Login

Shailesh

Abstract Code

- User enters email/nickname($UserID) and password($Password) and clicks on the Login button.
- User credentials check task

```
SELECT email from TradePlazaUser where (TradePlazaUser.email='$UserID'  or TradePlazaUser.nickname='$UserID') and TradePlazaUser.password='$Password';
```

  - If an empty row is returned through the query, display an error message in the screen stating as an invalid login credentials.
  - If a user row is returned, store the returned email address in variable '$UserID' and navigate the user to Main Menu.

Registration

Shailesh

Abstract Code

- User enters email($UserID), password($Password), nickname($NickName), firstname($FirstName), lastname($LastName) and postalcode($PostalCode) and clicks on the Register button.
- New user registration credentials check task

```
INSERT INTO TradePlazaUser (email, password, nickname, first_name, last_name, FK_Address_PostalCode) VALUES ($UserID, $Password), $NickName, $FirstName, $LastName, $PostalCode));
```

  - If an error is returned from the query, display an error message in the screen stating as an invalid registration credentials.
  - If a user row is returned, store the returned email address in variable '$UserID' and navigate the user to Main Menu.

Main Menu

Shailesh

Abstract Code

- Fetch and display firstname, lastname and nickname of user using $UserID.
- User Details fetch task

```
SELECT first_name, last_name, nickname from `User` where (User.email='$UserID'  or
User.nickname='$UserID');
```

- Display firstname, lastname and nickname of the user in the main menu.

- Fetch average response time for accepting/rejecting a trade by the user.
- Fetch average response time task

```
Select ROUND(avg(TIMESTAMPDIFF(DAY,accept_reject_date,NOW())),1) from Trade Inner Join
(
Select Item.item_number, ItemJoin.email from Item
NATURAL JOIN (
Select BoardGame.item_number, BoardGame.email from BoardGame
UNION
Select PlayingCardGame.item_number, PlayingCardGame.email from PlayingCardGame
UNION
Select CollectibleCardGame.item_number, CollectibleCardGame.email from CollectibleCardGame
UNION
Select ComputerGame.item_number, ComputerGame.email from ComputerGame
UNION
Select VideoGame.item_number, VideoGame.email from VideoGame
) AS ItemJoin where '$UserID' = ItemJoin.email
)
AS TradeJoin ON Trade.counter_party_item_number= TradeJoin.item_number
where Trade.trade_status='Accepted' or Trade.trade_status='Rejected'
GROUP BY TradeJoin.email;
```

- If nothing gets returned, display None with black text.
- If data is returned, display text with corresponding color.

- Fetch number of unaccepted trades by the user.
- Fetch number of unaccepted trades task

```
Select count(*) from
Trade Inner Join
(
Select Item.item_number, ItemJoin.email from Item
NATURAL JOIN (
Select BoardGame.item_number, BoardGame.email from BoardGame
UNION
Select PlayingCardGame.item_number, PlayingCardGame.email from PlayingCardGame
UNION
Select CollectibleCardGame.item_number, CollectibleCardGame.email from CollectibleCardGame
UNION
Select ComputerGame.item_number, ComputerGame.email from ComputerGame
```

```
UNION
Select VideoGame.item_number, VideoGame.email from VideoGame
) AS ItemJoin where '$UserID' = ItemJoin.email
)
AS TradeJoin ON Trade.counter_party_item_number= TradeJoin.item_number
where Trade.trade_status='Proposed'
GROUP BY TradeJoin.email;
```

- If nothing gets returned, display 0 without any links.
- If greater than 0 but less than 2, provide link to Accept/Reject form.
- If greater or equal to 2, change color to red(underlined) and provide link to Accept/Reject form.

- Fetch current trade rank of the user.
- Fetch current trade rank task

```
Select count(*) from Trade Inner Join
(
Select Item.item_number, ItemJoin.email from Item
NATURAL JOIN (
Select BoardGame.item_number, BoardGame.email from BoardGame
UNION
Select PlayingCardGame.item_number, PlayingCardGame.email from PlayingCardGame
UNION
Select CollectibleCardGame.item_number, CollectibleCardGame.email from CollectibleCardGame
UNION
Select ComputerGame.item_number, ComputerGame.email from ComputerGame
UNION
Select VideoGame.item_number, VideoGame.email from VideoGame
) AS ItemJoin where '$UserID' = ItemJoin.email
)
AS TradeJoin ON Trade.counter_party_item_number= TradeJoin.item_number
Inner Join ON Trade.proposer_item_number= TradeJoin.item_number
where Trade.trade_status='Accepted'
GROUP BY TradeJoin.email;
```

- If nothing gets returned, display None.
- Display the badge according to the count returned.

