Table of Contents

Data Types

User		
Attribute	**Data type**	**Nullable**
username	String	Not Null
password	String	Not Null
first_name	String	Not Null
last_name	String	Not Null

Admin_User		
Attribute	**Data type**	**Nullable**
position	String	Not Null

Item		
Attribute	**Data type**	**Nullable**
item_id	Integer	Not Null
item_name	String	Not Null
description	String	Not Null
category	String	Not Null
condition	Integer	Not Null
starting_bid	Decimal	Not Null
min_sale_price	Decimal	Not Null
get_now_price	Decimal	NULL
returnable	Boolean	Not Null
auction_length	Integer	Not Null
auction_end_time	Datetime	Not Null
sale_price	Decimal	NULL

Category		
Attribute	**Data type**	**Nullable**
category_name	String	Not Null

Rating		
Attribute	**Data type**	**Nullable**
rating_time	Datetime	Not Null
nbr_stars	Integer	Not Null
comment	String	NULL

Bid		
Attribute	**Data type**	**Nullable**
bid_amount	Decimal	Not Null
bid_time	Datetime	Not Null

Business Logic Constraints

User

- **Users who are new to GT Bay must register first.**
- **Users who have an existing GT Bay account will not be able to register.**
- **Regular Users may be assigned Administrative privileges (making them Administrative Users) only through DBA actions.**
- **Only Administrative Users can view two administrative reports, Category Report and User Report**

Item

- **Condition takes a value from 1, 2, 3, 4, and 5, representing New, Very Good, Good, Fair, and Poor respectively.**
- **Starting Bid, Minimum Sale Price, Get It Now Price (if not NULL), Bid Amount are all in USD and are all positive numbers with 2 decimal places.**
- **Starting Bid must be less than or equal to Minimum Sale Price.**
- **Minimum Sale Price must be less than or equal to Get It Now Price.**
- **Auction Length takes a value from 1, 3, 5, and 7 days.**
- **Number of stars must be 0, 1, 2, 3, 4, or 5 in a rating.**
- **Only the listing user can edit the item's description.**
- **The "Get It Now!" option/button is only available if a Get It Now Price has been specified.**

- **Purchase using the "Get It Now!" option is only allowed for items whose auctions have not yet ended.**
- **Bidding is only allowed for items whose auctions have not yet ended.**
- **Bid Amount must be at least $1 higher than the highest Bid Amount on the same item earlier in time, and must be less than the Get It Now Price (if one has been set).**
- **Search results only include items whose auctions have not yet ended.**
- **Auction results only include items whose auctions have already ended.**
- **Category report aggregates information only about items in the category that have Get It Now Price.**

Category

- **Categories can be added to. The names of categories can be changed, with the initial Category names being Art, Books, Electronics, Home & Garden, Sporting Goods, Toys, and Other.**

Assumptions

- How auction winners will be calculated:
  - Winner is stored in the DB as a field of **Item** table, and is NULL when a new item is inserted.
  - When a User purchases an Item using the Get It Now option, Winner is updated to be the Username of the person who made the purchase.
  - When ***Auction Results*** button is clicked from the **Main Menu** form, all Items whose Auction End Time is no later than the current time are checked for winners. If Winner is NULL, bids on the Item are checked. If no bid is found for the Item, leave Winner as NULL, otherwise Winner is updated to be the Username of the person who placed the latest bid on the Item.

Task Decomposition with Abstract Code

Login

Task Decomp

Lock Types: Read-only on User table

Enabling Conditions: None

Frequency: Medium
Consistency (ACID): not critical, order not critical
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User enters username (*$userName*), password (*$password*)
- User clicks / enter ***Login*** button
    - **login validation process** executes
        - If userName exists and passwords match
            - store $User object with ($User.userId, $User.userName, $User.position, ...) to session
            - go to **Main Menu** form
        - else
            - return invalid user password error message
            - go to **Login** form

Register

Task Decomp

Lock Types: Read-Write on User table
Enabling Conditions: None
Frequency: Low
Consistency (ACID): critical that userName is unique
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User Enters: *$firstName. $lastName, $userName, $password, $confirmation*
- User clicks / enter ***Register*** button
    - **Registration validation process** executes
        - If all fields have data continue
        - else
            - return field validation error message(s)
            - go to **Register** form

- If $password and $confirm match continue
- else
  - return confirmation validation error message
  - go to **Register** form
- If $userName is unique continue
- else
  - return non unique username error message
  - go to **Register** form
- (validation passed)
  - persist User data
  - add *$User* object with *($User.userId, $User.userName, $User.position, ...*) to session
  - go to **Main Menu** form

Main Menu / Navigation Bar

Task Decomp

- **Lock Types:** None
- **Enabling Conditions:** Triggered by successful login.
- **Frequency:** High
- **Consistency (ACID):** not critical, order not critical
- **Subtasks:** Mother Task is not needed. No decomposition needed.

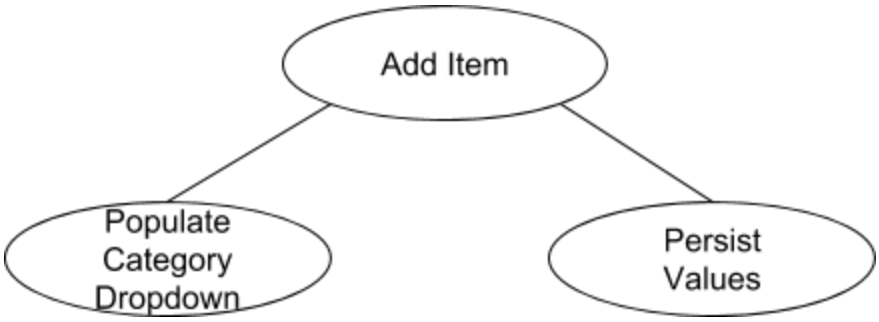
Abstract Code

- Show ***Search for Items***, ***List an Item for Sale***, ***View Auction Results*** buttons.
- If current User is an Administrative User
  - Show "***Category Report***" and "***User Report***" buttons.
  - Show administrative Position
- Show $userName
- Show "***Log Out***" button.
- Upon:
  - Click ***Search for Items*** button- Jump to the **Search For Items** task.
  - Click ***List an Item for Sale*** button- Jump to the **Add Item For Sale** task.
  - Click ***View Auction Results*** button- Jump to the **View Auction Results** task.

- Click ***View Category Report*** button- Jump to the **View Category Report** task.
- Click ***View User Report*** button- Jump to the **View User Report** task.
- Click ***Log Out*** button- Invalidate login session and go back to the **Login** form.

Add Item for Sale

Task Decomp



Lock Types: Read on Category table, Write on Item table
Enabling Conditions: All are enabled by User login
Frequency: Medium - all subtasks have the same frequency.
Consistency (ACID): Dropdowns must be populated before allowing User input. Values must be validated and derived values must be calculated before persisting to database.
Subtasks: Mother Task is needed to make sure subtasks are executed sequentially, all values are read and all input values are valid, derived values are computed, and values are saved to database

Abstract Code

User clicked on ***List my item*** button on the **Main Menu** form**:**
Show **New Item for Auction** form
Populate *Category*, *Condition*, and *Auction Length* dropdowns from pre-existing values
Wait for either the ***Cancel*** or the ***List My Item*** button to be pushed.
If the ***Cancel*** button was pushed:

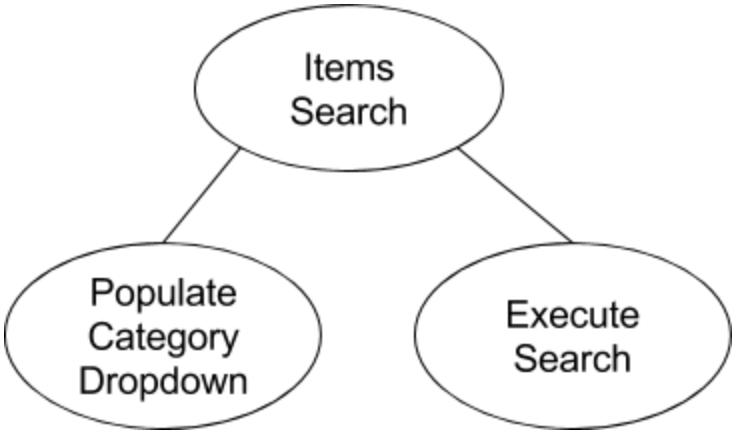
- Dismiss the **New Item for Auction** form
- Display the **Main Menu** form

If the ***List My Item*** button was pushed, do the following:

- Make sure *Item Name* and *Description* fields are not empty
- Check to make sure *Start auction bidding at* and *Minimum sale price* fields are not empty and have values that parse as US currency greater than $0.00
- If the *Get it Now price* field is not empty, make sure the value is >= the *Minimum sale price field* value and parses as US currency
- Make sure *Start Auction bidding at* value is less than or equal to any other monetary values listed.
- If any of the above are violated display appropriate error messages and wait for user interaction
- If all values are valid:
    - persist item data from the values in all fields, dropboxes, and check boxes
    - Generate a unique ItemID and save it along with item data
    - Calculate the auction end time by adding Auction Length to the current time, and save this auction end time along with item data
    - Display **Main Menu**

Search for Items

Task Decomp



Lock Types: Read on Category table and Item table
Enabling Conditions: Enabled by a user's login
Frequency: High

Consistency (ACID): not critical
Subtasks: Mother Task is needed to make sure subtasks are executed sequentially.

Abstract Code

- User clicked on the ***Search for Items*** button on the **Main Menu** form
- **Item Search** form is displayed
    - Jump to **Populate Category Dropdown** task
- Upon:
    - Click ***Cancel*** button **Item Search** form is dismissed and **Main Menu** form displayed
    - Click ***Search*** button:
        - If ((*$Keyword* != empty) and ((*$Minimum_price* is not supplied) or (*$Minimum_price* is supplied and is >= 0.00)) and ((*$Maximum_price* is not supplied) or (*$Maximum_price* is supplied and is >= *$Minimum_price*))
            - Note: The ***Category*** and ***Condition*** dropdowns will have default values and thus will not require validation. The default value for both dropdowns is blank. Blank is equivalent to “all” categories and “all” conditions respectively.
            - run a database query using the values on this form plus checking only running auctions (only items in running auctions that meet all of the supplied criteria will be returned), then go to **Item Search Results**
        - Else
            - return field validation error message(s)
            - Go to **Item Search** form

Item Search Results

Task Decomp

Lock Types: Read on Item table

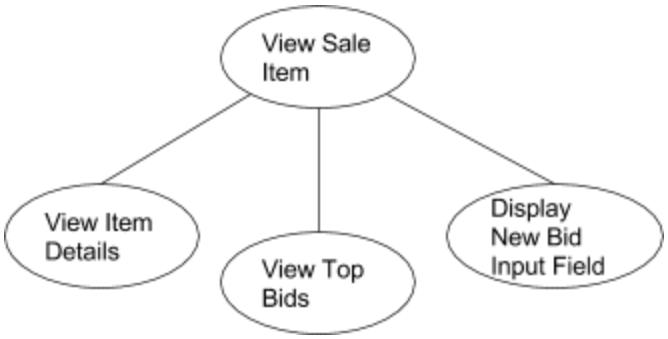
Enabling Conditions: A search has occurred
Frequency: Often
Consistency (ACID): not critical
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- ***Search*** button on **Item Search** form was clicked and a query based on its values was run
- Headings *ID, Item Name, Current Bid, High Bidder, Get It Now Price,* and *Auction Ends* heading are displayed on row one of results
- Under each heading are listed the itemID, Item name, current bid amount, high bidder user name, item Get It Now Price amount, and when the auction for item ends, displayed as a date and time for each item result, one item per row, in ascending chronological order by *Auction Ends*
- Wait for User interaction. Upon
    - Click *Item Name* go to **Item for Sale** form
    - Click *Back to Search* return to **Search for Items** form

View Sale Item

Task Decomp



Lock Types: Reads on Item table, Category table, Bid table, and User table
Enabling Conditions: All are enabled by a user's login and an item search
Frequency: High, all subtasks have the same frequency

Consistency (ACID): not critical, even if the item details (description) are being edited and/or new bids are being placed, order does not matter
Subtasks: All tasks must be done, but can be done in parallel. Mother task is required to coordinate subtasks. Order is not necessary.

Abstract Code

- User clicked on ***Item Name*** on **Search Results**
- Run the **View Sale Item** task: query for information about the item where the item ID is that of the item clicked in the **Search Results**.
- **View Item Details** subtask will:
    - Retrieve the *Item ID, Item Name, Description, Category, Condition, Returnable,* and *Auction End time* from the Item table and Category table by querying on Item ID and display their values in the corresponding forms of the **Item for Sale** form.
    - If Item.get_now_price is not NULL, display *Get It Now Price* and ***Get It Now!*** button
    - If the listing User of the item matches $User stored by the current session at **Log In**, display ***Edit Description*** button.
- **View Top Bids** subtask will:
    - Retrieve information about the top 4 bids from a combination of User data and Bid data, based on the item ID. The *Bid Amount, Time of Bid,* and *Username* will be displayed on the **Item for Sale** form, sorted by the Time of Bid with the latest one first.
- **Display New Bid Input Field** subtask will:
    - Retrieve information about the Get It Now Price and the Starting Bid of the item and the Bid Amount of the current bid from a combination of Item data and Bid data, based on the item ID.
    - If there is no current bid, set the minimum bid equal to the Starting Bid of the item, else set the minimum bid to be $1 above the current bid.
    - If the minimum bid is less than the Get It Now Price or the Get It Now Price is not set, display *Your bid* input field together with the minimum bid, else display a message suggesting using the ***Get It Now!*** option instead.

- The **Item for Sale** form will then wait for interaction from the user, which comprises 5 possible actions:
    - User clicks ***View Ratings*** button to activate **View Ratings** task
    - User clicks ***Edit Description*** button, only available to the owner of the ITEM/poster of the auction, to activate **Update Description** task
    - User clicks ***Get It Now*** button to activate **Get It Now** task
    - User clicks ***Cancel*** button to return to **Item Search Results** form
    - User clicks ***Bid On This Item*** button to activate **Bid on Item** task

Bid on Item

Task Decomp

Lock Types: Read on Item table, Read-Write on Bid table
Enabling Conditions: An active Auction is underway and the Item has not been Won
Frequency: High
Consistency (ACID): critical as last highest bid must be maintained to ensure a fair auction
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User Enters: *$bidAmount*
- User clicks / enter ***Bid On this Item*** button
    - **Bid validation process** executes
        - Get item auction status (read from Item table)
        - If the item auction has not ended
            - Users bid must be one dollar higher than current highest $bidAmount and must also be less than the $getItNowPrice
            - else
                - return field validation error message(s)

        - If $bidAmount >= $getItNowPrice error message to suggest user Get It Now Button
        - If $bidAmount < (current highest $bidAmount + 1) error message bid not accepted, amount too low
    - go to **Item for Sale** form
- (validation passed)
    - persist bid data
    - add *$Bid* object with *($Bid.amount, $Bid.timestamp, $Bid.user, $Bid.item, ...*)
    - go to **Main Menu** form

Get Item Now

Task Decomp

Lock Types: Read-Write on Item table
Enabling Conditions: Enabled by an item's lookup, the item's Get It Now price having been set and an active auction underway for the item.
Frequency: Medium
Consistency (ACID): critical as Get It Now will end auction and must manage if multiple users try to get it now at the same time or near same time.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User clicks / enter ***Get It Now!*** button
- If Item.winner is NULL and Item.auction_end_time is later than the current time

