Stockpile

Software Design Detail

CS 364 Section 01

Braden Brown
Catherine Flores
Dallen Cluff
Devin Vial
Dustin Christensen
Eli Andrew
Eric Powell
Frank Rincon Medina
Hayden Simmons
Heather Mason
Hyrum Gomez
Jacob Call
Jared Brown
Joe Canto
Joe Everett
Jonathan Carlson
Jonathan Sirrine
Justin Hurst
Katya Palchykova
Kevin Marsh
Mark Mayeda
Matthew Pickett
Nathan Tagg
Ryan Hafen
Spencer Browning
Steven Anderson
Sydney Palmer
Tim Lee
Vlad Kazandzhi
Zac Radford

Final Version

December 15, 2018

1.0 Version History

Date	Version	Comments
November 10, 2018	0	This version contained a list of entities and the traceability matrix
November 17, 2018	1	This version contains: 1) Formatting 2) Front matter 3) Viewpoint identification 4) Second attempt at identification and description of entities
December 1, 2018	2	Changes to this version include: 1) Large viewpoints were broken down into several smaller entities 2) Most sections marked TBD were completed 3) Changes to formatting were made 4) Class definitions were added to database viewpoint. 5) The editing and revising process was begun on any section that was complete.
December 8, 2018	3	Changes to this version include: 1) Attempted to implement Controllers 2) Moved Viewpoint Sources to end of document 3) Fixed Function signatures
December 15, 2018	4	1) Added CRUD Functions to Table Classes 2) Corrected Section Numbering 3) Corrected table of contents page numbering 4) Prepared Final Formatting

2.0 Table of Contents

Table of Contents

3.0 Introduction

Stockpile is an application that allows a user to track and manage their personal pantry. Stockpile manages inventory items, assists in stocking the pantry, keeps track of expiration dates, notifies the user when items are on sale, and suggests and saves recipes. It also allows the user to plan their weekly menu based on items available in their inventory.

3.1 Purpose

The purpose of Stockpile is to improve the user's ability to manage their personal pantry's inventory. The system will suggest recipes based on the user's inventory, then automatically remove items from their inventory when the recipe is used. The user can designate lower limits on certain items that will trigger the system to automatically add those items to their grocery list. Additionally, Stockpile will allow the user to schedule a reminder to update their inventory, and to track expiration dates.

3.2 Scope

Stockpile will include an interface, a cloud database, an Android Room database, user information, pantry item information, meal planning information, a login process, an interactive barcode scanning process using Google Vision API, and a process to interact with Recipe/Ingredient API.

Stockpile will assist the user by providing a way to add and remove items manually from their inventory and, when available, use the device's camera as a barcode scanner. The system will then suggest recipes that the user can make based on ingredients in their pantry. The system will then generate a grocery list of items that the user desires to continually have in stock, based on a lower limit designated by the user. Stockpile will also allow the user to share their pantry items and grocery lists with other users.

The user shall operate stockpile through the touch screen of their Android device. In addition to built in user interface elements such as buttons and input boxes, Stockpile shall also utilize the device's built in onscreen keyboard.

3.3 Design Stakeholders and Their Concerns

Stockpile's design stakeholders include the following: the project sponsor, the project team, the development team, and the subject matter experts involved during elicitation engagements. The stakeholders' concerns include:

- The interface should be easy to use, providing automated features wherever possible.
- The database should be simple and efficient.
- The application should be easily maintainable.
- The application should be easily modifiable in order to add future features.
- The application should be secure.
- The application should meet the performance requirements in section 3.4 of the SRS document including supporting up to 4 users from the household interacting with the inventory at one time and meeting meeting the availability requirements specified.

3.4 Glossary

3.4.1 Abbreviations

Acronym/ Abbreviation	Definition
API	Application Program Interface - Code that allows software programs to communicate with each other.
GUI	Graphical User Interface - Allows the user to interact with the device through graphical icons and visual indicators.
HTTP/2	HyperText Transfer Protocol – a protocol that describes how messages are formatted and transmitted, and what actions web servers and browsers should take in response to various commands. HTTP/2 is a major revision of the original HTTP that enables more efficient communication over the world wide web.
REF	Abbreviation for "References"
SQL	SQL (structured query language) describes a technology that is used to manage modern relational databases
UPC	Universal Product Code - A set of numbers commonly found alongside a barcode used by manufacturers to identify products.

3.4.2 Definitions

Term	Definition
Administrator	The user with authorization and all system privileges in a given app.
Alert	A visual and/or auditory notification sent to the user that provides necessary app information.
Analytics	Statistics created using data gathered about items a user has purchased in the past month, 2 months, 6 months, year, or all-time.
Android Operating System	The software designed to run on Android devices for basic support functions.
Barcode	A unique symbology used by manufacturers to identify a product.
Barcode Scanner	A built-in software tool that utilizes the device's camera for scanning product barcodes.

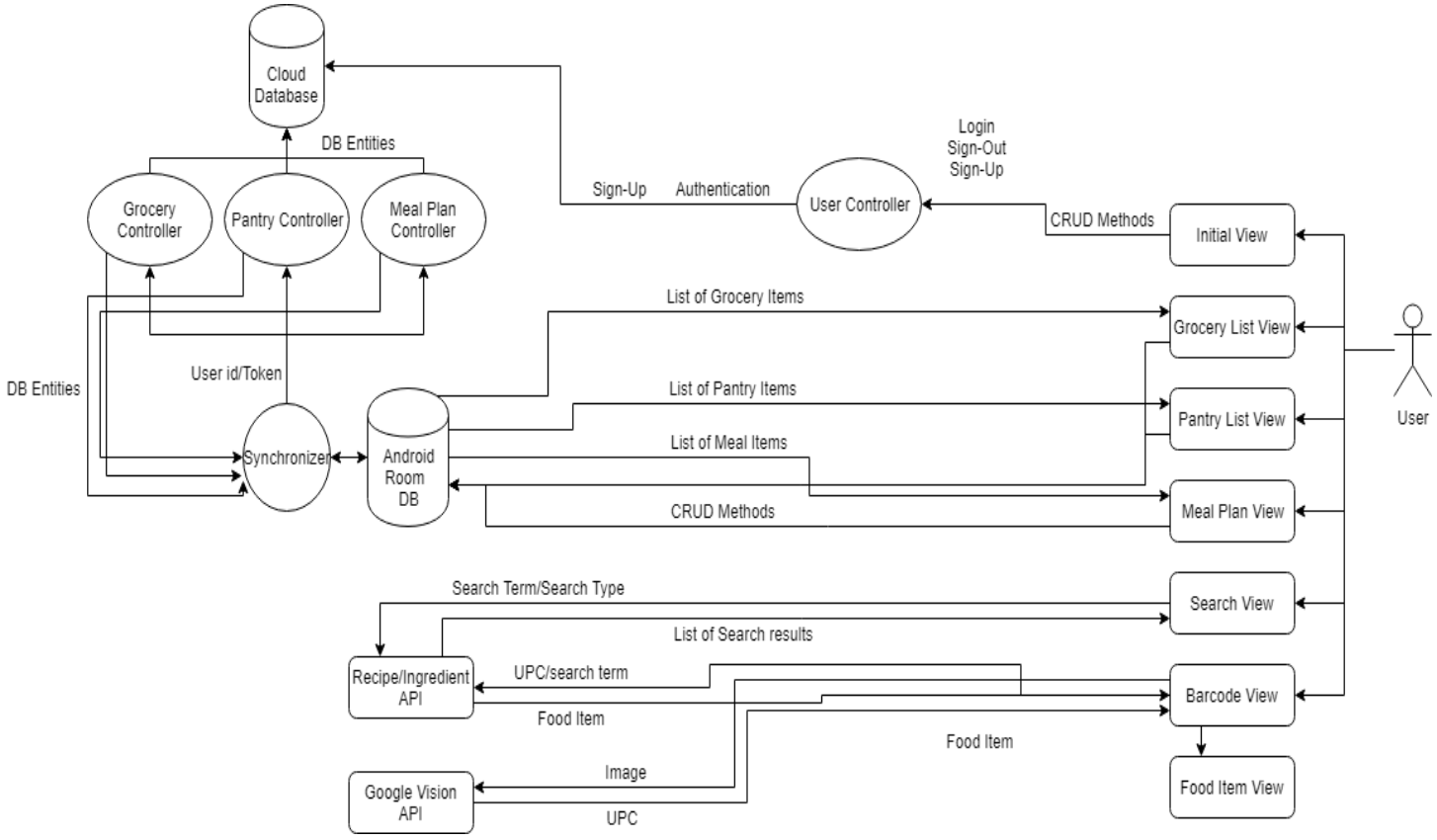
Term	Definition
Cellular Internet Connection	Internet availability on a device through a carrier company.
Checkbox	a small box on a form into which a check or other mark is entered to indicate selection
Cloud Storage	An external space for storing application data.
Control Panel	The application's main page that links to all other functions within the application.
Database	A structured set of data held in a computer, especially one that is accessible in various ways.
Display Output	The textual or visual information that the user sees on the screen of the device.
Encode/Encrypt	To convert information into another form, typically to a cipher for security purposes.
Encryption	Encoded data that is sent to and from the server for security purposes.
Expiration Date	A date indicating the last day that a product is safe to consume.
General User	A non-administrative user with limited privileges and authority.
Grocery List	1.) A list of items the user needs to purchase, which Stockpile will add to the inventory after the purchase is complete. 2.) The name of a page in the application.
Hash	Hashing refers to the encryption process of applying a transformation upon a string of characters (like a password) to change its value. A HASH is a transformed string that can only be decoded with the proper function.
Household	A collaborative user group in the application with up to eleven members; one administrator and ten general users.
Hyperlink	A link from a hypertext file to another location or file, typically activated by clicking on a highlighted word or image on the screen.
Meal Planning Feature	The page on which the user has the ability to determine what meal he or she wants to prepare on one or more specified days.
Navigate	Traversing through different app menus/screens to locate desired information.
Onboard Camera	The internal camera on the device.
Online	Controlled by or connected to another computer or to a network.
Pantry	A user’s physical supply of food items.
Pantry Inventory	The database on Stockpile which contains a list of items a user’s physical pantry contains

Term	Definition
Pantry Item	Food item in a user's virtual pantry inventory; describes the item name, an image of the item, and an expiration date.
Plain Text	Data that represent only characters of readable material that has not been specially formatted or written in code
Prototype	A partial or incomplete version of the system with some but not all app features met.
Query	An operation performed on a database used to retrieve information.
Recipe Box	A collection of recipes.
Recipe Suggestion	Recipes selected and displayed to the user based on similar previously used recipes and frequently used online recipes selected by other users.
Reminder	A scheduled notification intended to help the user remember the occurrence of an event.
Runtime Error	An error that can cause the application to stop running due to a flaw in the code or a conflict with multiple users.
Server	A computer program that provides a service to another computer program.
Sync	An attempt to match local data modifications stored on the app onto data stored on the server, or vice versa.
Tour	Tour is a navigating guide explaining a user purpose of Stockpile and how to orient in the application. The guide is represented as a two-minute video.
Update	A change that builds upon the current system to fix and/or modify the system in a beneficial way.
User Preferences	Customizable settings that allow the user to personalize the appearance of the application.
Wi-Fi	A trademarked term meaning IEEE 802.11x. A wireless connection to the internet.

4.0 Design Views

4.1 Design Overview

4.1.1 Data Flow Diagram



4.1.2 Design Elements

4.1.2.1 Android Room

Type	Entity
Purpose	Stores the local data for the Stockpile application
Description	Android Room is a SQLite wrapper. It contains application data in database tables.

4.1.2.2 Microsoft SQL Server

Type	Entity
Purpose	Stores the data for the Stockpile application
Description	Accepts data saved locally to be stored. It also returns data to the Stockpile application at startup.

4.1.2.3 Google Vision API

Type	External Entity
Purpose	Barcode Scanner
Description	Parse barcodes and returns json data

4.1.2.4 Recipes/Ingredient API

Type	External Entity
Purpose	Provides Recipes
Description	Accepts array of ingredients, based off the current Stockpile inventory, and returns recipes that match the ingredients.

4.1.3 Analytical Method

This separates the databases, the views, and the business layers, following the MVC design pattern.

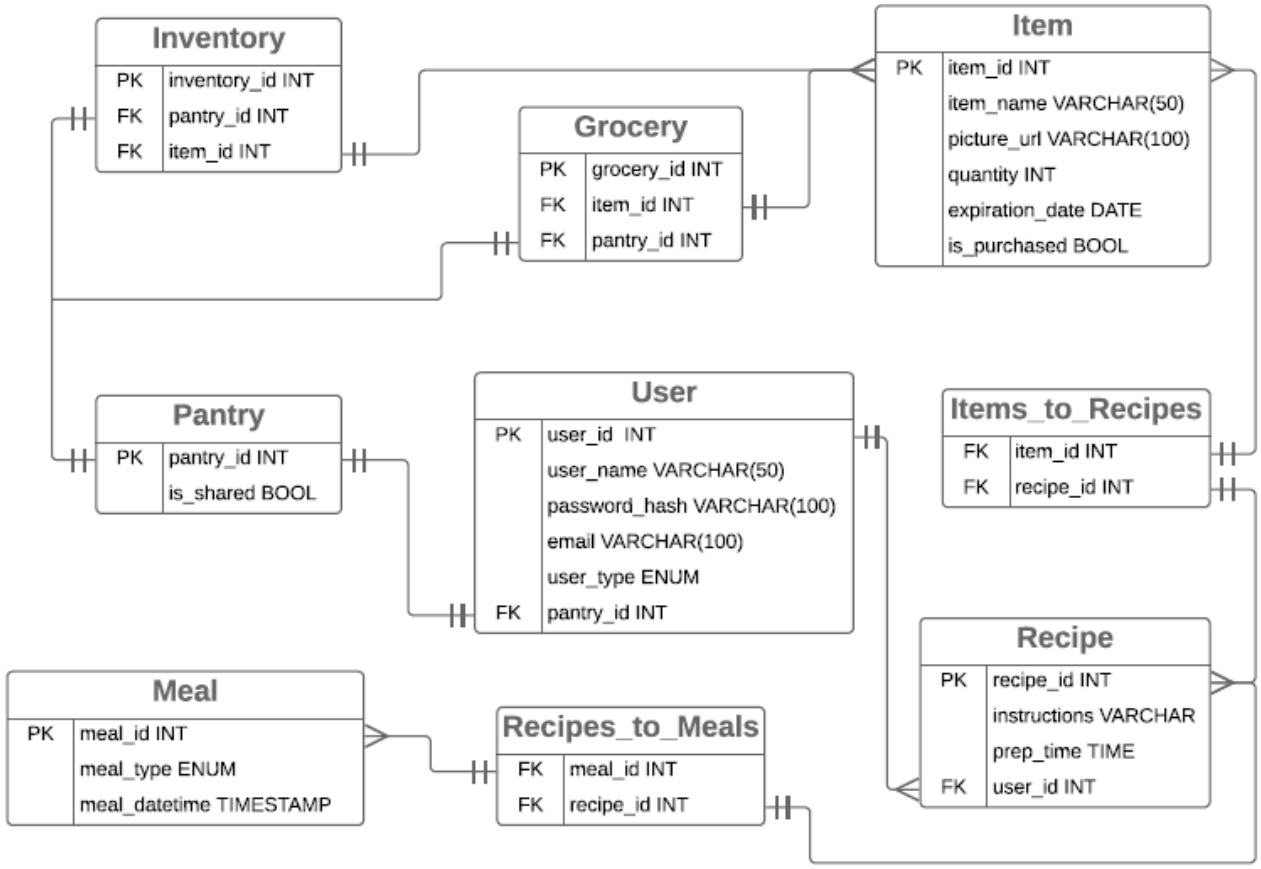
4.1.4 Design Concerns

SRS Section	Description

SRS 3.7.1	Stockpile shall interface with an image scanner API
SRS 3.7.2	Stockpile shall interface with the Microsoft SQL Server database to periodically synchronize all of the user's local database data to the cloud.
SRS 3.7.3	Stockpile shall interface with a recipe API

4.2 Database

4.2.1 Entity-Relationship Diagram



4.2.2 Design Elements

4.2.2.1 User Table

The User Table stores information relevant to the user. This includes the username, password_hash, and email. Each time a user completes the registration process, these three variables will be saved into the User Table. Each time a user completes the login process, the username and password_hash will be returned to authenticate user.

