CODING:

Front Hand:

```
package minipro;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
import java.util.Vector;
import java.text.SimpleDateFormat;

public class DonationTrackingSystem extends JFrame {
 // --- Configure DB connection here ---
 private static final String DB_URL =
"jdbc:mysql://localhost:3306/donation_db?useSSL=false&serverTimezone=UTC";
 private static final String DB_USER = "root";
 private static final String DB_PASS = "2006";
 // -----------------------------------

 // Swing components
 private JTable donorsTable;
 private DefaultTableModel donorsModel;
 private JTable donationsTable;
 private DefaultTableModel donationsModel;

 // input fields for donor
 private JTextField txtName, txtEmail, txtPhone;
 private JTextArea txtAddress;

 // donation fields
 private JComboBox<String> comboDonor;
 private JTextField txtAmount;
 private JTextField txtDate; // format: yyyy-MM-dd
 private JTextField txtPayment;
 private JTextField txtRemarks;

 public DonationTrackingSystem() {
    setTitle("Donation Tracking System");
    setSize(1000, 650);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);

    JTabbedPane tabs = new JTabbedPane();

    tabs.addTab("Manage Donors", createDonorsPanel());
    tabs.addTab("Record Donation", createDonationPanel());
    tabs.addTab("View Donations", createViewDonationsPanel());
    tabs.addTab("Search", createSearchPanel());

    add(tabs);
    loadDonorsIntoTable();
    loadDonorsIntoCombo();
    loadDonationsIntoTable();
  }

  // ----- DB utility -----
  private Connection getConnection() throws SQLException {
    return DriverManager.getConnection(DB_URL, DB_USER, DB_PASS);
  }

  // ----- Donors Panel -----
  private JPanel createDonorsPanel() {
    JPanel p = new JPanel(new BorderLayout());

    // Top form
    JPanel form = new JPanel(new GridBagLayout());
    GridBagConstraints c = new GridBagConstraints();
    c.insets = new Insets(5,5,5,5);
    c.anchor = GridBagConstraints.WEST;

    c.gridx=0; c.gridy=0; form.add(new JLabel("Name:"), c);
    c.gridx=1; txtName = new JTextField(20); form.add(txtName, c);

    c.gridx=0; c.gridy=1; form.add(new JLabel("Email:"), c);
    c.gridx=1; txtEmail = new JTextField(20); form.add(txtEmail, c);

    c.gridx=0; c.gridy=2; form.add(new JLabel("Phone:"), c);
    c.gridx=1; txtPhone = new JTextField(15); form.add(txtPhone, c);

    c.gridx=0; c.gridy=3; form.add(new JLabel("Address:"), c);
    c.gridx=1; txtAddress = new JTextArea(3,20); form.add(new
JScrollPane(txtAddress), c);

    JButton btnAdd = new JButton("Add Donor");
    JButton btnUpdate = new JButton("Update Selected");
    JButton btnDelete = new JButton("Delete Selected");

    JPanel btns = new JPanel();
    btns.add(btnAdd); btns.add(btnUpdate); btns.add(btnDelete);

    JPanel top = new JPanel(new BorderLayout());
    top.add(form, BorderLayout.CENTER);
    top.add(btns, BorderLayout.SOUTH);

    // Table
    donorsModel = new DefaultTableModel(new
String[]{"ID","Name","Email","Phone","Address","Created At"}, 0) {
        public boolean isCellEditable(int r,int c){return false;}
    };
    donorsTable = new JTable(donorsModel);
    JScrollPane tableScroll = new JScrollPane(donorsTable);

    p.add(top, BorderLayout.NORTH);
    p.add(tableScroll, BorderLayout.CENTER);

    // button actions
    btnAdd.addActionListener(e -> addDonor());
    btnUpdate.addActionListener(e -> updateSelectedDonor());
    btnDelete.addActionListener(e -> deleteSelectedDonor());

    donorsTable.addMouseListener(new MouseAdapter(){
        public void mouseClicked(MouseEvent e){
            int row = donorsTable.getSelectedRow();
            if(row >= 0){
                txtName.setText(donorsModel.getValueAt(row,1).toString());
                txtEmail.setText(donorsModel.getValueAt(row,2).toString());
                txtPhone.setText(donorsModel.getValueAt(row,3).toString());
                txtAddress.setText(donorsModel.getValueAt(row,4).toString());
            }
        }
    });

    return p;
}

// ----- Record Donation Panel -----
private JPanel createDonationPanel() {
    JPanel p = new JPanel(new BorderLayout());

    JPanel form = new JPanel(new GridBagLayout());
    GridBagConstraints c = new GridBagConstraints();
    c.insets = new Insets(6,6,6,6);
    c.anchor = GridBagConstraints.WEST;

    c.gridx=0; c.gridy=0; form.add(new JLabel("Donor:"), c);
    c.gridx=1; comboDonor = new JComboBox<>();
comboDonor.setPreferredSize(new Dimension(300,25)); form.add(comboDonor, c);

    c.gridx=0; c.gridy=1; form.add(new JLabel("Amount:"), c);
    c.gridx=1; txtAmount = new JTextField(15); form.add(txtAmount, c);

    c.gridx=0; c.gridy=2; form.add(new JLabel("Date (yyyy-MM-dd):"), c);
    c.gridx=1; txtDate = new JTextField(12); txtDate.setText(new
SimpleDateFormat("yyyy-MM-dd").format(new java.util.Date())); form.add(txtDate, c);

    c.gridx=0; c.gridy=3; form.add(new JLabel("Payment Mode:"), c);
    c.gridx=1; txtPayment = new JTextField(15); form.add(txtPayment, c);

    c.gridx=0; c.gridy=4; form.add(new JLabel("Remarks:"), c);
    c.gridx=1; txtRemarks = new JTextField(30); form.add(txtRemarks, c);

    JButton btnRecord = new JButton("Record Donation");
    form.add(btnRecord, new GridBagConstraints(){{
      gridx=1; gridy=5; anchor=GridBagConstraints.CENTER; insets=new
Insets(10,0,0,0);
    }});

    p.add(form, BorderLayout.NORTH);

    btnRecord.addActionListener(e -> recordDonation());

    return p;
  }

  // ----- View Donations Panel -----
  private JPanel createViewDonationsPanel() {
    JPanel p = new JPanel(new BorderLayout());

    donationsModel = new DefaultTableModel(new String[]{"Donation
ID","Donor","Amount","Date","Payment Mode","Remarks","Created At"},0){
      public boolean isCellEditable(int r,int c){return false;}
    };
    donationsTable = new JTable(donationsModel);
    JScrollPane sp = new JScrollPane(donationsTable);

    JButton btnRefresh = new JButton("Refresh");
    JButton btnDelete = new JButton("Delete Selected");

    JPanel top = new JPanel(new FlowLayout(FlowLayout.LEFT));
    top.add(btnRefresh); top.add(btnDelete);

    p.add(top, BorderLayout.NORTH);
    p.add(sp, BorderLayout.CENTER);

    btnRefresh.addActionListener(e -> loadDonationsIntoTable());
    btnDelete.addActionListener(e -> deleteSelectedDonation());

    return p;
 }

 // ---- Search Panel -----
 private JPanel createSearchPanel() {
    JPanel p = new JPanel(new BorderLayout());
    JPanel top = new JPanel(new FlowLayout(FlowLayout.LEFT));
    JTextField txtSearch = new JTextField(30);
    JButton btnSearch = new JButton("Search Donor by Name");
    top.add(txtSearch); top.add(btnSearch);

    DefaultTableModel searchModel = new DefaultTableModel(new
String[]{"ID","Name","Email","Phone","Address"},0){
       public boolean isCellEditable(int r,int c){return false;}
    };
    JTable searchTable = new JTable(searchModel);
    JScrollPane sp = new JScrollPane(searchTable);

    p.add(top, BorderLayout.NORTH);
    p.add(sp, BorderLayout.CENTER);

    btnSearch.addActionListener(e -> {
       String q = txtSearch.getText().trim();
       searchModel.setRowCount(0);
       if(q.isEmpty()){ JOptionPane.showMessageDialog(this, "Enter name to search");
return; }
       try (Connection conn = getConnection();
            PreparedStatement ps = conn.prepareStatement("SELECT
