**1. Create a Feature Module**

* Use the Angular CLI to generate a new module:

Bash

ng generate module products --module=app

This will create a new module named ProductsModule within the app folder.

**2. Create Component(s) for the Feature Module**

* Generate components for your feature module:

Bash

ng generate component product-list --module=products

ng generate component product-detail --module=products

**3. Configure Routing within the Feature Module**

* Create a products-routing.module.ts file within the products folder:

TypeScript

import { NgModule } from '@angular/core';

import { RouterModule, Routes } from '@angular/router';

import { ProductListComponent } from './product-list/product-list.component';

import { ProductDetailComponent } from './product-detail/product-detail.component';

const routes: Routes = [

{ path: '', component: ProductListComponent },

{ path: ':id', component: ProductDetailComponent }

];

@NgModule({

imports: [RouterModule.forChild(routes)], // Use forChild() for lazy-loaded modules

exports: [RouterModule]

})

export class ProductsRoutingModule {}

**4. Import and Configure the Feature Module in AppRoutingModule**

* In app-routing.module.ts:

TypeScript

import { NgModule } from '@angular/core';

import { RouterModule, Routes } from '@angular/router';

const routes: Routes = [

{

path: 'products',

loadChildren: () => import('./products/products.module').then(m => m.ProductsRoutingModule)

},

// ... other routes

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule {}

* loadChildren: This function dynamically loads the ProductsModule only when the user navigates to the /products route.

**5. (Optional) Preload Strategies**

* You can configure preloading strategies to improve the user experience:

TypeScript

RouterModule.forRoot(routes, {

preloadingStrategy: PreloadAllModules // Preload all lazy-loaded modules

})

* Other preloading strategies are available, such as PreloadModules (preloads modules based on route configuration) and custom preloading strategies.

**Key Benefits of Lazy Loading:**

* **Improved Initial Load Time:** Only necessary modules are loaded initially, reducing the initial bundle size and improving the application's startup time.
* **Better Performance:** Reduces memory usage and improves overall application performance.
* **Smaller Bundle Sizes:** Smaller initial bundle sizes lead to faster downloads, especially on slower connections.

By implementing lazy loading, you can significantly improve the performance and user experience of your Angular applications, especially for larger and more complex applications.