When a new user registers as an Administrator, a new pantry will be created and added to the Pantry Table (4.2.2.2). When a new user registers as a General User after being invited to access an Administrator's pantry, then the Administrator's pantry ID will be associated with the new user.

Variable	Description
user_id	A unique, table specific, numeric ID representing a specific user. This ID is automatically generated and incremented by the database.
user_name	A unique string that represents the user's name.
password_hash	A string that represents a hashed password.
email	A unique string that represents the user's email.
user_type	The user's account type. There are only two options: "Administrator" or "General".
pantry_id	A foreign key that is used to associate an instance of a pantry from the Pantry Table (4.2.2.2) with a user. One pantry can be associated with multiple users if the pantry instance has been shared by the administrator (the user who created the pantry instance).
household_id	A foreign key representing which household a particular user is associated with.

4.2.2.2 Pantry Table

The Pantry Table is very tightly coupled with the User Table (4.2.2.1) to represent users' pantries. For every new user, there must be a pantry associated with that user. When a new user is added to the User Table, a new pantry will be added to the Pantry Table if the new user chooses to be an Administrator. If the new user will not be an Administrator, but will instead be a General User, then a new pantry will not need to be added to this table for the new user.

Variable	Description
pantry_id	A unique, table specific, numeric ID representing a specific pantry. This ID is automatically generated and incremented by the database.
pantry_name	A unique string that represents the name of the pantry.
is_shared	"True" or "False". "True" means the pantry is being shared with other users, and "False" means the pantry is not being shared with other users.
household_id	A foreign key representing which household a pantry is associated with.

4.2.2.3 Grocery Table

The Grocery Table is used together with the Pantry Table (4.2.2.2) and the Item Table (4.2.2.7) to represent a grocery list. For an item to be stored in the Grocery Table, an item needs to have an "is_purchased" value of "False". This value indicates that the item is to be purchased but has not yet been purchased.

Variable	Description
grocery_id	A unique, table specific, numeric ID representing a specific grocery list. This ID is automatically generated and incremented by the database.
grocery_name	A unique string that represents the name of a grocery list.
item_id	A foreign key that is used to associate an instance of an item from the Item Table (4.2.2.7) with a grocery list.
pantry_id	A foreign key that associates a Pantry Table (4.2.2.2) instance with a grocery list.

4.2.2.4 Inventory Table

The Inventory Table stores information about the user's pantry. This includes a reference to Pantry Table (4.2.2.2) and Item Table (4.2.2.7).

Variable	Description
inventory_id	A unique, table specific, numeric ID representing a specific inventory list. This ID is automatically generated and incremented by the database.
inventory_name	A unique string that represents the name of the inventory.
pantry_id	A foreign key used to identify an instance of a pantry that is associated with the inventory list. This ID is retrieved from the Pantry Table.
item_id	A foreign key used to identify the item that is associated with the inventory list. This ID is retrieved from the Item Table.

4.2.2.5 Recipe Table

The Recipe Table will store recipes. A recipe will be saved to the Recipe Table when the user chooses to save a recipe previously retrieved from the Recipe/Ingredients API, or when a custom recipe is created by the user.

Variable	Description
recipe_id	A unique, table specific, numeric ID representing a specific recipe. This ID is automatically generated and incremented by the database.
recipe_name	A unique string that represents the name of the recipe.
instructions	A unique string that represents a set of instructions that the user will need to follow to prepare the food associated with the recipe.
prep_time	An integer that represents the estimated amount of time in minutes that it will take for the user to prepare the recipe.
user_id	A foreign key used to identify the user associated with a recipe. This ID is retrieved from the User Table (4.2.2.1).

4.2.2.6 Meal Plan Table

The Meal Table will store meal plans that users have created. Users will be able to specify multiple recipes to use for a meal plan (4.2.2.8).

Variable	Description
mealPlan_id	A unique, table specific, numeric ID representing a specific meal plan. This ID is automatically generated and incremented by the database.
mealPlan_name	A unique string that represents the name of the meal plan.
mealPlan_type	Users will select one of 4 set strings that represent a meal plan type. These types are “Breakfast”, “Lunch”, “Dinner”, or “Other
mealPlan_datetime	A unique string representing the date and time that the user has specified for the planned meal.
household_id	A foreign key representing which household a particular meal plan is associated with.

4.2.2.7 Item Table

The Item Table stores food items. An item can be added to a grocery list in the Grocery Table (4.2.2.3). An item can also be added to an inventory list in the Inventory Table (4.2.2.4). The boolean value of “is_purchased” indicates whether the item belongs in the Grocery Table or the Inventory Table. A value of “False” means “to be purchased” (Grocery Table), and a value of “True” means “purchased” (Inventory Table).

Variable	Description
item_id	A unique, table specific, numeric ID representing a specific food item. This ID is automatically generated and incremented by the database.
item_name	A unique string representing to the name of the item.
picture	A unique string containing the URL of a representative picture of the item.
quantity	An integer indicating the number of desired/purchased items.
expiration_date	The date that a food item is set to expire.
is_purchased	A boolean that represents whether an item has been purchased or not. “False” means not purchased, which means the item belongs to the Grocery Table. “True” means purchased, which means the item belongs to the Inventory Table.

4.2.2.8 RecipesToMeals Table

The RecipesToMeals Table is a join table to allow a user to assign multiple recipes to one meal, and to assign one recipe to multiple meals (many-to-many relationship).

Variable	Description
meal_id	A foreign key representing a specific meal. This ID is associated with an instance of a meal from the Meal Table (4.2.2.6).
recipe_id	A foreign key representing a specific recipe. This ID is associated with an instance of a recipe from the Recipe Table (4.2.2.5).

4.2.2.9 ItemsToRecipes Table

The ItemsToRecipes Table is a join table to allow multiple food items (ingredients) from the Item Table (4.2.2.7) to be associated with multiple recipes from the Recipe Table (4.2.2.5) (many-to-many).

Variable	Description
recipe_id	A foreign key representing a specific recipe. This ID is associated with an instance of a recipe from the Recipe Table (4.2.2.5).
item_id	A foreign key representing a specific food item (ingredient). This ID is associated with an instance of a food item from the Item Table (4.2.2.7).

4.2.2.10Household Table

The Household table holds each unique household. The household will be tied to the user table, pantry table, and Meal Plan table.

Variable	Description
household_id	A unique, table specific, numeric ID representing a specific household. This ID is automatically generated and incremented by the database.
pantry_Id	A foreign key representing the Pantry ID that is associated with a Household (4.2.2.2).
administrator_Id	A foreign key of a User ID that represents the Administrator of a Household (4.2.2.1).
user_Ids	A list of foreign keys of userIds representing users assigned to a particular Household. The maximum amount of users assigned to a Household, including the Administrator, is 11 (4.2.2.1).

4.2.3 Analytical Method

Our system will require a large amount of data storage. The information stored is complex, and is best implemented using a database. Our database will adhere to the proper normalization techniques, specifically the 3NF (Third Normal Form), to keep the table free of update, insertion and deletion anomalies. For more information about normalization see this article.

Encryption of all data being sent to and from the data will be handled by the APIs and Microsoft SQL Server database services.

4.2.4 Design Concerns

SRS Section	Description
SRS 3.5.1	Positive integer values will be used to display all quantity information.
SRS 3.5.1.1	For all lists, Stockpile will display a button that allows the user to select a grid view of their items or row view of their items.
SRS 3.5.1.2	For every list, Stockpile shall display information about each item.
SRS 3.5.1.2.1	Stockpile shall display the name of each item.
SRS 3.5.1.2.2	Stockpile shall display the quantity of each item.
SRS 3.5.1.3	Stockpile shall display a picture of each item.
SRS 3.5.2	Each item image shall be clickable to allow the user to manually change or select an image file for each item.
SRS 3.5.2.1	The system shall display a list of all the items in the pantry inventory.
SRS 3.1.8.1.2	In addition to the default item information, the system shall display an expiration date for each item.
SRS 3.1.8.2	When the user selects an item, Stockpile will display a submenu to edit each item.

4.3 User Class

4.3.1 UML Class Diagram

User
- userID : int - username : String - passwordHash : String - email : String - userType : enum
+ createUser(username : String, passwordHash : String, email : String, userType : enum) + verifyEmail(email : String) : boolean + verifyPassword(password : String) : boolean

+ updateUser(username : String, passwordHash : String, email : String, userType : enum) + deleteUser(userID): boolean

4.3.2 Design Elements

4.3.2.1 userID

Type	Entity
Purpose	Represent User ID
Description	This will be the ID from the database associated with a User.

4.3.2.2 username

Type	Entity
Purpose	Represent User's chosen name.
Description	This string will be the name that the User has chosen to be referred to as throughout the Stockpile app.

4.3.2.3 passwordHash

Type	Entity
Purpose	Represent hashed version of the User's password.
Description	This string will contain the hashed version of the User's password to be used when verification is needed.

4.3.2.4 email

Type	Entity
Purpose	Represent User's email.
Description	This string will contain the User's email address.

4.3.2.5 userType

Type	Entity
Purpose	Represent the type of user.
Description	The only two possible types of users are Administrator and General User. This determines what kind of reading and writing privileges the user has. See 4.15 for more details.

4.3.2.6 createUser(username : String, passwordHash : String, email : String, userType : enum)

Type	Function
Purpose	Initialize the user object.
Description	When initialized, the user must have a username, passwordHash, email, and userType. verifyEmail() and verifyPassword() are called to ensure the email and password are well formed, respectively. Internally, the database is then referenced to pull and initialize the User ID.

4.3.2.7 verifyEmail(email : String) : boolean

Type	Function
Purpose	Verifies that the user's email address is valid.
Description	This function will verify that the submitted email address is a valid form of an email. Returns true if valid, false otherwise.

4.3.2.8 verifyPassword(password : String) : boolean

Type	Function
Purpose	Verifies that the password meets the password standards.
Description	Check the character length, standards for uppercase/lowercase, numbers, and character to ensure password is valid. Returns true if password meets requirements, otherwise false is returned.

4.3.2.7 updateUser(username : String, passwordHash : String, email : String, userType : enum)

Type	Function
Purpose	Update User
Description	When the user updates their information, the function will take the new information and update it in the user table.

4.3.2.7 deleteUser(userID : int)

Type	Function
Purpose	Deletes user
Description	Deletes the user from the user database.

4.3.3 Analytical Method

The User class was carefully designed to adhere to the requirements of the SRS. This class represents a user and is used in conjunction with 4.13 (AccountLogin) to allow the user to log in to the application, so it must have proper cohesion and coupling.

4.3.4 Design Concerns

SRS Section	Description
SRS 3.1.4.1.1	Stockpile shall display input fields for the user to enter a valid email and password.
SRS 3.2.5.1.3	Passwords shall be unique from any passwords previously used by an individual user.
SRS 3.2.5.1.4	Stockpile shall verify a user's identity with an email address and a password against previously saved credentials.

4.4 Household Class

4.4.1

Household
- householdId : int - pantryId : int - administratorId : int - userIds : ArrayList<int>
+ createHousehold() : void + addUser(userId : int) : boolean + addAdministrator(userId : int) : boolean + getHouseholdUsers() : ArrayList<int> + reassignAdministrator(userId : int) boolean + removeUser(userId : int) : boolean + isAdministrator(userId : int) boolean + deleteUser(userId : int) : boolean

4.4.2 Design Elements

4.4.2.1 householdId

Type	Entity
Purpose	Represents Household ID.

Description	This will be the ID from the database associated with a Household.

4.4.2.2 pantryId

Type	Entity
Purpose	Represents Pantry ID.
Description	This will be the ID from the database associated with the Pantry that is associated with the Household.

4.4.2.3 administratorId

Type	Entity
Purpose	Represents Administrator ID.
Description	This will be the ID from the database associated with a user that is the one and only administrator of the Household.

4.4.2.4 userIds

Type	Entity
Purpose	All users associated with a householdID
Description	The list of userIds assigned to a particular Household. The maximum amount of users assigned to a Household, including the Administrator, is 11.

4.4.2.5 createHousehold

Type	Function
Purpose	Creates an instance of a Household.
Description	During the setup of Stockpile, the user will have the option to create a new Household. Doing this will initialize a new Household with the current user's ID as the default Administrator. A new Household table will also be created in the database with the current user's ID as the Administrator.

4.4.2.6 addUser(userId : int)

Type	Function
Purpose	Adds a user to a Household.

Description	As long as the number of Household members does not exceed 11, add the userId from the parameter to the ArrayList of users. The userId is also added to the Household table in the database. Returns true if size of userIds is not above 11 and if user was successfully added to the database, returns false otherwise.

4.4.2.7 addAdministrator(userId : int)

Type	Function
Purpose	Adds the Administrator of a Household.
Description	Accepts a userId as a parameter. As long as the Administrator has not been assigned, the userId is assigned the Administrator of the Household. The new Administrator ID is then added to the database. Returns true if new Administrator was successfully added, false otherwise.

4.4.2.8 getHouseholdUsers()

Type	Function
Purpose	Retrieves specific Household Users account details.
Description	Takes the userId from a list of Household Users and retrieves all data related to that userId. This includes settings, items added to inventory, grocery list, recipe, etc.

4.4.2.9 reassignAdministrator(userId : int)

Type	Function
Purpose	Updates the Administrator to a Household.
Description	Takes the userId from the parameter and reassigns the Administrator ID. The Administrator ID is then updated in the database. Returns true if database update was successful, false otherwise.

4.4.2.10 removeUser(userId : int) boolean

Type	Function
Purpose	Removes Household User from the Household account.
Description	Takes a userId and deletes all setting, likes, dislikes associated with the Household userId. Optional deletion shall include recipe list, grocery list, inventory item contributed by specified userId.

4.4.2.11 isAdministrator(userId : int) : booean

Type	Function
Purpose	Checks if user is Administrator of Household.
Description	Compares the userId specified in parameter to the Administrator ID. Returns true if ID comparison matched, false otherwise.

4.4.2.12 deletesUser(userId : int) : boolean

Type	Function
Purpose	Deletes user from database.
Description	Deletes the user based off of user id from the user database and removes all associations to other tables.

4.4.3 Analytical Method

During the initial setup process of Stockpile, the user will need to either create or join a Household. The user can create a Household when the CreateHousehold function is called. By default, this will assign the current user as the Administrator of the Household. Additionally, upon Household creation, the Pantry that is associated with the current user will be assigned to the Household. The user can join another Household when the AddUser is function is called. This will associate the current user with another Household and the Pantry that is associated with that Household.

