Auction Web Application Requirements Specification

This document specifies the conditions and capabilities that must be met or possessed by the Auction Web Application. It is developed jointly between the Acme Auctions, Inc. client and BevoTech Co. development team.

This specification applies to release 1.0 of the Auction Web Application.

Approval

Recommended for approval:

Dot Matrix	Mon Capitane
Dot Matrix Acme Auctions, Inc. IT Project Coordinator	Mon Capitane BevoTech Co. Project Manager
13 Jan 2014	13 Jan 2014
Date	Date

Approved for development:

D. Bigg Bozz	Bill M. Moore
D. Bigg Bozz Acme Auctions, Inc. Vice President	Bill M. Moore BevoTech Co. Account Executive
13 Jan 2014	13 Jan 2014
Date	Date

Overview

The rest of this document presents the context of the system, then specifies interface requirements, functional requirements, performance requirements, design requirements, and required system attributes.

Product Context

The Auction Web Application is a stand-alone system that will be the core of the Acme Auctions, Inc. Internet business line. It enables members of the public to, via the World Wide Web, list items to be sold at auction, bid on listed items, and pay for items purchased via winning bids.

Eventually, there will be additional Acme Auctions, Inc. business processes and IT systems in the Auction Web Application's environment, but not at the time of initial release.

System Interfaces

The system shall have one user interface, presented over HTTP in standard Web browsers.

The system, in its initial release, has no requirements for other hardware, software, or communications interfaces beyond the platform APIs.

Operations

The system shall operate continuously, and not require any periods of unavailability for batch operations, backup, etc.

Product Functions

The major functions of the system are adding a new listing, bidding on a listing, and paying for the purchase.

User Characteristics

The users of the system are members of the general public. No particular auction expertise or technical expertise can be assumed.

Constraints

There are several security-sensitive aspects of the system: User personal data (names, e-mail addresses, etc.) and passwords, payment data, and the auction process. These require confidentiality and integrity protection.

The system shall run on the UTCS "z" node's LAMP environment, and be implemented in standard dialects of the PHP, SQL, and JavaScript programming languages.

User Interface Requirements

The Auction Web application user interface shall be served as HTML over HTTP, to be presented in a standard Web browser.

[There should be much more detail here, but for this class, just use your creativity. Here are some UI design principles from ISO 9241: The user interface should be:

- Suitable for the task and user's skill level,
- Self-descriptive: It should be clear what the user should do next,
- In conformance with user expectations: Consistent with itself and other common UIs,
- Controllable: User sets pace and sequence of interaction, and
- Error-tolerant: Forgiving of user mistakes.

Use these principles and the use cases below to design your UI.]

No online help documents are required, but guidance should be incorporated into the design of the UI pages.

The UI must not require any particular browser, version, OS, plug-in, or display dimensions. The UI must be equally usable with a variety of input technologies, specifically mice, trackpads, and touch displays.

See the "Design Requirements" for technology constraints.

The language of the UI shall be US English.

[Normally, there should be a requirement to meet Web Content Accessibility Guidelines (WCAG) 2.0 here, but for the sake of the timeframe of this class, this is *not* required.]

Functional Requirements

The system's functional requirements are specified as use cases, which are elaborated on the following pages.

Actors

ac01 Seller

ac02 Buyer

ac03 System timer

User Cases

uc01 List item

uc02 Cancel listing

uc03 Browse listings

uc04 Place bid

uc05 Close auction

uc06 Pay for purchase

uc07 Update listing

uc08 Register as user

Use Case: uc01 List item

Use Case Title:	List item	**Use Case ID:** uc01
Actors:	• ac01 Seller	
Brief Description:	Seller provides information on an item, and offers it for sale at auction.	
Preconditions:	• User has registered (uc08), and therefore has a logon ID.	
Triggering Business Event:	Seller decides to sell an item at auction.	
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**
	1. Select the "list item" function and log on. 2. Provide item details, such as category, name, description, condition, photo, etc. 3. Also provide reserve price and auction bid deadline date-time.	• Validate logon data. • Validate item data. • Start action timer. • Confirm listing posted.
Postconditions (always):	• User is informed of listing creation success or failure.	
Postconditions (success):	• Item is listed as available for bidding. • Auction timer is running.	
Alternate flows and exceptions:	• If item data is invalid, permit user to correct data.	
Nonfunctional requirements:		
Assumptions:		
Issues:	• How are categories determined?	
Sources:		

Use Case: uc02 Cancel listing

Use Case Title:	Cancel listing	**Use Case ID:**	uc02
Actors:	• ac01 Seller		
Brief Description:	Seller cancels auction of previously listed item for sale.		
Preconditions:	• Seller previously listed the item (uc01).		
Triggering Business Event:	Seller decides to withdraw item from sale at auction.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. Log on, view list of active auctions. 2. Select auction to cancel. 3. Select "cancel auction" function, and confirm.	• Validate logon data. • Verify auction has not closed. • Confirm cancellation.	
Postconditions (always):	User is informed of cancellation success or failure.		
Postconditions (success):	• Auction is cancelled, and all bids in that auction are rescinded.		
Alternate flows and exceptions:	• If selected auction has closed before user confirms cancellation, inform user of failure to cancel.		
Nonfunctional requirements:			
Assumptions:			
Issues:			
Sources:			

