Software Requirements Specification

for

Online Auction system for students to buy and sell their personal items

Version <1.0>

Prepared by

Group Members

Anvitha Sajja	**SE22UARI147**	**se22uari147@mahindrauniversity.edu.in**
Lasya Priya	**SE22UARI186**	**se22uari186@mahindrauniversity.edu.in**
Nithin Reddy	**SE22UARI114**	**Se22uari114@mahindrauniversity.edu.in**
Sharath Kotte	**SE22UARI077**	**Se22uari077@mahindrauniversity.edu.in**

Instructor: ***Avinash Arun Chauhan***

Course: **Software Engineering**

Lab Section: ***AI***

Teaching Assistant: ***Murali Krishna Bukkasamudram***

Date: **<place the date of submission here>**

Contents

CONTENTS I

REVISIONS II

1 INTRODUCTION 1

1.1 Document Purpose 1
1.2 Product Scope 1
1.3 Intended Audience and Document Overview 1
1.4 Definitions, Acronyms and Abbreviations 2
1.5 Document Conventions 2
1.6 References and Acknowledgments 3

2 OVERALL DESCRIPTION 3

2.1 Product Overview 3
2.2 Product Functionality 5
2.3 Design and Implementation Constraints 5
2.4 Assumptions and Dependencies 3

3 SPECIFIC REQUIREMENTS 8

3.1 External Interface Requirements 8
3.2 Functional Requirements 9
3.3 Use Case Model 10

4 OTHER NON-FUNCTIONAL REQUIREMENTS 12

4.1 Performance Requirements 12
4.2 Safety and Security Requirements 13
4.3 Software Quality Attributes 14

5 OTHER REQUIREMENTS 15

APPENDIX A – DATA DICTIONARY 16

APPENDIX B - GROUP LOG 9

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft Type and Number	Full Name	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	00/00/00

1 Introduction

The **Online Auction System for Students** is a platform where students can buy and sell personal items through auctions. It allows users to list items, place bids, and complete transactions securely. This section covers the system's core features, including user registration, item listing, bidding, and payment processing. It also highlights the platform's benefits, such as affordability, ease of use, and secure transactions within the student community.

1.1 Document Purpose

This document serves as a comprehensive guide for the development of the **Online Auction System for Students**. The primary objective of this project is to create a secure and efficient platform where students can buy and sell personal items through an auction-based system. It aims to streamline the process of listing items, placing bids, and completing transactions while ensuring a user-friendly experience.

The document outlines the system's functional and non-functional requirements, architecture, design specifications, implementation details, and testing strategies. It serves as a reference for stakeholders, including students, developers, and administrators, ensuring that the system meets user needs.

By automating the auction process, this system aims to:

- Provide a secure and convenient marketplace for students.
- Enable fair and transparent bidding on listed items.
- Minimize manual effort in managing transactions.
- Offer a user-friendly interface for browsing, bidding, and selling.

This document acts as a blueprint for the system's development, testing, and deployment while ensuring adherence to software engineering best practices.

1.2 Product Scope

The Online Auction System for Students is a web-based platform that enables students to buy and sell personal items through an auction system. It ensures secure transactions, fair bidding, and a user-friendly experience while eliminating the need for external marketplaces. *Key objectives include automating the auction process, ensuring transparency, and providing an intuitive interface for browsing, bidding, and selling. The system promotes sustainable second-hand trading and creates a trusted, efficient marketplace for students.*

1.3 Intended Audience and Document Overview

- **Client**: Understands the system's capabilities, benefits, and scope.
- **Professor**: Evaluates the project's functionality, technical depth, and feasibility.

- **Developers**: Uses system specifications for implementation.
- **Testers**: Reviews testing requirements and expected system behavior.
- **Documentation Writers**: Refers to system details for creating user manuals and guides.

Document Structure

- **Overview Section**: Describes the purpose, scope, and objectives of the auction system.
- **Functional & Non-Functional Requirements**: Defines core features, such as user registration, bidding, and transaction handling.
- **System Design & Architecture**: Explains system components and their interactions.
- **Testing, Deployment & Maintenance**: Provides guidelines for ensuring system reliability and scalability.

Suggested Reading Sequence

1. **Overview Section** – For a broad understanding of the system.
2. **Functional & Non-Functional Requirements** – For clients, professors, and developers.
3. **System Design & Architecture** – For developers and testers.
4. **Testing & Deployment Guidelines** – For testers and maintainers.

