Table of Contents:

Home

Abstract Code:

Show "***Log In***", "***View Bunks/Rooms***", "***View Meals Remaining***" tabs.
Upon:

Click ***Log In*** button-Jump to the **Log In** task.
Click ***View Bunks/Rooms*** button-Jump to the **View Bunks/Rooms** task.
Click ***View Meals Remaining*** button-Jump to the **View Meals Remaining** task.

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

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

Go back to **Log In** form, with error message.

Else:

Store login information as session variable $UserID.

Else *username* and *password* input fields are invalid, display **Log In** form, with error message

Main Menu

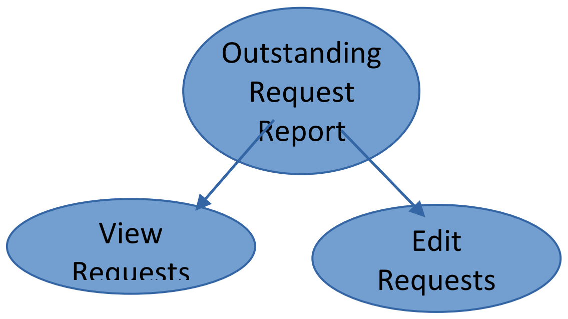
Abstract Code:

With user logged in, show "**View Bunks/Rooms**", "**View Meals Remaining**", "**View Services**", "**Search Items**", "**Request Items**", "**Request Status**" and "**Log out**" tabs.
Upon:

Click ***View Bunks/Rooms*** button-Jump to the **View Bunks/Rooms** task.
Click ***View Meals Remaining*** button-Jump to the **View Meals Remaining** task.
Click ***View Services*** button-Jump to the **View Services** task.
Click ***Search Items*** button-Jump to the **Search Items** task.
Click ***Request Items*** button-Jump to the **Request Items** task.
Click ***Request Status*** button-Jump to the **Request Status** task.

Outstanding Request Report

Task Decomp



Abstract Code

View Requests:
With session variable $SiteID which get from current user table, display all outstanding requests:

```
SELECT name, r.num_request, unit, storage_type, category, sub_category
FROM Item AS I join Request AS r on i.item_ID=r.item_ID where i.site_id=$SiteID;
```

If the user chooses a column to sort, either ‘storage_type’, ‘category’, ‘sub_category’ or ‘num_request’, store the column name as session variable $SortCol and display the sorted result:

```
SELECT name, r.num_request, unit, storage_type, category, sub_category
FROM Item AS I join Request AS r on i.item_ID = r.item_ID where i.site_id=$SiteID
order by $SortCol;
```

Indicate the shortfall in inventory:

```
SELECT name
FROM (
SELECT name, sum(r.num_request) as num_request, sum(unit) as unit
FROM Item AS I join Request AS r on i.item_ID = r.item_ID where i.site_id=$SiteID
GROUP BY name, i.item_id)
WHERE num_request>unit;
```

Show “***Edit Requests***” button, if click on the button, jump to the **Edit Requests** task.
Show “***Main Menu***” link, if click on the link, jump to the **Main Menu** form.

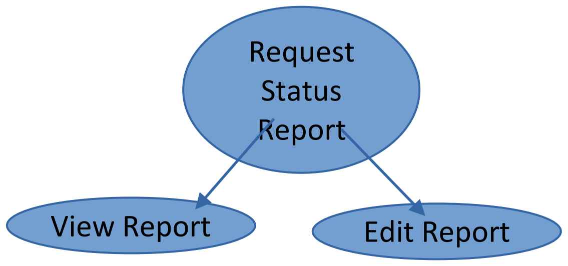
Edit Requests:
The user can Edit the request fulfilled in full by click the “***Edit***” button. If the button is clicked, user can enter $NumProvide and $Status in the input fields, and database is updated given current $ItemID and $SiteID:

```
UPDATE Request SET num_provide = $NumProvide, status='$Status' WHERE user_id =
$UserID AND item_id=$ItemID;
```

Show “***View Requests***” link, if click on the link, jump to the **View Requests** form.

