

線上拍賣系統專案規劃書 (Online Auction System Project Plan)

1. 專案背景與目標 (Project Background & Objectives)

隨著電子商務的普及，拍賣系統需要更彈性的機制來滿足不同交易場景。本專案旨在開發一個基於 **Java** 物件導向技術 (OOP) 的拍賣平台，整合多種拍賣模式、自動化競標邏輯以及完整的售後交易流程。

核心目標：

- 多樣化競標：支援一般競標、底價拍賣、荷蘭式遞減拍賣及直接購買。
- 自動化管理：實作自動出價 (Auto-Bidding) 與最後時刻出價自動延長機制。
- 完整生命週期：涵蓋從商品上架、即時通知、成交紀錄到付款運送的完整流程。

2. 系統架構設計 (System Architecture)

本系統採用 模組化分層設計，確保程式碼的可讀性與可維護性。

2.1 專案結構 (Project Package Structure)

- auction.interfaces**: 定義系統契約 (Biddable, Expirable, Watchable, Rateable)。
- auction.users**: 管理使用者角色 (Buyer, Seller, Admin)。
- auction.bidding**: 處理出價邏輯、自動出價與事件封裝 (AuctionEvent)。
- auction.listings**: 實作各類拍賣模式 (Standard, Dutch, Reserve, BuyNow)。
- auction.core**: 管理售後核心流程 (Transaction, Payment, Shipping)。

2.2 設計模式應用 (Design Patterns)

- 觀察者模式 (Observer Pattern): 透過 **Watchable** 與 **AuctionEvent** 實作，當有新出價時自動通知所有關注者。
- 策略模式概念 (Strategy Concept): 不同的 **Listing** 子類別實作不同的 **placeBid** 邏輯。

3. 功能規格說明 (Functional Specifications)

3.1 拍賣類型與邏輯

模式	核心邏輯	關鍵功能
一般競標 (Standard)	價高者得	支援自動加價競爭、最後 5 分鐘出價自動延長。
底價拍賣 (Reserve)	門檻限制	結標時若未達 reservePrice 則以流標處理。

荷蘭式拍賣 (Dutch)	降價競標	價格隨時間遞減，首位出價者即成交。
直購商品 (BuyNow)	固定價格	無須競價，支付指定金額立即完成交易。

3.2 關鍵機制

- Sniper Protection** (防秒殺機制): 結標前 5 分鐘內出價，結束時間自動延長 5 分鐘。
- Auto-Bid** (自動加價): 使用者設定預算上限，系統代為以最小增額與競爭者對抗。

4. 實作細節與技術亮點 (Implementation Highlights)

4.1 抽象類別與繼承應用

系統核心 **Bid** 被定義為 抽象類別 (**Abstract Class**)，強制區分 **ManualBid** 與未來可能的各類擴展。**Listing** 類別則整合了多個介面，確保所有商品都具備時間管理與競標屬性。

4.2 資料完整性與錯誤處理

- 使用 **LocalDateTime** 進行精確的時間運算。
- 透過 **try-catch** 與自定義異常處理 (如 **IllegalArgumentException**)，確保無效出價不會破壞系統紀錄。

5. 測試計畫與展示場景 (Testing & Demo Scenarios)

本專案透過 **AuctionDemo** 實作以下關鍵測試：

- 競爭者對抗: 模擬 Bob 與 Alice (AutoBid) 的出價攻防戰。
- 時間延長驗證: 測試在結束前 1 分鐘出價，系統是否正確延長至 6 分鐘。
- 底價邏輯驗證: 測試當最高價未滿底價時，系統判定為「流標」而非成交。
- 售後流程自動化: 自動產生成交紀錄並串接付款與運送模組。

6. 未來擴展方向 (Future Enhancements)

- 資料庫整合: 將目前記憶體中的 List 轉換為 MySQL 資料庫存儲。
- 圖形化介面 (GUI): 開發 Java Swing 或 JavaFX 介面提升使用者體驗。
- 網路連線功能: 利用 Socket 編程實作多人在線即時競標。