Angular

Tahaluf Training Center   2021

Chapter 3

Moving from one page to another

Since app component are the first components which be loaded when the project has run and you want to go to another page we will use routing in the app component.

Moving from one page to another

First You must use this tag in the app.component.html

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

Generate new component called about us and another component called contact us in app module.

Generate about us component in app module :

```
PS C:\Users\User\Desktop\Training\TrainingWebSite> ng g c aboutus
CREATE src/app/aboutus/aboutus.component.html (22 bytes)
CREATE src/app/aboutus/aboutus.component.spec.ts (633 bytes)
CREATE src/app/aboutus/aboutus.component.ts (279 bytes)
CREATE src/app/aboutus/aboutus.component.css (0 bytes)
UPDATE src/app/app.module.ts (479 bytes)
PS C:\Users\User\Desktop\Training\TrainingWebSite>
```

Moving from one page to another

Generate contact us component in app module:

```
PS C:\Users\User\Desktop\Training\TrainingWebSite> ng g c contactus
CREATE src/app/contactus/contactus.component.html (24 bytes)
CREATE src/app/contactus/contactus.component.spec.ts (647 bytes)
CREATE src/app/contactus/contactus.component.ts (287 bytes)
CREATE src/app/contactus/contactus.component.css (0 bytes)
UPDATE src/app/app.module.ts (573 bytes)
PS C:\Users\User\Desktop\Training\TrainingWebSite>
```

Moving from one page to another

The first page will load is app components and if you want to go to about us page in the same module **(App Module)**

In **app-routing.module.ts ,** There is an array called **routes** this array is used to add the route.

Each route is an object and each route consists of a path
 and component Where :

path: where you want to go for example /aboutus but you must write the page name without using / .

component: component name which you want to display.

Moving from one page to another

```
const routes: Routes = [
    {
        path: 'aboutus',
        component: AboutusComponent
    }
];
```

Note : Once you write the component name it will import this component like this:

```
import { AboutusComponent } from './aboutus/aboutus.component';
```
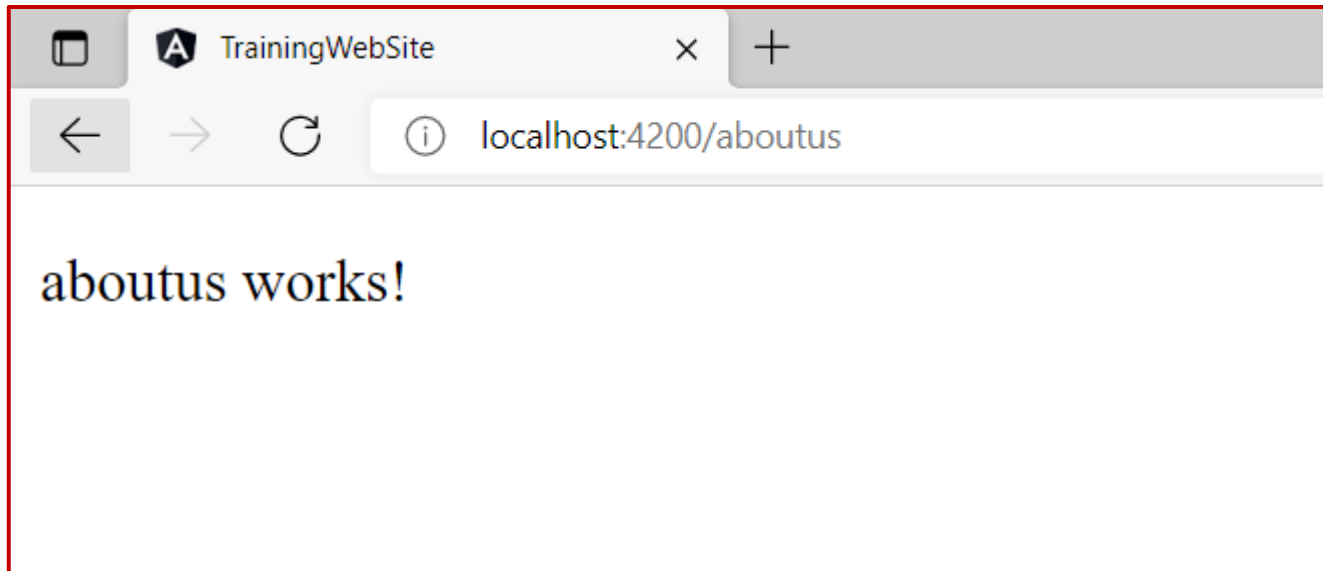
Moving from one page to another

