```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;


class AuctionItem {

    String name;

    double basePrice;

    double highestBid;

    String highestBidder;


    AuctionItem(String name, double basePrice) {

        this.name = name;

        this.basePrice = basePrice;

        this.highestBid = basePrice;

        this.highestBidder = "None";

    }


    void placeBid(String bidder, double bid) {

        if (bid > highestBid) {

            highestBid = bid;

            highestBidder = bidder;

        }

    }

}


public class AuctionManagementSystem {

    private JFrame frame;

    private ArrayList<AuctionItem> items;

    private DefaultListModel<String> itemListModel;
```

```java
public AuctionManagementSystem() {
    items = new ArrayList<>();
    itemListModel = new DefaultListModel<>();
    initialize();
}

private void initialize() {
    frame = new JFrame("Auction Management System");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(500, 400);
    frame.setLayout(new BorderLayout());

    // Top panel for adding items
    JPanel addItemPanel = new JPanel(new FlowLayout());
    JLabel itemNameLabel = new JLabel("Item Name:");
    JTextField itemNameField = new JTextField(10);
    JLabel basePriceLabel = new JLabel("Base Price:");
    JTextField basePriceField = new JTextField(10);
    JButton addItemButton = new JButton("Add Item");

    addItemPanel.add(itemNameLabel);
    addItemPanel.add(itemNameField);
    addItemPanel.add(basePriceLabel);
    addItemPanel.add(basePriceField);
    addItemPanel.add(addItemButton);

    // Center panel for displaying items
    JPanel centerPanel = new JPanel(new BorderLayout());
    JList<String> itemList = new JList<>(itemListModel);
    JScrollPane scrollPane = new JScrollPane(itemList);
```

```
        centerPanel.add(new JLabel("Auction Items:"), BorderLayout.NORTH);
        centerPanel.add(scrollPane, BorderLayout.CENTER);

        // Bottom panel for placing bids
        JPanel bidPanel = new JPanel(new FlowLayout());
        JLabel bidderLabel = new JLabel("Bidder Name:");
        JTextField bidderField = new JTextField(10);
        JLabel bidLabel = new JLabel("Bid Amount:");
        JTextField bidField = new JTextField(10);
        JButton placeBidButton = new JButton("Place Bid");

        bidPanel.add(bidderLabel);
        bidPanel.add(bidderField);
        bidPanel.add(bidLabel);
        bidPanel.add(bidField);
        bidPanel.add(placeBidButton);

        // Add panels to the frame
        frame.add(addItemPanel, BorderLayout.NORTH);
        frame.add(centerPanel, BorderLayout.CENTER);
        frame.add(bidPanel, BorderLayout.SOUTH);

        // Action listeners
        addItemButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String name = itemNameField.getText();
                String basePriceText = basePriceField.getText();
                if (name.isEmpty() || basePriceText.isEmpty()) {
                    JOptionPane.showMessageDialog(frame, "Please fill in all fields.", "Error",
JOptionPane.ERROR_MESSAGE);
```

```
                    return;
                }
                try {
                    double basePrice = Double.parseDouble(basePriceText);
                    AuctionItem item = new AuctionItem(name, basePrice);
                    items.add(item);
                    itemListModel.addElement(item.name + " (Base Price: $" + basePrice + ")");
                    itemNameField.setText("");
                    basePriceField.setText("");
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(frame, "Invalid base price.", "Error",
JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        placeBidButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                int selectedIndex = itemList.getSelectedIndex();
                if (selectedIndex == -1) {
                    JOptionPane.showMessageDialog(frame, "Please select an item to bid on.", "Error",
JOptionPane.ERROR_MESSAGE);
                    return;
                }
                String bidder = bidderField.getText();
                String bidText = bidField.getText();
                if (bidder.isEmpty() || bidText.isEmpty()) {
                    JOptionPane.showMessageDialog(frame, "Please fill in all fields.", "Error",
JOptionPane.ERROR_MESSAGE);
                    return;
                }
                try {
                    double bid = Double.parseDouble(bidText);
                    AuctionItem selectedItem = items.get(selectedIndex);
                    if (bid > selectedItem.highestBid) {
                        selectedItem.placeBid(bidder, bid);
                        itemListModel.set(selectedIndex, selectedItem.name + " (Highest Bid: $" + bid + ", Bidder: " + bidder + ")");
                        bidderField.setText("");
                        bidField.setText("");
                    } else {
                        JOptionPane.showMessageDialog(frame, "Bid must be higher than the current highest bid.", "Error", JOptionPane.ERROR_MESSAGE);
                    }
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(frame, "Invalid bid amount.", "Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        frame.setVisible(true);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(AuctionManagementSystem::new);
    }
}
```