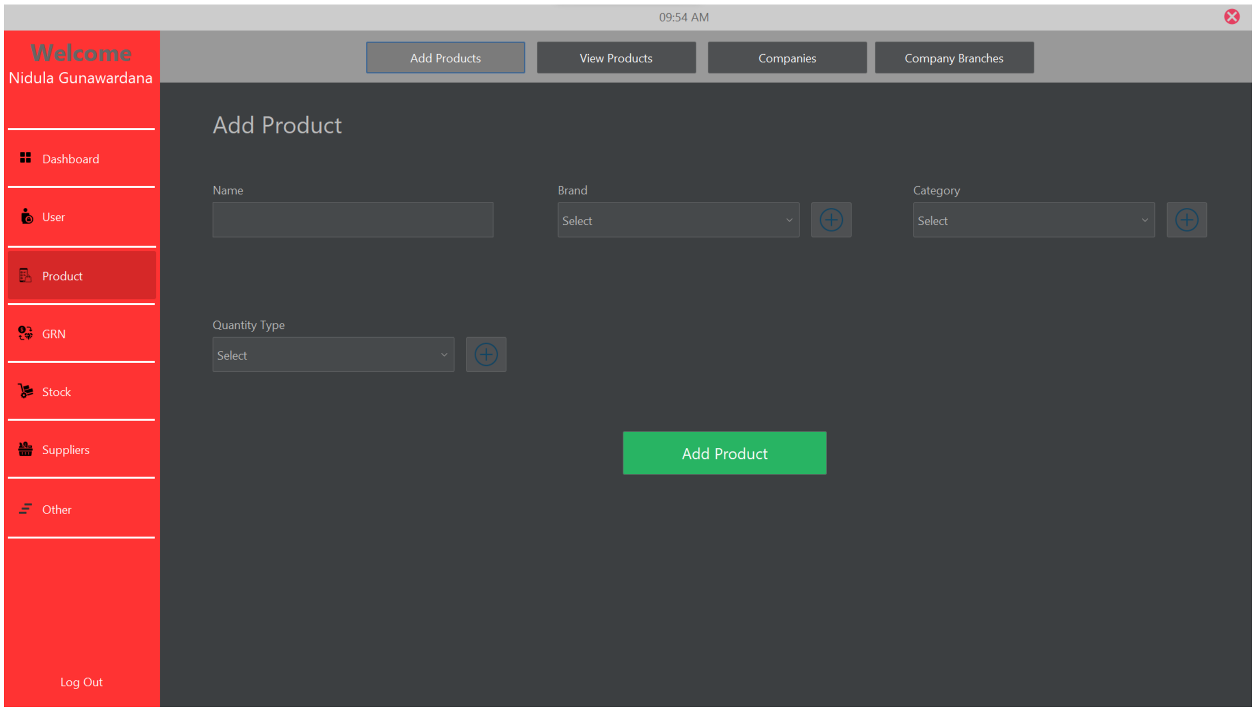
```
}

private void jTextField10KeyReleased(java.awt.event.KeyEvent evt) {
    searchCashiers();
}

private void jTextField11KeyReleased(java.awt.event.KeyEvent evt) {
    searchCashiers();
}

private void jTextField12KeyReleased(java.awt.event.KeyEvent evt) {
    searchCashiers();
}

private void jTable2MouseClicked(java.awt.event.MouseEvent evt) {
    int selectedRow = jTable2.getSelectedRow();
    if (evt.getClickCount() == 2) {
        if (selectedRow != -1) {
            updateChange uc = new updateChange(ah, false, this, 2);
            uc.setVisible(true);
            this.to_updateCashierId = selectedRow;
        }


    }

}


private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

    jTextField10.setText("");

    jTextField11.setText("");

    jTextField12.setText("");

    searchCashiers();

}


private void jTabbedPane3StateChanged(javax.swing.event.ChangeEvent evt) {

    if (jTabbedPane3.getSelectedIndex() == 1 && jButton3.getText().equals("Update")) {

        clearAddFieldsCashier();

    }

}


private void jTextField13KeyReleased(java.awt.event.KeyEvent evt) {

    searchCustomers();

}
```

```
    private void jTextField14KeyReleased(java.awt.event.KeyEvent evt) {
        searchCustomers();
    }


    private void jTextField15KeyReleased(java.awt.event.KeyEvent evt) {
        searchCustomers();
    }


    private void jTable3MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable3.getSelectedRow();
            if (selectedRow != -1) {

                int con = JOptionPane.showConfirmDialog(this, "Do you want to update?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
                if (con == JOptionPane.YES_OPTION) {
                    ToUpdateCustomerDetails tucd = new ToUpdateCustomerDetails();
                    tucd.setCustomerId(jTable3.getValueAt(selectedRow, 0).toString());
                    tucd.setFname(jTable3.getValueAt(selectedRow, 1).toString().split("\\s")[0]);
                    tucd.setLname(jTable3.getValueAt(selectedRow, 1).toString().split("\\s")[1]);
                    tucd.setMobile(jTable3.getValueAt(selectedRow, 2).toString());
                    tucd.setGender(jTable3.getValueAt(selectedRow, 3).toString());
                tucd.setLine1(jTable3.getValueAt(selectedRow, 4).toString().split(",")[0]);

                tucd.setLine2(jTable3.getValueAt(selectedRow, 4).toString().split(",")[1]);

                tucd.setCity(jTable3.getValueAt(selectedRow, 4).toString().split(",")[2]);


                CustomerRegistration cr = new CustomerRegistration(ah, true, tucd, this);

                cr.setVisible(true);


            }

        }


    }

}


private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

    jTextField13.setText("");

    jTextField14.setText("");

    jTextField15.setText("");

    jComboBox5.setSelectedIndex(0);

    searchCustomers();


}
```

```
    private void jComboBox5ItemStateChanged(java.awt.event.ItemEvent evt) {
        searchCustomers();
    }
```

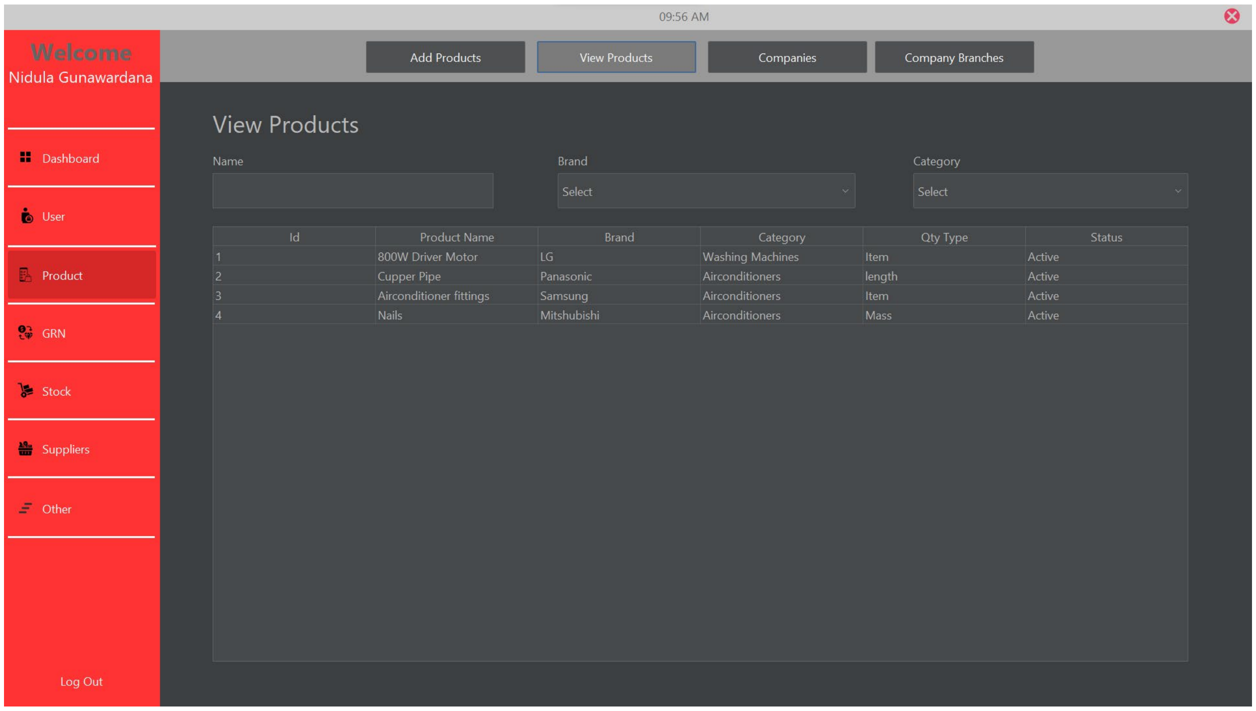
Products

Add products



```
    AdminHome ah;
    ToUpdateDataProduct tudp;
    Product pr;

    public AddProducts() {
        initComponents();
    loadBrand();

    loadCategory();

    loadQtyType();

    jButton5.setVisible(false);

}

public AddProducts(AdminHome ah) {

    initComponents();

    loadBrand();

    loadCategory();

    loadQtyType();

    this.ah = ah;

    jButton5.setVisible(false);

}

public AddProducts(AdminHome ah, ToUpdateDataProduct tudp, Product pr) {

    initComponents();

    loadBrand();

    loadCategory();

    loadQtyType();

    this.ah = ah;

    jTextField1.setText(tudp.getName());

    jComboBox1.setSelectedItem(tudp.getBrand());

    jComboBox2.setSelectedItem(tudp.getCategory());
    jComboBox3.setSelectedItem(tudp.getQttyType());

    jButton4.setText("Update");

    jLabel2.setText("Update Product");

    this.tudp = tudp;

    this.pr = pr;

}

public void loadBrand() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `brand`;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox1.setModel(dcm);

    } catch (Exception e) {

        e.printStackTrace();

    }

}

public void loadCategory() {

    try {
        ResultSet rs = MySQL.search("SELECT * FROM `category`;");
        Vector v = new Vector();
        v.add("Select");
        while (rs.next()) {
            v.add(rs.getString("name"));
        }
        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
        jComboBox2.setModel(dcm);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
public void loadQtyType() {
    try {
        ResultSet rs = MySQL.search("SELECT * FROM `qty_type`;");
        Vector v = new Vector();
        v.add("Select");
        while (rs.next()) {
            v.add(rs.getString("name"));
        }
        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
        jComboBox3.setModel(dcm);
    } catch (Exception e) {

        e.printStackTrace();

    }

}

public void clearFields() {

    jTextField1.setText("");

    jComboBox1.setSelectedIndex(0);

    jComboBox2.setSelectedIndex(0);

    jComboBox3.setSelectedIndex(0);

    jButton4.setText("Add Product");

    jLabel2.setText("Add Product");

}
```

View Products



Id	Product Name	Brand	Category	Qty Type	Status
1	800W Driver Motor	LG	Washing Machines	Item	Active
2	Cupper Pipe	Panasonic	Airconditioners	length	Active
3	Airconditioner fittings	Samsung	Airconditioners	Item	Active
4	Nails	Mitshubishi	Airconditioners	Mass	Active

```
AdminHome ah;

   Product pr;

   int sel_row;



   /**

    * Creates new form AddProducts

    */

   public ViewProducts() {

      initComponents();

      jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

      loadBrand();
    loadCategory();

    loadProducts();

}


public ViewProducts(AdminHome ah, Product pr) {

    initComponents();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    loadBrand();

    loadCategory();

    loadProducts();

    this.ah = ah;

    this.pr = pr;

}


public void loadBrand() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `brand`;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }
        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
        jComboBox1.setModel(dcm);

    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void loadCategory() {
    try {
        ResultSet rs = MySQL.search("SELECT * FROM `category`;");
        Vector v = new Vector();
        v.add("Select");
        while (rs.next()) {
            v.add(rs.getString("name"));
        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
        jComboBox2.setModel(dcm);

    } catch (Exception e) {
        e.printStackTrace();

    }

}


public void clearFields() {

    jTextField1.setText("");

    jComboBox1.setSelectedIndex(0);

    jComboBox2.setSelectedIndex(0);


}


public void loadProducts() {

    String name = jTextField1.getText();

    String brand = jComboBox1.getSelectedItem().toString();

    String category = jComboBox2.getSelectedItem().toString();


    Vector queryv = new Vector();


    if (!name.isBlank()) {

        queryv.add("`product`.`name` LIKE '%" + name + "%' ");

    }
```

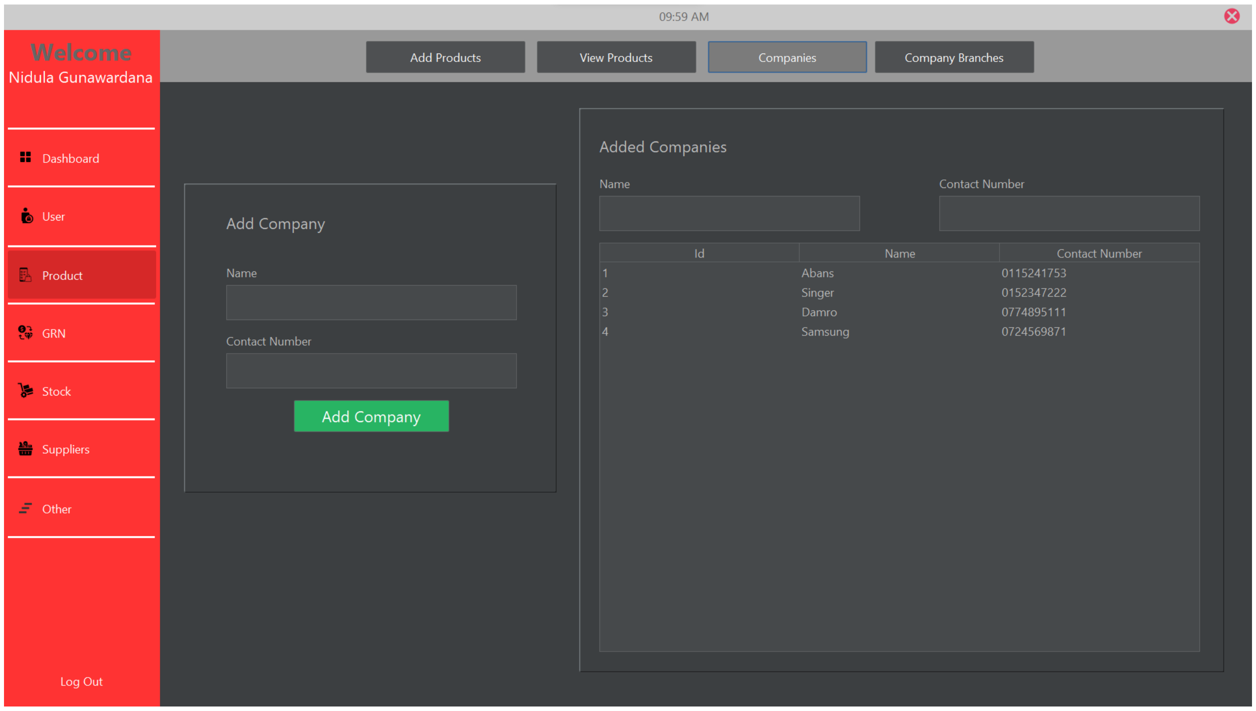
```
        if (!brand.equals("Select")) {
            queryv.add("`brand`.`name` = '" + brand + "' ");
        }

        if (!category.equals("Select")) {
            queryv.add("`category`.`name` = '" + category + "' ");
        }

        String query = "";

        for (int i = 0; i < queryv.size(); i++) {
            if (i == 0) {
                query += "WHERE ";
            }

            query += queryv.get(i);
            if (i + 1 != queryv.size()) {
                query += "AND ";
            }

        }
//        System.out.println(query);
```

```
        try {
            ResultSet rs = MySQL.search("SELECT * FROM `product` INNER JOIN `category` ON
`category`.`id` = `product`.`category_id` INNER JOIN `brand` ON `brand`.`id` =
`product`.`brand_id` INNER JOIN `qty_type` ON `qty_type`.`id` = `product`.`qty_type_id` INNER
JOIN `ststus` ON `ststus`.`id` = `product`.`ststus_id` " + query + " ORDER BY `product`.`id`;");

            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("product.id"));
                v.add(rs.getString("product.name"));
                v.add(rs.getString("brand.name"));
                v.add(rs.getString("category.name"));
                v.add(rs.getString("qty_type.name"));
                v.add(rs.getString("ststus.name"));
                dtm.addRow(v);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
private void jTextField1KeyReleased(java.awt.event.KeyEvent evt) {
        loadProducts();
    }

    private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
        loadProducts();
    }

    private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {
        loadProducts();
    }

    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                this.sel_row = selectedRow;
                updateChange uc = new updateChange(ah, false, this, 3);
                uc.setVisible(true);

            }
        }
```

```
   }

public void statusChange() {

     statusChangeAdmin aca = new statusChangeAdmin(ah, false, this, sel_row, 3);

     aca.setVisible(true);

  }


  public void updateProduct() {

     ToUpdateDataProduct tudp = new ToUpdateDataProduct();

     tudp.setProductId(jTable1.getValueAt(sel_row, 0).toString());

     tudp.setName(jTable1.getValueAt(sel_row, 1).toString());

     tudp.setBrand(jTable1.getValueAt(sel_row, 2).toString());

     tudp.setCategory(jTable1.getValueAt(sel_row, 3).toString());

     tudp.setQttyType(jTable1.getValueAt(sel_row, 4).toString());

     this.pr.loadForUpdate(tudp);


  }
```