4.4.4 Design Concerns

SRS Section	Description
SRS 3.1.5.2	The system shall display an option for the user to create a new household.
SRS 3.1.5.2.1	If the user creates a new household, the user will become the administrator for the household.
SRS 3.1.5.2.2	Once the user creates a new household, the system shall direct the user to the control panel.
SRS 3.1.5.3	The system shall display the option for the user to choose to view households they are already part of.
	If the user chooses to view a household that they are already part of, the system shall display those households.

	Once the user selects a household from the displayed list, the system shall direct the user to the control panel.

4.5 Inventory Class

4.5.1 UML Class Diagram

Inventory
- inventoryId : int - pantryId : int - inventoryList : ArrayList<Item>
+ createInventory(): boolean + getItems() : ArrayList<Item> + verifyItem(item : Item) : boolean + addItem(item : Item) : boolean + removeItem(item : Item) : boolean + updateItem(item : Item) : boolean + sortAlphabeticalAsc() : ArrayList <Item> + sortAlphabeticalDesc() : ArrayList<Item>

+ sortPriceAsc() : ArrayList<Item> + sortPriceDesc() : ArrayList<Item> + emptyInventory() : boolean + deleteInventory(inventoryID : int) : boolean

4.5.2 Design Elements

4.5.2.1inventoryId : int

Type	Entity
Purpose	Represents Inventory ID.
Description	This will be the ID from the database associated with an inventory.

4.5.2.2 Inventory

Type	Entity
Purpose	Keeps track of all the Items within the user's pantry.
Description	An ArrayList of all of the Items in a user's pantry.

4.5.2.3 createInventory() : boolean

Type	Function
Purpose	Create a new inventory
Description	A new inventory will be created when a new household is created.

4.5.2.4 getItems()

Type	Function
Purpose	Retrieves all inventory items from database
Description	This function performs a query on the database to retrieve all the items from the database. This query will include an order by clause to specify the initial order of the list, which will be ascending alphabetical order based on the name of the pantry Item.

4.5.2.5 verifyItem(item : Item)

Type	Function
Purpose	Verifies that an Item object is valid.
Description	This function checks each property value of an Item object to ensure that it is valid before any additions or updates are made to the Item table in the database. In order for the item to be considered valid, all of the following must be met: ● description must be a string less than 250 characters in length, ● the name must be a string less than 50 characters in length, ● the price must be a double, ● the quantity must be an integer, ● isPurchased must have a value of true or false.

4.5.2.6 addItem(item : Item)

Type	Function
Purpose	Adds an Item to the local ArrayList and to the Item table in the database.
Description	Accepts an instance of an Item. After calling verifyItem(item : Item), the Item will be added to the ArrayList of Items and to the Item table in the database. Returns true if Item was successfully added to the Item table, false otherwise. Upon the completion of this function, the corresponding sort() function (below, determined by the current sort-state of the list) will be called to reformat the list.

4.5.2.7 removeItem(item : Item)

Type	Function
Purpose	Removes an Item from the local ArrayList and removes the same Item from the item table in the database.
Description	Accepts an instance of an Item that will be removed. After calling verifyItem(item : Item), the Item will be removed from the ArrayList of Items and the Item table in the database. Returns true if deletion was successful, false otherwise.

4.5.2.8 updateItem(item : Item)

Type	Function
Purpose	Updates an Item in the local ArrayList and updates the Item in the database.
Description	Accepts an instance of an Item that will be updated. After calling verifyItem(item : Item), the item will be updated in the ArrayList of Items and in the Item table in the database. Returns true if update was successful, false otherwise. Upon the completion of this function, the corresponding sort() function (below, determined by the current sort - state of the list) will be called to re-format the list.

4.5.2.9 sortAlphabeticalAsc()

Type	Function
Purpose	Sorts all of the Items in a user's Inventory List alphabetically.
Description	This function sorts the Items in a user's Inventory List alphabetically from 'A' to 'Z', using a custom comparator. It returns the sorted ArrayList of Items. This function uses a quick sort logic algorithm to sort the inventory items. This function will be called when: ● The user adds a new item to the inventory with addItem(), and the list was previously sorted alphabetically. ● The user changes an existing inventory item with updateItem(), and the list was previously sorted alphabetically. ● If the user selects the option to sort the inventory list A - Z. Note: This function will NOT be called after removeItem() as this function does not potentially jeopardize the proper order of the list.

4.5.2.10 sortAlphabeticalDesc()

Type	Function
Purpose	Sorts all of the Items in a user's Inventory List in reverse-alphabetical order.
Description	This function sorts the Items in a user's Inventory List alphabetically from 'Z' to 'A'. It returns the sorted ArrayList of Items. This function uses a quick sort logic algorithm to sort the inventory items. This function will be called when: ● The user adds a new item to the inventory with addItem(), and the list was previously sorted reverse - alphabetically. ● The user changes an existing inventory item with updateItem(), and the list was previously sorted reverse - alphabetically. ● If the user selects the option to sort the inventory list Z - A. Note: This function will NOT be called after removeItem() as this function does not potentially jeopardize the proper order of the list.

4.5.2.11 sortPriceAsc()

Type	Function
Purpose	Sorts all of the Items in a user's Inventory List by price from lowest to highest.
Description	This function sorts the Items in a user's Inventory List by price from lowest to highest. It returns the sorted ArrayList of Items. This function uses a quick sort logic algorithm to sort the inventory items. This function will be called when: ● The user adds a new item to the inventory with addItem(), and the list was previously sorted price: lowest to highest. ● The user changes an existing inventory item with updateItem(), and the list was previously sorted price: lowest to highest. ● If the user selects the option to sort the inventory list price: lowest to highest. Note: This function will NOT be called after removeItem() as this function does not potentially jeopardize the proper order of the list.

4.5.2.12 sortPriceDesc()

Type	Function
Purpose	Sorts all of the Items in a user's Inventory List by price from highest to lowest. This function uses a quick sort logic algorithm to sort the inventory items. This function will be called when: ● The user adds a new item to the inventory with addItem(), and the list was previously sorted price: highest to lowest. ● The user changes an existing inventory item with updateItem(), and the list was previously sorted price: highest to lowest. ● If the user selects the option to sort the inventory list price: highest to lowest. Note: This function will NOT be called after removeItem() as this function does not potentially jeopardize the proper order of the list.
Description	This function sorts the Items in a user's Inventory List by price from highest to lowest. It returns the sorted ArrayList of Items.

4.5.2.13 emptyInventory()

Type	Function
Purpose	Removes all of the Items from the user's Inventory List.
Description	As long as the Inventory List is not empty, the Items will all be deleted from the Inventory List as well as the Item table in the database. Returns true if deletion was successful, false otherwise.

4.5.2.14 deleteInventory(inventoryId : int)

Type	Function
Purpose	Deletes the inventory.
Description	Deletes the inventory based off of the inventory id.

4.5.3 Analytical Method

The Inventory class contains everything a user needs to view, edit and update their personal pantries. As such, it provides functionality to sort their Inventory by price and alphabetically. The user will also be able to completely empty their Inventory, or update, add, or remove a particular Item in their Inventory.

4.5.4 Design Concerns

SRS Section	Description
SRS 3.1.1.4	Positive integer values will be used to display all quantity information.
SRS 3.1.1.5	For all lists, Stockpile will display a button that allows the user to select a grid view of their items or row view of their items.
SRS 3.1.1.6	For every list, Stockpile shall display information about each item.
SRS 3.1.1.6.1	Stockpile shall display the name of each item.
SRS 3.1.1.6.2	Stockpile shall display the quantity of each item.
SRS 3.1.1.6.3	Stockpile shall display a picture of each item.
SRS 3.1.1.6.3.1	Each item image shall be clickable to allow the user to manually change or select an image file for each item.
SRS 3.1.8.1	The system shall display a list of all the items in the pantry inventory.
SRS 3.1.8.1.2	In addition to the default item information, the system shall display an expiration date for each item.
SRS 3.1.8.2	When the user selects an item, Stockpile will display a submenu to edit each item.

4.6 Pantry Class

4.6.1 UML Class Diagram

Pantry
- pantryId : int - inventoryList : Inventory - groceryList : Grocery - reminders: ArrayList<Reminder>
+ addItemToList(item : Item) : boolean + getItem(item : Item) : boolean + removeItemFromList(item : Item) : boolean + removeAllItemsFromInventory(listName : String) : boolean + updateItemInList(item : Item) : boolean + addReminder(reminder: Reminder) : void + getReminders() : ArrayList<Reminder> + removeReminder(reminder : Reminder) : void + createReminder() : Reminder + createPantry() + deletePantry(pantryID : int)

4.6.2 Design Elements

4.6.2.1 pantryId : int

Type	Entity
Purpose	Represents Pantry ID.
Description	This will be the ID from the database associated with Pantry.

4.6.2.2 addItemToList(item : Item)

Type	Function
Purpose	Adds an Item to either the Inventory or Grocery list.
Description	Accepts an Item as a parameter. If the Item has an isPurchased value of true, the Item is added to the Inventory list. Otherwise, the Item is added to the Grocery list. Returns true if the Item was successfully added, false otherwise.

4.6.2.3 getItem(item : Item)

Type	Function

Purpose	Retrieves an Item to either the Inventory or Grocery list.
Description	Accepts an Item as a parameter. If the Item has an isPurchased value of true, the Item is returned from the Inventory list.

4.6.2.4 removeItemFromList(item : Item)

Type	Function
Purpose	Removes an Item from either the Inventory or Grocery list.
Description	Accepts an Item as a parameter. If the Item has an isPurchased value of true, the Item is removed from the Inventory list. Otherwise, the Item is removed from the Grocery list. Returns true if the Item was successfully removed, false otherwise.

4.6.2.5 removeAllItemsFromList(listName : String)

Type	Function
Purpose	Removes all Items from either the Inventory or the Grocery list.
Description	If the argument passed for listName is “inventory”, this function will remove all of the Items from the user’s Inventory list. If the argument passed for listName is “grocery”, this function will remove all of the Items from the user’s Grocery list. The function will attempt to remove all of the Items from the specified list. Once the attempt has been made, the function will return a boolean value of true upon successful removal, or false upon failure.

4.6.2.6 updateItemInList(item : Item)

Type	Function
Purpose	Updates an Item in either the Inventory or Grocery list.
Description	Accepts an Item as a parameter. If the Item has an isPurchased value of true, the Item is updated in the user’s Inventory list. Otherwise, the Item is updated in the user’s Grocery list. Returns true if the Item was successfully updated, false otherwise.

4.6.2.7 addReminder(reminder : Reminder)

Type	Function
Purpose	Add a reminder that the User specified to the ArrayList of reminders.

Description	Given the Reminder that is passed in, it will add that Reminder to the reminders array. If there was a failure in this process, this method will return false, otherwise, it will return true.

4.6.2.8 getReminders()

Type	Function
Purpose	Return the list of reminders
Description	This method will return an ArrayList of Reminders to the caller.

4.6.2.9 removeReminder(reminder : Reminder)

Type	Function
Purpose	Remove a reminder
Description	This method will remove the reminder that is passed in from the reminders list.

4.6.2.10 createReminder()

Type	Function
Purpose	Create a reminder
Description	This method will allow the user to input the information needed to create a Reminder.

4.6.2.11 createPantry ()

Type	Function
Purpose	Create a pantry
Description	Creates a new pantry in the pantry table

4.6.3 Analytical Method

The Pantry class contains all the functionality a user needs to keep their Pantry (also referred to above as Inventory) up to date. It is able to get the Inventory and Grocery lists in order to show the user what they have and what they want, it is also able to add and remove items from their Inventory. The user also has the capability to completely empty out their Pantry. The PantryAppController will be able to create, add, get, and remove Reminders.

4.6.4 Design Concerns

SRS Section	Description
SRS 1.3.1.7	Stockpile shall be capable of an array of operations including updating the pantry database, setting reminders, sending notifications, and saving recipes and grocery lists.
SRS 3.1.6.3	When the user selects the “Pantry” button, Stockpile will display the user’s pantry inventory.
SRS 3.1.6.4	When the user selects the “Keep in Stock” button, Stockpile will display a list of all the desired items that the user wishes to keep in stock.
SRS 3.1.6.5	When the user selects the “Grocery List” button, Stockpile will display all the items added to the grocery list.
SRS 3.1.8.1	Stockpile shall display a list of all the items in the pantry inventory.
SRS 3.1.8.2	When the user selects an item, Stockpile will display a submenu to edit each item.
SRS 3.1.8.2.1	When the user selects an item, Stockpile will display a submenu to edit each item.
SRS 3.1.8.2.2	The submenu shall display an option for the user to change quantities of each item.
SRS 3.1.9.1	Stockpile shall display a list of items that the user wants to purchase during their next trip to the grocery store.
SRS 3.1.9.2.1	The submenu shall display the following options: “Delete item” and “Change Quantity”.
SRS 3.1.9.2.2	When the user selects “Delete item,” Stockpile will display a message to the user verifying that the user actually wants to delete the item.
SRS 3.1.9.2.3	When the user selects “Change Quantity,” Stockpile will display an input field allowing the user to enter in the new desired quantity.
SRS 3.2.2.1	The pantry inventory shall interface with the grocery list to add items to the inventory list.

4.7 Item Class

4.7.1 UML Class Diagram

Item
- itemID : int - itemName : String - itemDescription : String - itemPrice : double - quantity : int - isPurchased : boolean - expirationDate : Date
+ Item(itemName : String, itemDescription : String, itemPrice: double, quantity : int, isPurchased : boolean) + decrementQuantity() : void + incrementQuantity() : void + updateItem(itemName : String, itemDescription : String, itemPrice : double, quantity : int, isPurchased : boolean) : boolean + deleteItem (itemID : int) : boolean

4.7.2 Design Elements

4.7.2.1 itemID : int

Type	Entity
Purpose	Represents Item ID.
Description	This will be the ID from the database associated with item.

4.7.2.2 itemDescription

Type	Entity
Purpose	A brief description of the pantry Item.
Description	The description will be displayed in the detailed view of the pantry item. This will be a string variable which will either be auto-filled by information received from an online database or filled by the user of the application. The description variable can be set using the “setDescription(description : string)” function and accessed using the “getDescription()” function of the item class.

4.7.2.3 itemName

Type	Entity
Purpose	The name of the pantry Item.
Description	This string variable will be defined either from information from an online database or manually entered by the user. The name of the pantry item will be displayed in both list and detailed views and can be set using the “setName(name : String)” function and accessed using the “getName()” function of the item class.

4.7.2.4 itemPrice

Type	Entity
Purpose	Represents the retail price of an Item.
Description	This double variable will either be auto-filled from information from an online database or manually filled by the user. This double variable will be displayed in the detailed view of the pantry item and can be set using the “setPrice(price : double)” function, and called using the “getPrice()” function.

4.7.2.5 Quantity

Type	Entity
Purpose	Represents the number of the selected pantry Items in the inventory.
Description	This integer variable will be displayed in both list and detailed view on the application. This number can be set using the “setQuantity(quantity : int)” function, and accesses using the “getQuantity()” function.

4.7.2.6 Item(name : String, description : String, price: double, quantity : int, isPurchased : boolean)

Type	Function
Purpose	Item class constructor.
Description	This is the non-default constructor of the item class. When called, this will create an item object. This function accepts a name, description, quantity, and price as its parameters, then sets the class variables to equal those parameters. This constructor is the primary way to create a pantry item.

4.7.2.7 decrementQuantity()

Type	Function
Purpose	Decrements quantity.
Description	This function accepts and returns nothing. It is called when the user taps a “+” button in the items description to decrease the number of that item by 1.

