Utilisation d’un fichier JSON dans Angular (Standalone)

1. Ajout du fichier JSON

Placer le fichier `communes.json` dans le dossier :
→ Fichier : src/assets/data/communes.json
Il doit contenir les données structurées par province (Nord, Sud, Iles) avec les informations des communes.

2. Création du modèle TypeScript

→ Fichier : src/app/models/commune.ts

À quoi ça sert :
Le modèle TypeScript sert à définir la structure des objets de type Commune. Cela permet à Angular de connaître le format attendu des données, ce qui facilite l’auto-complétion, la vérification de type et la maintenance du code.

3. Création du service Angular

→ Fichier : src/app/services/commune.service.ts

À quoi ça sert :
Le service sert à centraliser l’accès aux données du fichier JSON et à les rendre disponibles à tous les composants de l’application. Il permet de faire une requête HTTP GET pour charger le fichier JSON placé dans `assets`.

Comment ça marche :
- Angular utilise HttpClient (fourni par @angular/common/http) pour effectuer des appels réseau.
- HttpClient envoie une requête GET vers `/assets/data/communes.json`
- Le JSON est automatiquement transformé en objet TypeScript grâce à la typage `Observable<{ [province: string]: Commune[] }>`
- Le résultat est consommé dans un composant avec `.subscribe(...)`

Configuration nécessaire :
Pour utiliser HttpClient, il faut importer `HttpClientModule` dans `app.config.ts` ou `main.ts` (si Angular standalone), par exemple :
`import { provideHttpClient } from '@angular/common/http';`

4. Utilisation dans un composant

Comment le faire :
- Injecter le service dans le composant via le constructeur
- Appeler `this.communeService.getCommunes().subscribe(...)` dans `ngOnInit()` pour

récupérer les données
- Stocker les données dans une variable et les afficher dans le HTML

5. Exemple d’affichage HTML avec Bootstrap

Comment le faire :
- Utiliser des classes Bootstrap comme `container`, `row`, `col`, `card`, etc.
- Boucler sur les données avec `*ngFor`
- Afficher les propriétés avec `{{ commune.Nom }}`, `{{ commune.Population }}`, etc.

6. Configuration nécessaire

Comment faire :
- S’assurer que le fichier JSON est bien placé dans le dossier `assets`
- Ajouter `provideHttpClient()` dans `app.config.ts` ou `main.ts`
- Importer `HttpClient` dans le service
- Utiliser `Observable` et `subscribe` pour consommer les données dans les composants