Inicio proyecto sin haber instaldo del modo SSR

2-instalamos tailwindcss

<https://tailwindcss.com/docs/guides/angular>

3 -**Server-side rendering**

<https://angular.dev/guide/ssr>

4- Prerendering (SSG)

<https://angular.dev/guide/prerendering>

5- Prerendering - Static Site Generation – SSG

<https://angular.dev/guide/prerendering>

1. creamos el archivo routes.txt  
   Texto

   Descripción generada automáticamente con confianza baja

con la las rutas dinámicas de nuestra aplicación.  
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente



Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

5- Texto

Descripción generada automáticamente

Nos crea las paginas existentes en nuestra aplicación:  
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

6- construir el archivo routes.txt de manera automáticamente  
**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

Creamos el archivo: prerender-routes.js

*const* TOTAL\_POKEMONS = 151;  
*const* TOTAL\_PAGES = 5;  
  
  
( *async* () => {  
  
 *const fs* = *require*( 'fs' );  
  
 *// Pokémons por Ids  
 const* pokemonIds = Array.from( { length: TOTAL\_POKEMONS }, ( \_, i ) => i + 1 );  
 *let* fileContent = pokemonIds.map(  
 id => `/pokemons/${ id }`  
 ).join( '\n' );  
  
  
 *// Paginas de Pokémons  
 for* ( *let* index = 1; index <= TOTAL\_PAGES; index++ ) {  
 fileContent += `\n/pokemons/page/${ index }`;  
 }  
  
  
 *// Por nombres de Pokémons  
 const* pokemonNameList = *await fetch*( `https://pokeapi.co/api/v2/pokemon?limit=${ TOTAL\_POKEMONS }` )  
 .then( res => res.json() );  
  
 fileContent += '\n';  
 fileContent += pokemonNameList.results.map(  
 pokemon => `/pokemons/${ pokemon.name }`  
 ).join( '\n' );  
  
  
  
 *fs*.*writeFileSync*( 'routes.txt', fileContent );  
  
 console.log( 'Routes.txt Generated' );  
  
} )();

**y en mi archivo **

**Texto

Descripción generada automáticamente**