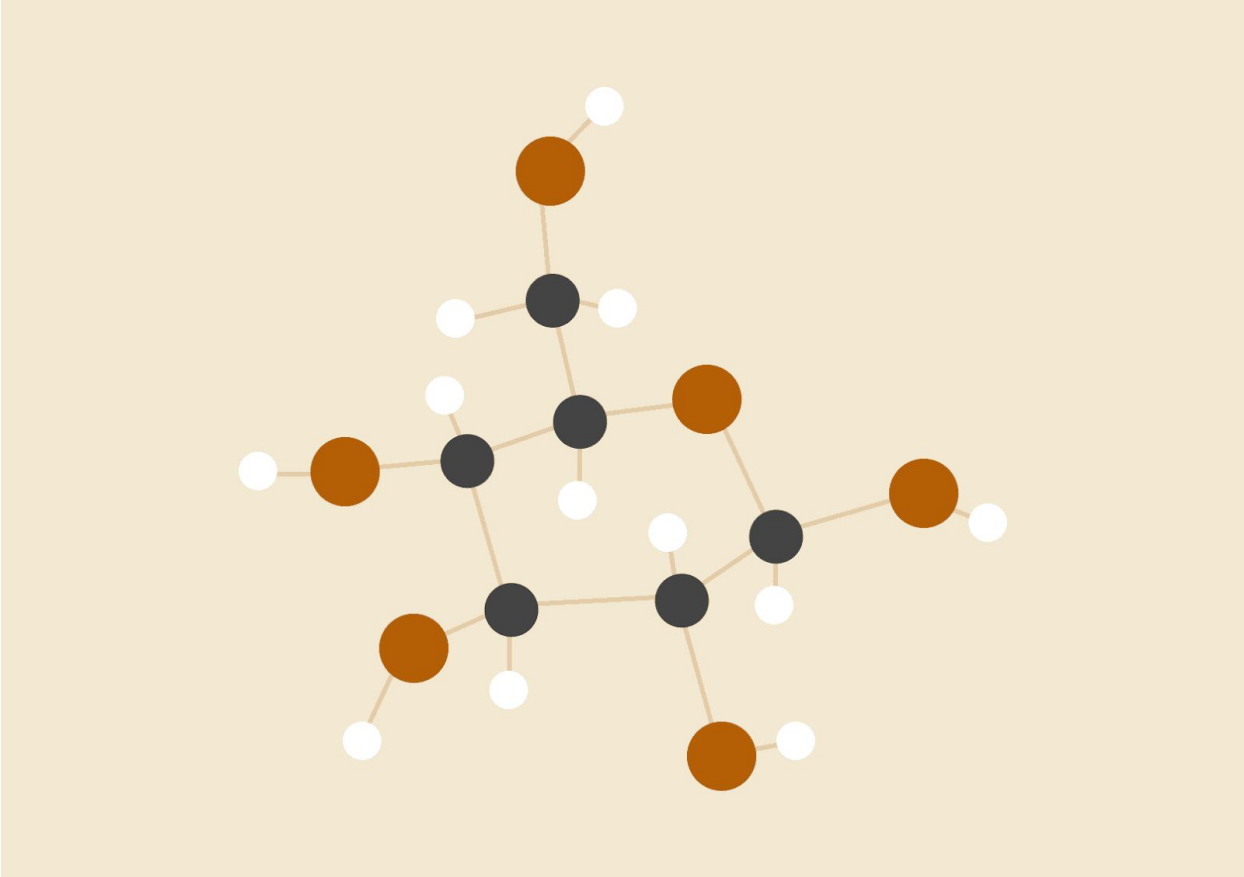
Phase 2: Abstract Code + SQL

CS6400: Team 105



Jung, Sungjin; sjung93@gatech.edu
Kazkayasi, Mehmet O; mkazkayasi3@gatech.edu
Katnis, Nichole Marlene; nkatnis3@gatech.edu

