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

Detailed Design Specification
CSE 4317: Senior Design II
Spring 2022

UTA STeam
Groco

Patrick Faulkner
Andrew Hands
Hozefa Tankiwala
Karki Kiran
Uyen Do

Revision History

Revision	Date	Author(s)	Description
0.1	1.30.2022	AH	document creation

Contents

List of Figures

List of Tables

1 Introduction

Groco aims to streamline the way we find recipes, make shopping lists, and find the best locations to buy ingredients. We accomplish this buy having a centralized place for people to add, search, and view recipes. Whenver a given user decides they like a recipe, the recipe can be added to their shopping list by the click of a button. Whenever a user decides to to go shopping, by the click of a button they can be given a list of stores and items to shop at which is optimized by their shopping preferences. Additionally, we add customization in the form of meal plans, which users can create and use as common collections of recipes that they can add to their shopping list when needed.

For additional information, consider taking a look at the architectual design and system requirement specifications, which can be found at https://tinyurl.com/2p8psxc8 and https://tinyurl.com/2p8snvxf, respectively.

2 System Overview

This section should reintroduce the full data flow diagram from the architectural specification, and discuss at a high level the purpose of each layer. You do not need to include a subsection for each layer, a 1 - 2 paragraph recap is sufficient.

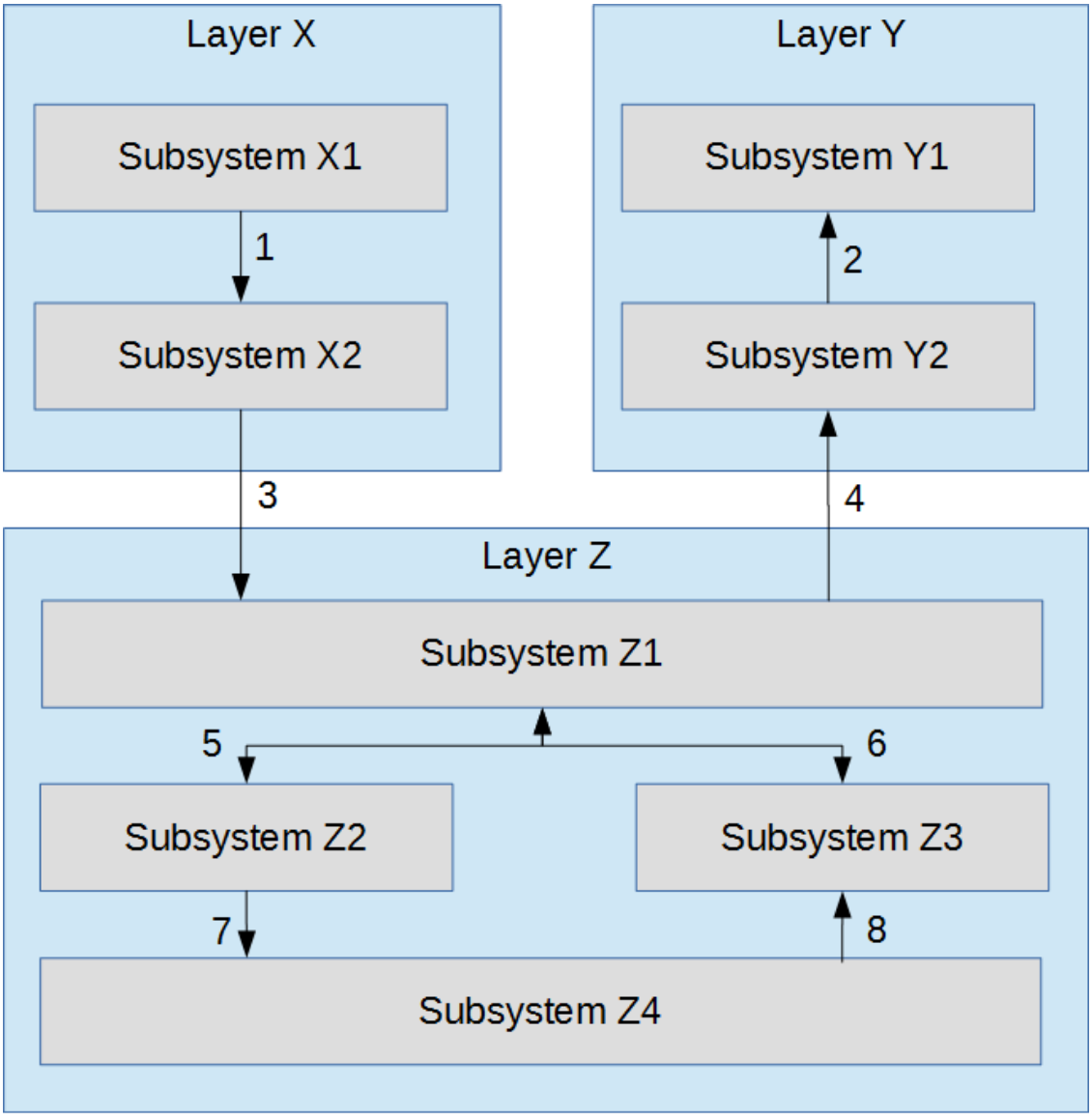

Figure 1: System architecture

3 Front-End Layer Subsystems

3.1 Layer Software Dependencies

The entire front-end will depend on Bootstrap and the React library. The front-end will also depend on a consistent connection to the Query Manager on the Server layer.

3.2 Login Subsystem

The Login Subsystem will provide an interface for users to login into the application. Users will be able to log in with their email and password credentials or their Google accounts.

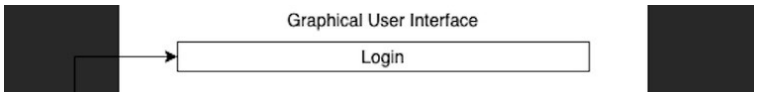

Figure 2: Login Subsystem diagram

3.2.1 Subsystem Software Dependencies