- User clicks on the logout Button, User is directed to the login screen.
- User clicks on the List Item Button, User is directed to Listing Items Page.
- User clicks on the My Items Button, User is directed to the My Items Page.
- User clicks on the Search Items Button, User is directed to the Search Items Page.
- User clicks on the Trade History Button, User is directed to the Trade History Page.

Propose Trade

Abstract Code

- If the counterparty distance exceeds 100.0, a warning message with the distance displayed at the top should be displayed.

```
SELECT proposer_item_number, counter_party_item_number, (3958.75 * 2
* POWER(ATAN(SQRT((POWER(SIN((offered_address.Latitude -
my_address.Latitude) / 2), 2) + COS(my_address.Latitude) *
COS(offered_address.Latitude) * POWER(SIN((offered_address.Longitude
- my_address.Longitude) / 2), 2))), SQRT(1 -
(POWER(SIN((offered_address.Latitude - my_address.Latitude) / 2), 2)
+ COS(my_address.Latitude) * COS(offered_address.Latitude) *
POWER(SIN((offered_address.Longitude - my_address.Longitude) / 2),
2)))), 2)) as distance
FROM Trade INNER JOIN
        (SELECT item_number, email FROM BoardGame UNION
                SELECT item_number, email FROM CollectibleCardGame
UNION
                SELECT item_number, email FROM ComputerGame UNION
                SELECT item_number, email FROM PlayingCardGame UNION
                SELECT item_number, email FROM VideoGame
        ) AS offered_item ON proposer_item_number =
offered_item.item_number INNER JOIN
        tradeplazauser as offered_user ON offered_user.email =
offered_item.email INNER JOIN
        address as offered_address ON offered_user.postal_code =
offered_address.postal_code INNER JOIN
        (SELECT item_number, email FROM BoardGame UNION
                SELECT item_number, email FROM CollectibleCardGame
UNION
                SELECT item_number, email FROM ComputerGame UNION
                SELECT item_number, email FROM PlayingCardGame UNION
                SELECT item_number, email FROM VideoGame
        ) AS my_item ON counter_party_item_number =
my_item.item_number INNER JOIN
        tradeplazauser as my_user ON my_user.email = my_item.email
INNER JOIN
```

```
    address as my_address ON my_user.postal_code =
my_address.postal_code
WHERE counter_party_item_number IS NULL
```

- Only items that are available for trading must be displayed (self).
- Provide an appropriate mechanism for selecting the desired item.
- Should be displayed Number of items, game type, title, and condition
- Items should by ItemNumber.

```
SELECT item_number, title, game_condition, game_type
FROM (
                SELECT item_number, title, game_condition, 'Board
Game' AS game_type FROM BoardGame UNION
                SELECT item_number, title, game_condition, 'Playing
Cards' as game_type FROM PlayingCardGame UNION
                SELECT item_number, title, game_condition,
'Collectible Card Game' AS game_type FROM CollectibleCardGame UNION
                SELECT item_number, title, game_condition, 'Video
Game' AS game_type FROM VideoGame UNION
                SELECT item_number, title, game_condition, 'Computer
Game' AS game_type FROM ComputerGame
        ) as all_games NATURAL JOIN tradeplazauser
WHERE (email = "$UserId" OR nickname = "nickname_100") AND
        item_number NOT IN (
                SELECT proposer_item_number as item_number FROM Trade
WHERE trade_status = "ACCEPT" OR trade_status = "REJECT"
        )
ORDER BY item_number ASC;
```

- Once the proposed item has been selected, the user must be given a confirm
- Button to confirm the proposal.
- Once confirmed, the proposed date is saved/written into the Trade table.

```
UPDATE Trade SET counter_party_item_number = "$CounterItemNumber"
WHERE proposer_item_number = "$ProposeItemNumber" AND
counter_party_item_number IS NULL
```

