



Lanka Nippon Biztech Institute

SKILL HUNT ADMIN

CMP120130

Object-Oriented Modelling and Development

Reg. No : UGC0122015

Name : Oneli Jayodya

Batch: SE-01

Contents

INTRODUCTION	3
FUNCTIONS AND NON-FUNCTIONS	4
EVIDENCE FOR THE SYSTEM IMPLEMENTATION.....	5
DATA VALIDATION & ERROR HANDLINGS.....	8
WIREFRAME.....	12
TEST PLAN	14
USER DOCUMENTATION	18
EVIDENCE OF CODE IMPLEMENTATION	19
GUI Interfaces code	19
Java Class Codes.....	43
REFERENCES.....	53

INTRODUCTION

This system is for job search. In the system, this is the admin panel part. This system's name is Skill Hunt. There is an admin who manages the category. In this system, there is only one admin. Admin can log the system. Admin has access to Add the category, View the Category list, Update the category, and Delete the category. When Admin adds the category then shows it also in the web application.

FUNCTIONS AND NON-FUNCTIONS

Functions:

Login: Admin logs into the system.

Add Category: Admin can add a new category to the system.

View Category List: Admin can view a list of all categories in the system.

Update Category: Admin can update the details of an existing category.

Delete Category: Admin can delete an existing category

.

Non-Functions:

Display Categories in Web Application: The categories added by the admin are displayed in the web application.

Category Management: The system allows the admin to manage categories, including adding, viewing, updating, and deleting them.

User Interface: The system provides a user interface for the admin to interact with the system and perform the necessary functions.

Data Storage: The system stores the categories in a database for future reference.

Integration with Web Application: The categories added by the admin are integrated with the web application, making them available to users.

EVIDENCE FOR THE SYSTEM IMPLEMENTATION.

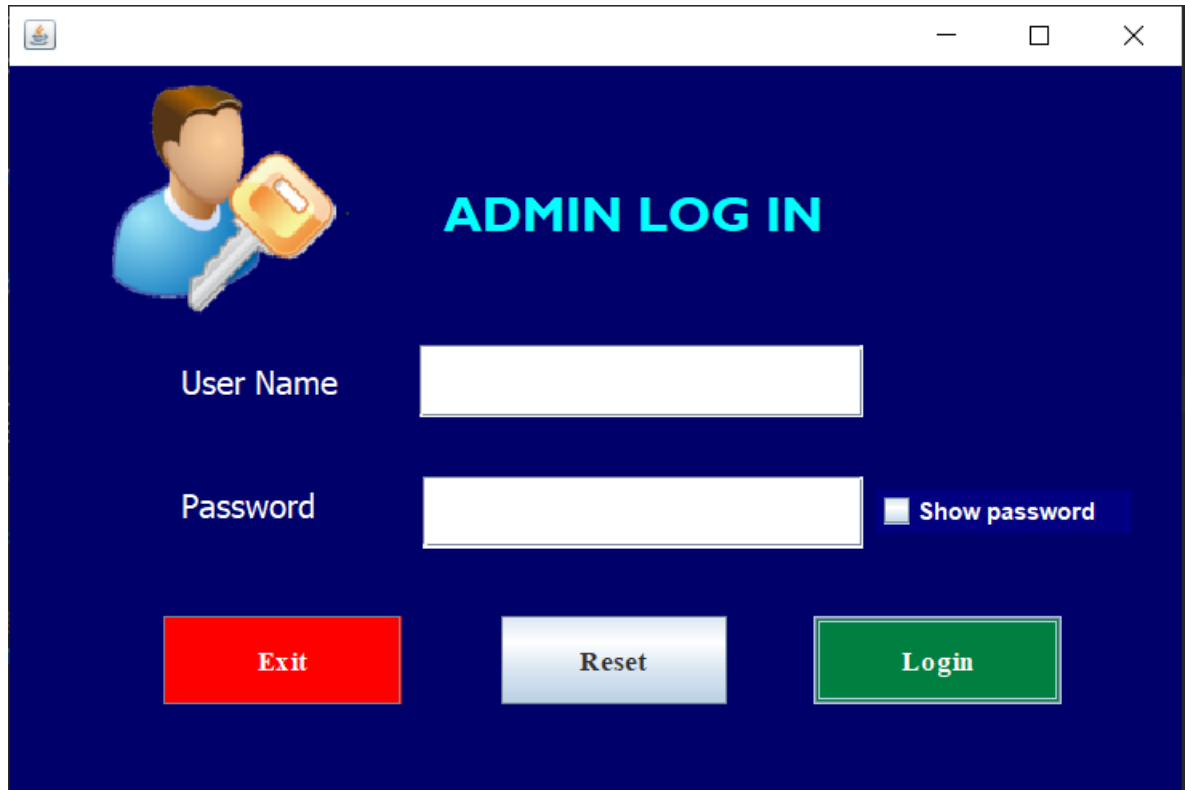
A screenshot of a web browser window displaying an 'ADMIN LOG IN' page. The page has a dark blue background. On the left, there is an illustration of a person's head and shoulders in a blue shirt, holding a large yellow key. To the right of this illustration, the text 'ADMIN LOG IN' is written in large, bold, cyan capital letters. Below the illustration and title, there are two white input fields. The first is labeled 'User Name' and the second is labeled 'Password'. To the right of the password field, there is a small square checkbox followed by the text 'Show password'. At the bottom of the page, there are three buttons: a red button labeled 'Exit', a light blue button labeled 'Reset', and a green button labeled 'Login'.

Figure 1 Admin login page

This is an admin login to the interface. The admin is already added to the database with a username and password. Then the admin should enter that username and password to log into the system. If it is correct then the admin can log in but if it is incorrect admin has an error message and try again. There is a password showing option then the admin can check whether the entered password is correct or not. Then there is the reset button admin can erase the values and when clicking the exit button admin can exit the system.

