**Lesson30 Lazy loading in angular**

**Notes:-**

**1-When the project expand , every time that the user navigate from page to another component the browser load all the modules and component , so the performance issue is appear , the solution by using the lazy loading**

**2-lazy loading: - it’s using to load component as demand, so it’s loading the specific feature module not all the feature modules.**

**3-we have to know when import feature module to app-routing.module , it’s contain all the modules and components in the app-routing.module**

**Rules of the lazy loading:-**

**1-all the Routing should have the same prefix**

**2-the feature module not import in the app.module**

**Steps:-**

**1-In the emp-routing-module.ts we clear the children prefix**

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

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

import { EmpEntryCompComponent } from './emp-entry-comp/emp-entry-comp.component';

import { EmpLstCompComponent } from './emp-lst-comp/emp-lst-comp.component';

const routes: Routes = [

{ path: '', component: EmpLstCompComponent },

{ path: 'create', component: EmpEntryCompComponent },

{ path: 'edit/:id', component: EmpEntryCompComponent },

{ path: '', redirectTo: '/list', pathMatch: 'full' }

]

// wild card route

@NgModule({

imports: [RouterModule.forChild(routes)],

exports: [RouterModule]})

export class EmpRoutingModule { }

**2-In the app-routing.module.ts we apply the lazy loading on the specific feature module**

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

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

import { PageNotFoundCompComponent } from './components/page-not-found-comp/page-not-found-comp.component';

import { HomeComponentComponent } from './components/MainComponent/home-component/home-component.component';

const routes: Routes = [

{ path: 'home', component: HomeComponentComponent },

{ path: '', redirectTo: '/home', pathMatch: 'full' },

//In the below command we apply load Children to specific feature module to //apply the lazy loading

{ path: 'employees', loadChildren: './components/Employee/employee.module#EmployeeModule' },

//we have to put wild card route in the last route

{ path: '\*\*', component: PageNotFoundCompComponent }];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]})

export class AppRoutingModule { }

**3-In the app.module.ts we remove the feature module from it**

import { BrowserModule } from '@angular/platform-browser';

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

import { FormsModule,ReactiveFormsModule} from '@angular/forms';

import { HttpClientModule } from '@angular/common/http';

import { AppRoutingModule } from './app-routing.module';

import { EmpServiceService } from './Services/emp-service.service';

import { AppComponent } from './app.component';

import { MenuCompComponent } from './components/menu-comp/menu-comp.component';

import { PageNotFoundCompComponent } from './components/page-not-found-comp/page-not-found-comp.component';

import { HomeComponentComponent } from './components/MainComponent/home-component/home-component.component';

@NgModule({

declarations: [

AppComponent,

MenuCompComponent,

PageNotFoundCompComponent,

HomeComponentComponent],

imports: [

BrowserModule,

FormsModule,

ReactiveFormsModule,

HttpClientModule,

AppRoutingModule

],

providers: [EmpServiceService],

bootstrap: [AppComponent]

})

export class AppModule { }

**4-In the menu-comp.html we modify the following html part**

<div class="container">

<nav class="navbar navbar-default">

<ul class="nav navbar-nav">

<li><a routerLinkActive="active" routerLink="home">Home</a></li>

<li><a routerLinkActive="active" routerLink="employees">List</a></li>

<li><a routerLinkActive="active" routerLink="employees/create">Create</a></li>

</ul></nav>

<router-outlet></router-outlet>

</div>

**5-In the emp-lst.com.ts we modify the following**

editButtonClick(employeeId:Number){

this.\_route.navigate(['employees/edit', employeeId]);}

**6-In the emp-entry.com.ts we modify the following**

SaveEmployee(): void {

this.mapFormValuesToEmployeeModel();

this.empService.addEmployee(this.empService.employee).subscribe(

() => this.router.navigate(['employees']),

(err: any) => console.log(err));}

//The Following method it will call the service method to call the web api //service method to update employee

UpdateEmployee(): void {

this.mapFormValuesToEmployeeModel();

this.empService.updateEmployee(this.empService.employee).subscribe(

() => this.router.navigate(['employees']),

(err: any) => console.log(err));}

**Additional Notes:-**

**1-**As you add more features to your application you will have more feature modules like ReportsModule, AdminModule etc. As you add more feature modules, the overall application size will continue to grow. At some point you'll reach a tipping point where the application takes a very long time to load.

**2-Without lazy loading all the modules in your angular application and their associated components, directives, pipes and services must be downloaded from the server every time you navigate from page to another.**