## Software Requirements Specification

### Existing System

The current situation for compiling a weekly grocery list is an involved process. First, a user must think of several meals they want to make that week, taking into consideration dietary restrictions and how many people they are cooking for. Then, the user must find recipes for each of those meals. Finally, the user must either make a list of each recipe's ingredients (a long and somewhat arduous process) or spend considerable time online shopping for said ingredients.

### Proposed New System

The new system we are proposing would considerably speed up that process. This system could potentially save users several hours per week by automating all work after the stage where the user comes up with the meals for the week - Meaning the recipe finding, list-making, and shopping could all be done through an automated system, giving users their time back.

### Key Operational Features

Our proposed solution features several key features that operate in stages. First, the user must decide what they want to eat that week. That information is then fed into the frontend UI of our system, which sends it to an efficient, cost-effective and low-latency LLM which can generate recipes that can be used to create each meal. When the LLM returns that query, it will be in a specific format that is easy to parse through and separate out the individual ingredients and the associated quantities of each needed for the meal. After that information is parsed, we feed it into the Kroger and Albertsons APIs. Kroger and Albertsons are the two leading grocery conglomerates in the pacific northwest, owning Safeway, Fred Meyer, QFC, and of course Albertsons itself. Given the significant coverage these four stores cover in the grocery market share, this should be enough to return results for any ingredient in any recipe that can generally be considered. After this data is sent to the APIs and the APIs return hits for where these ingredients are, The UI will display each recipe and with it an associated list of ingredients and quantities, as well as where they can be purchased. In future app updates, we could implement online shopping functionality to further streamline the process.

### Users

There is one main user class for this application - anyone who wants to streamline their weekly grocery process. This class has the functionality to input meals and receive back recipes along with ingredients and quantities needed and where the ingredients can be found.  Given this wide user scope, there are many potential users we can engage with for requirements gathering. Essentially, we can look for feedback from anyone we know who is interested in the system. To receive user feedback and validation as well as gather user wants, we will take an iterative approach where after each stage of development we will present our current work and a roadmap of future work to prospective users. We will ask them their thoughts and record their responses to ensure the correct direction for our work. This will ensure that we don't need to do any expensive and time-inefficient redesigns of our software or underlying systems. We will make sure the user base we are receiving feedback from is large enough ($n > 9$) and diverse in life background so that the system works for different people with different levels of free time and technical knowledge.

# Software Design Specification

## System Overview

This is an application that automates grocery shopping by integrating with grocery store APIs. Users can select a meal they want to make, and the application will determine the ingredients needed. The system will then search for these ingredients at online grocery stores, add them to a shopping cart, and redirect users to finalize their purchase. This system simplifies meal planning and grocery shopping, saving time and effort for users.

## Real-World Users & Requirements Elicitation

This project targets individuals who cook regularly but want to streamline grocery shopping. Potential user groups include:

- **Busy Professionals**: People with limited time for grocery shopping and meal planning.
- **Families**: Parents who need to efficiently plan meals and buy groceries.
- **Students**: Those living alone who want to cook without the hassle of searching multiple stores.
- **Elderly Individuals**: Users who may have difficulty going to physical stores.

## Software Architecture

The system follows a modular and scalable architecture:

- **Frontend**: Built using React (or a local app-based UI) for user interaction.
- **Backend**: Implemented in Python using Flask to handle API calls and database operations.
- **Database**: SQLite for storing recipes and past purchase history.
- **External APIs**: Integration with grocery store APIs for inventory data retrieval.

## Technology Stack

- **Frontend**: React or a local application UI
- **Backend**: Python with Flask
- **Database**: SQLite
- **API Integration**: Grocery store APIs
- **Additional Libraries**: Standard Python libraries for data handling and PDF generation

## Software Modules

The system consists of three main modules:

- **Recipe Parser Module**: Extracts ingredients from user-selected recipes using text parsing and keyword extraction techniques.
- **Ingredient Sourcing Module:** Fetches ingredient availability and pricing from grocery store APIs in real time.
- **Shopping Cart Automation Module:** Adds sourced ingredients to an online cart and manages shopping cart logic.

## 3-Week Project Timeline

### Week 1: Requirements & Design

- **Requirements Gathering (Day 1-2):**
  Identify system requirements through user discussions. Define core functionalities, user flows, and system needs.
- **System & UI/UX Design (Day 3-4):**
  Develop system architecture (Frontend, Backend, APIs, Database). Create UI wireframes and design database schema. Outline API integration points.
- **Setup & Initial Development (Day 5-7):**
  Set up development environment, initialize repositories, and CI/CD pipeline. Build basic frontend (React) and backend (Flask) structure. Establish database schema (SQLite).

### Week 2: Core Development & Integration

- **Core Feature Implementation (Day 8-10):**
  Implement Recipe Parser Module and basic frontend UI for meal input. Set up backend logic to handle parsing and data flow.
- **API Integration (Day 11-12):**
  Integrate Kroger and Albertsons APIs. Build Ingredient Sourcing Module to fetch ingredient availability and prices.
- **Shopping Cart Automation (Day 13-14):**
  Develop Shopping Cart Automation Module to compile ingredient lists and add them to online carts, enabling redirection to checkout pages.

### Week 3: Testing, Feedback & Delivery

- **Testing & Monitoring (Day 15-16):**
  Conduct unit and integration tests across modules. Implement monitoring for API calls and error handling.
- **User Testing & Feedback (Day 17):**
  Perform usability testing with diverse users ($n > 9$). Collect feedback and make necessary UI/UX adjustments.
- **Documentation & Polishing (Day 18-19):**
  Write technical and user documentation, including a detailed README.md. Finalize setup instructions and system overviews.
- **Final Testing & Delivery (Day 20-21):**
  Complete system verification and validation. Conduct installation and integration testing. Finalize bug fixes and deliver the complete system with documentation.