March 4, 2018
Online Master of Science in Computer Science
Georgia Tech

TABLE OF CONTENTS

Login

Abstract Code

- User enters *username* (‘$Username’), *password* (‘$Password’) input fields.
- If the data validation is successful for both *username* and *password* input fields, then:
    - When ***Login*** button is clicked:

```
SELECT Password FROM User WHERE Username = ‘$Username’;
```

        - If User record is found but User.password != '$Password':
            - Go back to **Login** form, with error message.
        - If User record did not found: Go back to **Login** form, with error message.
        - Else:
            - Store login information as session variable ‘$Username’
            - Go to **Main Menu**
    - Upon click ***Register*** button - Jump to **Register** task
- Else *username* or *password* inputs are invalid, display **Login** form, with error message.

Register

Abstract Code

- User enters *First Name* (‘$FName’), *Last Name* (‘$LName’), *Username* (‘$Username’), *Password* (‘$Password’) and *Confirm Password* (‘$CPassword’) input fields.
- When ***Register*** button is clicked:
    - If any of the fields is not filled in: Display **Register** form, with error message.
    - If ‘$Password’ != ‘$CPassword’: Display **Register** form, with error message.
    - If ‘$Username’ is used by another User: Display **Register** form, with error message.
    - Else, store new user information in User table:

```
INSERT INTO User (Username, Password, First_Name, Last_Name)
VALUES          ('$Username', '$Password', '$FName', '$LName');
```

    - Go to **Log In** form
- When ***Cancel*** button is clicked: Display **Login** form.

New Item For Auction

Abstract Code

- Get categories from Category.Category_Name values in dropdown list.

```
SELECT DISTINCT Category_Name FROM Category;
```

- User:
  - Enters *Item Name* (‘$ItemName’), *Description*(‘$Description’), *Start Auction Bidding* (‘$AuctionStartPrice’), *Minimum Sale Price* (‘$MinSalePrice’), *Minimum Sale Price* (‘$AuctionStartPrice’), and optionally *Get It Now Price* (‘$GetItNowPrice’).
  - Selects *Category* (‘$Category’), *Condition* (‘$Condition’), *Auction Length* (‘$AuctionLength’) input fields from dropdown lists,
  - Chooses to click or not on *Returns Accepted* (‘$ReturnsAccepted’) field.
- When ***List My Item*** button is clicked:
  - If any of the fields except *Get It Now Price* field is not filled in: Display **New Item for Auction** form, with error message.
  - If ‘$GetItNowPrice’ < ‘$MinSalePrice’: Display **New Item for Auction** form, with error message.
  - Calculate (‘$AuctionEndDate’) using Current date and time along with ‘$AuctionLength’
  - If *Get It Now Price* (‘$GetItNowPrice’) is not filled, assign NULL.
  - Store new item information in Item table

```
@New_Item_ID = auto-generated-ID

INSERT INTO Item (ItemID, Item_Name, Description, Cond,
            Returnable, Auction_Start_Datetime, Auction_Length,
            Min_Sale_Price, Get_It_Now_Price,
            Auction_End_Datetime, Category, Lister_Name,
            Starting_Bid)
VALUES     (@New_Item_ID, ‘$ItemName’, ‘$Description’,
            ‘$Condition’, ‘$ReturnsAccepted’, NOW(),
            ‘$AuctionLength’,‘$MinSalePrice’, ‘$GetItNowPrice’,
             NOW() + ‘$AuctionLength’, ‘$Category’, ‘$Username’,
            ‘$AuctionStartPrice’);
```

  - Display **Main Menu** form.
- When ***Cancel*** button is clicked: Display **Main Menu** form.

