Cheap Staples
Software Requirements Specification

Jan Bermudez(jb), Garret Bunkers(gb), Dylan Hopper(dh), Tim Nadeau(tn), Patrick Rodriguez (pr)

Table of Contents

1. SRS Revision History

Date	Name	Revision Description
2/25/2023	Patrick Rodriguez	Finished document formatting, section 3.1 (external interfaces) and section 3.2 (functions)
2/26/2023	Patrick Rodriguez	Finished section 3.3 (usability requirements)
2/27/2023	Garrett Bunkers	Finished 2.6 (Operational Scenarios)
2/27/2023	Dylan Hopper	Added Sections: Current System, Justification for a New One, Operational Features
2/28/2023	Jan Bermudez	Added Sections: Performance Requirements, Software System Attributes
2/28/2023	Garrett Bunkers	Finished 2.4(User Classes) 2.5(Modes of Operation)
3/12/2023	Garrett Bunkers	Revised 2.4(User Classes) and 2.6 (Operational Scenarios)

2. The Concept of Operations

2.1. Current System or Situation

The current system will allow lower to middle class people to keep track of the prices of staple foods. Currently, most people that care about prices of groceries will most likely either memorize the prices as they buy them, use Google Shopping, or look for the prices on the stores website. Our system will make a simple and easy to use program that will allow the user to see and track the prices of staple foods. This will allow them to see which stores have the cheaper staple foods. They can then use this information to make a decision as to which store they should shop at to save money.

2.2. Justification for a New System

There is a system that is quite similar which is Google Shopping. The key difference with google shopping and our system is that you have to specifically search for specific items with Google Shopping, as with our system, it will simply display all the staple foods all in one list.

2.3. Operational Features of the Proposed System

Our system, when run, will show a display of all the staple foods. The user will then select the foods that they are interested in seeing the price of then pressing the view selected button, or simply press the view all button. It will show which stores have the cheaper prices for each staple food. This will help inform users of the price comparisons between stores so they can then choose which store to shop at to save money. There will be a button that will allow the user to update the prices to what they currently are. This allows the program to be fast while also giving the user the ability to make sure the prices are correct if they haven't been updated for a considerable amount of time.

2.4. User Classes

One type of user that will be using this software are people in Eugene with a low-income or ones that live in a low-income household. This software will help these users find the necessary staple foods at the cheapest prices at stores in Eugene. This user class would contain students, disenfranchised people, and any resident of Euegene.

Another type of user is a user operating the software that wants to update the food prices. This user wants to update the food prices in the database for the software to maintain relevant prices. The software will provide a method to re-Scraping the data from stores' websites.

2.5. Modes of Operation

When using the software, a user can choose to select specific foods by clicking the boxes beside the food and then clicking the "View Selected" button. For someone who wants to look at all the prices there is a "View All" button that will show all the food prices and their corresponding stores.

For the user that wants to update the food prices that are inaccurate, the software will be operated to update the food prices by pressing the update button. This will run the ScrapeData software and update the MySQL database.

2.6. Operational Scenarios

Scenario 1. Disenfranchised person needs cheap staple foods to make a meal

This scenario describes how a user of the CheapStaples software would track certain food prices and find the cheapest staple food at grocery stores in Eugene by using the Cheap Staples software.

Meet the Actor: Greg is planning his meals for the week. He knows he will need some staple foods for all the meals this week and wants to be able to keep track of those staple food prices throughout the week, so he can get the cheapest prices possible. Greg wants to use the Cheap Staples software to keep track of his desired foods and find the stores with the cheapest price.

Steps to Complete the Task:

1. Greg creates a list of foods he wants to keep track for the week
2. Greg then gets on his computer with internet access and runs the *CheapStaples* program
3. Greg checks the boxes beside the foods he has on his list.
4. After pressing the "View" button, Greg gets to see the cheapest prices of the selected food at Safeway and Target.
5. Greg writes down the prices for that day
6. Repeat steps 2-5 every day throughout the week
7. Looks at his list of prices that he's written down to determine which store consistently has the cheapest prices

Postconditions:

Greg can now write down a list of the cheapest foods and their location. He is now confident he can afford these foods to purchase for his meals this week.

Extensions:

4a. Greg wants to look at all the prices of the foods:

4a1:Press the "View All" button

Scenario 2. A user wants to update the prices on the CheapStaples software.

This scenario describes how a person would update the food prices on the software.

Meet the Actor: Greg just used the CheapStaples software to find the cheapest foods in Eugene. However, one of the prices didn't match the price at the store. Greg wants to use the software to update the price of the food at the specific store.

Steps to Complete the Task:
1. Greg starts the software and clicks the “Update” button.
Postconditions:
After pressing update, Greg now can select his foods and the “View” button to see the updated prices. Greg is now pleased that the software is up-to-date with current prices.

3. Specific Requirements

3.1. External Interfaces

Input:

1. **Name of item:** Ingredient Chooser
2. **Description of purpose:** The user will be provided a list of staple grocery ingredients to choose from.
3. **Source of input:** The user will be able to choose necessary ingredients via checkboxes provided by the data processing module.
4. **Valid ranges of inputs:** Valid inputs are predetermined by the available checkboxes granted to the users.
5. **Units of Measure:** The data collected as inputs will be largely nominal and therefore will not be measured but stored.
6. **Data Format:** The data format will consist of strings.

