Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4316: Senior Design I
Fall 2021

UTA STeam

STEAM
GROCO

Patrick Faulkner
Hozefa Tankiwala
Andrew Hands
Kiran Karki
Uyen Do

REVISION HISTORY

Revision	Date	Author(s)	Description
0.1	10.29.2021	PF	document creation
0.2	11.05.2021	PK, HT, AH, KK, UD	first draft

Contents

List of Figures

List of Tables

1 Introduction

Groco is a grocery shopping web application that is accessible on PC, smartphones, and tablets. Users will be able to search for grocery items and based on the user's brand preference and location, the system will suggest the optimal grocery items. The system also allows users to search for recipes, add their recipes and meal plans into their shopping list to perform the optimization and navigation route.

The purpose of this product is to help users with grocery shopping. The key requirements are:

- searching for the item's availability in nearby stores
- compare item prices and stores distances to find the optimal grocery trip
- providing the convenience in planning meals
- providing optimal sequence of stores

The key requirement of this product is to optimize the grocery items based on users' preferences. The user's preferences include brand, number of stores, and max traveling distance. The system will choose the optimitmal item by prices and distance.

The scope of the product covers grocery stores that have their products and store information online and allow third party access to request or collect data through their public API or web-scraping within the United States. The product prototype should be completed within the budget of 800 dollars and by April 20, 2022.

The main assumptions in this project are that users have internet access, the items' prices and store information collected online are accurate. Groco is not responsible for the price changes or stock-out at the stores. It is the user's responsibility to make sure that a price and discount offer is present at the stores since they are subject to change without notice.

The product contains no obscene material; therefore it is suitable for general grocery shoppers and is made available publicly and free of charge for all users.

The final product will be tested and approved by the client. The development team is free to decide which developing tools and programming languages to be used.

2 System Overview

Groco consists of four main layers: the front-end (the client), the back-end (the server), the database, and the data collector. To create an account and store a user's information, the font end layer will take inputs from the users, send them to the back-end layer to validate, and finally store them in the database. Similarly, the front-end layer will send a request to the back-end and the back-end can retrieve the data from the database then return the appropriate data to display on the front-end. To search for the items, the front-end will take inputs from the users, send them to the back-end layer to process, the back-end will request data from the data collector, using the collected data the back-end complete the request and return the result to display on the front-end.

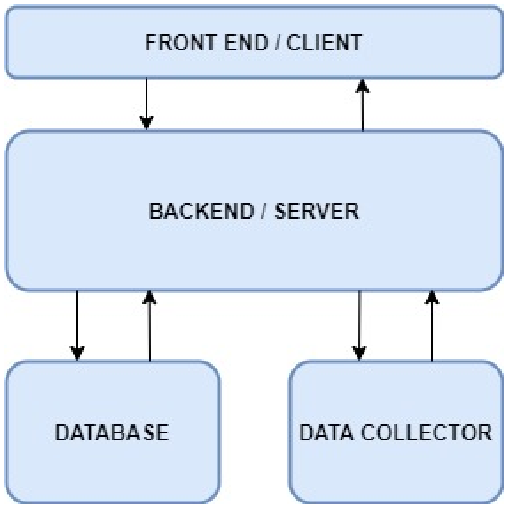

Figure 1: A simple architectural layer diagram

2.1 Front-end Description

The front-end layer includes all software and hardware that is part of a product interface. The hardware includes PC, mobile phones, and tablets. The software is the code that is executed on the client-side (typically HTML, CSS, and JavaScript) that runs in the user's browser to create the user interface. Users interact directly with different components of the front-end, including user-entered data, buttons, links, and other features. The front-end is designed to be accessible, pleasant, and easy to use.

2.2 Server Description

The back-end layer is the code that runs on the server. The back-end receives requests from the front-end (client) and contains the logic of the application to process the request and return the appropriate data to the client. The back-end can directly interact with the database and data collector to retrieve the necessary data to fulfill the request.

2.3 Database Description

The database layer stores and retrieves data in table format. Multiple tables are corresponding to specific functionalities.

2.4 Data Collector Description

The data collector layer is responsible for retrieving data from multiple sources. There is one subsystem in the data collector layer, the API Manager. The data collector retrieves data from various stores through API and returns it to the back-end layer.

3 Subsystem Definitions & Data Flow

Figure 2 is the data flow diagram for Groco. The diagram shows the Graphical User Interface, Server/Backend, Database, and Data Collector along with their subsystems and the data flow between them.