Item Search

Abstract Code

- Get Category.Category_Name values along with **"Any"** option in dropdown list.

```
SELECT DISTINCT Category_Name FROM Category;
```

- User:
  - Enters *Keyword*('$Keyword'),
  - Optionally user selects: *Minimum Price*('$MinPrice'), *Maximum Price* ('$MaxPrice'), *Category* ('$Category') and *Condition At Least* ('$ConditionAtLeast')
  - If not selected:
    - $MinPrice = 0
    - $MaxPrice = Inf
    - $ConditionAtLeast = 0
    - $Category = "All"
- When ***Search*** button is clicked:
  - Find Items that have Auction_End_Datetime < the current time in the selected category which has '$Keyword' in '$ItemName' or '$Description'
  - Optional search criteria used to further filter results
    - that have a current high Bid of at least '$MinPrice' and at most '$MaxPrice'
    - better '$Condition' than '$ConditionAtLeast'
  - For each Item:
    - Get Item.ItemId, Item.ItemName, Item.GetItNowPrice, Item.AuctionEndTime
    - Get Bid with max Bid.Time associated with Item and as Current Bid,
    - Get Bid Owner associated with Current bid as High Bidder
  - Display form **Search Results**
  - When any ***Item Name*** button is clicked: Display **Item for Sale** form.
- When ***Back to Search*** button is clicked: Display **Item Search** form.

```
SELECT  Item_ID
      , Item_Name
      , Highest_Bid
      , Current_Bidder
      , Get_It_Now_Price
      , Auction_End_Datetime
FROM
       (SELECT
              ITEM.Item_ID
            , ITEM.Item_Name
            , Q.Highest_Bid
            , Q.Current_Bidder
            , ITEM.Get_It_Now_Price
            , ITEM.Auction_End_Datetime
            , ITEM.Cond
            , ITEM.Description
            , ITEM.Category
        FROM ITEM
        LEFT JOIN (SELECT Item_ID
                        , Bid_Amount AS Highest_Bid
                        , Username AS Current_Bidder
                   FROM   BID
                  WHERE (Item_ID, Bid_Amount) IN (SELECT  Item_ID,
                                                          MAX(Bid_Amount) AS
                                                                  MaxPrice
                                                  FROM  BID
                                                  GROUP BY  Item_ID)) AS Q
          ON Q.Item_ID = ITEM.Item_ID) AS P
WHERE NOW() < P.Auction_End_Datetime
      AND '$ConditionAtLeast'  <= P.Cond
      AND (P.Highest_Bid IS NULL
      OR  ('$MaxPrice' > P.Highest_Bid AND '$MinPrice' < P.Highest_Bid))
      AND (P.Name LIKE '%$Keyword%' OR P.Description LIKE '%$Keyword%')
      AND ('$Category' = P.Category OR '$Category' = "All")


ORDER BY P.Auction_End_Datetime;
```

Item for Sale

Abstract Code

- Get Item.ItemName, Item.ItemID, Item.Description, Item.AuctionEndDate, Category.Category_name, and Item.Returnable that Item is related, Item.Cond, Item.GetItNowPrice (if not NULL), get top 4 bids with bids sorted by descending Bid.Time that Item is related as well as usernames associated with the Bid

```
/***Item information***/
SELECT
      Item_ID
    , Item_Name
    , Lister_Name
    , Description
    , Auction_End_Datetime
    , Returnable
    , Category
    , Cond
    , Get_It_Now_Price
FROM  Item
WHERE Item_ID = '$Item_ID';


/***Latest Bids Section***/
SELECT
      Bid_Amount
    , Bid_Datetime
    , Username
FROM  Bid
WHERE Item_ID = '$Item_ID'
ORDER BY Bid_Datetime DESC
LIMIT 4;
```

