Table of Contents

Data Types

User

Attribute	**Data type**	**Nullable**
EMAIL	String	Not Null
USERNAME	String	Not Null
PASSWORD	String	Not Null
FIRSTNAME	String	Not Null
LASTNAME	String	Not Null

Site

Attribute	**Data type**	**Nullable**
PRIMARY_PHONE_NO	Integer	Not Null
SHORT_NAME	String	Not Null
ST_NUMBER	Integer	No Null
STREET	String	Not Null
CITY	String	Not Null
STATE	String	Not Null
ZIP_CODE	Integer	Not Null
MARKED_FOR_DELETION	Boolean	Not Null

Client

Attribute	**Data type**	**Nullable**
FIRSTNAME	String	Not Null
LASTNAME	String	Not Null
ID_TYPE	String	Not Null
ID_NUM	Integer	Not Null

PHONE_NUM	Integer	Null
NOTES	String	Null
SITE	String	Null
DATE_TIME_STAMP	Date	Null

Shelter

Attribute	**Data type**	**Nullable**
FEMALE_BUNK_NUM	Integer	Null
MALE_BUNK_NUM	Integer	Null
MIXED_BUNK_NUM	Integer	Null
CONDITIONS_OF_USE	String	Not Null
DESCRIPTION	String	Null
HOURS_OF_OP	String	Not Null

Soup Kitchen

Attribute	**Data type**	**Nullable**
SEAT_COUNT	Integer	Not Null
CONDITIONS_OF_USE	String	Not Null
DESCRIPTION	String	Null
HOURS_OF_OP	String	Not Null

Food Pantry

Attribute	**Data type**	**Nullable**
CONDITIONS_OF_USE	String	Not Null
DESCRIPTION	String	Null
HOURS_OF_OP	String	Not Null

Food Bank

Attribute	**Data type**	**Nullable**
ITEM_NAME	String	Not Null
UNITS_NUM	Integer	Not Null
EXP_DATE	Date	Not Null

STORAGE_TYPE	String	Not Null

Food

Attribute	**Data type**	**Nullable**
FOOD_CATEGORY	String	Not Null

Supplies

Attribute	**Data type**	**Nullable**
SUPPLY_CATEGORY	String	Not Null

Request

Attribute	**Data type**	**Nullable**
ITEM_NAME	String	Not Null
ITEM_QTY	Integer	Not Null
STATUS	String	Not Null
FOOD_BANK_NAME	String	Not Null

Business Logic Constraints

- System will need to use the user data to authenticate users and only allow them to make modifications to the data for specific site they are associated with.
- Users at any site with a food pantry, shelter or soup kitchen may use the system to request items from one or more food banks. Users at a site with a food bank can approve or edit requests for their site's food bank.
- Clients are different from Users. Clients make use of soup kitchens, food pantries, and shelters. Users are the people that work or volunteer at the sites.
- If a site provides multiple services, all site users have full control of all data for all services.
- Users of a site are free to add additional services, or modify existing services, and the system must allow them to create, delete, and edit any of the following services only for their site. A site will have only one service of a particular type, so a site may have no more than 4 services (Shelter, Soup Kitchen, Food Pantry, Food Bank). [Users should not be able to remove the last service associated with their site, instead they will need to request that the database administrator removes the entire site for them.]
- Users from other sites may use the ASACS to review inventory and request items to stock their Food Pantries, Shelters or Soup Kitchens.
- System should provide functionality to reset the available bunk count(s) the next morning.
- All users of the ASACS system can search for items and those who are associated with a Shelter, Food Pantry or Soup Kitchen can put in a request for one or more (up to the maximum amount available) of them. These requests queue up until a user from that food bank processes them by looking at their "pending request" report. If a request is accepted as-is, the number of items in the food bank is reduced by that number. If a request is denied, the number of items provided will be zero and the Food Bank inventory will not change. If a request is only partially fulfilled, only that number of items will be removed from the Food Bank inventory.
- The ASA defines a "meal" as one unit each of a Vegetable, nuts/grains/beans, and Meat/seafood OR Dairy/eggs.
- Any user at a site with a Food Bank will need access to a form where they can add inventory into the ASACS as it is received.
- A Request is in pending status if it has not been acted upon by a user from the source food bank.
- A Request in in closed status if the request was fully or partially fulfilled.
- A Request can be fulfilled with a smaller number of items (as few as zero!) than requested, so the request must keep track of the number of items requested while in pending status, and the number of items provided once it is in closed status.
- It is possible for an item to be removed from the ASACS database (because its stock quantity reached zero) while a pending request for that item still exists. In this case, the request should be marked as closed with zero items provided.
- Clients can not be deleted by users, but their name, description and phone number can be modified.
- If a client's name or description field is modified, the original name/description field is stored in a "field modified" log entry.