Companies



Id	Name	Contact Number
1	Abans	0115241753
2	Singer	0152347222
3	Damro	0774895111
4	Samsung	0724569871

```
AdminHome ah;
    String toUpdateRow;


    /**
     * Creates new form AddProducts
     */
    public Companies() {
        initComponents();
        jButton5.setVisible(false);
        searchCompanies();
        jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));
```

```
}


public Companies(AdminHome ah) {

   initComponents();

   this.ah = ah;

   jButton5.setVisible(false);

   searchCompanies();

   jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

}


public void clearFields() {

   jTextField1.setText("");

   jTextField2.setText("");

   jButton5.setVisible(false);

   jButton4.setText("Add Company");

   jLabel2.setText("Add Company");

}


public void searchCompanies() {

   String name = jTextField3.getText();

   String contact = jTextField4.getText();
```

```
Vector queryv = new Vector();


if (!name.isBlank()) {

    queryv.add("`name` LIKE '%" + name + "%' ");

}


if (!contact.isBlank()) {

    queryv.add("`contact_no` LIKE '%" + contact + "%' ");

}


String query = "";


for (int i = 0; i < queryv.size(); i++) {

    if (i == 0) {

        query += "WHERE ";

    }


    query += queryv.get(i);

    if (i + 1 != queryv.size()) {

        query += "AND ";

    }
```

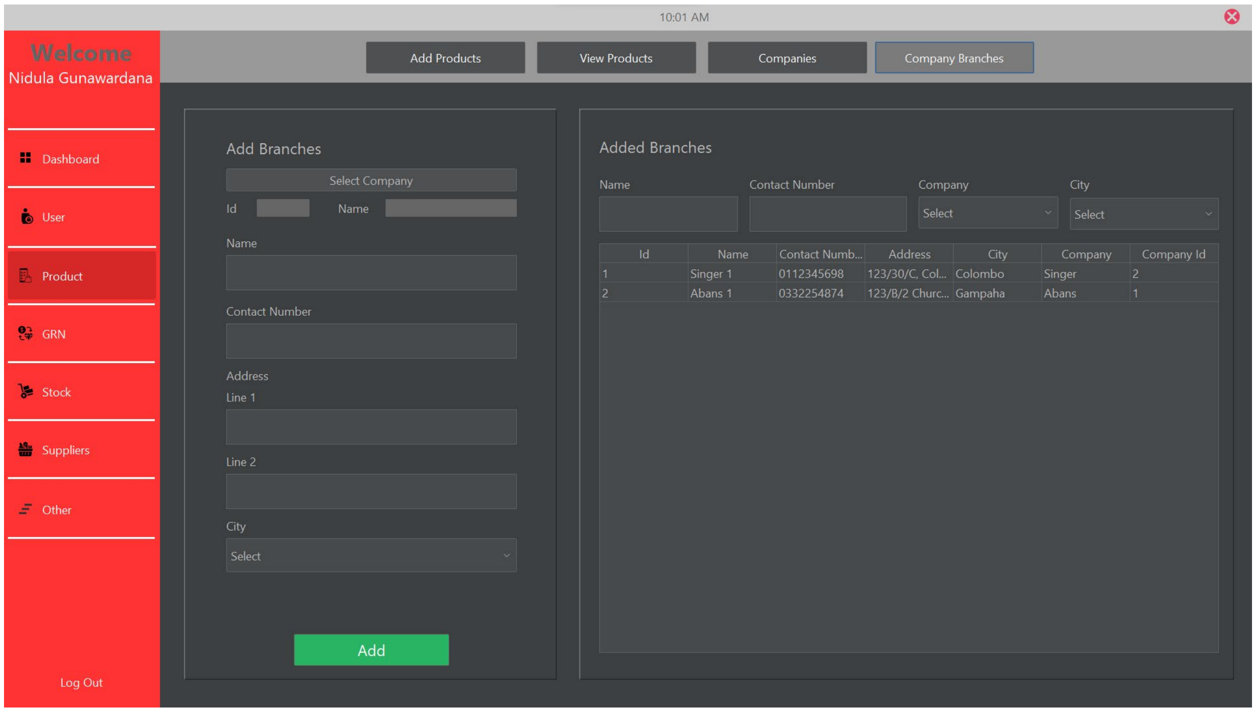
```
        }
        try {
            ResultSet rs = MySQL.search("SELECT * FROM `companies` " + query + " ORDER BY `id`;");


            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("id"));
                v.add(rs.getString("name"));
                v.add(rs.getString("contact_no"));


                dtm.addRow(v);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        String name = jTextField1.getText();
        String mobile = jTextField2.getText();
        if (name.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter The company name.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (mobile.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter The contact number.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (!Pattern.compile("[0-9]+").matcher(mobile).matches() || mobile.length() != 10) {
            JOptionPane.showMessageDialog(this, "Invalid Contact number.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            try {

                if (jButton4.getText().equals("Add Company")) {

                    //Search product if exists
                    ResultSet prs = MySQL.search("SELECT * FROM `companies` WHERE `name` ='" +
name + "' AND `contact_no` = '" + mobile + "';");
                    //Search product if exists
                    if (prs.next()) {
                        JOptionPane.showMessageDialog(this, "This company already exists.", "Warning",
JOptionPane.WARNING_MESSAGE);
                    } else {
```

```
                    MySQL.iud("INSERT INTO `companies`(`name`,`contact_no`) VALUES('" + name +
"','" + mobile + "');");
                    AddedSuccess as = new AddedSuccess(ah, false, "Added Successfully");
                    as.setVisible(true);
                    Runnable runnable = new Runnable() {
                        @Override
                        public void run() {
                            try {
                                Thread.sleep(1500);
                                as.dispose();
                            } catch (Exception e) {
                                e.printStackTrace();
                            }
                        }
                    };
                    Thread st = new Thread(runnable);
                    st.start();
                    clearFields();
                    searchCompanies();


                }
```

```
            } else {

                //Search product if exists

                ResultSet prs = MySQL.search("SELECT * FROM `companies` WHERE `name` ='" +
name + "' AND `contact_no` = '" + mobile + "' AND `id` NOT IN (" + toUpdateRow + ");");

                //Search product if exists

                if (prs.next()) {

                    JOptionPane.showMessageDialog(this, "This Company already exists.", "Warning",
JOptionPane.WARNING_MESSAGE);

                } else {

                    MySQL.iud("UPDATE `companies` SET `name`='" + name + "',`contact_no`='" +
mobile + "' WHERE `id` = '" + toUpdateRow + "';");

                    AddedSuccess as = new AddedSuccess(ah, false, "Updated Successfully");

                    as.setVisible(true);

                    Runnable runnable = new Runnable() {

                        @Override

                        public void run() {

                            try {

                                Thread.sleep(1500);

                                as.dispose();

                            } catch (Exception e) {

                                e.printStackTrace();

                            }
                    }

                  };

                  Thread st = new Thread(runnable);

                  st.start();

                  clearFields();

                  searchCompanies();


                }

              }

            } catch (Exception e) {

              e.printStackTrace();

            }


          }

        }


        private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

          clearFields();

        }


        private void jTextField2KeyTyped(java.awt.event.KeyEvent evt) {

          String mobile = jTextField2.getText();
```

```
    String text = mobile + evt.getKeyChar();


    if (mobile.length() == 10) {
        evt.consume();
    } else {
        if (!Pattern.compile("[0-9]+").matcher(text).matches()) {
            evt.consume();
        }
    }
}


private void jTextField4KeyTyped(java.awt.event.KeyEvent evt) {
    String mobile = jTextField4.getText();


    String text = mobile + evt.getKeyChar();


    if (mobile.length() == 10) {
        evt.consume();
    } else {
        if (!Pattern.compile("[0-9]+").matcher(text).matches()) {
            evt.consume();
            }
        }
    }

    private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {
        searchCompanies();
    }

    private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {
        searchCompanies();
    }

    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                int con = JOptionPane.showConfirmDialog(this, "Do you want To update?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
                if (con == JOptionPane.YES_OPTION) {
                    this.toUpdateRow = jTable1.getValueAt(selectedRow, 0).toString();
                    jTextField1.setText(jTable1.getValueAt(selectedRow, 1).toString());
                    jTextField2.setText(jTable1.getValueAt(selectedRow, 2).toString());
                    jButton4.setText("Update");

                    jButton5.setVisible(true);

                    jLabel2.setText("Update Company");

                }



            }

        }

    }
```

Branches



Id	Name	Contact Numb...	Address	City	Company	Company Id
1	Singer 1	0112345698	123/30/C, Col...	Colombo	Singer	2
2	Abans 1	0332254874	123/B/2 Churc...	Gampaha	Abans	1

```
AdminHome ah;

    String toUpdateRow;
```

```
/**
 * Creates new form AddProducts
 */
public CompanyBranches() {
    initComponents();
    jButton5.setVisible(false);
    loadCompany();
    loadCity();
    searchBranches();
    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));
}

public CompanyBranches(AdminHome ah) {
    initComponents();
    this.ah = ah;
    jButton5.setVisible(false);
    loadCompany();
    loadCity();
    searchBranches();
    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));
}
```

```
public void clearFields() {

    jTextField1.setText("");

    jTextField2.setText("");

    jTextField5.setText("");

    jTextField6.setText("");

    jComboBox1.setSelectedIndex(0);

    jLabel12.setText("");

    jLabel14.setText("");

    jButton5.setVisible(false);

    jButton4.setText("Add");

    jLabel2.setText("Add Branches");

}


public void loadCity() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `city` ;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
        jComboBox1.setModel(dcm);


        DefaultComboBoxModel dcm1 = new DefaultComboBoxModel(v);

        jComboBox3.setModel(dcm1);


    } catch (Exception e) {

        e.printStackTrace();

    }

}


public void loadCompany() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `companies` ;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox2.setModel(dcm);


    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void searchBranches() {
    String name = jTextField3.getText();
    String contact = jTextField4.getText();
    String company = jComboBox2.getSelectedItem().toString();
    String city = jComboBox3.getSelectedItem().toString();

    Vector queryv = new Vector();

    if (!name.isBlank()) {
        queryv.add("`company_branches`.`name` LIKE '%" + name + "%' ");
    }
    if (!contact.isBlank()) {
        queryv.add("`company_branches`.`co_number` LIKE '%" + contact + "%' ");
    }
    if (!company.equals("Select")) {
        queryv.add("`companies`.`name` LIKE '%" + company + "%' ");
    }
    if (!city.equals("Select")) {
        queryv.add("`city`.`name` LIKE '%" + city + "%' ");
    }


    String query = "";


    for (int i = 0; i < queryv.size(); i++) {
        if (i == 0) {
            query += "WHERE ";
        }


        query += queryv.get(i);
        if (i + 1 != queryv.size()) {
            query += "AND ";
        }


    }
    System.out.println(query);
    try {
        ResultSet rs = MySQL.search("SELECT * FROM `company_branches` INNER JOIN `city` ON
`city`.`id` = `company_branches`.`city_id` INNER JOIN `companies` ON `companies`.`id` =
`company_branches`.`companies_id` " + query + " ORDER BY `company_branches`.`id`;");
```

```
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

            dtm.setRowCount(0);

            while (rs.next()) {

                Vector v = new Vector();

                v.add(rs.getString("company_branches.id"));

                v.add(rs.getString("company_branches.name"));

                v.add(rs.getString("company_branches.co_number"));

                v.add(rs.getString("company_branches.line1") + ", " +
rs.getString("company_branches.line2"));

                v.add(rs.getString("city.name"));

                v.add(rs.getString("companies.name"));

                v.add(rs.getString("companies.id"));


                dtm.addRow(v);

            }

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

        String name = jTextField1.getText();

        String mobile = jTextField2.getText();
        String line1 = jTextField5.getText();

        String line2 = jTextField6.getText();

        String city = jComboBox1.getSelectedItem().toString();

        String company_id = jLabel12.getText();



        if (company_id.isBlank()) {

            JOptionPane.showMessageDialog(this, "Select The company.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (name.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter The company Branch name.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (mobile.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter The contact number.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (!Pattern.compile("[0-9]+").matcher(mobile).matches() || mobile.length() != 10) {

            JOptionPane.showMessageDialog(this, "Invalid Contact number.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (line1.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter The Address Line1.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (line2.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter The Address Line2.", "Warning",
JOptionPane.WARNING_MESSAGE);
```

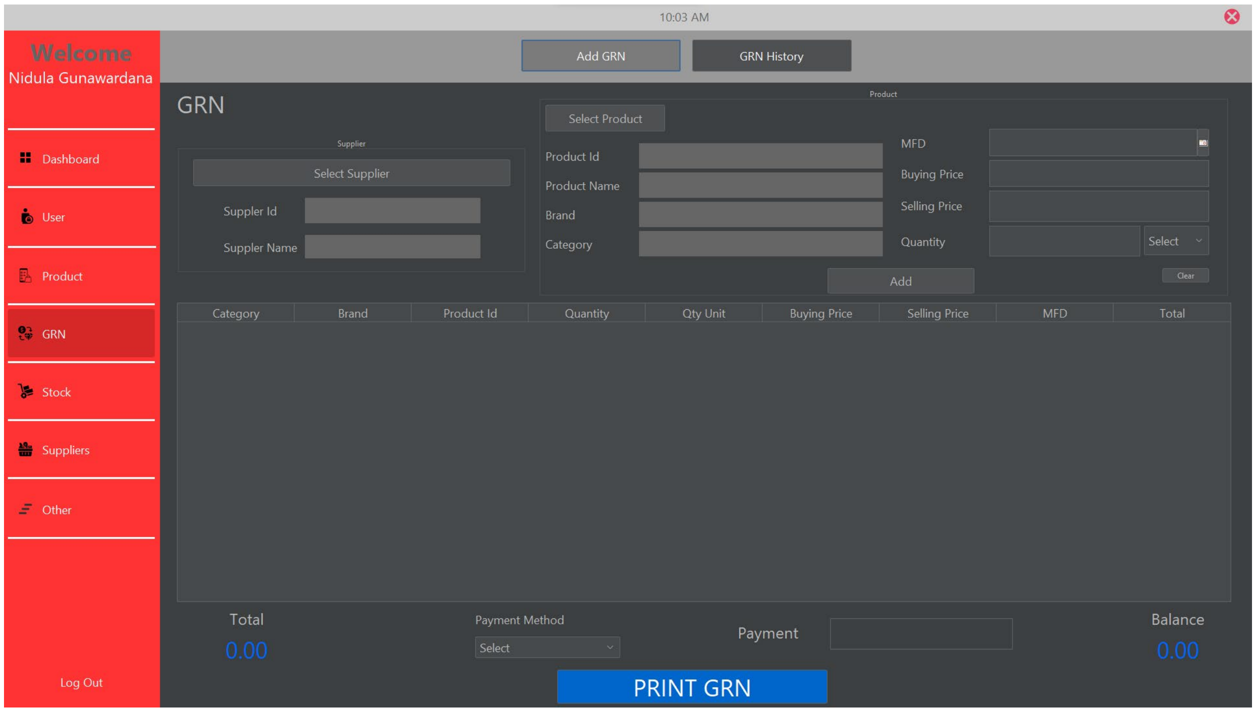
```
        } else if (city.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select City", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            try {
                //Search city
                ResultSet crs = MySQL.search("SELECT * FROM `city` WHERE `name` ='" + city + "';");
                crs.next();
                String city_id = crs.getString("id");
                //Search city
                if (jButton4.getText().equals("Add")) {

                    //Search product if exists
                    ResultSet prs = MySQL.search("SELECT * FROM `company_branches` WHERE `name`
='" + name + "' AND `co_number` = '" + mobile + "';");
                    //Search product if exists
                    if (prs.next()) {
                        JOptionPane.showMessageDialog(this, "This Branch already exists.", "Warning",
JOptionPane.WARNING_MESSAGE);
                    } else {
                        MySQL.iud("INSERT INTO
`company_branches`(`name`,`co_number`,`line1`,`line2`,`city_id`,`companies_id`) VALUES('" +
name + "','" + mobile + "','" + line1 + "','" + line2 + "','" + city_id + "','" + company_id + "');");
```

```
        AddedSuccess as = new AddedSuccess(ah, false, "Added Successfully");

        as.setVisible(true);

        Runnable runnable = new Runnable() {

            @Override

            public void run() {

                try {

                    Thread.sleep(1500);

                    as.dispose();

                } catch (Exception e) {

                    e.printStackTrace();

                }

            }

        };

        Thread st = new Thread(runnable);

        st.start();

        clearFields();

        searchBranches();


    }


} else {

    //Search product if exists
```

```
                ResultSet prs = MySQL.search("SELECT * FROM `company_branches` WHERE `name`
='" + name + "' AND `co_number` = '" + mobile + "' AND `id` NOT IN (" + toUpdateRow + ");");

                //Search product if exists

                if (prs.next()) {

                    JOptionPane.showMessageDialog(this, "This Branch already exists.", "Warning",
JOptionPane.WARNING_MESSAGE);

                } else {

                    MySQL.iud("UPDATE `company_branches` SET `name`='" + name +
"',`co_number`='" + mobile + "',`line1`='" + line1 + "',`line2`='" + line2 + "',`city_id`='" + city_id +
"',`companies_id`='" + company_id + "' WHERE `id` = '" + toUpdateRow + "';");

                    AddedSuccess as = new AddedSuccess(ah, false, "Updated Successfully");

                    as.setVisible(true);

                    Runnable runnable = new Runnable() {

                        @Override

                        public void run() {

                            try {

                                Thread.sleep(1500);

                                as.dispose();

                            } catch (Exception e) {

                                e.printStackTrace();

                            }

                        }

                    };
                Thread st = new Thread(runnable);

                st.start();

                clearFields();

                searchBranches();


              }

            }

        } catch (Exception e) {

            e.printStackTrace();

        }


    }

}


private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

    clearFields();

}


private void jTextField2KeyTyped(java.awt.event.KeyEvent evt) {

    String mobile = jTextField2.getText();


    String text = mobile + evt.getKeyChar();
    if (mobile.length() == 10) {
        evt.consume();
    } else {
        if (!Pattern.compile("[0-9]+").matcher(text).matches()) {
            evt.consume();
        }
    }
}

private void jTextField4KeyTyped(java.awt.event.KeyEvent evt) {
    String mobile = jTextField2.getText();

    String text = mobile + evt.getKeyChar();

    if (mobile.length() == 10) {
        evt.consume();
    } else {
        if (!Pattern.compile("[0-9]+").matcher(text).matches()) {
            evt.consume();
        }
    }
    }


    private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {
        searchBranches();
    }


    private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {
        searchBranches();
    }


    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                int con = JOptionPane.showConfirmDialog(this, "Do you want To update?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
                if (con == JOptionPane.YES_OPTION) {
                    this.toUpdateRow = jTable1.getValueAt(selectedRow, 0).toString();
                    jTextField1.setText(jTable1.getValueAt(selectedRow, 1).toString());
                    jTextField2.setText(jTable1.getValueAt(selectedRow, 2).toString());
                    jTextField5.setText(jTable1.getValueAt(selectedRow, 3).toString().split(",\\s")[0]);
                    jTextField6.setText(jTable1.getValueAt(selectedRow, 3).toString().split(",\\s")[1]);
                jComboBox1.setSelectedItem(jTable1.getValueAt(selectedRow, 4).toString());

                jLabel12.setText(jTable1.getValueAt(selectedRow, 6).toString());

                jLabel14.setText(jTable1.getValueAt(selectedRow, 5).toString());

                jButton4.setText("Update");

                jButton5.setVisible(true);

                jLabel2.setText("Update Branches");

            }


        }

    }

}


private void jTextField5KeyTyped(java.awt.event.KeyEvent evt) {

    String line1 = jTextField5.getText();


    String text = line1 + evt.getKeyChar();


    if (Pattern.compile("([\\w*\\W*]*[,])|([,])").matcher(text).matches()) {

        evt.consume();


    }

}
```

```
private void jTextField6KeyTyped(java.awt.event.KeyEvent evt) {
    String line1 = jTextField5.getText();


    String text = line1 + evt.getKeyChar();


    if (Pattern.compile("([\\w*\\W*]*[,])|([,])").matcher(text).matches()) {
        evt.consume();


    }
}


private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    SelectCompany sc = new SelectCompany(ah, true, this);
    sc.setVisible(true);
}


private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {
    searchBranches();
}


private void jComboBox3ItemStateChanged(java.awt.event.ItemEvent evt) {
        searchBranches();

    }
```

