**//ZOOMODULE**

package zoomodule;

import java.util.ArrayList;

abstract class ZooModule {

protected ArrayList<String[]> data = new ArrayList<>();

public abstract void showModule();

public abstract void addData(String[] fields);

public abstract void viewData();

public abstract void updateData(String[] fields);

public abstract void deleteData(String id);

}

**//ANIMAL MANAGEMENT SYSTEM**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

class AnimalManagement extends ZooModule {

@Override

public void showModule() {

JFrame frame = new JFrame("Animal Management");

frame.setSize(800, 600);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

frame.getContentPane().setBackground(Color.ORANGE); // Background color

JLabel titleLabel = new JLabel("Animal Management", SwingConstants.CENTER);

titleLabel.setBounds(0, 10, 800, 50);

titleLabel.setFont(new Font("Arial", Font.BOLD, 40));

frame.add(titleLabel);

JTextField idField = createTextField(frame, "ID:", 70);

JTextField nameField = createTextField(frame, "Name:", 130);

JTextField ageField = createTextField(frame, "Age:", 190);

JTextField locationField = createTextField(frame, "Location:", 250);

// Buttons for CRUD operations

createButton(frame, "Add", 350, e -> {

addData(new String[]{idField.getText(), nameField.getText(), ageField.getText(), locationField.getText()});

});

createButton(frame, "View", 420, e -> viewData());

createButton(frame, "Update", 490, e -> updateData(new String[]{idField.getText(), nameField.getText(), ageField.getText(), locationField.getText()}));

createButton(frame, "Delete", 560, e -> deleteData(idField.getText()));

createButton(frame, "Back to Menu", 200, e -> frame.dispose());

createButton(frame, "Exit", 70, e -> System.exit(0));

frame.setVisible(true);

}

private JTextField createTextField(JFrame frame, String labelText, int y) {

JLabel label = new JLabel(labelText);

label.setBounds(50, y, 150, 40);

label.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(label);

JTextField textField = new JTextField();

textField.setBounds(200, y, 250, 40);

textField.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(textField);

return textField;

}

private void createButton(JFrame frame, String text, int y, ActionListener actionListener) {

JButton button = new JButton(text);

button.setBounds(500, y, 250, 50);

button.setFont(new Font("Arial", Font.PLAIN, 30));

button.setBackground(Color.GREEN);

button.setForeground(Color.WHITE);

button.addActionListener(actionListener);

frame.add(button);

}

@Override

public void addData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

data.add(fields);

JOptionPane.showMessageDialog(null, "Animal Record Added Successfully.");

}

}

@Override

public void viewData() {

if (data.isEmpty()) {

JOptionPane.showMessageDialog(null, "No Animal Data Available to View.");

return;

}

StringBuilder viewData = new StringBuilder("Animal Data:\n");

for (String[] record : data) {

viewData.append("ID: ").append(record[0])

.append(", Name: ").append(record[1])

.append(", Age: ").append(record[2])

.append(", Location: ").append(record[3]).append("\n");

}

JOptionPane.showMessageDialog(null, viewData.toString());

}

@Override

public void updateData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

for (String[] record : data) {

if (record[0].equals(fields[0])) { // Match ID

record[1] = fields[1];

record[2] = fields[2];

record[3] = fields[3];

JOptionPane.showMessageDialog(null, "Animal Record Updated Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Animal Record Not Found.");

}

}

@Override

public void deleteData(String id) {

if (id == null || id.trim().isEmpty()) { // Validate ID

JOptionPane.showMessageDialog(null, "Please Enter an ID to Delete.");

return;

}

for (String[] record : data) {

if (record[0].equals(id)) { // Match ID

data.remove(record);

JOptionPane.showMessageDialog(null, "Animal Record Deleted Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Animal Record Not Found.");

}

/\*\*

\* Validates if all required fields are filled and valid.

\*

\* @param fields The array of fields to validate.

\* @return true if all fields are valid, false otherwise.

