Food-For-You

Software Design Specification

Linnea Gillius, Krishna Patel, Jerry Pi, Katherine Smirnov, Lauren Van Horn | 3-12-2023 | v4

1. SDS Revision History

Date	Author	Description
2-16-2023	Everyone	Created Initial SDS
2-23-2023	KP, LVH	Expanded on initial SDS
2-26-2023	KP, LVH	Updated UML Diagrams
3-12-2023	KP, LVH	Updated Design to reflect final state of system

2. System Overview

The Food-For-You project will allow users to check needs of donations and availability of food from a food bank as well as keep an up to date food resource database for food bank staff members to keep track of incoming and outgoing food products. This project will have five modules—Recipient Interface, Donor Interface, Staff Interface, Admin interface and Food Resource Database. The Food Resource Database will store the current items that are available in each food bank. The Food Bank Staff Interface will be able to connect and display the values within the Food Resource Database and perform updates to it. Both the Food Donor Interface and the Food Receiver Interface will interact with the Food Resource Database and retrieve data from the database. Donors will be able to donate to a food bank using the Food Donor Interface with impactful background knowledge and a receiver can register for food items within the food banks from the Food Receiver Interface.

3. Software Architecture

3.1. Components

1. **Food Bank Staff Interface Module**
    a. Will allow the addition, deletion, and modification of database entries. The user will be prompted to select the options they require using the interface. Using these inputs, SQL queries will be generated to interface with the database. Then the queries will be sent to the MySQL database in order to modify entries in the food_item table and the changes will update on the screen.
2. **Food Recipient Interface Module**
    a. The food recipient interface module will give the user access to information on what food banks have particular food resources. The user will be able to specify their data request with dropdown filters such as food category and neighborhood. If these dropdowns aren't specified, it will default to showing all food and/or all neighborhood food resource availability. The user can further specify what data they want displayed through four different checkboxes—Phone number, Street Address, Hours, Open Now. With these features, the system will display the additional information requested to the user. Once the user has submitted their search request using the interface, a query will be made to the database and the results will be displayed in both the standard output screen as well as saved in a .txt file for the user to be able to print if needed. The output

will be displayed in descending order of food banks and their quantity availability.

3. **Food Donor Interface Module**
    a. This interface will allow a donor to quickly see what food pantries/food banks are low on so they can donate their extra food. The user can further specify what data they want displayed through four different checkboxes—Phone number, Street Address, Hours, Open Now. With these features, the system will display the additional information requested to the user. A query will be made to the database after the user submits their input and the results will be displayed in both the standard output screen as well as saved in a .txt file for the user to be able to print if needed. The output will be displayed in ascending order of food banks and their quantity availability.
4. **Food Resource Database Module**
    a. This module will store all the data written and viewed by the other modules. It will contain the most up to date information with quantities of food and location. There are four tables—food_item, food_bank, hours, and outgoing. The food_item table stores the food category, food quantity, unit type, and food bank location. The food_bank table stores food bank id, phone number, street address, neighborhood, and location name. The hours table stores the open and close times for each day of the week as well as the food bank id. Lastly, outgoing is a table that stores the same information as the food_item table, except it tracks all the outgoing food that has been passed out.
5. **AdminView Interface Module**
    a. This module will allow the addition of new food banks with a file upload option. A file would be opened using a system dialogue opened by an "upload file" button. This file would then be validated and SQL queries would be generated to update the database with this new information. This module will also have fields required to insert a new food bank such as name, address, and hours of operation. This would generate a separate query to save to the database. Another tab will allow the administrator to see what food has been distributed to food recipients using data from an "outgoing table". The data shown will be item name, quantity, units, and location. An "export" button will create a .csv file that will be populated with the result from an SQL query with every entry in the outgoing table.
6. **Util Helper File**
    a. This module will store the connection functions used to initialize and maintain an interface with the MySQL database. It also has some

commonly repeated functions, along with theming implementations to keep a consistent style across the donor, staff, and recipient interfaces.

7. **Time Picker Helper File**
    a. This module includes a time picker that allows the user to select a time from a time range. The format of the time picker corresponds with the *hours* table in the Food Resource Database Module, and it can be accessed by the AdminView Interface Module to set for new times.

3.2 Interactions

The AdminView Module can be considered the "administration" interface and the inputs directly affect multiple tables in the database. Both the Food Donor Interface and Food Recipient Interface will interact with the database but will not modify any existing entries and will simply read. However, the interfaces will both dynamically update when changes are made by other interfaces. Lastly, the staff interface module is the only other one that can write to the database, but will only be able to modify two of the tables. The inputs made to insert, delete, modify, and move items will directly affect the database.

3.3 Design Rationale

The rationale behind creating these five components was mostly for ease of use, however it also allows for modularity. This allows for easy modification and addition of features for the four use cases. This would allow us to eliminate some potential problems that may arise from implementing additional features. The current design also allows for only the food bank to adjust the database ensuring some level of security. The modules also hide the SQL queries being made from the user so unintentional modifications to the tables in the database are minimized.