donor_id,name,email,phone,address FROM donors WHERE name LIKE ?")) {
           ps.setString(1, "%" + q + "%");
           try (ResultSet rs = ps.executeQuery()) {
              while(rs.next()){
                 searchModel.addRow(new Object[]{
                    rs.getInt("donor_id"),
                    rs.getString("name"),
                        rs.getString("email"),
                        rs.getString("phone"),
                        rs.getString("address")
                    });
                }
            }
        } catch(SQLException ex) {
            showError(ex);
        }
    });

    return p;
}

// ---- CRUD & load methods ----
private void addDonor() {
    String name = txtName.getText().trim();
    if(name.isEmpty()){ JOptionPane.showMessageDialog(this, "Name required");
return; }
    String email = txtEmail.getText().trim();
    String phone = txtPhone.getText().trim();
    String address = txtAddress.getText().trim();

    String sql = "INSERT INTO donors(name,email,phone,address) VALUES(?,?,?,?)";
    try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement(sql)) {
        ps.setString(1,name); ps.setString(2,email); ps.setString(3,phone);
ps.setString(4,address);
        ps.executeUpdate();
        JOptionPane.showMessageDialog(this, "Donor added");
        clearDonorForm();
        loadDonorsIntoTable();
        loadDonorsIntoCombo();
    } catch(SQLException ex){ showError(ex); }
}

private void updateSelectedDonor() {
    int row = donorsTable.getSelectedRow();
    if(row < 0){ JOptionPane.showMessageDialog(this, "Select donor row to update");
return; }
    int id = Integer.parseInt(donorsModel.getValueAt(row,0).toString());
    String name = txtName.getText().trim();
    String email = txtEmail.getText().trim();
    String phone = txtPhone.getText().trim();
    String address = txtAddress.getText().trim();
    try (Connection conn = getConnection();
        PreparedStatement ps = conn.prepareStatement("UPDATE donors SET
name=?,email=?,phone=?,address=? WHERE donor_id=?")) {
       ps.setString(1,name); ps.setString(2,email); ps.setString(3,phone);
ps.setString(4,address); ps.setInt(5,id);
       ps.executeUpdate();
       JOptionPane.showMessageDialog(this, "Updated");
       loadDonorsIntoTable(); loadDonorsIntoCombo();
    } catch(SQLException ex){ showError(ex); }
 }

 private void deleteSelectedDonor() {
    int row = donorsTable.getSelectedRow();
    if(row < 0){ JOptionPane.showMessageDialog(this, "Select donor row to delete");
return; }
    int id = Integer.parseInt(donorsModel.getValueAt(row,0).toString());
    int ans = JOptionPane.showConfirmDialog(this, "Delete donor and their
donations?","Confirm",JOptionPane.YES_NO_OPTION);
    if(ans != JOptionPane.YES_OPTION) return;
    try (Connection conn = getConnection();
        PreparedStatement ps = conn.prepareStatement("DELETE FROM donors
WHERE donor_id=?")) {
       ps.setInt(1,id);
       ps.executeUpdate();
       JOptionPane.showMessageDialog(this, "Deleted");
       loadDonorsIntoTable(); loadDonorsIntoCombo(); loadDonationsIntoTable();
    } catch(SQLException ex){ showError(ex); }
 }

 private void recordDonation() {
    Object selected = comboDonor.getSelectedItem();
    if(selected == null){ JOptionPane.showMessageDialog(this, "No donor selected");
return; }
    String donorItem = selected.toString(); // format: ID - Name
    int donorId = Integer.parseInt(donorItem.split(" - ")[0]);
    String amountS = txtAmount.getText().trim();
    String dateS = txtDate.getText().trim();
    String payment = txtPayment.getText().trim();
    String remarks = txtRemarks.getText().trim();

    if(amountS.isEmpty() || dateS.isEmpty()){ JOptionPane.showMessageDialog(this,
"Amount and Date required"); return; }
    try {
       double amount = Double.parseDouble(amountS);
       String sql = "INSERT INTO
donations(donor_id,amount,donation_date,payment_mode,remarks)
VALUES(?,?,?,?,?)";
```

```
        try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement(sql)) {
            ps.setInt(1, donorId);
            ps.setDouble(2, amount);
            ps.setDate(3, java.sql.Date.valueOf(dateS));
            ps.setString(4, payment);
            ps.setString(5, remarks);
            ps.executeUpdate();
            JOptionPane.showMessageDialog(this, "Donation recorded");
            loadDonationsIntoTable();
        }
    } catch(NumberFormatException nf){
        JOptionPane.showMessageDialog(this, "Invalid amount");
    } catch(SQLException ex){
        showError(ex);
    }
}

private void loadDonorsIntoTable() {
    donorsModel.setRowCount(0);
    try (Connection conn = getConnection();
         Statement st = conn.createStatement();
         ResultSet rs = st.executeQuery("SELECT
donor_id,name,email,phone,address,created_at FROM donors ORDER BY donor_id
DESC")) {
        while(rs.next()){
            donorsModel.addRow(new Object[]{
                rs.getInt("donor_id"),
                rs.getString("name"),
                rs.getString("email"),
                rs.getString("phone"),
                rs.getString("address"),
                rs.getTimestamp("created_at")
            });
        }
    } catch(SQLException ex){ showError(ex); }
}

private void loadDonorsIntoCombo() {
    comboDonor.removeAllItems();
    try (Connection conn = getConnection();
         PreparedStatement ps = conn.prepareStatement("SELECT donor_id,name
FROM donors ORDER BY name")) {
        try(ResultSet rs = ps.executeQuery()){
            while(rs.next()){
                comboDonor.addItem(rs.getInt("donor_id") + " - " + rs.getString("name"));
            }
        }
    } catch(SQLException ex){ showError(ex); }
}

private void loadDonationsIntoTable() {
    donationsModel.setRowCount(0);
    String sql = "SELECT d.donation_id, r.name AS donor_name, d.amount,
d.donation_date, d.payment_mode, d.remarks, d.created_at " +
                 "FROM donations d JOIN donors r ON d.donor_id = r.donor_id ORDER BY
d.donation_id DESC";
    try (Connection conn = getConnection();
         Statement st = conn.createStatement();
         ResultSet rs = st.executeQuery(sql)) {
        while(rs.next()){
            donationsModel.addRow(new Object[]{
                rs.getInt("donation_id"),
                rs.getString("donor_name"),
                rs.getDouble("amount"),
                rs.getDate("donation_date"),
                rs.getString("payment_mode"),
                rs.getString("remarks"),
                rs.getTimestamp("created_at")
            });
        }
    } catch(SQLException ex){ showError(ex); }
}

private void deleteSelectedDonation() {
    int row = donationsTable.getSelectedRow();
    if(row < 0){ JOptionPane.showMessageDialog(this, "Select donation to delete");
return; }
    int id = Integer.parseInt(donationsModel.getValueAt(row,0).toString());
    int ans = JOptionPane.showConfirmDialog(this, "Delete selected
donation?","Confirm",JOptionPane.YES_NO_OPTION);
    if(ans != JOptionPane.YES_OPTION) return;
    try (Connection conn = getConnection();
         PreparedStatement ps = conn.prepareStatement("DELETE FROM donations
WHERE donation_id=?")) {
        ps.setInt(1, id);
        ps.executeUpdate();
        JOptionPane.showMessageDialog(this, "Deleted");
        loadDonationsIntoTable();
    } catch(SQLException ex){ showError(ex); }
}
```

