Standalone components are Angular components that do not require being declared in an Angular module. They can be used independently and are designed to simplify the development process by reducing the need for extensive module management.

From the standalone components whatever imports we need, we can import in the ts file in the import array inside the **component decorator** as shown below. @Component section is called as component decorator.A screen shot of a computer screen

Description automatically generated Selector: Selector is something by which we can call or render a component into another component. Now suppose we want to render role component in the app component, so what we need to do. first import the role component in the app.component.ts file. A screen shot of a computer program

Description automatically generated

Interpolation: {{Variable\_Name}} like this we can bind directly by interpolation.

A black screen with white text

Description automatically generatedA screen shot of a computer program

Description automatically generated

Now once save it, we can see data is rendered in the browser as shown below.

A white background with black text

Description automatically generated

Now let’s bind other variable as well.

A screen shot of a computer program

Description automatically generated

A close up of a card

Description automatically generated

Now let's install the bootstrap. Go to the <https://www.npmjs.com/package/bootstrap> and search the bootstrap. It will provide the NPM command like **npm i bootstrap**

Open a new terminal and paste and enter.

So it will add the packages in the package.json

A screen shot of a computer

Description automatically generated

Once we installed the bootstrap or anything external libraries, we have to register that into the angular.json (angular.json) is our architectural file. So in angular.json file we have to add in the style section as shown below.

A screen shot of a computer

Description automatically generated

Now whenever we do any changes in angular.json file, we need to re-run the application.

Now we can verify whether our bootstrap classes are working correctly or not.

A screen shot of a computer program

Description automatically generated

We can see our bootstrap classes are working correctly. As shown below.

A white background with black dots

Description automatically generated

Data Binding:

1) Interpolation Binding

2) Property Binding

3) Two way binding

When we use interpolation, when inside any element we just need to print the data.

But if want to provide a value to any property so we will wrap that property into square [] bracket.

A screenshot of a computer screen

Description automatically generated

Currently if we change the value in the textbox, the value is not getting changed in the firstname but we want like if any changes are done in the textbox that value should be changed (assigned ) to the firstname.

So, to achieve the two way binding, first we need to import the form module in ts file in the import section.

A screen shot of a computer program

Description automatically generated

So now when we do any changes in the textbox that value is changing everywhere (wherever we have used firstname variable it will be changed.

A screen shot of a computer program

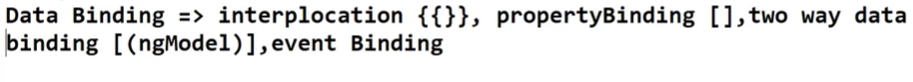
Description automatically generated

**Event Binding:**Html and ts file.A screen shot of a computer code

Description automatically generated

A screen shot of a computer program

Description automatically generated



Directives: CommonModule will contain the directives, we need to import it in the project where we are using.

A screen shot of a computer program

Description automatically generated

Structural directive will start from the \*. Like in below example, we are using \*ngif to show component in the browser based on condition.

A screen shot of a computer code

Description automatically generated

Similarly we can add the css class dynamically



A screen shot of a computer program

Description automatically generated

Before we make any API call, app.config.ts we have to import the

provideHttpClient like as below

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

A screen shot of a computer

Description automatically generated

Calling get API method as shown below.

A screen shot of a computer program

Description automatically generated

\*ngFor we have to use when we have iterate some elements. like foreach loop

A screen shot of a computer code

Description automatically generated

We can bind the role data as shown below

A computer screen with many white and blue text

Description automatically generated

In browser, we can see the data as shown below.

A screenshot of a computer

Description automatically generated

In API response, we are getting the data so we can create the interface and bind instead of any (any is a dynamic one in type script)

A screenshot of a computer program

Description automatically generated

Now in roles ts file we can modify

A screen shot of a computer program

Description automatically generated

**Services:** normally we don't make API call in the component, services are nothing but a class where we can keep our reusable code or code, or function which we are going to use in our overall the project.

So we are going to create a service as shown below. (command ng g s service\_name)

A screen shot of a computer

Description automatically generated

We have created a service and created a method to get all the designation.

We have set the return type as well as shown above screenshot.

This service we can call in our designation ts file. We can call the method as shown below assign the data to the designationList.

A screen shot of a computer program

Description automatically generated

Designation html file

A screen shot of a computer code

Description automatically generated

Data looks in UI

A screenshot of a computer

Description automatically generated

Update the routes in app.routes.ts file.

Below very first, if path is blank (“”) then it should be redirected to master. If path is master then it should be rendered our master component.

A screen shot of a computer

Description automatically generated

**Control flow statement:** using @if or @else or @for like we have used in below code, those are called as control flow statements.

A screen shot of a computer program

Description automatically generated

Isleader a variable we have set while API call and it looks like as below.

A screen shot of a computer program

Description automatically generated

**Template Variable:** here we have created the template variable for the personname (#personName = “ngModel”)and we have applied the required and minimum length validations.

A computer screen shot of a computer code

Description automatically generated

**Template Form:** In Angular, a **template-driven form** is a way to create forms using HTML and Angular's built-in directives. This approach focuses on simplicity and minimal code, using Angular's ngModel directive to bind data between the form and the component. Template form we achieve by using the Import **FormsModule.**

A screen shot of a computer program

Description automatically generated

In HTML form, we use the ng model for validation and we use Angular's ngModel directive for two-way data binding.