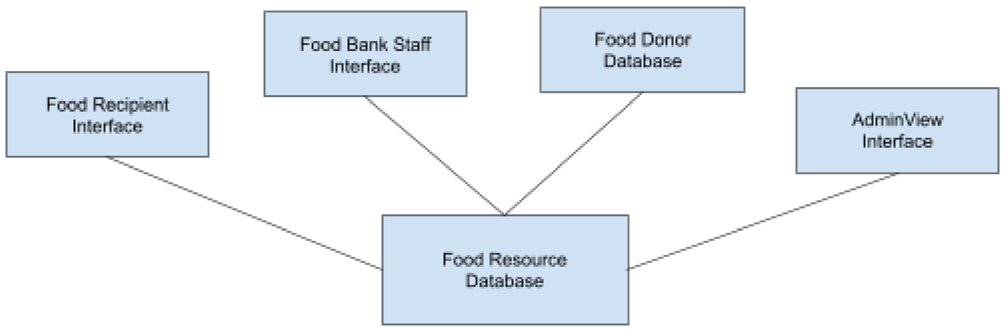


Figure 3.3 System Architecture

4. Software Modules

4.1. Food Bank Staff Interface Module

The module's role and primary function.

The purpose of the Food Bank Staff Interface module is to facilitate the interaction by an administrator with the database. This interface will be the only one that can change data in the SQL database. It will allow for the addition of items of food that are needed and their removal as they are received.

Interface Specification

This module will interact with the database module to write, delete, and read information. The staff interface module will generate a query based on input from the GUI and then make a connection to the database module to carry out the requested operation. **Please see figure 5.2, 5.3, and 5.4 for sequence diagrams of this behavior.**

Static Model:

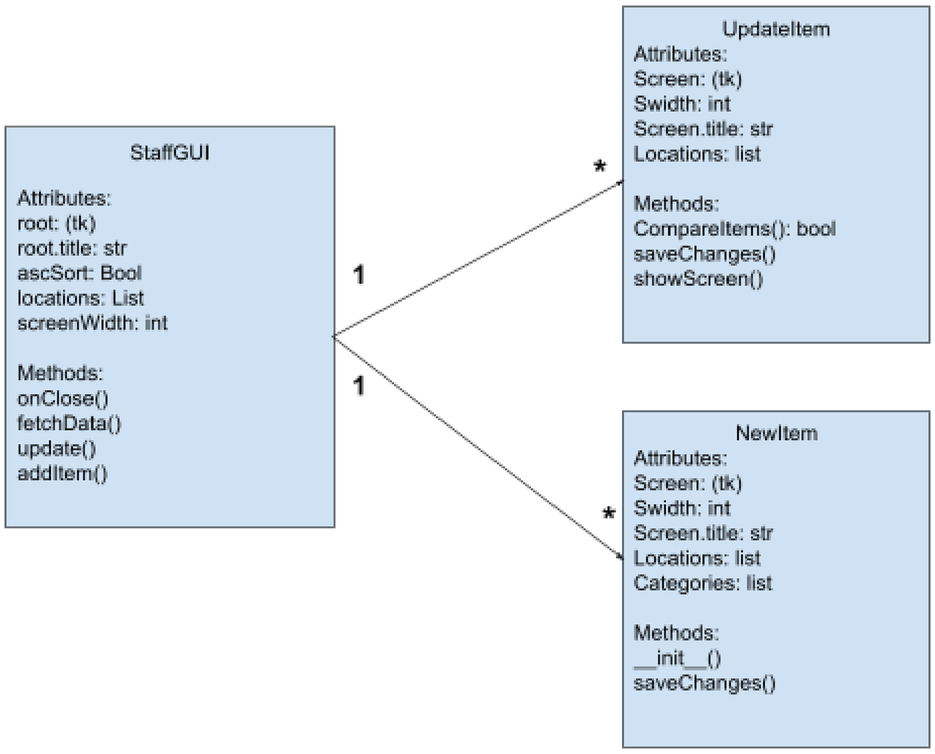


Figure 4.1.1 Food Bank Staff Interface Class Static Model

See Kieras UML Notation and Lecture Notes for key

Dynamic Model:

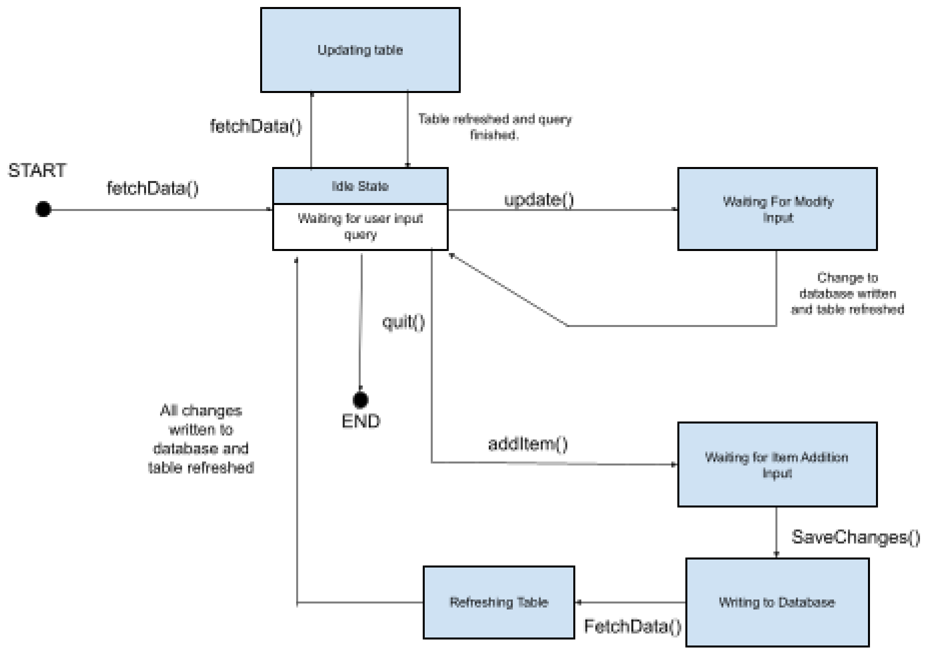

Updating table
fetchData()
Table refreshed and query finished.
START
fetchData()
Idle State
Waiting for user input query
update()
Waiting For Modify Input
Change to database written and table refreshed
quit()
END
All changes written to database and table refreshed
addItem()
Waiting for Item Addition Input
SaveChanges()
Refreshing Table
FetchData()
Writing to Database

Figure 4.1.2 Food Bank Staff Interface Class Static Model

Design rationale

The Food Bank Staff Interface has been separated from the Donor and Recipient interfaces because only the food bank staff should be able to adjust the entries in the database. Additionally, much of the functionality in this interface is irrelevant to recipients and donors.

4.2 Food Recipient Interface Module

The module's role and primary function.

The Food Recipient Interface will allow a user who needs access to food resources to locate which food pantries have the items they need. It will allow them to filter by category and location to see which food pantry has the resources they need.

Interface Specification

The Food Recipient Interface module will only interact with the database module. The options the user selects on the interface will generate a corresponding SQL query which will be sent to the database. The database will then process the query and return the result to the Food Recipient Interface. **Please see figure 5.1 for a sequence diagram of this behavior.**

Static Model:

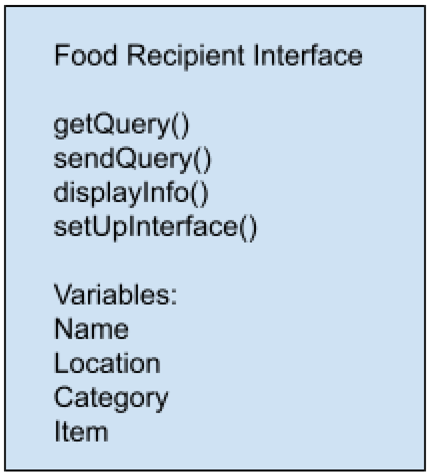


Figure 4.2.1 Food Recipient Interface Class Static Model

Dynamic Model:

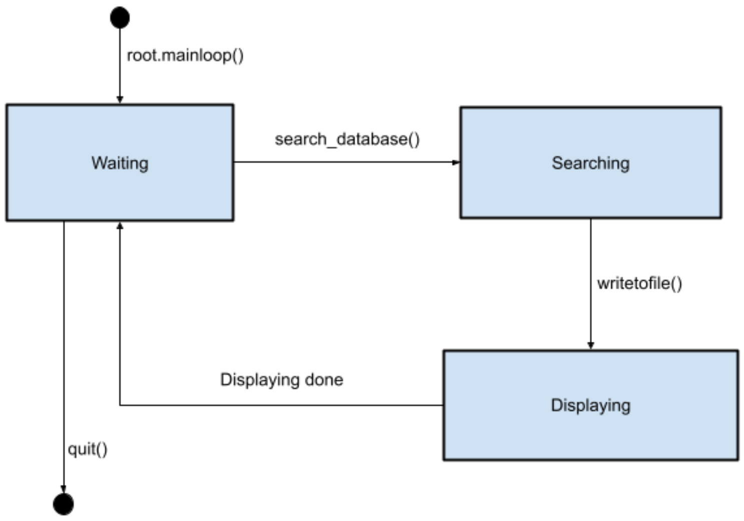


Figure 4.2.2 Food Recipient Interface State Dynamic Model

Design rationale

The Food Recipient Interface Module is separate from the other interfaces to allow for easy viewing by users. Since this application is meant to streamline the process of food distribution, adding functionality from the donor module to this one would make it hard to use. Having this interface be standalone also allows it to be more tailored for this use case.

4.3 Food Donor Interface Module

The Food Donor Interface role and primary function.

The Food Donor Interface is the module that interacts with solely Food Donor users. Its primary function is to take in queries from the user that focuses on which Food Banks need more of particular food categories. In this manner, it will help guide food donors in the area with suggestions of where food donations would make the most impact and what food is in greatest demand..