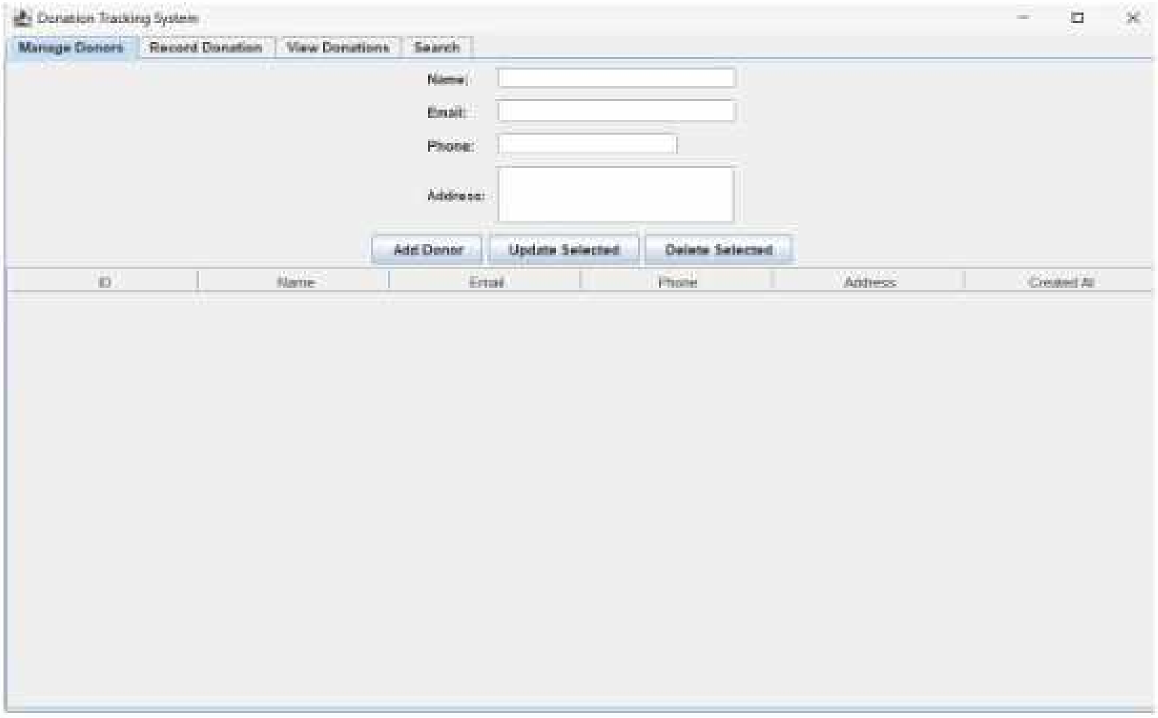
```
 private void clearDonorForm(){
    txtName.setText(""); txtEmail.setText(""); txtPhone.setText("");
txtAddress.setText("");
 }

 private void showError(Exception ex){
    ex.printStackTrace();
    JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage(), "Error",
JOptionPane.ERROR_MESSAGE);
 }

 public static void main(String[] args) {
    // Note: Ensure MySQL Connector/J is on classpath
    SwingUtilities.invokeLater(() -> {
       new DonationTrackingSystem().setVisible(true);
    });
 }
}
```