For a user to log in with their Google accounts, the Login Subsystem will depend on the GoogleLogin library.

3.2.2 Subsystem Programming Languages

The Login Subsystem will be developed using, React.js, HTML, and CSS.

3.2.3 Subsystem Data Processing

If the user logs in using their email and password, they will enter their credentials in the respective places and click the login button. Upon the login button being pressed, the Login Subsystem will send the user's credentials to the Query Manager in the backend layer. If the Login Subsystem receives a success signal, the user will be redirected to the Home Page. If the Login Subsystem receives a failure signal, a failure message will be displayed and users will be prompted to re-enter their credentials.

If the user decides to log in with their Google account, they will select the Google login button. The Login Subsystem will then call on the GoogleLogin API which users will log in with. If the login is successful, the Login Subsystem will redirect the user to the Home Page. If the login fails, the user will be given an error message and prompted to re-enter their credentials.

3.3 Register Subsystem

The Register Subsystem will provide an interface for users to register for an account.

Figure 3: Register Subsystem diagram

3.3.1 Subsystem Software Dependencies

The Register Subsystem does not have any additional dependencies.

3.3.2 Subsystem Programming Languages

The Register Subsystem will be developed using, React.js, HTML, and CSS.

3.3.3 Subsystem Data Processing

After entering their password and unique email and username, the user will press the Register button. Upon the Register button being pressed, the Register Subsystem will capture the user-entered information and send it to the Query Manager. If the Query Manager sends a success signal, the Register Subsystem will redirect the user to the Home Page. If the Query Manager sends a failure signal, the Register Subsystem will display an error message and prompt the user to re-enter their information.

3.4 Shopping List Subsystem

The Shopping List Subsystem will provide an interface for users to view all the items on their current shopping list. It will also allow users to add and remove items.