GRN



```
AdminHome ah;

    String toUpdateRow;

    ToUpdateDateStock tus;

    DecimalFormat df = new DecimalFormat("0.00");

    GRN st;



    /**

     * Creates new form AddProducts

     */
```

```
public AddGRN() {

    initComponents();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    loadQtyUnits();

    loadPaymentMethods();

}


public AddGRN(AdminHome ah) {

    initComponents();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    loadQtyUnits();

    loadPaymentMethods();

    this.ah = ah;

}


public AddGRN(AdminHome ah, ToUpdateDateStock tus, GRN st) {

    initComponents();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    this.ah = ah;

    this.tus = tus;

    this.st = st;

    loadPaymentMethods();
    loadQtyUnits(tus.getProductId());


}


public void clearFields() {

    jLabel26.setText("");

    jLabel27.setText("");

    jLabel29.setText("");

    jLabel31.setText("");

    jDateChooser2.setDate(null);

    jTextField5.setText("");

    jTextField6.setText("");

    jTextField7.setText("");

    loadQtyUnits();


}


public void loadPaymentMethods() {

    try {

        ResultSet rs = MySQL.search("SELECT * FROM `payment_type` ;");

        Vector v = new Vector();

        v.add("Select");
        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox3.setModel(dcm);


    } catch (Exception e) {

        e.printStackTrace();

    }

}


public void loadQtyUnits() {

    try {


        Vector v = new Vector();

        v.add("Select");


        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox2.setModel(dcm);


    } catch (Exception e) {

        e.printStackTrace();
        }

    }


    public void loadQtyUnits(String pid) {

        try {

            ResultSet rs = MySQL.search("(SELECT * FROM `qty_units` INNER JOIN `qty_type` ON
`qty_type`.`id` = `qty_units`.`qty_type_id` LEFT JOIN `product` ON `product`.`qty_type_id` =
`qty_type`.`id` WHERE `product`.`id` = '" + pid + "') UNION (SELECT * FROM `qty_units` INNER
JOIN `qty_type` ON `qty_type`.`id` = `qty_units`.`qty_type_id` RIGHT JOIN `product` ON
`product`.`qty_type_id` = `qty_type`.`id` WHERE `product`.`id` = '" + pid + "')");

            Vector v = new Vector();

            v.add("Select");

            while (rs.next()) {

                v.add(rs.getString("name"));

            }

            DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

            jComboBox2.setModel(dcm);


        } catch (Exception e) {

            e.printStackTrace();

        }

    }
    public void calculateTotal() {

        double tot = 0;

        for (int i = 0; i < jTable1.getRowCount(); i++) {

            tot = tot + Double.parseDouble(jTable1.getValueAt(i, 8).toString());

        }

        jLabel2.setText(new DecimalFormat("0.00").format(tot));

    }

private void jTextField5KeyTyped(java.awt.event.KeyEvent evt) {

        String qty = jTextField5.getText();



        String text = qty + evt.getKeyChar();



        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {

            evt.consume();



        }

    }



    private void jTextField6KeyReleased(java.awt.event.KeyEvent evt) {

        // TODO add your handling code here:

    }
    private void jTextField6KeyTyped(java.awt.event.KeyEvent evt) {
        String qty = jTextField6.getText();

        String text = qty + evt.getKeyChar();

        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {
            evt.consume();

        }
    }

    private void jTextField8KeyReleased(java.awt.event.KeyEvent evt) {
        if (!jTextField8.getText().isBlank()) {
            double payable = Double.parseDouble(jLabel2.getText());
            double payment = Double.parseDouble(jTextField8.getText());
            jLabel11.setText(new DecimalFormat("0.00").format(payable - payment));
        } else {
            jLabel11.setText("0.00");
        }
    }
    private void jTextField8KeyTyped(java.awt.event.KeyEvent evt) {
        String qty = jTextField8.getText();

        String text = qty + evt.getKeyChar();

        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {
            evt.consume();

        }
    }

    private void jTextField7KeyTyped(java.awt.event.KeyEvent evt) {
        String qty = jTextField7.getText();

        String text = qty + evt.getKeyChar();

        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {
            evt.consume();
        }

    }


    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        SelectSupplier ss = new SelectSupplier(ah, true, this);

        ss.setVisible(true);

    }


    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        SelectProduct sp = new SelectProduct(ah, true, this);

        sp.setVisible(true);

    }


    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

        String productId = jLabel26.getText();

        String productName = jLabel27.getText();

        String brand = jLabel29.getText();

        String category = jLabel31.getText();

        Date mfd = jDateChooser2.getDate();

        String buyingPrice = jTextField5.getText();

        String sellingPrice = jTextField6.getText();

        String qty = jTextField7.getText();
```

```
        String qtyUnit = jComboBox2.getSelectedItem().toString();


        if (productId.isBlank()) {
            JOptionPane.showMessageDialog(this, "Select The Product.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (mfd == null) {
            JOptionPane.showMessageDialog(this, "Select the MFD", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (buyingPrice.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Buying price", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(buyingPrice).matches()) {
            JOptionPane.showMessageDialog(this, "Invalid Buying price", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (sellingPrice.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Selling price", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(sellingPrice).matches()) {
            JOptionPane.showMessageDialog(this, "Invalid Selling price", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (qty.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Quantity.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (Double.parseDouble(buyingPrice) > Double.parseDouble(sellingPrice)) {
            JOptionPane.showMessageDialog(this, "Selling price is lower than buying price.",
"Warning", JOptionPane.WARNING_MESSAGE);
        } else if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(qty).matches()) {
            JOptionPane.showMessageDialog(this, "Invalid Quantity.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (qtyUnit.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select Quantity Unit.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            boolean isTruue = true;
            int row = -1;
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

            for (int i = 0; i < jTable1.getRowCount(); i++) {
                if (jTable1.getValueAt(i, 2).toString().equals(productId) && jTable1.getValueAt(i,
7).toString().equals(new SimpleDateFormat("yyyy-MM-dd").format(mfd))) {
                    isTruue = false;
                    row = i;
                }
            }

            try {

                ResultSet qtyUnitMul = MySQL.search("SELECT * FROM `qty_units` WHERE `name` = '"
+ qtyUnit + "';");

                qtyUnitMul.next();

                double mul = Double.parseDouble(qtyUnitMul.getString("multiplication"));

                if (isTruue) {



                    Vector v = new Vector();

                    v.add(category);

                    v.add(brand);

                    v.add(productId);

                    v.add(String.valueOf(Double.parseDouble(qty)));

                    v.add(qtyUnit);

                    v.add(buyingPrice);

                    v.add(sellingPrice);

                    v.add(new SimpleDateFormat("yyyy-MM-dd").format(mfd));

                    v.add(String.valueOf(Double.parseDouble(qty) * mul *
Double.parseDouble(buyingPrice)));

                    dtm.addRow(v);



                } else {
                    int con = JOptionPane.showConfirmDialog(this, "This Product Already Added. Do you
want to update the quantity?", "Warning", JOptionPane.YES_NO_OPTION,
JOptionPane.WARNING_MESSAGE);

                    if (con == JOptionPane.YES_OPTION) {

                        ResultSet qtyUnitMulOld = MySQL.search("SELECT * FROM `qty_units` WHERE
`name` = '" + jTable1.getValueAt(row, 4).toString() + "';");

                        qtyUnitMulOld.next();

                        double mulOld = Double.parseDouble(qtyUnitMulOld.getString("multiplication"));

                        double oldQty = Double.parseDouble(jTable1.getValueAt(row, 3).toString());

                        double newQty = oldQty + (Double.parseDouble(qty) * mul * mulOld);

                        jTable1.setValueAt(String.valueOf(newQty), row, 3);

                        jTable1.setValueAt(String.valueOf(newQty * Double.parseDouble(buyingPrice)),
row, 8);


                    }

                }

                clearFields();

                calculateTotal();

                System.gc();

            } catch (Exception e) {

                e.printStackTrace();

            }
```

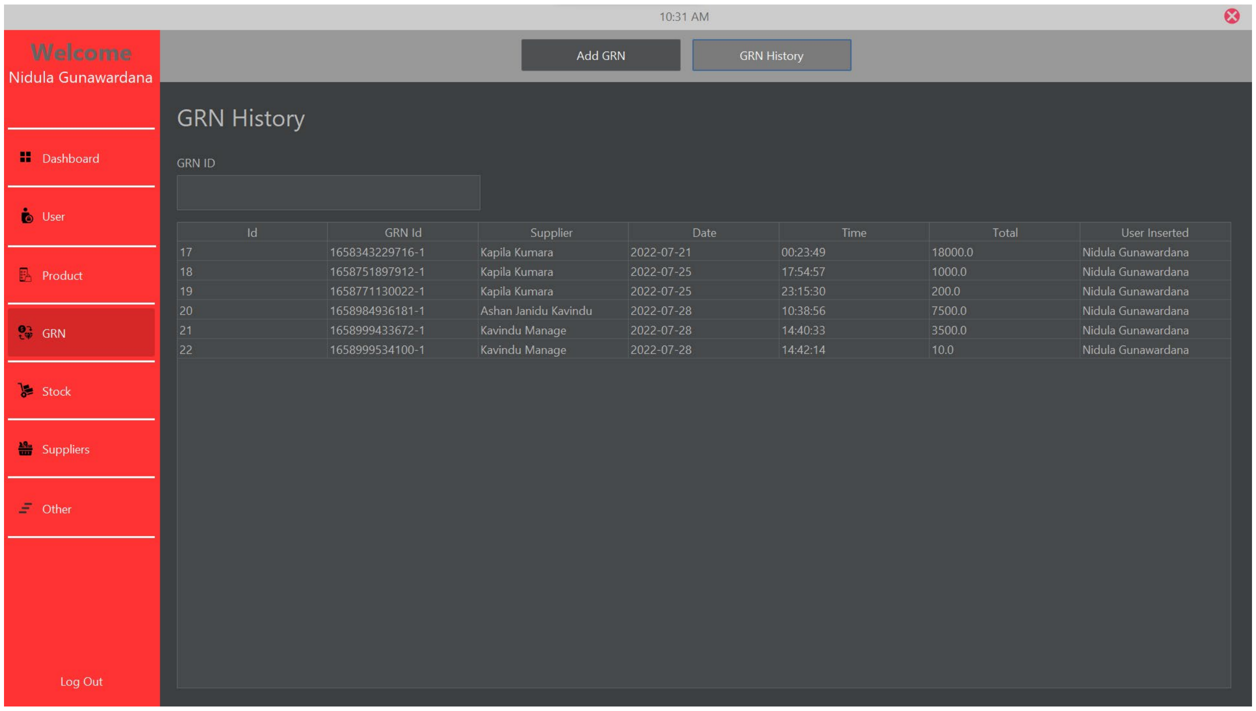
```
        }

    }

    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                int con = JOptionPane.showConfirmDialog(this, "Do you want To Remove?",
"Warning", JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
                if (con == JOptionPane.YES_OPTION) {
                    DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
                    dtm.removeRow(selectedRow);
                    calculateTotal();
                }
            }
        }
    }

    private void jComboBox3ItemStateChanged(java.awt.event.ItemEvent evt) {
        if (jComboBox3.getSelectedItem().toString().equals("Select")) {
            jTextField8.setEnabled(false);
        } else {
            jTextField8.setEnabled(true);
        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String supplierId = jLabel21.getText();
        String totalPayable = jLabel2.getText();
        String paymentType = jComboBox3.getSelectedItem().toString();
        String payment = jTextField8.getText();
        String balance = jLabel11.getText();

        if (supplierId.isBlank()) {
            JOptionPane.showMessageDialog(this, "Select Supplier.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (jTable1.getRowCount() == 0) {
            JOptionPane.showMessageDialog(this, "Please add atleast one product.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (payment.isBlank()) {
            JOptionPane.showMessageDialog(this, "Please Enter payment amount.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            if (!paymentType.equals("After Sold") && payment.equals("0")) {
                JOptionPane.showMessageDialog(this, "Invalid payment amount.", "Warning",
JOptionPane.WARNING_MESSAGE);
            } else {
                try {
                    long mTime = System.currentTimeMillis();
                    String uniqId = mTime + "-" + AdminSignIn.AdminId;
                    String dNow = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(new
Date());

                    MySQL.iud("INSERT INTO `grn`(`supplier_id`,`grn_id`,`date_time`,`user_id`)
VALUES('" + supplierId + "','" + uniqId + "','" + dNow + "','" + AdminSignIn.AdminId + "');");

                    ResultSet grs = MySQL.search("SELECT * FROM `grn` WHERE `grn_id` = '" + uniqId +
"';");
                    grs.next();
                    String grnId = grs.getString("id");

                    Vector vReport = new Vector();

                    for (int i = 0; i < jTable1.getRowCount(); i++) {

                        String productId = jTable1.getValueAt(i, 2).toString();
                        String qty = jTable1.getValueAt(i, 3).toString();

                        String buyingPrice = jTable1.getValueAt(i, 5).toString();

                        String sellingPrice = jTable1.getValueAt(i, 6).toString();

                        String mfd = jTable1.getValueAt(i, 7).toString();

                        String productTotal = jTable1.getValueAt(i, 8).toString();

                        String qtyUnit = jTable1.getValueAt(i, 4).toString();

                        //Qty unit Id

                        ResultSet quirs = MySQL.search("SELECT * FROM `qty_units` WHERE `name` = '" +
qtyUnit + "';");

                        quirs.next();

                        String qtyUnitId = quirs.getString("id");

                        //Qty Unit Id


                        String stockId;


                        //Search for stocks available

                        ResultSet stokRs = MySQL.search("SELECT * FROM `stock` WHERE `product_id` = '"
+ productId + "' AND `selling_price` = '" + sellingPrice + "' AND `mfd` = '" + mfd + "';");

                        if (stokRs.next()) {

                            stockId = stokRs.getString("id");

                            //Qty unit Id of stock
                        ResultSet quirsNew = MySQL.search("SELECT * FROM `qty_units` WHERE `id` =
'" + stokRs.getString("qty_units_id") + "';");

                        quirsNew.next();

                        double qtyUnitmulti =
Double.parseDouble(quirsNew.getString("multiplication"));

                        //Qty Unit Id of stock


                        double newQty = Double.parseDouble(stokRs.getString("qty")) +
(Double.parseDouble(qty) * Double.parseDouble(quirs.getString("multiplication")) *
qtyUnitmulti);


                        MySQL.iud("UPDATE `stock` SET `qty` = '" + String.valueOf(newQty) + "' WHERE
`id` = '" + stokRs.getString("id") + "';");


                } else {


                        MySQL.iud("INSERT INTO
`stock`(`selling_price`,`qty`,`qty_units_id`,`mfd`,`product_id`) VALUES('" +
df.format(Double.parseDouble(sellingPrice)) + "','" + df.format(Double.parseDouble(qty)) + "','"
+ qtyUnitId + "','" + mfd + "','" + productId + "');");


                        ResultSet newStoRs = MySQL.search("SELECT * FROM `stock` WHERE
`product_id` = '" + productId + "' AND `selling_price` = '" +
df.format(Double.parseDouble(sellingPrice)) + "' AND `mfd` = '" + mfd + "';");
```

```
                        newStoRs.next();

                        stockId = newStoRs.getString("id");

                    }
                    //Search for stocks available

                    MySQL.iud("INSERT INTO
`grn_items`(`grn_id`,`qty`,`qty_units_id`,`buying_price`,`stock_id`) VALUES('" + grnId + "','" + qty
+ "','" + qtyUnitId + "','" + buyingPrice + "','" + stockId + "')");
                    MySQL.iud("INSERT INTO
`stock_changes`(`description`,`date_time`,`stock_id`,`qty`,`qty_units_id`) VALUES('Added To
Stock From The GRN -" + uniqId + "','" + dNow + "','" + stockId + "','" + qty + "','" + qtyUnitId +
"');");

                    ResultSet proRs = MySQL.search("SELECT * FROM `product` WHERE `id` = '" +
jTable1.getValueAt(i, 2).toString() + "';");
                    proRs.next();

                    vReport.add(new GrnPaymentData(proRs.getString("name"), qty + qtyUnit, "Rs. "
+ buyingPrice, "Rs. " + productTotal));

                }
                //grn Payment type

                ResultSet paymentTypeRs = MySQL.search("SELECT * FROM `payment_type` WHERE
`name` = '" + paymentType + "';");

                paymentTypeRs.next();

                String paymentTypeId = paymentTypeRs.getString("id");

                //grn Payment type



                MySQL.iud("INSERT INTO
`grn_payments`(`grn_id`,`payment`,`balance`,`payment_type_id`) VALUES('" + grnId + "','" +
payment + "','" + balance + "','" + paymentTypeId + "');");

                MySQL.iud("INSERT INTO `money_changes`(`description`,`price`,`date_time`)
VALUES('Total Price For the GRN - " + uniqId + "','" +
String.valueOf(Double.parseDouble(jLabel2.getText()) * -1) + "','" + dNow + "');");



                //Reporting Part

                ResultSet supRs = MySQL.search("SELECT * FROM `supplier` INNER JOIN
`company_branches` ON `company_branches`.`id` = `supplier`.`company_branches_id` INNER
JOIN `city` ON `city`.`id` = `company_branches`.`city_id` WHERE `supplier`.`id` = '" + supplierId +
"';");

                supRs.next();

                JRBeanCollectionDataSource datasourse = new
JRBeanCollectionDataSource(vReport);

                HashMap parameters = new HashMap();
                parameters.put("grnid", uniqId);

                parameters.put("grndate", dNow.split("\\s")[0]);

                parameters.put("user", AdminSignIn.AdminId);

                parameters.put("supName", jLabel24.getText());

                parameters.put("subtotal", totalPayable);

                parameters.put("payment", payment);

                parameters.put("balance", balance);

                parameters.put("paymentmethod", paymentType);

                parameters.put("address", supRs.getString("company_branches.line1") + ", " +
supRs.getString("company_branches.line2") + ", " + supRs.getString("city.name"));

                parameters.put("mobile", supRs.getString("supplier.contact_no"));

                parameters.put("CollectionBeanDatasourse", datasourse);


                String filepath = "src//reports//grnNshop.jasper";

                JasperPrint jp = JasperFillManager.fillReport(filepath, parameters, new
JREmptyDataSource());


                //Reporting Part

                AddedSuccess as = new AddedSuccess(ah, false, "GRN Added Successfully");

                as.setVisible(true);

                Runnable runnable = new Runnable() {

                    @Override
```

```
    public void run() {
        try {
            Thread.sleep(1500);
            as.dispose();
            JasperViewer.viewReport(jp, false);
            JasperPrintManager.printReport(jp, true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
};
Thread st = new Thread(runnable);
st.start();
DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
dtm.setRowCount(0);
jLabel21.setText("");
jLabel24.setText("");
jLabel2.setText("0.00");
jComboBox3.setSelectedIndex(0);
jTextField8.setText("");
jLabel11.setText("0.00");
System.gc();
```

```
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}


private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    jLabel26.setText("");
    jLabel27.setText("");
    jLabel29.setText("");
    jLabel31.setText("");
    jDateChooser2.setDate(null);
    jTextField5.setText("");
    jTextField6.setText("");
    jTextField7.setText("");
    loadQtyUnits();
}
```

