Phase II Report
CS6400 – Spring 2017
Team 019

TABLE OF CONTENTS

Phase II Report
CS6400 – Spring 2017
Team 019

LEGEND

Bold Italics → Button

Bold Underline → Document/form

$DollarName → variable

Bold → task

tablename → Name of table in SQL database

```
programming language code
```
→ psedo-code, not SQL

#Table of contents

Phase II Report
CS6400 – Spring 2017
Team 019

VIEW WAITLIST

Abstract Code:

- User selects ***View Waitlist*** button from **User Main Menu** form.
- Get the site information that this user has admin-control on.

```
SELECT siteid from user WHERE username='$Username';
```

- Lookup WAITLIST table for all the entries matching that site.
- Display the entries in ascending order of WaitlistNumber.

```
SELECT * from waitlist WHERE siteId = '$Siteid' ORDER BY waitlistnum ASC;
```

- Upon
  - Click ***Edit*** button - run **Edit Waitlist** task**.**
  - Click ***Close*** button - go to **User Main Menu** form and run **Display User Main Menu** task**.**

EDIT WAITLIST

Abstract Code:

- User selects ***Edit*** button from **Waitlist Form**.
- Run **View Waitlist** task**.** Add two more input boxes to take *old waitlist number* ($owl) and *new waitlist number* ($nwl) as inputs. Let us call current waitlist number for each entry in the table ($cwl)
- Validate the input values are integers. Modify only those entires that belong to site which user has admin-control of.
- Click ***delete –*** If second box ($nwl) is filled, throw an error "Enter only one input in first box"

  Else:
  - delete the entry for which Siteid=$Siteid and waitlistnumber=$owl
  - For all entries where Siteid=$Siteid and Wlnumber > $owl, decrement wlnumber
  - User has to manually update available rooms number as this is a rare scenario.

```
DELETE FROM waitlist where siteid='$Siteid' and waitlistnum='$owl';
 UPDATE waitlist SET waitlistnum=waitlistnum-1 WHERE siteid='$Siteid' AND
waitlistnum>'$owl';
```

- Click ***change*** -
  - Throw Error if the given $cwl and $nwl are not positive integers or if any one of them is greater than length of the waitlist displayed.
  - Read the entry where Site-email=$Site-email && $WaitlistNumber = $owl
  - If $nwl < $owl :

    For all entries where (Siteid=$Siteid && $cwl >= $nwl && $cwl < $owl)

    (This would be decreasing order)

    Increment $cwl

```
DELETE FROM waitlist WHERE siteid='$Siteid' AND clientid='$Clientid';
DO
$do$
BEGIN
FOR i IN $owl-1..$nwl LOOP
UPDATE waitlist SET waitlistnum=waitlistnum+1 WHERE siteId = '$Siteid'  AND
   waitlistnum=i;
END LOOP;
END
$do$;
INSERT INTO waitlist (waitlistnum, siteid, clientid) VALUES ('$nwl', '$Siteid', '$Clientid');
```

    else :

    For all entries where (Siteid=$Siteid && $cwl >= $owl && $cwl < $nwl

    decrement $cwl

  - Insert the entry with WaitlistNumber=($nwl)

```
DELETE FROM waitlist WHERE siteid='$Siteid' AND clientid='$Clientid';
DO
$do$
BEGIN
FOR i IN $owl-1..$nwl LOOP
UPDATE waitlist  SET waitlistnum=waitlistnum -1  WHERE siteId = '$Siteid'  AND
    waitlistnum=i;
END LOOP;
END
$do$;
INSERT INTO waitlist (waitlistnum, siteid, clientid) VALUES ('$nwl', '$Siteid', '$Clientid');
```

- Run **View Waitlist** task.
- Click ***Close*** button -go to **User Main Menu** form and run **Display User Main Menu** task**.**

ADD HEAD OF HOUSEHOLD TO WAITLIST

Abstract Code:

- User selects ***Add to Waitlist*** from **Client report**.
- Get the site which the user has admin-control of, and clients ID proof num and type from **Client report**

```
SELECT siteid  FROM user WHERE username='$Username';
```

- Look for a matching entry with site id and ClientID . If it exists, throw an error and stay in **Client report** form.

```
SELECT * FROM waitlist WHERE siteid = '$Siteid' AND clientid = '$Clientid' ;
```

