Table of Contents

Logging into GTBay

Abstract code

If user clicks ***Register*** button:
    Go to **New User Registration** form
User provides *username* ($username) and *password* ($password)
If user clicks ***Login*** button:

```
SELECT count(username)
FROM RegularUser
WHERE  username='$username'
AND password = '$password';
```

    If number of records found is zero:
        Display error message:
        "Invalid Username & Password combination"
        and return to Login page.
    else:
        Go to **Main Menu** page and initiate session for user.

New User Registration

Abstract code

User clicked ***Register*** button from the **Login** page;
User enters *first name* ($first_name)*, last name* ($last_name)*, username* ($username)*, password* ($password), and *confirm password* ($confirm_password) in the corresponding fields;
If ***cancel*** button is clicked:
    Return to **Login** page ;
If ***Register*** button is clicked:
    If any of the fields is not filled:
        display the error message: "The field cannot be empty";
    If $password != $confirm_password:
        display the error message: "The password is not match";

    Else:
        Write the User table with the information provided;

```
INSERT INTO RegularUser
(username, first_name, last_name, password)
VALUES ('$username', '$first_name', '$last_name', '$password');
```

If $username exists in the database (Duplicate entry error returned):
    Display the error message: “The username has been used”;
    Return to **New User Registration** page;
Else
    Store login information as session variable $username;
    Display “Registration is successful”;
    Go to **Main Menu** page;

Main Menu

Abstract code
User successfully logged in or registered a new account;
Show “***Search for Items***”, “***List an Item for sale***”, and “***View Auction Results***” link;
Determine if the current user is an administrative user;

```
SELECT count(username) FROM AdminUser WHERE username = '$username';
```

If count == 1 (i.e. is admin):
    Display position of the current user;

```
SELECT position FROM AdminUser WHERE username = '$username';
```

    Display “***View Category Report***” and “***View User Report***” links;
Upon:
    User clicks ***Search for Items***: go to **Searching for Items** page;
    User clicks ***List an Item for sale***: go to **Selling Item** page;
    User clicks ***View Auction Results***: go to **Auction Results** page;
    User clicks ***View Category Report***: go to **Category Report** page;
    User clicks ***View User Report***: go to **User Report** page;

Selling Item

Abstract code:
User clicked ***List an Item for sell*** button from the **Main Menu** page;
Display the **New Item for Auction form;**
Create dropdown menu for "*category*";

```
SELECT category FROM Category ORDER BY category;
```

Create dropdown menu for "*Condition*";

```
SELECT condition_name FROM ItemCondition ORDER BY condition_value;
```

User enters *item name* ($item_name)*, description* ($description)*, starting bid* ($starting_bid)*, minimum sale price* ($min_sale_price)*, auction length* ($auction_len)*, and get it now price* ($get_it_now_price) in the corresponding fields, choose *category* ($category) and *condition* ($condition_name) from the dropdown menus, as well as specify *acceptance for return* ($returnable) in the checkbox;
If ***List My Item*** button is clicked:
    If any of the fields is not valid:
        Return to **Selling Item** page with meaningful error messages;
    Else:
        Use $auction_length to calculate auction end time ($auction_end_time);

```
INSERT INTO Item
(item_id, name, description, starting_bid_price, min_sale_price,
get_it_now_price, auction_length, auction_end_time, returnable,
listed_by_user, category, item_condition)
VALUES
($item_id, $item_name, $description, $starting_bid_price,
$min_sale_price, $get_it_now_price, $auction_length,
$auction_end_time, $returnable, $username, $category,
$condition_value);
```

Go to **Main Menu** page;
If ***Cancel*** button is clicked:
Go to **Main Menu** page;

Searching for items

Abstract code
User clicked ***Searching for Items*** link from the **Main Menu** page;
Display the **Item search form**;
Create dropdown menus for "*Category*" and "*Condition at least*":

```
SELECT category FROM Category ORDER BY category;
```

```
SELECT condition_name, condition_value FROM ItemCondition ORDER BY
condition_value;
```

The condition_name will be displayed in the dropdown box whereas condition_value will be mapped to it to be used in search query.
User enters keyword in the "*Keyword*" field ($keyword);
User can optionally select *category* ($category) and *condition* ($condition_value), as well as enter *minimum* ($min_price) and *maximum* ($max_price) prices to narrow down the searching;
If ***Search*** button is clicked:
Go to **Search Results** page
Read the Item and Bid tables to pull out existing non-expired auctions satisfy the specified criteria.
To apply search criteria, build up the WHERE clause of the query depending on whether the input values are blank or not.

```
SELECT
   Item.item_id AS ID,
   name AS 'Item name',
   highest_bid AS 'Current Bid',
   highest_bidder AS 'High Bidder',
   get_it_now_price AS 'Get It Now Price',
```

```
   auction_end_time AS 'Auction Ends'
FROM
   Item
      LEFT JOIN
   (SELECT
      bid_item_id AS item_id,
         MAX(Amount) AS highest_bid,
         bid_by_user AS highest_bidder
   FROM
      Bid
   GROUP BY bid_item_id) RES1 ON Item.item_id = RES1.item_id
WHERE
   auction_end_time > NOW() AND
(name LIKE '%$keyword%' OR description LIKE '%$keyword%')
```

