**Store Management Page – Documentation**

**Members:**  
Kundam Lasya  
Redrouthu Dharshini

**1. Overview**

The Store Management Page in the SweetSpot Doorstep Delight Admin Panel is the central hub for managing store branches. It enables administrators to add, edit, update status, and delete store records with a clear, organized interface.

The page uses a **component-based React structure**, styled with **TailwindCSS** for responsiveness, and leverages **Framer Motion animations** for smooth interactions. Mock data stored in JavaScript arrays powers the functionality during development, ensuring a seamless and instant user experience without external data dependencies.

**2. Purpose**

* Provide an intuitive interface for managing store branch details.
* Enable administrators to add new store branches with complete information.
* Support editing of existing store details for quick updates.
* Allow deletion of outdated or closed branches.
* Offer status toggle for quickly marking stores Active/Inactive.
* Ensure branch address details are included when creating a new store.

**3. Page Structure & Components**

**3.1 Header Section**

* Implemented within the Admin Dashboard layout.
* Displays the **"Store Management"** heading with the SweetSpot admin pink theme.

**3.2 Store List Section**

* Implemented via the StoreList component.
* Displays all stores in a **table/grid format**.
* Columns include: Store Name, Location, Contact, Manager, Status, Actions.
* Includes a search/filter bar for quick navigation.

**3.3 Add Store Form**

* Implemented via the AddStoreForm component.
* Fields: Store Name, **Branch Address**, Contact, Email, Manager Name, Opening Hours, Status.
* Includes validation to ensure all fields, including **branch address**, are filled before submission.

**3.4 Edit Store Functionality**

* Integrated within the Store List actions.
* Opens a pre-filled form for updating branch details.

**3.5 Delete Store Functionality**

* Delete action triggers a confirmation modal before removing a branch from the list.

**3.6 Status Toggle**

* Allows admins to instantly change a store’s availability between **Active/Inactive**.

**3.7 Footer Section**

* Admin panel footer with branding and optional quick links.

**4. Data Handling**

**4.1 Data Sources**

* Store data stored in **local JavaScript arrays** during development.

**4.2 State Management**

* **React useState** for managing store list and form data.
* Data updates trigger immediate UI refresh without page reload.

**5. Implementation Approach**

**5.1 Page Initialization**

* Page loads store data from a **local JavaScript array** into component state.

**5.2 Adding a Store**

1. Admin clicks **Add Store**.
2. Form opens, requiring all fields, **including Branch Address**, to be filled.
3. On submission, new store is added to the state array.
4. Store list updates instantly.

**5.3 Editing a Store**

1. Admin clicks **Edit** for a specific store.
2. Form opens with pre-filled data.
3. On save, updated store data replaces old entry in state.

**5.4 Deleting a Store**

1. Admin clicks **Delete**.
2. Confirmation modal appears.
3. On confirm, store entry is removed from state.

**5.5 Status Toggle**

* Clicking the status toggle updates the store’s status in state.

**6. User Flow**

1. Admin navigates to **Store Management** page from dashboard menu.
2. Views the list of all stores.
3. Adds a store with **branch address** and other required details.
4. Edits, deletes, or toggles store statuses as required.
5. UI updates instantly after every action.

**7. Key Features Summary**

* Add, edit, and delete store branches.
* Include **branch address** when creating a store.
* Toggle store status instantly.
* Search and filter store list.
* Responsive layout for desktop, tablet, and mobile.
* Smooth animations for modals and list updates.
* Consistent pink-themed admin design with Parastoo font.

**8. Dependencies**

* **React:** Component-based UI.
* **TailwindCSS:** Styling and responsiveness.
* **Framer Motion:** Smooth UI animations.
* **React Icons:** For action buttons (edit, delete).
* **@fontsource/parastoo:** Custom font styling.