To add more than one route separate them with a comma.

```typescript
const routes: Routes = [
    {
        path: 'aboutus',
        component: AboutusComponent
    },
    {
        path: 'contactus',
        component: ContactusComponent
    }
    ];
```
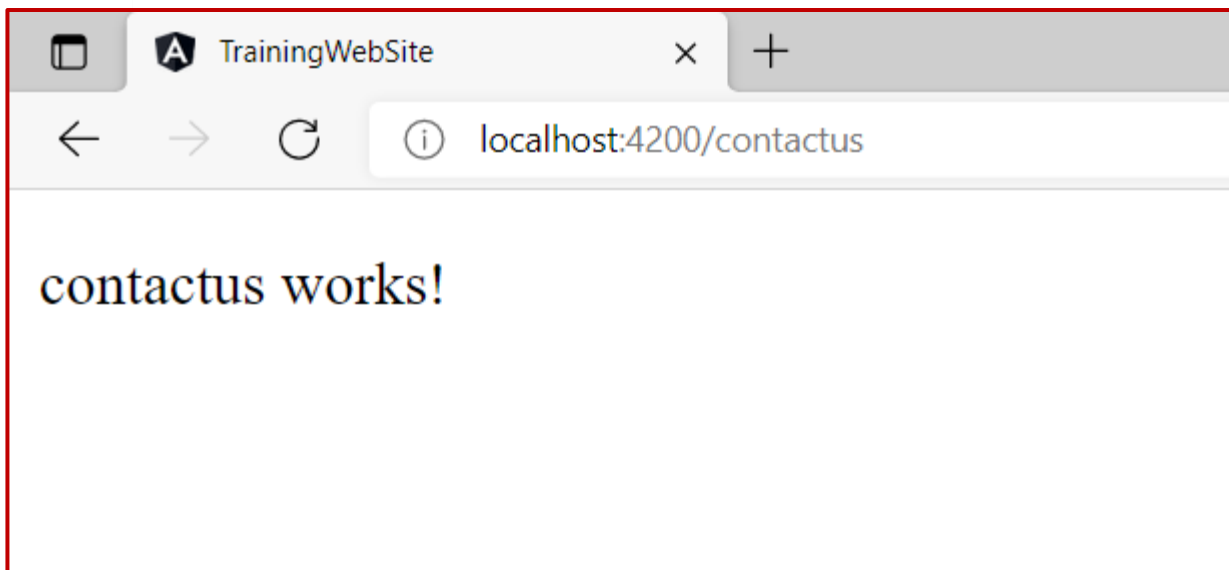
Moving from one page to another

Because about us page in the app module use /aboutus in the URL.

Moving from one page to another

/ contactus

Default route: path of the default route is empty string.

```
const routes: Routes = [
    {
        path: '',
        component: AboutusComponent
    },
]
```

In this way when the project was run about us page will load.

Exercise:

Create Route for auth module to login and register component.

Exercise Solution:

In the auth-routing.module.ts will add routing for login and register .

```typescript
const routes: Routes = [
    {
        path: 'login',
        component: LoginComponent
    },
    {
        path: 'register',
        component: RegisterComponent
    }
];
```