If *$category* is not empty, append to the above WHERE clause:

```
AND (category = '$category')
```

If *$condition_name* is not empty, append to the above query:

```
AND (item_condition >= '$condition_value')
```

If *$max_price* is not empty, append to the above query:

```
AND ( (highest_bid IS NOT NULL AND highest_bid <= $max_price) OR
        (highest_bid IS NULL AND starting_bid_price <= $max_price) )
```

If *$min_price* is not empty, append to the above query:

```
AND ( (highest_bid IS NOT NULL AND highest_bid >= $min_price) OR
        (highest_bid IS NULL AND starting_bid_price >= $min_price) )
```

Append the ORDER BY clause in the end to sort by auction_end_time, and complete the query by appending a semicolon ; and execute it.

```
ORDER BY Item.auction_end_time ASC;
```

Display the results of the query on the Search Results page with the following information:

Item ID, item name, highest bid, highest bidder, get it now price, and the auction end time

If ***Back to Search*** button is clicked:

Return to **Searching for items** page

If user clicks on the ***Item Name***:

Save the item_id ($item_id) associated with the clicked item

Go to **Item Description** page

If ***cancel*** button is clicked:

Return to **Main Menu** page;

View Item

Abstract code

User clicked the ***item name*** link on the **Search Result** page

Display ***View Rating*** link

Display the item's ID, name, description, category, condition, and return acceptance

```
SELECT
   item_id,
   name,
   description,
   category,
   condition_name,
   returnable,
   get_it_now_price,
   auction_end_time
FROM
   Item
```

```
    JOIN
  ItemCondition ON item_condition = condition_value
WHERE
  item_id = '$item_id';
```

If get_it_now_price != Null:
    Display get it now price of the item and ***Get it now*** button
Display the latest four bids on this item

```
SELECT
  amount AS 'Bid Amount',
  timestamp AS 'Time of Bid',
  bid_by_user AS 'Username'
FROM
  Bid
WHERE
  bid_item_id = '$item_id'
ORDER BY timestamp DESC
LIMIT 4;
```

Check if the current user is the listing user

```
SELECT listed_by_user FROM Item WHERE item_id = '$item_id'
```

If the item was listed by the current user:
    Display ***Edit Description*** link
    If the user clicks ***Edit Description*** link:
        Display a popup window asking for new *description*
        User enters new *description* ($new_description)
        If user clicks ***Submit*** button:
            Write the Item table with new description

```
UPDATE Item
SET description = '$new_description'
WHERE item_id = '$item_id'
```

Return to **Item Description** page

If user clicks ***Cancel*** button:

Return to **Item Description** page

If the user clicks ***Cancel*** button:

Return to **Search Results** page

If the user clicks ***View Ratings*** link:

Go to **Item Ratings** page

If the user clicks ***Get it now*** button:

Update the auction_end_time, winner, and final_sale_price

```
UPDATE Item
SET auction_end_time = timestamp, winner = '$username',
    final_sale_price = get_it_now_price
WHERE item_id = '$item_id'
```

Display appropriate message and return to **Search Results** page

If user enters a bid ($amount) and clicks ***Bid On This Item*** button:

If current time passes auction_end_time:

Display error message: “Auction has ended”

Else:

Get latest value for current highest bid for item again in case it has changed during the time that user viewed the page.

```
SELECT bid_item_id AS item_id,
       MAX(Amount) AS current_highest_bid
FROM Bid where bid_item_id = '$item_id';
```

If $amount >= current highest bid + 1 and $bid < get_it_now_price:

Write to the Bid table

```
INSERT INTO Bid (bid_item_id, bid_by_user, amount,
timestamp)
VALUES ('$item_id', '$user_id', '$amount',
'$timestamp')
```

Display successful message

Return to **Search Results** page

Else if $amount > get_it_now_price:

Display popup window with error message suggesting to click ***Get it now*** button
Return to **Item Description** page

Item Ratings

Abstract code
User clicked ***View Rating*** link on **Item Description** page
Display the id and name of the item

```
SELECT item_id, name
FROM Item
WHERE item_id = '$item_id'
```

Display average rating of the item and all associated ratings sorted by date (latest on top)

```
SELECT rated_by_user, stars, comment, timestamp, AVG(stars) FROM Rating
WHERE rated_item_id = '$item_id' ORDER BY timestamp DESC
```