- Else, get the $Count of entries from WAITLIST table matching $Siteid.

```
SELECT COUNT(*) AS waitlistcount FROM waitlist  WHERE siteid = '$Siteid';
```

- INSERT on WAITLIST table with site email, clientId, and $WaitlistNumber= ($Count + 1), if count == 0

```
INSERT INTO waitlist (siteid, clientid, waitlistnum) VALUES('$Siteid', '$Clientid',
'$waitlistcount'+1);
```

- Run **Get Client Logs and Data** task

CLIENT SEARCH

Abstract Code:

- User enters search text and should select one of ***search by name*** button or ***search by ID*** button.
- If ***search by name*** button

    Lookup CLIENTS table with partial match on Name attribute

```
select a.clientid as "Client ID", a.idnumber as "ID Number", a.idtype as "ID Type", a.firstname as "First
Name", a.lastname as "Last Name",
a.phonenumber as "Contact" from
(SELECT c.*, b.count FROM client c,
(SELECT count(*) count from client WHERE lower(firstname) LIKE lower('%$searchString%') OR
lower(lastname) LIKE lower('%$searchString%')) AS b
WHERE lower(firstname) LIKE lower('%$searchString%') OR lower(lastname) LIKE
lower('%$searchString%')) a
where count <= 5;
```

    Else

    Lookup CLIENTS table with partial match on ID Proof Number attribute

```
select a.clientid as "Client ID", a.idnumber as "ID Number", a.idtype as "ID Type", a.firstname as "First
Name", a.lastname as "Last Name",
a.phonenumber as "Contact" from
(SELECT c.*,b.count FROM client c,
(SELECT count(*) count from client WHERE lower(idnumber) LIKE lower('%$searchString%')) AS b
WHERE lower(idnumber) LIKE lower('%$searchString%')) a
where count <= 5;
```

- Upon selecting one of the Client, go to **Client Report** form and run **Get Client Logs and Data** task with the $Clientid.

ADD OR DELETE SERVICES

Abstract Code:

- **Display User Main Menu** task displays the list of services that are active and inactive with a ***Delete*** button for active tasks and ***Add*** button for inactive tasks.
- If only one service is active, then disable the ***Delete*** button next to it.
- Get the details of the logged in user and the site which he has admin-control of .

```
SELECT siteid from user WHERE username='$Username';
```

- Get the service which was selected and the new status.
- Depending on the service selected go to the corresponding table. SERVICEFOODPANTRY for food pantry, SERVICESOUPKITCHEN for soup kitchen, SERVICESHELTERS for shelters, and SERVICEFOODBANK for food banks.
- Find the entry matching the $Siteid and change the $Status attribute based on input.

```
If $service = FoodPantry
UPDATE servicefoodpantry SET currstate='$status' WHERE siteid = '$Siteid';
If $service = SoupKitchen
UPDATE servicesoupkitchen SET currstate='$status' WHERE siteid = '$Siteid';
If $service = Shelters
UPDATE serviceshelter SET currstate='$status' WHERE siteid = '$Siteid';
If $service = FoodBank
UPDATE servicefoodbank SET currstate='$status' WHERE siteid = '$Siteid';
```

- Run **Display User Main Menu** task.

DISPLAY USER MAIN MENU

Abstract Code:

- Upon successful login, get the User details from USER table using $Username
- Get the $Siteid by reading USERS table.

```
SELECT siteid AS siteId FROM user WHERE username ='$Username';
```

- Find the Site details from SITE table using SiteId and display them.

```
SELECT email as “Site Email ID”, shortname as “Site Short Name”, phonenumber as
    “Contact”, streetaddress as “Street Address”,city as “City”, statename as “State”,
    zipcode as “ZIP” FROM site WHERE siteid = '$Siteid';
```

- Find the status of all services by reading the entry corresponding from each of SERVICEFOODPANTRY, SERVICESOUPKITCHEN, SERVICESHELTERS, SERVICEFOODBANK and display the services in active and inactive categories respectively with ***Add/Delete*** buttons next to them.

```
SELECT currstate AS foodPantryState FROM servicefoodpantry WHERE EXISTS (SELECT *
    FROM servicefoodpantry WHERE siteid = '$Siteid');
SELECT currstate AS soupKitchenState FROM servicesoupkitchen WHERE EXISTS (SELECT *
    FROM servicesoupkitchen WHERE siteid = '$Siteid');
SELECT currstate AS serviceShelterState FROM serviceshelter WHERE EXISTS (SELECT *
    FROM serviceshelter WHERE siteid = '$Siteid');
```