Interface Specification

The Food Donor interface will only interact with the Food Resource Database Module. The Food Donor Interface will take in a query from the food donor and send it to the food resource database module to pull the appropriate data and present it back to the user. **Please see figure 5.1 for a sequence diagram of this behavior.**

A Static Model

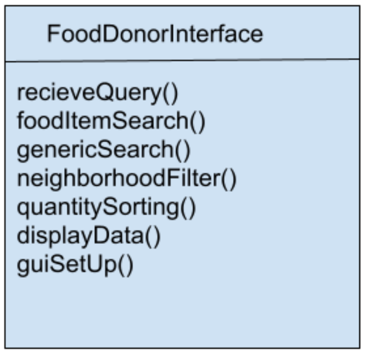


Figure 4.3.1 Food Donor Interface Class Static Model

A Dynamic Model

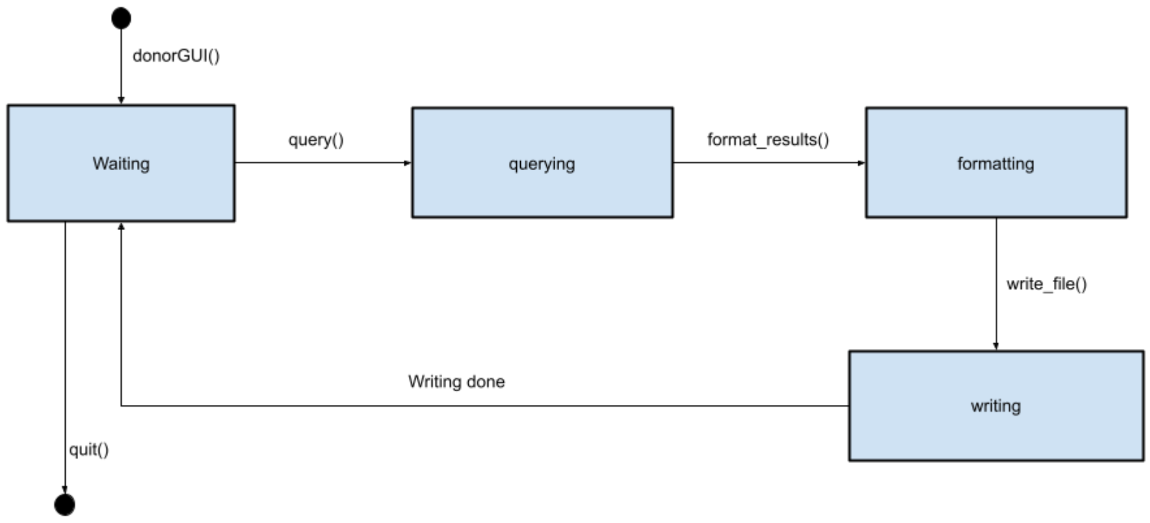


Figure 4.3.2 Food Donor Interface State Dynamic Model

Design rationale

We chose to separate the Food Donor and Food Recipient interfaces into two modules so the interface can more closely reflect the needs of the user group. The Food Donors are only interested in what is *lacking* in food banks while food recipients are in search of food that is *available*. Splitting up these two modules allows more detailed interfaces to increase user accessibility and understanding of functions.

4.4 Food Resource Database Module

Food Resource Database role and primary function.

The Food Resource Database Module will be used to store all the information of what food is and is not available at different Food Banks. In the database, we will also store information about where the food bank is located and the quantities of food. This module is key in how the other modules interact.

Interface Specification

The Food Resource Database will interact with every other module.

- With the Food Bank Staff Interface Module, the Food Resource Database will take in new information about the food quantities and update the database to reflect those changes.
- With the Food Recipient Interface, the Food Resource Database will take in queries and sort through the data to return the desired output. One example includes a recipient looking for a certain category of food and wants to know what food banks have that food. The database would receive that query and present to the recipient a list of food banks.
- With the Food Donor Interface, the Food Resource Database will take in queries and sort through the data to return the desired output. These queries will focus on finding Food Banks with the lowest supplies of a certain food category but will also take into account neighborhoods/distance to travel.

Please see section 5 to see sequence diagrams of this behavior.

