Angular Contact Book CRUD Application (Standalone Components)

This is a simple Contact Book application built with Angular using standalone components (without traditional NgModules). It provides full CRUD (Create, Read, Update, Delete) functionality for managing contacts.

Key Features

1. **Contact Management**:
   * View all contacts in a table
   * Add new contacts
   * Edit existing contacts
   * Delete contacts
2. **Technical Implementation**:
   * Built with Angular's standalone components API (no NgModules)
   * Uses Angular Router for navigation
   * Implements a service for data management
   * Uses reactive forms for contact input
   * Includes Bootstrap for basic styling

Project Structure

Copy

Download

src/app/

├── components/

│ ├── contact-form/ # Reusable form component

│ └── contact-list/ # Displays all contacts

├── models/

│ └── contact.ts # Contact interface

├── pages/

│ ├── add-contact/ # Add new contact page

│ └── edit-contact/ # Edit contact page

├── services/

│ └── contact.service.ts # Manages contact data

└── app.component.ts # Root component

Key Components Explained

1. Contact Service (contact.service.ts)

* Acts as the data layer
* Manages an in-memory array of contacts
* Provides methods for all CRUD operations:
  + getContacts() - Returns all contacts
  + getContactById() - Returns single contact
  + addContact() - Adds new contact
  + updateContact() - Updates existing contact
  + deleteContact() - Removes contact

2. Contact List Component

* Displays all contacts in a table
* Handles delete operations
* Provides links to edit and add pages
* Uses the ContactService to fetch data

3. Contact Form Component (Reusable)

* Handles contact input with validation
* Emits submit events
* Used by both Add and Edit components
* Uses Angular's FormsModule for two-way binding

4. Add Contact Page

* Uses the ContactForm component
* Handles form submission
* Calls ContactService to add new contact
* Navigates back to list after adding

5. Edit Contact Page

* Fetches contact data based on route parameter
* Uses the same ContactForm component
* Calls ContactService to update contact
* Navigates back to list after updating

Routing Configuration

The application has three routes:

1. / - Shows the contact list (default route)
2. /add - Shows the add contact form
3. /edit/:id - Shows the edit form for a specific contact

Routes are configured directly in main.ts using provideRouter().

Standalone Components Approach

Key differences from traditional Angular modules:

1. Each component is self-contained with its own imports
2. No shared NgModule declarations
3. Components declare their own dependencies
4. Routing is configured at application bootstrap

Data Flow

1. **Reading Data**:
   * Components request data from ContactService
   * Service returns the current contact list or specific contact
2. **Writing Data**:
   * Form submissions emit contact data
   * Page components call appropriate service methods
   * Service updates its internal array
   * UI updates automatically through Angular's change detection

Styling

* Uses Bootstrap for basic responsive styling
* Custom CSS in styles.css for additional styling
* Clean, table-based layout for the contact list
* Form styling with proper spacing and validation

Potential Enhancements

1. Add proper form validation messages
2. Implement search/filter functionality
3. Add contact categories or tags
4. Connect to a real backend API
5. Add proper error handling
6. Implement unit tests

This application demonstrates how to build a complete CRUD application in Angular using the modern standalone components approach, providing a foundation you can extend with additional features.