Stage 1: Detailed Project Description

Bennett Wu [bwu40]

Jeremy Lee [jeremyl6]

Paul Jeong [paulj3]

Shreya Singh [shreya15]

Summary

Team TableTurner's project will be a grocery comparison full stack application called GroceryAid. Websites such as Kayak, Expedia, and Honey serve to provide the best price possible for tickets and items across different sites. Grocery shopping can also benefit from this concept by having an application displaying the best prices for common grocery items compared at popular stores. Items can then be organized by stores and saved to a cart so shoppers can easily coordinate their grocery shopping. Grocery product datasets from popular stores like Walmart, Amazon, Costco, and others can be organized to show preferences based on categories, such as prices, items, and stores that are relevant to a grocery shopper's needs.

Description

The application - being a grocery comparison tool - will help solve spending issues in households across the country. When choosing between which stores to visit and what items to purchase at each store, a tool that assesses and compares the prices of target products between stores is invaluable to shoppers looking to maximize their income and minimize their essential spending. By compiling product prices through datasets made available through online sources and comparing common products amongst stores, households will be able to determine the best courses of action when making purchases. In essence, we want to solve the problem of determining where to source what products - enabling users to maximize their time and money while doing so.

Creative Component

One creative component and stretch goal would be to personalize the application by providing an auto grocery shopping list. People have unique and individual dietary and nutritional needs. A recipe dataset can be paired with our current application. Users can browse recipes and once one is chosen, it will add all the ingredients to the user's grocery list with the same details as before, such as the best price from a store.

Firstly, a recipe dataset with ingredients is needed. Preferably, one with ingredient names similar to item names from our store datasets. The challenging component will be having our application recognize these ingredients = items. SQL allows us to search for exact keywords, which will work to a certain extent that might be enough for our needs. That we will have to judge ourselves. We could also utilize the OpenAI api to have more accurate comparisons that would be more ideal. Once ingredient to item can be recognized, the application just has to run its basic function automatically to store items and its details into the list.

Usefulness

This application is designed to aid users in choosing the best or most appropriate stores for the user's needs - the application helps users figure out how to best utilize their time and money, and will be widely applicable to most users. By utilizing the application, users will be able to determine what store they need to visit in order to find everything they need, as well as what items to get from which stores if they want to spend the least money. The primary function of our web application will be the product comparison tool - the web application will include infrastructure and elements that allow users to compare pricing of products across various stores and categories. Users will be able to use this tool to determine the best course of action when purchasing products from the list of stores we have available.

A more complex feature that will be included in the website will be the item category sorter - within food types alone, there are several food groups (such as Dairy, Fruit, Vegetables, etc.) which will be able to be filtered based on the user's input. Enabling the user to filter out categories will allow users to gain a better understanding of the cost of items per category, as well as how each store holds up in terms of pricing and quantity across those categories. The website will also employ a data visualization element, to allow the user to visualize a breakdown of their spending, as well as visualize how the prices compare between stores.

While apps that compare grocery stores exist, their functionality and scope is limited in comparison to what we are trying to achieve - most applications limit price comparisons between stores, are limited in the products that are present within the app, and do not include visualization tools. With a goal of providing our users with a better understanding of where their money is going, our application will provide features that are more relevant to the shopping and saving experience.

Realness

We are using grocery product data from various US grocers.

The Walmart dataset comes from:
https://data.world/xfu022/walmart-grocery-product-dataset
The dataset is in the form of a .csv file, with a cardinality of 15 and degree of 568,534. It contains information about the category, name, brand, price, and more for various grocery products found online at Walmart.

The Amazon dataset is found at:
https://www.kaggle.com/datasets/shmalex/amazon-products
This dataset also comes in a .csv format, with 21 columns and 160543 rows, 13,098 of which are in the Grocery category. It contains information about the category, name, brand, price, and more of various products on Amazon.

For Whole Foods, the dataset comes from:
https://www.kaggle.com/datasets/thedevastator/new-whole-foods-on-sale-product-data-collection
In particular we are planning on using the four wholefoodsorders*.csv files. The fourth file has a cardinality of 11 while the others have a cardinality of 12, the difference being the existence of an index column. Together the dataset has a degree of 12,013. It contains the company, product name, regular price, sale price, prime price, category, and more of various grocery items from the on sale section of the Whole Foods website.

The Costco data comes from:
https://www.kaggle.com/datasets/elvinrustam/grocery-dataset
The dataset comes in a .csv file, with a cardinality of 8 and degree of 1,757. It contains information about the category, price, discounts, rating, name, currency, features, and description of grocery products found on Costco’s online marketplace.