1.4 Definitions, Acronyms and Abbreviations

- **API** – Application Programming Interface
- **CRUD** – Create, Read, Update, Delete
- **DBMS** – Database Management System
- **GUI** – Graphical User Interface
- **OAS** – Online Auction System (specific to this project)
- **SaaS** – Software as a Service
- **SQL** – Structured Query Language
- **SRS** – Software Requirements Specification
- **UI** – User Interface
- **UX** – User Experience

1.5 Document Conventions

- The document uses **Arial font, size 11 or 12**, for all text.
- Section and subsection titles follow the **IEEE template formatting**.
- Text is **single-spaced with 1-inch margins** on all sides.
- *Italics* are used for **comments and placeholders**.
- **Bold** is used for **headings and important terms**.
- **Bullet points and tables** improve readability.

1.5.2 Naming Conventions

- Acronyms and abbreviations (e.g., **OAS, UI, CRUD**) are defined in **Section 1.4**.
- Use case names follow a **consistent verb-noun format**, such as *"Place Bid"* or *"List Item"*.

- Database tables and attributes follow a **structured naming scheme**, such as *User_ID*, *Auction_Item*, and *Bid_Amount*.

1.6 References and Acknowledgments

*This **Software Requirements Specification (SRS)** document may refer to the following documents and resources:*

- ***User Interface Style Guide** – Defines the UI/UX standards for the system's design and usability.*
- ***Institutional Policies & Academic Regulations** – Specifies scheduling constraints, faculty workload limits, and classroom usage policies.*
- ***System Requirements Specification (SRS) Template** – Standard format for documenting software requirements.*
- ***Use Case Document** – Describes various user interactions with the system, including timetable generation, modifications, and conflict resolution.*
- ***Vision and Scope Document** – Outlines the project's purpose, scope, and high-level objectives.*
- ***Software Engineering Standards** – References IEEE or ISO standards for software development best practices.*

