Creating a Single-Page Application (SPA) with Angular involves leveraging its routing capabilities to load content dynamically without refreshing the page. Below is a step-by-step guide:

### 1. Set Up the Angular Project

#### **Install Angular CLI:**

npm install -g @angular/cli

1.

#### **Create a New Project:**

ng new my-spa-app cd my-spa-app

2.

#### Run the Development Server:

ng serve

3. Open your browser at http://localhost:4200.

# 2. Create Components

To structure your SPA, create components for each view or page.

#### **Generate Components:**

ng generate component home ng generate component about ng generate component contact

#### This will create:

- A HomeComponent for the home page.
- An AboutComponent for the about page.
- A ContactComponent for the contact page.

### 3. Set Up Angular Routing

Routing is essential for navigation in an SPA.

- 1. Open the app-routing.module.ts file, which is automatically generated when you create an Angular project.
- 2. Define the routes:

```
Code (app-routing.module.ts):
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { HomeComponent } from './home/home.component';
import { AboutComponent } from './about/about.component';
import { ContactComponent } from './contact/contact.component';
const routes: Routes = [
 { path: ", component: HomeComponent }, // Default route
 { path: 'about', component: AboutComponent },
 { path: 'contact', component: ContactComponent },
 { path: '**', redirectTo: ", pathMatch: 'full' }, // Wildcard route
];
@NgModule({
 imports: [RouterModule.forRoot(routes)],
 exports: [RouterModule]
})
export class AppRoutingModule { }
Import the AppRoutingModule into your AppModule:
import { AppRoutingModule } from './app-routing.module';
   3.
```

# 4. Define Navigation Links

Add navigation links in the main template to allow users to switch between pages.

```
display: flex;
gap: 15px;
background: #f8f9fa;
padding: 10px;
border-bottom: 1px solid #ddd;
}

a {
text-decoration: none;
color: #007bff;
font-weight: bold;
}

a.active {
text-decoration: underline;
}
```

## 5. Add Content to Components

Populate the components with content:

```
HomeComponent (home.component.html):
```

<h1>Welcome to the Home Page</h1>This is the main page of your SPA.

### AboutComponent (about.component.html):

<h1>About Us</h1>
Learn more about our organization.

### ContactComponent (contact.component.html):

<h1>Contact Us</h1>
Feel free to reach out for more information.

### 6. Enable SPA Features

Angular supports navigation without reloading the page by default. However, ensure the following:

#### Base Tag:

In index.html, add a <base> tag to handle relative paths:

```
<base href="/">
```

1.

### **Guard Against 404 Errors**:

Configure your server to serve index.html for unmatched routes so the Angular router can handle them. For example, in an NGINX server configuration:

```
location / {
  try_files $uri /index.html;
}
```

# 7. Test the Application

Start the app with:

ng serve

1.

2. Navigate to different routes (/about, /contact) and notice how the content changes without refreshing the page.

### 8. Optional Enhancements

#### 1. Lazy Loading:

Optimize performance by loading modules only when needed. Create feature modules for each section and lazy load them using loadChildren.

#### 2. Route Guards:

Protect specific routes with guards like CanActivate or CanDeactivate.

#### 3. Styling Frameworks:

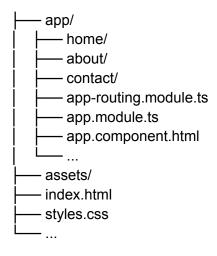
Use Angular Material, Bootstrap, or Tailwind CSS for pre-designed UI components.

#### 4. Animations:

Add transitions between routes using Angular's animation module.

### **Example Directory Structure**

src/



This guide sets up a basic SPA with Angular. Let me know if you need help with advanced features, like lazy loading, forms, or animations!