A Static Model

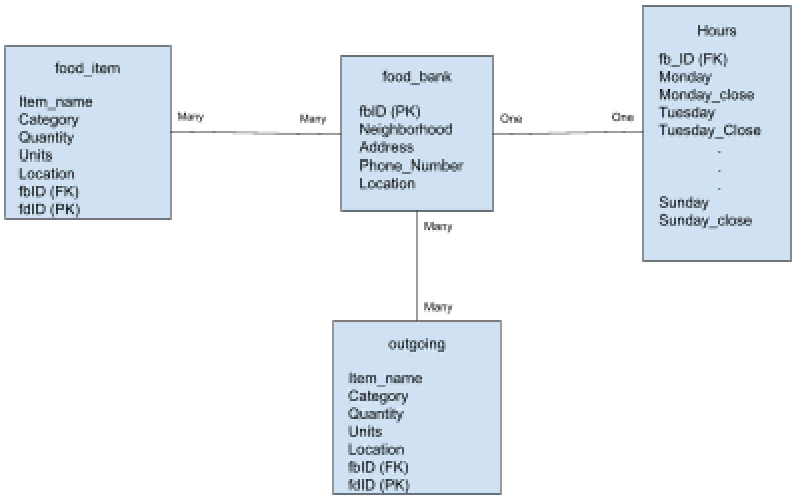


Figure 4.4.1 Food Resource Database Class Static Model

A Dynamic Model

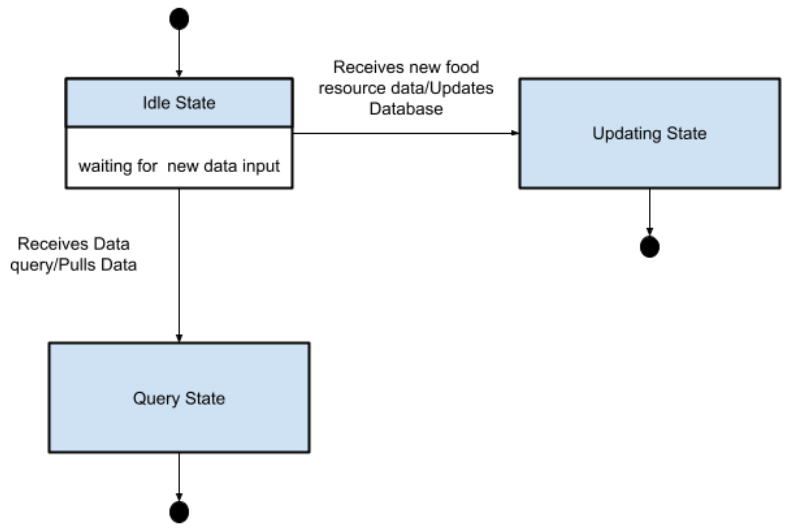

Idle State
waiting for new data input
Receives new food resource data/Updates Database
Updating State
Receives Data query/Pulls Data
Query State

Figure 4.4.2 Food Resource Database State Dynamic Model

Design rationale

By creating a Food Resource Database, we limit the number of modules that need to interact with each other. The only module that interacts with multiple other modules is the Food Resource Database. In this manner, it can serve as the central data access for all modules.

4.5 AdminView Interface Module

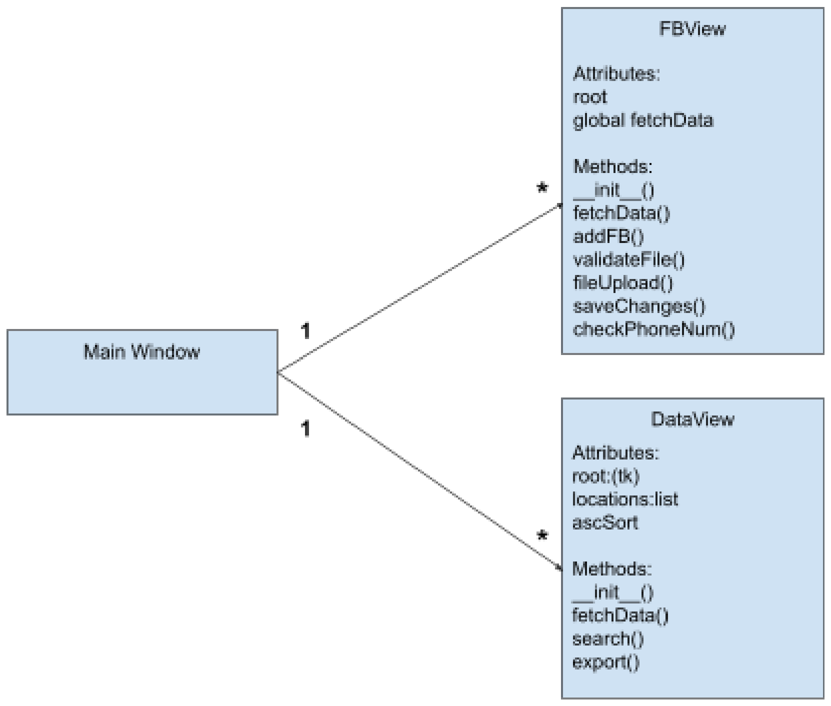
The module's role and primary function:

The purpose of the AdminView Interface Module is to allow a food bank administrator to add new food pantries and food banks to the database so food recipients can have the most up to date information. It will allow the insertion of a food bank/food pantry and initial food items at the same time. There is also a tab that allows for the viewing of what items have been given out so administrators can see what items are most in need.

Interface Specification:

This module will interact with the database to add a new food bank, new hours, and new food items. It will also read the database for the outgoing food table. This will all be done with SQL queries generated by user input. **Please see figures 5.4, 5.5, and 5.6 to see sequence diagrams of this behavior.**