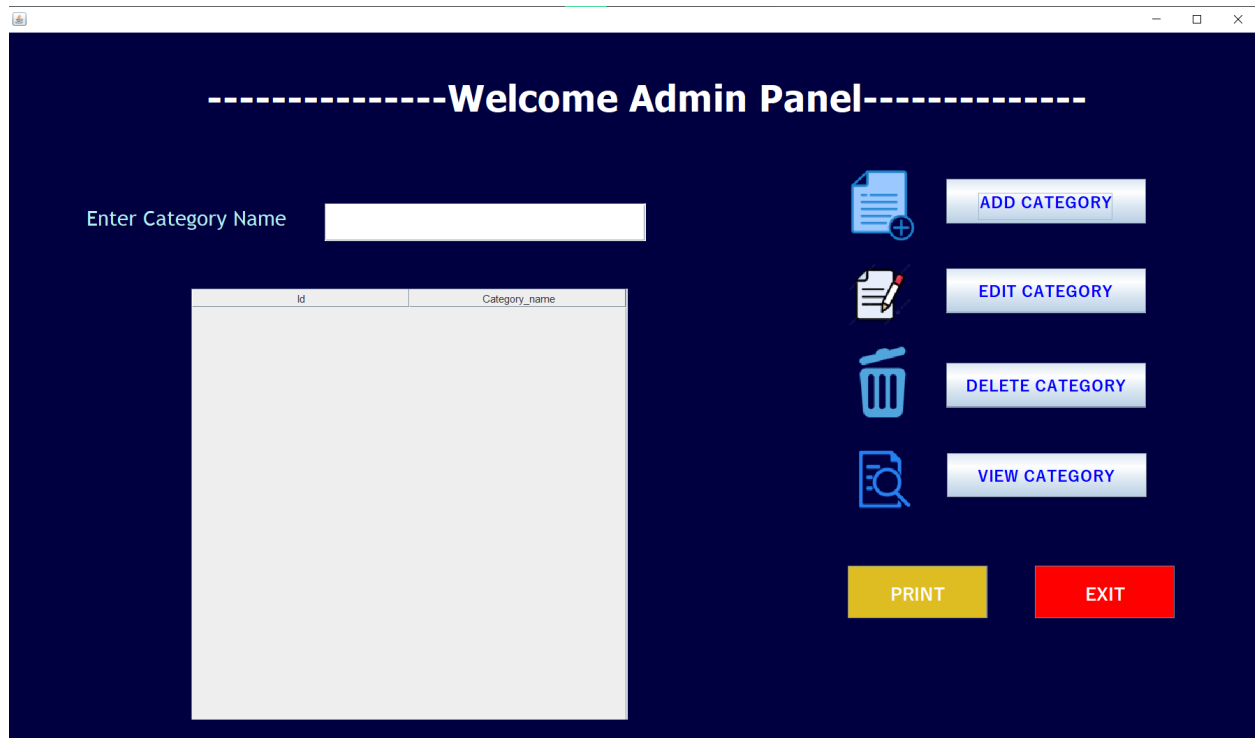
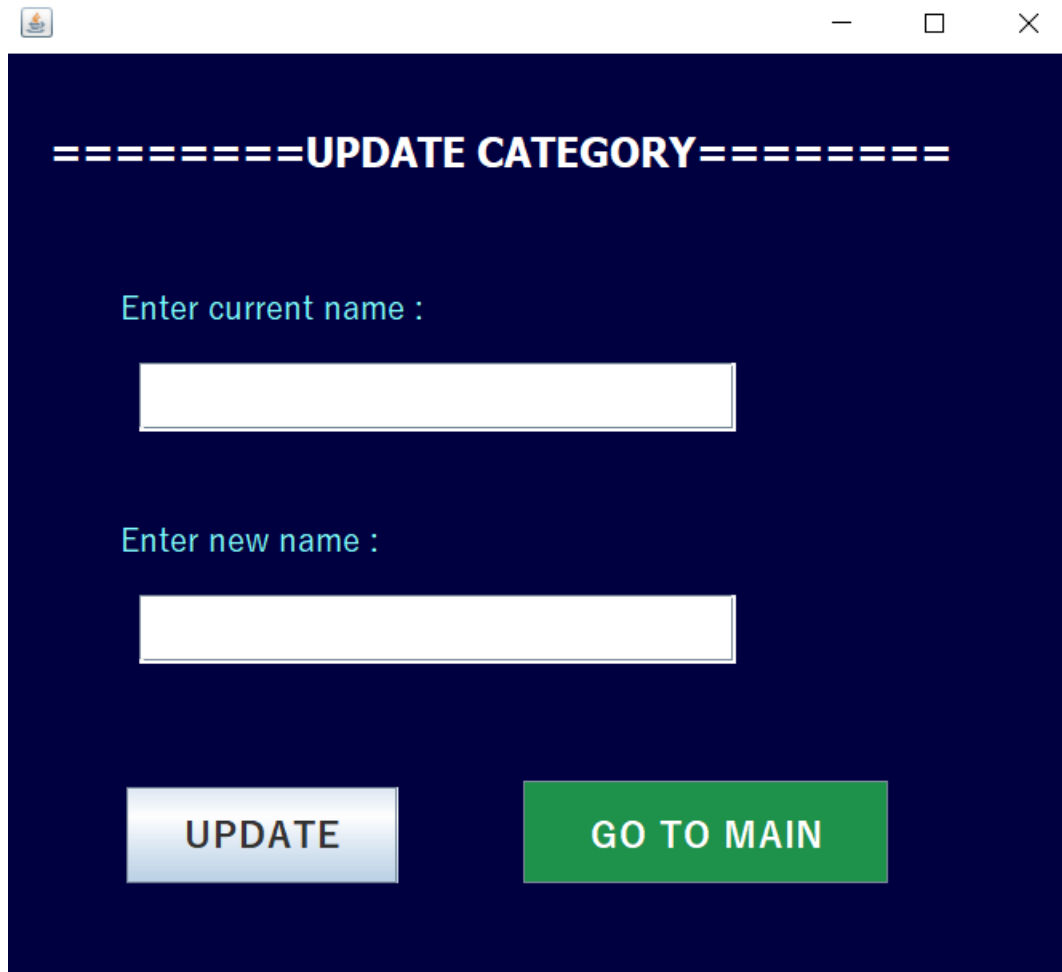


Figure 2 Admin panel

When the Admin login the admin can see this system interface. So there are Add category, update category, edit category, and delete category operations for admin. And if want the hard copy of the added categories there is a print button for that and if the admin wants to exit then the admin can exit.

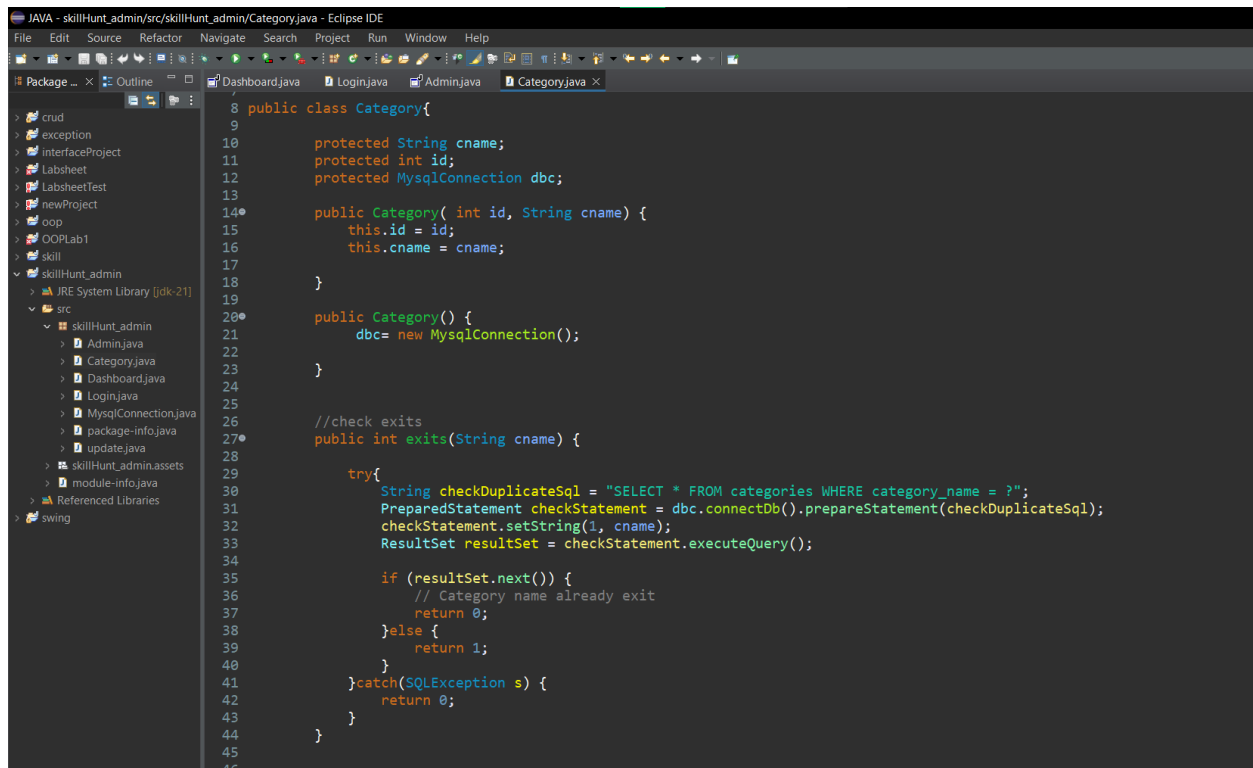


The screenshot shows a web application window with a dark blue background. At the top, the title bar contains a small icon on the left and standard window controls (minimize, maximize, close) on the right. The main content area has the title "=====UPDATE CATEGORY=====". Below the title, there are two text labels: "Enter current name :" and "Enter new name :". Each label is followed by a white rectangular input field. At the bottom of the form, there are two buttons: a light blue button labeled "UPDATE" and a green button labeled "GO TO MAIN".

Figure 3 update category

If you select the edit category button on the admin panel you can see this interface for updating categories by name. Once you finish you can back to the admin panel in *Figure 2*.