Back Hand:

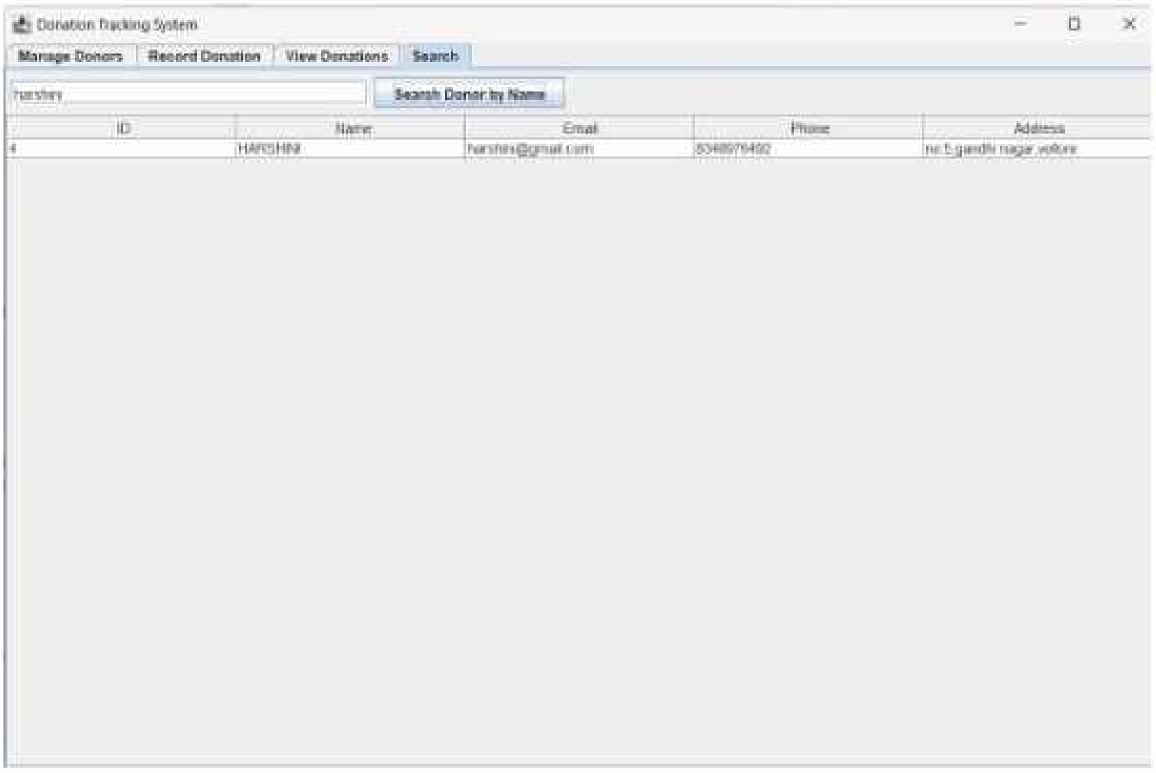
```
-- create database (run once)
CREATE DATABASE IF NOT EXISTS donation_db CHARACTER SET utf8mb4
COLLATE utf8mb4_general_ci;
USE donation_db;

-- donors table
CREATE TABLE IF NOT EXISTS donors (
  donor_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100),
  phone VARCHAR(20),
  address VARCHAR(255),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- donations table
CREATE TABLE IF NOT EXISTS donations (
  donation_id INT AUTO_INCREMENT PRIMARY KEY,
  donor_id INT NOT NULL,
  amount DECIMAL(10,2) NOT NULL,
  donation_date DATE NOT NULL,
  payment_mode VARCHAR(50),
  remarks VARCHAR(255),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (donor_id) REFERENCES donors(donor_id) ON DELETE CASCADE
);
SELECT*FROM DONORS;
```