Functionality

Description

Databases for our project:

1) User Information Database:
- Stores user information such as usernames, and passwords.
- Columns: primary_key(INT), username(VARCHAR), password(VARCHAR)

2) User Database
- Stores each user's queries.
- This database will be unique to each user.
- Columns: primary_key(INT), foriegn_key(INT) (to link this database to the user information database), item (VARCHAR), category of item (INT)

3) Store Database
- Stores the information of products for each store
- Current databases: Walmart, Amazon, Costco, WholeFoods
- Columns will be slightly different for each store depending on the information we have
- Columns: primary_key(INT), item (VARCHAR), category of item (INT), item_url (VARCHAR), price_item (FLOAT)

Functionality of website according to mock-up

Page 1 of the website:
This is the login page and the user will be able to enter their username and password to log into the website. The user will also be able to create a new account if this is their first time using the website.

Page 2 of the website:
This is the homepage and there will be a navigation bar on the top that will be displayed on all the pages. The navigation bar has the logo which will bring the user back to the homepage. The Categories button will take the user to the Categories page (5). The Frequently Search button will take the user to the Frequently Searched page (6).
The homepage will have search bars that will allow the user to "search by item" or "search by category". "Search by Item" will allow the user to look for items and obtain the store at which the item is at its cheapest, and also give them the price. "Search by Category" will allow the user

to enter a word like "dairy" or "vegetables", for example, and obtain a table that gives all the items in that category, along with the best store to buy them at, and the price.

Page 3 of the website:
This page will display the results of "Search by Item."

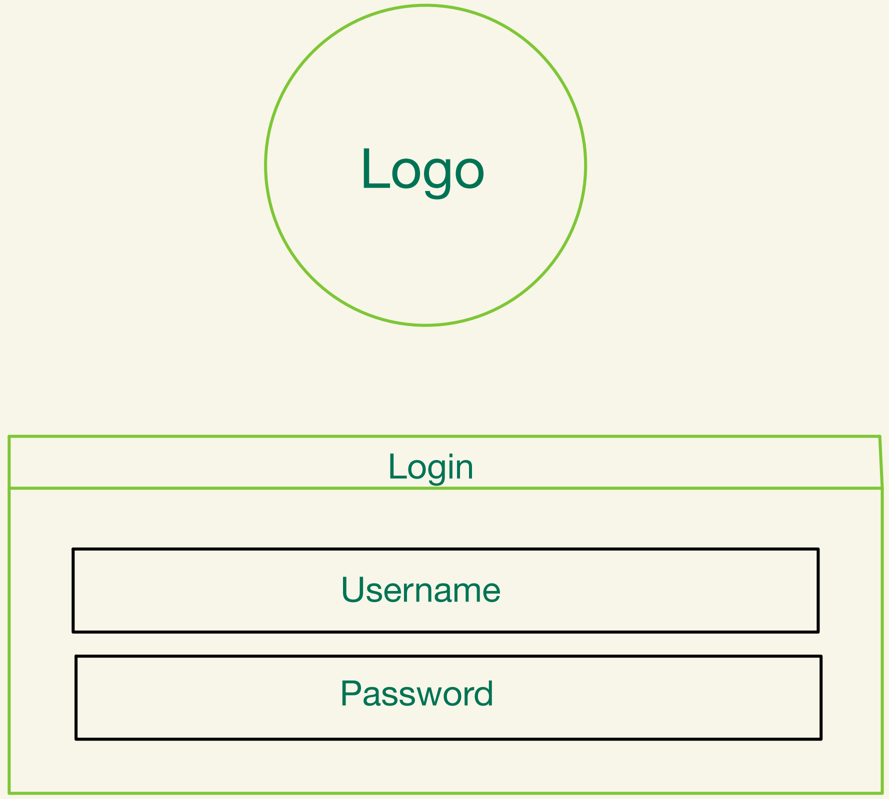
Page 4 of the website:
This page will display the results of "Search by Category."