DATA VALIDATION & ERROR HANDLINGS



```
8 public class Category{
9
10     protected String cname;
11     protected int id;
12     protected MySqlConnection dbc;
13
14     public Category( int id, String cname) {
15         this.id = id;
16         this.cname = cname;
17     }
18
19
20     public Category() {
21         dbc= new MySqlConnection();
22     }
23
24
25     //check exists
26     public int exits(String cname) {
27
28         try{
29             String checkDuplicateSql = "SELECT * FROM categories WHERE category_name = ?";
30             PreparedStatement checkStatement = dbc.connectDb().prepareStatement(checkDuplicateSql);
31             checkStatement.setString(1, cname);
32             ResultSet resultSet = checkStatement.executeQuery();
33
34             if (resultSet.next()) {
35                 // Category name already exist
36                 return 0;
37             }else {
38                 return 1;
39             }
40         }catch(SQLException s) {
41             return 0;
42         }
43     }
44 }
45
46
```

Figure 4 Check already exit

This is to check if the admin added the category to the system and whether it is already in the database then show the message it already exists if it is not then add the category and show added successfully.

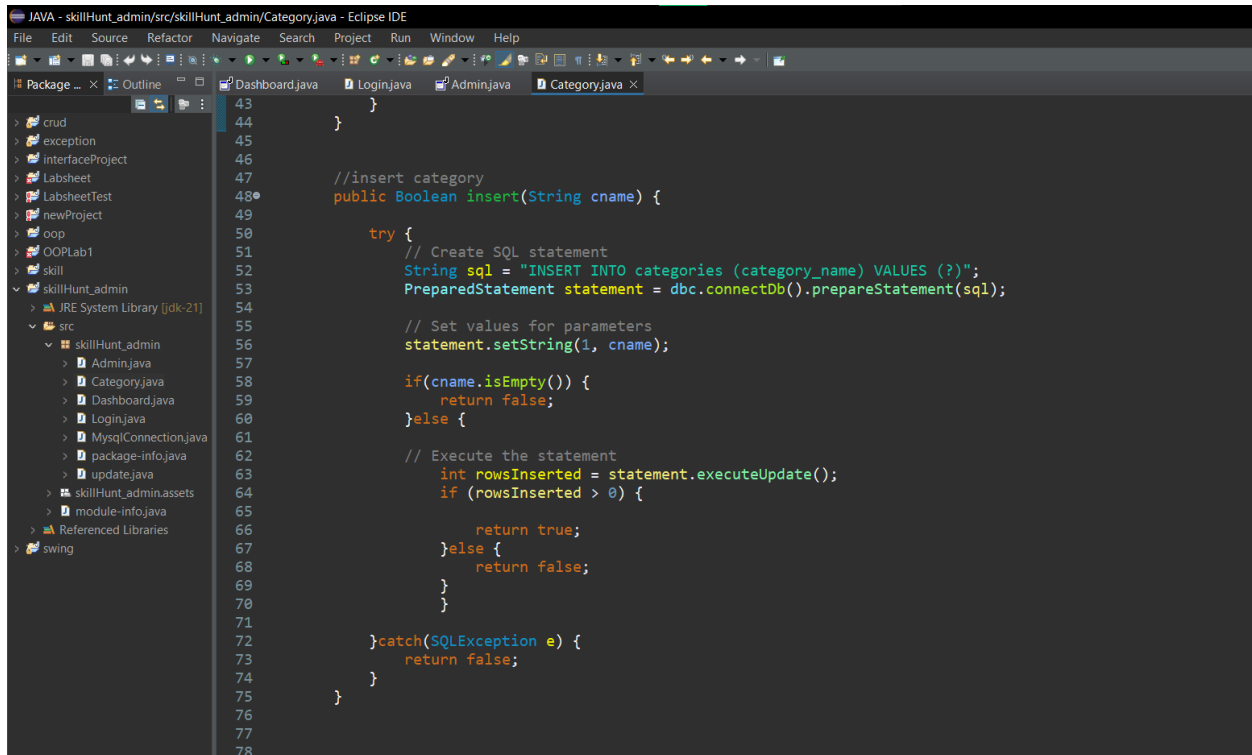


Figure 5 Check empty

This is checked when the admin clicks the add button when the values entered are not. If the admin does not enter the value and click the add button then show an error message.

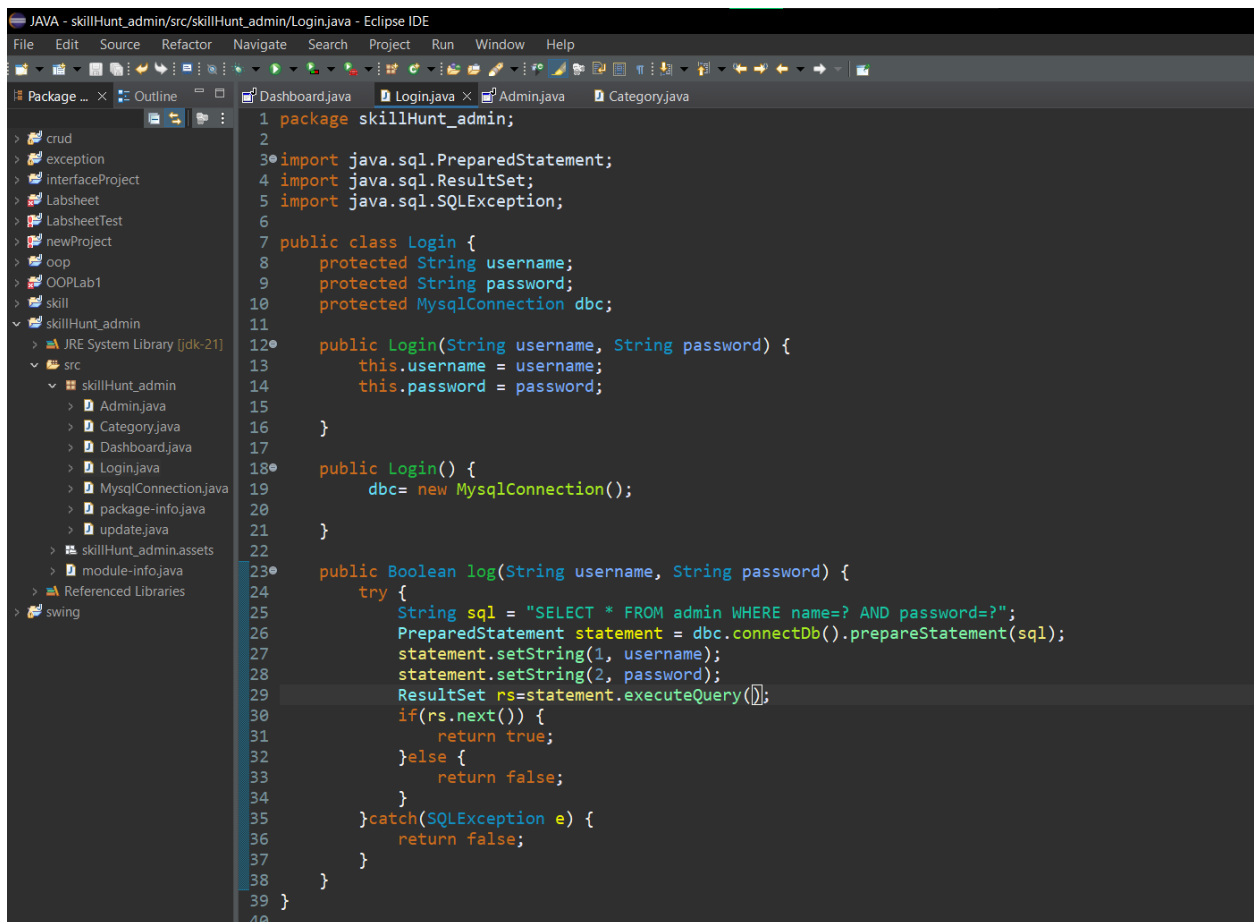
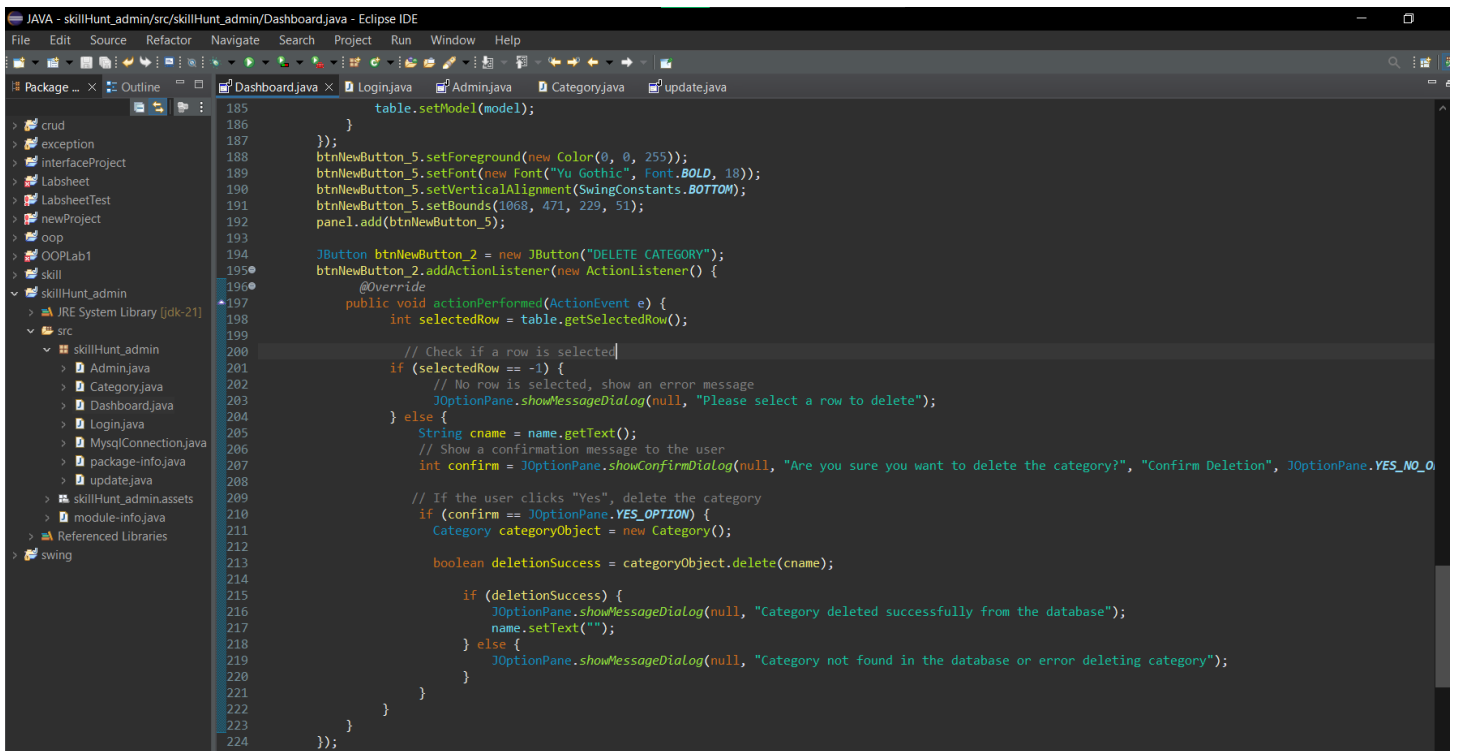