- As clients make use of shelters, soup kitchens and food pantries, the users at those sites will enter a log entry as part of the check-in process, which consists of a date/time stamp, the site, and a textual description of the service used and any extra notes.
- Web report can be pulled up by anyone with no user authentication.
- If no bunks are available throughout the entire ASA system, the available bunks report should display a message saying “Sorry, all shelters are currently at maximum capacity.”
- Meals remaining report shows a simple number of the meals remaining in inventory in all food banks in the ASA system. (It does not consider food already at soup kitchens or food pantries and no longer in the food bank inventory system.) In addition the report explains what type of donations are most needed to provide more meals (Either vegetables, nuts/grains/beans, or one of Meat/seafood or Dairy/eggs.). This report is made available to local newscasts, government officials, and the general public via the web.
- Client search, Item Search/Request, Outstanding Requests, Request Status Reports are only provided to logged in users.
- For Client Search/Report, to protect client privacy, users are never shown a full list of all clients. They must provide a search string that matches at least one client in the database, but it may NOT match more than 4 clients in the database.
- If the user uses a search key that provides fewer than 5 clients, a list of the client’s names and descriptions are shown so that the user can disambiguate between clients with a similar name or search key.
- For Item Search/Report users must be able to search all inventories by using wild-cards, or view inventories based upon filters of expiration date, storage type, Food/Supply, individual category within Foods or Supplies, or even a keyword search based upon item name / description.
- A user should NOT be able to request an item from the Food Bank associated with their site, even though they can see items from their site.
- If the report shows an item that is “owned” by the Food Bank associated with the site that the current user is associated with, the user will be able to modify the numbers of that the item.
- A user can only view a report of all outstanding requests of a site if the user is associated with it.
- The outstanding requests report should be able to be sorted by storage type, category/subcategory, or by quantity of items requested.
- The outstanding requests report should indicate a shortfall if more than one request for a particular item exists such that the total number of items requested is greater than the total number available. This should be done by highlighting numbers in red or some other suitable method.
- From the outstanding requests screen, the user should be able to mark the request fulfilled in full, or reduce the total number of items provided to some smaller number (possibly zero) and then mark it partially fulfilled or unable to be fulfilled.
- The ASACS should automatically reduce inventory of items as requests are marked fulfilled. If an inventory item reaches zero quantity, you may leave the record in the database.
- If a user has made requests for items, this report will show all the details of their requests as well as the request status (pending or closed).
- A closed request will also show the actual number of items that was provided (possibly zero).

- A user can cancel any of their outstanding requests from this screen which will remove it from the system.

Task Decomposition and Abstract Code

View Outstanding Request Report

Task Decomposition

Lock Types: Lookup, read and write on User, Request, and Food Bank
Number of Locks: Single
Enabling Conditions: successful login needed, needs a Food Bank service
Frequency: unknown
Consistency (ACID): critical
Sub-tasks: **generate_outstanding_requests_report, sort_outstanding_requests_report, mark_partially_fulfilled, fulfilled_as_is, marked_unable_to_fulfill, reduce_num_of_items**

