# Integración angular 6 con la página ASP.NET ASPX

integrar Angular 6 dentro de su proyecto ASP.NET existente donde sus páginas están en .aspx.

Configuración de Angular 6 en una nueva máquina.

Paso 1 - Instalar NodeJS

Puede instalar node.js y npm desde la siguiente URL.

*https://nodejs.org/en/download/*

Para Angular 6, debe instalar Node8.xy después de una instalación exitosa, puede verificar la versión: node -v desde el símbolo del sistema.

Use npm -v para verificar la versión de NPM. NPM debe ser 5.X

Paso 2: luego, instale la CLI angular globalmente.

Comando para instalar,

*npm install -g @ angular / cli*

*comando para verificar la versión:*
*tsc -v*

A continuación se muestra el código .aspx donde estoy usando <app-root name = "list-user"> </app-root> y agregué referencias de los archivos JS.

```
1.<%@ Page Language="C#" AutoEventWireup="true" CodeFile
="UserList.aspx.cs" %>
2.
3.<!DOCTYPE html>
4.
5.<html xmlns="http://www.w3.org/1999/xhtml">
6.<head runat="server">
7.  <title></title>
8.  <base href="/">
9.</head>
10.<body>
11.  <form id="form1" runat="server">
12.  <div>
13.  <app-root name="list-user"></app-root>
```

```
<script src="../AngularApp/dist/my-new-angular-app/main.js"></script>
<script src="../AngularApp/dist/my-new-angular-app/polyfills.js"></script>
<script src="../AngularApp/dist/my-new-angular-app/runtime.js"></script>
<script src="../AngularApp/dist/my-new-angular-app/styles.js"></script>
<script src="../AngularApp/dist/my-new-angular-app/vendor.js"></script>
</div>
</form>
</body>
</html>
```

El archivo app.component.ts donde el código de la directiva raíz de la aplicación -

```
import { Component, Input, ElementRef } from '@angular/core';

import { Router, ActivatedRoute } from '@angular/router';


@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']
})


export class AppComponent {
  @Input('name') name: string;
  title = 'app';

  constructor(private router: Router, private route: ActivatedRoute, private eleRef: ElementRef) { }

  ngOnInit(): void {
    let prop = this.eleRef.nativeElement.getAttribute('name');
    switch (prop) {
      case "testmodule":
        this.router.navigate(['/testAJ.aspx']);
        break;
      case "list-user":
        this.router.navigate(['/UserList.aspx']);
        break;
      default:
        this.router.navigate(['/']);
        break;
```

```
    }
  }
}
```

**app-routing.modules.ts**

Ahora, cree una matriz de rutas donde hayamos definido qué componente se llamará y luego registre esto en el Módulo raíz. Ahora, puede generar este componente de enrutamiento a través del <router-outlet> component.Module.

```
import { NgModule, ModuleWithProviders  } from '@angular/core';
import { RouterModule, Routes  } from '@angular/router';

import { ListUserComponent } from './listModule/list-user.component';

const routes: Routes = [
  { path: 'testAJ.aspx', component: TestModuleComponent },
  { path: 'UserList.aspx', component: ListUserComponent }
];
export const routing = RouterModule.forRoot(routes);
```

Agregue esto en app.component.html,

```
<router-outlet></router-outlet>
```

clase del modelo:

```
export class State {
  stateID: string;
  stateName: string;
}
```

Clase del servicio:

```
import { Injectable } from '@angular/core';
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { State } from "../model/state.model";

let httpOptions = {
  headers: new HttpHeaders({
    'Content-Type': 'application/json',
    'Cache-Control': 'no-cache'
  })
};

@Injectable()
```

```
13.export class UserService {
14.  constructor(private http: HttpClient) { }
15.  baseUrl: string = 'put your Api here'
16.
17.  getSatesByCountryCode(_countryCode: string) {
18.    let body = JSON.stringify({ CountryCode: _countryCode }
);
19.    return this.http.post(this.baseUrl + 'GetStatebyCountry
Id', body, httpOptions);
20.  }
21.}
```

El archivo list-user.component.ts:

```
import { Component, OnInit } from '@angular/core';
```

```
1.import { Router } from "@angular/router";
2.import { UserService } from "../service/user.service";
3.import { State } from "../model/state.model";
4.import { products } from '../model/products';
5.
6.@Component({
7.  selector: 'list-user',
8.  templateUrl: './list-user.component.html'
9.})
10.
11.export class ListUserComponent {
12.  states: any = [];
13.
14.  constructor(private router: Router, private userService:
UserService) { }
15.
16.  ngOnInit() {
17.    this.userService.getSatesByCountryCode('001')
18.      .subscribe(data => {
19.        this.states = data;
20.        this.states = JSON.parse(this.states.d);
21.      }, error => {
22.        console.log(error);
23.      }
24.    );
25.  }
26.}
```

El archivo list-user.component.html:

<table>

```
1.  <thead>
2.    <tr>
3.      <th>State ID</th>
4.      <th>State Name</th>
```

```
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let state of states">
      <td>{{state.StateID}}</td>
       <td>{{state.StateName}}</td>
    </tr>
  </tbody>
</table>

```

Conclusión

Esta es la explicación de como llamar a la API desde el servicio y vincular los datos a la página HTML con el uso de Angular 6 y ASP.NET.

A continuación se muestra la captura de pantalla de salida en el navegador.

| State ID | State Name |
|---|---|
| 00121 | 345452 |
| 00104 | ACT |
| 00119 | MELBURN |
| 00122 | NORTHAN |
| 00103 | NT |
| 00102 | QLD |
| 00106 | SA |
| 00120 | SFBASFB |
| 00118 | SOUTH SYDNEY |
| 00107 | TAS |
| 00101 | VIC |
| 00108 | WA |