\*/

private boolean validateFields(String[] fields) {

String[] fieldNames = {"ID", "Name", "Age", "Location"}; // Field labels for error messages

for (int i = 0; i < fields.length; i++) {

if (fields[i] == null || fields[i].trim().isEmpty()) { // Check for null or empty fields

JOptionPane.showMessageDialog(null, "Field '" + fieldNames[i] + "' must be filled.");

return false;

}

}

// Additional validation for Age (numeric and positive)

try {

int age = Integer.parseInt(fields[2]);

if (age <= 0) {

JOptionPane.showMessageDialog(null, "Age must be a positive integer.");

return false;

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(null, "Age must be a valid number.");

return false;

}

return true;

}

}

**//INVENTORY MANAGEMENT SYSTEM**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionListener;

public class InventoryManagement extends ZooModule {

@Override

public void showModule() {

JFrame frame = new JFrame("Inventory Management");

frame.setSize(800, 600);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

frame.getContentPane().setBackground(Color.RED); // Background color

JLabel titleLabel = new JLabel("Inventory Management", SwingConstants.CENTER);

titleLabel.setBounds(0, 10, 800, 50);

titleLabel.setFont(new Font("Arial", Font.BOLD, 40));

frame.add(titleLabel);

JTextField idField = createTextField(frame, "ID:", 70);

JTextField typeField = createTextField(frame, "Type:", 130);

JTextField quantityField = createTextField(frame, "Quantity:", 190);

JTextField locationField = createTextField(frame, "Location:", 250);

// Buttons for CRUD operations

createButton(frame, "Add", 350, e -> {

addData(new String[]{idField.getText(), typeField.getText(), quantityField.getText(), locationField.getText()});

});

createButton(frame, "View", 420, e -> viewData());

createButton(frame, "Update", 490, e -> updateData(new String[]{idField.getText(), typeField.getText(), quantityField.getText(), locationField.getText()}));

createButton(frame, "Delete", 560, e -> deleteData(idField.getText()));

createButton(frame, "Back to Menu", 200, e -> frame.dispose());

createButton(frame, "Exit", 70, e -> System.exit(0));

frame.setVisible(true);

}

private JTextField createTextField(JFrame frame, String labelText, int y) {

JLabel label = new JLabel(labelText);

label.setBounds(50, y, 150, 40);

label.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(label);

JTextField textField = new JTextField();

textField.setBounds(200, y, 250, 40);

textField.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(textField);

return textField;

}

private void createButton(JFrame frame, String text, int y, ActionListener actionListener) {

JButton button = new JButton(text);

button.setBounds(500, y, 250, 50);

button.setFont(new Font("Arial", Font.PLAIN, 30));

button.setBackground(Color.ORANGE);

button.setForeground(Color.WHITE);

button.addActionListener(actionListener);

frame.add(button);

}

@Override

public void addData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

data.add(fields);

JOptionPane.showMessageDialog(null, "Inventory Record Added Successfully.");

}

}

@Override

public void viewData() {

if (data.isEmpty()) {

JOptionPane.showMessageDialog(null, "No Inventory Data Available to View.");

return;

}

StringBuilder viewData = new StringBuilder("Inventory Data:\n");

for (String[] record : data) {

viewData.append("ID: ").append(record[0])

.append(", Type: ").append(record[1])

.append(", Quantity: ").append(record[2])

.append(", Location: ").append(record[3]).append("\n");

}

JOptionPane.showMessageDialog(null, viewData.toString());

}

@Override

public void updateData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

for (String[] record : data) {

if (record[0].equals(fields[0])) { // Match ID

record[1] = fields[1];

record[2] = fields[2];

record[3] = fields[3];

JOptionPane.showMessageDialog(null, "Inventory Record Updated Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Inventory Record Not Found.");

}

}

@Override

public void deleteData(String id) {

if (id == null || id.trim().isEmpty()) { // Validate ID

JOptionPane.showMessageDialog(null, "Please Enter an ID to Delete.");

return;

}

for (String[] record : data) {

if (record[0].equals(id)) { // Match ID

data.remove(record);

JOptionPane.showMessageDialog(null, "Inventory Record Deleted Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Inventory Record Not Found.");

}

/\*\*

\* Validates if all required fields are filled.

\*

\* @param fields The array of fields to validate.

\* @return true if all fields are valid, false otherwise.