If the rating was created by the current user (rated_by_user == $username):
    Display ***Delete My Rating*** link
    If user click ***Delete My Rating*** link:
    This rating is deleted from the rating table

```
DELETE FROM Rating WHERE
rated_item_id = '$item_id' AND rated_by_user = '$username';
```

    Refresh the **Item Ratings** page
    If the user clicks ***Cancel*** button:
        Return to **Item Description** page
Display the new rating form
User fills out the rating form

If the user clicks "**Rate This Item**" button:

Check if the current user has already rated this item before:

```
SELECT COUNT(rated_by_user) FROM Ratings WHERE rated_by_user =
'$username';
```

If count == 1:

Display error message and return to **Item Rating** page

Else if all fields except comment are entered:

Write to the Rating table

```
INSERT INTO Rating
VALUES ('$item_id', '$username', '$comments', '$stars',
'$current_time');
```

Return to **Item Rating** page

Else:

Display error message

Auction Results

Abstract code

User clicked "**Show Auction Result**" link from **Main Menu** page

Read the Item table and pull out all items that has ended already

If a winner has not been determined yet, then:

Update the Item table with the user who placed the highest bid as the winner.

Update the Item table with the highest bid price as the final sale price

Sort the items by their auction ended date and time (latest on top)

For each item:

Display the following details about the items:

Item ID, Item name, Sale price, Winner, and Auction ended date and time

If user clicks "**Done**" button:

Return to **Main Menu**

```
//Update the winner attribute and final_sale_price in the Item table for all the
items whose auction has ended and has at least one bid whose bid amount is
greater than or equal to minimum sale price.

UPDATE Item mainItem
      INNER JOIN
   (SELECT
      ANS.bid_item, ANS.max_amount, B.bid_by_user AS winner
   FROM
      (SELECT
      I.item_id AS bid_item, MAX(B.amount) AS max_amount
   FROM
      Item I
   INNER JOIN Bid B ON I.item_id = B.bid_item_id
      AND I.winner IS NULL
      AND I.auction_end_time < NOW()
      AND B.amount >= I.min_sale_price
   GROUP BY I.item_id) ANS, Bid B
   WHERE
      B.bid_item_id = ANS.bid_item
         AND B.amount = ANS.max_amount) ANS2 ON mainItem.item_id =
ANS2.bid_item
SET
   mainItem.winner = ANS2.winner,
   mainItem.final_sale_price = ANS2.max_amount
WHERE
   mainItem.item_id = ANS2.bid_item;

//Display Auction Results

SELECT
   item_id AS ID,
   name AS 'Item Name',
   IFNULL(final_sale_price, '-') AS 'Sale Price',
   IFNULL(winner, '-') AS Winner,
   auction_end_time AS 'Auction ended'
FROM
```

```
   Item
WHERE
   Item.auction_end_time < NOW()
ORDER BY auction_end_time DESC;
```

GTBay Administrative Reports

Category Report

Abstract Code

Administrative user clicks ***View Category Report*** from the **Main Menu** page
Display total number of items and minimum, maximum, and average of get_it_now_price for each category of items, items without get_it_now_price are excluded from calculation of min, max, and average.

```
SELECT *
FROM
  (SELECT
  category AS `Category`, COUNT(item_id) AS `Total Items`
  FROM Item
  GROUP BY category) RESULT1
LEFT JOIN
  (SELECT
   AVG(get_it_now_price) AS `Average Price`,
   MIN(get_it_now_price) AS `Min Price`,
   MAX(get_it_now_price) AS `Max Price`,
   category AS `Category`
  FROM Item
  WHERE
  get_it_now_price IS NOT NULL
  GROUP BY category) RESULT2
ON RESULT1.Category = RESULT2.Category
ORDER BY category ASC;
```

If administrative use clicks ***Done*** button:
Return to **Main Menu** page

User Report

Abstract Code
Administrative user clicks ***View User Report*** from the **Main Menu** page
Run **Auction Results** and store it in a separate "Ended Item" table

```
SELECT
  *
FROM
  (SELECT
    listed_by_user AS Username,
      COUNT(listed_by_user) AS Listed,
      COUNT(final_sale_price) AS Sold
  FROM
    Item
  GROUP BY listed_by_user) SOLD
    NATURAL JOIN
  (SELECT
    winner AS Username, COUNT(winner) AS Purchased
  FROM
    Item
  WHERE
    winner IS NOT NULL
  GROUP BY winner) PURCHASED
    NATURAL JOIN
  (SELECT
    rated_by_user AS Username, COUNT(rated_by_user) AS Rated
  FROM
    Rating
  GROUP BY rated_by_user) Ratings;
```

If administrative user clicks ***Done***:
Return to **Main Menu** page