Software Requirements Specification

Pantry Pal

1.0.0

Elijah Loy
Denver Woolard
Makaela Bennett

Feb 11, 2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Initialized PantryPal	01/28/25	Created the initial Git Repository	0.0.0
Initialized Home, Pantry, Shopping Cart, and Recipes	02/12/25	Basic Outline with raw HTML files	0.1.0
Setup Angular Environment	02/22/25	Added angular framework as the basis for the frontend	0.2.0
Updated Pantry UI	03/10/25	Visual changes to the pantry section	0.2.1
Updated UI	03/14/25	Visual changes to all sections	0.2.2
Visual and functional updates	03/15/25	Changes in all sections including a share button for recipes	0.2.3
Added Json uses when selecting ingredients	03/27/25	Making a simpler user experience	0.2.4
Added Database initialization files	03/27/25	Added database files for PostgreSQL	0.2.5
Added Spring boot implementation	04/07/25	Utilizing Spring Boot for the communication between our Angular frontend and PostgreSQL Database	0.3.0
Debugging Updates	04/08/25	Changing/adding necessary java files so things can start communicating	0.3.1
Angular & Spring Boot Updates	04/12/25	Pantry is completed while other sections need work. General Debugging	0.4.0
Recipes section is complete	04/21/25	An ‘Auto remove’ and ‘cook’ function is added. Barring unit testing, Recipes section is complete	0.4.1
Errors with database	04/22/25	Debugging Gradle build errors, as well as filling the database in with test values	0.4.2
Shopping Cart section is complete	04/22/25	Linked Shopping cart with Pantry section for ease of use. Auto adds ingredients to pantry when checking out.	0.4.3
Home directory cleanup	04/24/25	Removed unnecessary files in preparation for unit testing	0.5.0
Main UI changes	04/27/25	Updated color scheme and UI elements	0.5.1
Errors for centering titles	04/28/25	Fixed some of the centering for the titles	0.5.2
Errors for scrolling on pantry / titles	04/28/25	Fixed scrolling bug for the pantry ingredients / fixed the titles again	0.5.3
Release	04/29/25		1.0.0

(taken from the GitHub Commits)

1. Introduction

1.1 Purpose

PantryPal is a user-friendly web application designed to help individuals manage recipes and ingredients efficiently. The platform allows users to create, manage, and delete recipes, organize ingredients, and track pantry inventory dynamically. Additionally, the application includes a shopping cart feature for adding missing ingredients and supports user accounts for personalized recipe management. This document specifies the software requirements for PantryPal, including functional and non-functional requirements, constraints, and dependencies.

1.2 Document Conventions

This document follows standard software engineering conventions for Software Requirements Specification (SRS). Requirements are categorized into functional and non-functional categories and are prioritized accordingly. Bold text is used for section headings, and italics are used for emphasis when necessary. Numbered lists and bullet points are utilized to enhance readability and clarity.

1.3 References

This SRS references the following documents and web resources:

- PantryPal GitHub Repository: https://github.com/Ciriden/PantryPalDev
- CS3900 App Software Development GitHub Repository: https://github.com/pattonsgirl/CS3900-AppSoftwareDev/tree/main/Microservices
- Angular Documentation: https://angular.dev/
- Spring Boot Documentation: https://docs.spring.io/spring-boot/index.html
- ChatGPT: https://chat.openai.com/

2. Overall Description

2.1 Product Perspective

PantryPal is a new, self-contained web application designed to help individuals and families manage their pantry inventory and recipes more efficiently. It does not replace an existing system but provides an innovative approach to integrating meal planning, ingredient tracking, and shopping list generation in a seamless experience.

2.2 Product Features

PantryPal includes the following major features:

- **Recipe Management:** Create, manage, and delete recipes, organizing them into step-by-step instructions with ingredient amounts and units.
- **Ingredient Tracking:** Maintain a pantry inventory that dynamically updates as ingredients are used or purchased.
- **Shopping Cart:** Auto-generate a shopping list based on missing ingredients needed for selected recipes.
- **User Accounts:** Save personal recipes, track pantry inventory, and customize shopping lists.
- **Allergy and Substitution Options:** Offer alternative ingredient suggestions for dietary restrictions.
- **Scalable Servings:** Adjust recipes based on the desired number of servings.
- **Built-in Timer:** Assist user with tracking cooking and preparation times.
- **Minimalistic and Visual Design:** More images, less text, making it user-friendly for everyday people, not just experienced chefs.
- **Social Sharing (Stretch Goal):** Allow users to share and rate recipes.

