



Angular

Tahaluf Training Center 2021









Chapter 5

- 1 Shared Module
- Working on home page
- 3 Input and output



Shared Module



To use module, pipes, services from different modules we will generate a shared module contains all modules, services and pipes which is used from another modules







Example:

```
PS C:\Users\User\Desktop\angular\portalApp> ng g m shared CREATE src/app/shared/shared.module.ts (192 bytes)
PS C:\Users\User\Desktop\angular\portalApp> [
```







In shared module, import all module and declared all component or pipes you will used more than one module.

```
imports: [
   CommonModule,
   MatFormFieldModule,
   FormsModule,
   ReactiveFormsModule,
   MatInputModule,
   MatButtonModule,
   MatProgressSpinnerModule,
   NgxSpinnerModule
],
```







And you must export these module in export array:

```
exports : [
    MatFormFieldModule,
    FormsModule,
    ReactiveFormsModule,
    MatInputModule,
    MatButtonModule,
    MatProgressSpinnerModule,
    NgxSpinnerModule
]
```







Chapter 5

- 1 Review
- Working on home page
- 3 Input and output





- ➤ Before started, we will generate new module for the client called client module.
- The propose of this module is if the client entered to the website successfully (valid email and password in login page) he will go to home page .





Generate new module called client.

```
PS C:\Users\User\Desktop\portalApp> ng g m client --routing CREATE src/app/client/client-routing.module.ts (249 bytes)
CREATE src/app/client/client.module.ts (280 bytes)
PS C:\Users\User\Desktop\portalApp>
```







Inside client module , will generate new component called home component

```
PS C:\Users\User\Desktop\portalApp> ng g c client/home

CREATE src/app/client/home/home.component.html (19 bytes)

CREATE src/app/client/home/home.component.spec.ts (612 bytes)

CREATE src/app/client/home/home.component.ts (267 bytes)

CREATE src/app/client/home/home.component.css (0 bytes)

UPDATE src/app/client/client.module.ts (356 bytes)

PS C:\Users\User\Desktop\portalApp>
```





Since the auth module is loaded in in app module, client module will loaded in .

```
In auth -routing.module.ts
{
    path:'client',
    loadChildren: ()=> import('../client/client.
module')
    .then((m)=>m.ClientModule)
  }
```





In client -routing.module.ts inside routes array add the default routes.

```
{
    path: '',
    component: HomeComponent
}
```





If the user logged successfully the button submit will navigate it to the home page , so in login.component .ts :

```
submit(){
//Go to Loader
this.spinner.show();
setTimeout(() => {
    this.spinner.hide();
    //go to the home page
    this.router.navigate(['client'])
}, 2000)
}
```







- Now we will add the toolbar from angular material website.
- Add the import module in shared.module .ts :

```
import { MatToolbarModule } from
'@angular/material/toolbar';
```







Import the **MatToolbarModule** in import section :

In shared.module .ts:

```
imports: [
    CommonModule,
    ClientRoutingModule,
    MatToolbarModule
]
```







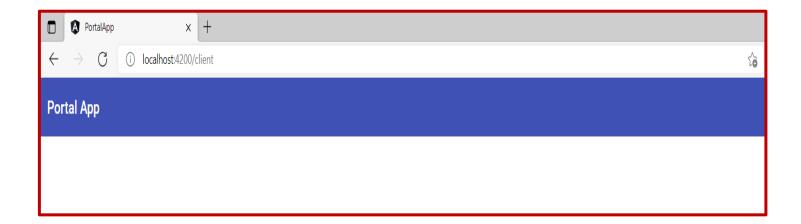
In Home.component.html:

```
<mat-toolbar color="primary">
<mat-toolbar-row>
<span>Portal App</span>
</mat-toolbar-row>
</mat-toolbar>
```





> The result will be:









- We will go to add card in home page, before that we will define the **Reusable component**.
- Reusable component: the component which is created for one time and use in more than one component







Generate a reusable component inside home component called PortalCard .

```
PS C:\Users\User\Desktop\angular\portalApp> ng g c client\home\PortalCard

CREATE src/app/client/home/portal-card/portal-card.component.html (26 bytes)

CREATE src/app/client/home/portal-card/portal-card.component.spec.ts (655 bytes)

CREATE src/app/client/home/portal-card/portal-card.component.ts (294 bytes)

CREATE src/app/client/home/portal-card/portal-card.component.css (0 bytes)

JPDATE src/app/client/client.module.ts (694 bytes)

PS C:\Users\User\Desktop\angular\portalApp>
```





Inside PortalCard component will creates cards for our PoratlApp example so we will go to angular material and import the pakage of cards in shared.module.ts:

```
import { MatCardModule } from
'@angular/material/card';
```

And import MatCardModule in import section.