4.7.2.8 incrementQuantity()

Type	Function
Purpose	Increments quantity.
Description	This function accepts and returns nothing. It is called when the user taps a “+” button in the items description to increase the number of that item by 1.

4.7.2.9 expirationDate

Type	Entity
Purpose	Represents the date that the item will be expired.
Description	This variable will used by PantryAppController when it searches for items that are expired to notify the user.

4.7.2.10 updateItem(itemName : String, itemDescription : String, itemPrice : double, quantity : int, isPurchased : boolean) : boolean

Type	Function
Purpose	Updates item to be added to the user’s inventory
Description	This function accepts input for the item from the user and then updates the item item information in the item table. Function returns true if the update was successful. Returns true if successful.

4.7.2.11 deleteItem(itemID : String) : boolean

Type	Function
Purpose	Deletes an item from the item table
Description	This function deletes an item from the item table and removes the association between the item and the inventory table. Returns true if successful.

4.7.3 Analytical Method

The Item class is responsible for creating one of the most basic object items of the Stockpile application. Each item object will represent an actual item stored in the user's pantry. The Item object is designed to contain all pertinent information needed in the Stockpile application including:

1. The name of the Item
2. A brief description of the Item
3. The manufacturer's suggested retail price of the Item
4. The quantity of the Item
5. The expiration date of the Item

The quantity will be represented as an integer whether the item be measured in containers, liters (if the item is a liquid), or kilograms (if the item cannot be measured using containers).

An item will be created using its constructor function, which will prompt the user for the above information and set the item's class variables accordingly. Item objects will be stored in a Java List to be used with other classes such as the Pantry List and the Grocery List. Some of this information may be extracted from and automatically filled using information from online databases. In either case, the user will be able to manually adjust each of the variables once an Item object is created.

4.7.4 Design Concerns

SRS Section	Description
SRS 3.1.8.1.2	In addition to the default item information, the system shall display an expiration date for each item.
SRS 3.2.1.1	Stockpile shall allow the user to manually adjust the quantity of each list item.
SRS 3.2.1.1.1	Stockpile shall allow the user to add new items to the list.
SRS 3.2.1.1.2	Stockpile shall allow the user to remove items from the list.

4.8 Recipe Class

4.8.1 UML Class Diagram

Recipe
- recipeId : int - name : String - description : String - ingredientList : ArrayList<Item> - instructions : String
+ Recipe(name : String, description : String, ingredientList : ArrayList<Item>, instructions : String) + addRecipe(recipe : Recipe) : void + getRecipes() : ArrayList<Recipe> + updateRecipes(recipes : ArrayList<Recipe>) : boolean + deleteRecipes(recipes : ArrayList<Recipe>) : boolean

4.8.2 Design Elements

4.8.2.1 recipeID : int

Type	Entity
Purpose	Represents Recipe ID.
Description	This will be the ID from the database associated with Recipe.

4.8.2.2 Name

Type	Entity
Purpose	The name of the recipe.
Description	This string variable will be defined from information from an online database. The name of the recipe will be displayed in both list and detailed views and can be accessed using the “getName()” function of the Item class.

4.8.2.3 Description

Type	Entity
Purpose	A brief description of the recipe.
Description	The description will be displayed in the detailed view of the

	recipe. This will be a string variable which will be auto-filled by information received from an online database. The description variable can be set using the “setDescription(description : String)” function and accessed using the “getDescription()” function of the Recipe class.

4.8.2.4 Instructions

Type	Entity
Purpose	Instructions for the recipe.
Description	The instructions will be displayed in the detailed view of the recipe. This will be a string variable which will be auto-filled by information received from an online database. The description variable can be set using the “setInstructions(instructions : String)” function and accessed using the “getInstructions()” function of the Item class.

4.8.2.5 addRecipe(recipe : Recipe)

Type	Function
Purpose	Adds an Item to the Recipe Table.
Description	Accepts a Recipe as a parameter. It then inserts the Recipe into the Recipe table.

4.8.2.6 getRecipes()

Type	Function
Purpose	Retrieves the list of recipes in that are currently in the Recipes Table.
Description	Queries the Database for all of the items in the Recipe Table and returns them.

4.8.2.7 updateRecipes(recipes : ArrayList<Recipe>) : boolean

Type	Function
Purpose	Updates recipes in the database.
Description	Takes in a list of Recipes and updates the recipes in the database. Returns true if the update was successful.

4.8.2.8 deleteRecipes(recipes : ArrayList<Recipe>) : boolean

Type	Function

Purpose	Deletes recipes from the Recipe Table.
Description	Takes in a list of recipes and deletes them from the database. Returns true if the delete was successful

4.8.3 Analytical Method

The Recipe class is responsible for storing recipes for the user. Each recipe object will contain actual items stored in the user's pantry. The recipe object is designed to contain all pertinent information for a recipe including:

1. The name of the recipe
2. A brief description of the recipe
3. The instructions for the recipe

4.8.4 Design Concerns

SRS Section	Description
SRS 3.2.4.1	The user will be able to create new recipes by tapping a "+" button. This button will call a new activity with 'editText' boxes where the user can fill in the parameters for the recipe. This activity will have a button called "Create Recipe" which will call the recipe class's constructor and fill the parameters with the user inputs. Once created, the recipe object will be added to the applicable list.
SRS 3.7.3.1	The recipe API will take in the name of a recipe (i.e. "Enchiladas") and return the resulting recipes that are found.
SRS 3.5.1.3	The cloud database should provide at least 300 default recipes for use by the household.

4.9 MealPlan Class

4.9.1 UML Class Diagram

MealPlan
- mealPlanId : int - recipeList : ArrayList<Recipe>
+ MealPlan(recipeList : ArrayList<Recipe>) + addItemToGroceryList(item : Item) : void + getItemsFromGroceryList() : ArrayList<Item> + updateItemsInGroceryList(items : ArrayList<Item>) : void + deleteItemsFromGroceryList(items : ArrayList<Item>) : void +updateMealPlan(recipeList : ArrayList<Recipe>): boolean +deleteMealPlan (mealPlanId : int) : boolean

4.9.2 Design Elements

4.9.2.1 mealPlanId : int

Type	Entity
Purpose	Represents Pantry ID.
Description	This will be the ID from the database associated with Pantry.

4.9.2.2 addItemToGroceryList(item : Item)

Type	Function
Purpose	Adds an Item to the Grocery list.
Description	Accepts an Item as a parameter. If the Item has an isPurchased value of false, the Item is added to the Grocery list. Returns true if the Item was successfully added and false otherwise.

4.9.2.3 getItemsFromGroceryList()

Type	Function
Purpose	Retrieves the list of items in that are currently in the Grocery List.
Description	Queries the Database for all of the items in the grocery list and returns them.

4.9.2.4 updateItemsInGroceryList(items : ArrayList<Item>)

Type	Function
Purpose	Updates items in the database.
Description	Takes in an ArrayList of Items as a parameter, and updates the Items' quantities in the database.

4.9.2.5 deleteItemsFromGroceryList(items : ArrayList<Item>)

Type	Function
Purpose	Deletes items from the Grocery List.
Description	Takes an ArrayList of Items the user wants deleted as a parameter and deletes the Items from the Grocery List.

4.9.2.6 mealPlanId

Type	Entity
Purpose	Unique identifier for each Meal Plan.
Description	An int value acting as a unique identifier for each separate Meal Plan.

4.9.2.7 recipeList

Type	Entity
Purpose	Holds a list of recipes.
Description	Holds a list of recipes that are currently available to the user.

4.9.2.8 updateMealPlan(recipeList : ArrayList<Recipe>): boolean

Type	Function
Purpose	Updates the list of recipes in the meal plan
Description	Takes an ArrayList of Recipes the user wants added to the meal plan and updates the meal plan. Returns true if the update was successful.

4.9.2.9 deleteMealPlan (mealPlanId : int) : boolean

Type	Function
Purpose	Deletes the meal plan

Description	Deletes the meal plan based off the mealPlanId. Returns true if the delete was successful.

4.9.3 Analytical Method

This class is responsible for holding a list of recipes the user can select from to create a weekly/monthly meal plan. If a selected recipe contains an item/items that are currently unavailable, the addItemToGroceryList function will add those items to the user's grocery list. Additionally, if the user removes a recipe from their meal plan, any items that were added to the grocery list from the associated recipe will be removed.

4.9.4 Design Concerns

SRS Section	Description
SRS 3.2.4.2	The user will be able to create new meal plan by tapping a "+" button. This button will call a new activity with 'selectList' of available recipes. This activity will have a button called "Add Recipe" which will call the meal plan class's constructor and the selected recipe to the list.
SRS 3.2.4.2.2	The meal plan object will use the addItemToGroceyList function to add items from selected recipes that are out of stock to the grocery list.

4.10 PantryApp Controller

4.10.1 UML Class Diagram

HomePage
- versionNumber : int - userLoggedIn : boolean - pantry : Pantry - accountCreator : AccountCreator
+ loggedInStatus() : void + getVersionNumber() : int + setVersionNumber(versionNumber : int) : void + greetingMessage() : void + launchTour() : void + launchPantry() : void + launchAccountManager : void + launchReminders() : void + checkReminders() : void + sendReminderNotification(remindersList : ArrayList<Reminder>) : void + checkExpirationDates() + sendExpiredNotifcation(expiredItemsList : ArrayList<Item>) : void + toggleGridView(gridView : boolean) : void

4.10.2 Design Elements

4.10.2.1 versionNumber

Type	Entity
Purpose	An int variable containing the version number of the application.
Description	The version number of the application. Ensures whether or not an update is necessary.

4.10.2.2 userLoggedIn

Type	Entity
Purpose	A bool variable determining whether or not the user has been logged in.
Description	A variable which indicates whether or not the user is logged in to the application.

4.10.2.3 pantry

Type	Entity
Purpose	The user's pantry.
Description	An encapsulation of the Pantry class (4.5).

4.10.2.4 accountCreator

Type	Entity
Purpose	The user's account creator.
Description	An encapsulation of the AccountCreator class (4.12).

4.10.2.5 loggedInStatus()

Type	Function
Purpose	Determines whether or not to show Home Page options.
Description	Checks the value of userLoggedIn. If it is true, then the Home Page screen is skipped and user is redirected to their Pantry by calling launchPantry(). If the value is false, the normal Home Page options are displayed to the user.

4.10.2.6 getVersionNumber()

Type	Function
Purpose	Returns the versionNumber of Stockpile.
Description	After testing whether or not the version number is null, the non-null value of Stockpile's current version is returned.

4.10.2.7 setVersionNumber(versionNumber : int)

Type	Function
Purpose	Sets the versionNumber of Stockpile.
Description	The versionNumber member variable is set to the parameter passed into the function upon application update.

4.10.2.8 greetingMessage()

Type	Function
Purpose	Contains the greeting to the Application.

Description	A string containing the following message: “Thank you for downloading Stockpile (version number)! Where should we begin?” The version number is obtained by calling the getVersionNumber() function.

4.10.2.9 launchTour()

Type	Function
Purpose	Allows user to view instructional video.
Description	Clicking the “Take a tour” button will launch an external web browser from the device to a YouTube video that explains how to use the application.

4.10.2.10 launchAccountManager()

Type	Function
Purpose	Redirects user to Account Creator page.
Description	Clicking the button “Create a profile/Sign in” will redirect the user to another screen containing an instance of the AccountCreator. The AccountCreator will allow the user to create a profile. If the user already has a profile, there will be an additional button on the screen that will redirect the user to a login page.

4.10.2.11 launchReminders()

Type	Function
Purpose	Allow the user to manage their reminders
Description	When the user clicks on reminders, this will invoke a pop up window/view where the user can create, add, or remove reminders.

4.10.2.12 checkReminders()

Type	Function
Purpose	Check User reminders.
Description	This is done by calling getReminders() on the Pantry object in this controller when Stockpile is opened and checking if the dates on the reminders matches the current date. If there are matches then sendReminderNotification() is called, passing an ArrayList of the given reminders, even if there is only one.

4.10.2.13 sendReminderNotification(remindersList : ArrayList<Reminder>)

Type	Function
Purpose	Send the User notifications regarding their reminders.
Description	A notification will be sent to the user for the given Reminder that is passed.

4.10.2.14 checkExpirationDates()

Type	Function
Purpose	Check expiration dates.
Description	This is done by calling getInventoryList() on the Pantry object in this controller when Stockpile is opened and comparing the current date with the dates of the Items from the List that is returned by getInventoryList(). If there is/are expired items, then they will be sent as an ArrayList to sendExpiredNotification(), if not, then nothing will happen.

4.10.2.15 sendExpiredNotification(expiredItemsList : ArrayList<Item>)

Type	Function
Purpose	Send the User notifications regarding their expired Items.
Description	A notification will be sent to the user for the Items that are passed in.

4.10.2.16 toggleGridView(gridView : boolean)

Type	Function
Purpose	Sets the view to list or grid.
Description	This button will allow the user to toggle back and forth between a list view or a grid view depending on their preference. A boolean variable will determine the current view status and switch to true or false when the user taps it, thus calling a different activity with a different view.

4.10.3 Analytical method

This class handles the Home Page, the Welcome Tour to Stockpile, and Account Creation. This is where the user starts their Pantry. The Home Page class is designed to allow the user to easily navigate to their desired location in Stockpile. The loggedInStatus function will help determine

whether a user will be shown the greetingMessage, which directs the user to the Home Page. This occurs when the user is not logged in. Otherwise, if the user is logged in, the greetingMessage function will be skipped and the launchPantry function will be executed instead, completely bypassing the Home Page. After the user has either logged in or created an account (via the launchAccountCreator function), the user will be able to open their pantry.

4.10.4 Design Concerns

SRS Section	Description
SRS 3.1.2.1	Stockpile shall display a greeting screen to the user when the user first opens the application.
SRS 3.1.2.2	The Greeting Screen shall display the options “Take a tour”, “Create a Profile”, and “Login”.
SRS 1.3.1.2.8	...Alert the user when there are items in the list with expiration dates that are older than the current date.
SRS 3.1.3.1	Stockpile shall display to the user a user manual explaining how to use the application and the various options.
SRS 3.1.3.2	Stockpile shall display a hyperlink at the bottom of the window that will open the user’s default browser and link to a video tutorial.

4.11 Account Controller

4.11.1 UML Class Diagram

AccountController
- user : User - currentView : View
+ createAccount() : void + updateView() : void + getLoginInformationInput() : void + isValidated() : bool + attemptLogin() : User + rememberMe() : void + signOut() : void

4.11.2 Design Elements

4.11.2.1 user : User

Type	Entity
Purpose	An entity which saves the current user.
Description	This entity holds the user which is currently logged in, the user which is being created, or will hold the user which is returned from the database on login.

4.11.2.2 currentView : View

Type	Entity
Purpose	The view which is currently being displayed.
Description	This entity represents the account page which needs to be displayed to the user.

4.11.2.3 createAccount() : void

Type	Function
Purpose	Calls the functions to allow the user to input information and verifies that information.
Description	Calls the getEmailAddress(), emailAddressExists(), getPassword(), passwordValid(), verifyEmailAddress() and addUser() functions to prompt users for data necessary , verify that data and create the user account.

4.11.2.4 updateView() : void

Type	Function
Purpose	Changes the view being displayed.
Description	Changes the view to be displayed based on the design element currentView.

4.11.2.5 getLoginInformationInput() : void

Type	Function
Purpose	Retrieve the user's email and password.

Description	Retrieves the user's login information from logging in.

