Phase 2 Abstract Code w/SQL

CS 6400 - Spring 2017 Team 067

Table of Contents:

Login

Abstract Code

- Display buttons:
  - **View Web Reports**
  - **Login**
- Upon:
  - Click **Web Reports** button
    - Jump to the **View Web Reports**
- User enters *user name* ($UserName), *password* ($Password) input fields.
- If data validation is successful for both username and password input fields, then:
  - When Enter button is clicked:

```
SELECT * FROM User
WHERE UserName='$UserName'
AND password='$Password';
```

If User record is found but User.password != '$Password':

- Go back to Login form, with error message.

  - Else:
    - Store login information as session variable $UserName.
    - Go to Main Menu form.
- Else *username* and *password* input fields are invalid, display **Login** form, error message. (See Reference 1 pg.22)

Web Reports

Abstract Code

- Display buttons:
  - **View Available Bunks/Rooms Report**
  - **View Meals Remaining Report**
- Upon:
  - Click **View Available Bunks/Rooms Report** button
    - Jump to the **View Available Bunks/Rooms Task**

    - Click **View Meals Remaining Report** button
        - Jump to the **View Meals Remaining Task**

View Available Bunks/Rooms Report

Abstract Code

- When View button is Clicked

```
SELECT
FacilityName,MaleBunkAvailable, FemaleBunkAvailable, MixtedBunkAvailable,
FamilyRoomAvailable, Phone, Address
FROM Shelter
JOIN Site on Shelter.SiteID=Site.SiteID
WHERE Shelter.SiteID='$SiteID'
AND (MaleBunkAvailable>0
OR FemaleBunkAvailable>0
OR MixtedBunkAvailable>0
OR FamilyRoomAvailable>0);
```

- Display all Shelters with available bunks or rooms.
- If a Shelter in the ASA System has an available bunks or rooms display:
    - Name
    - Site location
    - Phone number
    - Hours of operation and conditions
    - Number of male/female/mixed bunks and rooms available
- Else
    - Display “Sorry all shelters are currently at maximum capacity” message.
- Click **Back** button
    - Brings unauthenticated clients back to **Web Reports** form.

View Meals Remaining Report

Abstract Code

- When view is clicked:

```
SELECT min(totalUnits)
```

```
FROM
(SELECT
          CASE WHEN SubCategory=’Meats/seafood’ THEN ‘Protein’
          WHEN SubCategory=’Dairy/eggs’ THEN ‘Protein’
          WHEN SubCategory=‘Vegetables’ THEN ‘Vegie’
          ELSE ‘Seed’
          END AS mealComponents,
          SUM(NumberOfUnits) totalUnits
          FROM Item
         WHERE SubCategory in (‘Vegetables’, ‘nuts/grains/beans’, ‘Meat/seafood’,
‘Dairy/eggs’)
GROUP BY mealComponents
```

- Display all meals remaining in inventory in all Food Banks in ASA system.

```
SELECT mealComponets
FROM
(SELECT
          CASE WHEN SubCategory=’Meats/seafood’ THEN ‘Protein’
          WHEN SubCateogy=’Dairy/eggs’ THEN ‘Protein’
          WHEN SubCategory=‘Vegetables’ THEN ‘Vegie’
          ELSE ‘Seed’
          END AS mealComponents,
          SUM(NumberOfUnits) totalUnits
          FROM Item
         WHERE SubCategory in (‘Vegetables’, ‘nuts/grains/beans’, ‘Meat/seafood’,
‘Dairy/eggs’)
GROUP BY mealComponents
WHERE totalUnits=min(totalUnits)
LIMIT 1;
```

- Display:
    - Type of donations are most needed to provide more meals, Display vegetables if Vegie is the result, nuts/grains/beans if Seeds is the result, or one of meat/seafood or dairy/eggs if Protein is the result)
- Click **Back** button
    - Brings unauthenticated clients back to **Web Reports** form.