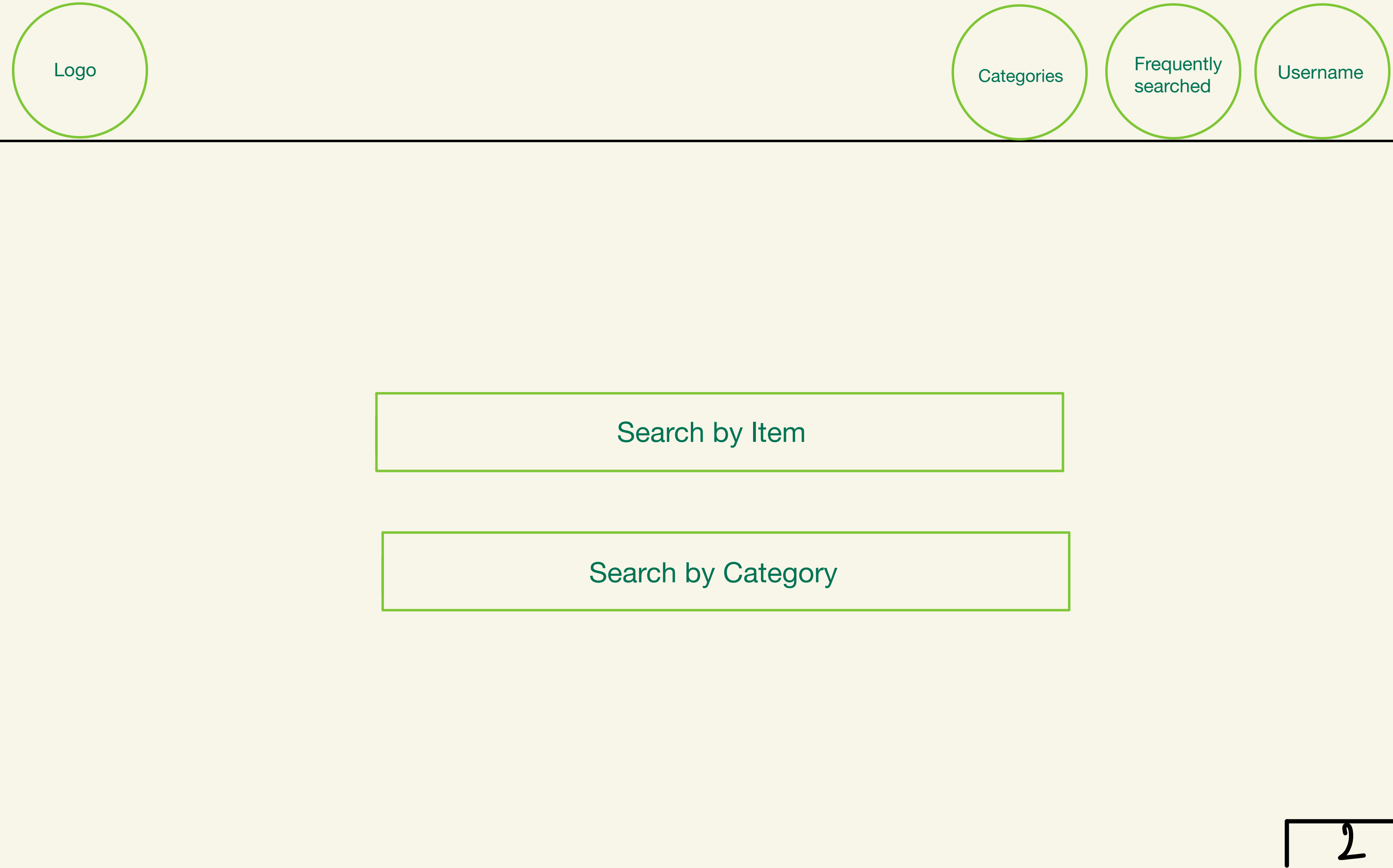
Page 5 of the website:
This page is the Categories page, which will display all the available categories in a table. The table will include the name of the category, the number of the category, and the examples of items included in that category.

Page 6 of the website:
This page is the Frequently Searched page, which displays the items frequently searched by current users, along with the best place to buy those items, along with the price. Every time a user makes a new search a new entry is made into the current user's database to make note of the transaction.