- The user should be able to return to the main menu after receiving a confirmation message

Accept/Reject Trade

Abstract Code

- Results of a Search
- List proposed trades in which the user is the counterparty and can accept or reject a trade. For each proposal show the date proposed, the desired item's title, the proposer's nickname, their rank, distance from the user (rounded to the hundredths), and the proposed item title, all ordered by proposal date.

```
SELECT proposed_date, my_item.title, my_item.item_number,
my_user.nickname, offered_user.nickname, offered_item.title,
offered_item.item_number, (3958.75 * 2 *
POWER(ATAN(SQRT((POWER(SIN((offered_address.Latitude -
my_address.Latitude) / 2), 2) + COS(my_address.Latitude) *
COS(offered_address.Latitude) * POWER(SIN((offered_address.Longitude
- my_address.Longitude) / 2), 2))), SQRT(1 -
(POWER(SIN((offered_address.Latitude - my_address.Latitude) / 2), 2)
+ COS(my_address.Latitude) * COS(offered_address.Latitude) *
POWER(SIN((offered_address.Longitude - my_address.Longitude) / 2),
2)))), 2)) as distance

FROM Trade INNER JOIN

        (SELECT item_number, email, title FROM BoardGame UNION

                SELECT item_number, email, title FROM
CollectibleCardGame UNION

                SELECT item_number, email, title FROM ComputerGame
UNION

                SELECT item_number, email, title FROM
PlayingCardGame UNION

                SELECT item_number, email, title FROM VideoGame

        ) AS offered_item ON proposer_item_number =
offered_item.item_number INNER JOIN

        tradeplazauser as offered_user ON offered_user.email =
offered_item.email INNER JOIN

        address as offered_address ON offered_user.postal_code =
offered_address.postal_code INNER JOIN

        (SELECT item_number, email, title FROM BoardGame UNION

                SELECT item_number, email, title FROM
CollectibleCardGame UNION

                SELECT item_number, email, title FROM ComputerGame
UNION

                SELECT item_number, email, title FROM
PlayingCardGame UNION

                SELECT item_number, email, title FROM VideoGame

        ) AS my_item ON counter_party_item_number =
my_item.item_number INNER JOIN

        tradeplazauser as my_user ON my_user.email = my_item.email
INNER JOIN

    address as my_address ON my_user.postal_code =
my_address.postal_code

WHERE my_user.email = "$UserId" or my_user.nickname = "nickname_1"
AND trade_status = "PENDING"
```

- Both item fields should contain a link to the item's detail page.

- A mechanism for accepting/rejecting should be provided: Accept, Reject
- If the trade is accepted, query the USER table and display a dialog containing the proposer's email address and first name.
- The trade's acceptance or rejection date is recorded in the database as part of the trade table
- If the proposed trade is turned down, a new trade for the same proposed item and the same desired item cannot be proposed.

```
UPDATE Trade SET trade_status = "$ACCEPT/REJECT", accept_reject_date
= NOW() WHERE proposer_item_number = "$ProposeItemNumber" AND
counter_party_item_number = "$CounterItemNumber"
```

- Accepting a trade removes the item from the listing (Search and Display Search Results), and the user should be returned to the main menu if no more trades need to be accepted or rejected.

List Item

Abstract Code

- Check if PendingUserTrades is greater than 2

```
SELECT COUNT(item_number) FROM (SELECT item_number FROM (
SELECT item_number FROM BoardGame WHERE owner_email='$UserID'
UNION
SELECT item_number FROM PlayingCardGame WHERE owner_email='$UserID'
UNION
SELECT item_number FROM CollectibleCardGame WHERE owner_email='$UserID'
UNION
SELECT item_number FROM ComputerGame WHERE owner_email='$UserID'
UNION
SELECT item_number FROM VideoGame WHERE owner_email= '$UserID') AS UserItems
INNER JOIN (SELECT counter_party_item_number FROM Trade WHERE trade_status =
"PENDING") AS PendingTrades) AS PendingUserTrades;
```