Figure 6 Valid login

This is to check Admin enter the correct password and username. If the admin enters an incorrect username and password then shows an error message and if it is correct the admin can enter the admin panel.

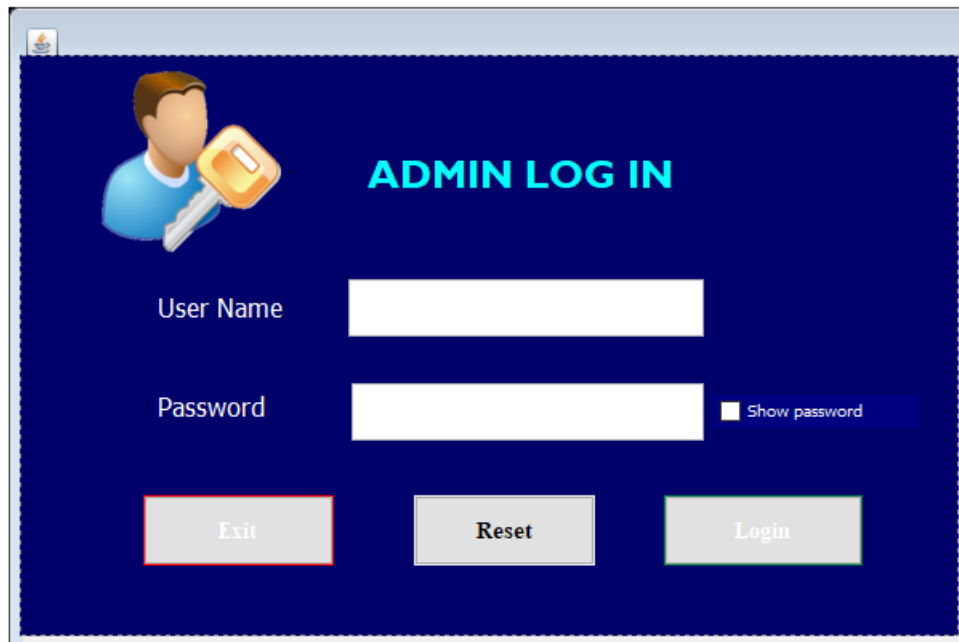
The image is a screenshot of the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The left sidebar shows a package explorer with a tree view of the project structure, including folders like crud, exception, interfaceProject, Labsheet, LabsheetTest, newProject, oop, OOPLab1, skill, skillHunt_admin, and swing. The main editor window displays the code for Dashboard.java. The code is in Java and implements an ActionListener for a delete button. It checks if a row is selected, shows a confirmation dialog, and then deletes the category if confirmed. The code is as follows:

```
185         table.setModel(model);
186     }
187 }
188 btnNewButton_5.setForeground(new Color(0, 0, 255));
189 btnNewButton_5.setFont(new Font("Yu Gothic", Font.BOLD, 18));
190 btnNewButton_5.setVerticalAlignment(SwingConstants.BOTTOM);
191 btnNewButton_5.setBounds(1068, 471, 229, 51);
192 panel.add(btnNewButton_5);
193
194 JButton btnNewButton_2 = new JButton("DELETE CATEGORY");
195 btnNewButton_2.addActionListener(new ActionListener() {
196     @Override
197     public void actionPerformed(ActionEvent e) {
198         int selectedRow = table.getSelectedRow();
199
200         // Check if a row is selected
201         if (selectedRow == -1) {
202             // No row is selected, show an error message
203             JOptionPane.showMessageDialog(null, "Please select a row to delete");
204         } else {
205             String cname = name.getText();
206             // Show a confirmation message to the user
207             int confirm = JOptionPane.showConfirmDialog(null, "Are you sure you want to delete the category?", "Confirm Deletion", JOptionPane.YES_NO_OPTION);
208
209             // If the user clicks "Yes", delete the category
210             if (confirm == JOptionPane.YES_OPTION) {
211                 Category categoryObject = new Category();
212
213                 boolean deletionSuccess = categoryObject.delete(cname);
214
215                 if (deletionSuccess) {
216                     JOptionPane.showMessageDialog(null, "Category deleted successfully from the database");
217                     name.setText("");
218                 } else {
219                     JOptionPane.showMessageDialog(null, "Category not found in the database or error deleting category");
220                 }
221             }
222         }
223     }
224 });
```