\*/

private boolean validateFields(String[] fields) {

String[] fieldNames = {"ID", "Type", "Quantity", "Location"}; // Field labels for error messages

for (int i = 0; i < fields.length; i++) {

if (fields[i] == null || fields[i].trim().isEmpty()) { // Check for null or empty fields

JOptionPane.showMessageDialog(null, "Field '" + fieldNames[i] + "' must be filled.");

return false;

}

}

// Additional validation for numeric fields

try {

int quantity = Integer.parseInt(fields[2]);

if (quantity <= 0) {

JOptionPane.showMessageDialog(null, "Quantity must be a positive integer.");

return false;

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(null, "Quantity must be a valid number.");

return false;

}

return true;

}

}

**//TICKET MANAGEMENT SYSTEM**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class TicketManagement extends ZooModule {

@Override

public void showModule() {

JFrame frame = new JFrame("Ticket Management");

frame.setSize(800, 600);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

frame.getContentPane().setBackground(Color.YELLOW); // Background color

JLabel titleLabel = new JLabel("Ticket Management", SwingConstants.CENTER);

titleLabel.setBounds(0, 10, 800, 50);

titleLabel.setFont(new Font("Arial", Font.BOLD, 40));

frame.add(titleLabel);

JTextField idField = createTextField(frame, "ID:", 70);

JTextField nameField = createTextField(frame, "Name:", 130);

JTextField ageField = createTextField(frame, "Age:", 190);

JTextField typeField = createTextField(frame, "Type:", 250);

// Buttons for CRUD operations

createButton(frame, "Add", 350, e -> {

addData(new String[]{idField.getText(), nameField.getText(), ageField.getText(), typeField.getText()});

});

createButton(frame, "View", 420, e -> viewData());

createButton(frame, "Update", 490, e -> updateData(new String[]{idField.getText(), nameField.getText(), ageField.getText(), typeField.getText()}));

createButton(frame, "Delete", 560, e -> deleteData(idField.getText()));

createButton(frame, "Back to Menu", 200, e -> frame.dispose());

createButton(frame, "Exit", 70, e -> System.exit(0));

frame.setVisible(true);

}

private JTextField createTextField(JFrame frame, String labelText, int y) {

JLabel label = new JLabel(labelText);

label.setBounds(50, y, 150, 40);

label.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(label);

JTextField textField = new JTextField();

textField.setBounds(200, y, 250, 40);

textField.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(textField);

return textField;

}

private void createButton(JFrame frame, String text, int y, ActionListener actionListener) {

JButton button = new JButton(text);

button.setBounds(500, y, 250, 50);

button.setFont(new Font("Arial", Font.PLAIN, 30));

button.setBackground(Color.GREEN);

button.setForeground(Color.WHITE);

button.addActionListener(actionListener);

frame.add(button);

}

@Override

public void addData(String[] fields) {

if (!validateFields(fields)) { // If validation fails, stop execution

return; // Exit the method early

}

data.add(fields); // Add the ticket only if validation passes

JOptionPane.showMessageDialog(null, "Ticket Added Successfully.");

}

@Override

public void viewData() {

if (data.isEmpty()) {

JOptionPane.showMessageDialog(null, "No Data Available to View.");

return;

}

StringBuilder viewData = new StringBuilder("Ticket Data:\n");

for (String[] record : data) {

viewData.append("ID: ").append(record[0])

.append(", Name: ").append(record[1])

.append(", Age: ").append(record[2])

.append(", Type: ").append(record[3]).append("\n");

}

JOptionPane.showMessageDialog(null, viewData.toString());

}

@Override

public void updateData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

for (String[] record : data) {

if (record[0].equals(fields[0])) { // Match ID

record[1] = fields[1];

record[2] = fields[2];

record[3] = fields[3];

JOptionPane.showMessageDialog(null, "Ticket Updated Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Ticket Not Found.");

}

}

@Override

public void deleteData(String id) {

if (id == null || id.trim().isEmpty()) { // Validate ID

JOptionPane.showMessageDialog(null, "Please Enter an ID to Delete.");

return;

}

for (String[] record : data) {

if (record[0].equals(id)) { // Match ID

data.remove(record);

JOptionPane.showMessageDialog(null, "Ticket Deleted Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Ticket Not Found.");

}

/\*\*

\* Validates if all required fields are filled and performs age validation.

\*

\* @param fields The array of fields to validate.

\* The "age" field should be at a specific position (e.g., index 1).

\* @return true if all fields are valid, false otherwise.