GRN History



GRN History

GRN ID

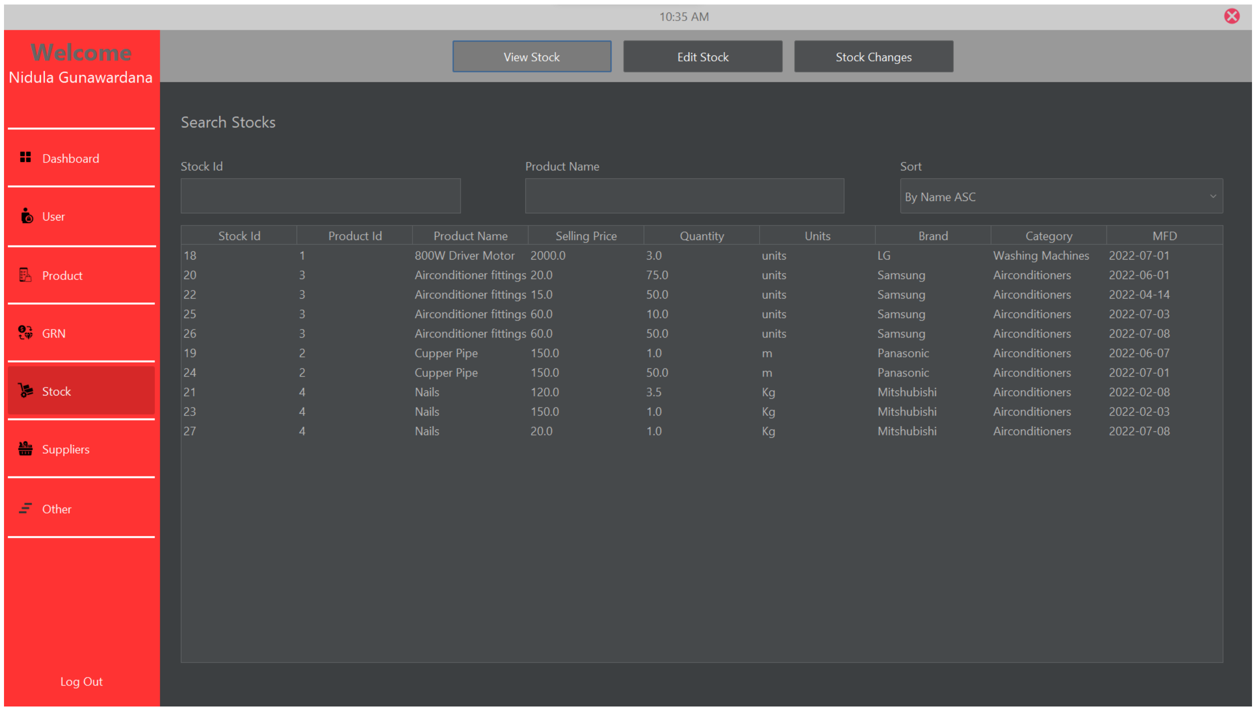
Id	GRN Id	Supplier	Date	Time	Total	User Inserted
17	1658343229716-1	Kapila Kumara	2022-07-21	00:23:49	18000.0	Nidula Gunawardana
18	1658751897912-1	Kapila Kumara	2022-07-25	17:54:57	1000.0	Nidula Gunawardana
19	1658771130022-1	Kapila Kumara	2022-07-25	23:15:30	200.0	Nidula Gunawardana
20	1658984936181-1	Ashan Janidu Kavindu	2022-07-28	10:38:56	7500.0	Nidula Gunawardana
21	1658999433672-1	Kavindu Manage	2022-07-28	14:40:33	3500.0	Nidula Gunawardana
22	1658999534100-1	Kavindu Manage	2022-07-28	14:42:14	10.0	Nidula Gunawardana

```
AdminHome ah;

    String toUpdateRow;

    ToUpdateDateStock tus;

    DecimalFormat df = new DecimalFormat("0.00");

    GRN st;



    /**

     * Creates new form AddProducts

     */

    public GrnHistory() {

        initComponents();
    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    loadGrnHistory();

}


public GrnHistory(AdminHome ah) {

    initComponents();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    this.ah = ah;

    loadGrnHistory();

}


public void clearFields() {



}


public void loadGrnHistory() {

    String grnId = jTextField1.getText();



    Vector queryv = new Vector();



    if (!grnId.isBlank()) {

        queryv.add("`grn`.`grn_id` LIKE '" + grnId + "%' ");
        }


        String query = "";


        for (int i = 0; i < queryv.size(); i++) {
            if (i == 0) {
                query += "WHERE ";
            }


            query += queryv.get(i);
            if (i + 1 != queryv.size()) {
                query += "AND ";
            }


        }
        System.out.println(query);
        try {
            ResultSet rs = MySQL.search("SELECT * FROM `grn` INNER JOIN `supplier` ON
`supplier`.`id` =  `grn`.`supplier_id` INNER JOIN `user` ON `user`.`id` = `grn`.`user_id` INNER JOIN
`grn_payments` ON `grn_payments`.`grn_id` = `grn`.`id` " + query + " ORDER BY
`grn`.`date_time` ASC;");
```

```
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

            dtm.setRowCount(0);

            while (rs.next()) {

                Vector v = new Vector();

                v.add(rs.getString("grn.id"));

                v.add(rs.getString("grn.grn_id"));

                v.add(rs.getString("supplier.fname") + " " + rs.getString("supplier.lname"));

                v.add(rs.getString("grn.date_time").split("\\s")[0]);

                v.add(rs.getString("grn.date_time").split("\\s")[1]);

                double tot = Double.parseDouble(rs.getString("grn_payments.payment")) +
Double.parseDouble(rs.getString("grn_payments.balance"));

                v.add(String.valueOf(tot));

                v.add(rs.getString("user.fname") + " " + rs.getString("user.lname"));



                dtm.addRow(v);

            }

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

  private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();

            if (selectedRow != -1) {

                ViewGrnItems vgi = new ViewGrnItems(ah, true, jTable1.getValueAt(selectedRow,
0).toString());

                vgi.jLabel26.setText(jTable1.getValueAt(selectedRow, 1).toString());

                vgi.jLabel27.setText(jTable1.getValueAt(selectedRow, 2).toString());

                vgi.jLabel30.setText(jTable1.getValueAt(selectedRow, 3).toString());

                vgi.jLabel32.setText(jTable1.getValueAt(selectedRow, 4).toString());

                vgi.jLabel34.setText(jTable1.getValueAt(selectedRow, 6).toString());

                vgi.setVisible(true);

            }

        }

    }


    private void jTextField1KeyReleased(java.awt.event.KeyEvent evt) {

        loadGrnHistory();

    }
```

Stock

View Stocks



Search Stocks

Stock Id

Product Name

Sort: By Name ASC

Stock Id	Product Id	Product Name	Selling Price	Quantity	Units	Brand	Category	MFD
18	1	800W Driver Motor	2000.0	3.0	units	LG	Washing Machines	2022-07-01
20	3	Airconditioner fittings	20.0	75.0	units	Samsung	Airconditioners	2022-06-01
22	3	Airconditioner fittings	15.0	50.0	units	Samsung	Airconditioners	2022-04-14
25	3	Airconditioner fittings	60.0	10.0	units	Samsung	Airconditioners	2022-07-03
26	3	Airconditioner fittings	60.0	50.0	units	Samsung	Airconditioners	2022-07-08
19	2	Cupper Pipe	150.0	1.0	m	Panasonic	Airconditioners	2022-06-07
24	2	Cupper Pipe	150.0	50.0	m	Panasonic	Airconditioners	2022-07-01
21	4	Nails	120.0	3.5	Kg	Mitshubishi	Airconditioners	2022-02-08
23	4	Nails	150.0	1.0	Kg	Mitshubishi	Airconditioners	2022-02-03
27	4	Nails	20.0	1.0	Kg	Mitshubishi	Airconditioners	2022-07-08

```
 AdminHome ah;

  String toUpdateRow;

  Stock st;



  /**

   * Creates new form AddProducts

   */

  public ViewStock() {

      initComponents();

      searchStock();
```

```
    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

}


public ViewStock(AdminHome ah) {

    initComponents();

    this.ah = ah;

    searchStock();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

}


public ViewStock(AdminHome ah, Stock st) {

    initComponents();

    this.ah = ah;

    searchStock();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    this.st = st;

}


public void clearFields() {

    jTextField3.setText("");

    jTextField4.setText("");

    jComboBox1.setSelectedIndex(0);
```

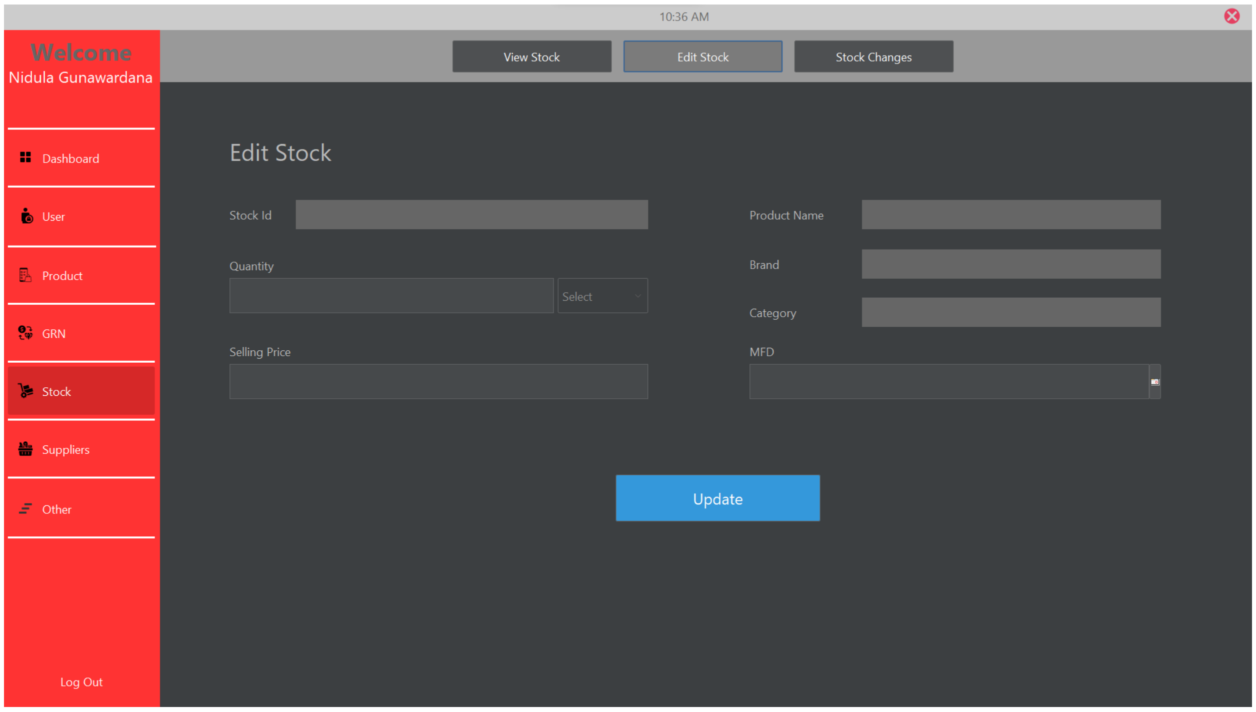
```
}

public void searchStock() {
    String stock_id = jTextField3.getText();
    String P_name = jTextField4.getText();
    String sort = jComboBox1.getSelectedItem().toString();

    Vector queryv = new Vector();

    if (!stock_id.isBlank()) {
        queryv.add("`stock`.`id` = '" + stock_id + "' ");
    }

    if (!P_name.isBlank()) {
        queryv.add("`product`.`name` LIKE '%" + P_name + "%' ");
    }
    String sortQue = "";
    if (sort.equals("By Name ASC")) {
        sortQue = "ORDER BY `product`.`name` ASC";
    } else if (sort.equals("By Name DESC")) {
        sortQue = "ORDER BY `product`.`name` DESC";
    } else if (sort.equals("By MFD ASC")) {
```

```
    sortQue = "ORDER BY `stock`.`mfd` ASC";

} else if (sort.equals("By MFD DESC")) {

    sortQue = "ORDER BY `stock`.`mfd` DESC";

} else if (sort.equals("By Quantity ASC")) {

    sortQue = "ORDER BY `stock`.`qty` DESC";

} else if (sort.equals("By Quantity DESC")) {

    sortQue = "ORDER BY `stock`.`qty` DESC";

}


String query = "";


for (int i = 0; i < queryv.size(); i++) {

    if (i == 0) {

        query += "WHERE ";

    }


    query += queryv.get(i);

    if (i + 1 != queryv.size()) {

        query += "AND ";

    }


}
```

```
//       System.out.println(query);

       try {

           ResultSet rs = MySQL.search("SELECT * FROM `stock` INNER JOIN `product` ON
`product`.`id` = `stock`.`product_id` INNER JOIN `brand` ON `brand`.`id` = `product`.`brand_id`
INNER JOIN `category` ON `category`.`id` = `product`.`category_id` INNER JOIN `qty_units` ON
`qty_units`.`id` = `stock`.`qty_units_id` " + query + " " + sortQue + ";");


           DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

           dtm.setRowCount(0);

           while (rs.next()) {

              Vector v = new Vector();

              v.add(rs.getString("stock.id"));

              v.add(rs.getString("product.id"));

              v.add(rs.getString("product.name"));

              v.add(rs.getString("stock.selling_price"));

              v.add(rs.getString("stock.qty"));

              v.add(rs.getString("qty_units.name"));

              v.add(rs.getString("brand.name"));

              v.add(rs.getString("category.name"));

              v.add(rs.getString("stock.mfd"));

              dtm.addRow(v);

           }
```

```
        } catch (Exception e) {

            e.printStackTrace();

        }

    }

private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {

        searchStock();

    }


    private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {

        searchStock();

    }


    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

        if (evt.getClickCount() == 2) {

            int selectedRow = jTable1.getSelectedRow();

            if (selectedRow != -1) {

                int con = JOptionPane.showConfirmDialog(this, "Do you want To update?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);

                if (con == JOptionPane.YES_OPTION) {

                    this.toUpdateRow = jTable1.getValueAt(selectedRow, 0).toString();

                    ToUpdateDateStock tus = new ToUpdateDateStock();

                    tus.setStock_id(toUpdateRow);
                    tus.setProductId(jTable1.getValueAt(selectedRow, 1).toString());

                    tus.setProductName(jTable1.getValueAt(selectedRow, 2).toString());

                    tus.setSellingPrice(jTable1.getValueAt(selectedRow, 3).toString());

                    tus.setQuantity(jTable1.getValueAt(selectedRow, 4).toString());

                    tus.setQty_unit(jTable1.getValueAt(selectedRow, 5).toString());

                    tus.setBrand(jTable1.getValueAt(selectedRow, 6).toString());

                    tus.setCategory(jTable1.getValueAt(selectedRow, 7).toString());

                    tus.setMfd(jTable1.getValueAt(selectedRow, 8).toString());

                    this.st.loadForUpdate(tus);

                }


            }

        }

    }


    private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {

        searchStock();

    }
```