Static Model



Dynamic Model

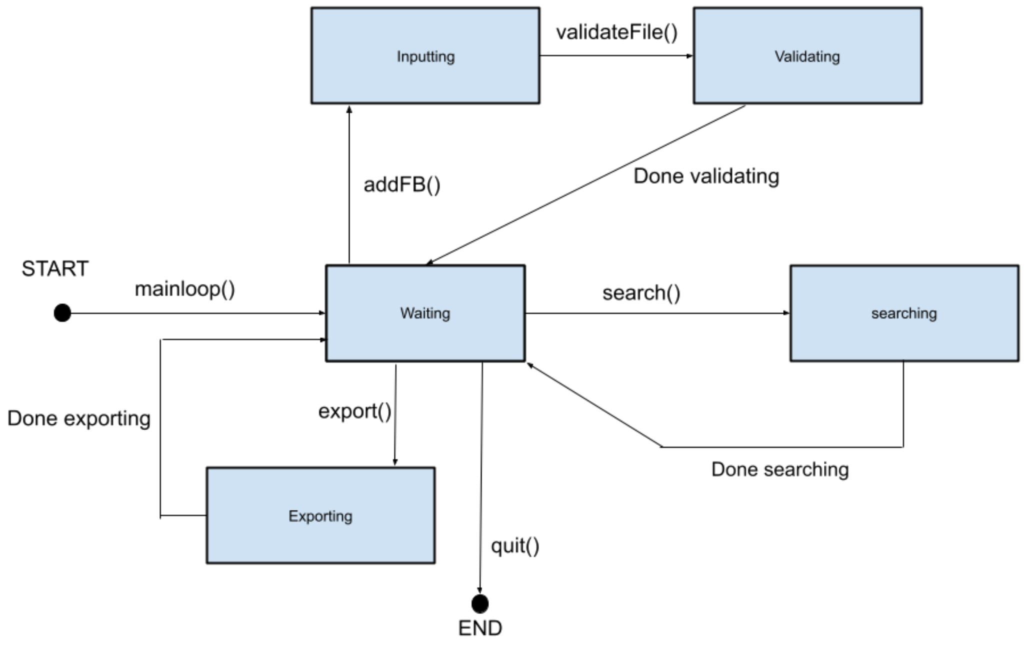


Figure 4.5.2 Food Bank Admin State Dynamic Model

Design Rationale

The AdminView module is its own independent module separate from the Staff Interface Module because elevated tasks are being performed which could impact the effectiveness of the system for food donors and recipients. If a staff member or volunteer accidentally adds a food bank, a donor or recipient may go to a food bank only for it to not exist, this module helps shield against that possibility, Additionally, keeping this interface as its own module helps food bank staff and volunteers as moving features to this module keeps the Staff Interface concise.

5. Dynamic Models of Operational Scenarios (Use Cases)

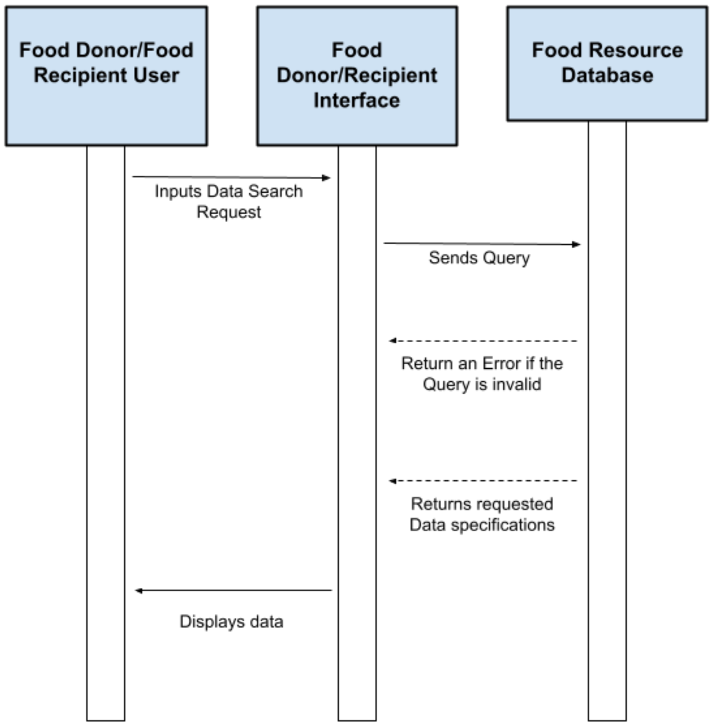
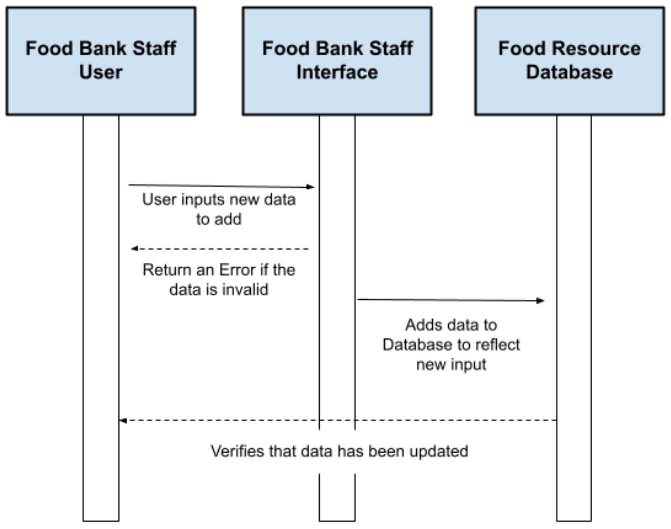

Food Donor/Food Recipient User
Food Donor/Recipient Interface
Food Resource Database
Inputs Data Search Request
Sends Query
Return an Error if the Query is invalid
Returns requested Data specifications
Displays data