4.11.2.6 isValidated() : bool

Type	Function
Purpose	Confirms if the user is successfully validated.
Description	Confirms that the database connection is valid and the user's information is valid.

4.11.2.7 attemptLogin() : User

Type	Function
Purpose	Attempts to login to the database with given login credentials.
Description	Attempt to login to the database. If the login is successful with the given credentials this function will return the related user information. Otherwise, nothing is returned.

4.11.2.8 rememberMe() : void

Type	Function
Purpose	Changes whether the User wishes to have his account information saved.
Description	When the user selects the checkbox labeled “Remember Me” this function shall either call the rememberUser() (4.13.2.5) function, or will change the checkbox to unchecked, indicating that the Users Login information will not be saved.

4.11.2.9 signOut() : void

Type	Function
Purpose	Signs the User out of the application.
Description	When the user selects the “Sign Out” button, this function will sign the user out of the current account. Also calls clearUser() (4.14.2.1).

4.11.3 Analytical method

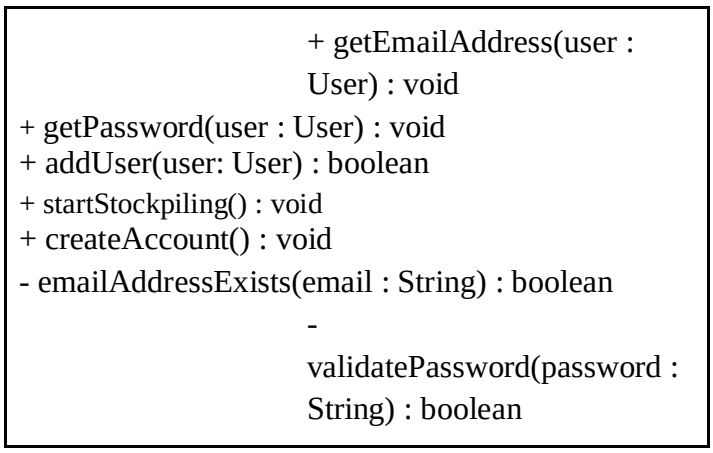
When the user first interacts with the stockpile app, they are required to create an account. This is done primarily through the createAccount() function. This function is used to call the various functions which prompt the user for the information necessary for creating an account and verifies this information.

When the user inputs their information, getLoginInformationInput() is called to retrieve the user’s input email and password. And then isValidated() is called to confirm that the database connection is valid and that the email and password entered in by the user are also valid and correct. Once the user’s email and password have been entered and validated, attemptLogin() is called to attempt to retrieve the account information from the database. If the attempt is successful than the related account information is then retrieved.

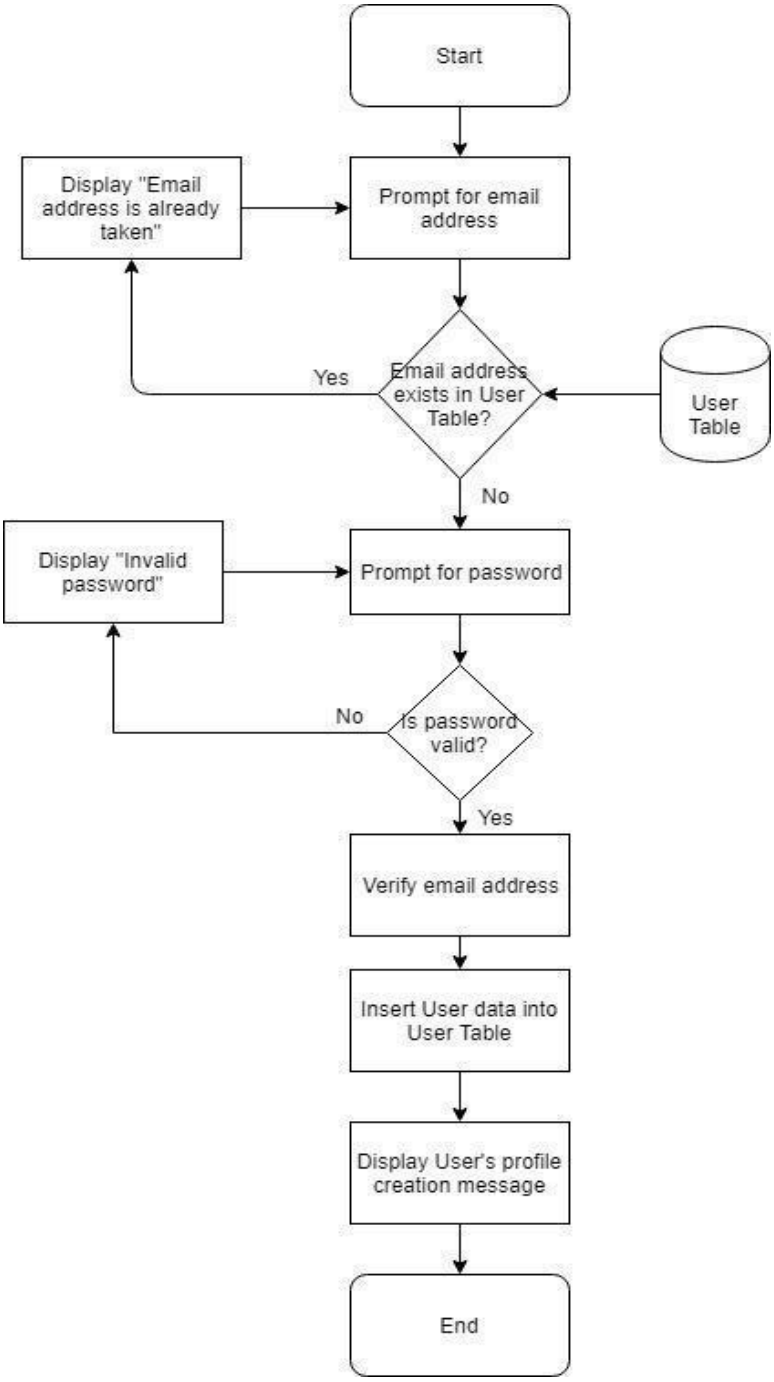
4.12 AccountCreator Class

4.12.1 UML Class Diagram

AccountCreator
- emailAddress : String - password : String - user : User



4.12.2 Flowchart


Start
Prompt for email address
Display "Email address is already taken"
Email address exists in User Table?
Yes
User Table
No
Prompt for password
Display "Invalid password"
Is password valid?
No
Yes
Verify email address
Insert User data into User Table
Display User's profile creation message
End

4.12.3 Design Elements

4.12.3.1 getEmailAddress(user : User) : void

Type	Function
Purpose	Allows the user to input an email address to create a profile.
Description	Displays a textbox in which the user will introduce an email address.

4.12.3.2 emailAddressExists(email : String) : boolean

Type	Function
Purpose	Allows Stockpile to verify if the email address introduced by the user already exists in the user table.
Description	Queries the User Table in the database to determine if the provided email address already exists. If it is already in the database, Stockpile will display a message to the user saying that the email address is already taken, and will prompt the User for the email address until a valid email address is provided.

4.12.3.3 getPassword(user : User) : void

Type	Function
Purpose	Allows the user to input a password to create a profile.
Description	Displays a textbox in which the user will introduce a password.

4.12.3.4 validatePassword(password : String) : boolean

Type	Function
Purpose	Allows Stockpile to know if the password has a valid length and contains only valid characters.
Description	Valid passwords must consist of at least one capital letter, at least one lowercase letter, at least one number or special character (! ? - @ ~), and be between 8-15 characters in length.

4.12.3.5 verifyEmailAddress(email : String) : boolean

Type	Function
Purpose	Allows Stockpile to know if the email address is valid by having the user verify its existence.
Description	Sends a verification email to the email address introduced by the user. The user must click on the link in the verification email to verify the email address.

4.12.3.6 addUser(user : User) : boolean

Type	Function
Purpose	Allows a new user object to be added
Description	Uses the email address and password received from the User object to create a new user. The user object represents the household. Returns true if user was successfully added to the User Table, false otherwise. Upon completion of this function, the “Start Stockpiling” button will be displayed to the user.

4.12.3.7 startStockpiling() : void

Type	Function
Purpose	Calls the PantryList activity
Description	A button placed on the account creator page that verifies the required fields are completed, then calls the PantryList class to allow the user to begin using Stockpile.

4.12.4 Analytical method

The function getEmailAddress() is designed to prompt the user to input an email address during profile creation. Once the user enters an email address, emailAddressExists() is called to check and verify if the email address is already in the system. Then, the user’s email is checked to determine if it is valid using verifyEmailAddress(). This function will check to insure the email is valid by sending an email to that email address. The user will have to verify the email address by clicking the link in the sent email. If the email is valid and isn’t already in the system, than the email will be successfully added to the User object.

The function getPassword() is then called and the user will then be prompted to enter a password into a textbox. This password will be required to access their account. To ensure the password is secure, passwordValid() is called checking the user’s input from getPassword() to determine if

input is of the correct length of 8-15 characters and only contains valid characters such as the required one capital letter, one lowercase letter, and a special character(! ? - @ ~). If the password is valid, then the password will be added to the User object.

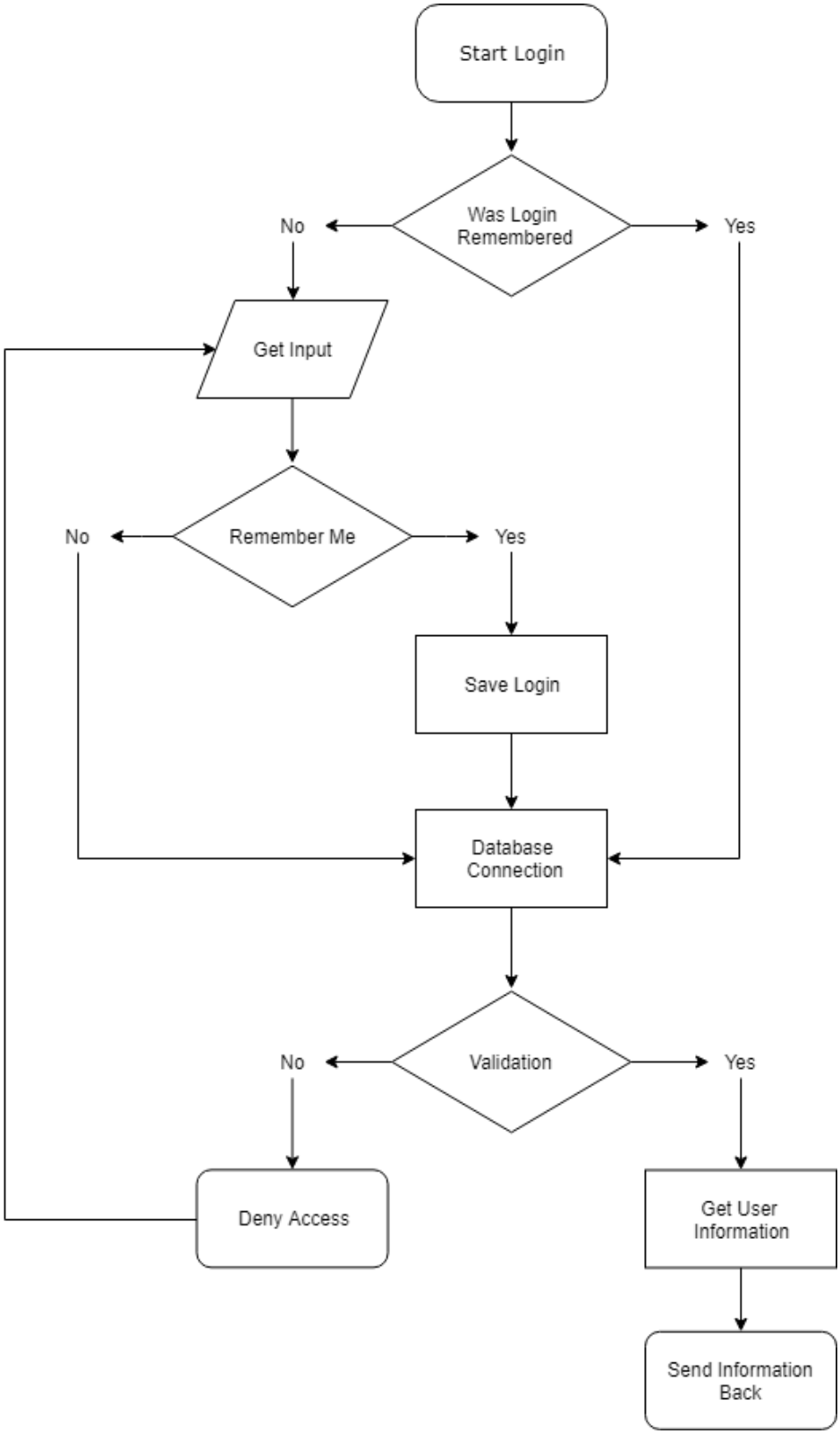
The user will be presented with text fields that ask for the user's email and a password so that their profile for Stockpile can be created. getEmailAddress() and getPassword() are called to take the users input. emailAddressExists() will be called to make sure that it is a valid email address. passwordValid() will be called to make sure that the user typed a password that meets the criteria detailed in section 3.2.5.1.2. When the user creates their profile successfully (no errors with email or password), verifyEmailAddress() will be called and send an email to the user so that they can verify the email. addUser() will be called after they create their profile so that a record can be saved into the database.

4.12.5 Design Concerns

SRS Section	Description
SRS 3.1.4.1	"If the user selects 'Create a Profile,' Stockpile will display a screen that will allow the user to create an account."
SRS 3.1.4.1.1	"Stockpile shall display input fields for the user to enter a valid email and a password."
SRS 3.1.4.1.2	Stockpile shall display a button labeled "Start Stockpiling" to complete the process of creating a profile.
SRS 3.1.4.1.3	"Stockpile shall display a message indicating whether the user's profile was created successfully."
SRS 3.1.4.1.4	Upon the successful creation of a profile, the user will be taken to the "Start Stockpiling" screen.
SRS 3.2.5.1	"Stockpile shall require an account to login for use."
SRS 3.2.5.1.1	"A verification email will be sent to verify the validity of an email address."
SRS 3.2.5.1.2	"Strong passwords should consist of at least one capital letter, at least one lowercase letter, at least one number or special character (! ? - @ ~), and be between 8-15 characters in length."

4.13 AccountLogin Class

4.13.1 Flowchart



4.13.2 Design Elements

4.13.2.1 getEmailAddress()

Type	Function
Purpose	Allows the User to input an email address to sign in.
Description	Displays a textbox in which the User will introduce an email address.

4.13.2.2 getPassword()

Type	Function
Purpose	Allows the User to input a password to sign in.
Description	Displays a textbox in which the User will introduce a password.

4.13.2.3 userStored()

Type	Function
Purpose	Determines if data is stored locally.
Description	Searches for whether the User's information was saved from previous login.

4.13.2.4 retrieveUser(email : String, password : String)

Type	Function
Purpose	Retrieves User data.
Description	If the rememberUser function returns true, places saved email address and password in displayed input fields.

4.13.2.5 rememberUser(email : String, password : String)

Type	Function
Purpose	Saves email and password.
Description	Saves email and password locally. The saved password will be saved as a hash and not plain text.

4.13.2.6 verifyEmailAndPassword(email : String, password : String)

Type	Process
Purpose	Verifies User.
Description	Compares inputted email and password with those stored in database. If Verification fails, an error message will be displayed and will return to fresh login page.

