**Angular**

Angular is a popular open-source web application framework developed and maintained by Google. It is written in TypeScript and is designed to simplify the process of building dynamic, single-page web applications (SPAs). Angular provides a comprehensive set of tools and features for developing client-side applications, including:

1. **Modular Architecture:** Angular applications are built using a modular architecture, with components being the basic building blocks. Components encapsulate the application's functionality and can be reused throughout the application.
2. **Two-Way Data Binding:** Angular offers two-way data binding, which means that changes in the user interface (such as input fields) automatically update the application's underlying data, and vice versa.
3. **Dependency Injection:** Angular has a powerful dependency injection system that helps manage the components' dependencies and promotes the creation of modular and maintainable code.
4. **Directives:** Directives are markers on a DOM element that tell Angular to attach a specific behavior to that element or transform it in a certain way. Angular comes with a set of built-in directives, and you can also create custom directives.
5. **Services:** Angular allows you to create and use services to organize and share code across components. Services are often used for tasks such as fetching data from a server, logging, or handling application-wide functionality.
6. **Routing:** Angular provides a powerful routing module that enables developers to build single-page applications with multiple views. This allows for a more structured and organized application flow.
7. **Forms:** Angular offers a robust forms module that simplifies the process of handling user input and form validation.

**Creating Angular Project**

**1.Install Node js**

**2.Check the version of node and npm**

node –v

npm –v

**Install Angular CLI:** Open a terminal or command prompt and run the following command to install the Angular CLI globally:

**npm install -g @angular/cli@16.0.0**

**Create new project:**ng new your-project-name

**Navigate to the Project Directory:** Change into the project directory:

cd your-project-name  
**Serve the Angular Application:** ng serve

**Create Componenet**

*ng generate component component-name*  
After running the command, the Angular CLI will output something like this, indicating the files it created:   
CREATE src/app/component-name/component-name.component.html (...)  
CREATE src/app/component-name/component-name.component.ts (...)  
CREATE src/app/component-name/component-name.component.scss (...)  
CREATE src/app/component-name/component-name.component.spec.ts (...)  
UPDATE src/app/app.module.ts (...)

**Component-name.component.html**

**Samplecode :**

**<p>home works!</p>**

<form action="">

    <div>

       <div>

        <label for="">Name</label>

       </div>

        <input type="text">

    </div>

   <div>

    <input type="text">

   </div>

</form>

**app.componenet.html**

<h1>Hello</h1>

<app-home></app-home>------------>This is delector of created component ie,

**component-name.componenet.ts**

**import { Component } from '@angular/core';**

@Component({

  selector: 'app-home',

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

  styleUrls: ['./home.component.css']

})

export class HomeComponent {

}

**Angular Expression**  
which is used to bind data and manipulate the DOM (Document Object Model) within an Angular application. Expressions are written within double curly braces (**{{ }}**)

**Example:**<p>My First expresion {{5+5}}</p>

**Angular Routing**

Routing to navigate between different components in your application. Routing is typically configured in the app-routing.module.ts file.

**1.create new project**

**2.create component:ng g c department-list-it-is**

**3.create componenet:ng c employee-list-it-is**

**4.app-routing.module.ts**

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

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

import { DepartmentListItIsComponent } from './department-list-it-is/department-list-it-is.component';

import { EmployeeListItIsComponent } from './employee-list-it-is/employee-list-it-is.component';

###############PATH OF EMPLOYEE AND DEPARTMENT

const routes: Routes = [

  {path:'department',component:DepartmentListItIsComponent},

  {path:'employee',component:EmployeeListItIsComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

###TO ROUTER###

export const routingComponents=[DepartmentListItIsComponent,EmployeeListItIsComponent]

5.**Add routingComponent to app.modules.ts**

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

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

###ADD routingComponenet

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

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

import { DepartmentListItIsComponent } from './department-list-it-is/department-list-it-is.component';

import { EmployeeListItIsComponent } from './employee-list-it-is/employee-list-it-is.component';

@NgModule({

  declarations: [

    AppComponent,

    DepartmentListItIsComponent/routingComponents,####THIS IS REPLACE By routingComponenet

    EmployeeListItIsComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

**6 Remove all data from app.componenet.html Add this code**

<div style="text-align: center;">

  <h1>Routing</h1>

  <nav>

    <button><a routerLink="/department">Department</a></button>

    <button><a routerLink="/employee">Employee List</a></button>

  </nav>

</div>

<router-outlet></router-outlet>

**Angular Directives**

Directives are a type of attribute or element that you can use to extend or manipulate the behavior of elements in the DOM (Document Object Model).

**Structural Directives:** Attribute directives modify the appearance or behavior of an element. They are applied to elements as attributes and are prefixed with **\*ng** in Angular. Common attribute directives include **\*ngIf**, **\*ngFor**, and **\*ngSwitch**.

