import java.awt.\*;

import java.io.\*;

import java.util.\*;

import javax.swing.\*;

class Expense {

    private String category;

    private String description;

    private double amount;

    public Expense(String category, String description, double amount) {

        this.category = category;

        this.description = description;

        this.amount = amount;

    }

    public String getCategory() {

        return category;

    }

    public String getDescription() {

        return description;

    }

    public double getAmount() {

        return amount;

    }

    @Override

    public String toString() {

        return String.format("%s | %s | $%.2f", category, description, amount);

    }

}

public class ExpenseTrackerGUI extends JFrame {

    private DefaultListModel<String> expenseListModel;

    private JList<String> expenseList;

    private JTextField categoryField, descriptionField, amountField;

    private Map<String, Expense> expenseMap;

    private static final String FILE\_NAME = "expenses.txt";

    public ExpenseTrackerGUI() {

        expenseMap = new HashMap<>();

        setTitle("Expense Tracker");

        setSize(400, 500);

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setLayout(new BorderLayout());

        // Expense list

        expenseListModel = new DefaultListModel<>();

        expenseList = new JList<>(expenseListModel);

        JScrollPane scrollPane = new JScrollPane(expenseList);

        add(scrollPane, BorderLayout.CENTER);

        // Input fields

        JPanel inputPanel = new JPanel();

        inputPanel.setLayout(new GridLayout(4, 2));

        inputPanel.add(new JLabel("Category:"));

        categoryField = new JTextField();

        inputPanel.add(categoryField);

        inputPanel.add(new JLabel("Description:"));

        descriptionField = new JTextField();

        inputPanel.add(descriptionField);

        inputPanel.add(new JLabel("Amount:"));

        amountField = new JTextField();

        inputPanel.add(amountField);

        add(inputPanel, BorderLayout.NORTH);

        // Buttons

        JPanel buttonPanel = new JPanel();

        JButton addButton = new JButton("Add Expense");

        addButton.addActionListener(e -> addExpense());

        buttonPanel.add(addButton);

        JButton deleteButton = new JButton("Delete Expense");

        deleteButton.addActionListener(e -> deleteExpense());

        buttonPanel.add(deleteButton);

        add(buttonPanel, BorderLayout.SOUTH);

        loadExpensesFromFile();

        setVisible(true);

    }

    private void addExpense() {

        String category = categoryField.getText();

        String description = descriptionField.getText();

        double amount;

        // Input validation for amount

        try {

            amount = Double.parseDouble(amountField.getText());

        } catch (NumberFormatException e) {

            JOptionPane.showMessageDialog(this, "Please enter a valid amount.");

            return;

        }

        Expense expense = new Expense(category, description, amount);

        expenseListModel.addElement(expense.toString());

        expenseMap.put(expense.toString(), expense);

        // Clear input fields

        categoryField.setText("");

        descriptionField.setText("");

        amountField.setText("");

        saveExpensesToFile();

    }

    private void deleteExpense() {

        String selected = expenseList.getSelectedValue();

        if (selected != null) {

            expenseListModel.removeElement(selected);

            expenseMap.remove(selected);

            saveExpensesToFile();

        }

    }

    private void loadExpensesFromFile() {

        File file = new File(FILE\_NAME);

        if (file.exists()) {

            try (BufferedReader reader = new BufferedReader(new FileReader(file))) {

                String line;

                while ((line = reader.readLine()) != null) {

                    expenseListModel.addElement(line);

                    String[] parts = line.split("\\|");

                    if (parts.length == 3) {

                        String category = parts[0].trim();

                        String description = parts[1].trim();

                        double amount = Double.parseDouble(parts[2].trim());

                        Expense expense = new Expense(category, description, amount);

                        expenseMap.put(line, expense);

                    }

                }

            } catch (IOException e) {

                e.printStackTrace();

            }

        }

    }

    private void saveExpensesToFile() {

        try (BufferedWriter writer = new BufferedWriter(new FileWriter(FILE\_NAME))) {

            for (int i = 0; i < expenseListModel.getSize(); i++) {

                writer.write(expenseListModel.getElementAt(i) + "\n");

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(ExpenseTrackerGUI::new);

    }

}

OUTPUT:

