Login

Abstract Code

User enters username($username), password($password) input fields.

If data validation is successful for both username and password input fields, then:

When **Enter** button is clicked:

"SELECT EXISTS(SELECT 1 FROM User WHERE username = $username AND password = $password))"

If return 0: go back to Login form, with error message.

Else:

Pull user's info from database and store them as session variable, $userID, $position(if applicable), $username, $lastname, $firstname.

"SELECT User.userID, position, username, lastname, firstname FROM User LEFT JOIN AdminUser ON User.userID=AdminUser.userID WHERE username=$username"

Go to **Main Menu** form.

Else username and password input fields are invalid, display Login form, with error message.

Registration
Abstract code
User clicked **Register** button:
user enters first name($firstname), last name($lastname), username($username), and passoword($password). Re-enter password to confirm.

Upon click Register button:
    if invalid inputs are found:
        Pop error message and go back to register form.
    else if username already exists in database:
        "SELECT EXISTS(SELECT * FROM `User` WHERE `username` = $username limit 1);"
        Pop error message and go back to register form.
    else:
        "INSERT INTO User (username, password, firstname, lastname) VALUES ($username, $password, $firstname, $lastname)"

Main Menu/ Navigation Bar
Abstract code
After user **login** successfully:
IF the user is regular user($position is null):

The page shows:
1. List new item button
2. Search item button
3. View Auction Results button
4. Logout button.

ELSE:(position is not null, so it's admin user)

The page shows:
1. List new item button
2. Search item button
3. View Auction Results button
4. Logout button
5. View Category Report button
6. View User Report button
7. 'Position'

• Click List new item button jump to "List New Item for Sale" task
• Click Search item button, jump to "Item Search" task
• Click View Auction Results button, jump to "View Auction Reports" task
• Click View Category report button, jump to "Category Reports" task
• Click View User report button, jump to "User Reports" task
• Click Logout button, terminate login session and jump back to Login GTbay PAGE

Search Item
Abstract code
User clicked on Search for items button from Main Menu:
Run the `Search` function: query for information about the items with matching criteria
“

```
SELECT `Item`.`itemID`
    , `Item`.`item_name`
    , `Bid`.`bid_price`
    , `Item`.`start_price`
    , `User`.`username`
    , `Item`.`get_it_now_price`
    , `Item`.`end_time`
FROM `Item`
  LEFT JOIN (`Bid` INNER JOIN (SELECT `itemID`, MAX(`bid_price`) AS `high_bid` FROM
`Bid` GROUP BY `itemID`) AS `b2`
                ON `Bid`.`itemID`=`b2`.`itemID` AND `Bid`.`bid_price`=`b2`.`high_bid`) ON
`Item`.`itemID`=`Bid`.`itemID`
  LEFT JOIN `User` ON `Bid`.`userID`=`User`.`userID`
WHERE `Item`.`itemID` NOT IN (SELECT `itemID` FROM `SoldItem`)
  AND ((ISNULL(`Bid`.`bid_price`) AND `Item`.`start_price`>$min_price AND
`Item`.`start_price`<$max_price) OR (`Bid`.`bid_price`>$min_price AND
`Bid`.`bid_price`<$max_price)) AND `Item`.`condition`=`$condition` AND
`Item`.`category`=`$category` AND (`Item`.`item_name` LIKE `%$keyword%` OR
`Item`.`description` LIKE `%$keyword%`)
```

"

If all criteria matched,
    Display itemID, item_name, current_bid high_bidder, get_it_now_price and auction_ends.
Else
    Return error message that no matching items.

List New Item for Sale
Abstract code
After successfully **Log** into GTBay:
click **List an Item for Sale**, begin to fill the item for sale information:

Fill the Itemname（ $item_name ）, description($description), category($category), condition($condition), start auction bidding($start_price), minimum sale price($min_price), auction last days (server calculate the $end_time = current time + auction last days), get it now price($get_it_now_price), returns accepted($returnable).

Constraint and data validation:

$category is pre-defiend
$condition is pre-defined
except the get it now price($get_it_now_price) is optional, all other forms cannot be null
$min_price >= $start_price
$min_price <= $get_it_now_price

IF all the data is valid,click List My Item buttom:

"INSERT INTO Item(condition, description, item_name, returnable, end_time, get_it_now_price, min_price, start_price, category, owner_userID)
VALUES($condition, $description, $item_name, $returnable, $end_time, $get_it_now_price, $min_price, $start_price, $category, $userId)"

click **Cancel** bottom, go back to **menu page**

Item Bidding
Abstract code
User clicked on **Item name** button from search result page, so we already have some of the item information, $itemID, $item_name, $description, $category, $owner_userID, $condition, $returnable, $get_it_now_price, $start_price, $end_time, $min_price, $current_bid, $high_bidder.

This page will show following information and options:
"View Ratings" link;
Item ID: $itemID;
Item Name: $item_name;
Description: $description; "Edit Description" link available if $owner_userID=$userID;
Category: $category;
Condition: $condition;
Returns Accepted: $returnable;
Get It Now price: $get_it_now_price; "Get it Now!" button;
Auction Ends: $end_time;

Latest Bids:
"SELECT `bid_price`, `bid_time`, `username` FROM `Bid` INNER JOIN `User` ON `Bid`.`userID`=`User`.`userID` WHERE `itemID`=$itemID ORDER BY `bid_time` DESC limit 4;"
Your bid input option: capture variable $my_bid_price

if the user click **View Ratings** link, it will takes the user to the ratings screen for this item.

if the user click **Edit Description** link:
Pop up a input window and user can input new description.
if user click submit after input new description($new_description):
"UPDATE `Item` SET `description`=$new_description WHERE `itemID`=$itemID;)
if user click cancel:
Close pop up window and do nothing.

if the user click **Get it Now!** button:
system pop up a window with option "yes" and "no" to ask if user want to buy item with $get_it_now_price
if user click yes:
set $my_bid_price = $get_it_now_price, and trigger "Bid On This Item" button.
if user click no:
Close pop up window and do nothing.

if the user click "Bid On This Item":
System will check if $my_bid_price is valid.
check if $my_bid_price is a valid decimal;
check if $my_bid_price >= $current_bid+1;

check if $my_bid_price <= $get_it_now_price;
if $my_bid_price passed system check:
"INSERT INTO `Bid` VALUES ($userID, $itemID, NOW(), $my_bid_price);"
if $my_bid_price == $get_it_now_price:
"INSERT INTO `SoldItem` VALUES ($itemID, $get_it_now_price, NOW(), $username);"

if the user click **Cancel**:
Send user back to Search Results screen.

Item ratings
User click **view rating item** from Item for sale page, so we already have $itemID and $item_name from Item bidding task, and we have current user's $userID from session variable.

First, we pull average rating from database and display it.
"SELECT AVG(`item_rating`) FROM `Rating` WHERE `itemID`=$itemID;"

Second, we pull latest review, and current user's review from database, and display them, system will check which review is latest by check time of records, and check if review is belongs to the user by check userID.
"SELECT * FROM `Rating` INNER JOIN `User` ON `Rating`.`userID`=`User`.`userID` WHERE `itemID`=$itemID AND (`ratetime`=(SELECT MAX(`ratetime`) FROM `Rating` WHERE `itemID`=$itemID) OR `userID`=$userID);"

If user have rated this item before:
User can delete his/her rating.
"DELETE FROM `Rating` WHERE `itemID`=$itemID AND `userID`=$userID;"
User can update his/her rating, by select My Rating($item_rating) and input Comments($comment) and then click "Rate This Item" button.
"UPDATE `Rating` SET `item_rating`=$item_rating, `comment`=$comment, `ratetime`=NOW() WHERE `itemID`=$itemID AND `userID`=$userID;"
else:
User can rate this item, by select My Rating($item_rating) and input Comments($comment) and then click **Rate This Item** button.
"INSERT INTO `Rating` VALUES ($itemID, userID, $item_rating, $comment, NOW());"
User will be send back to item Ratings form after click "Rate This Item" button.
User can click the cancel button at any time to go back the item description page.