Figure 2: Groco data flow diagram

4 X Layer Subsystems

4.1 Login Subsystem

The Login Subsystem will be the first subsystem that users will interact with when first entering the application. This subsystem will allow users to login with their unique username or email, and their password. Once the credentials are confirmed to be correct, the system will redirect the user.

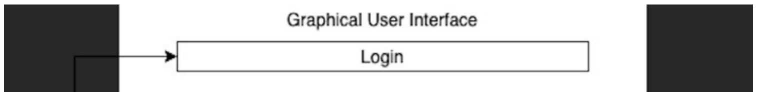

Figure 3: Login Subsystem

4.1.1 Assumptions

There will be the assumption that any user who attempts to login, will have already registered and received a unique username and set a password.

4.1.2 Responsibilities

The primary responsibility of the Login Subsystem will be to provide an interface for users to sign into the application. Upon receiving a users unique username or email and password, the Login Subsystem will relay these inputs to the Query Manager subsystem to be processed. The subsystem will then wait to receive confirmation. If the user entered credentials are incorrect, then the Login Subsystem will give an error message and prompt the user to re-enter their login information. If the credentials are correct, then the Login Subsystem will redirect the user to the home page.

4.1.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 2: Subsystem interfaces

ID	Description	Inputs	Outputs
#1	User Email or Unique Username and Password	Email or Username Password	Page Redirect or Error message

4.2 Register Subsystem

The Registration Subsystem will allows new users to sign up and create their account with a unique email. Users should only need to use this subsystem once.

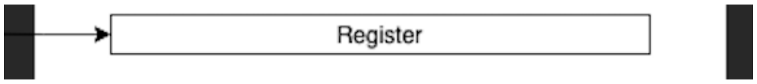

Figure 4: Register Subsystem

4.2.1 Assumptions

There will be the assumption that all users who register have an email that they can access.

4.2.2 Responsibilities

The primary responsibility of the Register Subsystem is to provide an interface for users to sign up to use the application. Users will enter their email address, unique username, and password. Once the Register Subsystem receives these inputs, it will relay them to Query Management subsystem. If the user enters an email or username that is already being used, the Register Subsystem will display an error message and prompt the user for a new email or username. If the user's password does not match the certain requirements, then the system will display an error message and prompt the user to enter a new password. If the user gives all valid inputs, the Register Subsystem will give a confirmation and redirect the user to the login page.

4.2.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 3: Subsystem interfaces

ID	Description	Inputs	Outputs
#1	User Email, Unique Username, and Password	Email or Username Password	Page Redirect or Error message

4.3 Shopping List Subsystem

The Shopping List Subsystem will provide an interface for users to search for their desired grocery item. This will also show users all the items that are currently on their shopping list.

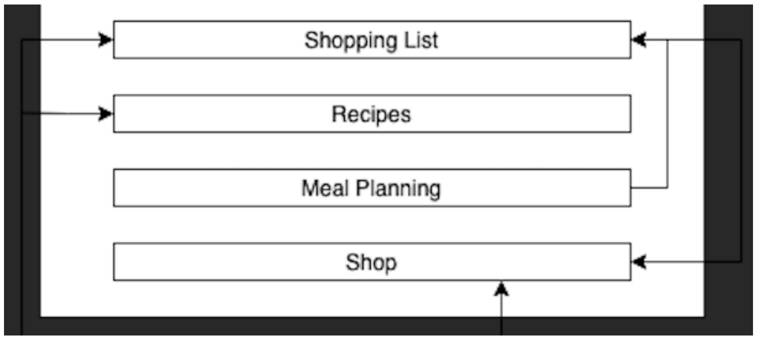

Figure 5: Shopping List Subsystem

4.3.1 Assumptions

There are no assumptions for this subsystem.

4.3.2 Responsibilities

The Shopping List Subsystem will have two responsibilities. First it will communicate with the Query Manager subsystem to obtain the users current grocery list, it will then display this list for the user to see. The Shopping List Subsystem will also allow user to search for item to add to their list.

Users will be able to type in the name of a desired grocery item. The system will send this input to the Query Manager to retrieve the results. This subsystem will then display all results. The user can the select the item they want. The Shopping List subsystem will send this selection to the Query Manager to be added to the users Shopping List.

Finally, if a user is ready to shop for all of their grocery items, they will select the option to shop and control will be transferred to the Shop Subsystem.