Low-Fidelity UI Mockup

Google Drive Link for UI Mockup: Untitled Notebook (6).pdf


Logo
Login
Username
Password


Logo
Categories
Frequently searched
Username
Search by Item
Search by Category
2

Logo

Categories

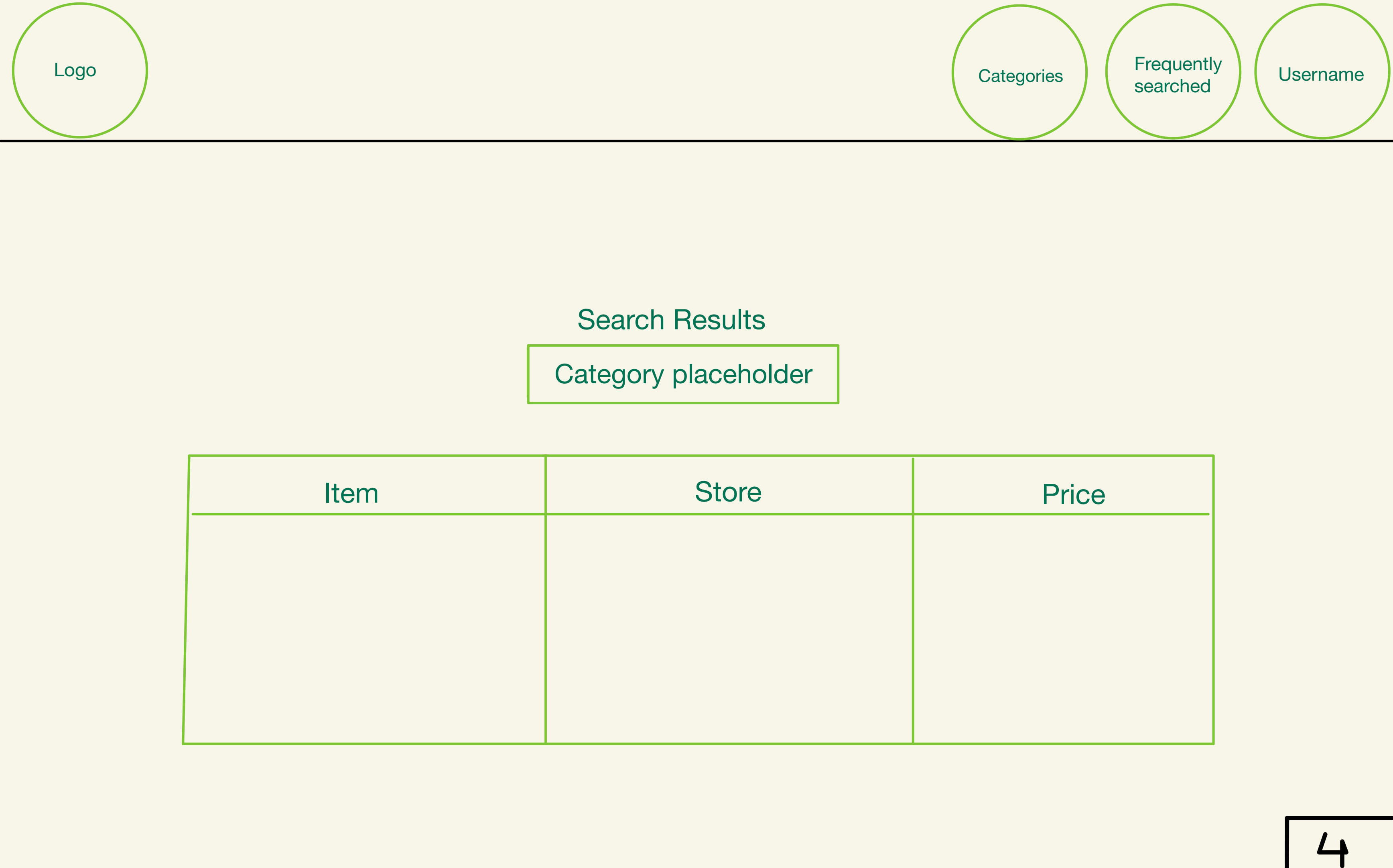
Frequently searched

Username

Search Results

Item placeholder

Best Store: Price:


Logo
Categories
Frequently searched
Username
Search Results
Category placeholder
Item
Store
Price
4