Use Case: uc03 Browse listings

Use Case Title:	Browse listings	**Use Case ID:**	uc03
Actors:	• ac02 Buyer		
Brief Description:	Buyer reviews items offered for sale.		
Preconditions:			
Triggering Business Event:	Buyer decides to investigate items that may be bid upon.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. View current auctions list. *Note:* There are multiple ways of viewing this list: recent listings, by category, by seller, search by keyword, and so on. [They should all be specified in use cases, but for this class, just use your creativity.]	• Present items for sale according to user's browse/search criteria	
Postconditions (always):			
Postconditions (success):			
Alternate flows and exceptions:			
Nonfunctional requirements:			
Assumptions:			
Issues:			
Sources:			

Use Case: uc04 Place bid

Use Case Title:	Place bid	**Use Case ID:**	uc04
Actors:	• ac02 Buyer		
Brief Description:	Buyer places a bid in an open auction.		
Preconditions:	• User has registered (uc08), and therefore has a logon id.		
Triggering Business Event:	Buyer decides to bid on an item for sale.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. Find an item offered for sale (see uc03). 2. Select "place bid" function and log on. 3. Enter bid amount, and confirm.	• Validate bid amount is valid: above current high bid and reserve. • Verify auction is still open at time of bid confirmation. • Record bid and confirm to user.	
Postconditions (always):	• User is informed of bid placement success or failure.		
Postconditions (success):	• Bid is recorded in auction. Current high bid is updated.		
Alternate flows and exceptions:	• If bid is not above current high bid and reserve, permit user to specify a new bid amount.		
Nonfunctional requirements:			
Assumptions:			
Issues:			
Sources:			

Use Case: uc05 Close auction

Use Case Title:	Close auction	**Use Case ID:**	uc05
Actors:	• ac03 System timer		
Brief Description:	Auction is closed at the designated bid deadline, and the winning bidder and seller are notified.		
Preconditions:	• Auction opened (uc01).		
Triggering Business Event:	An auction close timer fires.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
		• Validate that auction is open. • Change status of auction to closed. • Notify seller. • Notify winning bidder to pay for purchase.	
Postconditions (always):	• Auction is closed. • Seller notification is sent.		
Postconditions (success):	• Winning bidder is notified to pay for purchase.		
Alternate flows and exceptions:	• If no bid met the reserve price, there is no winning bid. Notify seller.		
Nonfunctional requirements:	• Auctions should be closed and notifications sent within 1 second of the specified deadline.		
Assumptions:			
Issues:			
Sources:			

Use Case: uc06 Pay for purchase

Use Case Title:	Pay for purchase	**Use Case ID:**	uc06
Actors:	• ac02 Buyer		
Brief Description:	The winning bidder in an auction enters payment details for the purchase.		
Preconditions:	• Buyer placed a valid bid in the auction (uc04), and the auction closed (uc05) with that bid be the highest.		
Triggering Business Event:	Buyer receives notification of winning bid.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. Log on and select “pay for purchase” function. 2. Enter payment details.	• Validate logon data. • Validate payment details.	
Postconditions (always):	• User is informed of payment success or failure.		
Postconditions (success):	• Payment confirmation is sent to the buyer and seller.		
Alternate flows and exceptions:	• If payment is invalid, prompt user to correct payment details.		
Nonfunctional requirements:	• Payment data must be kept confidential. • Payment data must be not be modifiable by attackers.		
Assumptions:			
Issues:			
Sources:			

Use Case: uc07 Update listing

Use Case Title:	Update listing	**Use Case ID:**	uc07
Actors:	• ac01 Seller		
Brief Description:	Seller changes details of an item offered for sale.		
Preconditions:	• Seller previously listed the item (uc01).		
Triggering Business Event:	Seller notices that details of an item offered for sale need to be corrected.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. Log on, view list of active auctions. 2. Select listing to update, and select "update listing" function. 3. Enter corrections, and confirm.	• Validate logon data. • Verify auction has not closed. • Validate updated data. • Confirm update.	
Postconditions (always):	• User is informed of update success or failure.		
Postconditions (success):	• Listing details are changed.		
Alternate flows and exceptions:	• If selected auction has closed before user confirms update, inform user of failure to update. • If updated data is invalid, permit user to correct data.		
Nonfunctional requirements:			
Assumptions:			
Issues:	• What if the seller attempts to update the reserve price when there already are bids? • Can all fields be updated, or only a subset?		
Sources:			

