```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;
import java.time.LocalDate;

public class BloodDonationManagementSystemGUI extends JFrame {

    static Connection connection;

    // UI Components
    JTextField donorIdField, donorNameField, donorContactField, donorBloodTypeField,
donorDobField, donorLastDonationField;
    JTextField inventoryBloodTypeField, inventoryQuantityField, inventoryExpirationDateField;
    JTable donorTable, donationTable, inventoryTable, appointmentTable;
    DefaultTableModel donorTableModel, donationTableModel, inventoryTableModel,
appointmentTableModel;

    public BloodDonationManagementSystemGUI() {
        try {
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/blood_donation",
"root", "ashwin");
            System.out.println("Connected to the database successfully!");

            // Create and configure the window
            setTitle("Blood Donation Management System");
            setSize(800, 600);
            setDefaultCloseOperation(EXIT_ON_CLOSE);
            setLayout(new BorderLayout());
```

```
// Create main panel
JPanel mainPanel = new JPanel();
mainPanel.setLayout(new GridLayout(2, 1));

// Top panel for forms
JPanel formPanel = new JPanel();
formPanel.setLayout(new GridLayout(2, 1));

// Donor input form
JPanel donorFormPanel = new JPanel(new GridLayout(7, 2));
donorFormPanel.setBorder(BorderFactory.createTitledBorder("Add Donor"));

donorIdField = new JTextField();
donorNameField = new JTextField();
donorContactField = new JTextField();
donorBloodTypeField = new JTextField();
donorDobField = new JTextField();
donorLastDonationField = new JTextField();
JButton addDonorButton = new JButton("Add Donor");

donorFormPanel.add(new JLabel("Donor ID:"));
donorFormPanel.add(donorIdField);
donorFormPanel.add(new JLabel("Name:"));
donorFormPanel.add(donorNameField);
donorFormPanel.add(new JLabel("Contact:"));
donorFormPanel.add(donorContactField);
donorFormPanel.add(new JLabel("Blood Type:"));
donorFormPanel.add(donorBloodTypeField);
donorFormPanel.add(new JLabel("Date of Birth (YYYY-MM-DD):"));
donorFormPanel.add(donorDobField);
donorFormPanel.add(new JLabel("Last Donation Date (YYYY-MM-DD):"));
```

```
        donorFormPanel.add(donorLastDonationField);
        donorFormPanel.add(addDonorButton);

        formPanel.add(donorFormPanel);

        // Inventory input form
        JPanel inventoryFormPanel = new JPanel(new GridLayout(4, 2));
        inventoryFormPanel.setBorder(BorderFactory.createTitledBorder("Update Inventory"));

        inventoryBloodTypeField = new JTextField();
        inventoryQuantityField = new JTextField();
        inventoryExpirationDateField = new JTextField();
        JButton updateInventoryButton = new JButton("Update Inventory");

        inventoryFormPanel.add(new JLabel("Blood Type:"));
        inventoryFormPanel.add(inventoryBloodTypeField);
        inventoryFormPanel.add(new JLabel("Quantity (in ml):"));
        inventoryFormPanel.add(inventoryQuantityField);
        inventoryFormPanel.add(new JLabel("Expiration Date (YYYY-MM-DD):"));
        inventoryFormPanel.add(inventoryExpirationDateField);
        inventoryFormPanel.add(updateInventoryButton);

        formPanel.add(inventoryFormPanel);

        // Bottom panel for displaying tables
        JPanel tablePanel = new JPanel();
        tablePanel.setLayout(new GridLayout(2, 1));

        // Donor table
        donorTableModel = new DefaultTableModel(new String[]{"ID", "Name", "Blood Type", "Last Donation"}, 0);
```

```
        donorTable = new JTable(donorTableModel);
        JScrollPane donorScrollPane = new JScrollPane(donorTable);

        // Inventory table
        inventoryTableModel = new DefaultTableModel(new String[]{"Blood Type", "Quantity",
"Expiration Date"}, 0);
        inventoryTable = new JTable(inventoryTableModel);
        JScrollPane inventoryScrollPane = new JScrollPane(inventoryTable);

        tablePanel.add(donorScrollPane);
        tablePanel.add(inventoryScrollPane);