Abstract Code

- The database is queried and checks to see if the user is associated with a site that has a food bank
- If user works at a site with a food bank
    - Query the database and build a report that contains food bank information as well as request information, storage type ('$storageType'), category ('$category') or sub-category ('$subCategory'), item ('$item'), and quantity of items ('$itemQuantity')
    - Query the database to see if more than one request for a particular item exists. such that the total number of items requested is greater than the total number available. If so, highlight (red) the quantity of items ('$itemQuantity') on the report
    - Each reporlt row will contain a drop-down with options: **Make a Selection**, **Fulfilled in Full, Partially Fulfilled**, and **Unable To Be Fulfilled.**
    - If **Partially Fulfilled** or **Unable To Be Fulfilled** options are selected
        - Show input box for that request allowing the user to reduce the quantity of the item ('$itemQuantity')
    - If user hits ***Save*** button, loop through all of the requests.
        - If Request fulfilled option drop-down is still **Make a Selection**
            - Skip the request and move onto the next one
        - Else If Request fulfilled option drop-down is **Fulfilled in Full**,**Partially Fulfilled**, or **Unable To Be Fulfilled**
        - Update the request in the database with the fulfilled, partially fulfilled, or unable to be fulfilled information
            - If the request is partially fulfilled or unable to be fulfilled and the user entered a new item quantity for the request, update the request with the new quantity
            - If the request is fulfilled, calculate the new item quantity and update the database
    - Else If the user hits the ***Cancel*** Button
        - If user selects ***Yes*** button, they will be returned to the **<u>Main</u>** screen
        - If user selects ***No*** button, the user will remain on the **<u>Outstanding Request Report</u>**

● Else if user does not work with a site that has a food bank, display a message stating “You do not work with a site that has a food bank. Unable to display report”

View Meals Remaining Reports

Task Decomposition

Lock Types: Lookup task, read-only on Food Bank table
Number of Locks: single
Enabling Conditions: Anyone can access this report
Frequency: unknown
Consistency: critical
Sub-tasks: no sub-tasks, task name is **generate_meals_remaining_report**

Abstract Code

● The database is queried and returns a list of food bank meal report information
● Each table row is populated with the food bank’s '$name', '$siteLocation', '$phone', '$hoursAndConditions', '$maximumMeals',and '$donations'
● If user hits ***Close*** Button, return to the **Main** screen

View Available Bunks Report

Task Decomposition

Lock Types: Lookup task, read-only on Shelter
Number of Locks: single
Enabling Conditions: Anyone can access this report
Frequency: unknown
Consistency: critical
Sub-tasks: no sub-tasks, task name is **generate_bunks_report**

Abstract Code

● The database is queried and returns a list of available shelters along with their information.
● If no bunks are available, a message that says “Sorry, all shelters are currently at maximum capacity.” will be displayed.
● Else if bunks are available
    ● Each table row is populated with the shelter’s '$name', '$siteLocation', '$phone', '$hoursAndConditions', '$maleBunkNumber', '$femaleBunkNumber', and '$mixedBunkNumber'
● If user hits ***Close*** button, return to the **Main** screen

Client Search

Task Decomposition

Lock Types: Lookup task, read-only on Client
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-tasks: no sub-tasks, task name is **client_search**

Abstract Code

- User enters full name('$fullName') and/or ID ('$clientID') into input fields
- If user hits the ***Search*** button, the database will be queried based on '$fullName' and/or '$clientID'
    - If only '$fullName' was entered and multiple matches occur
        - Display client name and descriptions in a drop-down list
            - User selects client and then the Client Report (**<u>View Client Report</u>**) will appear
    - Else If only '$fullName' and/or '$clientID' was entered and 1 match occurs,
        - Client Report (**<u>View Client Report</u>**) for the searched client will appear
    - Else If there isn't a match,
        - A message will appear on the screen stating the client was not found in the database
- If user hits ***Cancel*** button,
    - The user will be prompted to confirm cancellation
    - If user selects ***Yes***, they will be returned to the **<u>Main</u>** screen
    - If user selects ***No***, the user will remain on the **<u>Search Client</u>** Screen

View Client Report

Task Decomposition

Lock Types: Lookup, update, delete task (read and write) on Client
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-task: **client_check_in, service_log, edit_client**

Abstract Code

- User enters email ('$Email'), password ('$Password'), fullname ('$fullName'), input fields
- If user hits ***Save*** button, '$Email', '$Password','$fullName' will be inserted into the database
- If user hits ***Cancel*** button,
    - The user will be prompted to confirm cancellation
    - If user selects ***Yes***, they will be returned to the **Main** screen
    - If user selects ***No***,the user will remain on the **New Client** screen