In PortalCard.component.html :

```
<mat-card class="example-card">
<mat-card-header>
<div mat-card-avatar class="example-header-
image"></div>
<mat-card-title>Front end </mat-card-title>
<mat-card-subtitle>HTML</mat-card-subtitle>
</mat-card-header>
<img mat-card-image
src="https://www.onlinecoursereport.com/wp-
content/uploads/2020/06/shutterstock_153031724-
1024x768.jpg" alt="Photo of a Shiba Inu">
```





```
<mat-card-content>
<</pre>
```

The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript..





In PortalCode.component.css





Call PotalApp component in home component and repeat it eight times Add this code in home .component .html







```
home .component .css
.container {
    width: 98%;
    margin: 0 auto;
    margin-top: 40px;
.cards {
    display: flex;
    flex-direction: row;
    justify-content: space-evenly;
    flex-wrap: wrap;
```





Change the title, sub title and paragraph for each card.

> So, we will sent a permitter from home page component to portalCard components .







Chapter 5

- 1 Review
- Working on home page
- 3 Input and output





Change the home page component to :

<app-portal-card typeLang="Front End"
subText="HTML" description="The HyperText Markup
Language, or HTML is the standard markup language
for documents designed to be displayed in a web
browser. It can be assisted by technologies such as
Cascading Style Sheets (CSS) and scripting languages
such as JavaScript.. "></app-portal-card>





- To call typeLang, subTest and description in PortalCard component.
- Define this variable in typescript file In PortalCard.ts inside class :

```
@Input()typeLang:string | undefined;
@Input()subText:string | undefined;
@Input()description :string|undefined;
@Input()photo: string | undefined
```





In PortalCard.component.html

```
<mat-card-header>
<div mat-card-avatar class="example-header-</pre>
image"></div>
<mat-card-title>{{typeLang}} </mat-card-title>
<mat-card-subtitle>{{subText}}</mat-card-</pre>
subtitle></mat-card-header>
<img mat-card-image</pre>
src="https://www.onlinecoursereport.com/wp-
content/uploads/2020/06/shutterstock_153031724
-1024x768.jpg" alt="Photo of a Shiba Inu">
<mat-card-content>
 {{description}}
 </mat-card-content>
```





OR you can use array to do this, so in home.component.ts

```
export class HomeComponent implements OnInit {
    data: any[] = [{
        typeLang: 'Front End',
        subText: 'HTML',
        description: 'The HyperText Markup Language, or
HTML is the standard markup language for documents
designed to be displayed in a web browser. It can be
assisted by technologies such as Cascading Style
Sheets (CSS) and scripting languages such as
JavaScript.. '
    },
```





```
typeLang: "Back End",
   subText: "C#",
    description: " C# (pronounced see
sharp)[b] is a general-purpose, multi-
paradigm programming language encompassing
static typing, strong typing, lexically
scoped, imperative, declarative, functional,
generic, object-oriented (class-based), and
component-oriented programming
disciplines.[15]"
```







```
In home.component .html
<div class=container>
<div class="cards">
<app-portal-card</pre>
[typeLang]="data[0].typeLang"
[subText]="data[0].subText"
description="data[0].description"></app-</pre>
portal-card>
<app-portal-card
[typeLang]="data[1].typeLang"
[subText]="data[1].subText"
description="data[1].description"></app-</pre>
portal-card>
```





- Update home component and use less code using for loop in html
- > Syntax:

<*ngFor="let variable of array-name">



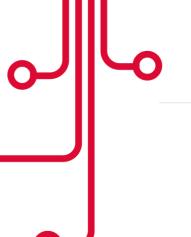




In home.component.html

```
<div class =container>
<div class ="cards">
<app-portal-card *ngFor ="let card of data"
[typeLang]="card.typeLang" [subText]="card
.subText" description="card.description">
</app-portal-card>
</div>
</div>
```







Write Event

If the user click to the Language will navigate to the profile for this language .







Exercise

Generate new component in client module called profile and give it a route







Solution

```
PS C:\Users\User\Desktop\angular\portalApp> ng g c client\profile

CREATE src/app/client/profile/profile.component.html (22 bytes)

CREATE src/app/client/profile/profile.component.spec.ts (633 bytes)

CREATE src/app/client/profile/profile.component.ts (279 bytes)

CREATE src/app/client/profile/profile.component.css (0 bytes)

UPDATE src/app/client/client.module.ts (780 bytes)

PS C:\Users\User\Desktop\angular\portalApp>
```







Solution

In client-routting.module.ts





```
In home .component.ts
goToprofile(){
```

```
this.router.navigate(['profile'])
}
```





Update home component





In portalCards.component.ts add in the class

```
@Output()openProfile=new EventEmitter();
And add this function to call
openprofile method

showCoursePorfile(){
    //call openProfile method();
    this.openProfile.emit();
}
```