\*/

private boolean validateFields(String[] fields) {

// Define field names for better error messages

String[] fieldNames = {"ID", "Name", "Age", "Type"};

// Loop through each field to check if they are not null or empty

for (int i = 0; i < fields.length; i++) {

if (fields[i] == null || fields[i].trim().isEmpty()) { // Check for null or empty fields

JOptionPane.showMessageDialog(null, "Field '" + fieldNames[i] + "' must be filled.");

return false;

}

}

// Additional validation for the "Age" field (must be numeric and positive)

try {

int age = Integer.parseInt(fields[2].trim()); // Age is in the 3rd field (index 2)

if (age <= 0) {

JOptionPane.showMessageDialog(null, "Age must be a positive integer.");

return false;

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(null, "Age must be a valid number.");

return false;

}

return true; // All fields are valid

}

}

**//EMPLOYEE MANAGEMENT SYSTEM**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class EmployeeManagement extends ZooModule {

@Override

public void showModule() {

JFrame frame = new JFrame("Employee Management");

frame.setSize(800, 600);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

frame.getContentPane().setBackground(Color.PINK); // Background color

JLabel titleLabel = new JLabel("Employee Management", SwingConstants.CENTER);

titleLabel.setBounds(0, 10, 800, 50);

titleLabel.setFont(new Font("Arial", Font.BOLD, 40));

frame.add(titleLabel);

JTextField idField = createTextField(frame, "ID:", 70);

JTextField nameField = createTextField(frame, "Name:", 130);

JTextField roleField = createTextField(frame, "Role:", 190);

JTextField shiftField = createTextField(frame, "Shift:", 250);

// Buttons for CRUD operations

createButton(frame, "Add", 350, e -> {

addData(new String[]{idField.getText(), nameField.getText(), roleField.getText(), shiftField.getText()});

});

createButton(frame, "View", 420, e -> viewData());

createButton(frame, "Update", 490, e -> updateData(new String[]{idField.getText(), nameField.getText(), roleField.getText(), shiftField.getText()}));

createButton(frame, "Delete", 560, e -> deleteData(idField.getText()));

createButton(frame, "Back to Menu", 200, e -> frame.dispose());

createButton(frame, "Exit", 70, e -> System.exit(0));

frame.setVisible(true);

}

private JTextField createTextField(JFrame frame, String labelText, int y) {

JLabel label = new JLabel(labelText);

label.setBounds(50, y, 150, 40);

label.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(label);

JTextField textField = new JTextField();

textField.setBounds(200, y, 250, 40);

textField.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(textField);

return textField;

}

private void createButton(JFrame frame, String text, int y, ActionListener actionListener) {

JButton button = new JButton(text);

button.setBounds(500, y, 250, 50);

button.setFont(new Font("Arial", Font.PLAIN, 30));

button.setBackground(Color.GREEN);

button.setForeground(Color.WHITE);

button.addActionListener(actionListener);

frame.add(button);

}

@Override

public void addData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

data.add(fields);

JOptionPane.showMessageDialog(null, "Employee Added Successfully.");

}

}

@Override

public void viewData() {

if (data.isEmpty()) {

JOptionPane.showMessageDialog(null, "No Employee Data Available to View.");

return;

}

StringBuilder viewData = new StringBuilder("Employee Data:\n");

for (String[] record : data) {

viewData.append("ID: ").append(record[0])

.append(", Name: ").append(record[1])

.append(", Role: ").append(record[2])

.append(", Shift: ").append(record[3]).append("\n");

}

JOptionPane.showMessageDialog(null, viewData.toString());

}

@Override

public void updateData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

for (String[] record : data) {

if (record[0].equals(fields[0])) { // Match ID

record[1] = fields[1];

record[2] = fields[2];

record[3] = fields[3];

JOptionPane.showMessageDialog(null, "Employee Updated Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Employee Not Found.");

}

}

@Override

public void deleteData(String id) {

if (id == null || id.trim().isEmpty()) { // Validate ID

JOptionPane.showMessageDialog(null, "Please Enter an ID to Delete.");

return;

}

for (String[] record : data) {

if (record[0].equals(id)) { // Match ID

data.remove(record);

JOptionPane.showMessageDialog(null, "Employee Deleted Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Employee Not Found.");

}

/\*\*

\* Validates if all required fields are filled.