Figure 4: Shopping List Subsystem diagram

3.4.1 Subsystem Software Dependencies

The Shopping List Subsystem has no additional dependencies.

3.4.2 Subsystem Programming Languages

The Shopping List Subsystem will be developed using, React.js, HTML, and CSS.

3.4.3 Subsystem Data Processing

The Shopping List Subsystem will retrieve the current user's shopping list from the Query Manager and display all relevant information.

3.5 Recipes Subsystem

The Recipes Subsystem will provide an interface for users to view and shop for all recipes in the system's database. It will also allow users to create and edit their recipes.

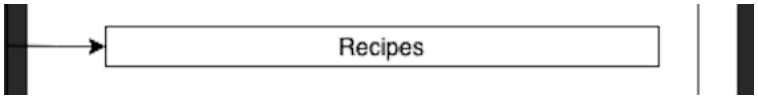

Figure 5: Recipes Subsystem diagram

3.5.1 Subsystem Software Dependencies

The Recipes Subsystem has no additional dependencies.

3.5.2 Subsystem Programming Languages

The Recipes Subsystem will be developed using, React.js, HTML, and CSS.

3.5.3 Subsystem Data Processing

The Recipes Subsystem will display all recipes retrieved by the Query Manager.

3.6 Meal Plan Subsystem

The Meal Plan Subsystem will provide an interface for users to view and edit their Meal Plans.

3.6.1 Subsystem Software Dependencies

The Meal Plan Subsystem has no additional dependencies.

3.6.2 Subsystem Programming Languages

The Meal Plan Subsystem will be developed using, React.js, HTML, and CSS.

Figure 6: Meal Plan Subsystem diagram

3.6.3 Subsystem Data Processing

The Meal Plan Subsystem will display the current user's Meal Plans that are retrieved by the Query Manager.

3.7 Shop Subsystem

The Shop Subsystem will provide an interface for users to view the results of the shopping trip being planned.

Figure 7: Shop Subsystem diagram

3.7.1 Subsystem Software Dependencies

The Shop Subsystem depends on a consistent connection to the Shop Manager on the Server layer.

3.7.2 Subsystem Programming Languages

The Shop Subsystem will be developed using, React.js, HTML, and CSS.

3.7.3 Subsystem Data Processing

The Shop Subsystem will display all information that is calculated by the Shopping Manager.

4 Y Layer Subsystems

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be ommitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project.

4.1 Layer Hardware

A description of any involved hardware components for the layer. For example, if each subsystem is a software process running on an embedded computer, discuss the specifics of that device here. Do not list a hardware component that only exists at the subsystem level (include it in the following sections).

4.2 Layer Operating System

A description of any operating systems required by the layer.

4.3 Layer Software Dependencies

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

4.4 Subsystem 1

Descibe at a high level the purpose and basic design of this subsystem. Is it a piece of hardware, a class, a web service, or something else? Note that each of the subsystem items below are meant to be specific to that subystem and not a repeat of anything discussed above for the overall layer.

