Software Requirements Specification

for

SmartBid - Auctioning Application

Version 1.0 approved

Prepared by Rishi Kadam, Manas More, Aayush Mohite

Shah and Anchor Kutchhi Engineering College

12/10/2025

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to describe the functional and non-functional requirements for the SmartBid online auction application. The system allows vendors to initiate and manage auctions while enabling users to place bids and complete payments upon winning. This SRS outlines all features, interfaces, and performance criteria essential for the project's development.

1.2 Document Conventions

- Headings follow the IEEE SRS standard structure. Each requirement is labeled as REQ-<number>.
- Functional and non-functional requirements are described using clear and verifiable language

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, testers, and stakeholders involved in the SmartBid project. Developers should focus on Sections 3 and 4 for implementation details, testers on the functional requirements, and project managers on overall system scope and dependencies.

1.4 Product Scope

SmartBid is a web-based auction platform where vendors can create and manage auctions, and users can place bids. The system automatically tracks the top three bidders. If the highest bidder fails to make the payment within the allotted time, the system declares the next highest bidder as the winner. The goal is to provide a transparent, secure, and efficient online auction experience.

1.5 References

- IEEE Software Requirements Specification Template (Karl E. Wiegers, 1999)
- ReactJS Documentation – https://reactjs.org
- Node.js Documentation – https://nodejs.org
- MySQL Documentation – https://dev.mysql.com

2. Overall Description

2.1 Product Perspective

SmartBid is a new, self-contained web application that operates in a client-server architecture. The frontend is developed using React, Tailwind, and Vite for rapid and efficient rendering. The backend uses Node.js, connected to a MySQL database hosted via XAMPP.

2.2 Product Functions

Major functionalities include:

- Vendor account management (start/stop auctions)
- User registration and bidding system
- Tracking of top three bidders per auction
- Payment processing for winning bidders
- Automatic reassignment of winners if payment deadlines are missed

2.3 User Classes and Characteristics

- Vendors: Create and manage auctions, monitor bids, end auctions.
- Users: Participate in auctions, place bids, make payments.
- Administrators: (Optional) Monitor the system, manage user activities, and resolve issues.

2.4 Operating Environment

- Frontend: React, Tailwind CSS, Vite
- Backend: Node.js (Express.js framework)
- Database: MySQL (via XAMPP server)
- Browser Support: Chrome, Firefox, Edge
- OS: Windows, Linux, macOS

2.5 Design and Implementation Constraints

- Must use Node.js and MySQL as specified
- Data consistency and concurrency during bidding must be maintained
- Internet connectivity required for all real-time auction operations

2.6 User Documentation

- User Guide: Explains registration, bidding, and payment procedures
- Vendor Guide: Details auction creation and management
- FAQ section for common troubleshooting steps

2.7 Assumptions and Dependencies

- Users have stable internet access
- Payments are processed via a third-party payment gateway
- Vendors and users register with valid credentials

3. External Interface Requirements

3.1 User Interfaces

- Home Page / Auction Listing Page – Displays all active, upcoming, and completed auctions with basic details like item name, current bid, remaining time, and vendor.
- Auction Detail Page – Shows complete details of a specific auction including item description, bid history, and a live updating section for current top bids.
- Bidding Interface – Allows users to place bids, view current bid status, and see notifications if they've been outbid.
- Vendor Dashboard – Shows all auctions created by the vendor along with their status (active, closed, upcoming).
- Auction Creation Form – Allows vendors to create new auctions by entering item details, base price, auction duration, and upload images.

3.2 Hardware Interfaces

The system runs on standard web-capable devices such as PCs, laptops, and smartphones.

3.3 Software Interfaces

- Node.js backend communicates with MySQL database
- REST APIs handle bid submissions, auction data, and payment status
- Frontend communicates with backend via HTTP/HTTPS requests

3.4 Communications Interfaces

- SmartBid uses HTTP/HTTPS for data transfer.
- WebSocket or polling may be used to update real-time bid changes.

4. System Features

4.1 Bidding System

4.1.1 Description and Priority:

Allows users to place bids on active auctions. The system records bids in real time and maintains a leaderboard of the top three bidders.

Priority: High

4.1.2 Stimulus/Response Sequences

User places bids → system checks if the bid is higher → If bid is higher the leaderboard is updated → Previous highest bidder is notified

4.1.3 Functional Requirements

- **REQ-1:** System must record every valid bid.
- **REQ-2:** Display top 3 bidders per auction.
- **REQ-3:** Notify bidders when outbid.

4.2 Auction Management

4.2.1 Description and Priority:

Vendors can create, start, and stop auctions. Auctions automatically close after a scheduled time or manual termination.

Priority: High

4.2.2 Stimulus/Response Sequences

Vendor creates auction → system manages the auction → auction is terminated after specified time → highest bidder is notified to make the payments

4.2.3 Functional Requirements

- **REQ-4:** Allow vendors to start or stop an auction.
- **REQ-5:** Restrict auction edits after bidding starts

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system should handle at least 100 concurrent users without performance degradation.
- Bid updates should reflect within 2 seconds.

5.2 Safety Requirements

The system must prevent data loss during transactions and ensure consistency of auction data even in case of failures.

5.3 Security Requirements

- User authentication via email and password.
- All sensitive data transmitted via HTTPS.
- Payment information handled securely.

5.4 Software Quality Attributes

- Reliable
- Maintainable
- Easy to use.
- UI should be responsive and accessible.

5.5 Business Rules

- Only registered users can place bids.
- Vendors can only manage their own auctions.
- Payment deadlines are fixed and non-negotiable.

6. Other Requirements

- Future enhancements may include AI-based bid recommendations
- Auction analytics dashboards

Appendix A: Glossary

- **Bid:** The amount offered by a user for an auction item.
- **Vendor:** User who creates and manages auctions.
- **Auction:** A process where users bid for an item, and the highest bidder wins.

Appendix C: To Be Determined List

- TBD-1: Payment gateway integration details
- TBD-2: Admin role access policies