Request Status Report

Task Decomp



Abstract Code:

The user can view the requests. With session variable $UserID, display:

```
SELECT i.name, r.num_request, r.num_provide, r.status FROM Request AS r join Item
AS i on i.item_id=r.item_id WHERE r.user_id=$UserID;
```

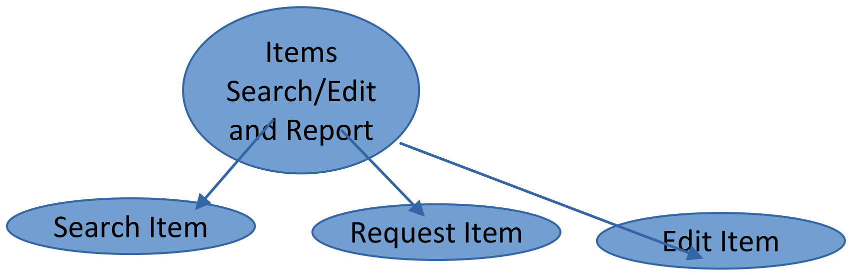
When ***Cancel*** button is clicked:

```
DELETE FROM Request WHERE user_id=$UserID AND item_id = $ItemID;
```

Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

Items Search/Edit and Report

Task Decomp



Abstract Code:

Search Items:
user can view all inventory of the item:

```
SELECT * FROM Item WHERE service_name='FoodBank';
```

If user chooses a specific food bank with $SiteID, then:

```
SELECT * FROM Item WHERE service_name='FoodBank' AND site_id=$SiteID;
```

If User chooses some filters of expiration_date, with session variable $ExpDate, display:

```
SELECT * FROM Item WHERE service_name='FoodBank' AND
expiration_date>=$ExpDate;
```

If User chooses some filters of storage_type, with session variable $StoreType, display:

```
SELECT * FROM Item WHERE service_name='FoodBank' AND storage_type
='$StoreType';
```

If based on a keyword with session variable $Keyword, display:

```
SELECT * FROM Item WHERE service_name='FoodBank' AND name like
lower('%$Keyword%');
```

Show "***Request Item***" button, if click on the button, jump to the **Request Items** task.

Request Items:
On the page, user can select item (create session variable $ItemID) from dropdown menu, and sourcing foodbank (create session variable $SiteID) from dropdown menu. Also user needs to input number requested for the item (create session variable $NumRequest). Also, we can get $NumProvide:

```
SELECT count(*) FROM Item WHERE service_name='FoodBank' AND item_id=$ItemID
AND site_id=$SiteID;
```

By default, $Status is set to 'pending'. If ***Save*** button is pressed, update the database:

```
INSERT INTO Request ((user_ID, item_ID, num_request, num_provide, status) VALUES
($UserID, $ItemID, $NumRequest, $NumProvide, $Status));
```

Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

Add Items:
User at a food bank can add items to the inventory. With session variables $UserID and $SiteID passed into the task, user need to specify $Name, $Unit, $StorageType, $ExpirationDate, $Category, $SubCategory by entering these in the input fields. If ***Save*** button is clicked, update the database:

```
INSERT INTO Item ((name, unit, storage_type, expiration_date, category,
sub_category, site_id, service) VALUES ($Name, $Unit, $StorageType, $ExpirationDate,
$Category, $SubCategory, $SiteID, 'FoodBank'));
```

Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

Edit Items:
User at a food bank can update the inventory. With session variables $UserID and $SiteID passed into the task, user need to choose an item from dropdown menu (create session variable $ItemID) and input the updated units $Unit.
When ***Save*** button is clicked:

```
UPDATE Item SET unit= $Unit WHERE item_id=$ItemID, site_id=$SiteID AND
service='FoodBank' ;
```

Also, check whether the count of some items reaches zero and remove those from database:

```
DELETE FROM Item WHERE unit=0;
```

Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

View/Edit Available Bunks/Rooms

Abstract Code

View Bunks/Rooms
On this page, show the list of available bunks and rooms using the following two queries:

```
 SELECT s.name, s. street_address, s.city, s.state, s.zip_code, s.phone_number,
b.type, b.available_count
from Site s join Bunk b on s.site_id = b.site_id
where b.available_count>0;
```

```
SELECT s.name, s. street_address, s.city, s.state, s.zip_code, s.phone_number,
h.hours_of_operation, h.condition_for_use, b.familyroom_count from Site s join
Shelter h on s.site_id = h.site_id where familyroom_count>0;
```

Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

Edit Bunks/Rooms
On this page, a user at a shelter (with $SiteID) can edit available bunk count or modify the room available number. By choosing a bunk from a dropdown menu, a session variable $BunkID is created. User can specify the new available count number $AvaiCnt by entering the input field. If ***Save*** button is clicked:

```
UPDATE Bunk SET available_count = $AvaiCnt WHERE bunk_id = $BunkID AND
site_id=$SiteID;
```