Now If you want to use /login, it won't work because the auth module doesn't load.
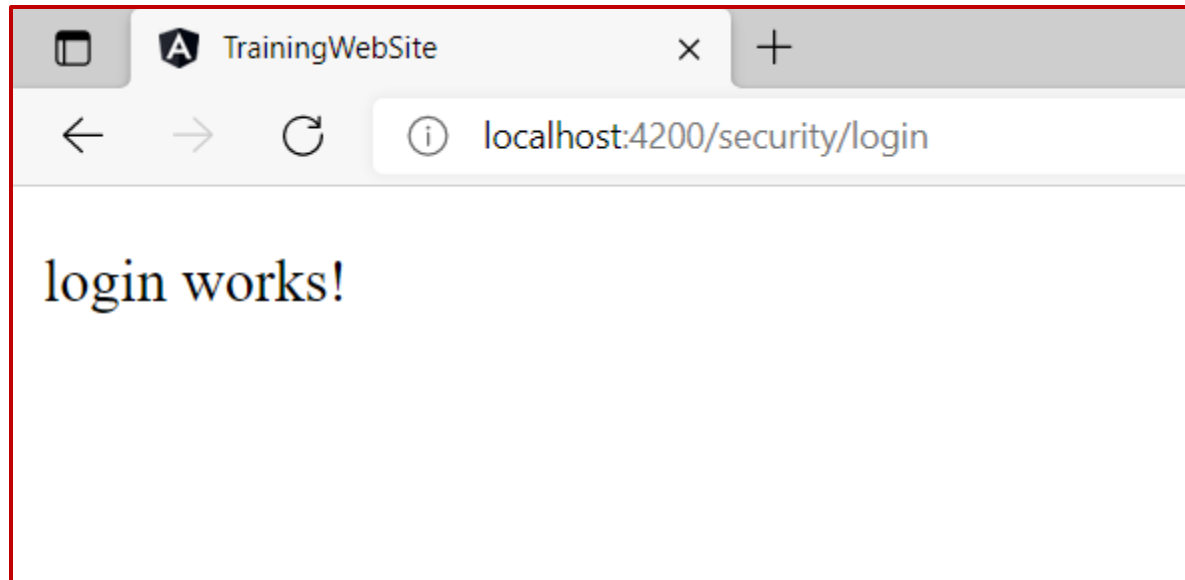
And to route the auth module , go to app-routing.module.ts and add this code :

```
{
    path: 'security',
        loadChildren : () =>
import('./auth/auth.module')
.then((m) => m.AuthModule)
}
];
```

Moving from one page to another

To go to login page or register page you must use **/security/login** or **/security/register** in URL.

But If you want to make the login page the first page to be loaded.

In app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        loadChildren: () =>
import('./auth/auth.module').
            then((m) => m.AuthModule)
    },
```
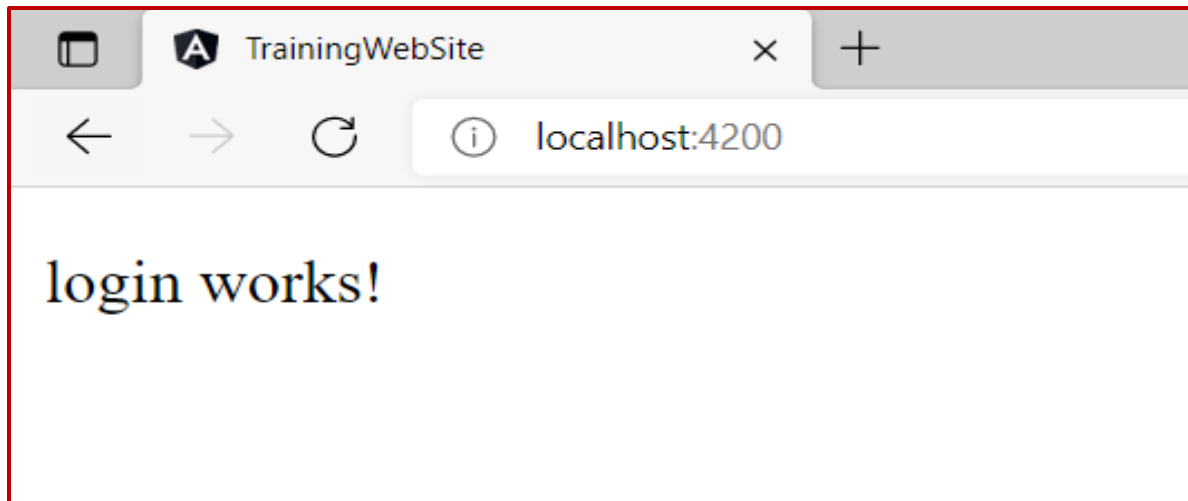
Moving from one page to another

And In auth-routing.module.ts :