    - update Item.winner with $User.username where $User.username is the username of the current user stored by the current session at **Log In**
    - update Item.auction_end_time with the current time
    - update Item.sale_price with Get It Now Price
    - go to **Main Menu** form
- Else go to **Main Menu** form, with error message

Update Description

Task Decomp

Lock Types: Write on Item table
Enabling Conditions: User Viewing Item
listed the Item for Sale
Frequency: Low
Consistency (ACID): not critical
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User clicks / enter ***Edit Description*** button
    - **Edit Description process** executes
        - Popup edit form with current description is displayed to user
        - User modifies description text
        - If user selects ***Save Button***
            - Validate that $description field has data.
                - If no data return error message
                - else
                    - Persist updated description
                    - dismiss popup

        - return to **Item for Sale** form
    - If user selects ***Cancel Button***
        - dismiss popup
        - return to **Item for Sale** form

View Item Ratings

Task Decomp

Lock Types: Read on Rating table and User table
Enabling Conditions: User's Login and separate view item ratings request
Frequency: Often
Consistency (ACID): not critical
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User clicks/ enter the ***View Ratings*** button
    - Go to **Item Ratings** form
    - Find current Item using Item.itemID
        - Display Item.itemID
        - Display Item.itemName
    - Find each Rating for the Item
        - Collect User.userName of the associated User
        - Collect nbrStars (number of stars)
        - Collect ratingTime
        - Collect comment
        - increment rating tally
        - increment star tally
    - Display Average Rating (star tally / rating tally)
    - In most recent to least recent order from top to bottom for each Rating

        - If User.userName == current user
            - Display ***Delete My Rating*** button, User.userName, nbrStars, ratingTime, comment
        - Else
            - Display collected User.userName, nbrStars, ratingTime, comment
    - Display collected User.userName, nbrStars, ratingTime, comment, in most recent to least recent order from top to bottom (paginate or scroll as needed).
    - Display
    - If the ***Cancel*** button is pressed
        - Close the **Item Ratings** form
        - Go back to the **Item for Sale** form
    - If the ***Rate This Item*** button is pressed
        - Close the **Item Ratings** form
        - Run the **Add Item Rating** task
    - If the ***Delete My Rating*** button is pressed
        - Close the **Item Ratings** form
        - Run the **Delete Item Ratings** task

Add/Delete Item Rating

Task Decomp

Lock Types: Write on Rating table
Enabling Conditions: User is logged in and viewing ratings of a particular item
Frequency: Medium
Consistency (ACID): not critical
Subtasks: Mother Task is not needed. No decomposition needed.

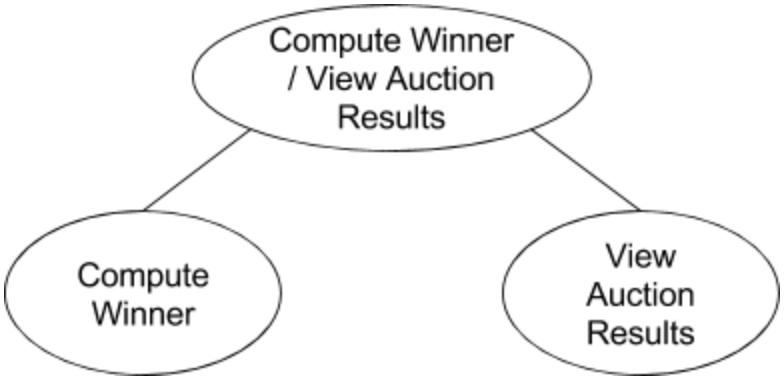
Abstract Code

- If user clicked on ***Rate This Item*** button on **Item Ratings** form

    - Add record in the appropriate table that corresponds to the logged-in User ID and the item ID was being viewed on the **Item Ratings** form
        - Save the number of stars clicked in *My Rating*, and the RatingTime.
        - Record any comments in *Comments* field from this User along with rating
    - Return to **Item Ratings** form
- If User clicked on ***Delete My Rating*** button on **Item Ratings** form
    - Delete record in the appropriate table that corresponds to the logged-in User ID and the item ID was being viewed on the **Item Ratings** form
    - Return to **Item Ratings** form