- If user has > 2 unaccepted trades Then display error "Cannot list item. Check pending trades"
    - Back to ***Main Menu*** button
- Else:
    - Show **New Item Listing** form
    - Show empty title textbox
    - Show empty description textbox
    - Game type dropdown is populated with game type ENUM
    - Condition dropdown is populated with condition ENUM
    - If Collectable card game selected
        - Show text box for number of cards being offered
    - If video game selected
        - Show dropdown for platform

```
SELECT name AS 'PlatformName' from platform
```

```
```

        - Show dropdown for media – populated with media ENUM
    - If computer game selected
        - Show dropdown for platform – populated with computer game platform ENUM
- On click List Item button
- Validate user input
    - If failed validation, no query
    - Else:

```
title = gt_txtbox.GetValue()
description = gd_txtbox.GetValue()
condition = gc_combobox.GetStringSelection()
INSERT INTO Item () VALUES ();
INSERT INTO BoardGame (item_number, title, description, game_condition, owner_email)
SELECT NewItemNum.item_count, str(title), str(description, str(condition),'$UserID'
FROM (SELECT COUNT(item_number) AS item_count FROM Item) AS NewItemNum;
```

- Display screen with a success message pop up.
- End

My Items

Abstract Code

- Get game counts for logged user

```
SELECT COUNT(item_number) AS GameCount FROM BoardGame WHERE user_email = '$UserID';
SELECT COUNT(item_number) AS GameCount FROM CollectibleCardGame WHERE user_email =
'$UserID';
SELECT COUNT(item_number) AS GameCount FROM ComputerGame WHERE user_email =
'$UserID';
SELECT COUNT(item_number) AS GameCount FROM PlayingCardGame WHERE user_email =
'$UserID';
SELECT COUNT(item_number) AS GameCount FROM VideoGame WHERE user_email = '$UserID';
```

- Get each item's parameters and display them

```
SELECT title, condition, description from BoardGame WHERE user_email = '$UserID';
    • display all the user's listed BoardGame items. if len(description) > 100: description =
      description[:100] + "..."
SELECT title, condition, description from PlayingCardGame WHERE user_email = '$UserID';
    • display all the user's listed PlayingCardGame items. if len(description) > 100: description =
      description[:100] + "..."
SELECT title, condition, description from ComputerGame WHERE user_email = '$UserID';
    • display all the user's listed ComputerGame items. if len(description) > 100: description =
      description[:100] + "..."
SELECT title, condition, description from CollectibleCardGame WHERE user_email = '$UserID';
    • display all the user's listed CollectibleCardGame items. if len(description) > 100: description =
      description[:100] + "..."
SELECT title, condition, description from VideoGame WHERE user_email = '$UserID';
```

- display all the user's listed VideoGame items. if len(description) > 100: description = description[:100] + "..."
- For each item, display a button to bring up a detailed item view