4.3.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 4: Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Display Shopping List	-	Shopping List from Query Manager
#2	Grocery Item Search	Grocery Item from User	User input to Query Manager
#3	Search Results	Search Results from Query Manager	Display Search Results to User
#4	User Select Results	User Selected Item	Selection to Query Manager

4.4 Recipes Subsystem

The Recipes Subsystem will allow users to view recipes, add a recipe's ingredients to their shopping list, and add their own recipes.

Figure 6: Recipes Subsystem

4.4.1 Assumptions

User will be responsible for ensuring that recipes do not conflict with their dietary restrictions. It will also be assumed that users will not add dangerous recipes.

4.4.2 Responsibilities

The first responsibility of the Recipes Subsystem will be to provide an interface for users to view and search recipes. Upon entering the Recipes subsystem, users will be shown random recipes that were randomly pulled from the Query Manager. Users will then have the ability to search for recipes by using keywords or searching the name of known recipe. Upon receiving search text from the user, the system will relay that to the Query Manager to retrieve then display all results matching the search.

After viewing the recipes, users will be able to select individual recipes. This will allow them to see all the ingredients and instructions. If the user likes the recipe, they will be able to add all the ingredients directly to their shopping list or meal plan.

The Recipes Subsystem will also enable user to share their own recipes. If a user decides to do this, users will first select all the ingredients and the amounts needed. After all ingredients are selected, the user will then be able to enter step-by-step instructions that describe how to create the recipe. Once this is complete, the system will send the user provided information to the Query Manager to be added to the recipe database.

4.4.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 5: Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Recipe Search	User Search Text	User Search Text to Query Manager
#2	Recipe Search Results	Results of Search from Query Manager	Search Results to User
#3	Select Recipe to View	User Recipe Selection	Recipe Selection to Query Manager
#4	Display Recipe to View	Recipe Ingredients and Instruction from Query Manager	Display Recipe Ingredients and Instruction
#5	Add Recipe to Shopping List	User Choice to Add Recipe to Shopping List	All Ingredients to Query Manager
#6	Add Recipe to Meal Plan	User Choice to Add Recipe to Meal Plan	Recipe ID to Query Manager
#7	Add Own Recipe	All Ingredients and Instructions	User Given Ingredients and Instructions to Query Manager

4.5 Meal Plan Subsystem

The Meal Plan Subsystem will allow users to view and create a collection of recipes called Meal Plans. Users will be able to add recipes (via the Recipes Subsystem), delete recipes, and add all ingredients to the shopping list.

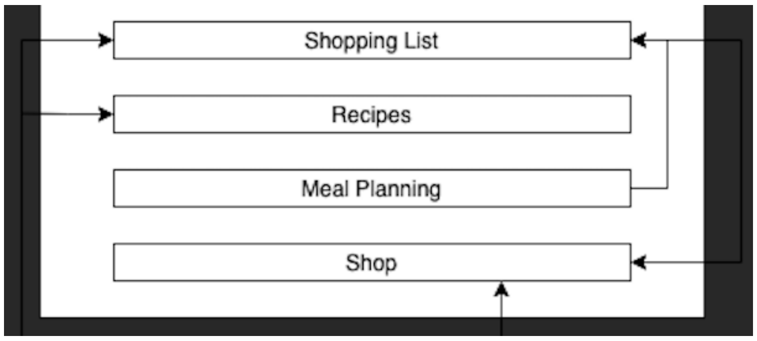

Figure 7: Meal Plan Subsystem

4.5.1 Assumptions

Users will not be able to add ingredients that are not in a recipe to a Meal Plan.

4.5.2 Responsibilities

The Meal Plan Subsystem will allow users to create new Meal Plans that will be empty. Once a user chooses to do this, the Query Manager will be responsible for the creation of the Meal Plan in the database. A user will be able to add recipes to the their Meal Plan via the Recipes System. Once there is more than one recipe in a Meal Plan, the system will allow the user to view each Meal Plan and select ones to view individually. If a user selects a Meal Plan to view, the Meal Plan subsystem will retrieve all the its recipes through the Query Manager and display them to the user.

If a user decides they would like to use a Meal Plan, they will be able to add it to their shopping list. This will be done by adding getting all the ingredients from each recipe via the Query Manager and then adding them to the Shopping List.