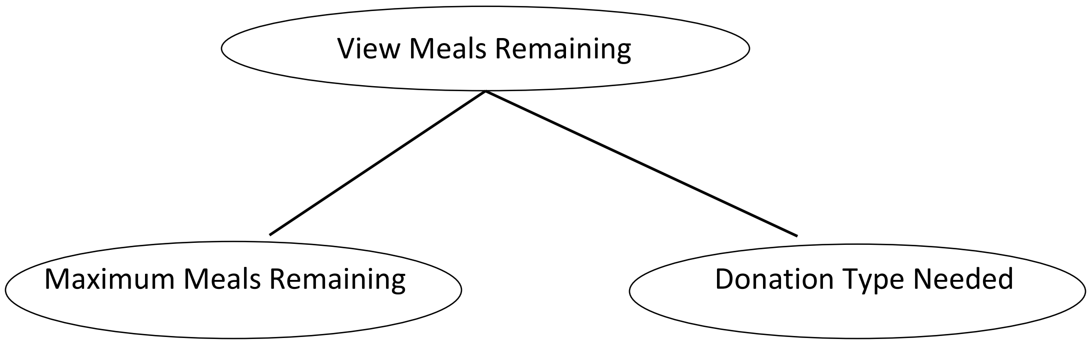
Similarly, user can specify the new available count number of family room $AvaiCntFamily by entering the input field. If ***Save*** button is clicked:

```
UPDATE Shelter SET familyroom_count=$AvaiCntFamily WHERE site_id=$SiteID;
```

Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

View Meals Remaining

Task Decomposition:



Abstract Code

```
SELECT sum(unit) FROM Item WHERE subcategory='Vegetables';
```

Store the number as session variable $NumVeg.

```
SELECT sum(unit) FROM Item WHERE subcategory='nut/grains/beans';
```

Store the number as session variable $NumNut.

```
SELECT sum(unit) FROM Item WHERE subcategory='Meat/seafood' or subcategory='Dairy/eggs';
```

Store the number as session variable $NumMeat.
Find the minimum of these three numbers and display the number as number of meals remaining.
If $NumVeg is the smallest, display 'Vegetables' as the category that needs donation. If $NumNut is the smallest, display 'nut/grains/beans'' as the category that needs donation. If $NumMeat is the smallest, display 'Meat/seafood' or 'Dairy/eggs'' as the category that needs donation.
Show "***Main Menu***" link, if click on the link, jump to the **Main Menu** form.

View Site Services

Task Decomp

Abstract Code:

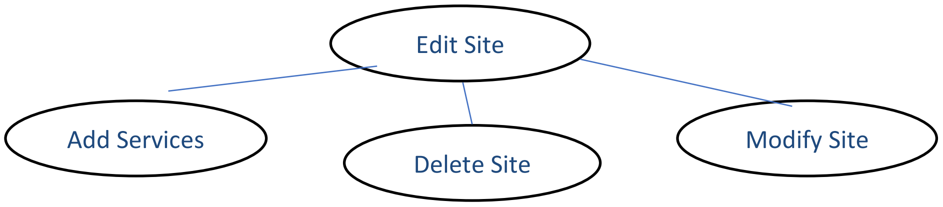
Run the ***View Site Services*** task based in $Site_ID

```
SELECT * FROM Foodbank WHERE Site_ID = '$Site_ID';
SELECT * FROM Food_Pantry WHERE Site_ID = '$Site_ID';
SELECT * FROM Soup_Kitchen WHERE Site_ID = '$Site_ID';
SELECT * FROM Shelter WHERE Site_ID = '$Site_ID';
```

```
Find the current Site using Site_ID
Find each service for the site
        {Display Service_Name;
        If subService button is clicked:
                Display Description, Hours of Operation, Condition for Use;
                If Servie_Name is Soup Kitchen, display Seat_Capacity and Seat_Available.
                If Servie_Name is Shelter, display FamilyRoom_Count, Bunk Type, Bunk_ID, Count and Available
        Count
        }
```

Edit Site Service

Task Decomp



Abstract Code:

```
        Run the Edit Site Services task based in $Site_ID
        View Site Services, populate service dropdowns
        If ADD button is pressed:  View Site Services; display the form with room for another service;
```

```
INSERT INTO Foodbank (Site_ID , Service_Name)VALUES('$Site_ID', 'Foodbank') ;

INSERT INTO Food_Pantry (Site_ID , Service_Name, Description, Hours_of_Operation,
Condition_for_Use)VALUES('$Site_ID', 'Food_Pantry', '$Description', '$Hours_of_Operation',
'$Condition_for_Use') ;

INSERT INTO Soup_Kitchen (Site_ID , Service_Name, Description, Hours_of_Operation, Condition_for_Use,
Seat_Capacity, Seat_Available)VALUES('$Site_ID', 'Soup_Kitchen', '$Description', '$Hours_of_Operation',
'$Condition_for_Use', '$Seat_Capacity', '$Seat_Available') ;
```

```
INSERT INTO Shelter ((Site_ID , Service_Name, Description, Hours_of_Operation, Condition_for_Use,
FamilyRoom_Count)VALUES('$Site_ID', 'Shelter', 'Shelter', '$Description', '$Hours_of_Operation',
'$Condition_for_Use', '$FamilyRoom_Count') ;
```