        // Add the formPanel and tablePanel to the main panel
        mainPanel.add(formPanel);
        mainPanel.add(tablePanel);

        // Add main panel to the frame
        add(mainPanel);

        // Add Donor button event handler
        addDonorButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    addDonor();
                } catch (SQLException ex) {
                    ex.printStackTrace();
                }
            }
        });
```

```
        // Update Inventory button event handler
        updateInventoryButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    updateInventory();
                } catch (SQLException ex) {
                    ex.printStackTrace();
                }
            }
        });

        // Load initial data
        loadDonorData();
        loadInventoryData();

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

// Method to add a new donor
private void addDonor() throws SQLException {
    int id = Integer.parseInt(donorIdField.getText());
    String name = donorNameField.getText();
    String contact = donorContactField.getText();
    String bloodType = donorBloodTypeField.getText();
    LocalDate dateOfBirth = LocalDate.parse(donorDobField.getText());
    LocalDate lastDonationDate = LocalDate.parse(donorLastDonationField.getText());
```

```
        String query = "INSERT INTO donors (id, name, contact, bloodType, dateOfBirth,
lastDonationDate) VALUES (?, ?, ?, ?, ?, ?)";
        PreparedStatement stmt = connection.prepareStatement(query);
        stmt.setInt(1, id);
        stmt.setString(2, name);
        stmt.setString(3, contact);
        stmt.setString(4, bloodType);
        stmt.setDate(5, Date.valueOf(dateOfBirth));
        stmt.setDate(6, Date.valueOf(lastDonationDate));

        stmt.executeUpdate();
        JOptionPane.showMessageDialog(this, "Donor added successfully!");

        loadDonorData();  // Reload donor data into table
    }

    // Method to update the inventory
    private void updateInventory() throws SQLException {
        String bloodType = inventoryBloodTypeField.getText();
        int quantity = Integer.parseInt(inventoryQuantityField.getText());
        LocalDate expirationDate = LocalDate.parse(inventoryExpirationDateField.getText());

        String query = "INSERT INTO inventory (bloodType, quantity, expirationDate) VALUES (?, ?, ?)";
        PreparedStatement stmt = connection.prepareStatement(query);
        stmt.setString(1, bloodType);
        stmt.setInt(2, quantity);
        stmt.setDate(3, Date.valueOf(expirationDate));

        stmt.executeUpdate();
        JOptionPane.showMessageDialog(this, "Inventory updated successfully!");
```

```
    loadInventoryData();  // Reload inventory data into table
}

// Method to load donor data from the database into the table
private void loadDonorData() throws SQLException {
    donorTableModel.setRowCount(0);  // Clear existing rows

    String query = "SELECT * FROM donors";
    Statement stmt = connection.createStatement();
    ResultSet rs = stmt.executeQuery(query);

    while (rs.next()) {
        donorTableModel.addRow(new Object[]{
                rs.getInt("id"),
                rs.getString("name"),
                rs.getString("bloodType"),
                rs.getDate("lastDonationDate")
        });
    }
}

// Method to load inventory data from the database into the table
private void loadInventoryData() throws SQLException {
    inventoryTableModel.setRowCount(0);  // Clear existing rows

    String query = "SELECT * FROM inventory";
    Statement stmt = connection.createStatement();
    ResultSet rs = stmt.executeQuery(query);

    while (rs.next()) {
        inventoryTableModel.addRow(new Object[]{
```

```
                rs.getString("bloodType"),
                rs.getInt("quantity"),
                rs.getDate("expirationDate")
            });
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new BloodDonationManagementSystemGUI().setVisible(true);
            }
        });
    }
}
```