4.13.2.7 startStockpiling(email : String, password : String)

Type	Function
Purpose	Directs new users to the "Start Stockpiling" screen.
Description	Once there is a new record of email and password registered in the system, it is an indication that a new user's profile was created. startStockpiling() is responsible for bringing new users, who just registered for the first time in the application, to the "Start Stockpiling" screen.

4.13.3 Analytical method

The user will be presented with a few text fields to enter information. This information will be used to login into the app. The user will enter their email and password and then tap "Login". verifyEmailAndPassword() will check to see if those credentials exist. If they do, they will go into the app; if they don't, the user will be presented with an error message. After the fail message is presented, the app will refresh the screen/input fields so the user can try again. As the user is filling out their credentials, there will be a checkbox to allow the user to choose if they want their email address to be saved. The method rememberUser() will be invoked when they select this checkbox. The next time the user opens the app, the method userStored() will check if the user saved their credentials, and if they did, it will bypass the Login screen and go into the app.

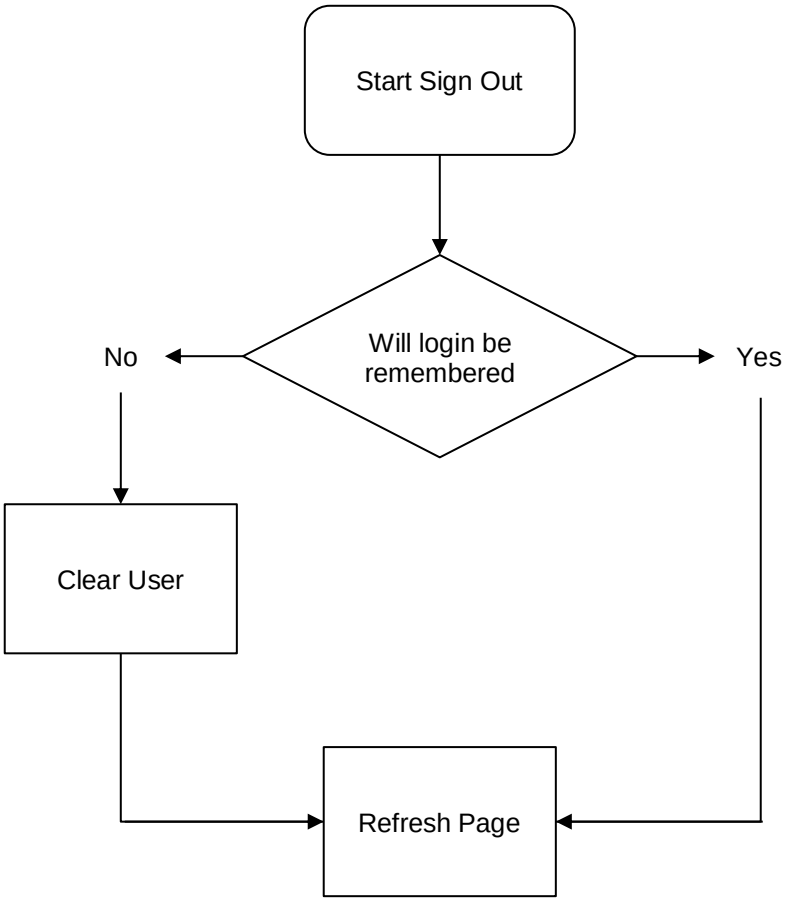
4.13.4 Design Concerns

SRS Section	Description
SRS 3.1.4.2	"If the user selects "Login," Stockpile shall display the login screen that will allow the user to login to an existing account."
SRS 3.1.4.2.1	"The login screen shall display input fields for the email address and password and also a checkbox labeled "remember me"."
SRS 3.1.4.2.2	"If the user previously selected the "remember me" option, the email address will be displayed in plain text in

	its corresponding input field.”
SRS 3.1.4.2.3	“If the user previously selected the “remember me” option, the password shall be displayed as ellipses or asterisks (*) in its corresponding input field.”
SRS 3.1.4.2.5	“Stockpile shall display an error message if the login was not successful.”
SRS 3.1.4.2.6	“Upon successfully logging in, Stockpile shall direct the user to the “Start Stockpiling” screen.”
SRS 3.7.4.1.3	Password hashes shall only be stored on the device if the user checked the “Remember me” checkbox on the login screen.

4.14 Sign Out

4.14.1 Flowchart



4.14.2 Design Elements

4.14.2.1 clearUser() : boolean

Type	Function
Purpose	Clears user's session data.
Description	After the user presses the "Sign Out" button, account credentials will be forgotten if the "remember me" checkbox wasn't selected. If "remember me" is selected, then the user's account credentials will be saved to allow a quicker login in the future.

4.14.2.2 refreshPage() : void

Type	Function
Purpose	Redirects user to the login page.
Description	Once the user is signed out, refreshPage() will take the user to the login screen.

4.14.3 Analytical method

When the user wants to sign out of the Stockpile app, they must activate the button labeled "Sign Out" which calls the signOut() function from the Account Controller Class. If the rememberUser() function (See 4.5.3.5) has been previously called, the user data will not be cleared. If it has not been called, the user's session data will be deleted. After the user has logged out of the Stockpile app, the refreshPage() function from the Account Controller Class will show the login page on the device's screen.

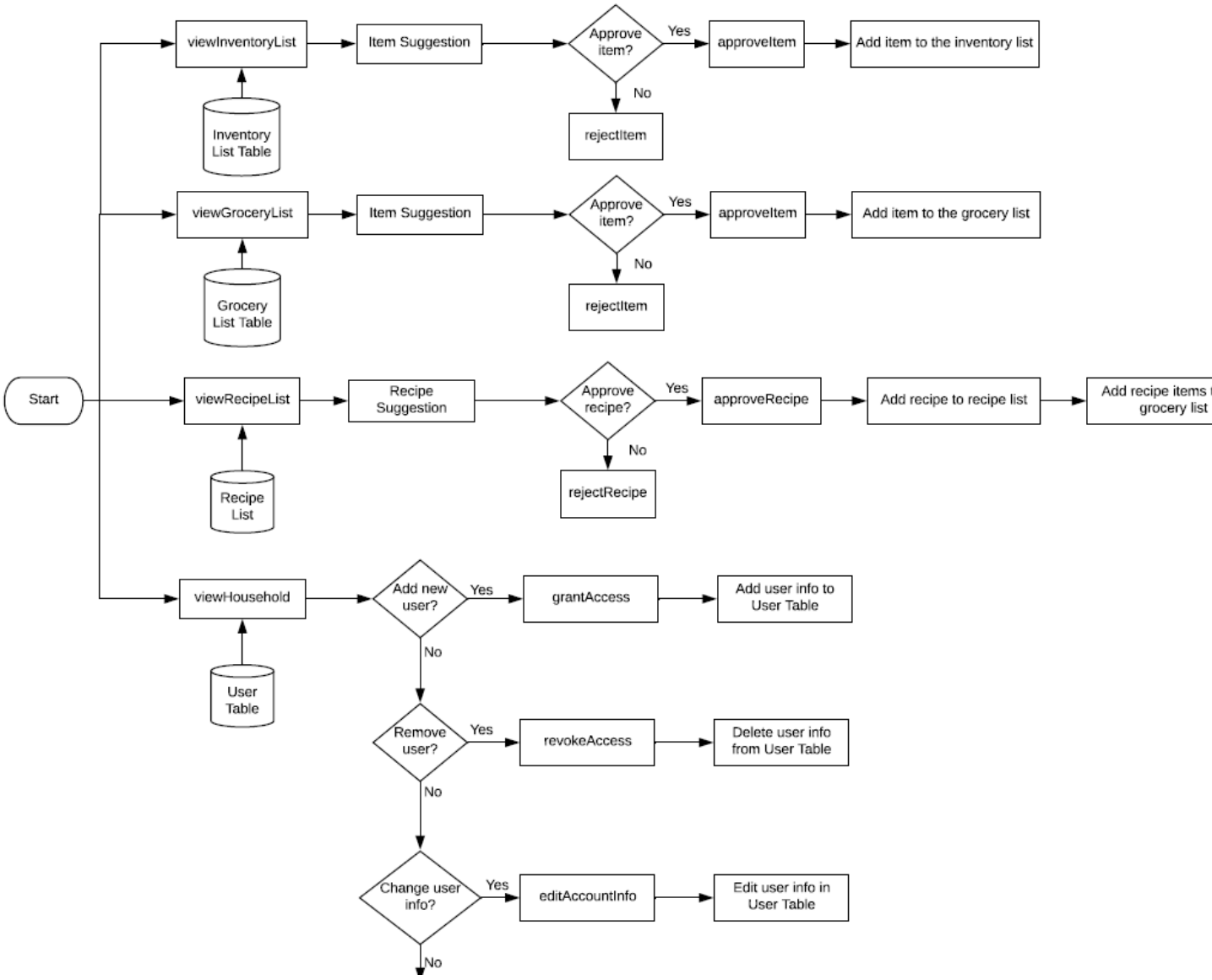
4.14.4 Design Concerns

SRS Section	Description
SRS 3.1.6.7	"When the user selects the sign out option, the system will log the user out of their account."
SRS 3.1.1.7	Stockpile shall display a sign-out feature that will log the user out of their account.
SRS 3.2.5.1.5	"Unless the user specifies "remember me," any saved credentials will be forgotten when the user signs out, requiring the user to re-enter their credentials to log back in."
SRS 3.8.1.6	"The system shall require a sign out button to enable users to sign out of the current account. The system shall require that if no user is currently logged in, the sign out feature will not be visible to the user. The system shall

	require that only one user shall be logged in at any time. This means that the user cannot be logged into multiple accounts at once.”

4.15 Admin Interface

4.15.1 Flowchart



4.15.2 Design Elements

4.15.2.1 viewHousehold()

Type	Function
Purpose	Allows the admin user to view all members of the current household.
Description	Gives the admin user access to view preferences, settings, likes, access level, etc. of each household member.

4.15.2.2 viewInventoryList()

Type	Function
Purpose	Allows the admin user to view the Inventory List.
Description	Gives the admin the ability to view all items in the Inventory List.

4.15.2.3 viewGroceryList()

Type	Function
Purpose	Allows the admin user to view the Grocery List.
Description	Gives the admin the ability to view all Items on the Grocery List.

4.15.2.4 viewRecipeList()

Type	Function
Purpose	Allows the admin user to view the Recipe List.
Description	Gives the admin the ability to view all Items on the Recipe List.

4.15.2.5 editAccountInfo(user : User)

Type	Function
Purpose	Allows the admin user to edit the information associated with a particular user account in the household.
Description	Gives the admin user access to edit preferences, settings, likes, access level, etc. of each household member.

4.15.2.6 approveItem(item_id : int) : boolean

Type	Function
Purpose	Allows the admin user to approve a suggested item to be added to the grocery or inventory list.
Description	Gives the admin user access to the suggested items to be added to the grocery or inventory list of each household member and the authority to accept the suggestion.

4.15.2.7 rejectItem(item_id : int) : boolean

Type	Function
Purpose	Allows the admin user to reject a suggested item from being added to the grocery or inventory list.
Description	Gives the admin user access to the suggested items to be added to the grocery or inventory list of each household member and the authority to reject the suggestion.

4.15.2.8 approveRecipe(recipe_id : int) : boolean

Type	Function
Purpose	Allows the admin user to approve a suggested recipe and add its ingredients to the grocery list.
Description	Gives the admin user access to the suggested recipes to be added to the local recipe database of each household member and the authority to approve the recipe.

4.15.2.9 rejectRecipe(recipe_id : int) : boolean

Type	Function
Purpose	Allows the admin user to reject a suggested recipe.
Description	Gives the admin user access to the suggested recipes to be added to the local recipe database of each household member and the authority to reject the recipe.

4.15.2.10 grantAccess(user_id : int) : boolean

Type	Function
Purpose	Allows the admin user to add users to the household.
Description	Gives the admin user access to add new users household membership to the existing household account.

4.15.2.11 revokeAccess(user_id : int) : boolean

Type	Function
Purpose	Allows the admin user to remove users from the household.
Description	Gives the admin user access to remove existing users household membership from the existing household account.

4.15.3 Analytical method

The Admin Interface (**4.4**) provides the methods necessary for the admin user to accomplish administrative tasks. All of the provided methods return a Boolean value indicating if the operation with the database was completed successfully.

Each household will have different users. The **viewHousehold**() method will allow the admin user to view the different members of the household.

It is the admin user's responsibility to add, remove, and modify Users.The grantAccess() method will allow the admin user to add a new user to the household. The new user data is retrieved from a User referenced in its parameter. **editAccountInfo**() takes the User's id as a parameter as well as a User object reference. **revokeAccess**() only needs the User's ID.

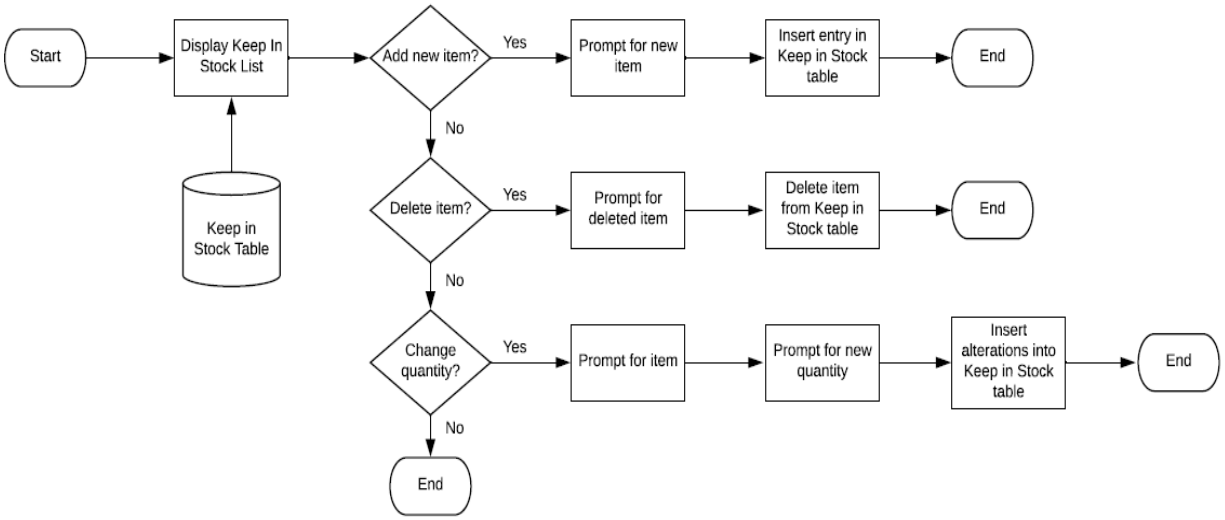
Any user may suggest an item or a recipe to be added; however, only the admin user has the privilege to approve them. Items and recipes are approved or rejected using the **approveItem**(), **rejectItem**(), **approveRecipe**(), **rejectRecipe**() methods.

4.15.4 Design Concerns

SRS Section	Description
SRS 3.2.5.2	The system shall allow administrative privileges to one user per household.
SRS 3.2.5.2.1	The system shall allow the administrator to grant or revoke access to the household's inventory for general users.
SRS 3.2.5.2.2	The system shall allow the administrator to approve or deny items added to the grocery list by users and the recipe planning feature.

4.16 Keep in Stock List

4.16.1 Flowchart



4.16.2 Design Elements

4.16.2.1 KeepInStockList(name : String, qty : int, price : double)

Type	Dictionary
Purpose	Keeps track of pantry items which the users would like to keep in stock.
Description	A dictionary with the key set to the Pantry Item in question, and an additional value stock amount which represents the quantity the user wants the pantry Item to stay at.