\*

\* @param fields The array of fields to validate.

\* @return true if all fields are valid, false otherwise.

\*/

private boolean validateFields(String[] fields) {

for (String field : fields) {

if (field == null || field.trim().isEmpty()) { // Check for null or empty fields

JOptionPane.showMessageDialog(null, "All fields must be filled.");

return false;

}

}

return true;

}

}

**//VISITER MANAGEMENT SYSTEM**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class VisiterManagement extends ZooModule{

@Override

public void showModule() {

JFrame frame = new JFrame("Visiter Management");

frame.setSize(800, 600);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

frame.getContentPane().setBackground(Color.GRAY); // Background color

JLabel titleLabel = new JLabel("Visiter Management", SwingConstants.CENTER);

titleLabel.setBounds(0, 10, 800, 50);

titleLabel.setFont(new Font("Arial", Font.BOLD, 40));

frame.add(titleLabel);

JTextField idField = createTextField(frame, "ID:", 70);

JTextField nameField = createTextField(frame, "Name:", 130);

JTextField typeField = createTextField(frame, "Type:", 190);

JTextField numberField = createTextField(frame, "Number:", 250);

// Buttons for CRUD operations

createButton(frame, "Add", 350, e -> {

addData(new String[]{idField.getText(), nameField.getText(), typeField.getText(), numberField.getText()});

});

createButton(frame, "View", 420, e -> viewData());

createButton(frame, "Update", 490, e -> updateData(new String[]{idField.getText(), nameField.getText(), typeField.getText(), numberField.getText()}));

createButton(frame, "Delete", 560, e -> deleteData(idField.getText()));

createButton(frame, "Back to Menu", 200, e -> frame.dispose());

createButton(frame, "Exit", 70, e -> System.exit(0));

frame.setVisible(true);

}

private JTextField createTextField(JFrame frame, String labelText, int y) {

JLabel label = new JLabel(labelText);

label.setBounds(50, y, 150, 40);

label.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(label);

JTextField textField = new JTextField();

textField.setBounds(200, y, 250, 40);

textField.setFont(new Font("Arial", Font.PLAIN, 30));

frame.add(textField);

return textField;

}

private void createButton(JFrame frame, String text, int y, ActionListener actionListener) {

JButton button = new JButton(text);

button.setBounds(500, y, 250, 50);

button.setFont(new Font("Arial", Font.PLAIN, 30));

button.setBackground(Color.GREEN);

button.setForeground(Color.WHITE);

button.addActionListener(actionListener);

frame.add(button);

}

@Override

public void addData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

data.add(fields);

JOptionPane.showMessageDialog(null, "Visitor Added Successfully.");

}

}

@Override

public void viewData() {

if (data.isEmpty()) {

JOptionPane.showMessageDialog(null, "No Visitor Data Available to View.");

return;

}

StringBuilder viewData = new StringBuilder("Visitor Data:\n");

for (String[] record : data) {

viewData.append("ID: ").append(record[0])

.append(", Name: ").append(record[1])

.append(", Type: ").append(record[2])

.append(", Number: ").append(record[3]).append("\n");

}

JOptionPane.showMessageDialog(null, viewData.toString());

}

@Override

public void updateData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

for (String[] record : data) {

if (record[0].equals(fields[0])) { // Match ID

record[1] = fields[1];

record[2] = fields[2];

record[3] = fields[3];

JOptionPane.showMessageDialog(null, "Visitor Updated Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Visitor Not Found.");

}

}

@Override

public void deleteData(String id) {

if (id == null || id.trim().isEmpty()) { // Validate ID

JOptionPane.showMessageDialog(null, "Please Enter an ID to Delete.");

return;

}

for (String[] record : data) {

if (record[0].equals(id)) { // Match ID

data.remove(record);

JOptionPane.showMessageDialog(null, "Visitor Deleted Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Visitor Not Found.");

}

/\*\*

\* Validates if all required fields are filled.

\*

\* @param fields The array of fields to validate.

\* @return true if all fields are valid, false otherwise.