2 Overall Description

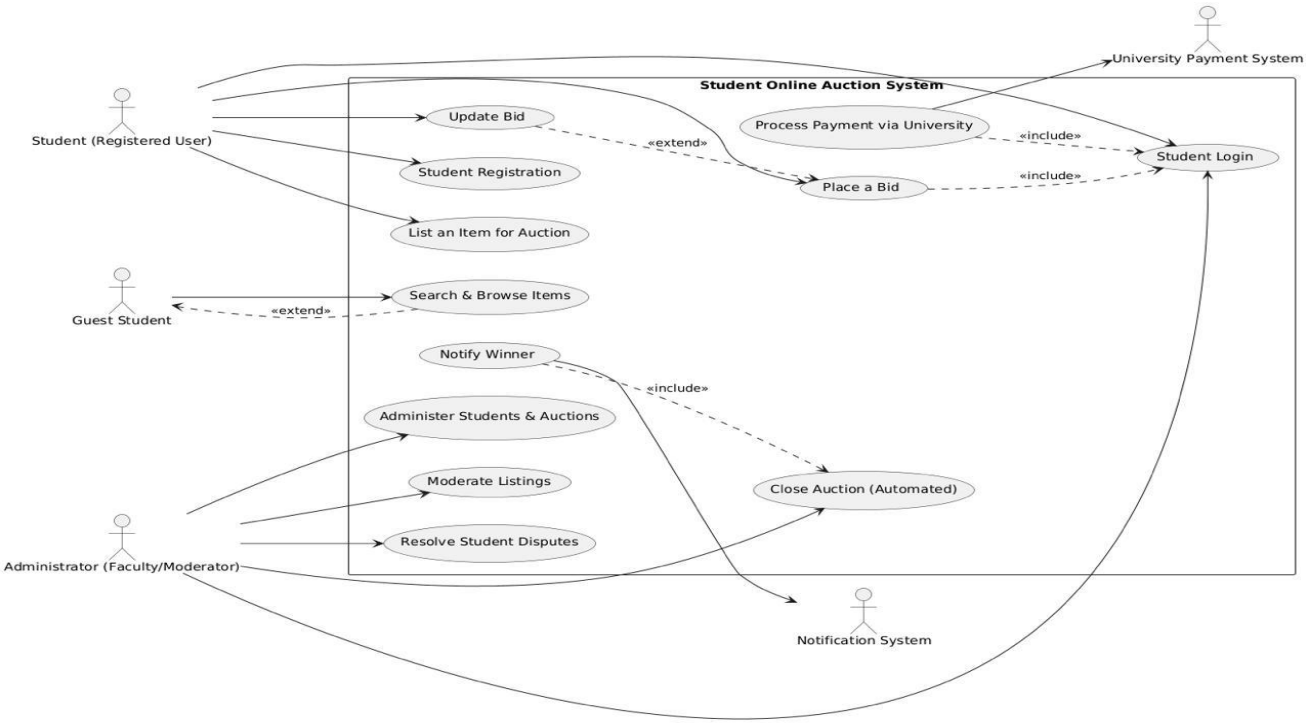
2.1 Product Overview

The **Online Auction System for Students** is a **self-contained, web-based marketplace** designed to replace traditional student-to-student selling methods with an efficient, auction-based platform. The system addresses common challenges such as **lack of transparency, price negotiation difficulties, and security concerns** by providing a structured bidding mechanism. It ensures **fair pricing, secure transactions, and an intuitive user experience** tailored specifically for student communities.

This system operates as a **standalone application** but can integrate with **student authentication portals and payment gateways** for seamless transactions. It allows students to list items, place bids, and complete purchases in a secure environment while ensuring fair competition and real-time auction updates.

The **primary external interfaces** of the system include:

- **Database System**: Stores user profiles, auction listings, bids, and transaction history.
- **User Interface**: Provides an intuitive web-based platform for listing, browsing, bidding, and managing transactions.
- **Auction Engine**: Handles bid placements, real-time updates, and determines winning bids based on predefined rules.
- **Payment Processing Module**: Manages secure transactions through external payment gateways.
- **Notification System**: Sends real-time alerts about bid status, auction completion, and payment confirmation.



2.2 Product Functionality

2.3 User Registration and Authentication – Allows students and sellers to register and securely log into the system.

2.4 Auction Item Listing – Enables sellers to list items with details like starting price, description, and images.

2.5 Bidding System – Allows students to place bids on items and receive updates on their bid status.

2.6 Auto-Bidding Feature – Supports automatic bidding within a set limit to compete for an item.

2.7 Auction Monitoring & Closing – Tracks ongoing auctions, determines the highest bidder, and closes auctions at the scheduled time.

2.8 Payment Processing – Integrates with a secure payment gateway to handle transactions between buyers and sellers.

2.9 Item Delivery & Pickup Coordination – Provides a mechanism for sellers and buyers to arrange item exchanges.

2.10 User Feedback & Ratings – Allows buyers and sellers to rate and review each other after transactions.

2.11 Admin Dashboard – Enables administrators to manage users, resolve disputes, and monitor auctions.

2.12 Design and Implementation Constraints

The development of the Online Auction System is subject to the following constraints:

1. **Modeling Language**

- The Unified Modeling Language (UML) must be used for system design and documentation, including class diagrams, sequence diagrams, and use case diagrams.
- Reference: Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide*. Pearson

2. Hardware Constraints

The system must run efficiently on standard institutional hardware, including:

- Minimum **4 GB RAM** and **Intel i3 processor** for client-side applications.
- Minimum **8 GB RAM** and **Intel i5 processor** for server-side processing.
- Cloud-based deployment must be optimized for cost-effective scaling.

3. Technology Stack

- **Programming Language:** Java (for backend logic and optimization algorithms) and Java (for frontend development).
- **Frameworks:**
  - Java (Spring Boot) for backend development.
  - Java Swing for frontend UI.
- **Database:** MySQL for storing auction data.

4. External Interfaces

- The system must integrate with external **student information systems (SIS)** and faculty management portals.

5. Parallel Operations & Scalability

- The system should support concurrent users, including administrators, faculty, and students, without performance degradation.
- It should handle a large number of auction entries without excessive load times.

6. Security Considerations

- **Role-Based Access Control (RBAC)** to ensure different access levels for administrators, buyers, and sellers.
- **Secure Authentication** using OAuth or institutional Single Sign-On (SSO).

7. Design Conventions & Standards

- Code must follow **Java Coding Standards** and best practices.
- Compliance with **IEEE 830-1998** for Software Requirements Specifications (SRS).

- Maintainability and documentation must follow **Agile and SCRUM best practices** for software development.