If **EDIT** button is pressed: display the form with current services to be modified.

```
//Nothing to be updated for Food Bank.

UPDATE Food_Pantry SET  Description = '$Description',Hours_of_Operation = '$Hours_of_Operation',
Condition_for_Use = '$Condition_for_Use' WHERE Site_ID = $Site_ID AND Service_Name = $Service_Name;

UPDATE Soup_Kitchen SET  Description = '$Description',Hours_of_Operation = '$Hours_of_Operation',
Condition_for_Use = '$Condition_for_Use',  Seat_Capacity = '$Seat_Capacity', Seat_Available =
'$Seat_Available' WHERE Site_ID = $Site_ID AND Service_Name = $Service_Name;

UPDATE Shelter SET  Description = '$Description',Hours_of_Operation = '$Hours_of_Operation',
Condition_for_Use = '$Condition_for_Use',  FamilyRoom_Count  = '$FamilyRoom_Count' WHERE Site_ID =
$Site_ID AND Service_Name = $Service_Name;
```

If **DELETE** button is pressed:
If only one service available, display error message 'Last service can only be deteted by database Administrator'
Else **Delete Service**; **View Site Services**;

```
Delete FROM Foodbank WHERE Site_ID = $Site_ID AND Service_Name = 'Foodbank';
Delete FROM Food_Pantry WHERE Site_ID = $Site_ID AND Service_Name = 'Food_Pantry';
Delete FROM Soup_Kitchen WHERE Site_ID = $Site_ID AND Service_Name = 'Soup_Kitchen';
Delete FROM Shelter WHERE Site_ID = $Site_ID AND Service_Name = 'Shelter';
```

If **SAVE** button is pressed: save change. **View Site Services.**

Search Clients

Abstract Code:

```
Display text box input($input)
Click on button search:
   Search and validate text box's content
   $client_list= Find all clients with Name like *$input*
```

```
SELECT count(*),Client_ID, PhoneNumber, Name, ID_Description FROM Client WHERE Name = '%$input%'
GROUP BY Client_ID, ORDER BY Name;
```

```
   If count(*)>5, return nothing and pop out "warning: refine the search keyword"
   Else if count(*) == 0, go to Client Register Form.
   Else
      For each $client
            Find and display $Client.Client_ID, $Client.Name, $Client.ID_Description.
            Click on $Client.Client_ID, will let you go to Check-in Report.
```

View Client Report

Abstract Code:

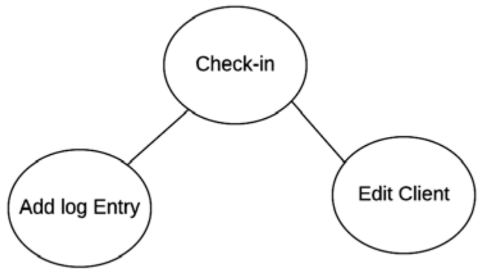
Run the **View Client Report** task based on $Client.Client_ID.

```
SELECT *, FROM Client NATURAL JOIN log WHERE Client.Client_ID = $Client;
```

Find the current client's Name, Phone_Number, ID_Description, Head_of_Household and Log history.
Display the current client's Name, Phone_Number, ID_Description, Head_of_Household and Log history.

Add/Modify Client Report

Task Decomposition:



Abstract Code:

Display **Check-in Report** with text box (can be edited).
Display ***edit*** button next to each attribute (current client's Name, Phone_Number, ID_Description, Head_of_Household,).
Display text box and ***add log*** button at the end of client form.
When ***edit*** Button is pressed:

```
UPDATE Client SET Phonenumber = '$Phonenumber ', Head_of_Household = '$Head_of_Household', Name =
'$Name', ID_Description = '$ID_Description' WHERE Client_ID = $Client_ID;
```

Validate input data type and length constraint.
Write and update the related attributes of current client.

When ***add log*** button is pressed:

```
INSERT INTO log (Client_ID , DateTime , Site_ID, Description, Field_Modified)VALUES('$Client_ID',
'$DateTime','$Site_ID', '$Description', '$Field_Modified') ;
```

Validate input data type and length constraint.
And a new log entry and write the attributes (*Field Modified, Site_ID, Date/Time, Discription*).