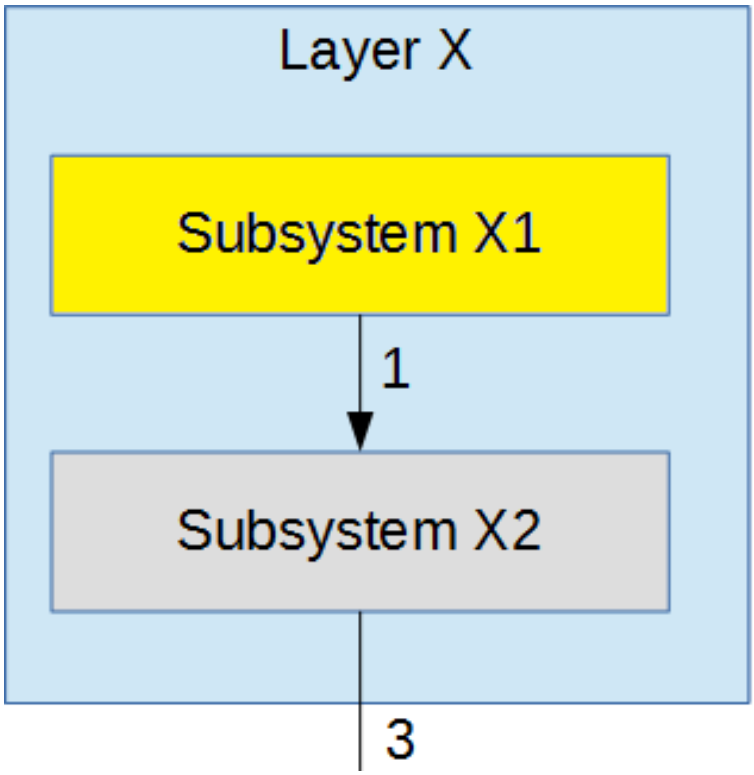

Figure 8: Example subsystem description diagram

4.4.1 Subsystem Hardware

A description of any involved hardware components for the subsystem.

4.4.2 Subsystem Operating System

A description of any operating systems required by the subsystem.

4.4.3 Subsystem Software Dependencies

A description of any software dependencies (libraries, frameworks, design software for mechanical parts or circuits, etc) required by the subsystem.

4.4.4 Subsystem Programming Languages

A description of any programming languages used by the subsystem.

4.4.5 Subsystem Data Structures

A description of any classes or other data structures that are worth discussing for the subsystem. For example, data being transmitted from a microcontroller to a PC via USB should be first be assembled into packets. What is the structure of the packets?

4.4.6 Subsystem Data Processing

A description of any algorithms or processing strategies that are worth discussing for the subsystem. If you are implementing a well-known algorithm, list it. If it is something unique to this project, discuss it in greater detail.

5 Z Layer Subsystems

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be ommitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project.

5.1 Layer Hardware

A description of any involved hardware components for the layer. For example, if each subsystem is a software process running on an embedded computer, discuss the specifics of that device here. Do not list a hardware component that only exists at the subsystem level (include it in the following sections).

5.2 Layer Operating System

A description of any operating systems required by the layer.

5.3 Layer Software Dependencies

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

5.4 Subsystem 1

Descibe at a high level the purpose and basic design of this subsystem. Is it a piece of hardware, a class, a web service, or something else? Note that each of the subsystem items below are meant to be specific to that subystem and not a repeat of anything discussed above for the overall layer.