Figure 7 Confirm delete

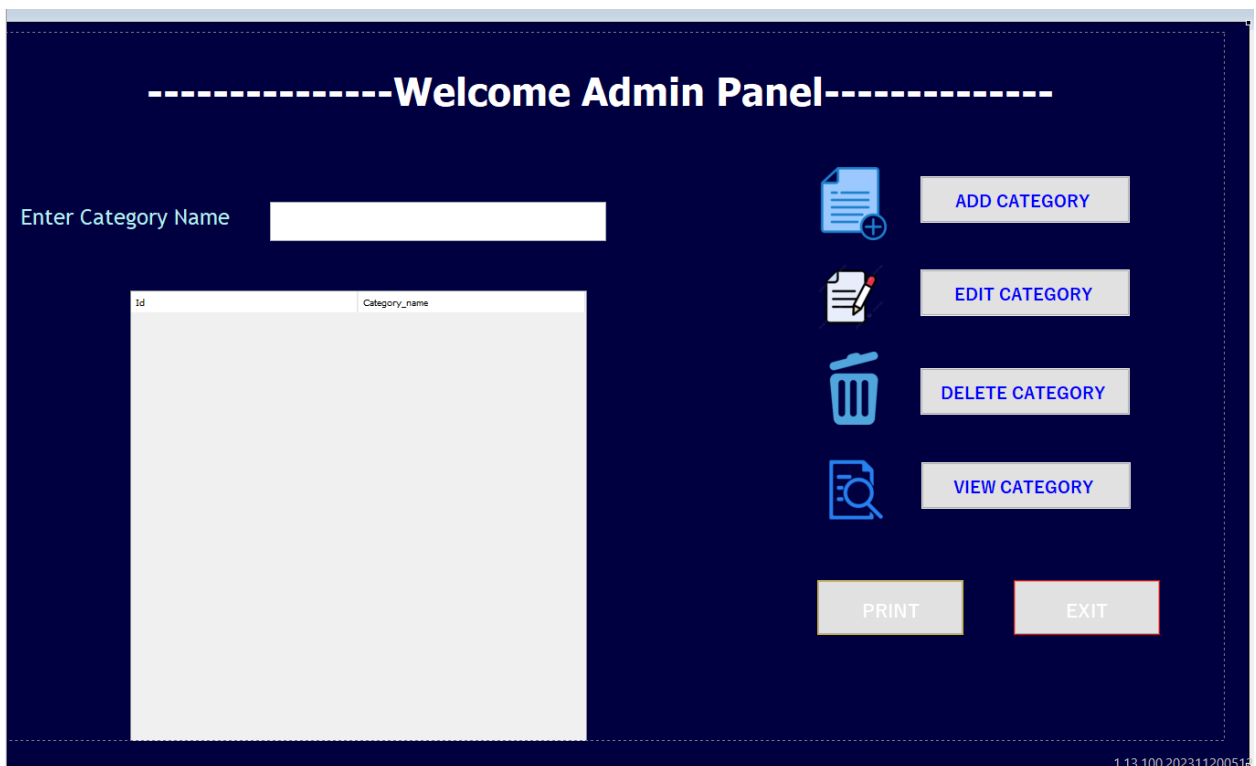
This is checked when the admin clicks on the delete button and checks if the admin selects the row first. If it is not then shows Please select the row. If the admin selects the row and clicks on the delete button then asks if you want to delete this then the admin confirms then delete if it is not then do not delete the row.

WIREFRAME



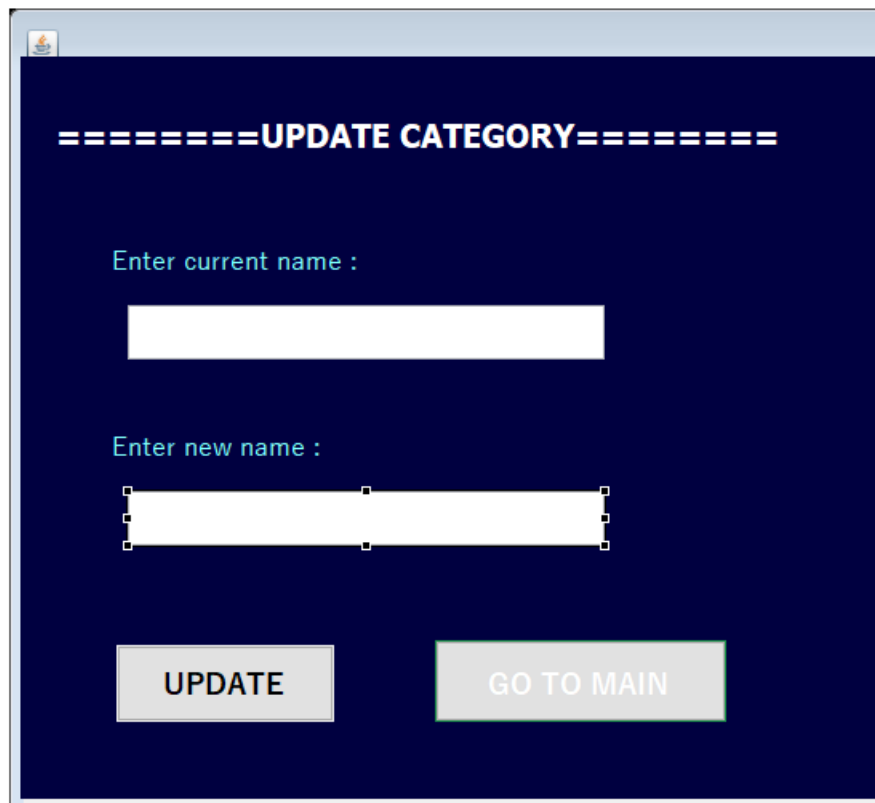
The wireframe shows an 'ADMIN LOG IN' window with a dark blue background. On the left, there is an illustration of a person holding a large key. The title 'ADMIN LOG IN' is displayed in bright blue text. Below the title, there are two white input fields: 'User Name' and 'Password'. To the right of the 'Password' field is a checkbox labeled 'Show password'. At the bottom, there are three buttons: 'Exit' (highlighted with a red border), 'Reset', and 'Login'.

Figure 8 Login



The wireframe shows a 'Welcome Admin Panel' dashboard with a dark blue background. At the top, the title 'Welcome Admin Panel' is centered and flanked by dashed lines. On the left, there is a text label 'Enter Category Name' followed by a white input field. Below this is a table with two columns: 'Id' and 'Category_name'. The table body is currently empty. On the right side, there is a vertical list of four buttons, each preceded by a blue icon: 'ADD CATEGORY' (document with plus), 'EDIT CATEGORY' (document with pencil), 'DELETE CATEGORY' (trash can), and 'VIEW CATEGORY' (document with magnifying glass). At the bottom right, there are two more buttons: 'PRINT' and 'EXIT' (highlighted with a red border). A small text string '1.13.100.202311200515' is visible in the bottom right corner of the dashboard area.

Figure 9 Dashboard



The screenshot shows a web application window with a dark blue background. At the top, the title "=====UPDATE CATEGORY===== " is displayed in white, bold, uppercase letters. Below the title, there are two text input fields. The first field is preceded by the label "Enter current name : " in a light blue font. The second field is preceded by the label "Enter new name : " in the same light blue font. Both input fields are white with black borders. Below the input fields, there are two buttons: "UPDATE" and "GO TO MAIN". Both buttons are light gray with black text. The window has a standard browser-like title bar at the top with a small icon on the left and a close button on the right.

Figure 10 Update

TEST PLAN

Test Case	
Test Unit: Login	Tester: Oneli jayodya
Test Case ID: 01	Test Type: Black Box
Test Description: Enter the Dashboard	Test Execution Date: 24/02/2024
Title: Admin Login	Test Execution Time: 11.30 a.m.

Step no.	Test step	Test case ID	Test Input	Expect Result	Actual Result	Test Result
01.	Login to the system	01	Username: admin Password: a23ewd	Not access to Login dashboard	Invalid username or password	Pass
02.	Login to the system	01	Username: oneli Password: a2024	Login to dashboard	Login Successful	pass
03.	Login to the system	01	Username: Password:	Not access to Login dashboard	Fill the details	Pass

Test Case	
Test Unit: Add category	Tester: Oneli jayodya
Test Case ID: 02	Test Type: Black Box
Test Description: Add category to system	Test Execution Date: 24/0822024
Title: Add category	Test Execution Time: 11.35 a.m.

Step no.	Test step	Test case ID	Test Input	Expect Result	Actual Result	Test Result
01.	Add category to the system	02	Enter category name: Web design	Already in the database	Category already exists!	Pass
02.	Add category to the system	02	Enter category name:	Enter category	Failed to insert category. Please try again.	pass
03.	Add category to the system	02	Enter category name: shop manager	Add to database	The category was inserted successfully!	Pass

Test Case	
Test Unit: Edit category	Tester: Oneli jayodya
Test Case ID: 03	Test Type: Black Box
Test Description: edit category to system	Test Execution Date: 24/0822024
Title: Edit category	Test Execution Time: 11.50 a.m.

Step no.	Test step	Test case ID	Test Input	Expect Result	Actual Result	Test Result
01.	Edit category to the system	03	Enter current name: Enter new name:	Enter details	Please enter values!	Pass
02.	Edit category to the system	03	Enter current name: web design Enter new name: Web development	Edit Category	Category Updated successfully !!	pass

Test Case	
Test Unit: Delete category	Tester: Oneli jayodya
Test Case ID: 04	Test Type: Black Box
Test Description: Delete category to system	Test Execution Date: 24/0822024
Title: Delete category	Test Execution Time: 12.00 a.m.

