```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

public class TaskScheduler {
    // Task class to represent individual tasks
    static class Task {
        String description;
        String dueTime;
        boolean isCompleted;

        Task(String description, String dueTime) {
            this.description = description;
            this.dueTime = dueTime;
            this.isCompleted = false;
        }
        public String toString() {
            return (isCompleted ? "[Completed] " : "") + description + " (Due: " + dueTime + ")";
        }
    }

    public static void main(String[] args) {
        // List to store tasks
        ArrayList<Task> taskList = new ArrayList<>();
```

```
        // Frame setup
        JFrame frame = new JFrame("Task Scheduler");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(500, 400);

        // Components
        DefaultListModel<Task> taskModel = new DefaultListModel<>();
        JList<Task> taskListView = new JList<>(taskModel);
        taskListView.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

        JTextField taskField = new JTextField(20);
        JTextField timeField = new JTextField(10);
        JButton addButton = new JButton("Add Task");
        JButton deleteButton = new JButton("Delete Task");
        JButton completeButton = new JButton("Mark as Completed");

        // Add task button action
        addButton.addActionListener(e -> {
            String description = taskField.getText().trim();
            String dueTime = timeField.getText().trim();
            if (!description.isEmpty() && !dueTime.isEmpty()) {
                Task task = new Task(description, dueTime);
                taskModel.addElement(task);
                taskField.setText("");
                timeField.setText("");
            } else {
                JOptionPane.showMessageDialog(frame, "Please fill out both fields!", "Error",
JOptionPane.ERROR_MESSAGE);
            }
        });
```

```
        // Delete task button action
        deleteButton.addActionListener(e -> {
            int selectedIndex = taskListView.getSelectedIndex();
            if (selectedIndex != -1) {
                taskModel.remove(selectedIndex);
            } else {
                JOptionPane.showMessageDialog(frame, "Please select a task to delete!", "Error",
JOptionPane.ERROR_MESSAGE);
            }
        });

        // Mark as completed button action
        completeButton.addActionListener(e -> {
            int selectedIndex = taskListView.getSelectedIndex();
            if (selectedIndex != -1) {
                Task selectedTask = taskModel.get(selectedIndex);
                selectedTask.isCompleted = true;
                taskModel.set(selectedIndex, selectedTask); // Refresh the view
            } else {
                JOptionPane.showMessageDialog(frame, "Please select a task to mark as completed!",
"Error", JOptionPane.ERROR_MESSAGE);
            }
        });

        // Layout
        JPanel inputPanel = new JPanel();
        inputPanel.add(new JLabel("Task:"));
        inputPanel.add(taskField);
        inputPanel.add(new JLabel("Due Time:"));
        inputPanel.add(timeField);
```

```
        inputPanel.add(addButton);

        JPanel buttonPanel = new JPanel();
        buttonPanel.add(deleteButton);
        buttonPanel.add(completeButton);

        frame.setLayout(new BorderLayout());
        frame.add(inputPanel, BorderLayout.NORTH);
        frame.add(new JScrollPane(taskListView), BorderLayout.CENTER);
        frame.add(buttonPanel, BorderLayout.SOUTH);

        // Show frame
        frame.setVisible(true);
    }
}
```