View Auction Results

Task Decomp



Lock Types: Read-Write on Item table
Enabling Conditions: User login
Frequency: Medium
Consistency (ACID): not critical
Subtasks: Mother Task is needed to make sure subtasks are executed sequentially. Order does matter (**Compute Winner** task must be run before **View Auction Results** task).

Abstract Code

- User clicks the ***View Auction Results*** button on the **Main Menu** form

- Go to **Auction Results** form
- Display Headings *ID, Item Name, Sale Price, Winner, Auction Ended*
- Find each Item that does not have a declared winner and that has ended (based on current time)
  - if the Item has a winner
    - update Item winner
- Find each Item for which the associated auction has ended:
  - Collect Item.itemID
  - Collect Item.itemName
  - Collect sale price (if no winner, the value of sale price will be Null)
  - Collect winner (if no winner, the value of winner will be Null)
  - Collect auction ended (date & time)
- Display collected Item.itemID, Item.itemName, sale price, winner, auction ended sets in most recent to least recent order from top to bottom (paginate or scroll as needed).
- If the ***Done*** Button is pressed
  - Close the **Auction Results** form
  - Go to the **Main Menu** form

View Category Report

Task Decomp

Lock Types: Read-only on Item and Category tables
Enabling Conditions: Administrative User's Login
Frequency: Low
Consistency (ACID): not critical
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User clicks the ***View Category Report*** button on the **Main Menu** form
  - Go to **View Category Report** form

- Display Headings *Category, Total Items, Min Price, Max Price, Average Price*
- For each Category:
  - Collect each Category
  - Find each Item in Category
    - increment item tally
    - if Item has a "Get It Now Price" (GINP)
      - if item GINP price < min GINP price
        - min GINP price = item GINP price
      - if item GINP price > max GINP price
        - max GINP price = item GINP price
      - item GINP price total += item GINP price
  - Collect the sum of the items in the Category
  - Collect the min GINP price of the items in the Category
  - Collect the max GINP price of the items in the Category
  - Collect the average GINP price (item GINP price total / item tally) of the items in the Category
- Display collected Category, items tally, min GINP price, max GINP price, average GINP price sets in alphabetical order from top to bottom.
- If the ***Done*** Button is pressed
  - Close the **View Category Report** form
  - Go to the **Main Menu** form

View User Report

Task Decomp

Lock Types: Read-only on User, Item and Rating tables
Enabling Conditions: Administrative User's Login
Frequency: Low
Consistency (ACID): not critical

Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User clicks the ***View User Report*** button on the **Main Menu** form
  - Go to **View User Report** form
  - Display Headings *Username, Listed, Sold, Purchased, Rated*
  - For each User:
    - Collect User.username
    - Find each Item listed by the User
      - increment listed tally
      - If Item sold:
        - increment sold tally
    - Collect the sum of items listed by the User
    - Collect the sum of items sold by the User
    - Find each Item purchased by the username
      - increment purchased tally
    - Collect the sum of items purchased by the User
    - Find each Item rated by the User
      - increment rated tally
    - Collect the sum of items rated by the User
  - Display collected User.username, listed tally, sold tally, purchased tally, rated tally sets in descending order from top to bottom.
  - If the ***Done*** Button is pressed
    - Close the **View User Report** form
    - Go to the **Main Menu** form

Test Cases

TC#	Area	Test Case Description
L000	Login	
L001		Login page entry (username, password) credentials
L002		Valid credentials (username, password) accepted

L003		reject invalid (username, password) when user does not exist
L004		reject invalid (username, password) when user exists, password invalid