Product Requirements & Specification Document

Project Name

PropView - Real Estate Listing Portal UI

Description

PropView is a responsive frontend portal for browsing, filtering, and viewing real estate listings. It features image galleries, interactive maps, and dynamic content. Built with React, advanced CSS, and Tailwind, PropView targets entrepreneurs and open-source contributors.

1. Goals & Objectives

Goal	Description
User-Friendly Browsing	Enable users to easily browse and search real estate listings
Responsive Design	Ensure seamless experience across devices
Interactive Media	Provide image galleries and map integration for each listing
Open-Source Ready	Clean, modular, and well-documented codebase

2. Core Features

Feature	Description
Listings Grid/List	Display properties in grid/list view with key info
Filters & Search	Filter by price, location, type, bedrooms, etc.; keyword search
Listing Details Page	Detailed view with gallery, map, description, and property info
Image Gallery	Interactive, swipeable image gallery per listing
Interactive Map	Map view with property markers; click to view details
Responsive Layout	Mobile-first, adaptive design using Tailwind CSS
Pagination/Infinite Scroll	Efficient navigation through large datasets
Open-Source Structure	Modular components, clear documentation, MIT license

3. User Stories

As a…	I want to…	So that…
Visitor	Browse and filter listings	I can find properties of interest
Visitor	View property details with images and map	I can evaluate properties visually

Visitor	Use the portal on any device	I have a consistent experience
Contributor	Understand and extend the codebase	I can contribute to the project

---

4. Functional Requirements

4.1 Listings Display

- Fetch and render property listings from a mock API or static JSON.
- Support grid and list views.
- Show thumbnail, price, address, type, and summary.

4.2 Filtering & Search

- Filter by price range, location, property type, bedrooms, bathrooms.
- Keyword search (address, description).

4.3 Listing Details

- Show full property info, image gallery, and map location.
- Display contact or inquiry button (non-functional placeholder).

4.4 Image Gallery

- Swipeable/scrollable gallery with thumbnails.
- Responsive to touch and mouse events.

4.5 Interactive Map

- Integrate with a map provider (e.g., Mapbox, Leaflet, or Google Maps).
- Show property location marker; click to highlight listing.

4.6 Responsive Design

- Use Tailwind CSS for layouts.
- Support mobile, tablet, and desktop breakpoints.

4.7 Pagination/Infinite Scroll

- Paginate or lazy-load listings for performance.

4.8 Open-Source Readiness

- Modular React components.
- Clear README and code comments.
- MIT license.

---

5. Non-Functional Requirements

Requirement	Specification
Performance	Initial load < 2s, smooth UI interactions
Accessibility	WCAG 2.1 AA compliance, keyboard navigation

Browser Support	Latest Chrome, Firefox, Safari, Edge
Code Quality	ESLint, Prettier, functional components
Documentation	README, component docs, contribution guide

---

6. Technical Specifications

Area	Specification
Framework	React (v18+)
Styling	Tailwind CSS, custom CSS modules
State Mgmt	React Context or useState/useReducer
Data Source	Mock API (JSON file or local server)
Map Integration	Mapbox GL JS, Leaflet, or Google Maps JS API
Routing	React Router (v6+)
Tooling	Vite or Create React App, ESLint, Prettier
Testing	Jest, React Testing Library (optional)

---

7. UI/UX Guidelines

- Clean, modern, and minimal interface
- Prominent search and filter controls
- Consistent spacing, typography, and color palette
- Accessible navigation and controls
- Mobile-first layouts

---

8. Milestones & Deliverables

Milestone	Deliverable
1. Project Setup	Repo, tooling, base Tailwind config
2. Listings UI	Grid/list view, mock data integration
3. Filters & Search	Filtering, search bar, UI controls
4. Listing Details	Details page, image gallery, map integration
5. Responsive Design	Mobile/tablet/desktop layouts
6. Documentation	README, code comments, contribution guide

---

9. Out of Scope

- Backend/API development

- User authentication
- Real-time chat or messaging
- Payment or booking functionality

---

10. Appendix

Example Listing Data Structure

```
{
  "id": "123",
  "title": "Modern Loft in Downtown",
  "price": 350000,
  "address": "123 Main St, Cityville",
  "type": "Apartment",
  "bedrooms": 2,
  "bathrooms": 2,
  "images": ["img1.jpg", "img2.jpg"],
  "location": { "lat": 40.7128, "lng": -74.0060 },
  "description": "Spacious loft with city views..."
}
```

End of Document