Main Menu

Abstract Code

- Look up $Username and show the following buttons contingent on what $ServiceID are connected to the users $SiteID.
- Display buttons:
  - **View Site Services**
  - **Add Site Services**
  - **View Client Search/Report**
  - **Add Item to Foodbank** (if Users $SiteID is associated with a Foodbank)
  - **View Item Search/Report**
  - **View Waitlist Report** (if Users $SiteID is associated with a Shelter)
  - **View Outstanding Requests Report** (if Users $SiteID is associated with a Food Bank)
  - **View Requests Status Report**
  - **Log Off**
- Click on **View Site Services** button
  - Jump to **View Site Services** Task
- Click on **Add Site Services** button
  - Jump to **Add Site Services** Task
- Click on **View Client Search/Report** button
  - Jump to **View Client Search/Report** Task
- Click on **Add Item to Foodbank** button
  - Jump to Add Item to Foodbank Task
- Click on **View Item Search/Report** button
  - Jump to **View Item Search/Report** Task
- Click on **View Waitlist Report** button
  - Jump to **View Waitlist Report** Task.
- Click on **View Outstanding Requests Report** button
  - Jump to **View Outstanding Requests Report** Task.
- Click on **View Requests Status Report** button
  - Jump to **View Requests Status Report** Task.
- Click **Log Off** button
  - Invalidate login session and go back to the **Login** form.

View Site Services

Abstract Code

- Get all the services which are associated with a $SiteID which are associated with $Username.
- Lookups all Services associated with the current User's $SiteID.

```
SELECT FacilityName, HoursOfOperation, SeatsCapacity, SeatsAvailable
FROM
SoupKitchen JOIN
Site on SoupKitchen.SiteID=Site.SiteID
WHERE
SiteID='$SiteID';

SELECT ConditionForUse
FROM SoupKitchenConditionForUse
WHERE ServiceID=(SELECT ServiceID
                 FROM SoupKitchen
                 WHERE SiteID='$SiteID');
```

```
SELECT FacilityName, HoursOfOperation
FROM
FoodPantry JOIN
Site on FoodPantry.SiteID=Site.SiteID
WHERE
SiteID='$SiteID';

SELECT ConditionForUse
FROM FoodPantryConditionForUse
WHERE ServiceID=(SELECT ServiceID
                 FROM FoodPantry
                 WHERE SiteID='$SiteID');
```

```
SELECT FacilityName, HoursOfOperation, MaleBunkAvailable,
FemaleBunkAvailable, MixedBunkAvailable, FamilyRoomAvailable
FROM
Shelter JOIN
```

```
Site on Shelter.SiteID=Site.SiteID
WHERE
SiteID='$SiteID';

SELECT ConditionForUse
FROM ShelterConditionForUse
WHERE ServiceID=(SELECT ServiceID
                 FROM Shelter
                 WHERE SiteID='$SiteID');
```

```
SELECT FacilityName
FROM
FoodBank JOIN
Site on FoodBank.SiteID=Site.SiteID
WHERE SiteID='$SiteID';
```

- For each Service the User is associated with, display all the basic information for each Service.
- Click on a service
  - Jump to **Edit Site Service** Task
- Click **Back** button
  - Brings User back to **Main Menu** form.

Add/Edit/Delete Site Service

Abstract Code

- View Site Services.
- Add, Edit, Delete the services for a site that the user is associated with.
- Get the ServiceID corresponding to the specific SiteID and Service.

  - If an update of the existing service
    - Update Soup Kitchen

```
Update SoupKitchen
SET FacilityName='$FacilityName',
HoursOfOperation='$HoursOfOperation',
SeatsCapacity='$SeatsCapacity', SeatsAvailable='$SeatsAvailable'
WHERE ServiceID='$ServiceID';

Update SoupKitchenConditionForUse
SET ConditionForUse='$ConditionForUse'
WHERE ConditionForUse='$OldConditionForUse'
AND ServiceID='$ServiceID';
```

- Update Food Pantry

```
Update FoodPantry
SET FacilityName='$FacilityName',,
HoursOfOperation='$HoursOfOperation',
WHERE ServiceID='$ServiceID';

Update FoodPantryConditionForUse
SET ConditionForUse='$ConditionForUse'
WHERE ConditionForUse='$OldConditionForUse'
AND ServiceID='$ServiceID';
```

- Update Selter

```
Update Shelter
SET FacilityName='$FacilityName',
HoursOfOperation='$HoursOfOperation',
MaleBunkAvailable='$MaleBunkAvailable',
FemaleBunkAvailable='$FemaleBunkAvailable',
MixedBunkAvailable='$MixedBunkAvailable',
FamilyRoomAvailable='$FamilyRoomAvailable',
WHERE ServiceID='$ServiceID';

Update ShelterConditionForUse
SET ConditionForUse='$ConditionForUse'
WHERE ConditionForUse='$OldConditionForUse'
AND ServiceID='$ServiceID';
```