Add Client

Abstract Code:

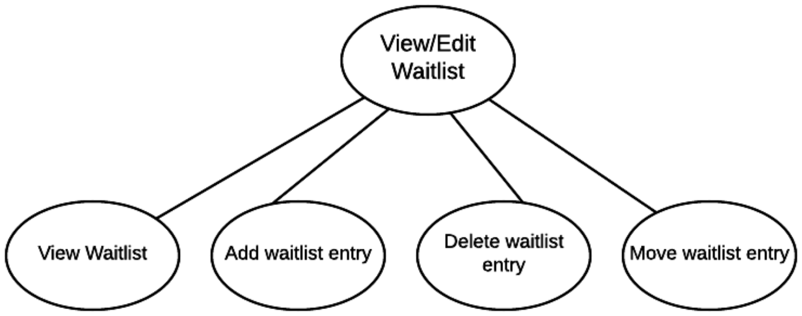
Display the **Client register Form** with textbox of *Name, Phone_Number, ID_Description, Head_of_Household and Log(Field Modified, Site_ID, Date/Time,Description)* attributes.
When ***submit*** Button is pressed:

```
INSERT INTO Client (PhoneNumber, Name, Head_of_Household, ID_Description)VALUES('$PhoneNumber',
'$Name','$Head_of_Household', '$ID_Description') ;
```

Validate input data type and length constraint.
Write and update the related attributes of current client.

View/Edit Waitlist

Task Decomposition:



Abstract Code:

View waitlist:

```
SELECT * FROM WaitList WHERE Site_ID = '$Site_ID'  AND Service_Name = 'Shelter' ORDER BY
Waitinglist_Ranking ASC;
```

Determine waitlist associated with shelter and associated with client.
For each $waitlist on waitlist, sorted by $waitlist.Waitlist_Ranking.
Display $waitlist on waitlist, sorted by $waitlist.Waitlist_Ranking.

Delete waitlist entry:
If ***delete*** button is pressed:

```
Delete FROM WaitList WHERE Site_ID = $Site_ID AND Service_Name = 'Shelter' AND Client_ID = '$Client_ID';
```

```
    Delete the current entry in waitlist relationship table.
    For each $entry with position below $current_entry:
```

```
UPDATE WaitList SET Waitinglist_Ranking = Waitinglist_Ranking-1 WHERE Site_ID = '$Site_ID'  AND
Service_Name = 'Shelter'' AND Client_ID = '$Client_ID';
```

```
        Decrement $selected_entry.Waitlist_Ranking by 1.
```

Move waitlist entry:
If ***move up*** button is pressed:

```
    Validate input and check whether the current entry is at the top.
    Set $previous.Waitlist_Ranking to $current_ entry.Waitlist_Ranking.
```

```
UPDATE WaitList SET Waitinglist_Ranking = Waitinglist_Ranking-1 WHERE Site_ID = '$Site_ID'  AND
Service_Name = 'Shelter'' AND Client_ID = '$Client_ID';
```

```
    Set $current_entry. Waitlist_Ranking to $new_position
    Set $new_position to $previous. Waitlist_Ranking
```

```
UPDATE WaitList SET Waitinglist_Ranking = Waitinglist_Ranking+1 WHERE Site_ID = '$Site_ID'  AND
Service_Name = 'Shelter'' AND Client_ID = '$Client_ID';
```

If ***move down*** button is pressed:

```
    Validate input, check whether the current entry is at the bottom.
    Set $next.Waitlist_Ranking to $current_ entry.Waitlist_Ranking.
```

```
UPDATE WaitList SET Waitinglist_Ranking = Waitinglist_Ranking+1 WHERE Site_ID = '$Site_ID'  AND
Service_Name = 'Shelter'' AND Client_ID = '$Client_ID';
```

```
    Set $current_entry.Waitlist_Ranking to $new_position
    Set $new_position to $next.Waitlist_Ranking
```

```
UPDATE WaitList SET Waitinglist_Ranking = Waitinglist_Ranking-1 WHERE Site_ID = '$Site_ID'  AND
Service_Name = 'Shelter'' AND Client_ID = '$Client_ID';
```

Add waitlist entry:

If ***add*** button is pressed:

```
    Validate input data type(date/time) and length constraint.
    Write a new entry with attributes of client_id, service_id and date/time into waitlist relationship table.
```

```
INSERT INTO WaitList (Site_ID, Service Name, Client_ID, Waitinglist_Ranking, DateTime)VALUES('$Site_ID',
'Shelter'','$Client_ID', '$Waitinglist_Ranking', '$DateTime') ;
```