2.3 User Classes and Characteristics

PantryPal is designed for a wide range of users, including:

- **Home Cooks:** Individuals who frequently prepare meals and want an easy way to manage recipes and pantry inventory.
- **Busy Families:** Parents and caregivers looking for a streamlined approach to meal planning and grocery shopping.
- **Heal-Conscious Users:** People with dietary restrictions or allergies who need an easy was to track substitutions.
- **Beginner Cooks:** Users new to cooking who benefit from simple, guided recipes.

2.4 Operating Environment

PantryPal will operate in the following environment:

- **Hardware:** Compatible with desktop computers, laptops, tablets, and smartphones.
- **Operating Systems:** Windows, macOS, Linux, iOS, and Android.
- **Web Browsers:** Google Chrome, Mozilla Firefox, Safari, Microsoft Edge.
- **Backend Technologies:** Java SpringBoot/ Gradle, PostgresSQL for database management.
- **Frontend Technologies:** Angular framework with TypeScript and HTML/CSS.

2.5 Design and Implementation Constraints

The development of PantryPal is subject to the following constraints:

- **Project Timeline:** The development period is limited to a single semester.
- **Security Considerations:** User authentication and authorization for account security.
- **Third-Party Libraries:** Use of external libraries for UI components and database interactions.
- **Scalability:** The system should handle multiple users without performance degradation.

2.6 Assumptions and Dependencies

The development of PantryPal assumes the following:

- **Internet Connectivity:** The application relies on an active internet connection for cloud-based storage and data retrieval.
- **Third-Party APIs:** Integration with external services (e.g., grocery stores for shopping list fulfillment) is a potential future addition but not required for the initial release.
- **Hosting Platform:** PantryPal will be deployed using a container such as Docker.
- **User Device Compatibility:** The web app will be optimized for modern web browsers and mobile devices but may not fully support outdated browsers.

3. System Features

3.1 Recipe Management

3.1.1 Description and Priority

This feature allows users to create, manage, and delete recipes, organizing them into step-by-step instructions with ingredient amounts and units. Priority: **High**.

3.1.2 Stimulus/Response Sequences

- **User Action:** User creates a new recipe by providing a title, ingredients, and steps. **System Response:** System saves and displays the recipe.
- **User Action:** User edits an existing recipe. **System Response:** System updates the recipe details.
- **User Action:** User deletes a recipe. **System Response:** System removes the recipe from the database.

3.1.3 Functional Requirements

REQ-1: The system must allow users to create, edit, and delete recipes.
REQ-2: The system must store recipes with a title, ingredients, steps, and optional images.
REQ-3: Users must be able to scale ingredient amounts based on the number of servings.
REQ-4: The system should provide error handling for missing or invalid inputs.

3.2 Ingredient Tracking

3.2.1 Description and Priority

This feature enables users to maintain a pantry inventory that dynamically updates as ingredients are used or purchased. Priority: **High**.

3.1.2 Stimulus/Response Sequences

- **User Action:** User adds a new ingredient to the pantry. **System Response:** System saves and displays the ingredient.
- **User Action:** User removes an ingredient from the pantry. **System Response:** System updates the inventory.
- **User Action:** User uses an ingredient in a recipe. **System Response:** System deducts the ingredient quantity from the pantry.

3.1.3 Functional Requirements

REQ-1: The system must allow users to create, edit, and delete recipes.

REQ-2: The system must store recipes with a title, ingredients, steps, and optional images.

REQ-3: Users must be able to scale ingredient amounts based on the number of servings.

REQ-4: The system should provide error handling for missing or invalid inputs.

3.3 Shopping Cart

3.2.1 Description and Priority

This feature auto-generates a shopping list based on missing ingredients needed for selected recipes. Priority: **Medium**.

3.1.2 Stimulus/Response Sequences

- **User Action:** User selects a recipe and adds missing ingredients to the shopping cart. **System Response:** System compiles a shopping list.
- **User Action:** User manually removes an item from the shopping list. **System Response:** System updates the list.

3.1.3 Functional Requirements

REQ-1: The system must generate a shopping list based on missing ingredients.

REQ-2: Users must be able to add and remove items from the shopping list manually.