Output:

1. **Name of item:** Location and price display
2. **Description of purpose:** The user will be provided with the cheapest prices and the location of chosen ingredients found in Eugene, Oregon.
3. **Destination of outputs:** Another window will open and display the prices and corresponding grocery store locations of the ingredients chosen by the user.
4. **Valid ranges of outputs:** Locations whose website displays prices for the appropriate chosen ingredient will be valid and displayed,
5. **Units of Measure:** The number of valid ingredients found at all locations.
6. **Data Format:** For each location a given ingredient is available at, there will be a string representing the grocery store mapped to a float value which represents its price. Each price-location entry will be stacked vertically sorted by price. Cheapest prices will be colored green and will get progressively more red as prices increase among the other locations. If multiple ingredients are chosen, then entries will be sorted by ingredient.

3.2. Functions

3.2.1. Update Prices Function

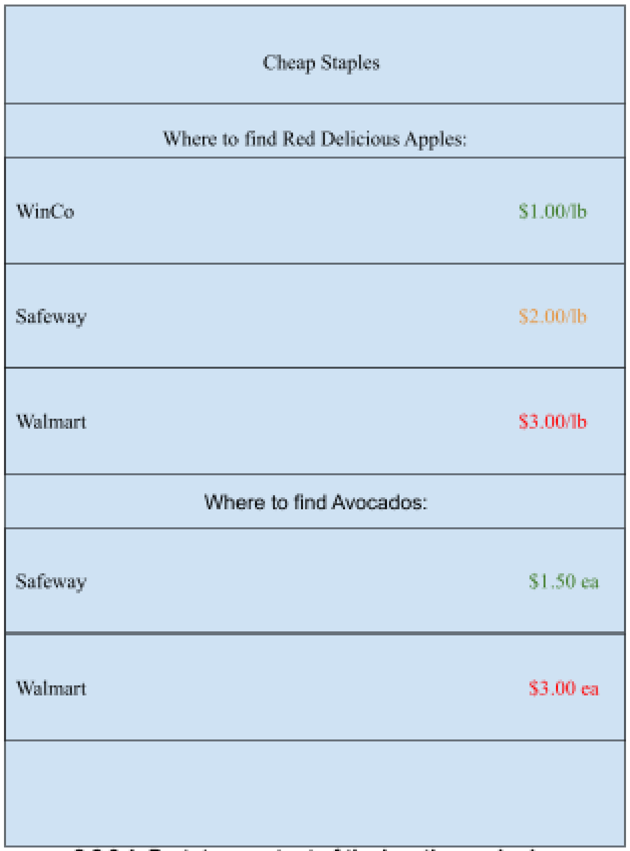
The user will be provided a button responsible for updating the prices displayed on the user interface. This function will call individual grocery store scrapers whose return data will be used to repopulate the data entry prices. No parameters will be required from the user.

3.2.2. Display Ingredients Function

The user can either choose to select all staple foods or select from a list of staples to compare prices from. The selected foods will be used as parameters to gather appropriate data entries to format and display to the user.

3.2.3. Display Prices and Locations Function

For each ingredient specified by the user, each location and corresponding price in which that ingredient is found will be displayed adjacent to each other. Cheapest prices will be displayed green with other prices becoming increasingly red to signify that better options are available. Take a look at the following prototype display to better understand the output:



Cheap Staples

Where to find Red Delicious Apples:

WinCo	$1.00/lb
Safeway	$2.00/lb
Walmart	$3.00/lb

Where to find Avocados:

Safeway	$1.50 ea
Walmart	$3.00 ea

3.2.3.1. Prototype output of the location and price display external interface

3.3. Usability Requirements

The usability requirements for the system are as follows:

- The user is able to update the store prices through the click of a button responsible for performing web scraping from designated grocery store websites. At no time should the scraping function timeout when requesting HTML code.
- The user will be provided an interface of ingredients to choose from through checkboxes. If there are too many to fit in one window, a scrollbar will be displayed to easily navigate through the different options.
- Each ingredient specified will contain a header which below will contain the locations that ingredient is found at and its corresponding price.Cheapest prices are labeled in color green and other prices will become increasingly red to signify that better options are available.

3.4. Performance Requirements

Static numerical requirements:

- The Cheap Staples system must be able to handle filter options that the user checks off on the user interface.
- The system should be able to scrape data from grocery website and convert them to “Table” objects
- The system must display on the current window the cheapest prices of staples with their respective color and their locations.
- The system should be able to display a few select ingredients or all of them at the same time.

Dynamic numerical requirements:

- The Cheap Staples system should process user input and generate a graphical table with locations and prices within 3 seconds after the user clicks the search button.
- Grocery Web scraping and Location-Price-Ingredient database updating should be done processing in no more than 4 minutes.

3.5. Software System Attributes

The required attributes of the software product are:

- Reliability: The Cheap Staples system must be reliable as it is expected to perform consistently under different conditions and use cases. Cheap Staples might not be a system that performs critical tasks but an unreliable system causes inconvenience and frustration for users leading to decreased user satisfaction.
    - In order to increase reliability, the Cheap Staples system will use modularity to help isolate faults to specific modules and prevent them from affecting the entire system. The system will also implement robust error handling checks throughout its development such as the case to check whether the user chose a valid input for ingredients.

- Maintainability: The system must be maintainable as it reduces the cost and time to modify or improve the system. A software system that is maintainable is also easier to adapt, and extend to meet the changing needs and requirements of users. This flexibility is essential in today's rapidly changing technological landscape.
    - Maintainability will be achieved by a comprehensive documentation of the system architecture, design and code. Documentation will be clear and precise so that any developer is able to understand the software system without difficulty.

4. Acknowledgements

This SRS document was based off of the SRS template document provided by professor Hornof at the University of Oregon.