Documentation for Aurora Food Pantry Project

All projects are expected to produce working code and a set of documentation that describes:

**The Class Architecture:**

1. Our Person Class is the Super class to our Admin, Volunteer, and EmployeeClass classes has holds the basic information that any person object stored in our database should have. These classes are utilized by DBAdmin, DBVolunteer, and DBEmployee to upload to the MySQL database server.
2. We have a series of Controllers and Fxml files that we swap scenes in order to form a working GUI that can manipulate Database and Data objects

**The Database Architecture:**

Our section of the database has its own schema which holds three different table: admin, employees, and volunteers. These tables hold the various data of users/persons in their respected table. Each table holds information based on our Person class. Our Person class consists of these data fields: first name, middle initial, last name, email, phone number, gender, birth date, address, and emergency contact. The different tables each hold ID number, username, and password. Employees have also working hours, and Volunteers have volunteering hours and whether they are court ordered to volunteer.

**How The Data Structures/Sort and Search Methods Are Used:**

1. Queues - Implemented in the QuickSort Method for least hours.
2. Arrays - Most of our objects are transferred from the HashMap Storage to an array before it is sorted via a sorting method.
3. ArrayLists - The Search by Last Name buttons implementation stores any object with matching last name into an arraylist which is displayed in the bottom text area in the Admin Scene.
4. Linked Lists - In the Admin Scene when Sort By ID data is stored into a LinkedList before it is displayed and then the LinkedList is displayed on the table view.
5. Stacks - Are used in the implementation Sorting by Most Hours data is displayed in the table from a stack.
6. Hashes - A HashMap is what we store our Admin, Employee, and Volunteer objects that are retrieved from the tables in our database.
7. Iterator - Used for iterating through hashmap key sets when going through a hashmap.
8. Comparator is extended by all subclasses of person and person to allow for the comparing of separate data fields in objects.
9. Binary Search - used for searching for a specific ID.
10. Bubble Sort - used for sorting by Last Name.
11. Merge Sort - used for sorting by Most Hours.
12. Quick Sort - used for sorting by Least Hours.
13. Heap sort - used for sorting by First Name.

**A description of the role and code developed by each person;**

1. Angel Vivanco - Was responsible for documenting conversations over Discord group chat Developed and fully implemented the Volunteer Scene in the GUI. He also Implemented Quick Sort in Admin scene to sort by Least Hours.
2. Rachel Skonning - Group leader -> kept us on task and made sure that our project was fulfilling the requirements laid out by our instructor. She developed all of the code that links the database and tables to our GUI. Created the tableview to display the data from the database. Also, implemented the Binary Search used to search for an Employee by ID.
3. Luke Guadiano - Documentation/Commenting of code -> add comments to code and help other group members understand the workings of everyone else's code. Created the Employee Class and implemented the Sort By Most Hours button in the Admin Scene.
4. Ben Jaynes - Tasked with developing a template GUI, implementing a lot of the working functions of the GUI, and scheduling meet up times via real world and/or discord group chat because we all had conflicting schedules -> Developed most of the GUI and fully implemented Employee scene, Working Hour scene, and part of the Admin scene.