**NgFor Example:**

**create a componenet.**

**product-componenet.html**

<div \*ngFor="let item of [1,3,5]">

    <p>Number is:{{item}}</p>

</div>

**app-componenet.html**

<div style="text-align: center;">

  <h1>Routing</h1>

  <nav>

    <button><a routerLink="/department">Department</a></button>

    <button><a routerLink="/employee">Employee List</a></button>

    <button><a routerLink="/product">Product</a></button>####Product page

  </nav>

</div>

<button (click)="getData('button clicked')">Click</button>

<br>

<br>

<input type="text">

<div>

</div>

<router-outlet></router-outlet>

**app-routing.module.ts**

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

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

import { DepartmentListItIsComponent } from './department-list-it-is/department-list-it-is.component';

import { EmployeeListItIsComponent } from './employee-list-it-is/employee-list-it-is.component';

import { ProductComponent } from './product/product.component';

const routes: Routes = [

  {path:'department',component:DepartmentListItIsComponent},

  {path:'employee',component:EmployeeListItIsComponent},

  {path:'product',component:ProductComponent},######product path######

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

export const routingComponents=[DepartmentListItIsComponent,EmployeeListItIsComponent]

**Angular Sevices And \*forlopp**

Services are a way to organize and share code across different parts of your application. Angular services are singleton objects that are instantiated only once during the lifetime of an application.

**Note:**Reuse the code for multiple componenet

create a service folder:**ng generate service services/userdata**

**two folder created**

**1.userdata.service.spec.ts**

**2.userdata.service.ts**

**userdata.service.ts**

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

@Injectable({

  providedIn: 'root'

})

export class UserdataService {

  constructor() { }

  user(){

    return[

      {name:'peter',age:17,email:'peter@gmail.com'},

      {name:'sam',age:27,email:'sam@gmail.com'},

      {name:'raju',age:37,email:'raju@gmail.com'},

    ]

  }

}

**create componenet footer**

**app.componenet.html**

<h1>{{title}}</h1>

<ul>

    <li \*ngFor="let user of users">

    {{user.name}} ------------{{user.age}}

</li>

</ul>

<app-footer></app-footer>

**app.componenet.ts**

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

import { UserdataService } from './services/userdata.service';

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'services';

  users: any;

  constructor(private userData: UserdataService) {

    console.warn(userData.users());

    this.users=userData.users()

  }

}

**Angular Event**

**create new project Event**

**app.componenet.ts**

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'event';

  getData(val:string)

  {

    console.warn(val);

  }

}

**app.componenet.html**

<!-- app.component.html -->

<!-- app.component.html -->

<h1>{{ title }}</h1>

<button (click)="getData('Button clicked!')">Click Me</button>

**BUTTON CLICK:**

**app.componenet.ts**

// app.component.ts

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  public count = 0; ####first count##

  title = 'event';

  getData(val: string) {

    console.warn(val);

  }

##############BUTTON CLICK#########

  increment() {

    this.count++;

  }

  decrement(){

    this.count--;

  }

}

**app.componenet.html**

<div>

  <div style="text-align: center;">

    <h2>Count: {{ count }}</h2>

    <div>

      <button (click)="increment()">Increment</button>

      <button (click)="decrement()">Decrement</button>

    </div>

  </div>

</div>

**Angular DataBinding**

Angular supports several types of data binding, which is a powerful feature that allows you to synchronize data between your component and the template.

**1. Interpolation (One-way binding):**

* Interpolation is a one-way binding from the component to the view (template).
* It allows you to embed expressions within double curly braces ({{ }}) in the template.

**// Component**

export class MyComponent {

public message = 'Hello, Angular!';

}

**<!-- Template -->**

<p>{{ message }}</p>

**Two-way Binding:**

* Two-way binding combines property binding and event binding into a single syntax using ngModel.
* It allows data to flow in both directions between the component and the view.

**App.componenet.html**

<h2 style="text-align: center;"> App Component Data Binding</h2>

<br>

<div style="text-align: center;">

  <input type="number" [(ngModel)]="count">

  <button (click)="check()">Check Count</button>

  <button (click)="clear()">Clear</button> <!-- Corrected the syntax -->

  <div>

    <h1 [hidden]="is\_hidden">{{ count }}--{{ str }}</h1>

  </div>

</div>

* This input element is of type "number" and is using two-way data binding **([(ngModel)]="count")** to bind its value to the count property in the associated TypeScript component.
* The **[hidden]="is\_hidden"** attribute binding is used to conditionally hide or show the <h1> element based on the value of the is\_hidden property. If **is\_hidden is true,** the element is hidden; otherwise, it is visible.

**App.componenet.ts**

// app.component.ts

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  public count: number = 0;

  public str='';

  public is\_hidden=true

  check() {

    this.is\_hidden=false

    if (this.count % 2 === 0) {

      this.str='Even'

      // Add any additional actions or logic for even count

    } else {

      this.str='Odd'

      // Add any additional actions or logic for odd count

    }

  }

clear(){

  this.is\_hidden=true

}

}

app.module.ts

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

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

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

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

@NgModule({

  declarations: [

    AppComponent,

    // Other components, directives, or pipes

  ],

  imports: [

    BrowserModule,

    FormsModule,

  ],

  bootstrap: [AppComponent],

})

export class AppModule {}

**Angular Dependency Injection`**