Product Requirements Document (PRD)

Project Name: Recipe Generator Web Application

Prepared By: Abeerah Aamir
Last Updated: July 17, 2025

---

1. Project Overview

The Recipe Generator Web Application is a mobile-first, component-based frontend built using Next.js (App Router). It allows users to generate, view, save, rename, delete, and filter recipes based on ingredients, serving size, and cooking time. It integrates with Supabase, MongoDB, NextAuth, and n8n to support dynamic recipe generation and management.

---

2. Objectives & Goals

- Build a modern, responsive, and clean user interface.
- Follow mobile-first design principles.
- Serve all dummy data via service abstraction in `/services`.
- Implement smooth animations using Framer Motion.
- Prepare the frontend for easy integration with backend systems.

---

3. Target Audience

- Home cooks, culinary enthusiasts, and food bloggers.
- Users interested in personalized or automated recipe generation.

---

4. Directory & Architecture Requirements

- `/services`: Contains all data fetching logic (dummy or real).
- `/components`: Contains modular and reusable UI components.
- `/app/pageName/page.tsx`: Structure for all pages.
- Modern theme with generous spacing, soft shadows, and clean typography.

---

5. Functional Requirements

5.1 Landing Page (`/`)

- Hero Section: Includes title, tagline, and a "Get Started Now" button.
- Features Section: Lists the application's capabilities.
- CTA Button navigates to `/dashboard` with Framer Motion transitions.

5.2 Dashboard (`/dashboard`)

- **No Recipes State**:
    - Centered "Generate Recipe" button.
- **Recipes Present**:
    - "Generate Recipe" button at top-right.
    - Display recipe cards:
        - Includes title, short description, and image.
        - Hover animations.
        - Clicking card navigates to `/recipe-details/[id]`.
        - "More" (3-dots) icon reveals options:
            - Rename (inline or via modal).

        - Delete (with confirmation).

- Filter Options:
    - Sort by Date or Name (ASC/DESC).

5.3 Recipe Generation Page (`/recipe-generation`)

- Input Fields:
    - Ingredients (text or tag input).
    - Serving Size (number input).
    - Time to Cook (dropdown or number input).
- Generate Button:
    - Navigates to `/recipe-details/[id]` with mock data.

5.4 Recipe Details Page (`/recipe-details/[id]`)

- Header:
    - Recipe title.
    - Save button (top-right):
        - Saves to DB (mocked).
        - Redirects to `/dashboard`.
- Body Sections:
    - Ingredients
    - Instructions
    - Cooking Time & Servings
    - Additional Notes (if any)

---

6. Non-Functional Requirements

- **Performance**: Lazy loading, optimized Tailwind usage.
- **Accessibility**: Semantic HTML, keyboard navigation.
- **Responsiveness**: Fully functional across all screen sizes.
- **Animations**: Smooth transitions using Framer Motion.

---

7. Technology Stack

- **Frontend**: Next.js (App Router), TypeScript, TailwindCSS, Framer Motion
- **Backend Integration (future)**: Supabase, MongoDB, NextAuth, n8n
- **Icon Library**: Heroicons or Lucide-react

---

8. Project Scope

In Scope (MVP):

- UI for landing, dashboard, recipe generation, and recipe detail pages.
- Recipe card interactions (rename, delete).
- Framer Motion transitions.
- Dummy data served via service abstraction.

Out of Scope:

- User authentication.
- Real database storage.

- Workflow automation.

---

9. Dependencies

- Framer Motion for animations.
- TailwindCSS for utility-first styling.
- Next.js App Router for routing.
- Supabase, MongoDB, NextAuth, and n8n (for future integration).

---

10. Timeline (Suggestive)

Phase	Tasks	Duration
Phase 1	Landing Page, Hero/Features Section	2 days
Phase 2	Dashboard UI, Card Components, Filters	2 days
Phase 3	Recipe Generation Form & Navigation	1 day
Phase 4	Recipe Details Layout and Save Button Routing	1 day
Phase 5	Polishing, Refactor, Mobile Testing	1 day

---

11. Open Questions

- What is the backend format for recipe data?
- Should dashboard access be protected via auth?
- Should localStorage be used as a placeholder for DB?

- Are rename/delete actions mocked or functional in MVP?