- Upon
  - Click ***View Inventory*** button - go to **Food Bank Status Form** and run **View Current Inventory**.
  - Click ***Item Search*** button- go to **Item Search Form**
  - Click ***Add*** / ***Delete*** button-run **Add Or Delete Services**
  - Click ***Add Client*** button- go to **Add Client Form** and run **Add Client** task
  - Click ***Add Items*** button– go to **Add Items Form**
  - Click displayed services to edit attributes of corresponding service.
  - Click ***Search Clients*** button- go to **Client Search Form**
  - Click ***View Waitlist*** button- go to **Waitlist form**
  - Click ***Logout*** button- run **Logout** task and go to **Global Reports Form**

LOGIN

Abstract Code:

- User enters *username* ($Username) and *password* ($Password) fields.
- If ***login*** is selected then
  - Validate data for both user *username* and *password* fields
  - Search USERS table for the $Username provided, get the entry details.

```
SELECT * FROM user WHERE username='$Username';
```

- If entry is found and password = $Password (I.e count == 1)

  Store login information as session variable $Username.

  Else:

  Display error message and redisplay **Login form**.

  If ***cancel*** button is selected go to **Global Reports Form**

ADD CLIENT

Abstract Code:

- User clicks on ***Add Client*** button from **User Main Menu** form.
- The user has to enter Client First Name, Last name, ID/Description, Phone # (Optional) as the input fields.
- Validate,
  - First Name, Last Name, ID and Description are valid varchar fields and within the character limits of their respective fields. Throw error otherwise
  - Phone number is a number of 10 digits or NULL
- Upon
  - Click ***Submit***
    - Validate mandatory fields and all input data, throw an error message if not valid.
    - Look up for the same client in the database using ID/Description as key.

```
SELECT COUNT(*) AS count FROM client WHERE idnumber = '$idNumber' AND idtype =
'$idType';
```

- If client exists (count != 0)

  throw error

  Else

  perform an insert in CLIENTS table with the given inputs and

```
INSERT INTO client(firstname, lastname, phonenumber, idnumber, idtype)
    VALUES('$firstname', '$lastname', '$phoneNumber', '$idNumber', '$idType');
```

- Goto **User Main Menu** form and run **Display User Main Menu** task.
- Click ***Cancel*** - Jumps to **User Main Menu** form without making any Update.

GET CLIENT LOGS AND DATA

Abstract code:

- The User can view Client Report when he selects a particular Client from Client Search.
- Once the User selects a particular Client, get the Client's ID .
- Read the client's details from CLIENTS table by $Clientid and populate report.

```
SELECT firstname AS "First Name", lastname AS "Last Name", idnumber AS "ID Number",
    idtype AS "ID Type" From client WHERE clientid = '$Clientid';
```

- Read the CLIENTLOGS table and get all entries which match with the Client details and sort them in descending order with timestamp.

```
SELECT timest AS "timestamp", description AS "Description" FROM clientlogs WHERE
clientid = '$Clientid' ORDER BY TIMEST DESC;
```

- Upon
  - Click ***Add To Waitlist*** button- run **Add Head Of Household To Waitlist** task**.**
  - Click ***Edit Client*** button- run **Edit Client** task.
  - Click ***Add Service log*** button- run **Add log** task.

EDIT CLIENT

Abstract code:

User clicks on ***Edit client*** button from the **Client Report** form.

- Client *First name*, *Last Name*, *ID Number, ID Proof Description*, *Phone Number fields* needs to be retrieved by using a Select on the CLIENTS table using ClientId.

```
SELECT firstname AS “First Name”, lastname AS “Last Name”, idnumber AS “ID Number”,
    idtype AS “ID Type”, phonenumber as “Contact” From client WHERE clientid = '$Clientid';
```

- The user can Edit any of the given fields and Submit. The fields which are not edited, hold old values.
  - Upon
  - Click ***Submit***:
    - ¡ Validate the input
    - ¡ Show appropriate error message if any validation fails
    - ¡ If input is valid, update the CLIENTS table with modified client information.

```
UPDATE client SET firstname= '$firstname',
lastname = '$lastname',
idnumber = '$idnumber',
idType = '$idtype',
phonenumber = '$phonenumber' WHERE clientid = '$Clientid;
```

- Click ***Cancel*** - Jumps to **User Main Menu** without making any update.
- Run **Get Client Logs and Data**.