Edit Stocks

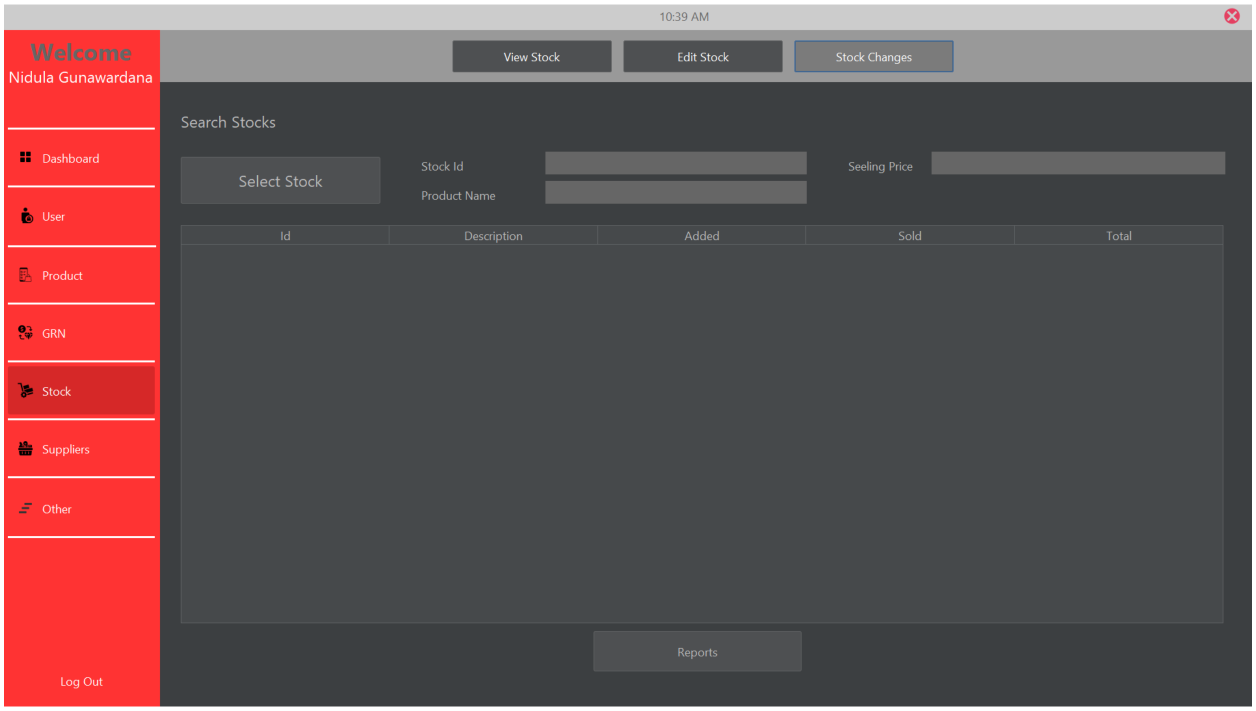


```
AdminHome ah;

    String toUpdateRow;

    ToUpdateDateStock tus;

    DecimalFormat df = new DecimalFormat("0.00");

    Stock st;



    public EditStock() {

        initComponents();

        loadQtyUnits();

    }

    public EditStock(AdminHome ah) {
        initComponents();

        loadQtyUnits();

        this.ah = ah;

    }

    public EditStock(AdminHome ah, ToUpdateDateStock tus, Stock st) {

        try {

            initComponents();

            this.ah = ah;

            this.tus = tus;

            this.st = st;

            loadQtyUnits(tus.getProductId());

            jLabel14.setText(tus.getStock_id());

            jLabel16.setText(tus.getProductName());

            jLabel18.setText(tus.getBrand());

            jLabel20.setText(tus.getCategory());

            jTextField3.setText(df.format(Double.parseDouble(tus.getQuantity())));

            jComboBox1.setSelectedItem(tus.getQty_unit());

            jTextField4.setText(df.format(Double.parseDouble(tus.getSellingPrice())));

            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

            jDateChooser1.setDate(sdf.parse(tus.getMfd()));

        } catch (ParseException ex) {

            Logger.getLogger(EditStock.class.getName()).log(Level.SEVERE, null, ex);
        }

    }

    public void clearFields() {

        jLabel14.setText("");

        jLabel16.setText("");

        jLabel18.setText("");

        jLabel20.setText("");

        jTextField3.setText("");

        jTextField4.setText("");

        loadQtyUnits();

        jComboBox1.setSelectedIndex(0);

        jDateChooser1.setDate(null);

    }

    public void loadQtyUnits() {

        try {

            ResultSet rs = MySQL.search("SELECT * FROM `qty_units` ;");

            Vector v = new Vector();

            v.add("Select");

            while (rs.next()) {

                v.add(rs.getString("name"));

            }

            DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
```

```
            jComboBox1.setModel(dcm);


        } catch (Exception e) {
            e.printStackTrace();
        }
    }


    public void loadQtyUnits(String pid) {
        try {
            ResultSet rs = MySQL.search("SELECT * FROM `qty_units` LEFT JOIN `qty_type` ON
`qty_type`.`id` = `qty_units`.`qty_type_id` LEFT JOIN `product` ON `product`.`qty_type_id` =
`qty_type`.`id` WHERE `product`.`id` = '" + pid + "';");
            Vector v = new Vector();
            v.add("Select");
            while (rs.next()) {
                v.add(rs.getString("name"));
            }
            DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
            jComboBox1.setModel(dcm);


        } catch (Exception e) {
            e.printStackTrace();
        }

    }

private void jTextField4KeyTyped(java.awt.event.KeyEvent evt) {

        String qty = jTextField4.getText();


        String text = qty + evt.getKeyChar();


        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {

            evt.consume();


        }

    }


    private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {

        String qty = jTextField3.getText();


        String text = qty + evt.getKeyChar();


        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {

            evt.consume();
        }
    }

    private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String stockIs = jLabel14.getText();
        String qty = jTextField3.getText();
        String qty_unit = jComboBox1.getSelectedItem().toString();
        String sprice = jTextField4.getText();
        Date mfd = jDateChooser1.getDate();

        if (stockIs.isBlank()) {
            JOptionPane.showMessageDialog(this, "Select Stock", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (qty.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Qty", "Warning",
JOptionPane.WARNING_MESSAGE);
```

```
        } else if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(qty).matches()) {

            JOptionPane.showMessageDialog(this, "Invalid Qty Format", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (qty_unit.equals("Select")) {

            JOptionPane.showMessageDialog(this, "Select Qty Unit", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (sprice.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter selling price", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(sprice).matches()) {

            JOptionPane.showMessageDialog(this, "Invalid Selling Price Format", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (mfd == null) {

            JOptionPane.showMessageDialog(this, "Select MFD", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else {

            try {

                //Qty unit search

                ResultSet qu = MySQL.search("SELECT * FROM `qty_units` WHERE `name` = '" +
qty_unit + "';");

                qu.next();

                String qtyUnitId = qu.getString("id");
                //Qty unit search

                //search previous qty details

                ResultSet prqrs = MySQL.search("SELECT * FROM `stock` INNER JOIN `qty_units` ON
`qty_units`.`id` = `stock`.`qty_units_id` WHERE `stock`.`id` = '" + stockIs + "';");

                prqrs.next();

                double prevQty = Double.parseDouble(prqrs.getString("stock.qty"));

                //search previous qty details

                SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

                MySQL.iud("UPDATE `stock` SET `selling_price` = '" + sprice + "',`qty` = '" + qty +
"',`qty_units_id` = '" + qtyUnitId + "',`mfd` = '" + sdf.format(mfd) + "' WHERE `id` = '" + stockIs +
"';");


                double updatedQty = Double.parseDouble(qty) - prevQty;

                if (updatedQty != 0) {

                    MySQL.iud("INSERT INTO
`stock_changes`(`description`,`date_time`,`stock_id`,`qty`,`qty_units_id`) VALUES('Quantity
changed by user -" + AdminSignIn.AdminId + "','" + new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss").format(new Date()) + "','" + stockIs + "','" + String.valueOf(updatedQty) + "','" +
qtyUnitId + "');");

                }

                AddedSuccess as = new AddedSuccess(ah, false, "Updated Successfully");

                as.setVisible(true);

                Runnable runnable = new Runnable() {
                    @Override
                    public void run() {
                        try {
                            Thread.sleep(1500);
                            st.resetToview();
                            as.dispose();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                };
                Thread st = new Thread(runnable);
                st.start();
                clearFields();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
```

Stock Changes



```
AdminHome ah;

   String toUpdateRow;

   Stock st;



   /**

    * Creates new form AddProducts

    */

   public ViewStockChanges() {

      initComponents();

      jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

   }
```

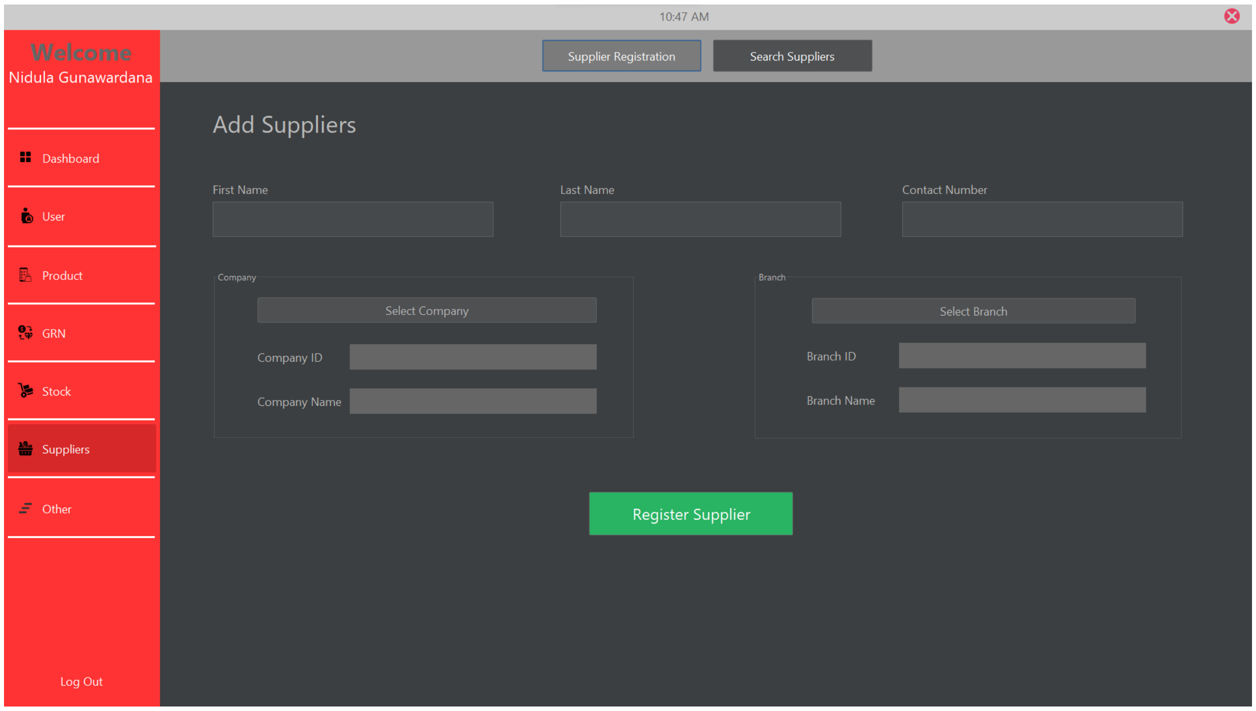
```
public ViewStockChanges(AdminHome ah) {

    initComponents();

    this.ah = ah;

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

}


public ViewStockChanges(AdminHome ah, Stock st) {

    initComponents();

    this.ah = ah;

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    this.st = st;

}


public void clearFields() {



}



public void searchStock(String stId) {



    try {

        double totalQty = 0;
        double newQty = 0;

        ResultSet rs = MySQL.search("SELECT * FROM `stock_changes` INNER JOIN `stock` ON
`stock`.`id` = `stock_changes`.`stock_id` INNER JOIN `qty_units` ON `qty_units`.`id` =
`stock_changes`.`qty_units_id` WHERE `stock`.`id` = '" + stId + "' ORDER BY
`stock_changes`.`date_time` DESC;");


        DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

        dtm.setRowCount(0);

        while (rs.next()) {

          if (totalQty == 0) {

            totalQty = Double.parseDouble(rs.getString("stock.qty"));

            newQty = totalQty;

          }

          Vector v = new Vector();


          if (Double.parseDouble(rs.getString("stock_changes.qty")) < 0) {

            //Qty unit Id

            ResultSet quirs = MySQL.search("SELECT * FROM `qty_units` WHERE `id` = '" +
rs.getString("stock_changes.qty_units_id") + "';");

            quirs.next();

            String mul = quirs.getString("multiplication");

            //Qty Unit Id

            v.add(rs.getString("stock_changes.id"));
```

```
                v.add(rs.getString("stock_changes.description"));

                v.add("");

                v.add(rs.getString("stock_changes.qty") + " " + rs.getString("qty_units.name"));

                v.add(String.valueOf(newQty));

                newQty = newQty - (Double.parseDouble(rs.getString("stock_changes.qty")) *
Double.parseDouble(mul));


            } else if (Double.parseDouble(rs.getString("stock_changes.qty")) > 0) {

                //Qty unit Id

                ResultSet quirs = MySQL.search("SELECT * FROM `qty_units` WHERE `id` = '" +
rs.getString("stock_changes.qty_units_id") + "';");

                quirs.next();

                String mul = quirs.getString("multiplication");

                //Qty Unit Id

                v.add(rs.getString("stock_changes.id"));

                v.add(rs.getString("stock_changes.description"));

                v.add(rs.getString("stock_changes.qty") + " " + rs.getString("qty_units.name"));

                v.add("");

                v.add(String.valueOf(newQty));

                newQty = newQty - (Double.parseDouble(rs.getString("stock_changes.qty")) *
Double.parseDouble(mul));

            }
```

```
                dtm.addRow(v);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                int con = JOptionPane.showConfirmDialog(this, "Do you want To update?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
                if (con == JOptionPane.YES_OPTION) {
                    this.toUpdateRow = jTable1.getValueAt(selectedRow, 0).toString();
                    ToUpdateDateStock tus = new ToUpdateDateStock();
                    tus.setStock_id(toUpdateRow);
                    tus.setProductId(jTable1.getValueAt(selectedRow, 1).toString());
                    tus.setProductName(jTable1.getValueAt(selectedRow, 2).toString());
                    tus.setSellingPrice(jTable1.getValueAt(selectedRow, 3).toString());
                    tus.setQuantity(jTable1.getValueAt(selectedRow, 4).toString());
                    tus.setQty_unit(jTable1.getValueAt(selectedRow, 5).toString());
                    tus.setBrand(jTable1.getValueAt(selectedRow, 6).toString());
                    tus.setCategory(jTable1.getValueAt(selectedRow, 7).toString());

                    tus.setMfd(jTable1.getValueAt(selectedRow, 8).toString());

                    this.st.loadForUpdate(tus);

                }

            }

        }

    }


    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        SelectStock ss = new SelectStock(ah, true, this);

        ss.setVisible(true);

    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        if (jLabel14.getText().isBlank()) {

            StockReports sr = new StockReports(ah, false);

            sr.setVisible(true);

        } else {

            StockReports sr = new StockReports(ah, false, jLabel14.getText(), jLabel15.getText());

            sr.setVisible(true);

        }

    }
```

Suppliers

Supplier registration



```
AdminHome ah;
    ToUpdateDataSuppliers tudp;
    Supplier su;


    public AddSuppliers() {
        initComponents();
        jButton5.setVisible(false);
    }


    public AddSuppliers(AdminHome ah) {
```