4.16.2.2 addItem(name : String, qty : int, price: double)

Type	Function
Purpose	Adds a new item to KeepInStockList.
Description	This function adds a new item to KeepInStockList using parameters containing the Item's attributes specified by the user or the database.

4.16.2.3 deleteItem(name : String, qty : int)

Type	Function
Purpose	Deletes an item from KeepInStockList.
Description	This function removes an item from the KeepInStockList. The user will indicate that the item should be removed by choosing "delete item" option in the item's submenu in the "keep in stock" section of the application.

4.16.2.4 changeQuantity(name : String, qty : int)

Type	Function
Purpose	Changes the quantity of an item in the KeepInStockList.

Description	This function allows the user to change the quantity of items they want to always have in the Keep in Stock list by incrementing, decrementing or setting the quantity to a user specified amount. The user will specify this amount in an input that will display itself after the user selects the “change quantity” option in the item’s submenu in the “keep in stock” section of the application.

4.16.2.5 displayKeepInStock()

Type	Function
Purpose	Displays all items from the Keep in Stock list.
Description	This function displays all items from the user’s Keep in Stock list (SRS 3.1.7.1).

4.16.2.6 keepInStockAlert()

Type	Function
Purpose	Alerts the user when the quantity of specified items is running low.
Description	This function compares the quantity of the KeepInStockList to the PantryList. If the quantity falls below a certain threshold, this function will display an alert to the user with an option to add the Item/Items to the current Grocery List.

4.16.3 Analytical method

KeepInStockList is a class responsible for tracking amounts of desired items in the pantry and alerting the user when they should purchase more. The main functions are **addItem(), deleteItem(),** and **changeQuantity()**. These functions will allow the user to specify which items should be tracked and at what quantity. The KeepInStockList is compared with the PantryList. If items in the KeepInStockList do not exist in the PantryList, or if the quantity of existing items falls below a user specified threshold, the **keepInStockAlert()** function will alert the user and prompt them to add such items to the GroceryList.

4.16.4 Design Concerns

SRS Section	Description
SRS 3.1.7.1	The system shall display a list of all the items in the Keep in Stock list.

SRS 3.2.1.12, SRS 3.2.1.1.3	The system shall allow the user to change the Keep in Stock list by adding or deleting items.
SRS 3.2.1.1	The system shall allow the user to change the quantity of items in the Keep in Stock list.
SRS 3.18.4	When the quantity of an item falls below of the desired "Keep in Stock" amount, the system shall display a message asking the user if they want to add an item to the grocery list.
SRS 3.1.7.2	When the user selects an item, Stockpile shall display a submenu that will display available options for each item next to the user's selection.
SRS 3.1.7.2.1	The submenu shall display the following options: "Delete item" and "Change Quantity".
SRS 3.1.7.2.2	When the user selects "Delete item", Stockpile will display a message to the user verifying that the user actually wants to delete the item.
SRS 3.1.7.2.3	When the user select "Change Quantity", Stockpile will display an input field allowing the user to enter in the new desired quantity.

4.17 BarcodeScanner

4.17.1 UML Class Diagram

BarcodeScanner
- scanner : BarcodeScanner + addItemBarcode(barcode : String) : void + displayCameraScreen() : void + displayBarcodeInterface() : void - displayIfSuccessfulScan() : void - cancelScan() : void - scanItem() : void - askPermission(): void

4.17.2 Design Elements

4.17.2.1 BarcodeScanner

Type	Entity
Purpose	Allows the user to scan an item's barcode using the camera.
Description	Stockpile shall display a rectangle on the camera screen showing the user where the barcode will be on the screen (SRS 3.1.11.1.1).

4.17.2.2 addItemBarcode(barcode : String)

Type	Function
Purpose	Adds a scanned item into the user's pantry list.
Description	Uses the scanned barcode in order to determine the UPC. Using the UPC, a search is done through a database containing UPCs in order to identify the scanned item and its corresponding information (SRS 3.2.7.2).

4.17.2.3 displayCameraScreen()

Type	Function

Purpose	To display the camera screen to the user.
Description	Displays the device's camera screen to the user so that they can see what they are attempting to scan with the barcode scanner.

4.17.2.4 displayBarcodeInterface()

Type	Function
Purpose	To display a barcode scanner overlay on the camera screen.
Description	Displays a barcode scanner overlay on the camera screen to help guide the user to use the barcode scanner properly.

4.17.2.5 displayIfSuccessfulScan()

Type	Function
Purpose	To signify to the user that an item has been successfully scanned with the barcode scanner and added to the inventory list.
Description	After the system confirms that the item has been successfully scanned with the barcode scanner and then added to the inventory list, a big green check mark is displayed on the screen to notify the user that the item was successfully added. The check mark then fades after a few seconds.

4.17.2.6 cancelScan()

Type	Function
Purpose	Allows the user to cancel an initiated scan.
Description	The user will be returned to the previous screen that they were on before the scan was initiated.

4.17.2.7 scanItem()

Type	Function
Purpose	Allows the user to scan an item using their camera and add the said item to their inventory list.
Description	When the user navigates to the barcode scanner from the pantry inventory screen, Stockpile will display the camera screen to the user (SRS 3.1.11).

4.17.2.8 askPermission(): boolean

Type	Function
Purpose	Allows the user to indicate if the application can access the device's built in camera.
Description	Returns true if the user grants permission, otherwise returns false

4.17.3 Analytical method

The barcode scanner utilizes Google Vision API, which allows the user to take a picture of a product's barcode, and then extracts an item's information. The information is then saved and used to create a new Item object. This object is then inserted it into the users' purchase inventory.

4.17.4 Design Concerns

SRS Section	Description
SRS 3.1.8.3	Stockpile shall display a button, which will allow the user to add items using the item's barcode.
SRS 3.1.11.1	When the user navigates to the barcode scanner from the pantry inventory screen, the system will display the camera screen to the user.
SRS 3.1.11.1.1	When the user navigates to the barcode scanner from the pantry inventory screen, the system will display the camera screen to the user.

SRS 3.1.11.1.2	The system shall display a message instructing the user how to scan an item into their pantry inventory.
SRS 3.1.11.2	The system shall indicate to the user if the item was successfully scanned in.
SRS 3.1.11.3	Stockpile shall display a way for the user to update the quantity of an item once the user has successfully scanned into their pantry inventory.
SRS 3.1.11.3.1	Stockpile shall display the updated quantity of the item.
SRS 3.2.2.3	Stockpile shall add items to the pantry inventory as the user scans an item's barcode.
SRS 3.2.7.1	The system shall use the device's camera to scan a barcode.
SRS 3.2.7.2	The system shall search for the product using the item's UPC in a database.
SRS 3.6.2	Stockpile shall request and obtain access to the camera from the user before Stockpile enables the barcode scanner feature.
SRS 3.6.2.1	The barcode scanner feature shall be disabled if the user denies the app access to the device's camera.
SRS 3.7.1.1	Stockpile shall use a picture of the barcode to get the UPC and return product information.

4.18 GroceryList

4.18.1 UML Class Diagram

GroceryList
- groceryList : ArrayList<Item> - displayChoice : ArrayList<String>

+ displayGroceryList() : void
+ transferGroceryList() : void
+ modifyGroceryListItem(name : String) : void
+ deleteGroceryListItem(name : String) : void
+ changeQuantityOfGroceryListItem(name : String, qty : int) : void
+ addNewItemToGroceryList(name : String, qty : int, price : double) : void
- sortGroceryList() : void

4.18.2 Design Elements

4.18.2.1 displayGroceryList()

Type	Function
Purpose	Displays the items in the Grocery List.
Description	This function will pull up all the items in the GroceryList and display them to the user.

4.18.2.2 transferGroceryList()

Type	Function
Purpose	Updates the pantry inventory with the items in the GroceryList.
Description	This function interfaces with the PantryList and modifies the PantryList to include or change the quantity of items found in the GroceryList.

4.18.2.3 modifyGroceryListItem(name : String)

Type	Function
Purpose	To make changes to one item in the Grocery List.
Description	Give the user the ability to modify an item that the User selects by deleting it or changing the quantity of the item.

4.18.2.4 deleteGroceryListItem(name : String)

Type	Function
Purpose	To remove an Item from the Grocery List.
Description	Removes an item chosen by the user from the Grocery List. Will prompt the user to confirm the deletion before it takes place.

4.18.2.5 changeQuantityOfGroceryListItem(name : String, qty : int)

Type	Function
Purpose	To change the quantity of the Item in the Grocery List.
Description	Changes/updates the quantity of an Item in the Grocery List.

4.18.2.6 addNewItemToGroceryList(name : String, qty : int, price : double)

Type	Function
Purpose	To add a new Item object to the Grocery List.
Description	Add a new Item to the Grocery List. Used by outside classes such as Pantry when an item is out of stock or is critically low. Used internally in the class as well when the User wants to add an Item themselves.

4.18.2.7 sortGroceryList()

Type	Function
Purpose	To change the order in which the User sees the Grocery List.
Description	Change the order in which the Grocery List is displayed to the User to alphabetical or by type/category (Dairy, Produce, Grains, Meat, etc.)

4.18.2.8 groceryList

Type	Entity

Purpose	ArrayList of Items. Contains a list of all the Items in the Grocery List.
Description	Takes advantage of the Item class functionality.

4.18.2.9 displayChoice

Type	Entity
Purpose	ArrayList of strings that describe different view options for lists.
Description	Options for the user to choose which way to display the list.

4.18.3 Analytical Method

The Grocery List interface allows the user to access and modify the Grocery List database. The user can choose to view the grocery list stored in the database with the displayGroceryList() method. This method will transfer the data from the database to the user's view. If the data contained in the grocery list is invalid or unwanted, the user can choose to remove the data from the list with the deleteGroceryListItem() function. This function will take the selected items as a parameter to delete them from the database. To change an item on the list, the user can use the modifyGroceryListItem() function to change a record with new data the user will provide. If the user wants only to change the quantity of an item, the changeQuantityOfGroceryListItem() function will take the new quantity from the view and update the quantity field in the for that item. When the user finishes shopping for groceries, they can use the transferGroceryList() function to add new items and quantities to the pantry list, or update the quantity of an item that is already present. To better view the data in the grocery list, the user can sort the items in the grocery list alphabetically or by type or category with the sortGroceryList() method. This method will sort according to the name, type, or category fields that are stored in the database.

4.18.4 Design Concerns

SRS Section	Description
SRS 3.1.9.1	"Stockpile shall display a list of items that the user wants to purchase during their next trip to the grocery store."
SRS 3.1.9.2	"When the user selects an item, Stockpile will display a submenu for each item."

SRS 3.1.9.2.1	“The submenu shall display the following options: ‘Delete item” and ‘Change Quantity.’”
SRS 3.1.9.2.2	“When the user select ‘Delete item,’ Stockpile will display a message to the user verifying that the user actually wants to delete the item.”
SRS 3.1.9.2.3	“When the user selects ‘Change Quantity,’ Stockpile will display an input field allowing the user to enter in the new desired quantity.”
SRS 3.1.9.3	“Stockpile shall display a checkbox next to each item in the grocery list.”
SRS 3.1.9.4	“The Grocery List shall allow the user to sort the list alphabetically or by type.”
SRS 3.2.3.1	“Stockpile shall automatically add items to the grocery list after certain events occur.”
SRS 3.2.3.1.1	“Stockpile shall add items to the grocery list when item quantity falls below the desired ‘Keep in Stock’ amount.”
SRS 3.2.3.1.2	“Stockpile shall add items to the grocery list when the user adds recipes to their meal plan.”
SRS 3.2.2.1	“The inventory list shall interface with the grocery list to add items to the inventory list.”
SRS 3.2.2.2	“The system shall add an item to the inventory list when the user taps the item’s checkbox in the grocery list and then taps the ‘add to pantry’ button.”

4.19 Meal Plan

4.19.1 Meal plan UML Class Diagrams

4.19.1.1 Meal plan UML Class Diagram

Meal Plan
+ isScheduled : boolean + picture : File

+ pantryItems : ArrayList<Item> + baseIngredients : ArrayList<Item> + directions : String + numberServed : int + recipeList : ArrayList<Recipe> + recipeTitle : String
- getBaseIngredients() : ArrayList<Item> - changePantryItem(ingredientQty : int, itemQty : int) : boolean - addItemToShoppingList() : boolean - updatePicture(file : File) : boolean - addRecipe(recipe : Recipe) : Recipe - displayRecipeList() : void - removeRecipie(recipe : Recipe) : void

4.19.1.2 Meal Calendar UML Class Diagram

MealCalendar
+ mealCalendar : MealCalendar
+ mealList : ArrayList<MealPlan>
- displayMealCalendar() : void

4.19.2 Meal Plan Entities

4.19.2.1 Directions

Type	Entity
Purpose	A list of Directions of how to create the selected meal.
Description	The Directions will be displayed after the user selects a recipe from the list of saved recipes. This will be a string variable which will either be auto-filled by information received from an online database or filled by the user of the application.

4.19.2.2 isScheduled

Type	Entity
Purpose	Represents whether or not the user has chosen to make this meal plan.
Description	This boolean variable will be used to indicate whether the user has selected to create the selected meal plan in the future. If isSheduled is set to true the system will notify of any ingredients that are missing and prompt the user if they wish to add the missing items to their grocery list using the

	addItemsToShoppingList function.

4.19.2.3 picture

Type	Entity
Purpose	A picture of the finished meal.
Description	Holds an image of the finished meal. This variable will hold the file path to the image. The user will have the option to update this image using the updatePicture(file) function.

4.19.2.4 pantryItems

Type	Entity
Purpose	A list of all items in a pantry.
Description	Gets the list of all items in the pantry. Every item is represented by a variable named pantryItem, which is string data type.

4.19.2.5 baseIngredients

Type	Entity
Purpose	A list of all of the ingredients required to make a recipe.
Description	Gets a list of all ingredients involved in a recipe.

4.19.2.6 numberServed

Type	Entity
Purpose	An integer noting how many people this meal can feed.
Description	A number which represents the total number of people which can be feed with the given meal.

4.19.2.7 recipeList

Type	Entity
Purpose	A list of all of the recipes created by the user.
Description	An array containing each recipe input by the user.

4.19.2.8 recipeTitle

Type	Entity

Purpose	The name of a recipe.
Description	A string containing the name of a recipe.

4.19.2.9 mealSchedule

Type	Function
Purpose	A list of all of the meals currently scheduled.
Description	An array of recipes with an attached date value to signify when the recipe is meant to be made and eaten.

4.19.2.10 getBaseIngredients()

Type	Function
Purpose	Retrieve the list of ingredients from the Recipe.
Description	Gets the list of ingredients required for the Recipe.

4.19.2.11 changePantryItem(ingredientQty : int, itemQty : int)

Type	Function
Purpose	Changes the quantities of pantry items.
Description	Subtracts the quantities from getBaseIngredients() from the quantity of the matching pantry list item to reflect the use of the ingredient in the recipe. Returns “True” if the operation was successful, otherwise “False”.

4.19.2.12 addItemToShoppingList() : boolean

Type	Function
Purpose	Adds items to the shopping list that are missing from the pantry list.
Description	Adds items to the shopping list if an item in getBaseIngredients() is either not in the pantry list or if the quantity required in getBaseIngredients() exceeds the quantity of the item currently in the pantry list. Returns “True” if the operation was successful, otherwise “False”.

