Team Deliverable 2 – Requirements Specification

Project: What-Can-We-Cook?

Team: Pseudocode Warriors

Course: ITIS 3300-002

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements specifications for the system and web application developed by the Pseudocode Warriors team. This document serves as the foundation for project design, implementation, and evaluation.

The document is intended for project stakeholders, including the development team, the course instructors, and teaching assistants. It establishes the functional requirements, non-functional requirements, and interface specifications necessary to implement the system successfully. The requirements are defined in detail to guide development, ensure consistency across the team, and provide criteria for validation and testing of the final product.

1.2 Scope

The system is a web application that enables users to manage their pantry, browse and search recipes, follow step-by-step instructions, and generate grocery lists for missing ingredients. The system will support recipe management by allowing users to view titles, ingredients, preparation steps, and optional video links. Instructional support will be provided through both written steps and embedded YouTube videos. Core functionality will focus on these features, with basic data development done through SQLite.

Out of scope for this project are recipe scaling, favorites, advanced filters, or basic recipe submissions with approval, but these are not required for the successful completion of the system.

1.3 References

Youtube Video Embed:
https://developers.google.com/youtube/iframe_api_reference

USDA Food Dataset:
https://fdc.nal.usda.gov/download-datasets

2. Overall Description

2.1 System Overview

The system is a web-based application and users interact with the system through a simple browser interface. The backend will be implemented using FastAPI and will handle requests, apply business logic, and communicate with the database. Data storage will be managed by either a relational database (SQLite), depending on project direction. Recipes and pantry information are stored persistently, while external services such as YouTube may be used to embed video instructions.

2.2 System Structure

- **User Interface:** Implemented using HTML, CSS, JS, and Bootstrap to provide a responsive and accessible design.
- **Application Backend:** Built on FastAPI, responsible for handling recipe management, pantry logic, and grocery list generation.
- **Database Layer:** Provides persistent storage of recipes, ingredients, and user pantry data.

2.3 Subsystems

- **Recipe Management:** Handles recipe storage, searching, and retrieval of details including ingredients, steps, and optional videos.
- **Instructional Support:** Displays step-by-step cooking instructions and embeds YouTube videos when available.
- **Pantry:** Allows users to maintain an up-to-date list of available ingredients.
- **Smart Grocery List:** Compares selected recipe requirements to the pantry contents and generates a list of missing items for shopping.

3. Specific Requirements

3.1 Functional Requirements

- FR1.1: The system shall allow users to search recipes by name.
- FR1.2: The system shall allow users to search recipes by ingredient.
- FR 2.1: The system shall display the recipe title.
- FR 2.2: The system shall display the recipe description.
- FR 3: The system shall provide step-by-step cooking instructions.

- FR 4: The system shall embed an instructional video if a valid link is provided.
- FR 5.1: The system shall allow users to add items in their pantry.
- FR 5.2: The system shall allow users to remove items in their pantry.
- FR 6.1: The system shall compare recipe ingredients with pantry items.
- FR 6.2: The system shall identify and display missing ingredients.

3.2 Non-Functional Requirements

1. The system should return search results and load recipe details within seconds under normal conditions.
2. User pantry data shall be restricted to that user and not accessible to others.
3. The interface shall be intuitive, with simple navigation and responsive design to support both desktop and mobile browsers.
4. The system shall run on common web browsers across Windows, macOS, and Linux platforms without requiring installation.

3.3 Interface Requirements

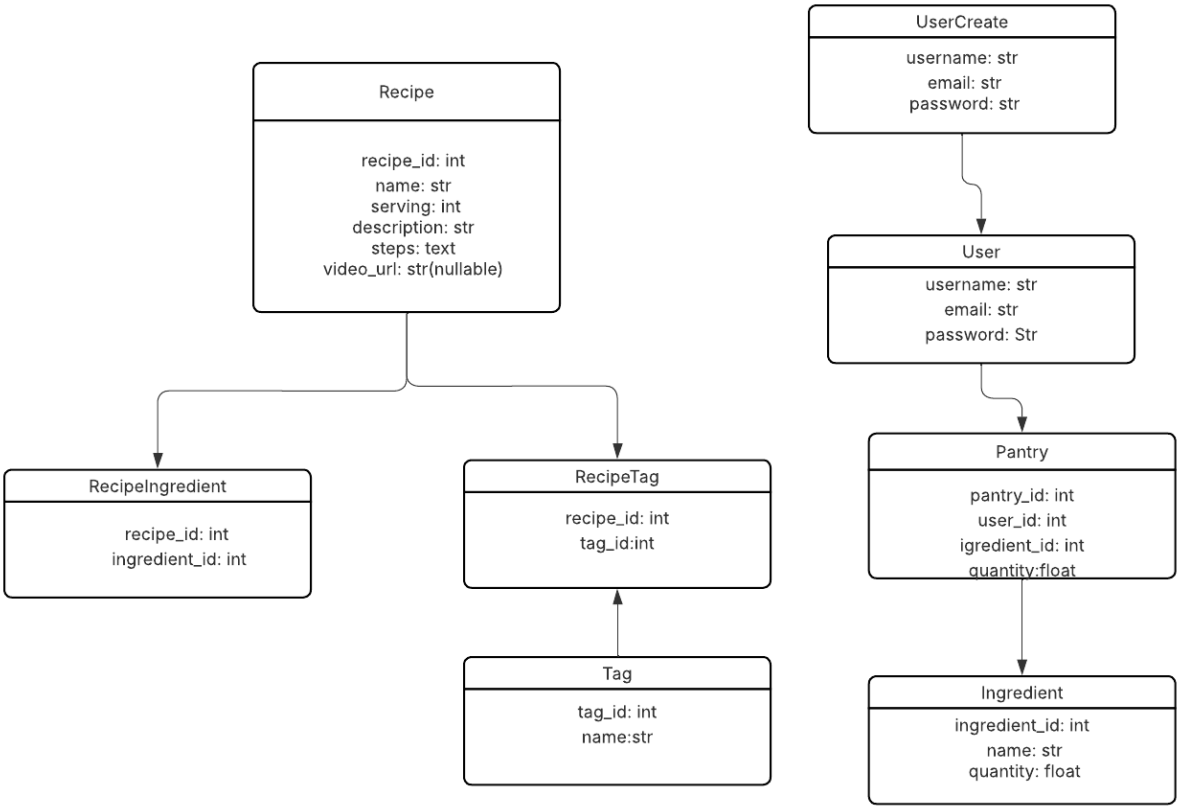
The system will operate through a standard browser-based user interface that can be accessed on both desktop and mobile devices. No specialized hardware is required beyond a computer or smartphone with internet access. Software interfaces will consist of a FastAPI backend connected to a database such as SQLite for data persistence, along with support for embedding external media such as YouTube videos for instructional purposes. Communication between the frontend and backend will rely on standard HTTPS protocols, ensuring compatibility across common web browsers.