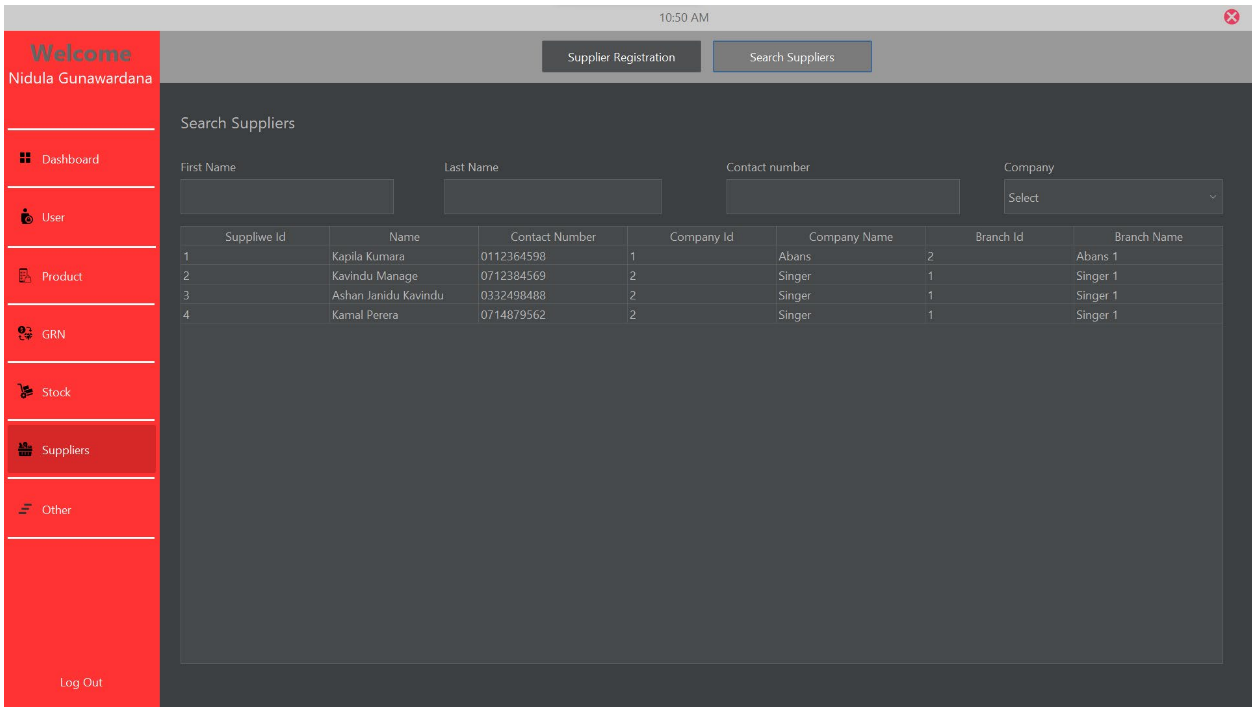
```
    initComponents();

    this.ah = ah;

    jButton5.setVisible(false);

}


public AddSuppliers(AdminHome ah, ToUpdateDataSuppliers tudp, Supplier su) {

    initComponents();

    this.ah = ah;

    jTextField1.setText(tudp.getFname());

    jTextField2.setText(tudp.getLname());

    jTextField3.setText(tudp.getContact());

    jLabel26.setText(tudp.getCompanyId());

    jLabel28.setText(tudp.getCompanyName());

    jLabel30.setText(tudp.getBranchId());

    jLabel32.setText(tudp.getBranchName());

    jButton4.setText("Update");

    jLabel2.setText("Update Supplier");

    this.tudp = tudp;

    this.su = su;

}

public void clearFields() {

    jTextField1.setText("");
```

```
        jTextField2.setText("");

        jTextField3.setText("");

        jLabel26.setText("");

        jLabel28.setText("");

        jLabel30.setText("");

        jLabel32.setText("");

        jButton4.setText("Register Supplier");

        jLabel2.setText("Add Suppliers");

        jButton5.setVisible(false);

    }

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

        String fname = jTextField1.getText();

        String lname = jTextField2.getText();

        String contact = jTextField3.getText();

        String company = jLabel26.getText();

        String branch = jLabel30.getText();


        if (fname.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter The First name.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (lname.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter The Last name.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (contact.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Contact Number", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (!Pattern.compile("[0-9]+").matcher(contact).matches() || contact.length() != 10) {
            JOptionPane.showMessageDialog(this, "Invalid Contact number.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (company.isBlank()) {
            JOptionPane.showMessageDialog(this, "Select The company", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (branch.isBlank()) {
            JOptionPane.showMessageDialog(this, "Select The Branch", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            try {
                if (jButton4.getText().equals("Register Supplier")) {
                    //Search product if exists
                    ResultSet prs = MySQL.search("SELECT * FROM `supplier` WHERE `fname` ='" +
fname + "' AND `lname` = '" + lname + "' AND `contact_no` = '" + contact + "' AND
`company_branches_id` = '" + branch + "' ;");
                    //Search product if exists
                    if (prs.next()) {
```

```
                    JOptionPane.showMessageDialog(this, "This Supplier already exists.", "Warning",
JOptionPane.WARNING_MESSAGE);
                } else {
                    MySQL.iud("INSERT INTO
`supplier`(`fname`,`lname`,`contact_no`,`company_branches_id`) VALUES('" + fname + "','" +
lname + "','" + contact + "','" + branch + "');");
                    AddedSuccess as = new AddedSuccess(ah, false, "Added Successfully");
                    as.setVisible(true);
                    Runnable runnable = new Runnable() {
                        @Override
                        public void run() {
                            try {
                                Thread.sleep(1500);
                                as.dispose();
                            } catch (Exception e) {
                                e.printStackTrace();
                            }
                        }
                    };
                    Thread st = new Thread(runnable);
                    st.start();
                    clearFields();
                }
            } else {
                //Search product if exists
                ResultSet prs = MySQL.search("SELECT * FROM `supplier` WHERE `fname` ='" + 
fname + "' AND `lname` = '" + lname + "' AND `company_branches_id` = '" + branch + "' AND 
`contact_no` = '" + contact + "' AND `id` NOT IN (" + tudp.getSupplierId() + ");");
                //Search product if exists
                if (prs.next()) {
                    JOptionPane.showMessageDialog(this, "This product already exists.", "Warning", 
JOptionPane.WARNING_MESSAGE);
                } else {
                    MySQL.iud("UPDATE `supplier` SET `fname`='" + fname + "',`lname`='" + lname + 
"',`company_branches_id`='" + branch + "',`contact_no` = '" + contact + "' WHERE `id` = '" + 
tudp.getSupplierId() + "';");
                    AddedSuccess as = new AddedSuccess(ah, false, "Updated Successfully");
                    as.setVisible(true);
                    Runnable runnable = new Runnable() {
                        @Override
                        public void run() {
                            try {
                                Thread.sleep(1500);
                                su.reSet();
                                as.dispose();
                            } catch (Exception e) {
                                e.printStackTrace();
                            }
                        }
                    };
                    Thread st = new Thread(runnable);
                    st.start();
                    clearFields();
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        }
        }

        private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
            su.reSet();
        }

        private void jTextField3KeyTyped(java.awt.event.KeyEvent evt) {
            String mobile = jTextField3.getText();
```

```
        String text = mobile + evt.getKeyChar();

        if (mobile.length() == 10) {
            evt.consume();
        } else {
            if (!Pattern.compile("[0-9]+").matcher(text).matches()) {
                evt.consume();
            }
        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        SelectCompany sc = new SelectCompany(ah, true, this);
        sc.setVisible(true);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        if (jLabel26.getText().isBlank()) {
            JOptionPane.showMessageDialog(this, "Select the company first.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            SelectBranches sb = new SelectBranches(ah, true, this, jLabel26.getText());

            sb.setVisible(true);

        }

    }
```

Search Suppliers



Search Suppliers

First Name | Last Name | Contact number | Company: Select

Suppliwe Id	Name	Contact Number	Company Id	Company Name	Branch Id	Branch Name
1	Kapila Kumara	0112364598	1	Abans	2	Abans 1
2	Kavindu Manage	0712384569	2	Singer	1	Singer 1
3	Ashan Janidu Kavindu	0332498488	2	Singer	1	Singer 1
4	Kamal Perera	0714879562	2	Singer	1	Singer 1

```
 AdminHome ah;

    String toUpdateRow;

    Supplier su;



    /**

     * Creates new form AddProducts

     */
```

```
public ViewSuppliers() {

    initComponents();

    loadCompnany();

    searchSuppliers();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

}


public ViewSuppliers(AdminHome ah) {

    initComponents();

    this.ah = ah;

    loadCompnany();

    searchSuppliers();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

}


public ViewSuppliers(AdminHome ah, Supplier su) {

    initComponents();

    this.ah = ah;

    loadCompnany();

    searchSuppliers();

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    this.su = su;
```

```
}

public void clearFields() {
    jTextField3.setText("");
    jTextField4.setText("");
    jTextField5.setText("");
    jComboBox1.setSelectedIndex(0);
}

public void loadCompnany() {
    try {
        ResultSet crs = MySQL.search("SELECT * FROM `companies`;");
        Vector v = new Vector();
        v.add("Select");
        while (crs.next()) {
            v.add(crs.getString("name"));
        }
        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
        jComboBox1.setModel(dcm);
    } catch (Exception e) {
        e.printStackTrace();
    }
```

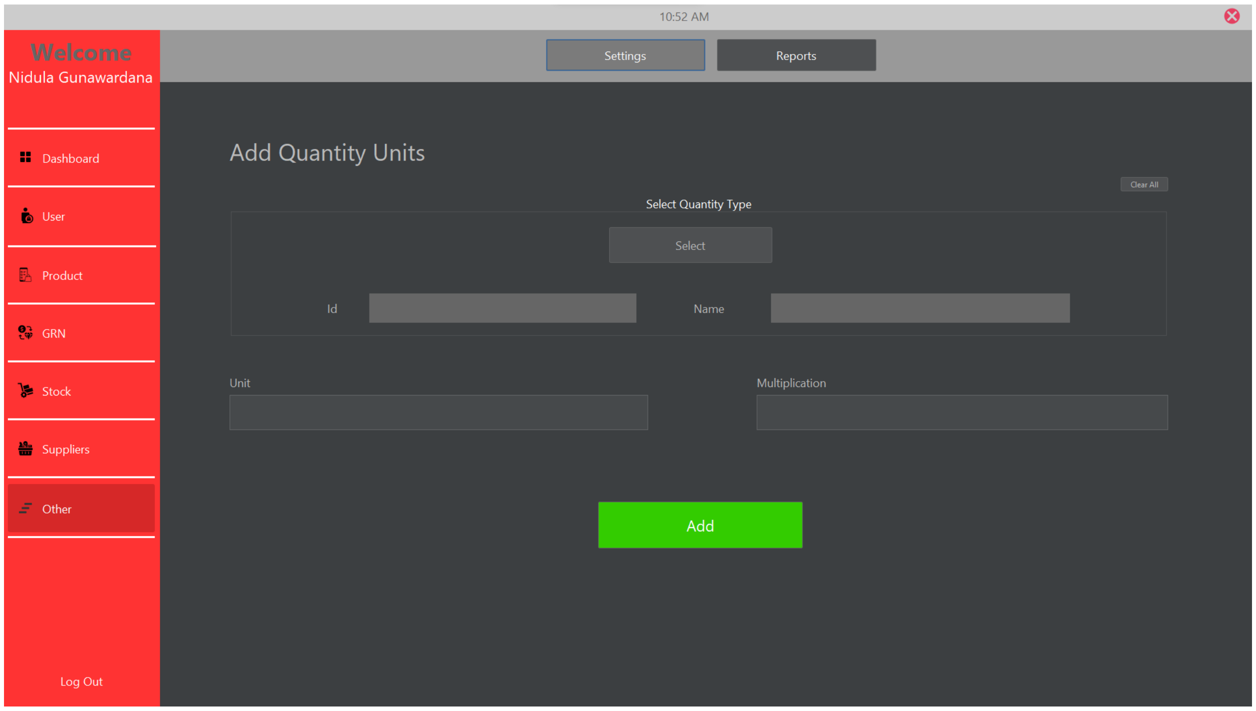
```
}


public void searchSuppliers() {

    String fname = jTextField3.getText();

    String lname = jTextField5.getText();

    String contactNo = jTextField4.getText();

    String company = jComboBox1.getSelectedItem().toString();


    Vector queryv = new Vector();


    if (!fname.isBlank()) {

        queryv.add("`supplier`.`fname` LIKE '%" + fname + "%' ");

    }


    if (!lname.isBlank()) {

        queryv.add("`supplier`.`lname` LIKE '%" + lname + "%' ");

    }


    if (!contactNo.isBlank()) {

        queryv.add("`supplier`.`contact_no` LIKE '%" + contactNo + "%' ");

    }
```

```
        if (!company.equals("Select")) {

            queryv.add("`companies`.`name` = '" + company + "' ");

        }


        String query = "";


        for (int i = 0; i < queryv.size(); i++) {

            if (i == 0) {

                query += "WHERE ";

            }


            query += queryv.get(i);

            if (i + 1 != queryv.size()) {

                query += "AND ";

            }


        }

        try {

            ResultSet rs = MySQL.search("SELECT * FROM `supplier` INNER JOIN
    `company_branches` ON `company_branches`.`id` =  `supplier`.`company_branches_id` INNER
    JOIN `companies` ON `companies`.`id` = `company_branches`.`companies_id` " + query + "
    ORDER BY `supplier`.`id` ASC;");
```

```
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("supplier.id"));
                v.add(rs.getString("supplier.fname") + " " + rs.getString("supplier.lname"));
                v.add(rs.getString("supplier.contact_no"));
                v.add(rs.getString("companies.id"));
                v.add(rs.getString("companies.name"));
                v.add(rs.getString("company_branches.id"));
                v.add(rs.getString("company_branches.name"));
                dtm.addRow(v);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {
        searchSuppliers();
    }

    private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {
        searchSuppliers();
    }


    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                int con = JOptionPane.showConfirmDialog(this, "Do you want To update?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
                if (con == JOptionPane.YES_OPTION) {
                    this.toUpdateRow = jTable1.getValueAt(selectedRow, 0).toString();
                    ToUpdateDataSuppliers tus = new ToUpdateDataSuppliers();
                    tus.setSupplierId(toUpdateRow);
                    tus.setFname(jTable1.getValueAt(selectedRow, 1).toString().split("\\s")[0]);
                    tus.setLname(jTable1.getValueAt(selectedRow, 1).toString().split("\\s")[1]);
                    tus.setContact(jTable1.getValueAt(selectedRow, 2).toString());
                    tus.setCompanyId(jTable1.getValueAt(selectedRow, 3).toString());
                    tus.setCompanyName(jTable1.getValueAt(selectedRow, 4).toString());
                    tus.setBranchId(jTable1.getValueAt(selectedRow, 5).toString());
                    tus.setBranchName(jTable1.getValueAt(selectedRow, 6).toString());
                    this.su.loadForUpdate(tus);
                }
            }
        }
    }


    private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
        searchSuppliers();
    }


    private void jTextField5KeyReleased(java.awt.event.KeyEvent evt) {
        searchSuppliers();
    }
```

Other

Settings



Add Quantity Units

Select Quantity Type

Id

Name

Unit

Multiplication

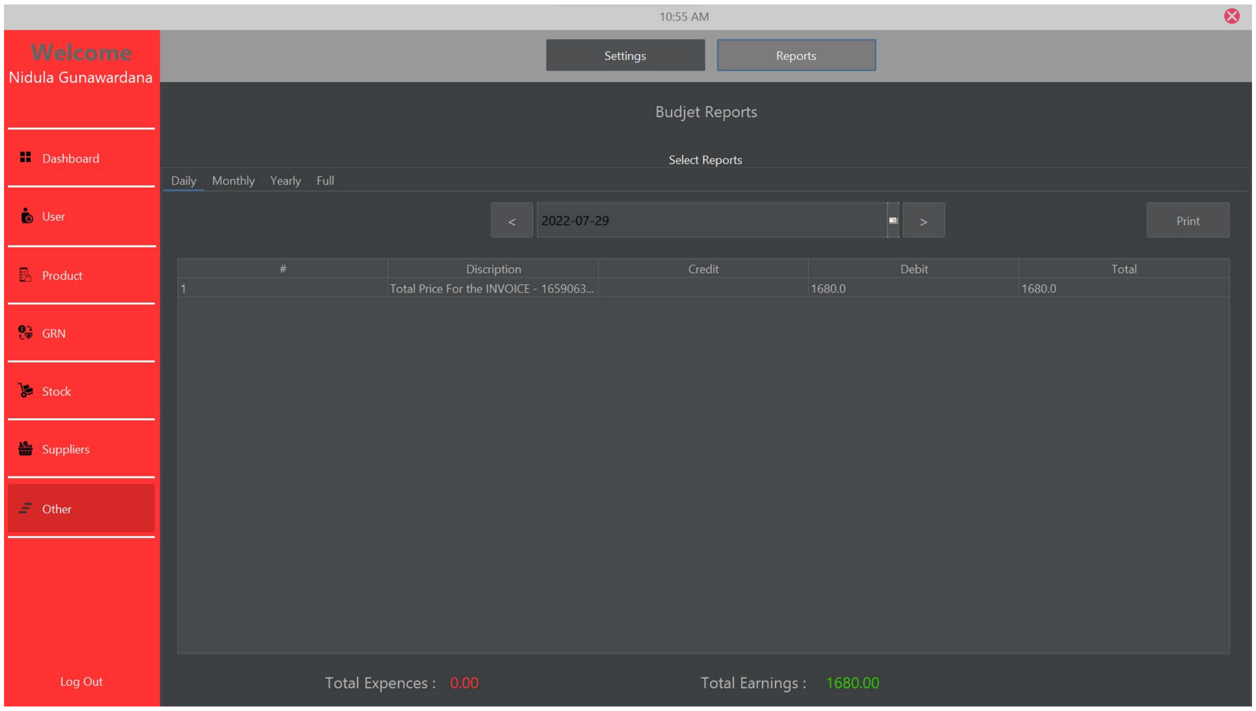
```
AdminHome ah;

String toUpdateRow;

ToUpdateDateStock tus;

DecimalFormat df = new DecimalFormat("0.00");

Stock st;



/**

 * Creates new form AddProducts

 */

public Settings() {
        initComponents();

    }


    public Settings(AdminHome ah) {

        initComponents();

        this.ah = ah;

    }


    public void clearFields() {

        jLabel14.setText("");

        jLabel16.setText("");

        jTextField3.setText("");

        jTextField4.setText("");

    }

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        String qtyTypeid = jLabel14.getText();

        String name = jTextField4.getText();

        String multi = jTextField3.getText();


        if (qtyTypeid.isBlank()) {

            JOptionPane.showMessageDialog(this, "Select Qty Type.", "Warning",
JOptionPane.WARNING_MESSAGE);
```

```
        } else if (name.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Qty unit name", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (multi.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Multiplication.", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (!Pattern.compile("[0][.]?[0-9]*|[1-9]+[0-9]*[.]?[0-9]*").matcher(multi).matches())
{
            JOptionPane.showMessageDialog(this, "Invalid Multiplication Format", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {
            try {

                ResultSet qusrs = MySQL.search("SELECT * FROM `qty_units` WHERE `name` = '" +
name + "' AND `qty_type_id` = '" + qtyTypeid + "' AND `multiplication` = '" +
String.valueOf(Double.parseDouble(multi)) + "';");
                if (qusrs.next()) {
                    JOptionPane.showMessageDialog(this, "This Unit already added.", "Warning",
JOptionPane.WARNING_MESSAGE);
                } else {

                    MySQL.iud("INSERT INTO `qty_units`(`name`,`qty_type_id`,`multiplication`)
VALUES('" + name + "','" + qtyTypeid + "','" + String.valueOf(Double.parseDouble(multi)) + "');");
```

```
            AddedSuccess as = new AddedSuccess(ah, false, "Added Successfully");

            as.setVisible(true);

            Runnable runnable = new Runnable() {

                @Override

                public void run() {

                    try {

                        Thread.sleep(1500);

                        as.dispose();

                    } catch (Exception e) {

                        e.printStackTrace();

                    }

                }

            };

            Thread st = new Thread(runnable);

            st.start();

            clearFields();


        }

    } catch (Exception e) {

        e.printStackTrace();

    }

}
```

```
}


private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    SelectQtyType sqt = new SelectQtyType(ah, true, this);
    sqt.setVisible(true);
}


private void jTextField3KeyTyped(java.awt.event.KeyEvent evt) {
    String qty = jTextField3.getText();

    String text = qty + evt.getKeyChar();

    if (!Pattern.compile("[0][.]?[0-9]*|[1-9]+[0-9]*[.]?[0-9]*").matcher(text).matches()) {
        evt.consume();

    }
}
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    clearFields();
}
```

