Product Requirements & Specification Document

Project Name

RecipeBox - Recipe Organizer

Description

RecipeBox is an open-source web application for users to add, edit, categorize, and manage recipes, including ingredients and preparation steps. Built with React, JavaScript, HTML, CSS, and Tailwind.

---

1. Goals & Objectives

- Enable users to create, edit, and delete recipes.
- Allow categorization and filtering of recipes.
- Provide ingredient and step management for each recipe.
- Deliver a clean, responsive, and user-friendly interface.
- Ensure codebase is open-source and easy to contribute to.

---

2. Core Features

Feature	Description
Add Recipe	Users can add new recipes with title, category, ingredients, and steps.
Edit Recipe	Users can modify existing recipes.
Delete Recipe	Users can remove recipes.
Categorize Recipes	Users can assign categories (e.g., Dessert, Main Course) to recipes.
Ingredient Management	Add, edit, and remove ingredients per recipe.
Step Management	Add, edit, and remove preparation steps per recipe.
Recipe List & Search	View all recipes, filter by category, and search by title.
Responsive UI	Application is usable on desktop and mobile devices.

---

3. User Stories

ID	As a…	I want to…	So that…
US1	User	Add a new recipe	I can save my favorite dishes
US2	User	Edit a recipe	I can update or correct details
US3	User	Delete a recipe	I can remove unwanted recipes

US4	User	Categorize recipes	I can organize my collection
US5	User	Manage ingredients and steps	I can accurately follow recipes
US6	User	Search and filter recipes	I can quickly find what I need

---

4. Functional Requirements

ID	Requirement
FR1	Users can create, edit, and delete recipes.
FR2	Each recipe includes: title, category, ingredients (name, quantity), and steps (ordered text).
FR3	Users can assign and filter by categories.
FR4	Users can add, edit, and remove ingredients and steps within a recipe.
FR5	Users can search recipes by title.
FR6	UI is responsive and accessible.

---

5. Non-Functional Requirements

ID	Requirement
NFR1	Built with React, JavaScript, HTML, CSS, and Tailwind.
NFR2	Codebase is open-source and well-documented.
NFR3	Application loads quickly and performs smoothly.
NFR4	No backend; data is stored in browser localStorage.

---

6. Technical Specifications

Architecture

- **Frontend Only:** Single Page Application (SPA) using React.
- **State Management:** React state/hooks.
- **Data Storage:** Browser localStorage for persistence.
- **Styling:** Tailwind CSS for all UI components.

Component Structure

```
App
 ├── Header
 ├── RecipeList
 │    ├── RecipeCard
 ├── RecipeForm
 └── RecipeDetail
```

Data Model (Pseudocode)

```
Recipe = {
  id: string,
  title: string,
  category: string,
  ingredients: [{ name: string, quantity: string }],
  steps: [string]
}
```

---

7. UI/UX Requirements

Area	Requirement
Layout	Clean, minimal, and responsive design using Tailwind.
Accessibility	Keyboard navigable, semantic HTML, color contrast compliant.
Forms	Simple, with validation for required fields.
Navigation	Easy access to add, edit, and view recipes.

---

8. Open Source & Contribution

- Repository hosted on GitHub with MIT License.
- Clear README with setup, usage, and contribution guidelines.
- Issues and pull requests enabled for community contributions.

---

9. Out of Scope

- User authentication or accounts.
- Cloud or server-side data storage.
- Advanced sharing or export features.

---

10. Milestones

Milestone	Description	Target Date
Project Setup	Repo, tooling, base structure	Week 1
Core Features	Add/edit/delete, localStorage	Week 2
UI/UX & Responsiveness	Tailwind styling, mobile support	Week 3
Documentation & Release	README, contribution guide, launch	Week 4

---

11. Acceptance Criteria

- All core features implemented and tested.

- Responsive and accessible UI.
- Code is open-source and documented.
- No critical bugs; smooth user experience.

End of Document