Software Requirements Specification

Online Auction

File Name	Online Auction
Creation Date	22.09.2025
Authored by	Robert Kadak
Reviewed by	

REVISION HISTORY

VERSION	DATE	DESCRIPTION	AUTHOR
1.0	22.09.2025	Initial version	**Robert Kadak**

CONTENTS

1 SW System Overview

Specify the purpose and the overview of the SRS.

This SRS describes the requirements for a software system that supports an online auction platform. The system enables sellers to list products with different sale methods (immediate sale, auction, negotiated), and buyers to browse, register, bid, or purchase products. The system automates bidding, payment processing, commission deduction, and notification of winners.

1.1 Purpose

Describe the purpose of the system. What problem does it solve? Who are the intended users? Why is it being developed?

The purpose of the system is to provide a platform where sellers can list products and buyers can purchase them through different modes of sale.

1.2 Scope

Define the scope of the system. What functionality is included? What is explicitly excluded? Mention benefits and key features.

- Seller product listing with three sale types:
  - Immediate sale (fixed price, time limit, description, delivery).
  - Auction (starting bid, time limit, description, delivery).
  - Negotiated (starting price; auction staff set other parameters).
- Buyer registration, login, and category-based browsing.
- Bid management (placing, updating bids; tracking highest bidder).
- Automatic winner notification after auction deadline.
- Multiple payment options (e-wallets, bank accounts, etc.).
- Automatic commission deduction and seller payout.
- Exchange of buyer contact details to the seller after payment.

1.3 Use-Case Diagram

Provide a high-level UML use-case diagram showing main actors and their interactions with the system.

(TBD)

1.4 General Constraints

List technical and business constraints such as programming language, operating system, performance limitations, and standards.

- Implementation language: C++.
- Platform: Windows/Linux console.
- Database: Filesystem.
- Performance:
  - Bid placement and updates must process within ≤ 2 seconds.
  - Auction winner determination must complete immediately after the deadline.
- Standards:
  - UML notation for diagrams.
- User Interface: CLI.

1.5 Assumptions and Dependencies

State assumptions (e.g., availability of internet, supported devices) and dependencies (e.g., external APIs, hardware).

- Stable access to electricity and a local PC.
- No internet connection or external APIs are required.

1.6 Acronyms and Abbreviations

List all acronyms and abbreviations used in the document along with their explanations.

Terms Used	Description of terms

2 SW Functional Requirements

2.1 Features / Functions to be Implemented

All functional requirements should be derived from User Stories or Use Cases.
This means that instead of listing abstract features, you first describe how users interact with the system and what goals they achieve.
User Stories – short, simple descriptions of a feature told from the perspective of the user (e.g., "As a registered user, I want to reset my password so that I can regain access to my account.").
Use Cases – structured scenarios that describe interactions between actors and the system, including preconditions, steps, and outcomes.
From these stories/cases, you can then identify:

- *User interactions (e.g., authentication, profile management).*
- *Business processes (e.g., order processing, reporting).*
- *Integrations (e.g., with external APIs or third-party systems).*
- *System logic (e.g., validation, workflows, automation).*
- *Algorithms (if required, e.g., recommendation or prediction).*

Each function must be traceable back to a User Story or Use Case, ensuring that the system is built strictly according to user and business needs.

All functional requirements are derived from User Stories and Use Cases.
User Stories

- As a seller, I want to list a product with details (title, description, delivery terms) so that buyers can view and purchase it.
- As a seller, I want to choose a sale type (Immediate Sale, Auction, Negotiated) so that I can control how the item is sold.
- As a seller, I want to receive payment (minus commission) after a successful sale so that I can get compensated for my product.
- As a buyer, I want to browse products by category so that I can find items of interest.
- As a buyer, I want to register an account so that I can place bids or make purchases.
- As a buyer, I want to place a bid in an auction so that I can try to win the item.
- As a buyer, I want to pay for a purchased item using different methods (e-wallets, bank transfer) so that I can complete the transaction securely.
- As auction staff, I want to configure sale parameters for negotiated deals so that sellers and buyers have clear rules.
- As auction staff, I want to notify the winning bidder when the auction ends so that transactions can proceed.
- As auction staff, I want to deduct a commission automatically so that the auction platform generates revenue.