ADD LOG

Abstract code:

- User clicks on ***Add Log*** button from the **Client Report** form.
- Take the description entered by User as input $description. Checkins at Soupkitchens and Food pantries are also handled by this task.
- Validate that the enter value of $description is varchar and within the character limit of the field definition. Throw error otherwise.
- Display three check-boxes to mark if it is bunk check-in one for each Male bunk or Female bunk or a Mixed bunk.

- On Click ***Save***: Validate the input and append it with ClientID details and $Timestamp and insert into CLIENT_LOGS table and run **Get Client Logs and Data**.

```
INSERT INTO clientlogs (clientid,  description) VALUES ('$Clientid',  '$description');
```

Timestamp is updated automatically.

- Decrement the number of bunks of the particular type for this site by one if any checkbox is marked. Throw error if number is already at zero.

```
If checkbox = malebunks
    Update serviceshelter SET malebunks = malebunks – 1 WHERE siteid = '$Siteid';
If checkbox = femalebunks
    Update serviceshelter SET femalebunks = femalebunks – 1 WHERE siteid = '$Siteid';
If checkbox = mixedbunks
    Update serviceshelter SET mixedbunks = mixedbunks - 1 WHERE siteid = '$Siteid';
```

- On Click ***Cancel*** - Jumps to **User Main Menu** form without making any update.

LOGOUT

Abstract code:

- User clicks on ***Logout*** button from the **User Main Menu** form.
- Invalidate the session and go to **Global Reports Form.**

VIEW CURRENT INVENTORY

Abstract Code:

- Lookup ITEMS table based on the current $Siteid

```
SELECT itemid AS "Item ID", name AS "Name", category AS "Category",
subcategory AS "Sub-Category", storagetype AS "Storage Type", numunits AS
"Units Available", expiry AS "Expiry Date" FROM item WHERE siteid='$Siteid';
```

- Upon:
  - Click ***Back*** button - Go back to **User Main Menu** form.

VIEW NUMBER OF SEATS

Abstract Code:

- Lookup SERVICESOUPKITCHEN based on $SiteId

```
SELECT seats_available AS "Seats Available", currstate as “status” FROM servicesoupkitchen
WHERE siteid = '$Siteid'
```

- Upon:
  - Click ***Back*** button - Go back to **User Main Menu** form.
  - Click ***Edit*** button - Run **Edit Seats Available** task

EDIT NUMBER OF SEATS

Abstract code:

- User clicks on ***Edit*** button from **Soup Kitchen Form**
- Read the new $NewSeatCount from input. Validate that the entered value of $NewSeatCount is a valid integer and non-negative. Number of seats can be upated even when service is inactive.
- Upon
  - Click ***Submit*** -

```
UPDATE servicesoupkitchen SET seats_available = '$NewSeatCount' WHERE siteid = '$Siteid';
```

  - Click ***Cancel*** - Jumps to **Soup Kitchen Form** without making any Update.

VIEW NUMBER OF BUNKS AND ROOMS

Abstract Code:

- Lookup SERVICESHELTERS based on $SiteId

```
SELECT malebunks as “Male Bunks Available”, femalebunks as “Female Bunks Available”,
mixedbunks as “Mixed Bunks Available”, roomsavail as “Rooms Available”, currstate as “Status”
        FROM serviceshelter WHERE siteid = '$Siteid';
```

- Upon:
  - Click ***Back*** button - Go back to **User Main Menu** form.
  - Click ***Edit*** button - Jump to **Edit Number Of Bunks And Rooms** task

EDIT NUMBER OF BUNKS AND ROOMS

Abstract Code:

- User clicks on ***Edit*** button from **Shelters Form**
- Read the new number of bunks $NewMaleNumBunks, $NewFemaleNumBunks, $NewMixedNumBunks and $NewNumRooms from input.
- Upon
  - Click ***Submit*** -

```
UPDATE serviceshelter
        SET malebunks = '$NewMaleNumBunks',
        SET femalebunks = '$NewFemaleNumBunks',
        SET mixedbunks = '$NewMixedNumBunks',
        SET roomsavail = '$NewNumRooms'
        WHERE siteid = '$Siteid';
```

  - Click ***Cancel*** - Jumps to **Shelters form** without making any Update.