    - Client log will appear in a table along with menu options to ***Modify Client Information***, ***Add New Service Log Entry, Check Client Into a Bunk***, or ***Provide a Meal***
    - If user selects ***Modify Client Information*** option,go to **Modify Client Screen**
    - Else If user selects ***Add New Service Log Entry*** option,go to **New Service Log Entry** screen.
    - Else If user selects ***Check Client into a Bunk*** option, go to **Check In Client** screen
    - Else If user selects ***Provide a Meal*** option,go to **Provide a Meal** screen
    - Else If user hits ***Cancel*** button
        - The user will be prompted to confirm cancellation
        - If user selects ***Yes*** button, they will be returned to **Client Search Screen**
    - If user selects ***No*** button,the user will remain on the **Client Report**

- User selects client from a drop-down list, the the client's email ('$Email'), password ('$Password'), full name, ('$fullName'), and phone number ('$phone') input fields are populated with the proper data pulled from the database
- User may modify email ('$Email'), password ('$Password'), fullname, ('$fullName'), and phone number ($phone) input fields
- If user hits ***Save*** button, the system will check what fields were changed and will write out the changes to a log file. Then the database will be updated
- If user hits ***Cancel*** button,
    - The user will be prompted to confirm cancellation.
    - If user selects ***Yes*** button, they will be returned to the **Main** screen
    - If user selects ***No*** button,the user will remain on the **Modify Client Screen**

- User enters log entry ('$logEntry') input field
- If user hits ***Save*** button, the system will submit the new log entry ('$logEntry') into the database

- If user hits ***Cancel*** button
  - The user will be prompted to confirm cancellation
  - If user selects Yes, they will be returned to **Client Report**
  - If user selects No,the user will remain on the **New Service Log Entry** screen

Sign Up Form

Task Decomposition

Lock Types: Lookup, update task (read and write) on Client
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-tasks: no sub-tasks, task name is **enroll_client**

Abstract Code

- User may type in optional notes ('$notes') into input box. '$notes' will be appended to '$description' ("Male Bunk Checkin")
- If user hits ***Check In Client*** button,
  - Database will check bunk availability
  - If bunk is available,
    - Database is updated with client's check in time, notes,and other checkin information.
  - Else If bunk is unavailable,
    - A message will appear stating that there are no bunks available
- Else If user hits ***Cancel*** button,
  - The user will be prompted to confirm cancellation
  - If user selects Yes, they will be returned to the **Client Report** screen
  - If user selects No,the user will remain on the **Check In Client** screen

Login Form

Task Decomposition

Lock Types: Read only on User
Number of Locks: single
Enabling Conditions: None
Frequency: unknown
Consistency: not critical, order is not critical
Sub-tasks: no sub-tasks, task name is **authenticate_user**

Abstract Code

- User enters email ('$Email'), password ('$Password') input fields

- If data validation is successful for both username and password input fields, then:
    - When ***Enter*** button is clicked:
        - If User record is found but user.password != '$Password':
            - Go back to **<u>Login</u>** form, with error message
        - Else:
            - Store login information as session variable '$UserID'
            - Go to **<u>View Menu</u>** form
- Else email and password input fields are invalid, display **<u>Login</u>** form, with error message

Food Bank Service Actions

Task Decomposition

Lock Types: Read only on User
Number of Locks: single
Enabling Conditions: None
Frequency: unknown
Consistency: not critical, order is not critical
Sub-tasks: no sub-tasks, task name is **authenticate_user**

Abstract Code

- Show ***Generate Inventory Report***, ***Manage Items***, options
- Upon:
    - Click ***Generate Inventory Report*** button
        - Jump to the **<u>Generate Inventory Report</u>** screen
    - Click ***Manage Items*** button
        - Jump to the **<u>Manage Items</u>** screen

Generate Inventory Report

- Show Table with pending requests
- User selects options on each pending request in the table to ***Approve***, ***Deny***, or they ignore it

- If User hits ***Save*** button
    - Loop through each request
    - If ***Approve***, query the database and update the table using '$requestID' and set it to ***Approve***
    - Else If ***Deny***, query the database and update the table using '$requestID' and set it to ***Deny***
    - Else If option isn't selected, do nothing
- If User hits ***Cancel*** button
    - Go back to **<u>Food Bank Service Actions</u>** screen