Auction Results
Abstract Code
User clicked on **View Auction Results** button from Main Menu:

1. Trigger “calculate result” function: run SQL query to calculate winner from who hold highest bid of items that already reached auction end_time. We will also exclude all item that already been calculated before(which has been insert into a result table). Then we insert those calculated result into database result table, we’ll be also include item that auction ended without a winner here for future reference.

```
"INSERT INTO `SoldItem`
SELECT `Item`.`itemID`, `BidInfo`.`bid_price`, `Item`.`end_time`, `User`.`username` FROM
`Item` LEFT JOIN (SELECT `Bid`.`bid_price`, `Bid`.`itemID`, `Bid`.`userID` FROM `Bid`
INNER JOIN (SELECT `itemID`, MAX(`bid_price`) AS `high_bid` FROM `Bid` GROUP BY
`itemID`) AS `b2` ON `Bid`.`itemID`=`b2`.`itemID` AND `Bid`.`bid_price`=`b2`.`high_bid`
INNER JOIN Item AS i2 ON Bid.itemID=i2.itemID WHERE Bid.bid_price>=i2.min_price) AS
`BidInfo` ON `Item`.`itemID`=`BidInfo`.`itemID` LEFT JOIN `User` ON
`BidInfo`.`userID`=`User`.`userID` WHERE `Item`.`end_time` < NOW() AND `Item`.`itemID`
NOT IN (SELECT `SoldItem`.`itemID` FROM `SoldItem`)"
```

2. Read from database result table to and display ID, Item Name, Sale Price, Winner, Auction Ended for each record, and sorted by auction ended time with the most recently ended auction listed first.

```
"SELECT `SoldItem`.`itemID`, `Item`.`item_name`, `SoldItem`.`sold_price`,
`SoldItem`.`winner`, `SoldItem`.`sold_time` FROM `SoldItem` INNER JOIN `Item` ON
`SoldItem`.`itemID`=`Item`.`itemID` ORDER BY `SoldItem`.`sold_time` DESC"
```

3. User can click “Done” back to main menu.

GTBay Administrative Reports

Abstract Code
• Admin Users logged in successfully will have the two more options View Category Report and View User Report.

Admin User clicked on **View Category Report** button:
• Run the View Category Report task, then
o Find and display all the Categories, listed in alphabetical order;
o Find and display the total number of items in the category; o Find the minimum Get It Now Price across all items in the category and display under Min Price column.

o Find the maximum Get It Now Price across all items in the category and display under Max Price column.

o Calculate the average Get It Now Price across all items in the category and display it under Average Price column.(If one item doesn't have a Get It Now Price don't include this item in calculation.)

\\aggregation function ignore the null
"SELECT `category`, COUNT(`ItemID`), MIN(`get_it_now_price`), MAX(`get_it_now_price`), AVG(`get_it_now_price`)
FROM `Item`
GROUP BY `category`
ORDER BY `category`
"
When finish, click Done button go back to the Admin **User Main menu** Page.

Abstract Code
Admin User clicked on **View User Report** button:
• Run the View User Report task, then
o Find and display all the Username, listed in total number of listings(descending) order;
o Find and display the total number of items listed by the Username;
o Find the the total number of items sold by the username display under Sold column;
o Find the the total number of items purchased by the username display under Purchased column;
o Find the the total number of items rated by the username display under Rated column.

```
"SELECT  `User`.`username`
        , COUNT( DISTINCT (`Item`.`owner_userID`) ) AS  `Listed`
        , COUNT( DISTINCT (`SoldItem`.`ItemID`) ) AS  `Sold`
        , COUNT( DISTINCT (s2.`Winner`) ) AS  `Purchased`
        , COUNT( DISTINCT (`Rating`.`ItemID`) ) AS  `Rated`
FROM  `User`
        LEFT JOIN  `Item` ON  `User`.`userID` =  `Item`.`owner_userID`
        LEFT JOIN  `SoldItem` ON  `Item`.`owner_userID` =  `SoldItem`.`ItemID`
        LEFT JOIN  `SoldItem` s2 ON s2.`Winner` =  `User`.`username`
        LEFT JOIN  `Rating` ON  `Rating`.`userID` =  `User`.`userID`
        GROUP BY  `User`.`username`
"
```

When finish, click Done button go back to the **Admin User Main menu** Page