Figure 5.1 Food Donor/Recipient Sequence Diagram

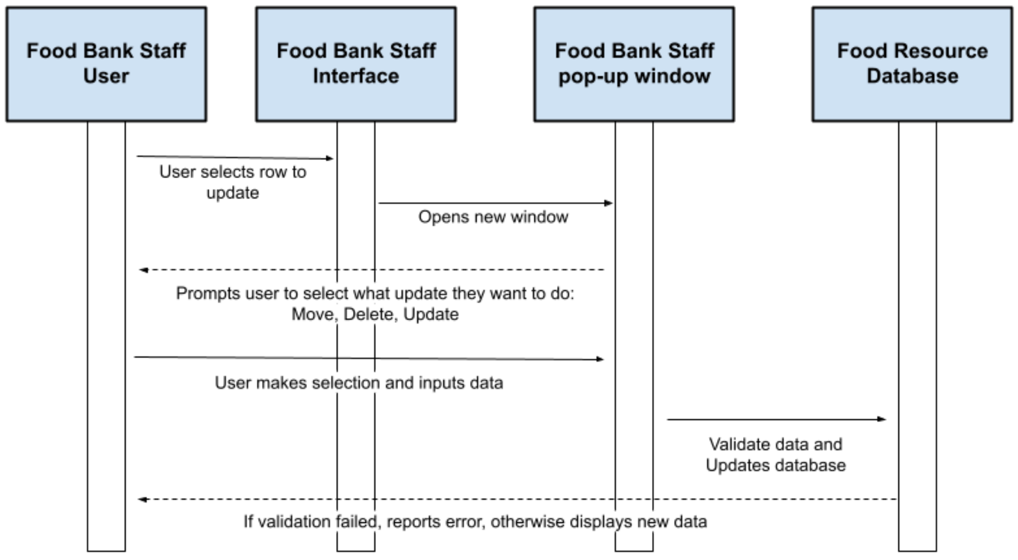
In Figure 5.1 we see a Food Donor or Food Recipient request a particular data set to be pulled. The food donor would be interested in seeing what food banks are lacking in certain food categories while the food recipient would be interested in seeing what food is readily available for pick up. The user can specify data based on itemDescription (i.e. Carrot, Soup, etc), location (street address of food bank), neighborhood, or food category (i.e. produce, dry goods).



Food Bank Staff User
Food Bank Staff Interface
Food Resource Database
User inputs new data to add
Return an Error if the data is invalid
Adds data to Database to reflect new input
Verifies that data has been updated

Figure 5.2 Food Bank Staff Sequence Diagram to add new item

In figure 5.2 we see one Food Bank Staff use case. In this case, the staff member has received new food resources and needs to insert the data into the database. In this case, the food item doesn't exist in the database yet.



Food Bank Staff User
Food Bank Staff Interface
Food Bank Staff pop-up window
Food Resource Database
User selects row to update
Opens new window
Prompts user to select what update they want to do: Move, Delete, Update
User makes selection and inputs data
Validate data and Updates database
If validation failed, reports error, otherwise displays new data

Figure 5.3 Food Bank Staff Sequence Diagram to update existing item

In figure 5.3 we see another Food Bank Staff use case. In this case, the staff member has received new food resources or handed out food resources and needs to update the database with the most up-to-date records available. In this case, the food item already existed in the database table.

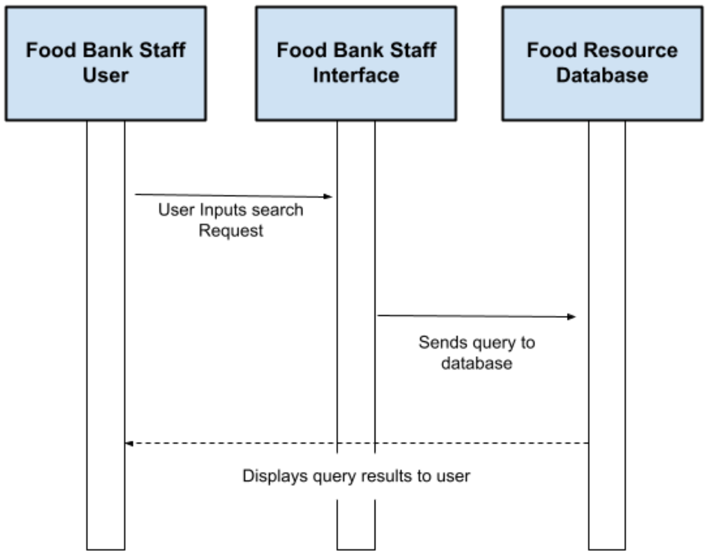

Food Bank Staff User
Food Bank Staff Interface
Food Resource Database
User Inputs search Request
Sends query to database
Displays query results to user

Figure 5.4 Food Bank Staff/Admin Sequence Diagram to search database

In figure 5.4 we see another Food Bank Staff and Food Bank Admin use case. In this case, the staff/admin member wants to search and display the current data in the database. The Food Bank Staff/Admin may want to do this to demonstrate how much food has been passed out between food banks or visualize what food banks have the greatest supply of food available/unavailable.