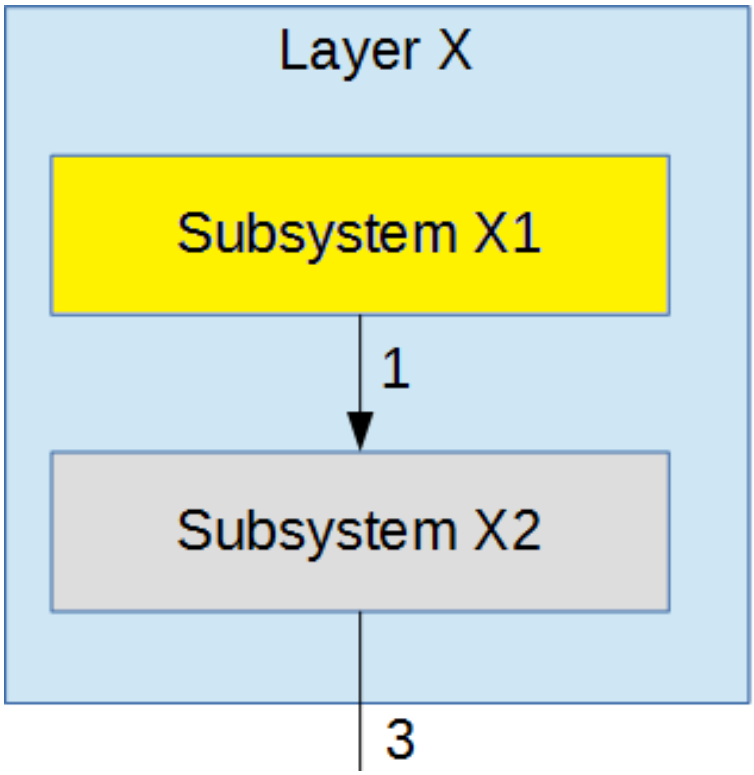

Figure 9: Example subsystem description diagram

5.4.1 Subsystem Hardware

A description of any involved hardware components for the subsystem.

5.4.2 Subsystem Operating System

A description of any operating systems required by the subsystem.

5.4.3 Subsystem Software Dependencies

A description of any software dependencies (libraries, frameworks, design software for mechanical parts or circuits, etc) required by the subsystem.

5.4.4 Subsystem Programming Languages

A description of any programming languages used by the subsystem.

5.4.5 Subsystem Data Structures

A description of any classes or other data structures that are worth discussing for the subsystem. For example, data being transmitted from a microcontroller to a PC via USB should be first be assembled into packets. What is the structure of the packets?

5.4.6 Subsystem Data Processing

A description of any algorithms or processing strategies that are worth discussing for the subsystem. If you are implementing a well-known algorithm, list it. If it is something unique to this project, discuss it in greater detail.

6 Data Controller Layer Subsystem

The data controller layer is responsible for getting data about grocery items including data about prices, stock and store location from different stores then transfer the data to the Shopping Manager subsystem in the Server/Back-end layer. There is one subsystem in the data controller layer, the API manager. Since many stores do not allow web scraping, the developer team decided to use public APIs from stores to gather grocery data.

6.1 Layer Operating System

Data collector layer get the data from the store public web APIs.

6.2 Layer Software Dependencies

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

6.3 API Manager

The API manager is a core component of the data collector. The API manager provides one central point for data collection through web APIs across different stores.

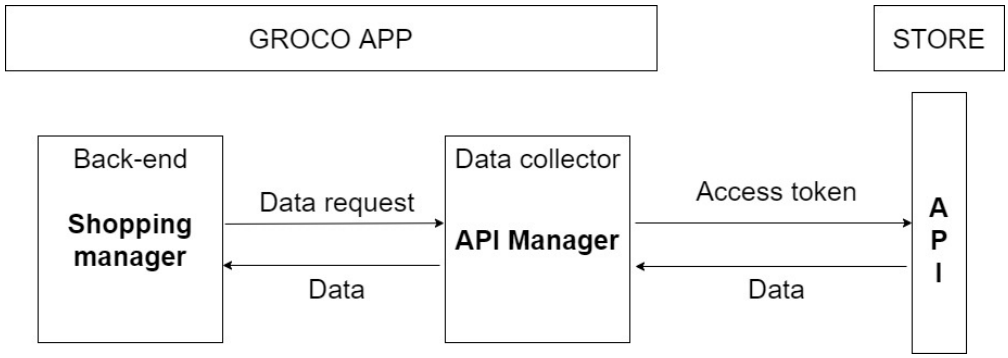

Figure 10: Data Controller Subsystem

6.3.1 Subsystem Operating System

Data collector layer get the data from the store public web APIs.

6.3.2 Subsystem Software Dependencies

Since many stores do not allow web scraping and do not have public APIs, the development team decide to use Kroger's public web APIs to gether grocery data.

6.3.3 Subsystem Programming Languages

The API manager is written in JavaScript

6.3.4 Subsystem Data Structures

6.3.5 Subsystem Data Processing

The API Manager is responsible for storing access tokens to get data from different APIs. The retrieved data will be sent to the Shopping Manager subsystem in the Back-end layer to fulfil the request. The API Manager is also responsible for refreshing the access tokens when they are expired.

7 APPENDIX A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.