Search

Abstract code

1. Begin
2. Show a rundown of radio buttons which permit to look by watchword, in clients postal code, inside X miles, in determined postal code
3. Client selects one of the options ($search_option)
4. On Click Search!:
    1. Query the database to retrieve items which could be part of the new trade. These items should satisfy below three conditions:
        1. Item should not belong to the user
        2. Items should not be a part of an existing trade
        3. With calculated distance based on postal_code of $SessionID (logged in user's email) and the postal_code of users items which satisfies one of the search conditions

```
WITH items_union AS(

   select

       temp.item_number,

       title,

       description,

       game_condition,

       game_type,

       email

   from
```

```
        Item it
        INNER JOIN (
            (
                select
                    item_number,
                    title,
                    description,
                    game_condition,
                    "Video game" as game_type,
                    email
                from
                    VideoGame
            )
            UNION
            (
                select
                    item_number,
                    title,
                    description,
                    game_condition,
                    "Computer game" as game_type,
                    email
                from
                    ComputerGame
            )
            UNION
            (
```

```
        select
            item_number,
            title,
            description,
            game_condition,
            "Collectible card game" as game_type,
            email
        from
            CollectibleCardGame
    )
    UNION
    (
        select
            item_number,
            title,
            description,
            game_condition,
            "Playing card game" as game_type,
            email
        from
            PlayingCardGame
    )
    UNION
    (
        select
            item_number,
            title,
```

```
                    description,
                    game_condition,
                    "Board game" as game_type,
                    email
                from
                    BoardGame
            )
        ) temp ON temp.item_number = it.item_number
),
query_on_user AS(
    -- Used to get logged in TradePlazaUser's email
    SELECT
        u.*
    FROM
        TradePlazaUser u
    WHERE
        u.email = '$SessionID'
),
accepted_trade_items AS (
    -- Used to get list of accepted items (proposer and counterparty) so that we can
exclude them
    SELECT
        proposed_item_number,
        counter_party_item_number,
        trade_status
    FROM
        Trade s
),
```

```
items_to_find_dist AS(
    SELECT
        item_number,
        game_type,
        title,
        game_condition,
        description,
        u.postal_code AS item_postal_code,
        u.email
    FROM
        items_union i
        LEFT JOIN TradePlazaUser u ON i.email = u.email
    WHERE
        u.email <> '$SessionID'
        AND item_number NOT IN(
            (
                SELECT
                    proposed_item_number
                FROM
                    accepted_trade_items
            )
            UNION
            (
                SELECT
                    counter_party_item_number
                FROM
                    accepted_trade_items
```

```
            )
        )
),
response_time AS(
    select
        i.email,
        ROUND(
            avg(TIMESTAMPDIFF(DAY, accept_reject_date, NOW()), 1)
        ) as Response_Time
    from
        items_union i
        INNER JOIN TradePlazaUser u ON i.email = u.email
        INNER JOIN Trade tr on i.item_number = tr.counter_party_item_number
    where
        trade_status = "Accepted"
        or trade_status = "Rejected"
    Group By
        i.email
),
user_rank AS(
    select
        i.email,
        Count(*) as trade_count,
        CASE
            WHEN trade_count >= 10 THEN 'Alexandinium'
            WHEN trade_count >= 8
            AND trade_count <= 9 THEN 'Platinum'
```

```
            WHEN trade_count >= 6
            AND trade_count <= 7 THEN 'Gold'
            WHEN trade_count >= 4
            AND trade_count <= 5 THEN 'Silver'
            WHEN trade_count >= 3
            AND trade_count <= 4 THEN 'Bronze'
            WHEN trade_count >= 1
            AND trade_count <= 2 THEN 'Aluminium'
        END as user_rank
    from
        items_union i
        INNER JOIN TradePlazaUser u ON i.email = u.email
        INNER JOIN ((
            SELECT
                proposed_item_number as tr_item_no
            FROM
                accepted_trade_items
            where
                trade_status = "Accepted"
        )
    UNION
    (
        SELECT
            counter_party_item_number as tr_item_no
        FROM
            accepted_trade_items
        where
```

```
            trade_status = "Accepted"
    )) tr on i.item_number = tr.tr_item_no
    Group By
        i.email
),
lat_lon AS(
    SELECT
        items_to_find_dist.*,
        query_on_user.postal_code AS user_postal_code,
        RADIANS(a.latitude) AS lat1,
        RADIANS(a2.latitude) AS lat2,
        RADIANS(a.longitude) AS lon1,
        RADIANS(a2.longitude) AS lon2,
        RADIANS(a2.latitude - a.latitude) :: NUMERIC(9, 6) AS delta_lat,
        RADIANS(a2.longitude - a.longitude) :: NUMERIC(9, 6) AS delta_lon
    FROM
        items_to_find_dist
        LEFT JOIN response_time on items_to_find_dist.email = response_time.email
        LEFT JOIN user_rank on items_to_find_dist.email = user_rank.email
        CROSS JOIN query_on_user
        LEFT JOIN Address a ON items_to_find_dist.item_postal_code = a.postal_code
        LEFT JOIN Address a2 ON query_on_user.postal_code = a2.postal_code
),
haversine AS(
    SELECT
        DISTINCT ON (item_postal_code, user_postal_code) item_postal_code,
        user_postal_code,
```

```
        (SIN(delta_lat / 2.0) * SIN(delta_lat / 2.0)) + (
            COS(lat1) * COS(lat2) * SIN(delta_lon / 2.0) * SIN(delta_lon / 2.0)
        ) AS haversine_a
    FROM
        lat_lon
)
SELECT
    la.item_number,
    la.game_type,
    la.title,
    la.game_condition,
    la.description,
    h.item_postal_code,
    h.user_postal_code,
    3958.75 * 2 * (ATAN2(SQRT(haversine_a), SQRT(1 - haversine_a))) AS distance
FROM
    lat_lon la
    LEFT JOIN haversine h ON (
        la.user_postal_code = h.user_postal_code
        AND la.item_postal_code = h.item_postal_code
    ) --WHERE CLAUSE ADDED HERE
```