4. External Interface Requirements

4.1 User Interfaces

The PantryPal application will feature a user-friendly, responsive graphical user interface (GUI) designed for both desktop and mobile devices. The interface will follow modern UI/UX principles, with clean layouts, intuitive navigation, and consistent styling. The primary interface components inlude:

- **Dashboard**: Displays quick access to recipes, pantry items, and shopping list.
- **Recipe Manager**: Interface for creating, editing, scaling, and deleting recipes.
- **Pantry Tracker**: A table/grid view for tracking available ingredients and quantities.
- **Shopping Cart**: List of missing ingredients with the ability to add or remove items.
- **Navigation Bar**: Present on all screens, providing access to main features and account settings.

The UI will include accessible color contrasts, readable fonts, and touch-friendly elements. Standard icons will be used to ensure visual consistency.

4.2 Hardware Interfaces

PantryPal is designed to be compatible with the following device types:

- **Desktop and Laptop Computers**: Mouse and keyboard inputs supported.
- **Mobile Devices and Tablets**: Touchscreen input supported, with mobile-optimized layout.
- **Hardware Requirements**: No special hardware beyond a standard computing device with internet access.

No direct communication with specialized external hardware is required for the initial version of the application.

4.3 Software Interfaces

PantryPal will interface with the following software components:

- **Backend Framework**: Java SpringBoot/Gradle (handles business logic, API endpoints, and data processing).
- **Frontend Framework**: Angular (TypeScript) for dynamic and modular client-side rendering.
- **Database**: PostgreSQL for storing user data, recipes, ingredients, and preferences.

Data transmitted between frontend and backend will follow JSON format via RESTful APIs.

4.5 Communications Interfaces

PantryPal requires the following communication protocols:

- **HTTP/HTTPS**: All client-server communication will be secured using HTTPS
- **Web Browsers**: Compatible with Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge
- **Network Server Protocols**: Standard TCP/IP for data transfer.

- **Message Formats**: JSON will be used for request and response payloads.

Future versions may include optional email notifications for reminders for shared recipe features, which would integrate with a third-party email API service (e.g., SendGrid or Mailgun).

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The PantryPal application will load the main dashboard and user recipe list within two seconds under normal network conditions. The system will support up to one-hundred concurrent users without performance degradation. Backend API calls should respond within one second for common operations (e.g., fetching recipes, updating pantry items). These performance goals ensure a smooth and responsive user experience, particularly during peak usage times such as evenings and weekends.

5.2 Safety Requirements

Although PantryPal does not involve physical safety, data integrity and user error prevention are key concerns. The system must implement confirmation prompts for destructive actions such as deleting recipes or clearing pantry inventory. Additionally, unsaved changes should trigger warning dialogs to prevent unintentional data loss. Data backups will be performed daily during development to safeguard against accidental data loss.

5.3 Security Requirements

User data and account information must be protected through secure authentication and authorization methods. PantryPal will require users to register with a valid email and password and use encrypted HTTPS connections to prevent eavesdropping. Passwords will be stored using a salted hashing algorithm. Session management must follow best practices to prevent hijacking. Sensitives user data, including allergy preferences and account credentials, must not be exposed via client-side code or unsecured endpoints.

6. Key Resource Requirements

Major Project Activities	Skill/Expertise Required	Internal Resource	External Resource
Requirements Gathering	Business Analysis	Team Members	Professor/Product Interview
UI/UX Design	User Interface & Experience Design	Angular/HTML /CSS Devs	ChatGPT/ CSS Documentation
Frontend Development	Angular, HTML/CSS/JS	Angular Devs	Angular Documentation / Chat GPT
Backend Development	Java SpringBoot/ Gradle, RESTful APIs	Java Devs	Springboot Documentation / Chat GPT
Mobile App Integration	Docker (Containers)	DevOps Engineers	Docker Hub
Database Design	PostgreSQL, Database Schema	Database Admin	PostgreSQL Documentation / ChatGPT
Testing & QA	Unit testing, integration testing	QA Engineer	Jest
Security Implementation	HTTPS, Authentication, Encryption	Team Members	General security guidelines
Deployment & DevOPps	CI/CD, Cloud hosting	DevOps	GitHub Action/ Docker
Documentation	Technical Writing, SRS/ User Manuals	Technical Writer	Markdown Documentation/ Provided Project Templates
Maintenance & Support	Debugging, Performance Optimizations	Full-Stack Devs	N/A
Project Management	Agile/Scrum Principles	Project Lead	N/A