Manage Items

- Display options to ***Add Item***, ***Remove Item***, or ***Search Item***

- If ***Add Item*** is selected, display the <u>**Add Item**</u> form
    - User puts in new item ('$itemName') in input field
    - User selects from drop-down whether item is “**Food**” or a “**Supply**”
    - If the user selects **Food**
        - A drop-down will appear that contain the following categories: **Vegetables**, **Nuts/Grains/Beans**, **Meat/Seafood**, **Dairy/Eggs**, **Sauce/Condiment/Seasoning**, and **Juice/Drink**
        - User will select food category ('$itemCategory') from the drop-down list
    - .● Else If the user selects **Supply**
        - A drop-down will appear that contain the following categories: **Personal Hygiene**, **Clothing**, **Shelter**, and **Other**
        - User will select supply category ('$itemCategory') from the drop-down list
    - User selects storage type ('$storageType') from a drop-down. The drop-down options are: **Dry Good, Refrigerated**, and F**rozen**
    - User will type in expiration date ('$expirationDate') into input field
    - User will type in number of item ('$itemQuantity') into input field

    - If user hits ***Save*** button
        - ○ Inserts new item from the input field into the database: '$itemName', '$storageType', '$itemCategory', '$itemQuantity', '$expirationDate'.
        - ○ Go back to <u>**Manage Items**</u> task, displaying to-do options
    - If user hits ***Cancel*** button
        - Go back to <u>**Manage Items**</u> task, displaying to-do options
- Else If ***Remove Item*** is selected, display list of items
    - If user selects an item to remove and hits the ***Save*** button
        - Display prompt for user to save changes
            - If user selects ***Yes***
                - Delete the item by itemID in the database.
                - Display a message notifying the user that the item was deleted
            - Else stay on the <u>**Manage Items**</u> form
        - Delete the item by itemID in the database
    - Else If user hits ***Cancel*** button
        - Go back to <u>**Manage Items**</u> task, displaying to-do options
- Else If ***Search Item*** is selected, display a search form
    - If user hits ***Search*** button
        - Query database for the item searched for
        - If found, display information on screen
        - Else display a message stating the item was not found
    - If user hits ***Cancel*** button
        - Go back to <u>**Manage Items**</u> task, displaying to-do options

Kitchen Service Actions

Task Decomposition

Lock Types: Lookup, update (read and write) on Soup Kitchen
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-tasks: no sub-tasks, task name is **reduce_soup_kitchen_seats**

Abstract Code

- Display the current kitchen seat count and the button ***Reduce Soup Kitchen Seats***
- If user clicks ***Reduce Soup Kitchen Seats***
    - Display prompt for user to confirm
            - If user selects ***Yes***
                - Reduce the available Soup Kitchen Seats count by one in the DB
                - Display a message notifying the user that the seat count was reduced and go back to the **<u>Kitchen Service Actions</u>** screen
            - Else If user hits ***Cancel*** button
        - Go back to **<u>Kitchen Service Actions</u>**

Shelter Service Actions

Task Decomposition

Lock Types: Lookup, update task (read and write) on Shelter
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-tasks: **reset_bunk_bed_count, view_bunk_bed_count, reduce_bunk_bed_count**

Abstract Code

- Display the current Bunk Bed Count in three categories:
    - Male Bunk Capacity ('$maleBunkCapacity')
    - Female Bunk Capacity ('$femaleBunkCapacity')
    - Mixed Bunk Capacity ('$mixedBunkCapacity')
- Below display the following buttons: ***Reset Bunk Bed Count*** and ***Reduce Bunk Bed Count***
- If user clicks ***Reduce Bunk Bed Count***
    - Display drop-down menu to select from Male, Female or Mixed bunks
    - Display **<u>Confirmation</u>** screen
            - If user selects ***Yes***

- Reduce the respective available Bunk Bed type count by one in the DB
- Display a message notifying the user that the bunk count was reduced and go back to the **<u>Shelter Service Actions</u>** screen
- Else If user hits ***Cancel*** button
  - Go back to the **<u>Shelter Service Actions</u>** screen where the current Bunk Bed Count will be displayed