```ts
const routes: Routes = [
    {
        path: '',
        component: LoginComponent
    },
    {
        path: 'register',
        component: RegisterComponent
    }
];
```

Moving from one page to another

Chapter 3

Create and Validate form

We will use angular material to install packages :

https://material.angular.io/

To use angular material you must install this package :

ng add @angular/material

Create and Validate form


```
PS C:\Users\User\Desktop\Training\TrainingWebSite> ng add @angular/material
i Using package manager: npm
√ Found compatible package version: @angular/material@12.2.5.
√ Package information loaded.

The package @angular/material@12.2.5 will be installed and executed.
Would you like to proceed? Yes
: Installing package...
```

We will work on Login component.

To use form – filed you must add MatFormFieldModule in import section in auth.module.ts

```
import {MatFormFieldModule} from  '@angular/material/
form-field';
```

To use the input in auth.module.ts, Add MatInputModule
in import section.

```
import {MatInputModule} from  '@angular/material
/input';
```

Form Control: Form input instance with multiple functionality ,ex: validation , flag , and data binding.

Defined form control :

Syntax :

// Form Control object receives two parameter
Variable = new FormControl(**form state** , **Validation Option**)

Create and Validate form

In login.component.html

```html
<div class="container">
    <div class="loginCard">
        <mat-form-field class="example-full-
width"  >
            <mat-label>Email</mat-label>
            <input type="email"  matInput style="fon
t-size: x-large;"  [formControl]="emailControl"
                placeholder="Ex. pat@example.com">
        </mat-form-field>
    </div>
</div>
```

Create and Validate form

Add Password :

```html
<mat-form-field class="example-full-width">
    <mat-label>Password</mat-label>
    <input type="password" matInput style="font-size: x-large;" [formControl]="passwordControl"
        placeholder="Your Password">
</mat-form-field>
```

Create and Validate form

In login.component.ts

```typescript
export class LoginComponent implements OnInit {
  constructor() {}

  emailControl = new FormControl('', [Validators.required,
 Validators.email]);
  passwordControl = new FormControl('', [
    Validators.required,
    Validators.minLength(8),
  ]);
```

**But you must import FormsModule and
ReactiveFormsModule in auth.module.ts in import section :**

Create and Validate form

How to work ***ngIf** ?

In login.component .ts

```
showMessage = false;

toggleShowMessage() {
  this.showMessage = !this.showMessage;
}
```

In login.component .html

```
<h4 *ngIf="showMessage">This is message to show</h4>

<!--
 ifshowMessage==true then 'This is message to show'
else '' -->
<button (click)="toggleShowMessage()">
    toggle</button>
```

Add error message for email:

```
<mat-
error *ngIf="emailControl.hasError('email') && !emai
lControl.hasError('required')">
    Please enter a valid email address
</mat-error>
<mat-
error *ngIf="emailControl.hasError('required')">
    Email is <strong>required</strong>
</mat-error>
```

Add error message for Password:

```
<mat-
error *ngIf="passwordControl.hasError('minlength') &
& !passwordControl.hasError('required')">
    Password must be more than 8 char
</mat-error>
<mat-
error *ngIf="passwordControl.hasError('required')">
    Password is <strong>required</strong>
</mat-error>
```

To get value from form control:

In login.component .html

```
<button type="submit"  (click)="submit()">
    submit</button>
```

Create and Validate form

In login.component .ts

```typescript
submit() {
  //Go to Loader const emailValue=this.emailControl.value;
  const passwordValue = this.passwordControl.value;
  console.log(emailValue, passwordValue);
}
```

Make the button disabled if the user enter invalid username or password.

```
<button [disabled]="!emailFormControl.valid
||!passwordFormControl.valid"  mat-raised-button
style="flex:1;width: 100% ;margin-top: 50px;"color-
text="primary">Login</button>
```