5. If ($search_option) = “search by keyword”
    1. search only on name_title and description for keyword
    2. Add in to last line of above query where it is specified –WHERE

       CLAUSES ADDED HERE–

WHERE

```
(
    la.title LIKE '%$search_keyword%'
    OR la.description LIKE '%$search_keyword%'
)
```

6. If ($search_option) = “search by in user’s postal code”

   WHERE

   ```
   h.item_postal_code = h.user_postal_code
   ```

7. If ($search_option) = “within X miles”
   1. Take user input of X: ($miles)

      WHERE

      ```
      distance <= ' $ miles '
      ```

8. If ($search_option) = “search by specified postal code”
   1. Take user input of postal code: ($postal_code_search)

      WHERE

      ```
      h.item_postal_code = '$postal_code_search'
      ```

9. Assuming that results are found:
10. Show thing number, game sort, thing name/title, the condition and the initial 100 characters of the depiction (in the event that the portrayal is more noteworthy than 100 characters, place an ellipsis (…) toward the finish to demonstrate it has been shortened) , normal reaction time and dealer position of other client
11. In the event that hunt by catchphrase chose:
12. Then, at that point, feature the fields that match the catchphrase in blue

13. Assuming that pursuit by postcode is chosen AND the postcode is invalid:
14. Then show a mistake message
15. On click Detail:
16. Divert to a matching View Item page
17. ELSE
18. Show a message "Sorry, no outcomes found!"
19. END

View Item

Abstract code

- Click through from Detail CTA In Search or Detail CTA in My Items
- Determine if click through is from Search or My Items. Get item_number ($item_number) based on the item that the user clicked

1. If from My Items, display
    1. item_number, title, game type, condition, description (if not null)
    2. If item listed is a “Video Game”, “Computer Game” or “Collectible Card Game”, display specific fields

```
WITH items_union AS(

   select

        temp.item_number,

        title,

        description,

        game_condition,

        game_type,

        email

   from
```

```
        Item it
        INNER JOIN (
            (
                select
                    item_number,
                    title,
                    description,
                    game_condition,
                    "Video game" as game_type,
                    email,
                    null as number_of_cards,
                    null as cg_platform,
                    media,
                    name as vg_platform
                from
                    VideoGame
                    INNER JOIN platform on VideoGame.platform_id =
platform.platform_id
            )
            UNION
            (
                select
                    item_number,
                    title,
                    description,
                    game_condition,
                    "Computer game" as game_type,
                    email,
```

```
            null as number_of_cards,
            platform as cg_platform,
            null as media,
            null as vg_platform
        from
            ComputerGame
    )
    UNION
    (
        select
            item_number,
            title,
            description,
            game_condition,
            "Collectible card game" as game_type,
            email,
            number_of_cards,
            null as cg_platform,
            null as media,
            null as vg_platform
        from
            CollectibleCardGame
    )
    UNION
    (
        select
            item_number,
```

```
            title,
            description,
            game_condition,
            "Playing card game" as game_type,
            email,
            null as number_of_cards,
            null as cg_platform,
            null as media,
            null as vg_platform
        from
            PlayingCardGame
    )
    UNION
    (
        select
            item_number,
            title,
            description,
            game_condition,
            "Board game" as game_type,
            email,
            null as number_of_cards,
            null as cg_platform,
            null as media,
            null as vg_platform
        from
            BoardGame
```

```
            )

        ) temp ON temp.item_number = it.item_number

),

SELECT

   i.item_number,

   i.title,

   i.game_condition,

   i.description,

   i.game_type,

   i.vg_platform,

   i.media,

   i.cg_platform,

   i.number_of_cards



FROM items_union

WHERE

   i.item_number = '$item_number'

   AND i.email = '$SessionID'
```