Step no.	Test step	Test case ID	Test Input	Expect Result	Actual Result	Test Result
01.	Delete the category from the system	03	Not selected a row	Select row	Please select a row to delete	Pass
02.	Delete the category from the system	03	The web design row selected	Delete Category	The category deleted successfully from the database	pass

USER DOCUMENTATION

- When this system starts first you can see the admin login. Then you have to enter your username and password and click on login. Either If the entered values are correct now you can see the dashboard or not you can see the error message.
- After you login now you are in the Admin panel. You can add, edit, view, delete categories, and print data.
- When you doing an add category you have to enter the name and click on add then you can see the insert successful message.
- Then you want to edit, you can see a new update interface when you click on the edit button. Then you have to enter your current name and change the name you want to edit. Now you click on the update button you can see the success message. You can back to the dashboard when you click on the Back to Main button.
- If you want to see what are the categories then press on view button then display values in the table.
- If you want to delete the category first select the row that are you going to delete. Then click on the delete button after that you can see the confirmation message. If you select yes then delete it. if you select not then there is no change doing.

EVIDENCE OF CODE IMPLEMENTATION

GUI Interfaces code

Admin Login Interface

```
package skillHunt_admin;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.border.EmptyBorder;
import java.awt.Color;
import javax.swing.JLabel;
import javax.swing.JOptionPane;

import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.JButton;
import javax.swing.ImageIcon;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JCheckBox;
```

```

public class Admin extends JFrame {

    private static final long serialVersionUID = 1L;

    private JPanel contentPane;

    private JTextField un;

    private JTextField pw;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {

        EventQueue.invokeLater(new Runnable() {

            public void run() {

                try {

                    Admin frame = new Admin();

                    frame.setVisible(true);

                } catch (Exception e) {

                    e.printStackTrace();

                }

            }

        });

    }

    /**
     * Create the frame.
     */

```

```
public Admin() {

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(500, 200, 600, 400);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(contentPane);
    contentPane.setLayout(null);

    JPanel panel = new JPanel();
    panel.setBackground(new Color(0, 0, 106));
    panel.setBounds(0, 0, 586, 363);
    contentPane.add(panel);
    panel.setLayout(null);

    JLabel log = new JLabel("ADMIN LOG IN");
    log.setFont(new Font("Gill Sans MT", Font.BOLD, 25));
    log.setForeground(new Color(0, 255, 255));
    log.setBounds(217, 49, 210, 51);
    panel.add(log);

    JLabel user = new JLabel("User Name");
    user.setFont(new Font("Tahoma", Font.PLAIN, 16));
    user.setForeground(new Color(255, 255, 255));
    user.setBounds(86, 140, 119, 36);
    panel.add(user);
```

```

un = new JTextField();
un.setFont(new Font("Yu Gothic UI", Font.BOLD, 15));
un.setBounds(205, 140, 222, 36);
panel.add(un);
un.setColumns(10);

JLabel word = new JLabel("Password");
word.setFont(new Font("Tahoma", Font.PLAIN, 16));
word.setForeground(new Color(255, 255, 255));
word.setBounds(86, 202, 113, 36);
panel.add(word);

pw = new JPasswordField();
pw.setFont(new Font("Yu Gothic UI", Font.BOLD, 15));
pw.setBounds(207, 205, 220, 36);
panel.add(pw);
pw.setColumns(10);

JButton exit = new JButton("Exit");
exit.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JFrame frmLogin = new JFrame("Exit");
        if (JOptionPane.showConfirmDialog(frmLogin, "Confirm if you
want to exit", "Login Systems",
JOptionPane.YES_NO_OPTION)==JOptionPane.YES_NO_OPTION) {
            System.exit(0);
        }
    }
});

```

```
exit.setForeground(new Color(255, 255, 255));
exit.setBackground(new Color(255, 0, 0));
exit.setFont(new Font("Times New Roman", Font.BOLD, 14));
exit.setBounds(77, 275, 119, 44);
panel.add(exit);

JButton reset = new JButton("Reset");
reset.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        un.setText(null);
        pw.setText(null);
    }
});
reset.setFont(new Font("Times New Roman", Font.BOLD, 14));
reset.setBounds(246, 275, 113, 44);
panel.add(reset);

JButton login = new JButton("Login");
login.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String password= pw.getText();
        String username= un.getText();

        Login lobj = new Login();
        int mzg = lobj.log(username, password);
    }
});
```

```

if(mzg ==1) {
    JOptionPane.showMessageDialog(null, "Login Successful");
    dispose();
    Dashboard db = new Dashboard();
    db.setVisible(true);
}else if(mzg==3) {
    JOptionPane.showMessageDialog(null, "Fill the details");
}else {
    JOptionPane.showMessageDialog(null, "Invalid name or password");
    }
}

});

login.setForeground(new Color(255, 255, 255));
login.setBackground(new Color(0, 128, 64));
login.setFont(new Font("Times New Roman", Font.BOLD, 14));
login.setBounds(402, 275, 124, 44);
panel.add(login);


JLabel lblNewLabel = new JLabel("New label");
lblNewLabel.setIcon(new
ImageIcon(Admin.class.getResource("/skillHunt_admin/assets/user-login-icon-29 (1).png")));
lblNewLabel.setBounds(50, 10, 120, 120);
panel.add(lblNewLabel);


JCheckBox show = new JCheckBox("Show password");
show.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(show.isSelected()) {
            ((JPasswordField) pw).setEchoChar((char)0);

```



```
        }else {  
            ((JPasswordField) pw).setEchoChar('*');  
        }  
    }  
});  
show.setForeground(new Color(255, 255, 255));  
show.setBackground(new Color(0, 0, 128));  
show.setBounds(433, 212, 128, 21);  
panel.add(show);  
}  
}
```

Admin Dashboard Interface

```
package skillHunt_admin;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import java.awt.Color;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.SwingConstants;
import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import javax.swing.JScrollPane;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.print.PrinterException;
```

```

public class Dashboard extends JFrame {

    private static final long serialVersionUID = 1L;

    private JPanel contentPane;

    private JTextField name;

    DefaultTableModel model;

    private JTable table;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Dashboard frame = new Dashboard();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */

```

```

public Dashboard() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(0, 0, 1450, 850);
    contentPane = new JPanel();
    contentPane.setBackground(new Color(0, 0, 64));
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(contentPane);
    contentPane.setLayout(null);

    JPanel panel = new JPanel();
    panel.setBackground(new Color(0, 0, 64));
    panel.setBounds(10, 10, 1400, 776);
    contentPane.add(panel);
    panel.setLayout(null);

    JLabel lblNewLabel = new JLabel("-----Welcome Admin Panel--
-----");
    lblNewLabel.setBounds(189, 39, 1070, 51);
    lblNewLabel.setForeground(new Color(255, 255, 255));
    lblNewLabel.setFont(new Font("Tahoma", Font.BOLD, 42));
    lblNewLabel.setHorizontalAlignment(SwingConstants.CENTER);
    panel.add(lblNewLabel);

    JLabel lblNewLabel_1 = new JLabel("Enter Category Name ");
    lblNewLabel_1.setFont(new Font("Trebuchet MS", Font.PLAIN, 24));

