**Bidkings Online Auction**

The model we are adopting is the ascending price auction. To win, bidders must outbid each other forcing the price up. The winner is the bidder at the with the highest price end on the auction.

The online will allow many variations such as bidders can alter,cancel , bid on multiple items and buy bid packs .

**Auctioneer**

**HTML  
Browser**

**Bidder**

**Web Interface**

**Web Interface**

DB

**Dockers**

DJango

**DJango**

Admin Interface

**API**

Server

Fig 1

Figure 1 represents the high level software model for performing online bidding. There are two main parties, The Bidder(user) and Auctioneer.

The bidder interects with the auctioner via a web browser and an API.

The Auctionerr runs a webserver which contains a Docker container and Django runs in the Docker container.

The entire application is database driven. All state information like bids, timing and information about bid packs is contained in database . When a client submits a bid or request a purchase history, a DB transaction occurs which in turn generates dynamic web pages in response to the bidder activity.

**1- Processes Involved**

1. *Initialization*: The Auctioneer sets up the auction and advertises it.
2. *Registration:* Before you can place bids, bidders must Register with the Auctioneer.
3. *Price Quote*: Bidder obtains a price quote from the auctioneer for the product.
4. *Bidding*: A bidder submits a bid to the auctioneers.
5. *Winner Determination*: Winner is determined by the auctioneer according to the auctions rules (time, price etc)
6. *Transaction settlement/payment*: Collecting payment from bidder and delivering the goods the the winning bidder
7. *Bidder/Winner Notification:* Informing the bidder about his product which also includes a confirmation and bid receipt.

**2-Web Interface Navigation**

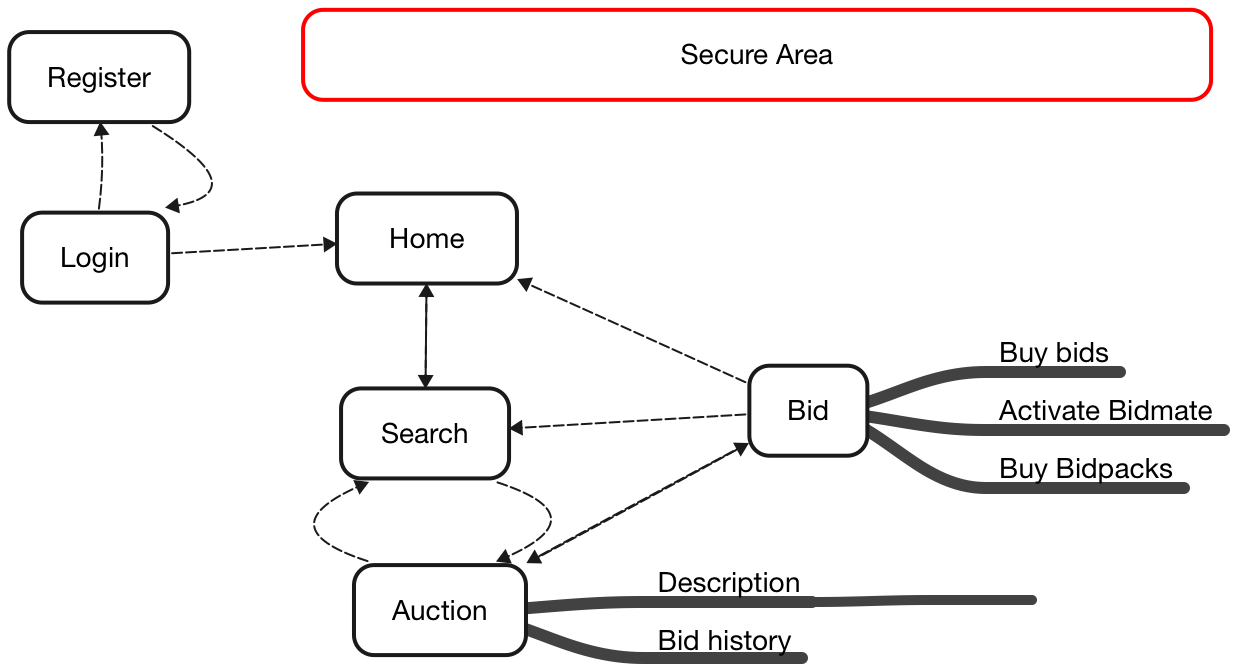


Fig 2 Web Interface Navigation

Fig 2 Illustrates the web navigation of the website . Dotted lines illustrates the links between sections and arrows indicate the navigational direction.

After registering, a user can log in using a password. Upon login a user enters the secure area (shown by the dash box in Figure 2). This is a collection of pages which contains operations that only a registered user can perform.

The home page is the main area for the bidder. Bidders are able to search for a listed auction. Once a bidder has selected an auction, they are able to obtain a description of the item and information regarding the auction (i.e., price quote, minimum increment and time remaining). The bidder can then use this information to submit a new bid. After submitting a bid, a bidder can return to the home page, search for a new auction or return to the current auction.

In the secure area, specific information regarding user(*id, passwords & user data)* is carried from one page to the other using session variables . The values in session variables exists until the session terminates. This allows information to be passed over web pages or to other websites.

An alternative approach for passing information between pages is to use query strings which Django fully support.

Session Variables must be destroyed after a predetermined amount of bidder activity (e.g. user who forgets to log off.)

Upon submitting a bid, the bidder receives an email confirming that his/her order has been received. The bidder will also be informed of such information via the web page.

**3-Object Model**

Auctioner

Database

**Bidder**

UserID

Password

Register login

Terminate auction search

list auction increment

price quote Time remaining

submit bid buy bid packs

logoff

Register

Determine bid

The web interface allows human bidders to interact with the Auctioneer. The greyed out background indicates its internal state and the bottom portion(blue background) indicates methods that it can use.

The bidder interacts with the Auctioneer via the Auctioneer methods.

Bidder has only two methods(additional methods will be created as app is developed) r*egister* and *determine bid*. During registration , a bidder obtains a *user id & password* . Bidder determins his/her bid according to individual preferences.

The *Auctioneer*  manages the database and application backend. The database contains all information regarding bidders, their bids and auctions conducted