- Update Food Bank

```
Update FoodBank
SET FacilityName='$FacilityName',
```

- If an new service
    - Add Soup Kitchen

```
INSERT INTO SoupKitchen(FacilityName, HoursOfOperation,
SeatsCapacity, SeatsAvailable)
VALUES('$FacilityName', '$HoursOfOperation', '$SeatsCapacity',
'$SeatsAvailable');

INSERT INTO SoupKitchenConditionForUse(ServiceID,
ConditionForUse)
VALUES('$ServiceID', '$ConditionForUse');
```

    - Add Food Pantry

```
INSERT INTO FoodPantry(FacilityName, HoursOfOperation)
VALUES('$FacilityName', '$HoursOfOperation');

INSERT INTO FoodPantryConditionForUse(ServiceID,
ConditionForUse)
VALUES('$ServiceID', '$ConditionForUse');
```

    - Add Shelter

```
INSERT INTO Shelter(FacilityName,HoursOfOperation,
MaleBunkAvailable, FemaleBunkAvailable, MixedBunkAvailable,
FamilyRoomAvailable)
VALUES('$FacilityName', '$HoursOfOperation',
'$MaleBunkAvailable', '$FemaleBunkAvailable',
'$MixedBunkAvailable', '$FamilyRoomAvailable');

INSERT INTO ShelterConditionForUse(ServiceID, ConditionForUse)
VALUES('$ServiceID', '$ConditionForUse');
```

    - Add Food Bank

```
INSERT INTO FoodBank(FacilityName)
VALUES('$FacilityName'');
```

- If delete a service
    - Delete Soup Kitchen

```
DELETE FROM SoupKitchen
WHERE SiteID='$SiteID';

DELETE FROM SoupKitchenConditionForUse
WHERE ServiceID=(SELECT ServiceID
                 FROM SoupKitchen
                 WHERE ServiceID='$ServiceID');
```

    - Delete Food Pantry

```
DELETE FROM FoodPantry
WHERE SiteID='$SiteID';

DELETE FROM FoodPantryConditionForUse
WHERE ServiceID=(SELECT ServiceID
                 FROM FoodPantry
                 WHERE ServiceID='$ServiceID');
```

    - Delete Selter

```
DELETE FROM Sheter
WHERE SiteID='$SiteID';

DELETE FROM ShelterConditionForUse
WHERE ServiceID=(SELECT ServiceID
                 FROM Shelter
                 WHERE ServiceID='$ServiceID');
```

    - Delete Food Bank

```
DELETE FROM FoodBank
WHERE SiteID='$SiteID';
```

- Run the View Site Service again.

- Click **Back** button
    - Brings User back to **Main Menu** form.

View Client Search / Report

Abstract Code

- Display input boxes
    - Client name
    - ID number
- Display **Search** and **Add** buttons
- Upon:
    - Click on **Search** button
        - Run **Search for Client Record** task

```
SELECT FirstName, LastName, IDNumber, IDDescription
FROM Client
Where FirstName LIKE ‘%$FirstName%’
OR LastName LIKE ‘%$LastName%’
OR IDNumber LIKE ‘%$IDNumber%’
LIMIT 5;
```

        - Display maximum 5 clients that matched the query. Each client row shows:
            - Client name
            - ID number
            - ID description
        - Click on a client
            - Jump to **View/Edit Client**
    - Click on **Add** button
        - Jump to **Add Client**

Add client

Abstract Code

- Display input boxes
    - Client name
    - ID number
    - ID description

    - Contact phone number
    - Is head of household? (Yes/No)
- Display **Cancel** and **Submit** buttons
- Upon:
    - Click on **Cancel** button
        - Jump to back to **View Client Search / Report**
    - Click on **Submit** button
        - Run **Add Client Entry** task

```
INSERT INTO Client (ClientName, IDNumber, IDDescription,
Phone, IsHeadOfHousehold)
Values($ClientName, $IDNumber, $IDDescription,
$Phone,$IsHeadOfHousehold)
```

        - Jump to back to **View Client Search / Report**

View/Edit Client

Abstract Code

- Display text boxes
    - Client name
    - ID number
    - ID description
    - Contact phone number
    - IsHeadOfHousehold

```
SELECT FirstName, LastName, IDNumber, IDDescription, Phone,
IsHeadOfHousehold
FROM Client
WHERE ClientID = '$ClientID';
```