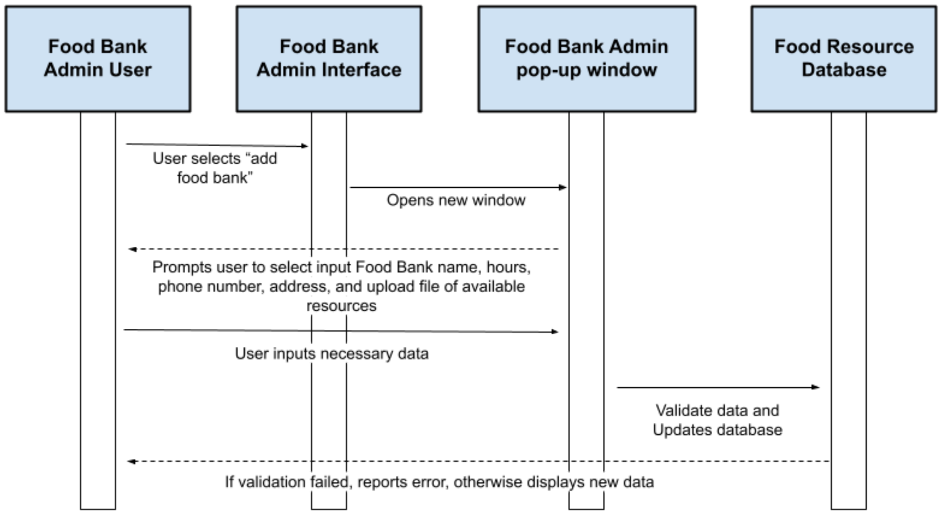

Food Bank Admin User
Food Bank Admin Interface
Food Bank Admin pop-up window
Food Resource Database
User selects “add food bank”
Opens new window
Prompts user to select input Food Bank name, hours, phone number, address, and upload file of available resources
User inputs necessary data
Validate data and Updates database
If validation failed, reports error, otherwise displays new data

Figure 5.5 Food Bank Admin Sequence Diagram to add new Food Bank to database

In figure 5.5 we see another Food Bank Admin use case. In this case, the admin member wants to add a new food bank to the database. The Food Bank Admin may want to do this because a new food bank has opened up.

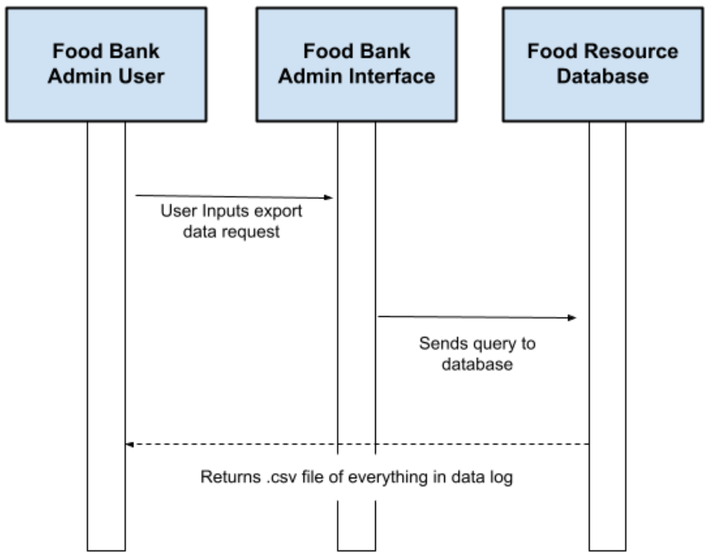

Food Bank Admin User
Food Bank Admin Interface
Food Resource Database
User Inputs export data request
Sends query to database
Returns .csv file of everything in data log

Figure 5.6 Food Bank Admin Sequence Diagram to export data log

In figure 5.6 we see another Food Bank Admin use case. In this case, the admin member wants to export data from the database. The Food Bank Admin may want to do this to keep a log of all food moved in and out of the food banks.

6. References

Faulk, Stuart. (2011-2017). CIS 422 Document Template. Downloaded from https://uocis.assembla.com/spaces/cis-f17-template/wiki in 2018. It appears as if some of the material in this document was written by Michal Young.

IEEE Std 1016-2009. (2009). IEEE Standard for Information Technology—Systems Design— Software Design Descriptions. https://ieeexplore.ieee.org/document/5167255

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Commun. ACM, 15*(12), 1053-1058.

7. Acknowledgements

This SDS is built slightly on a similar document produced by Stuart Faulk in 2017, and heavily on the publications cited within the document, such as IEEE Std 1016-2009.

We used the information from the Project Evaluation by Anthony Hornoff, in 2018 https://classes.cs.uoregon.edu/23W/cs422/Project_Evaluation.html to ensure that we completed all the necessary steps.

The UML diagrams in this SDS are based on Kieras and the Lecture Notes.

Concepts and basic knowledge from the *Software Engineering* textbook by Ian Sommerville: Sommerville (2015) *Software Engineering*, 10th Edition, Global Edition. ISBN 9781292096131.

This SDS is based on the template created by Anthony Hornof, in 2019.