2.4 Assumptions and Dependencies

- Internet access is required for system functionality.
- A third-party payment gateway will handle transactions.
- The system depends on MySQL for data storage, and any database migration may require modifications.
- The project relies on Java-based development frameworks such as Spring Boot (backend) and Java Swing (frontend). Updates or deprecations in these frameworks could require system adjustments.
- The authentication mechanism depends on institutional Single Sign-On (SSO) or OAuth-based authentication.
- The system assumes students will use institutional email accounts for registration and verification.
- The auction system assumes that users will comply with institutional policies regarding fair bidding practices.
- The system assumes that sellers will accurately describe the items listed for auction to maintain user trust.
- The platform integrates with Student Information Systems (SIS) for verifying student eligibility.
- The platform requires a secure messaging system for user-to-user communication regarding transactions.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

- ***Graphical Menus:*** *Simple navigation for auction creation, bidding, and tracking.*

- **Forms & Dropdowns:** Used for entering auction details and placing bids.
- **Search & Filter Options:** Allows users to find specific auctions quickly.
- **Notification System:** Alerts users on auction updates and bid outcomes.
- **Administrators:** Manage users, oversee auction activities, and resolve disputes.
- **Sellers:** List items, set starting prices, monitor bids, and finalize transactions.
- **Bidders:** Browse auctions, place bids, and receive notifications on bid status.

3.1.2 Hardware Interfaces

- **Client Devices** – Supports desktops, laptops (min. 4GB RAM, Intel i3), and mobile devices (Android 8.0+, iOS 12+).
- **Server Hardware** – Requires cloud/dedicated servers (min. 8GB RAM, Intel i5, SSD storage).
- **Database Server** – Uses MySQL with backup and redundancy mechanisms.
- **Networking Hardware** – Needs high-speed internet and load balancers for real-time bidding.
- **Peripheral Devices (Optional)** – Supports barcode scanners and printers for auction management.

3.1.3Software Interfaces

1.Operating System Compatibility

- Supports **Windows, macOS, Linux** for desktops/laptops.
- Compatible with **Android (8.0+) and iOS (12+)** for mobile access.

2.Web Browser Compatibility

- Works on **Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari**.

3. Database Management System (DBMS)

- Uses **MySQL** for storing auction data and transactions.

4.Backend Technologies

- Developed using **Java (Spring Boot) for server-side logic**.

5.Frontend Technologies

- Uses **Java Swing for UI design** (if a desktop version is required).
- Web version built with **HTML, CSS, JavaScript (React/Angular)**.

6.Authentication & Security

- Integrates **OAuth 2.0 or institutional Single Sign-On (SSO)** for secure login.
- Supports **JWT-based authentication** for session management.

7.Payment Gateway Integration

- Interfaces with **PayPal, Stripe, Razorpay, or institutional payment systems** for transactions.

8.External System Integration

- Connects with **Student Information Systems (SIS) and Faculty Management Portals** for user verification.

9.Cloud & Hosting Services

- Can be deployed on **AWS, Azure, or Google Cloud** for scalability and availability

3.2 Functional Requirements

F1: User Registration and Authentication

- The system shall allow students to register using institutional email and password.
- The system shall provide login authentication using **OAuth 2.0 or SSO**.

F2: Auction Item Listing

- The system shall allow sellers to create auction listings with item details (name, description, images, starting price, etc.).
- The system shall allow sellers to set auction duration and minimum bid increment.

F3: Bidding System

- The system shall allow registered users to place bids on auctioned items.
- The system shall provide real-time updates on the highest bid.

F4: Auto-Bidding Feature

- The system shall allow users to set a maximum bid limit, enabling automatic bidding.

F5: Auction Monitoring & Closing

- The system shall track ongoing auctions and automatically close them at the scheduled end time.

- The system shall notify winners after auction closure.

F6: Payment Processing

- The system shall integrate with **third-party payment gateways** (PayPal, Stripe, Razorpay, etc.) for secure transactions.

F7: Item Delivery & Pickup Coordination

- The system shall facilitate communication between buyers and sellers for item exchange.

F8: User Feedback & Ratings

- The system shall allow buyers and sellers to rate and review each other after a transaction.

F9: Admin Dashboard

- The system shall allow administrators to manage auctions, users, and resolve disputes.

F10: Notification System

- The system shall send email and in-app notifications for auction updates, bid status, and payment confirmations.