Screenshots:

Front Hand:

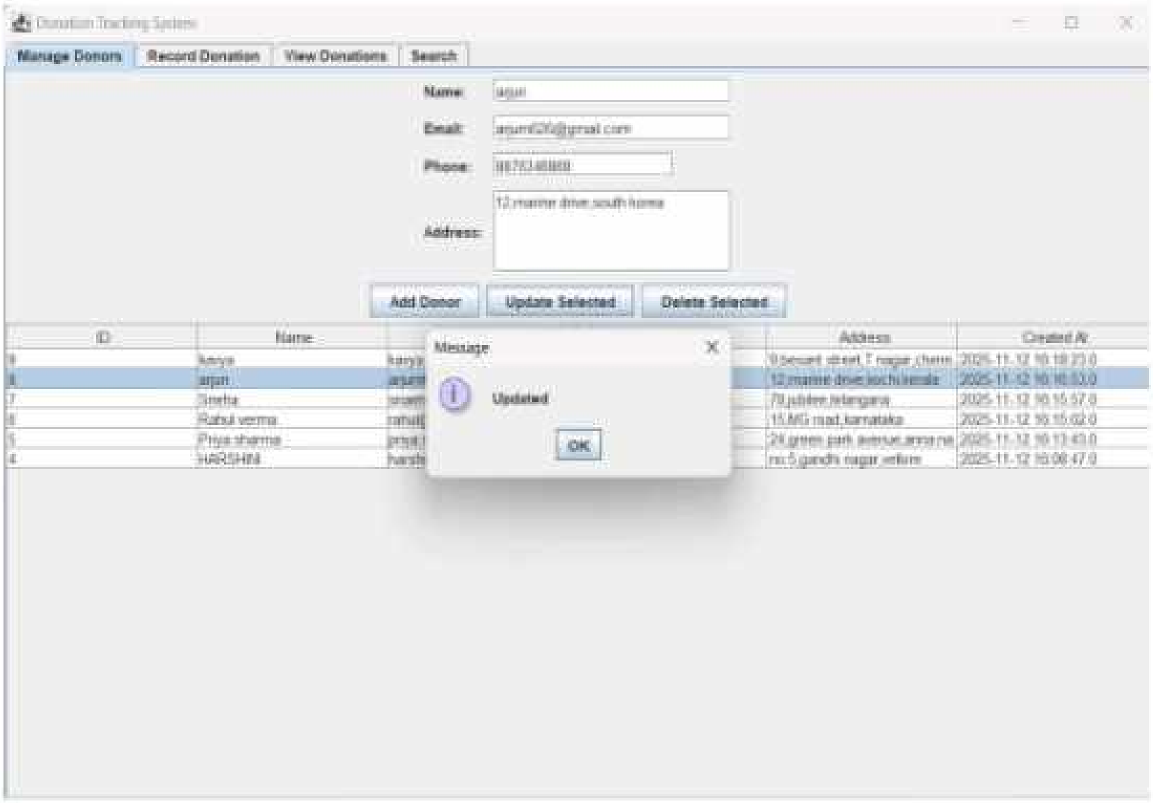


ID	Name	Email	Phone	Address	Created At



harshini

ID	Name	Email	Phone	Address
4	HARSHINI	harshini@gmail.com	8048975492	nr.5,gandhi nagar,vellore