\*/

private boolean validateFields(String[] fields) {

for (String field : fields) {

if (field == null || field.trim().isEmpty()) { // Check for null or empty fields

JOptionPane.showMessageDialog(null, "All fields must be filled.");

return false;

}

}

return true;

}

}

**//VETERINARY MANAGEMENT SYSTEM**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class VeterinaryManagement extends ZooModule {

@Override

public void showModule() {

JFrame frame = new JFrame("Veterinary Management");

frame.setSize(800, 600);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

frame.getContentPane().setBackground(Color.CYAN); // Background color

JLabel titleLabel = new JLabel("Veterinary Management", SwingConstants.CENTER);

titleLabel.setBounds(0, 10, 800, 50);

titleLabel.setFont(new Font("Arial", Font.BOLD, 40));

frame.add(titleLabel);

JTextField medicineField = createTextField(frame, "Medicine:", 70);

JTextField frequencyField = createTextField(frame, "Frequency:", 130);

JTextField durationField = createTextField(frame, "Duration:", 190);

JTextField dosageField = createTextField(frame, "Dosage:", 250);

// Buttons for CRUD operations

createButton(frame, "Add", 350, e -> {

addData(new String[]{medicineField.getText(), frequencyField.getText(), durationField.getText(), dosageField.getText()});

});

createButton(frame, "View", 420, e -> viewData());

createButton(frame, "Update", 490, e -> updateData(new String[]{medicineField.getText(), frequencyField.getText(), durationField.getText(), dosageField.getText()}));

createButton(frame, "Delete", 560, e -> deleteData(medicineField.getText()));

createButton(frame, "Back to Menu", 200, e -> frame.dispose());

createButton(frame, "Exit", 70, e -> System.exit(0));

frame.setVisible(true);

}

private JTextField createTextField(JFrame frame, String labelText, int y) {

JLabel label = new JLabel(labelText);

label.setBounds(50, y, 150, 40);

label.setFont(new Font("Arial", Font.PLAIN, 20));

frame.add(label);

JTextField textField = new JTextField();

textField.setBounds(200, y, 250, 40);

textField.setFont(new Font("Arial", Font.PLAIN, 20));

frame.add(textField);

return textField;

}

private void createButton(JFrame frame, String text, int y, ActionListener actionListener) {

JButton button = new JButton(text);

button.setBounds(500, y, 250, 50);

button.setFont(new Font("Arial", Font.PLAIN, 20));

button.setBackground(Color.GREEN);

button.setForeground(Color.WHITE);

button.addActionListener(actionListener);

frame.add(button);

}

@Override

public void addData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

data.add(fields);

JOptionPane.showMessageDialog(null, "Veterinary Record Added Successfully.");

}

}

@Override

public void viewData() {

if (data.isEmpty()) {

JOptionPane.showMessageDialog(null, "No Veterinary Data Available to View.");

return;

}

StringBuilder viewData = new StringBuilder("Veterinary Data:\n");

for (String[] record : data) {

viewData.append("Medicine: ").append(record[0])

.append(", Frequency: ").append(record[1])

.append(", Duration: ").append(record[2])

.append(", Dosage: ").append(record[3]).append("\n");

}

JOptionPane.showMessageDialog(null, viewData.toString());

}

@Override

public void updateData(String[] fields) {

if (validateFields(fields)) { // Validate all fields

for (String[] record : data) {

if (record[0].equals(fields[0])) { // Match Medicine name

record[1] = fields[1];

record[2] = fields[2];

record[3] = fields[3];

JOptionPane.showMessageDialog(null, "Veterinary Record Updated Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Veterinary Record Not Found.");

}

}

@Override

public void deleteData(String id) {

if (id == null || id.trim().isEmpty()) { // Validate Medicine name

JOptionPane.showMessageDialog(null, "Please Enter a Medicine Name to Delete.");

return;

}

for (String[] record : data) {

if (record[0].equals(id)) { // Match Medicine name

data.remove(record);

JOptionPane.showMessageDialog(null, "Veterinary Record Deleted Successfully.");

return;

}

}

JOptionPane.showMessageDialog(null, "Veterinary Record Not Found.");

}

/\*\*

\* Validates if all required fields are filled.

\*

\* @param fields The array of fields to validate.

\* @return true if all fields are valid, false otherwise.