Use Case: uc08 Register as user

Use Case Title:	Register as user	**Use Case ID:**	uc08
Actors:	• ac01 Seller • ac02 Buyer		
Brief Description:	Create a user account with user-provided data, including logon ID and password.		
Preconditions:			
Triggering Business Event:	User begins to perform use case that requires an account, and is prompted to login or register.		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. Select “register” function. 2. Agree to terms & conditions. 3. Provide requested account information.	• Display terms & conditions. • Prompt for account information. • Validate “reasonableness” of provided data. • Prevent creation of duplicate accounts. • Confirm creation of account.	
Postconditions (always):	• User is informed of account creation success or failure.		
Postconditions (success):	• User account is created.		
Alternate flows and exceptions:	• If user ID already exists, notify user and provide opportunity to select another ID.		
Nonfunctional requirements:	• Users’ personal data must be kept confidential. • Users’ personal data must be not be modifiable by attackers. • This function must not be usable by attackers to probe the system’s account data.		
Assumptions:			
Issues:			
Sources:			

Use Case: uc00 <title>

See Armour and Miller, Advanced Use Case Modeling*, ISBN 0-201-61592-4, for proper use.*

Use Case Title:	<Infinitive verb-object phrase describing actor goal. Examples: "Buy pizza", "Book flight">	**Use Case ID:**	uc00
Actors:	• <List the types of end users that *initiate* interaction with the system for this use case.>		
Brief Description:	<*Brief* is the important word here. Give an overview of the user's goal and the system's responsibility here. Leave the detail for the flow description below. Keep below about 40 words.>		
Preconditions:	• <Assertions about state that must true prior to initiating this use case. Example: "User has signed up for the service.">		
Triggering Business Event:	<This is an event *in the user's business process* that triggers this interaction with the system.>		
Flow of Events: *(Do NOT specify the user interface in the use case. Use the approach advocated by Wirfs-Brock and Larman.)*	**User View**	**System Responsibilities**	
	1. <Don't get bogged down in detail – just cover the externally visible behavior as it relates to the business process.>		
Postconditions (always):	• <Assertions that are always true after execution of the use case: Example: "Shopping basket is empty.">		
Postconditions (success):	• <Assertions that are true when the use case succeeds: Example: "Order is cancelled.">		
Alternate flows and exceptions:	• <List weird cases and error cases here>		
Nonfunctional requirements:	• <Requirements *for this use case* that are not visible in the sequence of events, such as security, performance, usability, or availability requirements.>		
Assumptions:	• <Assumption: "Supposition that something is true about how project goals and objectives are met". Add to the project assumptions tracking list! Also, don't confuse preconditions and assumptions.>		
Issues:	• <Issue: "Point of debate or controversy; a matter that is in dispute between two or more parties". Add to the project issues tracking list!>		
Sources:	• <List the people, meetings, and documents that led to this use case being built the way it is.>		

Performance Requirements

The system shall maintain response times ≤ 1 second for all transactions under a 5 concurrent active user load on the following hardware platform:

UTCS "z" node: 8 core Intel Xeon CPU X3460 @ 2.8 GHz, with 16 GB physical RAM.

Design Requirements

The application must run on the following server platform:

- Ubuntu Linux 2.6.32-55-server
- Apache httpd 2.2.22
- MySQL 5.1.72
- PHP version 5.3.2-1ubuntu4.22

The presented Web pages must conform to the following Web standards:

- *Hypertext Transfer Protocol -- HTTP/1.1*, RFC 2616. Internet Society.
- *Uniform Resource Identifier (URI): Generic Syntax*, RFC 3986. Internet Society.
- *HTML5: A vocabulary and associated APIs for HTML and XHTML*. World Wide Web Consortium.
- *Polyglot Markup: A robust profile of the HTML5 vocabulary*. World Wide Web Consortium.
- *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification*. World Wide Web Consortium.
- *ECMAScript Language Specification,* ECMA-262. 5.1 Edition. Ecma International.
- *The Unicode Standard*. Version 6.3.0.
- *Portable Network Graphics (PNG) Specification (Second Edition)*. World Wide Web Consortium.
- *Information Technology – Digital Compression and Coding of Continuous-Tone Still Images – Requirements and Guidelines,* CCITT Recommendation T.81. International Telecommunication Union. [JPEG]

Served HTML must validate as well-formed HTML5 and XML documents. Served CSS must validate as well-formed CSS 2.1.

The application must be primarily implemented in the PHP, SQL, and JavaScript programming languages, using standard dialects/versions thereof.

Monetary values must *NOT* exhibit floating point round-off errors.

Required System Attributes

Security

User personal data (names, e-mail addresses, etc.) and passwords must be kept confidential and protected from unauthorized modification.

Payment data must be kept confidential and protected from unauthorized modification.

Auctions must be protected from unauthorized modification or subversion.

Privacy

Sellers' identities for open auctions will be available to logged-in users. Winning bidders' identity will be available to the seller. No personal information may be disclosed in other cases.

Availability

Target system availability is ≥ 0.998. There are no specific requirements in the application design and implementation for high availability.

Maintainability

Application code should be readable, per generally-accepted good coding practices.