4.5.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 6: Subsystem interfaces

ID	Description	Inputs	Outputs
#1	View Meal Plans	-	All of the User's Meal Plans
#2	Create Meal Plan	User Selection to Create a Meal Plan	Creation Command to Query Manager
#3	Select Single Meal Plan	User Meal Plan Selection	Meal Plan Selection to Query Manager
#4	View Single Meal Plan	Recipes in Meal Plan from Query Manager	Display All Recipes to User

4.6 Shop Subsystem

The Shop Subsystem will act as an interface to present users with the optimal grocery trip.

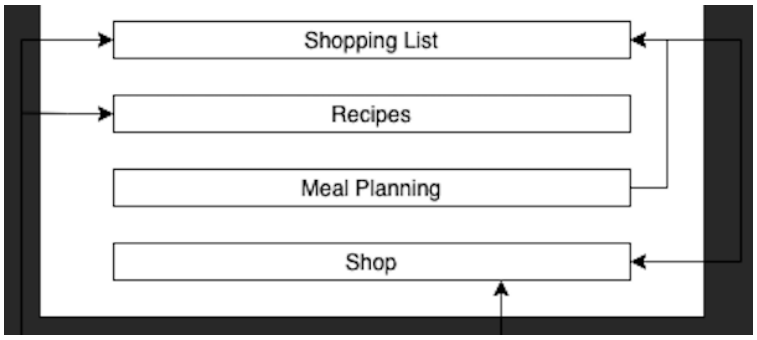

Figure 8: Shop Subsystem

4.6.1 Assumptions

There are no assumptions for this subsystem.

4.6.2 Responsibilities

The system will access the Shop Subsystem via the Shopping List Subsystem when a user decides they are ready to shop for the grocery items on their list. Once the subsystem receives the grocery items from the list, it will send that information to the Shopping Manager Subsystem along with the user defined preferences. All calculation will be done on the Server Layer. Once the Shop Subsystem receives the results, it will display the stores the user should visit, the items they will get at each stores, the sequence the stores should be visited, and a link to a map application to navigate their route.

4.6.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 7: Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Initiate Shopping	Shopping List Items User Preferences	Shopping List Items and User Preferences to the Shopping Manager
#2	Display Results	Optimal Shopping Trip From Shopping Manager	Display Optimal Shopping Trip to User

5 Y Layer Subsystems

In this section, the layer is described in some detail in terms of its specific subsystems. Describe each of the layers and its subsystems in a separate chapter/major subsection of this document. The content of each subsystem description should be similar. Include in this section any special considerations and/or trade-offs considered for the approach you have chosen.

5.1 Subsystem 1

This section should be a general description of a particular subsystem for the given layer. For most subsystems, an extract of the architectural block diagram with data flows is useful. This should consist of the subsystem being described and those subsystems with which it communicates.

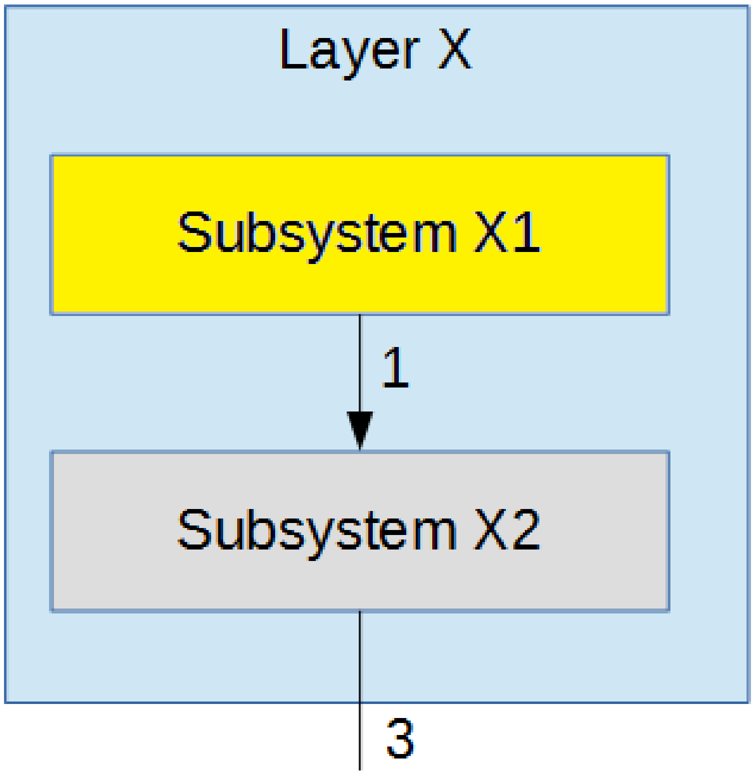

Figure 9: Example subsystem description diagram

5.1.1 Assumptions

Any assumptions made in the definition of the subsystem should be listed and described. Pay particular attention to assumptions concerning interfaces and interactions with other layers.