…

3.3 Use Case Model

3.3.1 Use Case #1 : User Login (U1)

- **Author**: John Smith
- **Purpose**:
  To allow registered users to securely log into the system to access their personal dashboard and system features.
- **Requirements Traceability**:
  R1 - The system shall allow users to log in using a valid username and password.
  R2 - The system shall validate user credentials.
  R3 - The system shall notify users of incorrect login attempts.
- **Priority**: High
- **Preconditions**:
- The user must already be registered in the system.
- The system must be connected to the user database.
- **Postconditions**:
- If login is successful, the user is redirected to their dashboard.
- If login fails, an error message is displayed to the user.
- **Actors**:

- User (Primary Actor)
- Authentication System (Secondary Actor)
- **Extends**:
  None

3.3.2 Flow of Events

Basic Flow:

1. The user navigates to the login page.
2. The user enters their username and password.
3. The user clicks the "Login" button.
4. The system verifies the username and password.
5. If credentials are valid, the system grants access and redirects to the user dashboard.

Alternative Flow:

- **Invalid Credentials Flow**:
  - If the username or password is incorrect, the system displays an error message: "Invalid username or password."
  - The user is prompted to try logging in again.

Exceptions:

- **System Error**:
  If the database is unreachable, the system displays: "System error. Please try again later."

ncludes (other use case IDs):

- Password recovery feature should be linked for users who forgot their password (covered in another use case).

4 Other Non-functional Requirements

4.1 Performance Requirements

< ***P1. System Response Time:***

- The system shall process and display auction listings within **2 seconds** of submission.
- The system shall update bid statuses and notify users within **1 second** of a new bid placement.

P2. Concurrent Users Handling:

- The system shall support at least **10,000 concurrent users** without significant performance degradation.

P3. Bid Processing Speed:

- The system shall process and validate new bids within **500 milliseconds** to ensure fair competition.

P4. Page Load Time:

- The homepage, auction listings, and user dashboard shall load within **3 seconds** under normal network conditions.

P5. Payment Processing:

- The system shall complete payment transactions within **5 seconds**, depending on third-party payment gateway response times.

P6. Database Performance:

- The system shall handle **1,000,000 auction records** efficiently without affecting retrieval speed.
- Database queries for searching auctions shall return results within **1 second** for typical queries.

P7. Notification Delivery:

- The system shall send email or in-app notifications within **2 seconds** of an event trigger (e.g., bid status change, auction ending).

P8. System Uptime and Reliability:

- The system shall maintain **99.9% uptime**, allowing minimal downtime for maintenance.

4.2 Safety and Security Requirements

4.2.1 Safety and Security Requirements for Online Auction System

S1. User Authentication and Access Control

- The system shall implement **OAuth 2.0** or institutional **Single Sign-On (SSO)** for secure authentication.
- Users must verify their identity using a **two-factor authentication (2FA)** method before placing bids or listing items.
- **Role-Based Access Control (RBAC)** shall be enforced to restrict administrative functions to authorized users only.

S2. Data Security and Privacy

- All user data, including personal information and payment details, shall be **encrypted using AES-256** encryption.
- Communication between the client and server shall be secured with **TLS 1.3 (HTTPS)** to prevent data interception.
- The system shall comply with **GDPR** and **institutional data privacy policies** for handling student information.

S3. Secure Payment Transactions

- All payments shall be processed via a **PCI-DSS compliant** third-party payment gateway.
- Sensitive financial information shall **never** be stored on the system's servers.

S4. Fraud Prevention and Bid Integrity

- The system shall **monitor and flag** suspicious bidding behavior (e.g., artificial price inflation, bot activity).
- Users involved in fraudulent activities shall be **automatically restricted or banned** from participating in auctions.

S5. System Availability and Reliability

- The system shall implement **automatic backups** every **24 hours** to prevent data loss.
- A **disaster recovery plan** shall be in place to restore system operations within **2 hours** of critical failures.

S6. Mobile and Network Security

- The mobile version of the auction system shall use **SSL pinning** to prevent man-in-the-middle attacks.

- Users shall be automatically logged out after **15 minutes of inactivity** to reduce unauthorized access risks.

4.3 Software Quality Attributes

4.3.1.1 Reliability

- **99.9% uptime** for continuous system availability.
- **Automated error logging** and alerts for quick issue resolution.
- **Backup servers** for failover in case of system failure.

4.3.1.2 Security

- **AES-256 encryption** for secure data storage and transactions.
- **Role-Based Access Control (RBAC)** to restrict sensitive actions.
- **Regular security testing** (penetration tests) to prevent cyber threats.