1. If from Search, display the same information as listed in My Items, but with the addition of the following information about the other user (referred as counterparty).
    1. nickname, city, state, postal code. Distance, item_number, rank, response time can be taken from Search task and forwarded to this display
    2. Using item_number ($item_number), perform query to get nickname of

       counterparty with item_number

SELECT

```
  u.nickname
FROM
   items_union i
   LEFT JOIN TradePlazaUser u ON i.email = u.email
WHERE
   item_number = ' $ item_number '
```

1. Using postal code for item for counterparty (forwarded from Search), query city and state

   SELECT

```
  city,
   state
FROM
   Address
WHERE
   postal_code = '$postal_code'
```

1. If the user's postal code!= the item's owner postal code,
2. Then display distance
3. IF 0.0 < distance < 25.0 miles
4. Then add green background
5. IF 25.00 < distance < 50.0 miles
6. Then add yellow background
7. IF 50.00 < distance < 100.0 miles
8. Then add orange background
9. IF distance > 100.0 miles
10. Then add red background
11. OTHERWISE Hide distance

12. Depending on the context, different elements will be displayed.
13. If the user has not accepted < 2 trades, then display Propose Trade.
14. OTHERWISE, Propose Trade is not displayed.
15. When you click Propose Trade, you will be redirected to Add Trade Proposal

View Trade History

Chris G

Abstract Code

- User clicked on ***View Trade History*** from **Main Menu**
- **Trade Summary Task**
  - Find current User using global variable *TradePlazaUser.email*, from login

```
SELECT Email FROM `User` WHERE TradePlazaUser.email='UserID';
```

  - Display **Trade Summary** form at the top of **Trade History** form, provides statistics when current User is Proposer or Counterparty, show Total appearance, Accepted count, Rejected count, and Reject rate

```
SELECT "My role"
        COUNT(*) AS Total,
        SUM( IF (trade_status="Accepted", 1, 0)) AS Accepted,
        SUM( IF (trade_status ="Rejected", 1, 0)) AS Rejected,
        FORMAT( Rejected / Total , 'P1') AS "Rejected %"
FROM (
        SELECT trade_status,
                IF(P.OwerID='$UserID', "Proposer", "Counterparty") AS 'My role',
        FROM Trade AS T
                INNER JOIN (
                    SELECT FK_Item_Item_Number, FK_User_Email from BoardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from PlayingCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from CollectibleCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from ComputerGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from VideoGame
                    ) AS PI ON T.FK_Proposer_Item_Item_number=PI.FK_Item_Item_Number
                INNER JOIN (
                    SELECT FK_Item_Item_Number, FK_User_Email from BoardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from PlayingCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from CollectibleCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from ComputerGame
```

```
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from VideoGame
                    ) AS CI ON T.FK_Proposer_Item_Item_number=CI.FK_Item_Item_Number

        WHERE CI.FK_User_Email='$UserID' OR PI.FK_User_Email='$UserID'
        )
GROUP BY "My role" ;
```

- If Rejected % >=50%, highlight background in red
- Below, display **Trade History Detail** form, provides detailed information related to trades, show Proposed Date, Accepted/Rejected Date, Trade Status, Response Time, My role, Proposed Item, Desired Item, Other User (Counterparty)

```
SELECT proposed_data AS "Proposed Date",
        accept_reject_data AS "Accepted/Rejected Date",
        trade_status AS "Trade status",
        ISNULL(proposed_date-accept_reject_date, proposed_date-FORMAT(
                getdate(), MM/dd/yyyy) ) AS "Response time (days)",
        IF(PI.FK_User_Email='$UserID', "Proposer", "Counterparty") AS 'My role',
        PI.title AS 'Proposed Item',
        CI.title AS 'Desired Item',
        IF( PI.FK_User_Email='$UserID', CU.nickname, PU.nickname ) AS 'Other User'
FROM Trade AS T
            INNER JOIN (
                    SELECT FK_Item_Item_Number, FK_User_Email, title, from BoardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from PlayingCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from CollectibleCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from ComputerGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from VideoGame
                    ) AS PI ON T.FK_Proposer_Item_Item_number=PI.FK_Item_Item_Number
            INNER JOIN (
                    SELECT FK_Item_Item_Number, FK_User_Email from BoardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from PlayingCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from CollectibleCardGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from ComputerGame
                    UNION
                    SELECT FK_Item_Item_Number, FK_User_Email from VideoGame
                    ) AS CI ON T.FK_Proposer_Item_Item_number=CI.FK_Item_Item_Number
            LEFT JOIN TradePlazaUser AS PU ON PU.email=PI.FK_User_Email
            LEFT JOIN TradePlazaUser AS CU ON CU.email=CI.FK_User_Email
WHERE PI.FK_User_Email ='$UserID' OR CI.FK_User_Email ='$UserID'
```

```
ORDER BY “Proposed Date” DESC,  “Response time (days)” DESC;
```

