Project Part 3: Draft
CS-GY 6083 sections A and INET Fall 2024
Prof Frankl

In part 3 of the project, you will use **the table definitions posted** to implement application code for WelcomeHome as a web-based application.

Application Coding Requirements:
You may use Python, PHP, Java, node.js, or Go for your application logic (backend) code. If you'd like to use some other language, check with the TAs by 11/17/2024. *You must use prepared statements if your application language supports them.* You **may not** use an ORM (Object-Relational Mapper) – your code must use SQL directly to interact with the database.

You must use SQL reasonably efficiently: For example if a feature requires data from multiple tables, use appropriate JOIN operations. (For example, **Do not** query T1 and then use application code to iterate through the results plugging them in to queries from T2 when instead you could use T1 JOIN T2).

You may use React or similar front-end development tools, but remember that the emphasis on the project is on the interactions with the database – a rudimentary front end that allows the required features will be fine.

Part 3 of the project will be **evaluated primarily** on your WelcomeHome application's **back-end**, i.e., interactions with the database, handling typical cases correctly and reasonably efficiently, handling corner cases, checking constraints that are not enforced by the DB, handling exceptions due to enforced DB constraint violations, etc. The front-end (user interface) needs to be web-based and usable, but does not need to be fancy or beautiful – this is not a web interface design course.

Necessary Security Mechanisms: You should have the necessary mechanism to prevent SQL Injection and XSS (cross-site script) attacks. Sample instructions will be posted. In the real-world, an application like this would use https to authenticate the server and encrypt traffic between the user's browser and the server, but for this project we will just use http.

Getting started: *We have supplied Python/pyMySQL/Flask code for a sample application that includes most of the constructs you'll need in order to do this assignment, along with videos that explain the code in detail. If you do not have prior experience with web and/or database application programming, you should study these and you may modify the provided sample code to implement WelcomeHome.*

Teams: You may continue to work alone or with the one or two others with whom you worked on parts 1 and/or 2 or you may form a team (two or three students, total): Any new teams or additions to a team must be reported to the TAs by 11/27/2024. Everyone on a team is expected to understand all the code.

REQUIRED APPLICATION FEATURES:

WelcomeHome should support the following use cases:

1. **Login & User Session Handling:** Only registered users are allowed to **log in** to WelcomeHome. You may either provide a registration function where users supply their information (including roles – staff, volunteer, etc) or you may assume that this information is pre-loaded in the database.
   Passwords should be stored using a cryptographic hash function, like SHA-256, with salt added before hashing.
   When a user tries to login, if the salted and hashed password they enter matches the record in the database, the user should be then redirected to the main page and relevant data should be stored in session variables. If the password doesn't match, the user should be informed of the failure and no session should be created.

2. **Find Single Item:**
   a. prompt the user to enter an itemID.
   b. return the locations of all pieces of that item.
3. **Find Order Items:**
   a. prompt the user to enter an orderID.
   b. Return list of the items in that order, along with the locations of all pieces of each item.
4. **Accept Donation:**
   a. Check that the user is a staff member
   b. Prompt the user for the donor's ID and check that they are registered as a donor
   c. Prompt the user for relevant data about the items donated, the donor, and the locations where the (pieces) of the donated item will be stored and record all of this data in the appropriate tables. You may either use auto-increment to assign itemIDs or enter itemIDs and check that they are unique.

Additional Features: In addition to the required features, you must implement implement **2* numberOfGroupMember** additional features. (That is, 2 additional features if you're working individually; 4 for a group of two; and 6 for a group of three). Here are some possibilities. If you have other ideas of similar level of complexity and relevance to WelcomeHome, you may propose them via Ed Discussion at least one week before the due date. These specifications are not spelled out in detail. You should use common sense and an understanding of the intention of the application to work out details of what needs to be updated when changes are made.

5. Start an order
    a. Check that logged in user is staff
    b. Prompt for username of client and check that it's OK
    c. Assign an order ID and save it in a session variable
6. Add to current order (shopping)
    a. Use a drop-down menu or similar interface for user to choose certain categories and subcategories
    b. Show items belonging to these categories that are available (not already ordered)
    c. Select an item, add it to the order, and mark it to indicate that it's been ordered

7. Prepare order
    a. Prompt for an order number or search by client's username
    b. Update the location of the items in an order to show that they're ready for delivery. You may have a designated "holding location" for items waiting for delivery. Once items are in this status, they should no longer be available for clients who are searching for items.

8. User's tasks: Show all the orders the current user (the user who is logged in) has a relationship with (e.g. as a client, a volunteer working on the order, etc, along with some relevant details)

9. Rank System! (you only need to implement either a or b to complete this task)
    a. Have a rank system which records which volunteer participates in the most tasks in a given time period.
    b. Have a rank system which records the most popular category/subcategory having the most number orders in a given time period.

10. Update enabled!
    a. Check if the current user is delivering or supervising an order and, if so, allow him/her to change the status of the order

11. Prepare data for a year-end report with information like the number of clients served, the number of items of each category donated, and some summary data about how clients were helped (the kind of information that you might want to include in a grant application).

12. Handle duplicate items (This will count for four features)
    a. Modify the database so that there can be multiple "copies" of the same item; show how this changes the ER and the schema, and note any assumptions you're making
    b. Update descriptions of features 2, 6, and 7 and implement them to handle multiple copies of items

Handin and Evaluation:

When the project is due, you will share your code with the course staff via GitHub and each team will meet with the TAs to demonstrate their project on a series of tests and show some of their code. All team members should be familiar with all aspects of the code and each team member should take responsibility for two of the additional features.

In addition to the code, you should prepare a brief report including

- Languages and frameworks you used
- Any changes you made to the schema, and their purpose
- Any additional constraints, triggers, stored procedures, etc;
- The main queries for each of the features you implemented. Show these as plain SQL, either with place-holders or dummy values for values that are being filled in based on users' choices (similar to the argument you would pass to execute a prepared statement).
- Any difficulties you encountered, lessons learned, etc
- Which team members did what.

This report can be written in bullet point format and should not be more than a few pages long.