5.1.2 Responsibilities

Each of the responsibilities/features/functions/services of the subsystem as identified in the architectural summary must be expanded to more detailed responsibilities. These responsibilities form the basis for the identification of the finer-grained responsibilities of the layer's internal subsystems. Clearly describe what each subsystem does.

5.1.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here. Create a table with an entry for each labelled interface that connects to this subsystem. For each entry, describe any incoming and outgoing data elements will pass through this interface.

Table 8: Subsystem interfaces

ID	Description	Inputs	Outputs
#xx	Description of the interface/bus	input 1 input 2	output 1
#xx	Description of the interface/bus	N/A	output 1

5.2 Subsystem 2

Repeat for each subsystem

5.3 Subsystem 3

Repeat for each subsystem

6 Z Layer Subsystems

In this section, the layer is described in some detail in terms of its specific subsystems. Describe each of the layers and its subsystems in a separate chapter/major subsection of this document. The content of each subsystem description should be similar. Include in this section any special considerations and/or trade-offs considered for the approach you have chosen.

6.1 Subsystem 1

This section should be a general description of a particular subsystem for the given layer. For most subsystems, an extract of the architectural block diagram with data flows is useful. This should consist of the subsystem being described and those subsystems with which it communicates.

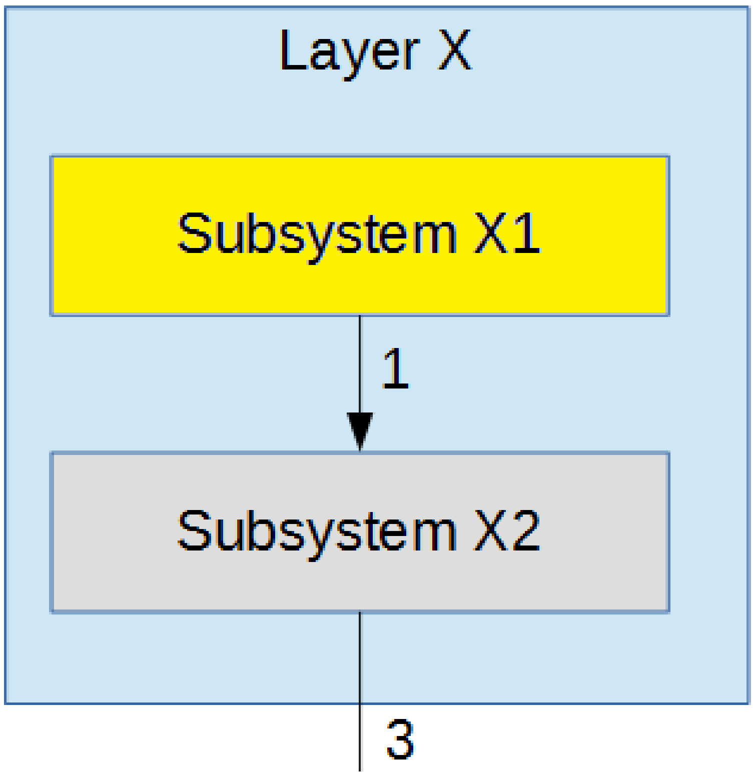

Figure 10: Example subsystem description diagram

6.1.1 Assumptions

Any assumptions made in the definition of the subsystem should be listed and described. Pay particular attention to assumptions concerning interfaces and interactions with other layers.

6.1.2 Responsibilities

Each of the responsibilities/features/functions/services of the subsystem as identified in the architectural summary must be expanded to more detailed responsibilities. These responsibilities form the basis for the identification of the finer-grained responsibilities of the layer's internal subsystems. Clearly describe what each subsystem does.

6.1.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here. Create a table with an entry for each labelled interface that connects to this subsystem. For each entry, describe any incoming and outgoing data elements will pass through this interface.

Table 9: Subsystem interfaces

ID	Description	Inputs	Outputs
#xx	Description of the interface/bus	input 1 input 2	output 1
#xx	Description of the interface/bus	N/A	output 1

6.2 Subsystem 2

Repeat for each subsystem

6.3 Subsystem 3

Repeat for each subsystem