VIEW ALL OUTSTANDING REQUESTS (FOR A SITE)

Abstract Code:

- Run the ***View All Outstanding Requests List*** task by $Siteid

```
SELECT
itemid as "Item ID",
name as "Item Name",
category as "Item Category",
subcategory as "Item Sub Category",
expiry as "Item Expiry Date",
numunits as "Num Available",
quantity as "Num Requested",
username as "Requesting User"
FROM item NATURAL JOIN request
WHERE request.destsiteid = '$Siteid' AND request.reqstate = 'pending';
```

- Upon:
  - Click ***Back*** button - Go back to **User Main Menu**.
  - Click ***Accept/Modify*** button - Jump to **Accept/Modify Incoming Requests** task

ACCEPT OR MODIFY OUTSTANDING REQUESTS

Abstract Code:

- Read all incoming requests based on $Siteid from REQUESTS table.
- Add an ***Accept, Modify*** and a ***Decline*** button to the View Outstanding Requests view.
- Disable the ***Accept*** button for all cases where the $NumUnits in REQUESTS table is greater than $NumUnits in ITEMS table for the same the same item.
- Upon:
  - Click ***Accept*** button for a request with request id as $ReqId:

```
-- Accept
SELECT itemid, numunits FROM request WHERE requestid = '$ReqId';
-- These values are stored as $Itemid and $NumUnits
UPDATE request
        SET reqstate = 'closed'
        WHERE requestid = '$ReqId';
UPDATE item
        SET numunits = numunits - $NumUnits
        WHERE itemid = '$ItemId'
        AND siteid = '$Siteid';
```

  - Click ***Modify*** button:
    - ¡ Make the number of Items editable.
    - ¡ Take the number of items ($NumUnits) as input for the request with request id as $ReqId.

```
SELECT itemid FROM request WHERE requestid = '$ReqId';
-- This value is stored as $ItemId
UPDATE request
        SET quantity = quantity - $NumUnits
        WHERE requestid = '$ReqId';
UPDATE item
        SET numunits = numunits - $NumUnits
        WHERE itemid = '$Itemid'
        AND siteid = '$Siteid';
UPDATE request
        SET reqstate = 'closed'
        WHERE requestid = '$ReqId';
```

- Click ***Decline*** button for the request with request id as $ReqId. :

```
-- DECLINE
UPDATE request
        SET reqstate = 'closed'
        WHERE requestid = '$ReqId';
```

- Click ***Back*** button - Jump to **View All Outstanding Requests List** task

ITEM SEARCH

Abstract Code:

- Run the ***Item Search*** task based on $SiteId, $ItemQuery, $ExpiryDate, $StorageType, $Category, and $SubCategory. Variables that are not entered by the user, will hold 'NULL'

```
SELECT itemid as "Item ID", name as "Item Name", category as "Item Category",
subcategory as "Item Sub Category", expiry as "Item Expiry Date", numunits as "Num Available",
siteid as "Site ID" FROM item WHERE siteid = '$SiteId' OR lower(name) LIKE lower('%$ItemQuery%') OR
expiry = '$ExpiryDate' OR storagetype = '$StorageType' OR category = '$Category' OR
subcategory = '$SubCategory';
```

- If no items are found, display and error message saying “No Items match search criteria. Try a different one” and stay on the Item Search form.
- ***Request*** button will not appear against items where $User→SiteId = $SiteId. User can edit the $NumCount Value of Item here and click ***Save*** to save the $NumCount value in the ITEMS table for ITEM with item id as $ItemId

```
-- Update NumCount
UPDATE item
        SET numunits = '$NumCount'
        WHERE itemid = '$ItemId';
```

- Upon:
    - Click on ***Request*** button. -
        - run **Request Items** task.
    - Click ***Clear*** button - Jump to new **Item Search** task

GENERATE AVAILABLE BUNKS AND ROOMS REPORT

Abstract Code:

- Run the **Generate Available Bunks And Rooms Report** task
- Display for all $Siteid, the $SiteShortName, $SitePhysicalLocation, $SitePhoneNumber from SITES table and $Category, $HoursOfOperation, $ConditionsOfUse and $bunks_avialable and $rooms_availabe from SERVICESHELTER Table.

```
SELECT siteid AS "Site ID", currstate AS "Current State", description AS "Description",
conditions AS "Conditions", starttime AS "Start Time", endtime AS "End Time",
makebunks AS "Male Bunks", femalebunks AS "Female Bunks", mixedbunks AS "Mixed
Bunks", roomsavail AS "Rooms" FROM serviceshelter where currstate='active'
AND (malebunks != 0  OR femalebunks !=0 OR mixedbunks!=0 OR roomsavail !=0);
```

- If the above query returns zero rows, display - “Sorry, all shelters are currently at maximum capacity.”
    - Upon
        - Click ***Back*** button - Go back to **Global Reports Form** .