A computer screen shot of text

Description automatically generated

**Reactive Form:** A **reactive form** in Angular is more programmatic and provides greater control over form state and validation. Unlike template-driven forms, reactive forms are defined in the component class, using Angular's FormGroup, FormControl, and Validators for form control and validation logic.

We import the **ReactiveFormsModule** to create a reactive form.

A screen shot of a computer code

Description automatically generated

In HTML, we use the formControlName as shown below.

A screen shot of a computer program

Description automatically generated

**FormGroup**: Represents the entire form.

**FormControl**: Represents individual form controls.

**Validators**: Built-in validators like required and email are used to validate inputs.

**[formGroup]**: Binds the form model (userForm) to the form element.

**formControlName**: Binds individual controls to their respective form fields.

**Pipe -** Pipe is used to format the data in the HTML.

There are some default pipe and we can create our own custom pipe. a **pipe** is a simple way to transform data directly in the template. It allows you to modify the display of data without needing to modify the underlying data.

In angular 18, we have to import the pipe as shown below. Like UpperCasePipe, DatePipe etc.

A screen shot of a computer program

Description automatically generated

Then we can use the pipe like as shown below.

A screen shot of a computer program

Description automatically generated

All name will be in upper case as we can see below.A screenshot of a computer

Description automatically generated

Similar way, when we are printing the date, we can use like below, we can give the format.

A screen shot of a computer

Description automatically generated



**Json Pipe:** **json pipe** in Angular is a built-in pipe that is used to convert an object or array into a JSON-formatted string. This is particularly useful for debugging or displaying structured data in a readable format in the template.

In our client TS, we have an object and if we try to print this in our client HTML we will see output as shown below. Object object.

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generatedA green and white rectangular object with black text

Description automatically generated

But if we want to see what things are there in the object, we can use JSONPipe.

So let’s add first JsonPipe and then add the pipe in the paragraph.

A screen shot of a computer program

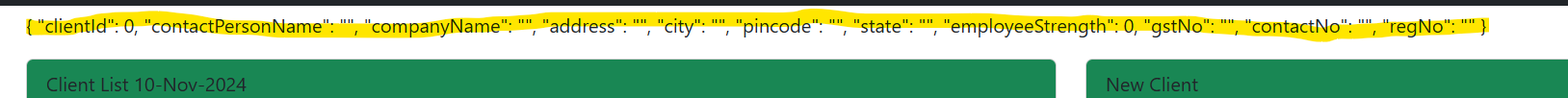
Description automatically generated

Once we added the import, now we can use jsonPipe and see the data which is inside the object.

A screen shot of a computer

Description automatically generated

We can see the Json object and the data inside the object.



**Async Pipe:** Generally we write and API call, and we subscribe, then store the output data in a variable and then that variable we use it in the HTML but instead of that we can directly subscribe the response from that API call in the HTML only by using async pipe.

The **async pipe** in Angular is a powerful built-in pipe that simplifies working with asynchronous data in templates. It automatically subscribes to an **Observable** or **Promise** and returns the latest value from it. When the data changes, the async pipe also automatically triggers a re-render of the template.

Now in the client service, we created a service to fetch all the users information.

A screen shot of a computer program

Description automatically generated

Now in Client.ts file we call this in the oninit and we created a observable to store this or hold this data.

A screen shot of a computer program

Description automatically generated

Now in the HTML form, we can loop through the our observable using async pipe.

A screen shot of a computer program

Description automatically generated

In the UI we can see the data.

A screen shot of a computer

Description automatically generated

Below is the API response, so we are printing only username in our HTML. We can use in this way.

A screenshot of a computer

Description automatically generated

**Constant:** we can keep our all URL, and validation messages in constant and then we can access them in the code.

A screen shot of a computer

Description automatically generated

So instead of hard coding in the main ts file, we can access them as shwon below.

A screen shot of a computer code

Description automatically generated

**Signal:** signal is like a creating a variable in angular.

A black background with white text

Description automatically generated

While reading or printing it we have to use it as a method

A screen shot of a computer

Description automatically generated

In UI, we can see we got the value.

A green box with black numbers

Description automatically generated

If we are reading this signal in ts file also we have to read like method only.

A screen shot of a computer program

Description automatically generated

If we want to change the value of this firstname we can do it by using the set method

A screen shot of a computer program

Description automatically generated

**Benefit of using Signal:**

in AppConfig.ts, we have something provideZoneChangeDetection like as below

providers: [provideHttpClient(), provideZoneChangeDetection({ eventCoalescing: true }), provideRouter(routes)]

so to read the change detection, we are not depend on the zone.js now we are using signal.

Whenever the value changes, Angular will only re-render the affected part of the UI, ensuring efficient updates.

**Reusable Component:** like we can use the alert component.

A screen shot of a computer screen

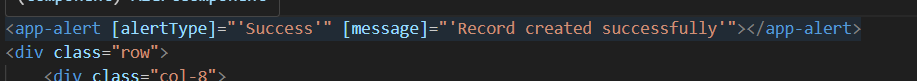
Description automatically generated

And we have two input parameters alertType and message

A screen shot of a computer screen

Description automatically generated

In parent component we can have like as below. Means we can add our selector.



Note: From parent we can send the data as input to the child component

@Input() alertType: string = ''

@Input() message : string = ''

So if we have to send data child to parent then we have to use the output.