- Display **Edit** button
    - Click on **Edit** button
        - Text boxes become input boxes

- Display table of Service Usage Log

```
SELECT TimeStamp, ServiceType
FROM ServiceUsageLogEntry
WHERE ClientID = ‘$ClientID’
ORDER BY TimeStamp DESC;
```

- Display table of Waitlist Log (only for shelter)

```
SELECT Shelter.ServiceID, WaitListEntry.OrderIndex
FROM WaitlistEntry INNER JOIN Shelter
ON WaitlistEntry.ServiceID = Shelter.ServiceID
WHERE ClientID = ‘$ClientID’
ORDER BY OrderIndex ASC;
```

- Display table of Field Modified Log

```
SELECT Timestamp, Description
FROM FieldModifiedLogEntry
WHERE ClientID = ‘$ClientID’
ORDER BY Timestamp DESC;
```

- Display table of available services as a scrollable view
    - Each service row should display
        - Type
        - ID Description
        - Hours of Operation
        - Conditions
        - Etc.
- Upon:
    - Click on a service row
        - If the service is available, highlight the row
        - If the service is not available, show a message
- Display **Cancel** and **Submit** buttons
- Upon:
    - Click on **Cancel** button
        - Jump to back to **View Client Search / Report**

- Click on **Submit** button
    - Run **Edit Client** task

```
UPDATE Client
SET FirstName = ‘$FirstName’, LastName = ‘$LastName’, IDDescription =
‘$IDDescription’ , Phone = ‘$phone’
WHERE ClientID = ‘$ClientID’ ;

INSERT INTO FieldModifiedLogEntry (ClientID, TimeStamp,Description)
VALUES  ( ‘$ClientID’, $TimeStamp, ‘$Description’);
```

Check in Client

Abstract Code

- Run **Record Client Check Into Service** task

```
INSERT INTO ServiceUsageLogEntry (ClientID, Timestamp, ServiceType )
VALUES ($ClientID, $Timestamp, $ServiceType);
```

- If the service is shelter, display bunk/room type that is available

```
SELECT FemaleBunkAvailable FROM Shelter WHERE ServiceID=’$ServiceID’
SELECT MaleBunkAvailable FROM Shelter WHERE ServiceID=’$ServiceID’
SELECT MixedBunkAvailable FROM Shelter WHERE ServiceID=’$ServiceID’
SELECT FamilyRoomAvailable FROM Shelter WHERE ServiceID=’$ServiceID’
```

- Update the number of available bunk/room if the bunk/room has more than one unit available.
- If checks into female bunk.

```
UPDATE Shelter
SET FemaleBunkAvailable=FemableBunkAvailable-1
WHERE ServiceID=’$ServiceID’
```

- If checks into male bunk.

```
UPDATE Shelter
SET MaleBunkAvailable=MaleBunkAvailable-1
WHERE ServiceID='$ServiceID'
```

- If checks into mixed bunk.

```
UPDATE Shelter
SET MixedBunkAvailable=MixedBunkAvailable-1
WHERE ServiceID='$ServiceID'
```

- If checks into family room.

```
UPDATE Shelter
SET FamilyRoomAvailable=FamilyRoomAvailable-1
WHERE ServiceID='$ServiceID'
```

- If the no available family room, insert the client into the waitlist.

```
INSERT INTO WaitlistEntry (ServiceID, ClientID, OrderIndex) VALUES (
$ServiceID, $ClientID, $OrderIndex)
```

- Jump to back to **View Client Search / Report**

Check out Client

Abstract Code

- If the client checks out at Shelter. Update the available of the units for the bunk/room type that the client checks out.
- If checks out female bunk.

```
UPDATE Shelter
SET FemaleBunkAvailable=FemableBunkAvailable+1
WHERE ServiceID='$ServiceID'
```

- If checks out of the male bunk.

```
UPDATE Shelter
SET MaleBunkAvailable=MaleBunkAvailable+1
```

```
WHERE ServiceID='$ServiceID'
```

- If checks out of the mixed bunk.

```
UPDATE Shelter
SET MixedBunkAvailable=MixedBunkAvailable+1
WHERE ServiceID='$ServiceID'
```

- If checks out of the family room.

```
UPDATE Shelter
SET FamilyRoomAvailable=FamilyRoomAvailable+1
WHERE ServiceID='$ServiceID'
```

View Wait List Report

Abstract Code

- If no clients on the waitlist, display "The wait list is empty"
- Else, display list of clients. Each client row should show:
    - Client's name
    - ID number
    - ID Description
    - Contact phone number
    - Rank

```
SELECT Client.FirstName, Client.LastName, Client.IDNumber, Client.Phone,
WaitlistEntry.OrderIndex
FROM WaitlistEntry
INNER JOIN Client ON WaitlistEntry.ClientID = Client.ClientID
WHERE ServiceID = '$ServiceID'
ORDER BY OrderIndex ASC;
```

- Able to drag and drop a client row to change its order and save in the database
- Store the old OrderIndex as $OldOrderIndex, and the new OrderIndex as $NewOrderIndex for $ClientID.