Reports



#	Discription	Credit	Debit	Total
1	Total Price For the INVOICE - 1659063...		1680.0	1680.0

Total Expences : 0.00 Total Earnings : 1680.00

```
AdminHome ah;


    DecimalFormat df = new DecimalFormat("0.00");


    /**
     * Creates new form AddProducts
     */
    public Reports() {
        initComponents();
        changeDate("");
        jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));
    jTable2.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    jTable3.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    jTable4.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    loadChanges("4");

}


public Reports(AdminHome ah) {

    initComponents();

    this.ah = ah;

    changeDate("");

    jTable1.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    jTable2.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    jTable3.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    jTable4.getTableHeader().setFont(new Font("Segoe UI", Font.PLAIN, 18));

    loadChanges("4");

}


public void loadChanges(String ty) {

    try {

        String dat;

        DefaultTableModel dtm = null;

        ResultSet rs = null;
        if (ty == "2") {
            dat = String.valueOf(jYearChooser1.getYear()) + "-" + new
DecimalFormat("00").format(jMonthChooser1.getMonth() + 1);
            dtm = (DefaultTableModel) jTable2.getModel();
            rs = MySQL.search("SELECT * FROM `money_changes` WHERE `date_time` > '" + dat +
"-01 00:00:00' AND `date_time` < '" + dat + "-30 23:59:59' ORDER BY `date_time` ASC");
        } else if (ty == "1") {
            dat = new SimpleDateFormat("yyyy-MM-dd").format(jDateChooser2.getDate());
            dtm = (DefaultTableModel) jTable1.getModel();
            rs = MySQL.search("SELECT * FROM `money_changes` WHERE `date_time` > '" + dat +
" 00:00:00' AND `date_time` < '" + dat + " 23:59:59' ORDER BY `date_time` ASC");
        } else if (ty == "3") {
            dat = String.valueOf(jYearChooser2.getYear());
            dtm = (DefaultTableModel) jTable3.getModel();
            rs = MySQL.search("SELECT * FROM `money_changes` WHERE `date_time` > '" + dat +
"-01-01 00:00:00' AND `date_time` < '" + dat + "-12-31 23:59:59' ORDER BY `date_time` ASC");
        } else if (ty == "4") {
            dtm = (DefaultTableModel) jTable4.getModel();
            rs = MySQL.search("SELECT * FROM `money_changes` ORDER BY `date_time` ASC");
        }

        int cou = 1;
        double tot = 0;
double totex = 0;

double totea = 0;

dtm.setRowCount(0);

while (rs.next()) {

    Vector v = new Vector();

    v.add(cou);

    v.add(rs.getString("description"));

    if (Double.parseDouble(rs.getString("price")) > 0) {

        v.add("");

        v.add(Double.parseDouble(rs.getString("price")));

        tot = tot + Double.parseDouble(rs.getString("price"));

        totea = totea + Double.parseDouble(rs.getString("price"));

    } else {

        v.add(Double.parseDouble(rs.getString("price")) * -1);

        v.add("");

        tot = tot + Double.parseDouble(rs.getString("price"));

        totex = totex + Double.parseDouble(rs.getString("price"));

    }

    v.add(tot);

    dtm.addRow(v);

    cou++;

}
```

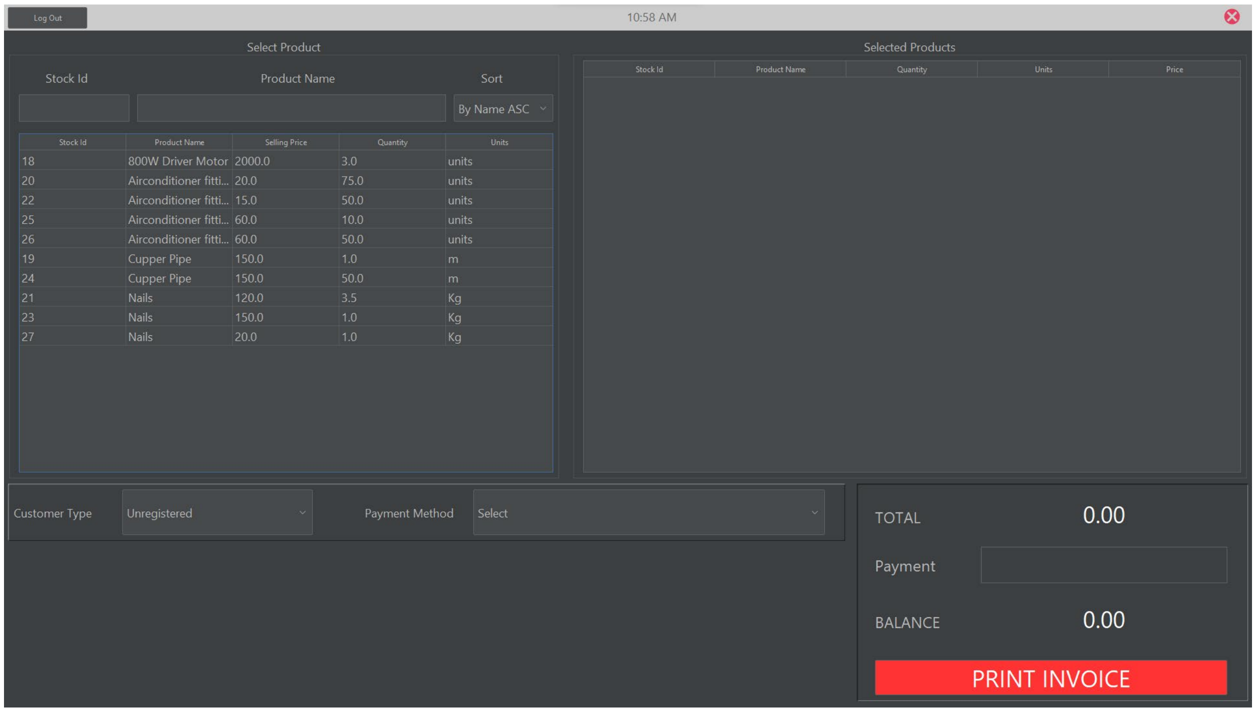
```
        if (totex != 0) {

            totex = totex * -1;

        } else {

            totex = 0.00;

        }

        if (ty == "1") {

            jLabel3.setText(df.format(totex));

            jLabel5.setText(df.format(totea));

        } else if (ty == "2") {

            jLabel7.setText(df.format(totex));

            jLabel9.setText(df.format(totea));

        } else if (ty == "3") {

            jLabel11.setText(df.format(totex));

            jLabel13.setText(df.format(totea));

        } else if (ty == "4") {

            jLabel15.setText(df.format(totex));

            jLabel17.setText(df.format(totea));

        }

    } catch (Exception e) {

        e.printStackTrace();

    }

}
```

```
    public void changeDate(String ico) {
        if (ico.equals("<")) {
            LocalDate d = LocalDate.parse(new SimpleDateFormat("yyyy-MM-
dd").format(jDateChooser2.getDate()));
            LocalDate newdate = d.minusDays(1);
            try {
                jDateChooser2.setDate(new SimpleDateFormat("yyyy-MM-
dd").parse(String.valueOf(newdate)));
            } catch (Exception e) {
                e.printStackTrace();
            }
        } else if (ico.equals(">")) {
            LocalDate d = LocalDate.parse(new SimpleDateFormat("yyyy-MM-
dd").format(jDateChooser2.getDate()));
            LocalDate newdate = d.plusDays(1);
            try {
                jDateChooser2.setDate(new SimpleDateFormat("yyyy-MM-
dd").parse(String.valueOf(newdate)));
            } catch (Exception e) {
                e.printStackTrace();
            }
        } else {
            String dNow = new SimpleDateFormat("yyyy-MM-dd").format(new Date());
            try {
                Date d = new SimpleDateFormat("yyyy-MM-dd").parse(dNow);
                jDateChooser2.setDate(d);
            } catch (ParseException ex) {
                ex.printStackTrace();
            }
        }
    }
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        changeDate(jButton1.getText());
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        changeDate(jButton2.getText());
    }

    private void jDateChooser2PropertyChange(java.beans.PropertyChangeEvent evt) {
        loadChanges("1");
    }

    private void jMonthChooser1PropertyChange(java.beans.PropertyChangeEvent evt) {
        loadChanges("2");
```

```
}


private void jYearChooser1PropertyChange(java.beans.PropertyChangeEvent evt) {
    loadChanges("2");
}


private void jYearChooser2PropertyChange(java.beans.PropertyChangeEvent evt) {
    loadChanges("3");
}


private void jTabbedPane1StateChanged(javax.swing.event.ChangeEvent evt) {
    if (jTabbedPane1.getSelectedIndex() == 0) {
        loadChanges("1");
    } else if (jTabbedPane1.getSelectedIndex() == 1) {
        loadChanges("2");
    } else if (jTabbedPane1.getSelectedIndex() == 2) {
        loadChanges("3");
    } else if (jTabbedPane1.getSelectedIndex() == 3) {
        loadChanges("4");
    }
}
```

```
    private void jTabbedPane1PropertyChange(java.beans.PropertyChangeEvent evt) {


    }


    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            if (jTable3.getRowCount() != 0) {
                JRTableModelDataSource datasourse = new
JRTableModelDataSource(jTable3.getModel());
                HashMap parameters = new HashMap();
                parameters.put("forwhat", "Year " + jYearChooser2.getYear());
                parameters.put("expenses", jLabel11.getText());
                parameters.put("earnings", jLabel13.getText());


                String filepath = "src//reports//budjetreport1.jasper";
                JasperPrint jp = JasperFillManager.fillReport(filepath, parameters, datasourse);
                JasperViewer.viewReport(jp, false);
                JasperPrintManager.printReport(jp, true);
            }


        } catch (Exception e) {
            e.printStackTrace();
        }

    }


    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        try {

            if (jTable1.getRowCount() != 0) {

                JRTableModelDataSource datasourse = new
JRTableModelDataSource(jTable1.getModel());

                HashMap parameters = new HashMap();

                parameters.put("forwhat", "Day " + new SimpleDateFormat("yyyy-MM-
dd").format(jDateChooser2.getDate()));

                parameters.put("expenses", jLabel3.getText());

                parameters.put("earnings", jLabel5.getText());



                String filepath = "src//reports//budjetreport1.jasper";

                JasperPrint jp = JasperFillManager.fillReport(filepath, parameters, datasourse);

                JasperViewer.viewReport(jp, false);

                JasperPrintManager.printReport(jp, true);

            }


        } catch (Exception e) {

            e.printStackTrace();
        }



    }


    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            if (jTable2.getRowCount() != 0) {
                JRTableModelDataSource datasourse = new
JRTableModelDataSource(jTable2.getModel());
                HashMap parameters = new HashMap();
                parameters.put("forwhat", "Month " + jYearChooser1.getYear() + " - " +
jMonthChooser1.getMonth());
                parameters.put("expenses", jLabel7.getText());
                parameters.put("earnings", jLabel9.getText());

                String filepath = "src//reports//budjetreport1.jasper";
                JasperPrint jp = JasperFillManager.fillReport(filepath, parameters, datasourse);
                JasperViewer.viewReport(jp, false);
                JasperPrintManager.printReport(jp, true);
            }
        } catch (Exception e) {

            e.printStackTrace();

        }

    }


    private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

        try {

            if (jTable4.getRowCount() != 0) {

                JRTableModelDataSource datasourse = new
JRTableModelDataSource(jTable4.getModel());

                HashMap parameters = new HashMap();

                parameters.put("forwhat", "All");

                parameters.put("expenses", jLabel15.getText());

                parameters.put("earnings", jLabel17.getText());


                String filepath = "src//reports//budjetreport1.jasper";

                JasperPrint jp = JasperFillManager.fillReport(filepath, parameters, datasourse);

                JasperViewer.viewReport(jp, false);

                JasperPrintManager.printReport(jp, true);

            }


        } catch (Exception e) {
          e.printStackTrace();
        }
    }
```

Cashier Home



Stock Id	Product Name	Selling Price	Quantity	Units
18	800W Driver Motor	2000.0	3.0	units
20	Airconditioner fitti...	20.0	75.0	units
22	Airconditioner fitti...	15.0	50.0	units
25	Airconditioner fitti...	60.0	10.0	units
26	Airconditioner fitti...	60.0	50.0	units
19	Cupper Pipe	150.0	1.0	m
24	Cupper Pipe	150.0	50.0	m
21	Nails	120.0	3.5	Kg
23	Nails	150.0	1.0	Kg
27	Nails	20.0	1.0	Kg

Stock Id	Product Name	Quantity	Units	Price

```
static private Thread timet;

    static private CashierHome ah;


    /**
     * Creates new form AdminHome
     */
    public CashierHome() {
        initComponents();
        ImageIcon i = new ImageIcon("src/resources/logomain.png");

        Image x = i.getImage();

        setIconImage(x);

        searchStock();

        loadPaymentMethods();

        jTextField5.setEnabled(false);

        jPanel7.setVisible(false);

        viewTime();

        jTable1.grabFocus();

    }


    public void viewTime() {

        DateTimeFormatter sdf = DateTimeFormatter.ofPattern("hh:mm a");


        Runnable runnable = new Runnable() {

            @Override

            public void run() {

                while (true) {

                    jLabel19.setText(sdf.format(LocalTime.now()));

                    try {

                        Thread.sleep(10000);

                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }

            }
        };

        timet = new Thread(runnable);
        timet.start();
    }

    public void searchStock() {
        String stock_id = jTextField3.getText();
        String P_name = jTextField4.getText();
        String sort = jComboBox1.getSelectedItem().toString();

        Vector queryv = new Vector();

        if (!stock_id.isBlank()) {
            queryv.add("`stock`.`id` = '" + stock_id + "' ");
        }
```

```
if (!P_name.isBlank()) {

    queryv.add("`product`.`name` LIKE '%" + P_name + "%' ");

}

String sortQue = "";

if (sort.equals("By Name ASC")) {

    sortQue = "ORDER BY `product`.`name` ASC";

} else if (sort.equals("By Name DESC")) {

    sortQue = "ORDER BY `product`.`name` DESC";

} else if (sort.equals("By MFD ASC")) {

    sortQue = "ORDER BY `stock`.`mfd` ASC";

} else if (sort.equals("By MFD DESC")) {

    sortQue = "ORDER BY `stock`.`mfd` DESC";

} else if (sort.equals("By Quantity ASC")) {

    sortQue = "ORDER BY `stock`.`qty` DESC";

} else if (sort.equals("By Quantity DESC")) {

    sortQue = "ORDER BY `stock`.`qty` DESC";

}



String query = "";



for (int i = 0; i < queryv.size(); i++) {

    if (i == 0) {
```