```

```
lblNewLabel_1.setForeground(new Color(177, 241, 243));  
lblNewLabel_1.setBounds(82, 174, 318, 57);  
panel.add(lblNewLabel_1);
```

```
name = new JTextField();  
name.setFont(new Font("Tahoma", Font.BOLD, 18));  
name.setBounds(355, 186, 368, 43);  
panel.add(name);  
name.setColumns(10);
```

```
JLabel lblNewLabel_2 = new JLabel("New label");  
lblNewLabel_2.setIcon(new  
ImageIcon(Dashboard.class.getResource("/skillHunt_admin/assets/add  
(1).png")));  
lblNewLabel_2.setBounds(954, 146, 81, 85);  
panel.add(lblNewLabel_2);
```

```
JLabel label = new JLabel("New label");  
label.setIcon(new  
ImageIcon(Dashboard.class.getResource("/skillHunt_admin/assets/edit  
(1).png")));  
label.setBounds(951, 248, 84, 85);  
panel.add(label);
```

```

JLabel lblNewLabel_3 = new JLabel("New label");
lblNewLabel_3.setFont(new Font("Yu Gothic", Font.BOLD, 18));
lblNewLabel_3.setIcon(new
ImageIcon(Dashboard.class.getResource("/skillHunt_admin/assets/delete
(1).png")));

lblNewLabel_3.setBounds(954, 348, 81, 85);
panel.add(lblNewLabel_3);


JButton btnNewButton_4 = new JButton("EXIT");
btnNewButton_4.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JFrame frame=new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if
you want to exit","MYSQL connector",
JOptionPane.YES_NO_OPTION)==JOptionPane.YES_NO_OPTION)
        {
            System.exit(0);
        }
    }
});

btnNewButton_4.setBackground(new Color(255, 0, 0));
btnNewButton_4.setVerticalAlignment(SwingConstants.BOTTOM);
btnNewButton_4.setFont(new Font("Yu Gothic", Font.BOLD, 20));
btnNewButton_4.setForeground(new Color(255, 255, 255));

```

```

        btnNewButton_4.setBounds(1169, 600, 160, 60);
        panel.add(btnNewButton_4);

        JScrollPane scrollPane = new JScrollPane();
        scrollPane.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                DefaultTableModel tblmodel =
                (DefaultTableModel)table.getModel();

                String tblname =
                tblmodel.getValueAt(table.getSelectedRow(), 2).toString();

                name.setText(tblname);

            }
        });
        scrollPane.setBounds(202, 283, 500, 500);
        panel.add(scrollPane);

        table = new JTable(model);
        scrollPane.setViewportViewView(table);
        table.setModel(new DefaultTableModel(
            new Object[][] {
                },

```

```

        new String[] {
            "Id", "Category_name"
        }
    ));

    JButton add = new JButton("ADD CATEGORY");
    add.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            String cname = name.getText();
            Category categoryObject = new Category();
            int suc = categoryObject.exists(cname);

            if (suc == 0) {
                JOptionPane.showMessageDialog(null, "Category already
exists!");
            } else {
                boolean success = categoryObject.insert(cname);

                if (success) {
                    JOptionPane.showMessageDialog(null, "Category
inserted successfully!");
                } else {
                    JOptionPane.showMessageDialog(null, "Failed to insert
category. Please try again.");
                }
            }
        }
    });

```



```

        }

    });

    add.setVerticalAlignment(SwingConstants.BOTTOM);
    add.setForeground(new Color(0, 0, 255));
    add.setFont(new Font("Yu Gothic", Font.BOLD, 18));
    add.setBounds(1067, 158, 229, 51);
    panel.add(add);

    JButton btnNewButton_5 = new JButton("VIEW CATEGORY");
    btnNewButton_5.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            Category cobj = new Category();
            model = cobj.view();
            table.setModel(model);
        }
    });

    btnNewButton_5.setForeground(new Color(0, 0, 255));
    btnNewButton_5.setFont(new Font("Yu Gothic", Font.BOLD, 18));
    btnNewButton_5.setVerticalAlignment(SwingConstants.BOTTOM);
    btnNewButton_5.setBounds(1068, 471, 229, 51);
    panel.add(btnNewButton_5);

    JButton btnNewButton_2 = new JButton("DELETE CATEGORY");
    btnNewButton_2.addActionListener(new ActionListener() {

```

@Override

```
public void actionPerformed(ActionEvent e) {  
    int selectedRow = table.getSelectedRow();  
  
    // Check if a row is selected  
    if (selectedRow == -1) {  
        // No row is selected, show an error message  
        JOptionPane.showMessageDialog(null, "Please select a  
row to delete");  
    } else {  
        String cname = name.getText();  
        // Show a confirmation message to the user  
        int confirm = JOptionPane.showConfirmDialog(null,  
"Are you sure you want to delete the category?", "Confirm Deletion",  
JOptionPane.YES_NO_OPTION);  
  
        // If the user clicks "Yes", delete the category  
        if (confirm == JOptionPane.YES_OPTION) {  
            Category categoryObject = new Category();  
  
            boolean deletionSuccess =  
categoryObject.delete(cname);  
  
            if (deletionSuccess) {  
                JOptionPane.showMessageDialog(null, "Category  
deleted successfully from the database");  
                name.setText("");  
            }  
        }  
    }  
}
```

```

        } else {
            JOptionPane.showMessageDialog(null, "Category not
found in the database or error deleting category");
        }
    }
}

});

btnNewButton_2.setForeground(new Color(0, 0, 255));
btnNewButton_2.setVerticalAlignment(SwingConstants.BOTTOM);
btnNewButton_2.setFont(new Font("Yu Gothic", Font.BOLD, 18));
btnNewButton_2.setBounds(1067, 368, 229, 51);
panel.add(btnNewButton_2);

JButton btnNewButton_1 = new JButton("EDIT CATEGORY");
btnNewButton_1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        dispose();
        update edit = new update();
        edit.setVisible(true);
    }
});

btnNewButton_1.setVerticalAlignment(SwingConstants.BOTTOM);
btnNewButton_1.setForeground(new Color(0, 0, 255));

btnNewButton_1.setFont(new Font("Yu Gothic", Font.BOLD, 18));

```

```
btnNewButton_1.setBounds(1067, 260, 229, 51);
```

```
panel.add(btnNewButton_1);
```

```
JButton btnNewButton = new JButton("PRINT");
```

```
btnNewButton.addActionListener(new ActionListener() {
```

```
    public void actionPerformed(ActionEvent e) {
```

```
        try {
```

```
            table.print();
```

```
        }catch(PrinterException p) {
```

```
            System.out.format("No printer found");
```

```
        }
```

```
    }
```

```
});
```

```
btnNewButton.setBackground(new Color(222, 189, 35));
```

```
btnNewButton.setFont(new Font("Yu Gothic", Font.BOLD, 20));
```

```
btnNewButton.setVerticalAlignment(SwingConstants.BOTTOM);
```

```
btnNewButton.setForeground(new Color(255, 255, 255));
```

```
btnNewButton.setBounds(954, 600, 160, 60);
```

```
panel.add(btnNewButton);
```

```
JLabel lblNewLabel_4 = new JLabel("");
```

```
lblNewLabel_4.setIcon(new ImageIcon(Dashboard.class.getResource("/skill  
Hunt_admin/assets/view (1).png"));
```

```
lblNewLabel_4.setBounds(954, 459, 81, 85);
```

```
panel.add(lblNewLabel_4);
```

```
}}
```

Update Interface

```
package skillHunt_admin;
```

```
import java.awt.EventQueue;
```

```
import javax.swing.JFrame;
```

```
import javax.swing.JPanel;
```

```
import javax.swing.border.EmptyBorder;
```

```
import java.awt.Color;
```

```
import javax.swing.JLabel;
```

```
import javax.swing.JOptionPane;
```

```
import java.awt.Font;
```

```
import javax.swing.JTextField;
```

```
import javax.swing.JButton;
```

```
import javax.swing.SwingConstants;
```

```
import java.awt.event.ActionListener;
```

```
import java.awt.event.ActionEvent;
```

```
public class update extends JFrame {
```

```
    private static final long serialVersionUID = 1L;
```

```
    private JPanel contentPane;
```

```
    private JTextField oldtext;
```

```
    private JTextField newtext;
```

```

/**
 * Launch the application.
 */
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                update frame = new update();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public update() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(500, 200, 550, 500);
    contentPane = new JPanel();
    contentPane.setBackground(new Color(0, 0, 64));
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
}