- If the $OldOrderIndex equals $NewOrderIndex, do nothing.
- If the $OldOrderIndex is greater than the $NewOrderIndex

```
UPDATE WaIitlistEntry
SET OrderIndex=OrderIndex+1
WHERE ServiceID=$ServiceID
AND ClientID IN (SELECT ClientID in WaitlistEntry
                 WHERE OrderIndex < $OldOrderIndex
                 AND OrderIndex >= $NewOrderIndex
                 AND ServiceID =$ServiceID)

UPDATE WaitlistEntry
SET OrderIndex=$NewOrderIndex
WHERE ServiceID = $ServiceID AND ClientID = $ClientID;
```

- If the OldOrderIndex is smaller than the NewOrderIndex

```
UPDATE WaIitlistEntry
SET OrderIndex=OrderIndex-1
WHERE ServiceID=$ServiceID
AND ClientID IN (SELECT ClientID in WaitlistEntry
                 WHERE OrderIndex > $OldOrderIndex
                 AND OrderIndex <= $NewOrderIndex
                 AND ServiceID =$ServiceID )

UPDATE WaitlistEntry
SET OrderIndex=$NewOrderIndex
WHERE ServiceID = $ServiceID AND ClientID = $ClientID;
```

- Display **Delete** button in each client row
- Upon:
  - Click on **Delete** button, client row disappears

```
DELETE FROM WaitlistEntry
WHERE ServiceID = ‘$ServiceID’ AND ClientID = ‘$ClientID’;
```

- Display Reset and Submit buttons
- Upon:
  - Click on **Reset** button
    - Revert back to previous form

        - Click on **Submit** button
            - Apply changes - Run **Add, Reduce, Change Position Of Client** task

Add Item to Foodbank

Abstract Code

- Display input boxes
    - ServiceID
    - Item Name
    - Number Of Units
    - Expiration Date
    - Storage Type
    - Category
    - SubCategory
- Display **Cancel** and **Submit** buttons
- Upon:
    - Click on **Cancel** button
        - Jump to back to **Main Menu**
    - Click on **Submit** button
        - Run **Add Item to Foodbank** task

```
INSERT INTO Item (ServiceID, ItemName, NumberOfUnits, ExpirationDate,
StorageType, Category, SubCategory)
VALUES('$ServiceID', '$ItemName', '$NumberOfUnits', '$ExpirationDate',
'$StorageType', '$Category', '$SubCategory')
```

        - **Jump back to Main Menu**

View Item Search/Report

Abstract Code

- Display filters:
    - Food Bank (Default: All)
    - Expiration Date (Default: All)
    - Storage Type (Default: All)
    - Food/Supply (Default: Food)
    - Sub-category (Default: depending on Food/Supply)

    - Item name / Description input box

```
SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory FROM Item WHERE ServicID = '$ServiceID' ;

SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory FROM Item WHERE ExperationDate >=
'$ExperiationDate'; ORDER by ExperationDate ASC;

SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory FROM Item  WHERE StorageType = '$StorageType';

SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory FROM Item WHERE Category = '$Category';

SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory FROM Item WHERE SubCategory = '$SubCategory';

SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory FROM Item WHERE ItemName = '$ItemName';
```

- Run query based on the above filters to list items. Item with 0 # of units will not be displayed.
- Upon:
    - Click on an item
        - If owned, jump to **View Owned Item**
        - If not owned, jump to **View Request For Item**

View Owned Item

Abstract Code

- Run **View Owned Item** task based on $ItemID

```
SELECT ItemName, NumberOfUnits, ExpirationDate, StorageType, Category,
SubCategory
FROM Item
```

```
WHERE ItemID = '$ItemID' AND ServiceID = '$ServiceID';
```