\*/

private boolean validateFields(String[] fields) {

String[] fieldNames = {"Medicine", "Frequency", "Duration", "Dosage"}; // Field labels for error messages

for (int i = 0; i < fields.length; i++) {

if (fields[i] == null || fields[i].trim().isEmpty()) { // Check for null or empty fields

JOptionPane.showMessageDialog(null, "Field '" + fieldNames[i] + "' must be filled.");

return false;

}

}

return true;

}

}

**//ZOO MANAGEMENT SYSTEM MAIN CLASS**

package zoomodule;

import javax.swing.\*;

import java.awt.\*;

public class ZooManagementSystem {

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> showLoginFrame());

}

public static void showLoginFrame() {

JFrame loginFrame = new JFrame("Zoo Management System - Login");

loginFrame.setSize(1000, 400);

loginFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

loginFrame.setLayout(null);

loginFrame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

loginFrame.getContentPane().setBackground(new Color(144, 238, 144)); // Background color

// Title Label

JLabel titleLabel = new JLabel("ZOO MANAGEMENT SYSTEM", JLabel.CENTER);

titleLabel.setBounds(0, 20, loginFrame.getWidth(), 50); // Positioning at the top center

titleLabel.setFont(new Font("Verdana", Font.BOLD, 40)); // Large bold font

titleLabel.setForeground(new Color(0, 100, 0)); // Title text color

loginFrame.add(titleLabel);

// Adding the second image

ImageIcon logoIcon2 = new ImageIcon("images.jpeg"); // Path to the second image

JLabel logoLabel2 = new JLabel(logoIcon2);

logoLabel2.setBounds(500, 400, logoIcon2.getIconWidth(), logoIcon2.getIconHeight()); // Set position and size for second image

loginFrame.add(logoLabel2);

// Adding the third image

ImageIcon logoIcon3 = new ImageIcon("images.jpeg"); // Path to the second image

JLabel logoLabel3 = new JLabel(logoIcon3);

logoLabel3.setBounds(700, 400, logoIcon3.getIconWidth(), logoIcon3.getIconHeight()); // Set position and size for second image

loginFrame.add(logoLabel3);

// Adding the fourth image

ImageIcon logoIcon4 = new ImageIcon("images.jpeg"); // Path to the second image

JLabel logoLabel4 = new JLabel(logoIcon4);

logoLabel4.setBounds(900, 400, logoIcon4.getIconWidth(), logoIcon4.getIconHeight()); // Set position and size for second image

loginFrame.add(logoLabel4);

// Adding the fifth image

ImageIcon logoIcon5 = new ImageIcon("images.jpeg"); // Path to the second image

JLabel logoLabel5 = new JLabel(logoIcon5);

logoLabel5.setBounds(1100, 400, logoIcon5.getIconWidth(), logoIcon5.getIconHeight()); // Set position and size for second image

loginFrame.add(logoLabel5);

// Adding the sixth image

ImageIcon logoIcon6 = new ImageIcon("images.jpeg"); // Path to the second image

JLabel logoLabel6 = new JLabel(logoIcon6);

logoLabel6.setBounds(270, 400, logoIcon6.getIconWidth(), logoIcon6.getIconHeight()); // Set position and size for second image

loginFrame.add(logoLabel6);

// Adding the seventh image

ImageIcon logoIcon7 = new ImageIcon("images.jpeg"); // Path to the second image

JLabel logoLabel7 = new JLabel(logoIcon3);

logoLabel7.setBounds(50, 400, logoIcon7.getIconWidth(), logoIcon7.getIconHeight()); // Set position and size for second image

loginFrame.add(logoLabel7);

// Labels and TextFields for login

JLabel userLabel = new JLabel("Username:");

userLabel.setBounds(250, 100, 150, 40);

userLabel.setFont(new Font("Arial", Font.PLAIN, 30));

loginFrame.add(userLabel);

JTextField userText = new JTextField();

userText.setBounds(500, 100, 250, 40);

userText.setFont(new Font("Arial", Font.PLAIN, 30));

loginFrame.add(userText);

JLabel passLabel = new JLabel("Password:");

passLabel.setBounds(250, 170, 150, 40);

passLabel.setFont(new Font("Arial", Font.PLAIN, 30));

loginFrame.add(passLabel);

JPasswordField passText = new JPasswordField();

passText.setBounds(500, 170, 250, 40);

passText.setFont(new Font("Arial", Font.PLAIN, 30));

loginFrame.add(passText);