- Else if user clicks ***Reset Bunk Bed Count***
  - Display prompt for user to confirm
    - If user selects ***Yes***
      - Reset the available Bunk Bed count to the max availability
      - Display a message notifying the user that the bunk count was reset and go back to **<u>Shelter Service Actions</u>** where the current Bunk Bed Count will be displayed
    - Else If user hits ***Cancel*** button
      - Go back to **<u>Shelter Service Actions</u>** where the current Bunk Bed Count will be displayed

Manage Services

Task Decomposition

Lock Types: Lookup, update, insert, delete task (read and write) on Services
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-tasks: **add_service, edit_service, request_DBA_for_site_deletion, delete_service**

Abstract Code

- Display options as buttons: ***Add Service, Edit Service, Delete Service***
- If user clicks ***Add Service*** option, jump to **<u>Add Service</u>** screen
- Else If user clicks ***Edit Service*** option, jump to **<u>Edit Service</u>** screen
- Else If user clicks ***Delete Service*** option, jump to **<u>Delete Service Screen</u>**
- Else If user clicks ***Cancel*** button, jump to the **<u>Main</u>** screen

Add Service Screen

- Query the database to see if site offers Shelter based on site ID
  - If shelter is not offered yet
    - add **Shelter** as a drop-down (availableServices) option
- Query the database to see if site offers Soup Kitchen based on site ID
  - If soup kitchen is not offered yet
    - add **Soup Kitchen** as a drop-down (availableServices) option

- Query the database to see if site offers Food Pantry based on site ID
    - If food pantry is not offered yet
        - add **Food Pantry** as a drop-down (availableServices) option
- Query the database to see if site offers Food Bank based on site ID
    - If food bank is not offered yet
        - add **Food Bank** as a drop-down (availableServices) option

- Display a drop-down (availableServices) with available services
    - Drop-down is dynamic and may include **Shelter**, **Soup Kitchen, Food Pantry**, **Food Bank**
- User selects from drop-down (**availableServices**) options
    - If selected option is **Shelter**, **Soup Kitchen**, or **Food Pantry**
        - Description ('$serviceDescription'), hours of operation ('$operationHours'), and Conditions for use ('$serviceCondtions') input boxes appear on the screen
        - If selected option is **Shelter**
            - Bunk Capacity ('$bunkCapacity') input box appears on the screen
        - Else If selected option is **Soup Kitchen**
            - Bunk Capacity ('$seatingCapacity') input box appears on the screen
        - User types in Description ('$serviceDescription'), hours of operation ('$operationHours'), and Conditions for use ('$serviceCondtions') input boxes
        - If selected option is **Shelter**
            - User types into Male Bunk Capacity ('$maleBunkCapacity') input box
            - User types into Female Bunk Capacity ('$femaleBunkCapacity') input box
            - User types into Mixed Bunk Capacity ('$mixedBunkCapacity') input box
        - Else If selected option is **Soup Kitchen**
            - User type into Bunk Capacity ('$seatingCapacity') input box
- If user hits ***Add Service*** button
    - If selected option is **Shelter**, **Soup Kitchen**, or **Food Pantry**
        - Service for that site is added to database: Available Service drop-down selection ('$serviceType'), '$description', '$operationHours', '$serviceCondtions'
        - If selected option is **Shelter**
            - Input '$maleBunkCapacity', '$femaleBunkCapacity', '$mixedBunkCapacity' into Shelter table based on site ID
        - Else If selected option is **Soup Kitchen**
            - Input '$seatingCapacity' into Soup Kitchen database table based on site ID
    - Else If selected option is **Food Bank**, add Food Bank to Services table.
- If user hits ***Cancel*** button
    - The user will be prompted to confirm cancellation
    - If user selects ***Yes*** button, they will be returned to the **Manage Services** screen
    - If user selects ***No*** button,the user will remain on the **Add Service Screen**

Edit Service Screen

- Query the database and retrieve all services (service type) based on site ID.
- Display a drop-down (**offeredServices**) with offered services.