GENERATE AVAILABLE MEALS REPORT

Abstract Code:

- Create a temporary table which lists all the subcategories. Left join this table with the items table so that the resulting table will have only those items which belong to these subcategories.
- In the resulting table replace all the 'meat/seafood' items and 'dairy/eggs' items with a single type as they are basically considered same when calculating meals available.
- Reduce this table such that it just lists only the subcategories and counts of the items.

```
DROP TABLE IF EXISTS foodItemSubCategory;
CREATE TEMP TABLE foodItemSubCategory as
SELECT 'nuts/grains/beans' as subcategory
UNION
SELECT 'vegetables'
UNION
SELECT 'meat/seafood'
UNION
SELECT 'dairy/eggs';

drop table if EXISTS temp1;
CREATE TEMP TABLE temp1 AS
select d.subcategory,count(*) as total_sum
FROM foodItemSubCategory d
LEFT JOIN
 (select * from ITEM WHERE  category = 'food' and expiry >=CURRENT_DATE  )
 c on c.subcategory=d.subcategory::subcategories
GROUP BY d.subcategory;

update temp1
SET  subcategory = 'meat/seafood or dairy/eggs'
WHERE subcategory = 'meat/seafood' or subcategory = 'dairy/eggs';

drop table if EXISTS temp2;
CREATE TEMP TABLE temp2 AS
select subcategory,SUM(total_sum) as total_sum
FROM temp1
GROUP BY subcategory;
```

- From the reduced table, get the minium count and inner join this with the original table so that it gives out all those categories where the count is equal to minimum count.

```
SELECT a.total_sum  as maxMealsPossible ,subcategory  as subcategoriesneeded
FROM temp2 as a
INNER JOIN  (SELECT MIN(total_sum) as total_sum FROM temp2 ) as b ON
a.total_sum=b.total_sum;
```

- Upon:
  - Click ***Back*** button - Go back to **User Main Menu** task.

CATEGORIZE AND INSERT ITEMS FROM DONATIONS

Abstract Code:

- User selects ***Add Items*** button from **User Main Menu** form.
- Populate the drop down boxes for Item Category and subcategory
- From the **Add Items Form**, read all the Item attributes
- If $ItemExpiry is in the past, throw error.
- On Click ***Add*** –
  - Lookup ITEMS table for matching item with (sub-category, name, expiration date)

    If a match is found

    increment the number of items for that entry by given input.

```
UPDATE item SET numunits=numunits+'$Num' WHERE subcategory='$Subcateg'
AND name='$Name' AND expiry='$Expdate';

-- (If this query returns  'UPDATE 0' then)
```

    Else :

    add a new entry in ITEMS table, with the given attributes.

```
INSERT INTO item(name, category, subcategory, storagetype, numunits, expiry,
siteid) VALUES('$Name','$Category', '$Subcategory', '$Storagetype', '$Num',
'Expdate', '$Siteid');
```

  - Upon

    Click *Close* button - go to **User Main Menu** form and run **Display User Main Menu** task.

VIEW ALL REQUESTS OF A USER

Abstract Code:

- Lookup REQUEST table for all entries where $Username matches with Current user's $Username.

```
SELECT requestid AS "Request ID", itemid AS "Item ID", reqstate AS "Status", quantity AS "Quantity Requested", destsiteid as "Destination Site ID", username as "Username" FROM request WHERE username='$Username'
```

- Display $Num, Item details and $Status from REQUESTS Table.
- Upon:
  - Click ***Back*** button - Go back to **User Main Menu** form.

REQUEST ITEMS

Abstract Code:

- User clicks on ***Request*** button from **Item Search Form**
- Popup an input screen asking for the number of items to request - take the input as $Num and read item selected as '$Itemid and its corresponding site '$Siteid'. Validate that $Siteid is valid and $Num is positive.
- Insert into requests table with the obtained values.

```
INSERT INTO request(itemid, reqstate, quantity, destsiteid, username)
VALUES('$Itemid', 'pending', '$Num', '$Siteid', '$Username');
```

- stay on **Item Search Form**