// Login Button

JButton loginButton = new JButton("Login");

loginButton.setBounds(450, 250, 150, 50);

loginButton.setFont(new Font("Arial", Font.PLAIN, 30));

loginButton.setBackground(Color.GREEN);

loginButton.setForeground(Color.WHITE);

loginButton.addActionListener(e -> {

String username = userText.getText();

String password = new String(passText.getPassword());

try {

if ("admin".equals(username) && "1234".equals(password)) {

JOptionPane.showMessageDialog(loginFrame, "Login Successful!");

loginFrame.dispose();

showModulesFrame();

} else {

throw new Exception("Invalid Username or Password.");

}

} catch (Exception ex) {

JOptionPane.showMessageDialog(loginFrame, ex.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);

}

});

loginFrame.add(loginButton);

loginFrame.setVisible(true);

}

public static void showModulesFrame() {

JFrame modulesFrame = new JFrame("Zoo Management System - Modules");

modulesFrame.setSize(600, 400);

modulesFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

modulesFrame.setLayout(new GridLayout(3, 2, 10, 10));

modulesFrame.setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full-screen mode

modulesFrame.getContentPane().setBackground(Color.BLACK); // Background color

// Buttons for each module

JButton animalManagementButton = new JButton("Animal Management");

animalManagementButton.setFont(new Font("Georgia", Font.PLAIN, 30));

animalManagementButton.setBackground(Color.ORANGE);

animalManagementButton.setForeground(Color.WHITE);

animalManagementButton.addActionListener(e -> {

AnimalManagement animalManagement = new AnimalManagement();

animalManagement.showModule();

});

JButton inventoryManagementButton = new JButton("Inventory Management");

inventoryManagementButton.setFont(new Font("Georgia", Font.PLAIN, 30));

inventoryManagementButton.setBackground(Color.CYAN);

inventoryManagementButton.setForeground(Color.WHITE);

inventoryManagementButton.addActionListener(e -> {

InventoryManagement inventoryManagement = new InventoryManagement();

inventoryManagement.showModule();

});

JButton ticketManagementButton = new JButton("Ticket Management");

ticketManagementButton.setFont(new Font("Georgia", Font.PLAIN, 30));

ticketManagementButton.setBackground(Color.MAGENTA);

ticketManagementButton.setForeground(Color.WHITE);

ticketManagementButton.addActionListener(e -> {

TicketManagement ticketManagement = new TicketManagement();

ticketManagement.showModule();

});

JButton employeeManagementButton = new JButton("Employee Management");

employeeManagementButton.setFont(new Font("Georgia", Font.PLAIN, 30));

employeeManagementButton.setBackground(Color.PINK);

employeeManagementButton.setForeground(Color.WHITE);

employeeManagementButton.addActionListener(e -> {

EmployeeManagement employeeManagement = new EmployeeManagement();

employeeManagement.showModule();

});

JButton visiterManagementButton = new JButton("Visiter Management");

visiterManagementButton.setFont(new Font("Georgia", Font.PLAIN, 30));

visiterManagementButton.setBackground(Color.BLUE);

visiterManagementButton.setForeground(Color.WHITE);

visiterManagementButton.addActionListener(e -> {

VisiterManagement visiterManagement = new VisiterManagement();

visiterManagement.showModule();

});

JButton veterinaryManagementButton = new JButton("Veterinary Management");

veterinaryManagementButton.setFont(new Font("Georgia", Font.PLAIN, 30));

veterinaryManagementButton.setBackground(Color.GRAY);

veterinaryManagementButton.setForeground(Color.WHITE);

veterinaryManagementButton.addActionListener(e -> {

VeterinaryManagement veterinaryManagement = new VeterinaryManagement();

veterinaryManagement.showModule();

});

// More module buttons (Ticket, Employee, Visitor, etc.) can be added here...

// Add buttons to frame

modulesFrame.add(animalManagementButton);

modulesFrame.add(inventoryManagementButton);

modulesFrame.add(ticketManagementButton);

modulesFrame.add(employeeManagementButton);

modulesFrame.add(visiterManagementButton);

modulesFrame.add(veterinaryManagementButton);

modulesFrame.setVisible(true);

}

}