4.19.3.13 updatePicture(file : File) : boolean

Type	Function
Purpose	Updates the picture for the recipe with the file passed into this

	function
Description	Will take in a file, and check to see if it is a valid file path. If the file path is to something over than a valid picture file, this function will return false. Otherwise, this function will return true. Returns “True” if the operation was successful, otherwise “False”.

4.19.2.14 addRecipe(recipe : Recipe) : Recipe

Type	Function
Purpose	Adds a user generated recipe to the recipe list
Description	The user enters the ingredients, name and directions of a recipe which are then combined into a recipe and added to the recipe list. Returns “True” if the operation was successful, otherwise “False”.

4.19.2.15 displayRecipeList() : void

Type	Function
Purpose	Displays the list of user generated recipes.
Description	Iterates through the list of recipes and displays each one on the screen.

4.19.2.16 displayMealCalendar() : void

Type	Function
Purpose	Displays a calendar with each of the planned meals on it.
Description	Generates a calendar with the meals scheduled by the user indicated on their associated planned days..

4.19.2.17 removeRecipe(recipe : Recipe) : void

Type	Function
Purpose	Removes a recipe from the list.
Description	Accepts a Recipe parameter and compares that object to existing objects in the list. If a match is found, the object is removed, otherwise the function returns NULL.

4.19.3 Analytical Method

The Meal Plan will allow the user to create a weekly or monthly scheduled list of recipes that they wish to make during that period of time. The user will either be able to select recipes from a

list of previously saved recipes, or will be able to manually input any number of new recipes using addRecipe(). Recipes that the user has previously saved using displayRecipeList() and all recipes whose baseIngredients array are contained within the pantryItems array will be displayed with priority. Once the selection of recipes is complete the user can choose to either schedule that meal plan or save it for another time. If isScheduled and a recipe added to the Meal Plan does not have all baseIngredients contained in the pantryItems array, addItemToShoppingList() is used to add any ingredient not contained in the pantryItems. If a Meal Plan isScheduled, displayMealCalender() is called. Once a recipe from the Meal Plan is made, changePantryItem() is called to update pantryItems.

4.19.4 Design Concerns

SRS Section	Description
SRS 3.1.6.6	When the user selects the "Meal Planning" button, Stockpile will display the meal-planning screen.
SRS 3.1.10.1	Stockpile shall display a list of recipes the user has selected from a list of user-added recipes.
SRS 3.1.10.2	Stockpile shall display the name of each recipe.
SRS 3.1.10.3	Stockpile shall display a picture of each recipe.
SRS 3.2.4.1	Stockpile shall provide a way for the user to manually add recipes to their recipe box.
SRS 3.2.4.2	Stockpile shall provide a way for the user to create a weekly or monthly meal plan.
SRS 3.2.4.2.1	Stockpile shall provide existing recipes from the database that the user could make with their current inventory.
SRS 3.2.4.2.2	Stockpile shall determine which ingredients from a selected recipe are not available in the inventory
SRS 3.2.4.2.3	Stockpile shall add the items from the recipe not found in the pantry inventory to the grocery list when the user adds a recipe to their meal plan.

4.20 Recipe Grocery List Interaction

4.20.1 UML Class Diagram

Recipe Grocery List Interaction
+ baseIngredients : ArrayList<Item> + pantryItems : ArrayList<Item>

- getBaseIngredients() : ArrayList<Item> - getPantryItems() : ArrayList<Item> - compareBaseToStock() : ArrayList<Item> - addItemsToGroceryList() : boolean

4.20.2 Design Elements

4.20.2.1 baseIngredients

Type	Entity
Purpose	A list of all of the ingredients required to make a recipe.
Description	Gets a list of all ingredients involved in a recipe.

4.20.2.2 pantryItems

Type	Entity
Purpose	A list of all items in a pantry.
Description	Gets the list of all items in the pantry. Every item is represented by a variable named pantryItem, which is string data type.

4.20.2.3 getBaseIngredients()

Type	Function
Purpose	Retrieve the list of ingredients from the Recipe.
Description	Gets the list of ingredients and quantities required to cook the recipe the recipe file.

4.20.2.4 getPantryItems()

Type	Function
Purpose	Retrieve the pantry items matching the ingredients from getBaseIngredients().
Description	Searches through the pantry items list and returns all pantry items matching the ingredients returned from getBaseIngredients() along with their associated quantities.

4.20.2.5 compareBaseToStock()

Type	Function
Purpose	Determine what items need to be added to the shopping list.

Description	Compares what is required for the recipe against what is currently in the pantry items list and returns a list of items to be passed to the addToShoppingList() function.

4.20.2.6 addItemsToGroceryList()

Type	Function
Purpose	Adds items to the shopping list that are missing from the pantry list.
Description	Adds items to the shopping list if an item in getBaseIngredients() is either not in the pantry list or if the quantity required in getBaseIngredients() exceeds the quantity of the item currently in the pantry list. Returns “True” if the operation was successful, otherwise “False”.

4.20.3 Analytical Method

The groceryScreenTransition class is responsible for creating a interface between adding to the user’s grocery list and the user’s meal planner and pantry list. The functions getBaseIngredients(), getPantryItems(), and compareBaseToStock() to check for missing ingredients in your pantry and recipes. addItemsToGroceryList() will add missing items to shopping lists.

4.20.4 Design concerns

SRS Section	Description
SRS 3.2.4.2.2	Stockpile shall determine which ingredients from a selected recipe are not available in the inventory.
SRS 3.2.4.2.3	Stockpile shall add the items from the recipe not found in the pantry inventory to the grocery list when the user adds a recipe to their meal plan.

4.21 MealPlanningPantry Interface

4.21.1 UML Class Diagram

MealPlanningPantry Interface
- pantry : ArrayList<Item>
+ getPantryItems() : ArrayList<Item> + loadPantry() : ArrayList<Item> + updatePantry(itemsList : ArrayList<Item>) : boolean

4.21.2 Design Elements

4.21.2.1 pantry

Type	Entity
Purpose	Represents all Items within the user's pantry
Description	An ArrayList of all of the Items in the user's pantry.

4.21.2.2 getPantryItems()

Type	Function
Purpose	Returns an ArrayList of all the Items in the Pantry from the database.
Description	This function will be the liaison (or connection) for any class or function that requires the use of the master list of Pantry Items. It does not accept parameters and return an array of the Items in the Pantry.

4.21.2.3 loadPantry()

Type	Function
Purpose	Returns an ArrayList of all of the Items in the Pantry from the database.
Description	This function will pull an updated list of all the Items in the Pantry and store it in a local variable.

4.21.2.4 updatePantry(itemsList : ArrayList<Item>)

Type	Function
Purpose	Updates the database with any changed or removed Items from the Pantry. Returns “True” if the operation was successful, otherwise “False”.
Description	This function will push an update to the database, it will take the updated array of Pantry Items in its parameters and update the database according to the changes made.

4.21.3 Analytical Method

The mealPlanningPantryInterface is the sole connection between the user’s Pantry and the Meal Planning view. The functions loadPantry and updatePantry will interact with the recipe table and the inventory table in the database to retrieve and update the user’s Pantry. getPantryItems returns all of the Items in the user’s Pantry.

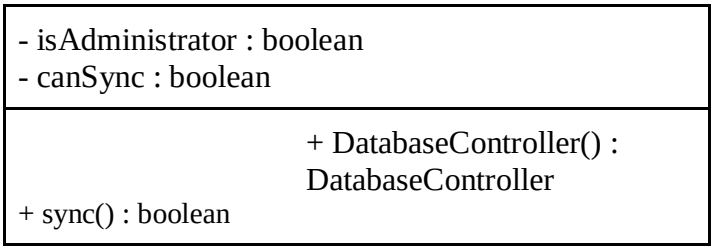
4.21.4 Design Concerns

SRS Section	Description
SRS 3.2.4.2.1	Stockpile shall provide existing Recipes from the database that the user could make with their current inventory.
SRS 3.2.4.2.2	Stockpile shall determine which ingredients from a selected recipe are not available in the inventory

4.22 Database Controller

4.22.1 UML Class Diagram

Database Controller
- user_id : int - cloudDatabaseReference : DatabaseReference - localDatabaseReference : DatabaseReference



4.22.2 Design Elements

4.22.2.1 DatabaseController

Type	Entity
Purpose	Class controller for database interaction.
Description	Controls interactions between the cloud and local databases.

4.22.2.2 user_id

Type	Entity
Purpose	Unique identifier for each User.
Description	Unique integer retrieved from the database that is associated with each User.

4.22.2.3 cloudDatabaseReference

Type	Entity
Purpose	Entry point for cloud database.
Description	Entry point for interactions with the cloud database.

4.22.2.4 localDatabaseReference

Type	Entity
Purpose	Entry point for local database.
Description	Entry point for interactions with the local database.

4.22.2.5 isAdministrator

Type	Entity
Purpose	User attribute identifier for administrator privileges.
Description	User attribute that identifies whether a user has administrative privileges or not.

4.22.2.6 canSync

Type	Entity
Purpose	Flag for whether the user allows synchronization over cellular data networks.
Description	Checks to see if cloud and local databases are ready to be merged. Returns true if sync is possible, false otherwise. If no internet connection is present, the function will return false.

4.22.2.7 sync()

Type	Function
Purpose	Initialize the DatabaseController Singleton. Data is merged between local database and cloud database.
Description	Synchronizes data between local and cloud databases and merges any changes that it detects so that both contain the same data.

4.22.3 Analytical Method

The DatabaseController class is a Singleton whose sole purpose is to synchronize data between the local database and the cloud database. There will only be one instance of the DatabaseController, which can be called statically from any class or activity to sync data.

When used for the first time, DatabaseController will initialize by determining if the current User Type is authorized to enact all or some CRUD operations (isAdministrator) and if the data connection is allowed to send large packets of data (canSync). The Username of the account is then used to retrieve and store the user ID. Finally, the local and cloud database entry points are initialized as cloudDatabaseReference and localDatabaseReference.

When sync() is called, data between the local database and cloud database will be merged. This can be used to keep Pantry Items synchronized if two users in the same household are updating their Inventory at the same time. It can also be used to pull down all data if a user switches to a new device.

4.22.4 Design Concerns

SRS Section	Description
SRS 3.5.1	Stockpile shall save data in a database hosted on a cloud storage service and in local memory.
SRS 3.5.2	The system database shall allow write, read, update, and delete abilities for the household administrator of an account.

SRS 3.5.3	The system database shall remain synchronized across all devices tied to the user's account.
SRS 3.5.3.1	The system shall allow the user to specify if the cloud database should be synchronized using a cellular internet connection.
SRS 3.6.1	Stockpile shall require an internet connection on the device before Stockpile attempts to synchronize data to and from the server's database.
SRS 3.6.1.2	When the device has re-established an internet connection, on-device app data will synchronize to the online app database

4.23 Reminder Class

4.23.1 UML Class Diagram

Reminder
- reminderText : string - dateToBeReminded : Date
+ Reminder(reminderText, dateToBeReminded : Date) : Reminder

4.23.2 Design Elements

4.23.2.1 Reminder(reminderText, dateToBeReminded)

Type	Function
Purpose	Creates a Reminder object with the given parameters.
Description	Creates a Reminder object with the given parameters. A Date object will be created with the given parameters and stored in dateToBeReminded.

4.23.2.2 reminderText

Type	Entity
Purpose	Holds text that the user entered.
Description	Contains the text of what the user wants to be reminded about.

4.23.2.3 dateToBeReminded

Type	Entity
Purpose	Holds the date the user would to be reminded on.
Description	Contains the date that the user would like be reminded on.

4.23.3 Analytical Method

The Reminder class will be used to create an object that will be used to save when the user would like to be reminded and what they want to be reminded about.

4.23.4 Design Concerns

SRS Section	Description
SRS 1.3.1.7	Stockpile shall be capable of an array of operations including updating the pantry database, setting reminders, sending notifications, and saving recipes and grocery lists.

4.24 Analysis Class

4.24.1 UML Class Diagram

Analysis
- purchasedList : ArrayList<Item>
- isAdmin() : boolean + trackPurchase(item : Item) : void + showPurchases() : ArrayList<Item> + displaySuggestion() : void

4.24.2 Design Elements

4.24.2.1 purchasedList

Type	Entity
Purpose	Represents all items the user has purchased.
Description	A list of purchased items updated by trackPurchase()

4.24.2.2 trackPurchase(item : Item)

Type	Function
Purpose	Store additions to the pantry by each user in the database.
Description	This function will be called whenever an item is added to the pantry. This will allow Stockpile to have a list of all the purchases by each user.

4.24.2.3 showPurchases()

Type	Function
Purpose	Returns an array of all the items in the purchased list from the database.
Description	This function will display to the user a list of all the purchases made, by who and the quantity of items purchased.

4.24.2.4 displaySuggestion()

Type	Function
Purpose	Displays similar items to what the user purchased in order to help them know what else they could add to their pantry in order to keep it fully stocked.
Description	This function will compare the purchased items list with the database in order to provide the user with useful information about what else they might need in their pantry.

4.24.2.5 isAdmin()

Type	Function
Purpose	Returns whether the user is an administrator.
Description	Because analytical data is only available to the Admin, the Analysis class will only call it’s functions if the user is an admin.

4.24.3 Analytical Method

The Analysis class is one of the most useful and interesting parts of the Stockpile app. The user must be able to understand their purchase history in order to have full control of their pantry. The functions within this class are centered around storing information for the user to be able to review at any time.

4.24.4 Design Concerns

SRS Section	Description
3.2.6.1	Stockpile shall provide analytical data to the administrative user
3.2.6.2	Stockpile shall provide data about purchases
3.2.6.2.1	Stockpile shall provide data about quantities of items purchased
3.2.6.3	Stockpile shall provide data about the household's inventory
3.2.6.3.1	Stockpile shall provide to following data about inventory items removed from the pantry inventory: the date of removal, the user who removed it, and the quantity of items used.
3.2.6.4	Stockpile shall display suggestions of items that could be added to the grocery list

5.0 Appendices

5.1 Viewpoint Sources:

5.1.1 Model View Controller Design Pattern

"IBM Knowledge Center", Ibm.com, 2017. [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSZLC2_7.0.0/com.ibm.commerce.developer.doc/concepts/csdmvcdespat.htm. [Accessed: 08- Dec- 2018].

5.1.2 Crow's Foot Notation

"Crow's Foot Notation", Www2.cs.uregina.ca, 2004. [Online]. Available: http://www2.cs.uregina.ca/~bernatja/crowsfoot.html. [Accessed: 08- Dec- 2018].

5.1.3 UML Class Diagram

S. Ambler, "UML 2 Class Diagrams: An Agile Introduction", Agilemodeling.com, 2013. [Online]. Available: http://agilemodeling.com/artifacts/classDiagram.htm. [Accessed: 08- Dec- 2018].

5.1.4 Flowchart

"What is a Flowchart", Lucidchart, 2017. [Online]. Available: https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial#top-info. [Accessed: 08- Dec- 2018].

5.1.5 Entity-Relationship Diagram

“What is an Entity Relationship Diagram,” Lucidchart, 18-Oct-2018. [Online]. Available: https://www.lucidchart.com/pages/er-diagrams. [Accessed: 15-Dec-2018].

5.1.6 Data Flow Diagram

“What is a Data Flow Diagram,” Lucidchart, 07-Sept-2018. [Online]. Available: https://www.lucidchart.com/pages/data-flow-diagram. [Accessed: 14-Dec-2018].

5.2 References

"Third normal form", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Third_normal_form#Normalization_beyond_3NF. [Accessed: 08-Dec- 2018].