- If selected option is **Shelter**, **Soup Kitchen**, or **Food Pantry**
        - Description ('$serviceDescription'), hours of operation ('$operationHours'), and Conditions for use ('$serviceCondtions') input boxes appear on the screen.
        - Query the database and retrieve service description ('$serviceDescription'), hours of operation ('$operationHours'), and conditions for use ('$serviceConditions'), and the ID of the service ('$serviceID')
        - Populate Description ('$serviceDescription'), hours of operation ('$operationHours'), and Conditions for use ('$serviceCondtions') input boxes with retrieved data.
        - If selected option is **Shelter**
            - Query the database and retrieve male bunk capacity ('$maleBunkCapacity'), female bunk capacity ('$femaleBunkCapacity'), and mixed bunk capacity ('$mixedBunkCapacity') based on '$serviceID'.
            - Populate input boxes with '$maleBunkCapacity', '$femaleBunkCapacity', '$mixedBunkCapacity' and display input boxes on the screen
    - Else If selected option is **Soup Kitchen**
            - Query the database and retrieve seat count ('$seatingCapacity') based on '$serviceID'
            - Populate input box with '$seatingCapacity' display input box on the screen
    - User modifies text in Description ('$serviceDescription'), hours of operation ('$operationHours'), and Conditions for use ('$serviceCondtions') input boxes
    - If selected option is **Shelter**

        User modifies male bunk capacity ('$maleBunkCapacity'), female bunk capacity ('$femaleBunkCapacity'), and mixed bunk capacity ('$mixedBunkCapacity') input boxes
    - Else If selected option is **Soup Kitchen**
        - User modifies seat count ('$seatingCapacity') input box

    - If User hits the ***Save*** button
        - Update database table with new '$serviceDescription', '$operationHours', and '$serviceConditions'
        - If selected option is **Shelter**
        - Update database table with new male bunk capcity ('$maleBunkCapacity'), female bunk capacity ('$femaleBunkCapacity'), and mixed bunk capacity ('$mixedBunkCapacity') using '$serviceID'
        - Else If selected option is **Soup Kitchen**
        - Update database table with seat count ('$seatingCapacity') using '$serviceID'

    - Else If user hits ***Cancel*** button
        - The user will be prompted to confirm cancellation
        - If user selects ***Yes***, they will be returned to the **<u>Manage Services</u>** screen
        - If user selects ***No***,the user will remain on the **<u>Edit Service Screen</u>**

Delete Service Screen

- Query the database and retrieve all Services (service type) based on site ID
- Display a drop-down (**offeredServices**) with offered services

- User selects service from drop-down list
- If user hits ***Delete Service*** button
    - The user will be prompted to confirm deletion
    - If user selects ***Yes*** button
        - If there is only one service left in that site
            - Confirmation screen: the user will be warned that the site will be "marked for deletion", to be deleted by a DBA
            - If user selects ***Yes*** button, that site will be marked for deletion
            - Else If user selects ***No*** button,the user will remain on the **<u>Delete Service Screen</u>**
        - If there are more than one service
        - The service will be deleted from the database based on serviceID
        - The drop-down list of offered services (**offeredServices**) is updated
    - Else If user selects ***No*** button, the user will remain on the **<u>Delete Service Screen</u>**.
- Else If user hits ***Cancel*** button
    - The user will be prompted to confirm cancellation
    - If user selects ***Yes*** button, they will be returned to the **<u>Manage Services</u>** screen
    - If user selects ***No*** button,the user will remain on the **<u>Delete Service Screen</u>**

Item Request

Task Decomposition

Lock Types: Lookup, update task (read and write) on Client
Number of Locks: single
Enabling Conditions: successful login needed
Frequency: unknown
Consistency: critical
Sub-tasks: **item_request, cancel_request, request_from_food_bank, generate_request_status_report**

Abstract Code

```
● Query database for the item searched for requests made by user.
● If requests are found
        ● Populate a list of requests that contain the request details and the request status.
Lastly, every request that is outstanding will have a button (Cancel Request) next to it to delete
the request from the database
        ● If the request's status shows it to be a closed request
                ● Display number of items provided
        ● If user hits Cancel Request button
                ● Prompt the user to cancel the request
                ● If user hits Yes button
                        ● Delete the request from the request table based on the request ID
                        ● Display a message stating "The request was successfully cancelled
                        ● Remove the request from the Request Status Report view
                ● Else If user selects No button
                        ● Return to Request Status Report
        ● If user hits Close button
                Return to the Main screen
```