- Display Current Item's # of units
- Display New Item's # of units input box
- Display **Cancel** and **Submit** buttons
- Upon:
    - Click on **Cancel** button
        - Jump back to **View Item Search / Report**
    - Click on **Submit** button
        - Run **Edit Item's # of Units** task

```
UPDATE Item
Set NumberOfUnits = '$NumberOfUnits'
WHERE ItemID = '$ItemID' AND ServiceID = '$ServiceID' ;
```

- Jump back to **View Item Search / Report**

View Request For Item

Abstract Code

- Run **Search for Items** task based on $ItemID

```
SELECT ServiceID, ItemName, NumberOfUnits, ExpirationDate, StorageType,
Category, SubCategory
FROM Item
WHERE ItemID = '$ItemID' ;
```

- Display Current Item's # of units
- Display # Items requested input box
- Display **Cancel** and **Submit** buttons
- Upon:
    - Click on **Cancel** button
        - Jump back to **View Item Search / Report**
    - Click on **Submit** button
        - Run **Request for Item** task

```
UPDATE Item
Set NumberOfUnits = '$NumberOfUnits'
WHERE ItemID = '$ItemID' AND ServiceID = '$ServiceID' ;
```

- Jump back to **View Item Search / Report**

View Outstanding Requests Report

Abstract Code

- Run the **View Requests** task based on $ServiceID
- Display all requests. Each request row should show:
    - Item name
    - Storage type
    - Category
    - Sub-category
    - # Item requested

```
SELECT Item.ItemName, Item.StorageType, Item.Category,
Item.SubCategory, Request.ItemQuantity
FROM Request
INNER JOIN Item ON Request.ItemID = Item.ItemID
WHERE ServiceID = '$ServiceID' ;
```

//can we display order from different

- For each request, compare #Item requested with Item's # of unit. If the #Item requested is more than Item's # of unit, display red on this request.
- Click on a request
    - Jump to **View/Edit Request**

View/Edit Request

Abstract Code

- Run the **View Request** task based on $RequestID
- A request should show:
    - Item name
    - Storage type
    - Category

  - Sub-category
  - # Item requested

```
SELECT Item.ItemName, Item.StorageType, Item.Category,
Item.SubCategory, Request.ItemQuantity
FROM Request
INNER JOIN Item ON Request.ItemID = Item.ItemID
WHERE RequestID= ‘$RequestID’ ;
```

- Display an input box for user to key in # Item provided

```
UPDATE Request
SET ItemProvided = ‘$ItemProvided’
WHERE RequestID = ‘$RequestID’;

UPDATE Item
SET Item.NumberOfUnits = Item.NumberOfUnits - ‘$ItemProvided’
WHERE ItemID = ‘$ItemID’
```

- User can also click on these 2 buttons to quickly change the input box
  - Fullfil: #Item Provided will be equal to #Item Requested

```
UPDATE Request
SET ItemProvided = ItemQuantity
WHERE RequestID = ‘$RequestID’;

UPDATE Item
SET Item.NumberOFUnits = Item.NumberOfUnits - ‘$ItemProvided’
WHERE ItemID = ‘$ItemID’;
```

  - Deny: #Item Provided will be 0

```
UPDATE Request
SET ItemProvided = ‘0’
WHERE RequestID = ‘$RequestID’;
```

- Display **Cancel** and **Submit** buttons

- Upon:
    - Click on **Cancel** button
        - Jump back to **View Outstanding Requests Status Report**
    - Click on **Submit** button
        - Run **Edit Outstanding Request** task
        - Jump back to **View Outstanding Requests Status Report**

View Requests Status Report

Abstract Code

- Run the **View Requests** task based on $Username
- Display all requests. Each request row should show:
    - Food Bank name
    - Item name
    - # Items requested
    - # Items provided (only for closed requests)
    - Status: pending/closed
    - **Cancel** button (only for pending requests)

```
SELECT FoodBank.FacilityName, Request.ItemName, Request.ItemQuantity,
Request.ItemProvided, Request.Status
FROM Request
INNER JOIN User ON Request.Username = User.Username;
INNER JOIN FoodBank ON User.SiteID = FoodBank.SiteID ;
WHERE Username = '$Username' ;
```

- Upon:
    - Click on **Cancel** button
        - Run **Cancel Request** task
    - Click on **Back** button.
        - Jump back to **Main Menu**.

Reference

1. *Phase 2 Abstract Code w/SQL CS 6400 Spring 2017 Template*