Identified Functions

- User interactions:
  - Account registration and authentication.
  - Seller dashboard for product listing.
  - Buyer interface for browsing, bidding, and purchasing.
  - Notifications (auction results, purchase confirmations).
- Business processes:
  - Sale lifecycle management (Immediate Sale, Auction, Negotiated).
  - Payment processing and commission handling.
  - Delivery coordination (providing buyer's contact info to seller).
- System logic:
  - Validation of bids (must be higher than current highest).
  - Automatic auction closing at deadline.
  - Commission calculation and payout to sellers.
- Integrations:
  - None
- Algorithms:
  - Auction winner determination (highest valid bid before deadline).
  - Sorting/filtering products by categories.

2.1 Acceptance Criteria

Define how each requirement will be validated: test cases, acceptance tests, or quality metrics.

- The system must prevent unregistered users from placing bids or purchases.
- A product listing must include at least: title, description, price/bid information, and delivery terms.
- For auctions:
  - Only bids higher than the current highest are accepted.
  - When the auction deadline passes, no further bids can be made.

    - The highest bidder receives a notification of deal closure.
- For immediate sales:
    - Once purchased, the product is no longer available to others.
- For negotiated sales:
    - Auction staff must be able to approve and finalize deal parameters.
- Payment:
    - Buyers must be able to select from at least two payment methods.
    - Payments must be logged with timestamp, buyer ID, and seller ID.
- Commission:
    - The system must deduct a configurable percentage from each successful payment.
- Delivery:
    - After payment, the system must release buyer contact information to the seller.

2.2 Implementation Requirements

Provide details of specific implementation requirements if applicable. For example, integration with existing systems, supported platforms, or algorithms.

- The platform must support CLI access.
- Auction timing must be implemented with a reliable scheduler.
- Commission percentage must be configurable by admin.
- UML diagrams must be delivered for Use Cases, Class Models, and Sequence Flows.

3 SW Non-Functional Requirements

3.1 Resource Consumption

Specify performance and resource limits (CPU, memory, storage, response time).
Average response time for browsing products or placing bids: <= 3 seconds.
Maximum memory usage: <= 200 MB.
Maximum size of daily transaction logs: <= 10 MB.

3.2 License Issues

State licensing requirements and constraints on third-party software or libraries.
Only open-source libraries with permissive licenses (MIT, Apache 2.0, BSD) may be used.

3.3 Coding Standard

Define coding style and standards that must be followed.
All classes, methods, and must include descriptive comments (purpose, parameters, return values).
Consistent naming conventions (snake_case for methods/variables, PascalCase for classes).
Error handling must use standardized exception classes with clear messages.

3.4 Modular Design

Specify architectural requirements such as modularity, extensibility, and maintainability.
The system must be designed in separate, loosely coupled modules to ensure scalability and maintainability:
User Management – registration, authentication, buyer/seller roles.
Product Management – listing, categorization, description, delivery conditions.
Auction Management – immediate sale, bidding system, negotiated sale.
Payment Processing – handling multiple payment methods, commissions, and transfers.
Notification System – informing buyers/sellers of bids, wins, and payments.
Logging & Reporting – transaction logs, system activity, error tracking.
Administration Panel – auction staff controls, approvals, and oversight.

3.5 Reliability

Define requirements for reliability, error handling, and fault tolerance.
The system must reject invalid bids, duplicate payments, or expired transactions without crashing.
All bids and payments must be transaction-safe and atomic (to prevent data loss).
Error messages must be logged to a secure file/database for troubleshooting.
In case of system failure, recovery must ensure no financial or bidding data is lost.
The auction timer must be reliable, ensuring deadlines cannot be bypassed.

3.6 Portability

List target platforms and environments where the system should operate.
Must run on Windows and Linux.

3.7 General Operational Guidelines

Provide guidelines for scalability, robustness, ease of use, and maintainability.
Code should allow for easy modification.