Donation Tracking System

Manage Donors | Record Donation | View Donations | Search

Name: arjun

Email: arjun026@gmail.com

Phone: 9876548888

Address: 12,marine drive,south korea

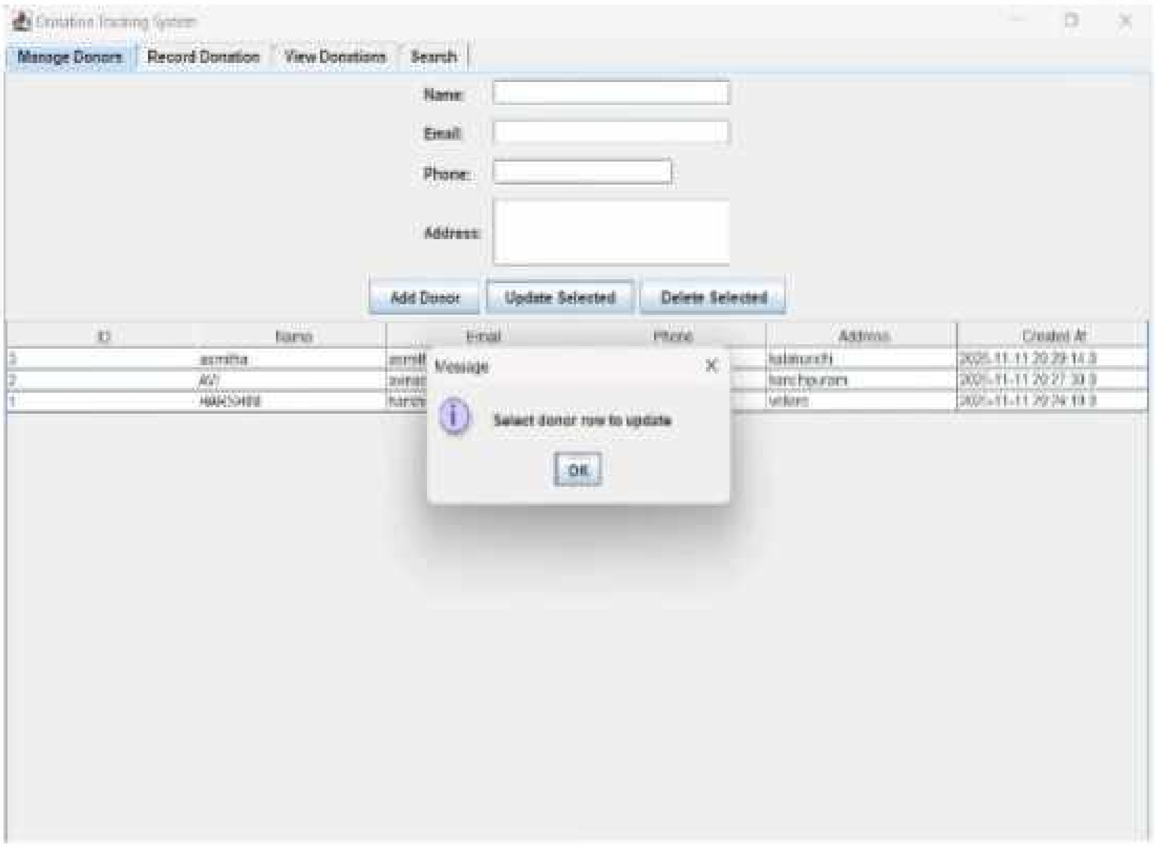
Add Donor | Update Selected | Delete Selected

ID	Name			Address	Created At
9	kavya	kavya		9,besant street,T nagar,chenn	2025-11-12 10:18:23.0
8	arjun	arjun		12,marine drive,kochi,kerala	2025-11-12 10:16:53.0
7	Sneha	sneh		79,jubilee,telangana	2025-11-12 10:15:57.0
6	Rahul verma	rahul		15,MG road,karnataka	2025-11-12 10:15:02.0
5	Priya sharma	priya		24,green park avenue,anna na	2025-11-12 10:13:43.0
4	HARSHINI	harsh		no.5,gandhi nagar,vellore	2025-11-12 10:08:47.0

Message

Updated

OK


Donation Tracking System

Manage Donors | Record Donation | View Donations | Search

Name:

Email:

Phone:

Address:

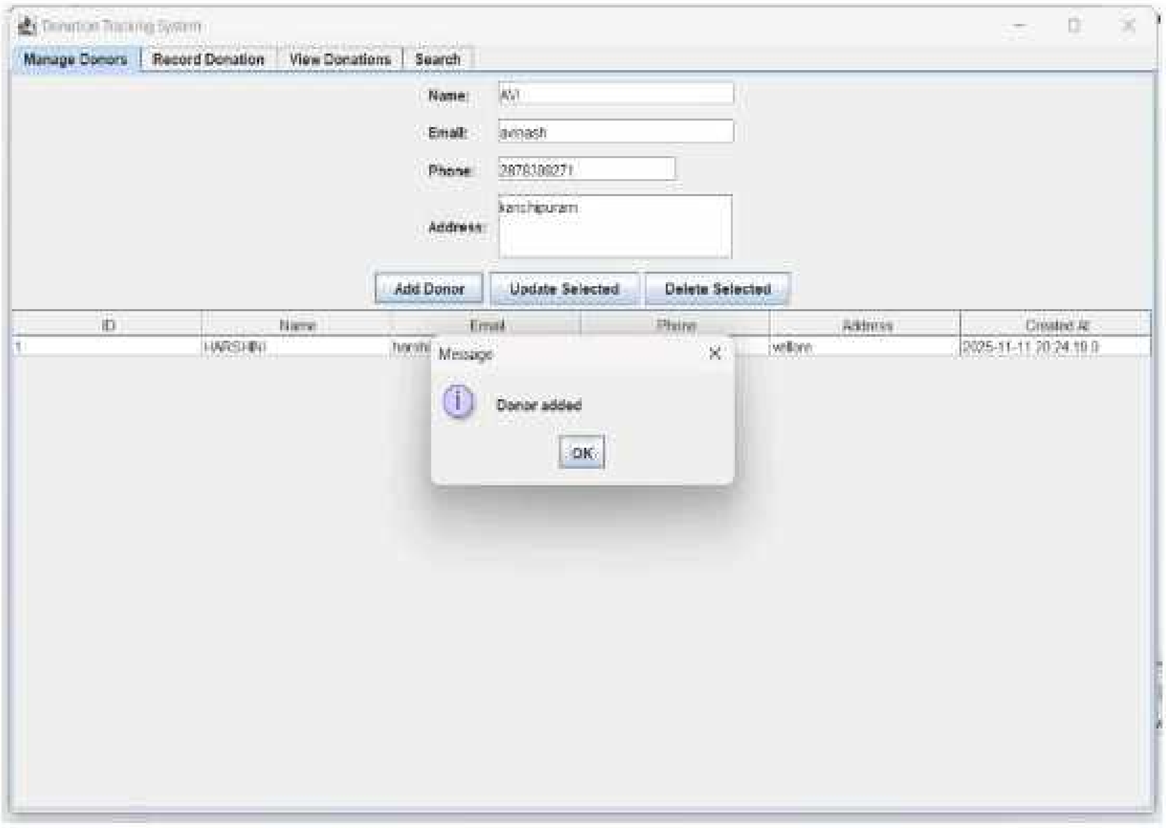
Add Donor | Update Selected | Delete Selected

ID	Name	Email	Phone	Address	Created At
3	asmitha	asmit		kallakurichi	2025-11-11 20:29:14.0
2	AVI	avinas		kanchipuram	2025-11-11 20:27:30.0
1	HARSHINI	harsh		vellore	2025-11-11 20:24:18.0

Message

Select donor row to update

OK

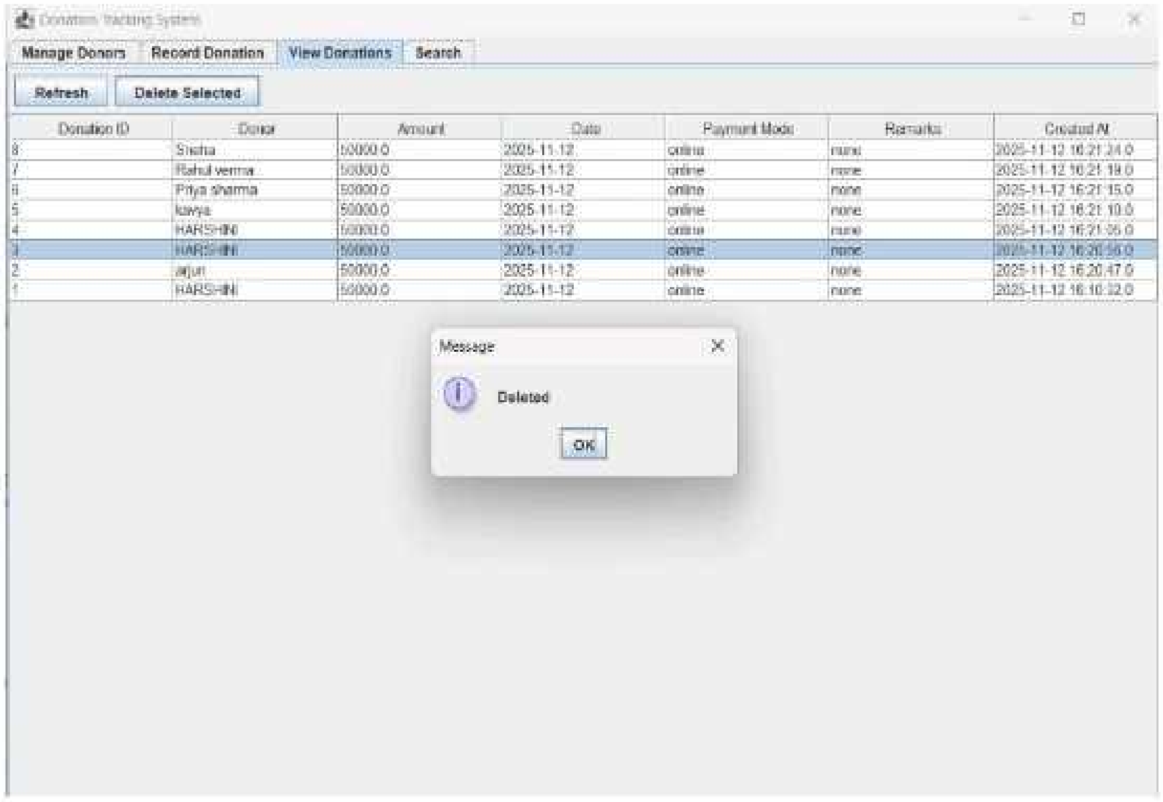


Name: AVI
Email: avinash
Phone: 2878108271
Address: kanchipuram

ID	Name	Email	Phone	Address	Created At
1	HARSHINI	harshi		willom	2025-11-11 20:24:19.0