- Show Edit Description button if '$UserID' belongs to the seller.
- Calculate ('$MinBid'), by adding one dollar on the highest bid.
- Show **Item for Sale** form
- Upon clicked ***Get It Now*** button:
    - Set Get It Now time variable (@gin_time) to be current time
    - Update Item.Auction_End_Datetime to be @gin_time
    - Insert Get_It_Now information as a record in Bid
    - Set
        - Bid.Username = '$Username'
        - Bid.Bid_Amount = Item.Get_It_Now_Price
        - Bid.Bit_Datetime = @gin_time

```
SET @gin_time = NOW()

SET @get_it_now = SELECT Get_It_Now_Price
                  FROM   Item
                  WHERE  Item_ID = '$Item_ID'


/***Update Item to update auction end***/
UPDATE Item SET Auction_End_Datetime = @gin_time
WHERE Item_ID = '$Item_ID';


/***Update Bid to store winning "bid" information***/
INSERT INTO Bid (Bid_Datetime, Username, Item_ID, Bid_Amount)
VALUES (@gin_time, '$Username', '$Item_ID', @get_it_now)
```

- Upon ***View Ratings*** button is clicked: Display form **Item Ratings**
- User can choose to bid on item. So, User enters Your Bid ('$YourBid'). Upon clicked ***Bid On This Item*** button:
    - If '$YourBid' is not populated: Display **Item For Sale** form with error message.
    - If '$YourBid' is not larger than one dollar added to highest bid Item has: Display **Item For Sale** form with error message.
    - If '$YourBid' is greater than or equal to the Item.Get_It_Now_Price: Display **Item For Sale** form with error message.
    - Else, create a new bid for the item. Assign Bid.Bid_Amount = '$YourBid' and Bid.Bid_Datetime = Current Time.

```
INSERT INTO Bid (Bid_Datetime, Username, Item_ID, Bid_Amount)
VALUES          (NOW(), '$Username', '$Item_ID', '$YourBid')
```

    - Display **Item For Sale** form.
- Upon ***Edit Description*** button is clicked: Enable textbox containing Item.Description for edit.
    - Save entered text as '$NewDescription'

```
UPDATE Item SET Description = '$NewDescription'
WHERE  Item_ID = '$Item_ID';
```

- When ***Cancel*** button is clicked: Display **Search Results** form.

Item Ratings

Abstract Code

- Find and display ratings (if any) for the Item_ID along with the Item.ItemName, Item.ItemID
- Get ratings that are associated with the Item and calculate the average rating
- Get all Rating information related to the Item: Rating.Rating_Datetime, Rating.Comment, and Rating.Number_Of_Stars sorted by descending Rating.Rating_Datetime

```
/***Get Each Comment For The Selected Item***/
SELECT
      Item.Item_ID
    , Item_Name
    , Rating_Datetime
    , Username
    , Number_Of_Stars
    , Comment
FROM  Item  LEFT JOIN Rating
            ON Item.Item_ID = Rating.Item_ID
WHERE Item.Item_ID = '$Item_ID'
ORDER BY Rating_Datetime DESC;

/***Get Average Number Of Stars The Item Is Rated***/
SELECT AVG(Number_Of_Stars)
FROM   Rating
WHERE  Item_ID = '$Item_ID';
```

- Check '$UserID' has a Rating and if it has: Display ***Delete My Rating*** button on '$UserID's rating
- Upon ***Delete My Rating*** button is clicked: Remove the Rating belongs to '$UserID'

```
DELETE FROM Rating WHERE  Username = '$Username';
```

- User fills Comments ('$Comment') and selects Number_Of_Starts ('$Stars').
- Upon ***Rate This Item*** button is clicked:
    - Store Rating.Rating_TimeDate as Current time, Rating.Comment, and Rating.Number_Of_Stars in Rating table.

```
INSERT INTO Rating
VALUES (NOW(), '$Username', '$Item_ID', '$Stars', '$Comment')
```

    - Display **Item Ratings** form.
- Upon ***Cancel*** button is clicked: Display **Item for Sale** form.