4 SW Design Artifacts

4.1 CRC Cards (Class–Responsibility–Collaboration)

List the main classes with their responsibilities (action verbs) and collaborators (related classes); keep items concise and implementation-agnostic.

Seller	
Responsibility	**Collaborator**
List products Choose type of sale	Product

Buyer	
Responsibility	**Collaborator**
Browse categories Register Buy or bid on products	Product

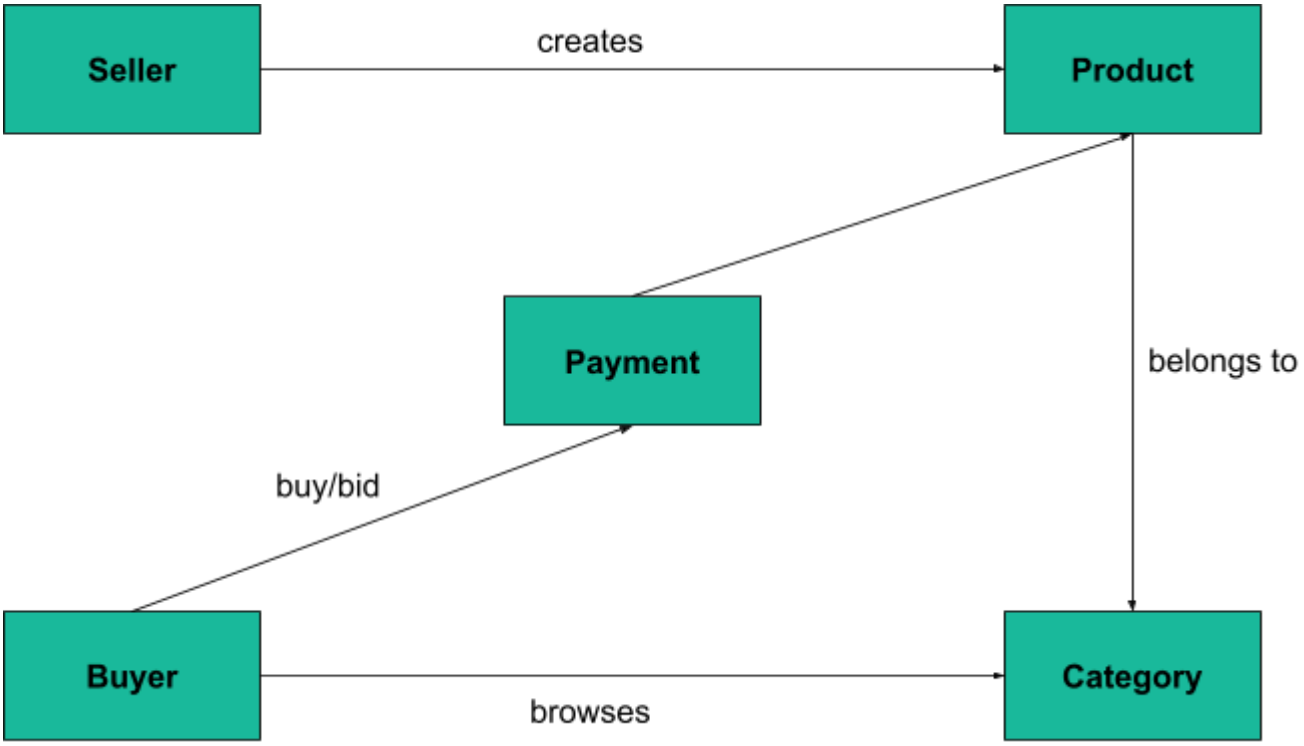
Product	
Responsibility	**Collaborator**
Store product information	Category Buyer Seller Payment

Category	
Responsibility	**Collaborator**
Describe products	Product

Payment	
Responsibility	**Collaborator**
Support multiple methods	Buyer Seller

4.2 Conceptual UML Diagram (entities & relationships)

Draw a conceptual class diagram with key entities and their relationships; focus on nouns from User Stories/Use Cases, omit methods and low-level details.


Seller
creates
Product
Payment
belongs to
buy/bid
Buyer
browses
Category