Message

Donor added

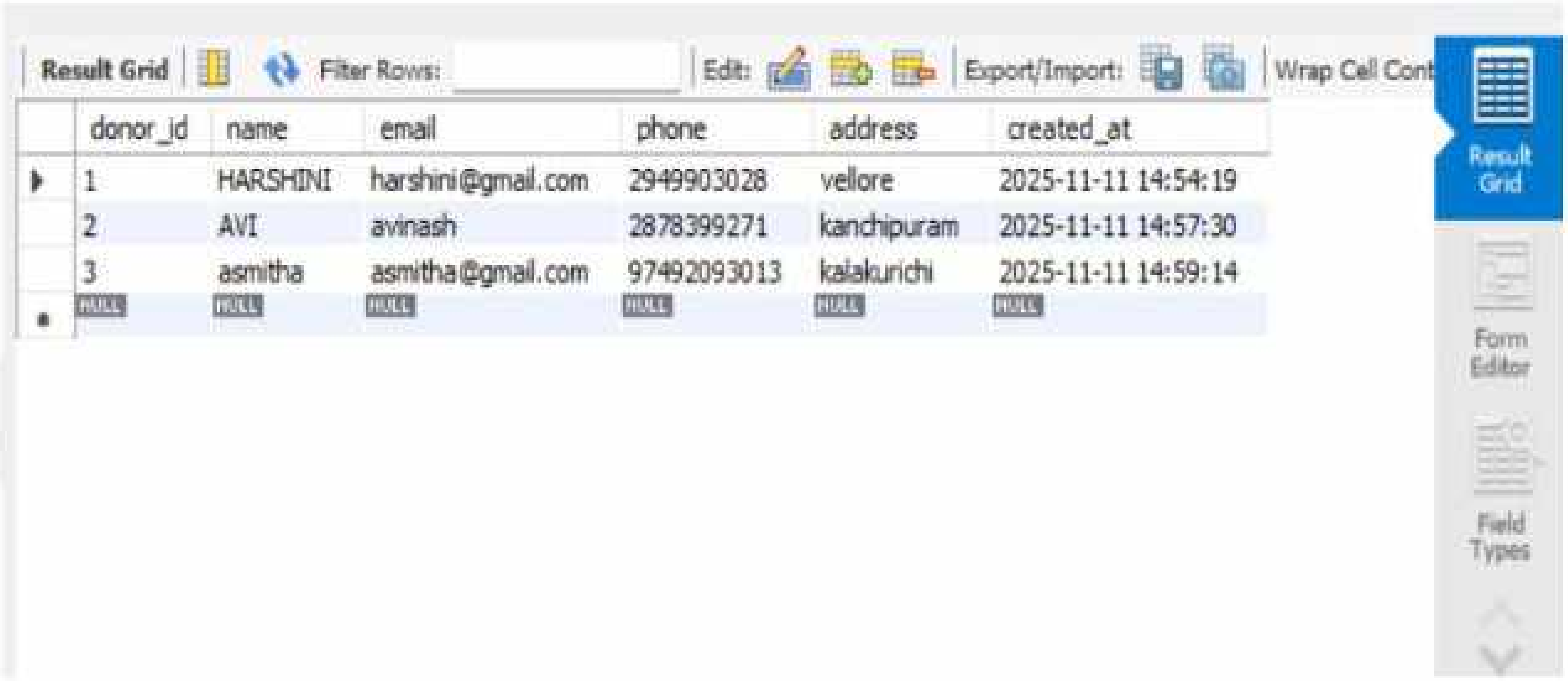


Donation ID	Donor	Amount	Date	Payment Mode	Remarks	Created At
8	Sneha	50000.0	2025-11-12	online	none	2025-11-12 16:21:24.0
7	Rahul verma	50000.0	2025-11-12	online	none	2025-11-12 16:21:19.0
6	Priya sharma	50000.0	2025-11-12	online	none	2025-11-12 16:21:15.0
5	kavya	50000.0	2025-11-12	online	none	2025-11-12 16:21:10.0
4	HARSHINI	50000.0	2025-11-12	online	none	2025-11-12 16:21:06.0
3	HARSHINI	50000.0	2025-11-12	online	none	2025-11-12 16:20:56.0
2	arjun	50000.0	2025-11-12	online	none	2025-11-12 16:20:47.0
1	HARSHINI	50000.0	2025-11-12	online	none	2025-11-12 16:10:32.0

Message

Deleted

Back Hand:



donor_id	name	email	phone	address	created_at
1	HARSHINI	harshini@gmail.com	2949903028	vellore	2025-11-11 14:54:19
2	AVI	avinash	2878399271	kanchipuram	2025-11-11 14:57:30
3	asmitha	asmitha@gmail.com	97492093013	kalakurichi	2025-11-11 14:59:14
NULL	NULL	NULL	NULL	NULL	NULL