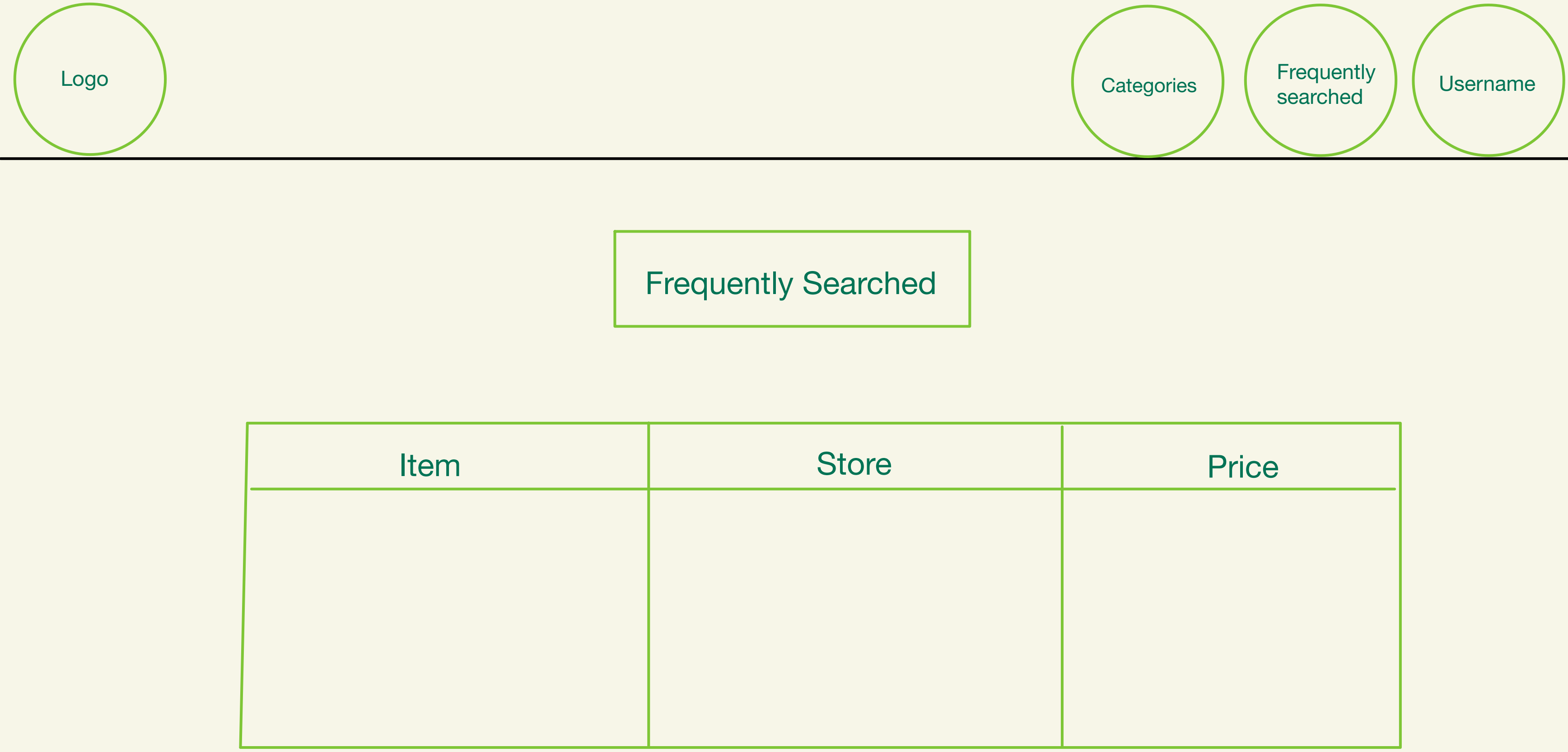
Logo

Categories

Frequently searched

Username

Categories		
Product Type	Number	Items Included
Dairy	1	Eggs, Milk
Vegetables	2	Cucumber, Eggplants
Fruits	3	Strawberries, Apples
Grains	4	Rice, Wheat


Logo
Categories
Frequently searched
Username
Frequently Searched
Item
Store
Price

6

Work Distribution

The work distribution for the proposed project is split up into Frontend tasks and Backend tasks. The following table details the tasks that need to be accomplished from a broad perspective.

Task	Frontend or Backend	Assigned To
Create UI Elements	Frontend	Shreya
Create Data Visualization Elements	Frontend	Jeremy
Create Table Elements	Frontend	Paul
Set up and Scrape Datasets	Backend	Bennett
Merge Datasets	Backend	Bennett
Create Food Comparison Tool	Backend	Shreya
Implement Front/Backend API	Frontend / Backend	Shreya
Create Category Filter	Backend	Jeremy
Create Price Filter	Backend	Paul
Create Quantity Filter and Comparison Tool	Backend	Jeremy
Create Data Visualization Metrics	Backend	Paul
Create Individual Records Table	Backend	Jeremy
Create Username and Passwords Table	Backend	Paul

Each task mentioned in the table above implements some core functionality or additional feature the application will employ. Several of the Backend elements will deal with the database and datasets on hand, whereas the Frontend elements will primarily deal with interfacing with the user as well as conveying information to the user as appropriate.