```

```

        setContentPane(contentPane);
        contentPane.setLayout(null);

        JLabel lblNewLabel = new JLabel("=====UPDATE
CATEGORY=====");
        lblNewLabel.setFont(new Font("Tahoma", Font.BOLD, 20));
        lblNewLabel.setForeground(new Color(255, 255, 255));
        lblNewLabel.setBounds(22, 25, 464, 48);
        contentPane.add(lblNewLabel);

        JLabel lblNewLabel_1 = new JLabel("Enter current name :");
        lblNewLabel_1.setFont(new Font("Yu Gothic Medium", Font.PLAIN,
16));
        lblNewLabel_1.setForeground(new Color(128, 255, 255));
        lblNewLabel_1.setBounds(57, 118, 238, 27);
        contentPane.add(lblNewLabel_1);

        oldtext = new JTextField();
        oldtext.setFont(new Font("Tahoma", Font.BOLD, 17));
        oldtext.setBounds(67, 155, 298, 34);
        contentPane.add(oldtext);
        oldtext.setColumns(10);

        JLabel lblNewLabel_2 = new JLabel("Enter new name :");
        lblNewLabel_2.setFont(new Font("Yu Gothic Medium", Font.PLAIN,
16));

```

```

lblNewLabel_2.setForeground(new Color(128, 255, 255));
lblNewLabel_2.setBounds(57, 234, 238, 27);
contentPane.add(lblNewLabel_2);

newtext = new JTextField();
newtext.setFont(new Font("Tahoma", Font.BOLD, 17));
newtext.setBounds(67, 271, 298, 34);
contentPane.add(newtext);
newtext.setColumns(10);

JButton btnNewButton = new JButton("UPDATE");
btnNewButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        String oldName = oldtext.getText();
        String newName = newtext.getText();
        Category categoryObject = new Category();

        int edit = categoryObject.update(oldName,newName);

        if (edit==0) {
JOptionPane.showMessageDialog(null, "Please enter values!");
        }else {

```



```
int confirm = JOptionPane.showConfirmDialog(null, "Are you sure you want to
edit the category?", "Confirm Update", JOptionPane.YES_NO_OPTION);
```

```
    if (confirm == JOptionPane.YES_OPTION) {
```

```
        if (edit==1) {
            JOptionPane.showMessageDialog(null, "Please enter
values!");
```

```
            JOptionPane.showMessageDialog(null, "Category
Updated successfully !!");
```

```
            oldtext.setText("");
```

```
            newtext.setText("");
```

```
        } else {
```

```
            JOptionPane.showMessageDialog(null, "Category not
found in the database or error updating category");
```

```
        }
```

```
    }
```

```
}
```

```
}
```

```
});
```

```
btnNewButton.setFont(new Font("Yu Gothic", Font.BOLD, 18));
```

```
btnNewButton.setVerticalAlignment(SwingConstants.BOTTOM);
```

```
btnNewButton.setBounds(60, 367, 136, 48);
```

```
contentPane.add(btnNewButton);
```

```
        JButton btnNewButton_1 = new JButton("GO TO MAIN");
        btnNewButton_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                dispose();
                Dashboard dash = new Dashboard();
                dash.setVisible(true);
            }
        });
        btnNewButton_1.setBackground(new Color(30, 145, 75));
        btnNewButton_1.setForeground(new Color(255, 255, 255));
        btnNewButton_1.setFont(new Font("Yu Gothic", Font.BOLD, 18));
        btnNewButton_1.setVerticalAlignment(SwingConstants.BOTTOM);
        btnNewButton_1.setBounds(259, 364, 182, 51);
        contentPane.add(btnNewButton_1);
    }
}
```

Java Class Codes

Connection Class

```
package skillHunt_admin;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class MySqlConnection {
    private String username;
    private String pwd;
    private String dbname;
    private Connection conn;

    public MySqlConnection() {
        this.dbname = "skill_hunt";
        this.username = "root";
        this.pwd = "";
        connectDb();
    }

    public Connection connectDb() {

        try {

            conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/"+dbname,username,pwd);

            return conn;
        }
    }
}
```

```
}  
  
    catch(SQLException e) {  
        System.out.println(e);  
        return conn;  
    }  
  
}  
  
}
```

Login Class

```
package skillHunt_admin;
```

```
import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
```

```
public class Login {
```

```
    protected String username;
```

```
    protected String password;
```

```
    protected MySqlConnection dbc;
```

```
    public Login(String username, String password) {
```

```
        this.username = username;
```

```
        this.password = password;
```

```
    }
```

```
    public Login() {
```

```
        dbc= new MySqlConnection();
```

```
    }
```

```
    public int log(String username, String password) {
```

```
        try {
```

```
String sql = "SELECT * FROM admin WHERE name=? AND password=?";
```

```
        PreparedStatement statement =  
dbc.connectDb().prepareStatement(sql);
```

```
        statement.setString(1, username);
```

```
        statement.setString(2, password);
```

```
        ResultSet rs=statement.executeQuery();
```

```
        if(username.isEmpty() | password.isEmpty() ) {
```

```
            return 3;
```

```
        }else {
```

```
            if(rs.next()) {
```

```
                return 1;
```

```
            }else {
```

```
                return 0;
```

```
            }
```

```
        }
```

```
    }catch(SQLException e) {
```

```
        return 2;
```

```
    }
```

```
}
```

```
}
```

Category Class

```
package skillHunt_admin;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.table.DefaultTableModel;

public class Category{

    protected String cname;
    protected int id;
    protected MySqlConnection dbc;

    public Category( int id, String cname) {
        this.id = id;
        this.cname = cname;
    }

    public Category() {
        dbc= new MySqlConnection();
    }

    //check exits
    public int exits(String cname) {
```

```

        try{
            String checkDuplicateSql = "SELECT * FROM categories WHERE category_name =
?";

            PreparedStatement checkStatement =
dbc.connectDb().prepareStatement(checkDuplicateSql);

            checkStatement.setString(1, cname);

            ResultSet resultSet = checkStatement.executeQuery();

            if (resultSet.next()) {
                // Category name already exist
                return 0;
            }else {
                return 1;
            }
        }catch(SQLException s) {
            return 0;
        }
    }

    //insert category
    public Boolean insert(String cname) {

        try {
            // Create SQL statement
            String sql = "INSERT INTO categories (category_name) VALUES (?)";
            PreparedStatement statement = dbc.connectDb().prepareStatement(sql);

            // Set values for parameters

```



```

statement.setString(1, cname);

if(cname.isEmpty()) {
    return false;
}

// Execute the statement
int rowsInserted = statement.executeUpdate();
if (rowsInserted > 0) {

    return true;
}

return false;
}

}catch(SQLException e) {
    return false;
}

}

//view category
public DefaultTableModel view() {
    DefaultTableModel model = new DefaultTableModel();

    try {

        // Retrieve data from the database

```

```

String sql = "SELECT * FROM categories";

PreparedStatement statement = dbc.connectDb().prepareStatement(sql);

ResultSet resultSet = statement.executeQuery();

// Get column names
int columnCount = resultSet.getMetaData().getColumnCount();
for (int i = 1; i <= columnCount; i++) {
    model.addColumn(resultSet.getMetaData().getColumnName(i));
}

// Get data rows
while (resultSet.next()) {
    Object[] rowData = new Object[columnCount];
    for (int i = 0; i < columnCount; i++) {
        rowData[i] = resultSet.getObject(i + 1);
    }
    model.addRow(rowData);
}

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

return model;
}

//edit category

```

```

public int update(String oldName, String newName) {
    try {
        if (newName.isEmpty()) {
            return 0;
        } else {
            String updateSql = "UPDATE categories SET category_name = ? WHERE
category_name = ?";

```

```

        PreparedStatement statement =
        dbc.connectDb().prepareStatement(updateSql);

        statement.setString(1, newName);
        statement.setString(2, oldName);

        int rowsUpdated = statement.executeUpdate();
        if (rowsUpdated > 0) {
            return 1;
        } else {
            return 2;
        }
    }
} catch (SQLException e) {
    return 2;
}
}

```

//delete category

```

public boolean delete(String cname) {

```

```
try {  
  
    // Delete category from the database  
  
    String deleteQuery = "DELETE FROM categories WHERE category_name = ?";  
    PreparedStatement preparedStatement =  
dbc.connectDb().prepareStatement(deleteQuery);  
    preparedStatement.setString(1, cname);  
    int rowsAffected = preparedStatement.executeUpdate();  
  
    if (rowsAffected > 0) {  
        return true;  
    }  
  
    } else {  
        return false;  
    }  
    } catch (SQLException e) {  
  
        return false;  
    }  
    }  
}
```

REFERENCES

<https://www.tutussfunny.com/java-mysql-oop-java-swing/>

<http://youtube.com>

<https://chat.openai.com/>