4.3.1.3 Usability

- **Responsive design** for mobile and desktop access.
- **Tooltips and guided walkthroughs** for new users.
- **Confirmation prompts** for critical actions (bidding, payments).

4.3.1.4 Maintainability

- **Modular code structure** for easy updates and bug fixes.
- **Well-documented system components** for future developers.
- **Git version control** for tracking code changes.

4.3.1.5 Scalability

- Supports **10,000+ concurrent users** without lag.
- **Load balancing** to distribute traffic efficiently.
- **Horizontally scalable database** for growing data needs.

4.3.1.6 Interoperability

- **Integration with payment gateways** (PayPal, Stripe).
- **Supports institutional SSO authentication** for easy login.

4.3.1.7 Testability

- **Automated unit and integration tests** for system verification.
- **Dedicated staging environment** for pre-release testing.

5 Other Requirements

5.1 Other Requirements

1. Database Requirements

- The system shall store user data, login credentials, and transaction records in a secure relational database (e.g., MySQL, PostgreSQL).
- All sensitive data such as passwords shall be stored using encryption (e.g., SHA-256 hashing with salt).
- Database backup shall be taken daily and stored securely.

2. Internationalization Requirements

- The system shall support multiple languages, including English and Spanish.
- All user-facing text shall be stored in external resource files for easy localization.

3. Legal and Compliance Requirements

- The system shall comply with data protection regulations such as GDPR.
- Users must accept the privacy policy and terms of service before account creation.
- All user data will be stored in compliance with local data residency laws.

4. Reuse Objectives

- The authentication module (login, registration) shall be designed as a reusable component for integration in other systems.
- The notification system (email/SMS alerts) will be designed as a service that can be used by other modules.

5. Performance Requirements

- The system shall handle up to 1000 concurrent users without performance degradation.
- System response time for login and dashboard loading shall be under 2 seconds.

6. Security Requirements

- The system shall implement SSL/TLS for secure communication.
- The system shall implement session timeout after 15 minutes of inactivity.
- User roles and permissions shall be enforced to restrict access to sensitive areas.

7. Mobile and Cross-platform Support

- The system shall be responsive and accessible from both desktop and mobile devices.
- A mobile app version may be considered in future updates.

Appendix A – Data Dictionary

Name	Type	Description	Possible Values / States	Related Operations	Related Requirements
User_ID	Integer (Primary Key)	Unique identifier for each user	Auto-increment, unique	Create, Read, Update, Delete (CRUD)	R1 - Manage user accounts
Username	String	Login name chosen by user	Alphanumeric, 5-15 characters	Create, Read, Update	R1 - User login
Password	String (Encrypted)	User's secret password (encrypted)	Encrypted string (e.g., hashed)	Create, Update, Validate	R2 - User authentication

Email	String	User's email address for notifications	Valid email format (e.g., user@example.com)	Create, Read, Update	R3 - Email notifications
Login_Status	Boolean	Indicates if the user is currently logged in	True (Logged In), False (Logged Out)	Update, Read	R4 - Manage sessions
Account_Status	String	Status of the user account	"Active", "Inactive", "Suspended"	Update, Read	R5 - Account management
Item_ID	Integer (Primary Key)	Unique identifier for each auction item	Auto-increment, unique	Create, Read, Update, Delete (CRUD)	R6 - Manage auction items
Item_Name	String	Name/title of the auction item	Alphanumeric	Create, Read, Update	R6 - Auction listing
Item_Description	Text	Detailed description of the item	Any text	Create, Read, Update	R6 - Auction listing
Start_Price	Decimal (Currency)	Minimum starting price for bidding	Any positive value	Create, Read	R7 - Bidding functionality
Current_Bid	Decimal (Currency)	Current highest bid placed	Any positive value	Read, Update	R7 - Bidding functionality
Bidder_ID	Integer (Foreign Key)	User ID of highest bidder	Matches existing User_ID	Read, Update	R7 - Track highest bidder
Bid_Time	DateTime	Timestamp when the bid was placed	Date and time format	Create, Read	R8 - Record bid timing
Auction_Status	String	Current status of the auction	"Open", "Closed", "Cancelled"	Update, Read	R9 - Auction lifecycle management
Payment_Status	String	Status of payment for won items	"Pending", "Completed", "Failed"	Update, Read	R10 - Payment processing