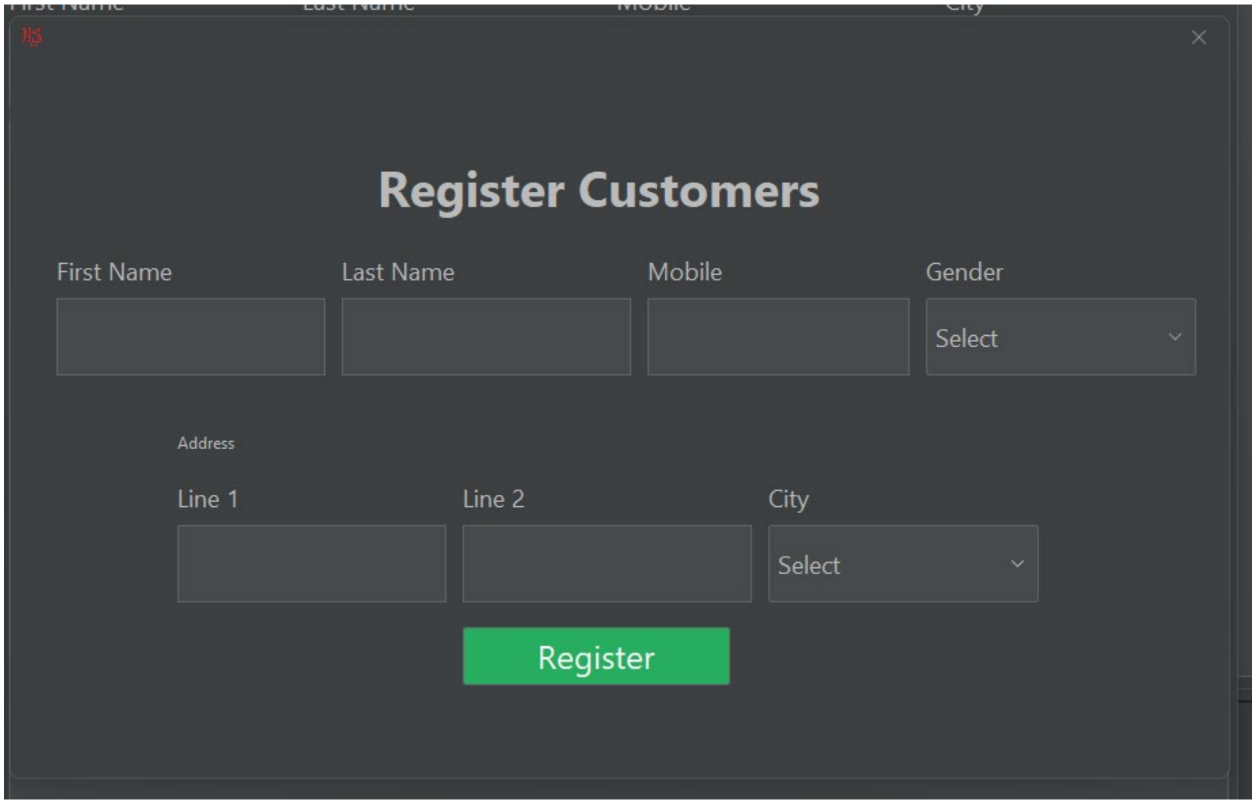
```
                query += "WHERE ";
            }


            query += queryv.get(i);
            if (i + 1 != queryv.size()) {
                query += "AND ";
            }


        }
//        System.out.println(query);
        try {
            ResultSet rs = MySQL.search("SELECT * FROM `stock` INNER JOIN `product` ON
`product`.`id` = `stock`.`product_id` INNER JOIN `brand` ON `brand`.`id` = `product`.`brand_id`
INNER JOIN `category` ON `category`.`id` = `product`.`category_id` INNER JOIN `qty_units` ON
`qty_units`.`id` = `stock`.`qty_units_id` " + query + " " + sortQue + ";");


            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("stock.id"));
                v.add(rs.getString("product.name"));
            v.add(rs.getString("stock.selling_price"));

            v.add(rs.getString("stock.qty"));

            v.add(rs.getString("qty_units.name"));


            dtm.addRow(v);

        }

    } catch (Exception e) {

        e.printStackTrace();

    }

}


public void calculateTotal() {

    double tot = 0;

    for (int i = 0; i < jTable2.getRowCount(); i++) {

        tot = tot + Double.parseDouble(jTable2.getValueAt(i, 4).toString());

    }

    jLabel3.setText(String.valueOf(tot));

}


public void loadPaymentMethods() {

    try {
            ResultSet rs = MySQL.search("SELECT * FROM `payment_type` WHERE `id` NOT IN
(4,3);");
            Vector v = new Vector();
            v.add("Select");
            while (rs.next()) {
                v.add(rs.getString("name"));
            }
            DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);
            jComboBox3.setModel(dcm);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void clearFields() {
        DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();
        dtm.setRowCount(0);
        jComboBox2.setSelectedIndex(0);
        jComboBox3.setSelectedIndex(0);
        jLabel3.setText("0.00");
        jTextField5.setText("");
        jLabel17.setText("0.00");

        con = -1;

    }

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        timet.stop();

        timet = null;

        this.dispose();


    }


    private void jTextField4KeyReleased(java.awt.event.KeyEvent evt) {

        searchStock();

    }


    private void jTextField4KeyTyped(java.awt.event.KeyEvent evt) {


    }


    private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {

        searchStock();

    }
    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 1) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                try {
                    ResultSet prs = MySQL.search("SELECT * FROM `stock` WHERE `stock`.`id` = '" +
jTable1.getValueAt(selectedRow, 0).toString() + "';");
                    prs.next();
                    String pid = prs.getString("product_id");
                    SetQuantity sq = new SetQuantity(this, true, jTable1.getValueAt(selectedRow,
1).toString(), pid, jTable1.getValueAt(selectedRow, 4).toString());
                    sq.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }

            }
        }
    }

    private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
        searchStock();
    }

    private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void jTable2MouseClicked(java.awt.event.MouseEvent evt) {
        if (evt.getClickCount() == 2) {
            int con = JOptionPane.showConfirmDialog(this, "Do you want to remove product?",
"Warning", JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
            if (con == JOptionPane.YES_OPTION) {
                int selectedRow = jTable2.getSelectedRow();
                DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();
                dtm.removeRow(selectedRow);
                calculateTotal();
            }
        }
    }

    private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {
        if (jComboBox2.getSelectedItem().toString().equals("Unregistered")) {
            jPanel7.setVisible(false);
            jLabel26.setText("");
        } else {
            if (jLabel26.getText().isBlank()) {
                SelectCustomers sc = new SelectCustomers(this, true);
                sc.setVisible(true);
            }
        }
    }

    private void jComboBox2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void jComboBox3ItemStateChanged(java.awt.event.ItemEvent evt) {
        if (!jComboBox3.getSelectedItem().toString().equals("Select")) {
            jTextField5.setEnabled(true);
        } else {
            jTextField5.setEnabled(false);
        }
    }

    private void jComboBox3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }


    private void jTextField5KeyReleased(java.awt.event.KeyEvent evt) {
        if (!jTextField5.getText().isBlank()) {
            double tottal = Double.parseDouble(jLabel3.getText());
            double payment = Double.parseDouble(jTextField5.getText());
            jLabel17.setText(String.valueOf(payment - tottal));
        } else {
            jLabel17.setText("0.00");
        }


    }


    private void jTextField5KeyTyped(java.awt.event.KeyEvent evt) {
        String qty = jTextField5.getText();


        String text = qty + evt.getKeyChar();


        if (!Pattern.compile("0|0[.]|0.[1-9]|0.[1-9][0-9]|[1-9][0-9]*[.]?[0-
9]{0,2}").matcher(text).matches()) {
            evt.consume();
        }
    }


    private void jTable1KeyPressed(java.awt.event.KeyEvent evt) {
        if (evt.getKeyCode() == 10) {
            int selectedRow = jTable1.getSelectedRow();
            if (selectedRow != -1) {
                try {
                    ResultSet prs = MySQL.search("SELECT * FROM `stock` WHERE `stock`.`id` = '" +
jTable1.getValueAt(selectedRow, 0).toString() + "';");
                    prs.next();
                    String pid = prs.getString("product_id");
                    SetQuantity sq = new SetQuantity(this, true, jTable1.getValueAt(selectedRow,
1).toString(), pid, jTable1.getValueAt(selectedRow, 4).toString());
                    sq.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }


            }
        }
    }

    int con = -1;

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        String paymentMethod = jComboBox3.getSelectedItem().toString();

        String total = jLabel3.getText();

        String payment = jTextField5.getText();

        String balance = jLabel17.getText();

        if (jTable2.getRowCount() == 0) {

            JOptionPane.showMessageDialog(this, "Please add atleast one product.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (jComboBox2.getSelectedItem().toString().equals("Unregistered") && con == -1) {

            con = JOptionPane.showConfirmDialog(this, "Is customer not registered?", "Warning",
JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);

            if (con == JOptionPane.NO_OPTION) {

                jComboBox2.setSelectedIndex(1);

            }

        } else if (paymentMethod.equals("Select")) {

            JOptionPane.showMessageDialog(this, "Select the payment method", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else if (payment.isBlank()) {

            JOptionPane.showMessageDialog(this, "Enter the payment.", "Warning",
JOptionPane.WARNING_MESSAGE);

        } else {
            String customerId = "0";

            if (!jLabel26.getText().isBlank()) {

                customerId = jLabel26.getText();

            }

            try {

                long mTime = System.currentTimeMillis();

                String uniqId = mTime + "-" + CashierSignIn1.cashierId;

                String dNow = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(new Date());


                //Invoice Insert

                MySQL.iud("INSERT INTO `invoice`(`invoice_id`,`date_time`,`customers_id`,`user_id`)
VALUES('" + uniqId + "','" + dNow + "','" + customerId + "','" + CashierSignIn1.cashierId + "');");


                ResultSet invrs = MySQL.search("SELECT * FROM `invoice` WHERE `invoice_id` = '" +
uniqId + "';");

                invrs.next();

                String invoiceId = invrs.getString("id");


                //Invoice Insert

                //Insert to invoice quantity

                Vector v = new Vector();

                for (int i = 0; i < jTable2.getRowCount(); i++) {
```

```
                //Qty unit id

                ResultSet qurs = MySQL.search("SELECT * FROM `qty_units` WHERE `name` = '" +
jTable2.getValueAt(i, 3).toString() + "';");

                qurs.next();

                String quid = qurs.getString("id");

                double newmul = Double.parseDouble(qurs.getString("multiplication"));

                //Qty unit id


                MySQL.iud("INSERT INTO `invoice_item`(`stock_id`,`qty`,`qty_units_id`,`invoice_id`)
VALUES('" + jTable2.getValueAt(i, 0).toString() + "','" + jTable2.getValueAt(i, 2).toString() + "','" +
quid + "','" + invoiceId + "');");


                //Search from stock

                ResultSet strs = MySQL.search("SELECT * FROM `stock` INNER JOIN `qty_units` ON
`qty_units`.`id` = `stock`.`qty_units_id` WHERE `stock`.`id` = '" + jTable2.getValueAt(i,
0).toString() + "';");

                strs.next();

                double stmul = Double.parseDouble(strs.getString("qty_units.multiplication"));

                //Search from stock


                double newQty = Double.parseDouble(strs.getString("stock.qty").toString()) -
(Double.parseDouble(jTable2.getValueAt(i, 2).toString()) * newmul * stmul);
```

```
                    MySQL.iud("UPDATE `stock` SET `qty` = '" + String.valueOf(newQty) + "' WHERE `id` =
'" + jTable2.getValueAt(i, 0).toString() + "';");

                    MySQL.iud("INSERT INTO
`stock_changes`(`description`,`date_time`,`stock_id`,`qty`,`qty_units_id`) VALUES('Bought From
Invoice -" + uniqId + "','" + dNow + "','" + jTable2.getValueAt(i, 0).toString() + "','-" +
jTable2.getValueAt(i, 2).toString() + "','" + quid + "');");

                    double unitPrice = Double.parseDouble(jTable2.getValueAt(i, 4).toString()) /
(Double.parseDouble(jTable2.getValueAt(i, 2).toString()) * newmul);

                    v.add(new InvoicePaymentData(String.valueOf(i), jTable2.getValueAt(i, 1).toString(),
"Rs. " + String.valueOf(unitPrice), jTable2.getValueAt(i, 2).toString() + " " + jTable2.getValueAt(i,
3).toString(), "Rs. " + jTable2.getValueAt(i, 4).toString()));

                }

                //Insert to invoice quantity



                //Invoice Payments

                ResultSet ptrs = MySQL.search("SELECT * FROM `payment_type` WHERE `name` = '" +
paymentMethod + "';");

                ptrs.next();

                String paymentMethodId = ptrs.getString("id");



                MySQL.iud("INSERT INTO
`invoice_payments`(`payment`,`balance`,`payment_type_id`,`invoice_id`) VALUES('" + payment
+ "','" + balance + "','" + paymentMethodId + "','" + invoiceId + "')");
```

```
            MySQL.iud("INSERT INTO `money_changes`(`description`,`price`,`date_time`)
VALUES('Total Price For the INVOICE - " + uniqId + "','" + jLabel3.getText() + "','" + dNow + "');");

            //Invoice data
            String customername;
            String address;

            if (jLabel26.getText().isBlank()) {
               customername = "N/A";
               address = "N/A";
            } else {
               customername = jLabel27.getText();
               address = jLabel29.getText() + ", " + jLabel30.getText() + ", " + jLabel31.getText() +
".";
            }

            JRBeanCollectionDataSource datasourse = new JRBeanCollectionDataSource(v);
            HashMap parameters = new HashMap();
            parameters.put("invoiceid", uniqId);
            parameters.put("date", dNow.split("\\s")[0]);
            parameters.put("customername", customername);
            parameters.put("total", "Rs. " + jLabel3.getText());
            parameters.put("payment", "Rs. " + payment);

            parameters.put("balance", "Rs. " + balance);

            parameters.put("paymentMethod", paymentMethod);

            parameters.put("address", address);

            parameters.put("tabledata", datasourse);


            String filepath = "src//reports//invoiceNshop.jasper";

            JasperPrint jp = JasperFillManager.fillReport(filepath, parameters, new
JREmptyDataSource());


            AddedSuccess as = new AddedSuccess(ah, false, "Success");

            as.setVisible(true);

            Runnable runnable = new Runnable() {

                @Override

                public void run() {

                    try {

                        Thread.sleep(1500);

                        as.dispose();

                        JasperViewer.viewReport(jp, false);

                        JasperPrintManager.printReport(jp, true);

                    } catch (Exception e) {

                        e.printStackTrace();
                    }
                }
            };
            Thread st = new Thread(runnable);
            st.start();
            //Invoice data
            searchStock();
            clearFields();
            //Invoice payments

            System.gc();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    int option = JOptionPane.showConfirmDialog(this, "Do you want to log out?",
"Confirmation", JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);
    if (option == JOptionPane.YES_OPTION) {
        timet.stop();

        new SelectSignin().setVisible(true);

        this.dispose();

    }

}
```

Register Customers



```
SelectCustomers cr;

    CashierHome ch;

    AdminHome ah;

    ToUpdateCustomerDetails tucd;

    Users u;
```

```
/**
 * Creates new form CustomerRegistration
 */
public CustomerRegistration(java.awt.Frame parent, boolean modal) {
    super(parent, modal);
    initComponents();
    ImageIcon i = new ImageIcon("src/resources/logomain.png");
    Image x = i.getImage();
    setIconImage(x);
    loadCity();
    loadGender();
    jButton4.setVisible(false);
}

public CustomerRegistration(CashierHome parent, boolean modal, SelectCustomers cr) {
    super(parent, modal);
    initComponents();
    ImageIcon i = new ImageIcon("src/resources/logomain.png");
    Image x = i.getImage();
    setIconImage(x);
    loadCity();
```

```
        loadGender();

        this.cr = cr;

        this.ch = parent;

        jButton4.setVisible(false);

    }


    public CustomerRegistration(AdminHome parent, boolean modal, ToUpdateCustomerDetails
tucd, Users u) {

        super(parent, modal);

        initComponents();

        ImageIcon i = new ImageIcon("src/resources/logomain.png");

        Image x = i.getImage();

        setIconImage(x);

        loadCity();

        loadGender();

        this.ah = parent;

        this.tucd = tucd;

        this.u = u;

        jTextField7.setText(tucd.getFname());

        jTextField8.setText(tucd.getLname());

        jTextField9.setText(tucd.getMobile());

        jComboBox4.setSelectedItem(tucd.getGender());
    jTextField10.setText(tucd.getLine1());

    jTextField11.setText(tucd.getLine2());

    jComboBox5.setSelectedItem(tucd.getCity());

    jButton3.setText("Update");

    jLabel13.setText("Update Customers");


}


public void loadGender() {


    try {

        ResultSet rs = MySQL.search("SELECT * FROM `gender`;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox4.setModel(dcm);


    } catch (Exception e) {

        e.printStackTrace();
    }


}


public void loadCity() {


    try {

        ResultSet rs = MySQL.search("SELECT * FROM `city`;");

        Vector v = new Vector();

        v.add("Select");

        while (rs.next()) {

            v.add(rs.getString("name"));

        }

        DefaultComboBoxModel dcm = new DefaultComboBoxModel(v);

        jComboBox5.setModel(dcm);


    } catch (Exception e) {

        e.printStackTrace();

    }


}
```

```
    public void clearAddFields() {

        jTextField7.setText("");

        jTextField8.setText("");

        jTextField9.setText("");

        jTextField10.setText("");

        jTextField11.setText("");

        jComboBox4.setSelectedIndex(0);

        jComboBox5.setSelectedIndex(0);

        jButton3.setText("Register");

        jLabel13.setText("Register Customers");

    }

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        String fname = jTextField7.getText();

        String lname = jTextField8.getText();

        String mobile = jTextField9.getText();

        String gender = jComboBox4.getSelectedItem().toString();

        String line1 = jTextField10.getText();

        String line2 = jTextField11.getText();

        String city = jComboBox5.getSelectedItem().toString();


        if (fname.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter First name", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (lname.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Last name", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (mobile.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Mobile Number", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (gender.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select Gender", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (line1.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Address Line 1", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (line2.isBlank()) {
            JOptionPane.showMessageDialog(this, "Enter Address Line 2", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else if (city.equals("Select")) {
            JOptionPane.showMessageDialog(this, "Select City", "Warning",
JOptionPane.WARNING_MESSAGE);
        } else {

            try {
```

```
            //Gender

            ResultSet grs = MySQL.search("SELECT * FROM `gender` WHERE `name` = '" + gender + 
"';");

            grs.next();

            String genderId = grs.getString("id");

            //Gender


            //Gender

            ResultSet crs = MySQL.search("SELECT * FROM `city` WHERE `name` = '" + city + "';");

            crs.next();

            String cityId = crs.getString("id");

            //Gender


            if (jButton3.getText().equals("Register")) {

                //Search For user

                ResultSet ars = MySQL.search("SELECT * FROM `customers` WHERE `mobile` = '" + 
mobile + "' AND `fname` = '" + fname + "' AND `lname` = '" + lname + "';");

                //Search For user


                if (ars.next()) {

                    JOptionPane.showMessageDialog(this, "Customer Already Exists.", "Warning", 
JOptionPane.WARNING_MESSAGE);
```

```
                } else {
                    MySQL.iud("INSERT INTO
`customers`(`fname`,`lname`,`mobile`,`gender_id`,`line1`,`line2`,`city_id`,`ststus_id`) VALUES('"
+ fname + "','" + lname + "','" + mobile + "','" + genderId + "','" + line1 + "','" + line2 + "','" + cityId
+ "','1');");
                    JOptionPane.showMessageDialog(this, "Customer Added Successfully.", "Success",
JOptionPane.INFORMATION_MESSAGE);
                    clearAddFields();
                    this.cr.searchCustomers();
                }
            } else {
                //Search For user
                ResultSet ars = MySQL.search("SELECT * FROM `customers` WHERE `mobile` = '" +
mobile + "' AND `fname` = '" + fname + "' AND `lname` = '" + lname + "' AND `id` NOT IN ('" +
tucd.getCustomerId() + "');");
                //Search For user

                if (ars.next()) {
                    JOptionPane.showMessageDialog(this, "Customer already exists try diffeent one.",
"Warning", JOptionPane.WARNING_MESSAGE);
                } else {
                    MySQL.iud("UPDATE `customers` SET `fname`='" + fname + "',`lname`='" + lname +
"',`mobile`='" + mobile + "',`gender_id`='" + genderId + "',`line1`='" + line1 + "',`line2`='" + line2 +
"',`city_id`='" + cityId + "' WHERE `id` = '" + tucd.getCustomerId() + "';");
```

```
                JOptionPane.showMessageDialog(this, "Customer Updated Successfully.",
"Success", JOptionPane.INFORMATION_MESSAGE);
                this.dispose();
                u.searchCustomers();
            }
            clearAddFields();

        }

    } catch (Exception e) {
        e.printStackTrace();
    }

  }
}

private void jTextField9KeyTyped(java.awt.event.KeyEvent evt) {
    String mobile = jTextField9.getText();

    String text = mobile + evt.getKeyChar();

    if (mobile.length() == 10) {
        evt.consume();
    } else {
        if (!Pattern.compile("[0-9]+").matcher(text).matches()) {
            evt.consume();
        }
    }
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    clearAddFields();
    this.dispose();
}
```