- After each row of **Trade History Detail** form, there is one ***Detail*** button, on click goes to **View Trade Detail** form
- A ***Back*** button should present, on click goes back to **Main Menu**

View Trade Details

Chris G

Abstract Code

- Detail view breaks down into four sections
    - Trade Details
    - User Details
    - Proposed Item
    - Desired Item

```
DECLARE @Dlat double(16,8)
DECLARE @Dlon double(16,8)
DECLARE @a double(16,8)
DECLARE @c double(16,8)
DECLARE @R double(16.8) = 3958.75

SELECT T.proposed_date AS “Proposed Date”,
        T.accept_reject_date AS “Accepted/Rejected Date”,
        T.trade_status AS “Trade status”,
        IF(PI.FK_User_Email=’$UserID’, “Proposer”, “Counterparty”) AS ‘My role’,
        ISNULL(T.proposed_date- T.accept_reject_date, T.proposed_date -FORMAT(
                getdate(), MM/dd/yyyy) ) AS ‘Response time (days)’,
        IF( PI.FK_User_Email=’$UserID’, CU.nickname, PU.nickname ) AS Nickname

        @Dlat= Address.Latitude- Address.Latitude,
        @Dlon= Address.Longitude- Address.Longtitude,
        @a= SQUARE(SIN(@Dlat/2)) + COS(Address.Latitude)* COS(Address.Latitude)*
SQUARE(SIN(@Dlon/2)),
        @c= 2*ATN2(SQRT(@a), SQRT(1-@a)),
        @R * @c AS Distance
        IF( PI.FK_User_Email=’$UserID’, CU.first_name + “ ” + CU.last_name, PU.first_name +
“ ” + PU.last_name ) AS Name
        IF( PI.FK_User_Email=’$UserID’, CU.email, PU.email) AS Email

        PI.FK_Item_Item_Number AS ‘Item #’,
        PI.title AS Title,
        PI. game_type AS "Game type”,
        PI.condition AS Condition,
        PI.Description AS Description,

        CI.FK_Item_Item_Number AS ‘Item #’,
```

```
        CI.title AS Title,
        CI. game_type AS "Game type",
        CI.condition AS Condition,

FROM `Trade` AS T
            INNER JOIN (
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Board Game" AS
game_type, condition, description from BoardGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Playing Card Game"
AS game_type, condition, description from PlayingCardGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Collectible Card
Game" AS game_type, condition, description from CollectibleCardGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Computer Game"
AS game_type, condition, description from ComputerGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Video Game" AS
game_type, condition, description from VideoGame
                ) AS PI ON T.FK_Proposer_Item_Item_number=PI.FK_Item_Item_Number
            INNER JOIN (
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Board Game" AS
game_type, condition from BoardGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Playing Card Game"
AS game_type, condition from PlayingCardGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Collectible Card
Game" AS game_type, condition from CollectibleCardGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Computer Game"
AS game_type, condition from ComputerGame
                UNION
                SELECT FK_Item_Item_Number, FK_User_Email, title, "Video Game" AS
game_type, condition from VideoGame
                ) AS CI ON T.FK_Proposer_Item_Item_number=CI.FK_Item_Item_Number
            LEFT JOIN TradePlazaUser AS PU ON PU.email=PI.FK_User_Email
            LEFT JOIN TradePlazaUser AS CU ON CU.email=CI.FK_User_Email
            INNER JOIN Address ON FK_Address_PostalCode=Address.postal_code
WHERE PI.FK_User_Email ='$UserID' OR CI.FK_User_Email ='$UserID'
```

- A ***Back*** button should present, on click goes back to **<u>View Trade History</u>**