Auction Results

Abstract Code

- Fetch Items that Item.AuctionEndDate has passed.
- Display Item.ItemID, Item.ItemName
- Get Bid with max Bid.Time associated with Item and as Current Bid,
- Get Bid Owner associated with Current bid as High Bidder
- Display **Auction Results** sorted by descending AuctionEndDate
- Upon ***Done*** button is clicked: Display **Item for Sale** form.

```
SELECT
       Item_ID
      ,Item_Name
      ,COALESCE(Highest_Bid, '-') AS Highest_Bid
      ,COALESCE(Current_Bidder, '-') AS Current_Bidder
      ,Auction_End_Datetime
      ,Min_Sale_Price
FROM (SELECT ITEM.Item_ID
            ,ITEM.Item_Name
            ,Q.Highest_Bid
            ,Q.Current_Bidder
            ,ITEM.Auction_End_Datetime
            ,ITEM.Min_Sale_Price
      FROM   ITEM
             LEFT JOIN
              (SELECT  DISTINCT Bid.Item_ID
                              , Bid_Amount AS Highest_Bid
                              , Username AS Current_Bidder
               FROM   BID, ITEM
               WHERE Item.Min_Sale_Price <= Bid.Bid_Amount
                     AND Bid.Item_ID = Item.Item_ID
                     AND (Bid.Item_ID, Bid_Amount)
                     IN (SELECT Bid.Item_ID, MAX(Bid_Amount) AS MaxPrice
                         FROM  BID GROUP BY  Bid.Item_ID)) AS Q
              ON Q.Item_ID = ITEM.Item_ID) AS P
WHERE NOW() > P.Auction_End_Datetime
ORDER BY P.Auction_End_Datetime DESC;
```

Category Report

Abstract Code

- Fetch Categories and sort by Category.CategoryName
- Aggregate and display Category.Listed, Category.MinPrice, Category.Tota_lItems, and Category.Max_Price, Category.Average_Price.

```
SELECT   Category,
         Count(*) AS Total_Items,
         MIN(Get_It_Now_Price) AS Min_Price,
         MAX(Get_It_Now_Price) AS Max_Price,
         AVG(Get_It_Now_Price) AS Average_Price
FROM     Item
GROUP BY Category
```

- Upon clicked Done button: Display **Main Menu** form.

User Report

Abstract Code

- Fetch Users and sort by User.Listed.
- Aggregate and display User.Listed, User.Purchased, User.Sold, and User.Rated values.

```
/***Listed***/
SELECT *
FROM (SELECT Lister_Name, COUNT(*) AS Listed
      FROM Item
      GROUP BY Lister_Name) AS Q
/***Sold***/
SELECT Lister_Name, COUNT(Item_ID) AS Sold
FROM (SELECT DISTINCT Item.Lister_Name, ITEM.Item_ID
      FROM   Bid, Item
      WHERE  NOW() > Item.Auction_End_Datetime AND
             Bid.Bid_Amount > Item.Min_Sale_Price AND
             Bid.Item_ID = Item.Item_ID) AS P
GROUP BY Lister_Name

/***Purchased***/
SELECT Current_Bidder, COUNT(*)
FROM   (SELECT Item.Item_ID, Item_Name, Q.Highest_Bid, Current_Bidder
        FROM ITEM JOIN
             (SELECT DISTINCT Bid.Item_ID,
                              Bid_Amount AS Highest_Bid,
                              Username AS Current_Bidder
              FROM   BID, ITEM
              WHERE Item.Min_Sale_Price <= Bid.Bid_Amount
                    AND Bid.Item_ID = Item.Item_ID
                    AND (Bid.Item_ID, Bid_Amount)
                    IN(SELECT Bid.Item_ID,MAX(Bid_Amount)AS MaxPrice
                       FROM  BID GROUP BY  Bid.Item_ID)) AS Q
                       WHERE Item.Item_ID = Q.Item_ID AND
                              NOW() > Item.Auction_End_Datetime) AS P
GROUP BY Current_Bidder

/***Rated***/
SELECT   Username, COUNT(Item_ID) AS Rated
FROM     (SELECT DISTINCT Rating.Username, ITEM.Item_ID
          FROM   Rating, Item
          WHERE  Rating.Item_ID = Item.Item_ID) AS P
GROUP BY Username
```

- Upon clicked Done button: Display **Main Menu** form.