4. Development Phases

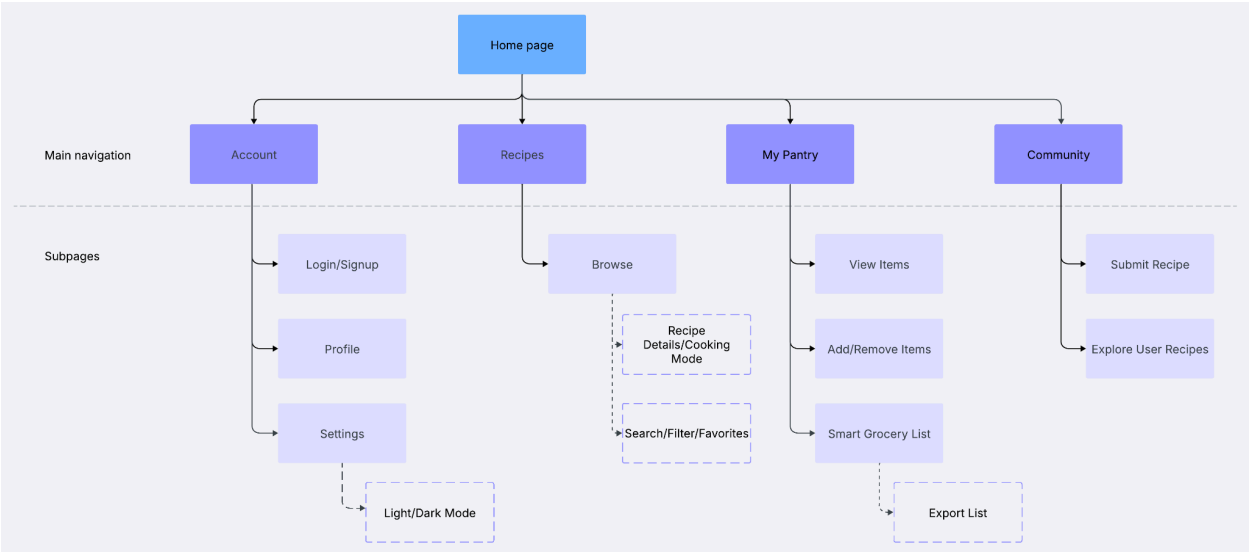
Phase	Tasks / Features	Details	Requirement	Priority
Phase I – Core System	Setup development environment	Configure GitHub repo, Trello board, virtual environments		High
	Database schema	Define tables/collections for recipes, ingredients, pantry	**FR1.1, FR1.2, FR2.1, FR2.2**	High

	Basic backend API	Implement FastAPI endpoints for recipes	**FR1.1, FR1.2, FR2.1, FR2.2** **NFR2**	High
	Pantry management	Enable add/remove items; persist pantry data	**FR5.1, FR5.2**	High
	Smart grocery list	Compare recipe ingredients vs. pantry, display missing items	**FR 6.1, FR6.2**	High
	Core Usability	Ensure UI works and website loads	**NFR1, NFR3, NFR4**	
Phase II – Enhance ments	Grocery list export	Allow download of lists		Medium
	Recipe submission/approval	Users submit new recipes, admin approves before display		Medium
	Tags, filters, favorites	Add search filters (time, cuisine, dietary) and bookmarking		Medium
	UI improvements	Refine layout, ensure mobile responsiveness	**NFR3, NFR4**	Medium
Phase III – Stretch Goals	Cooking mode	Step-by-step 'Next' view for instructions	**FR3, FR4**	Low
	Nutrition field	Add optional manual 'calories per serving'		Low
	Final testing and polish	QA, bug fixes, documentation, prepare final presentation		High

Project Entity Relationship Diagram



Website Map



Team Contributions

Member name	**Roles**	**Overall Contribution (%)**
Ben Oz Elhadad	UI/UX	17%
Darell Isaac Sam	Backend Dev	14%
David Lucero	PMO + Doc/QA	19%
Nicholas Smit	Backend Dev / Infrastructure	19%
Sebastian Lopez	Backend Dev	14%
Stanley Lu	Database Management	17%