SP-5 — Purple Groceries App

SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

Course: CS 4850 Senior Project

Section 03

Semester: Fall 2025

Date: September 12

Professor: Sharon Perry

 Carinne Tzurdecker Documentation	 Vanessa Espinoza Documentation	 Shane Hazeur Developer	 Piao Hou Developer

Table of Contents

1.0 Introduction

1.1 Overview

The AI Grocery & Recipe Planner gathers user goals (weight loss/gain/maintenance), dietary constraints (allergies, exclusions, inclusions), budget, time, and equipment, plus a short free chat "brief." It generates:

1.1.1 A personalized multi-day meal plan with recipes,
1.1.2 Consolidated grocery list (unit-normalized, de-duplicated)
1.1.3 Optional substitutions respecting constraints.

1.2 Project Goals

1.2.1 Provide fast, personalized weekly plans and shopping lists.
1.2.2 Respect hard constraints (allergens, exclusions) 100% of the time.
1.2.3 Minimize user edits (≤ 5 edits to accept a weekly plan).
1.2.4 Keep latency low (≤ 6 s for plan generation under typical load).

1.3 Definitions and Acronyms

1.3.1 LLM: Large Language Model (used for brief extraction and planning).
1.3.2 RAG: Retrieval-Augmented Generation (recipe retrieval by embeddings).
1.3.3 MVP: Minimum Viable Product.
1.3.4 JWT JSON Web Token

1.4 Assumptions

1.4.1 Users are the non-clinical general population (no medical advice).
1.4.2 Internet connectivity required.
1.4.3 Initial recipe corpus is licensed/owned by the team or properly attributed.
1.4.4 Mobile-first UI; desktop web is optional.

2.0 Design Constraints

2.1 Environment

2.1.1 Client: Android 10+ (Java), minSdk 26, targetSdk latest.
2.1.2 Server: Java 21, Spring Boot 3.x, Gradle build.

3.0 Functional Requirements

3.1 Account & Authentication (Android + Spring Security/JWT)

3.1.1 Create account (email + password).
3.1.2 Login with username/password.
3.1.3 Secure logout / token revocation.
3.1.4 Password reset (email link) (stretch).

3.2 User Profile (Goals & Constraints)

3.2.1 CRUD profile fields: goal, calories/protein targets (optional), diet tags, allergies, include/exclude foods, budget/week, max cook time, default servings, equipment.

3.2.2 Store “pantry” toggles for common staples (optional).

3.2.3 Validate entries (e.g., non-negative numbers, sane ranges).

3.3 Free-Chat Brief

3.3.1 Capture user’s natural-language preferences/notes in app chat.

3.3.2 Store brief with timestamp and link to active profile.

3.4 Recipe Management & Search

3.4.1 Store recipes (title, steps, ingredients {name, qty, unit}, tags, nutrition/serving).

3.4.2 Maintain embeddings for semantic search (pgvector).

3.4.3 Search recipes by keyword/tags and by vector similarity.

3.4.4 Admin import/curation endpoint for new recipes (MVP internal).

3.5 Meal Plan Generation

3.5.1 Generate plan for 3–7 days with breakfast, lunch, dinner (+ optional snack).

3.5.2 Hit daily calorie/protein targets within ±10% when targets exist.

3.5.3 Strictly enforce allergens/exclusions = 0 violations (pre- and post-validation).

3.5.4 Provide 2–5 swap candidates per meal.

3.6 Grocery List Builder

3.6.1 Consolidate ingredients across plan; normalize units; sum quantities.

3.6.2 Group items by aisle/category; show common substitutes.

3.6.3 Allow user edits: mark “already have,” adjust qty/unit, delete items.

3.6.4 Export/share list (Android ShareSheet; PDF optional).

3.7 Planner Controls & Modes

3.7.1 User selects plan duration (3–7 days) and default servings (1–6).

3.7.2 “Meal-prep mode” permits intentional repeats to reduce cost/time.

3.7.3 Budget mode prefers lower-cost equivalents when available.

3.8 Safety & Compliance

3.8.1 Block any candidate recipe containing allergens/exclusions.

4.0 Non-Functional Requirements

4.1 Security

4.1.1 All traffic over HTTPS/TLS 1.2+.
4.1.2 Passwords hashed (bcrypt/argon2).
4.1.3 JWT access tokens; refresh strategy; secure storage on device.
4.1.4 PII encrypted at rest; secrets via cloud KMS.
4.1.5 Parameterized queries; least-privilege DB roles.
4.1.6 Unit/integration tests for allergen enforcement & authorization.

4.2 Capacity

4.2.1 Support ~10 concurrent plan generations with p95 end-to-end ≤ 6 s.
4.2.2 Recipe search first page ≤ 800 ms p95.
4.2.3 Scale to ≥ 1,000 recipes without noticeable degradation (indexes, caching).

4.3 Usability

4.3.1 First-run setup ≤ 90 seconds, ≤ 10 required inputs.
4.3.2 Single-tap meal swap; inline qty edits on grocery list.
4.3.3 WCAG 2.1 AA contrast; dynamic type support.

4.4 Other (Performance, Reliability, Maintainability)

4.4.1 Error budget: < 1% failed generations/day.
4.4.2 Automated CI with unit tests (≥ 70% coverage for planner and merger).
4.4.3 Observability: metrics + alerts for latency/error spikes.

5.0 External Interface Requirements

5.1 User Interface Requirements

5.1.1 Android screens: Onboarding → Free-Chat → Plan Preview → Grocery List.
5.1.2 Plan Preview shows per-meal macros and swap carousel.
5.1.3 Grocery List groups by aisle; pantry toggles; quantity editor; share.

5.2 Hardware Interface Requirements

5.2.1 Device network (Wi-Fi/cellular).
5.2.2 (Stretch) Camera for barcode scanning.

5.3 Communication Interface Requirements

5.4.1 HTTPS; request timeout 15 s for generation; exponential backoff on idempotent GETs.
5.4.2 JWT in Authorization: Bearer <token> header; 401/403 handling on client.