



**UNIVERSITÀ
DI TRENTO**

**Department of
Information Engineering and Computer Science**

Course - Web Architectures

Developing a Simple Angular Application

Date : January 03,2021

Done By:

Ephrem Tibebe Mekonnen

Matricola: 213006

Introduction


This project intends to get my hands dirty with the angular framework. The project is to create an angular application that lists Scottish parliament members and provides detailed information about the person when the name of the member is clicked. Practically, I have two pages: one is where the list of Scottish members is displayed, and the second page is intended to show the person detail. It needs to interact with different JSON data found on the internet to get a list of Scottish parliament members and their details.

The first page, the list page, shows only the name and some icon to show that the person has a photo (i.e. the person has a photo if the value of the photoURI in the data is not empty).

The person detail page is triggered when the name of the member is clicked from the first page named list page. It shows the person's photo(if any), name, birth date, and personal website(if exists). It is also required to configure the routing for the application to work properly. Finally, the transpiled angular application is required to be embedded in a java web application and deploy on the tomcat server.

1.1. Implementation

The application has three components: App component(root), member-list component, person-detail component. They are created using “ ng generate c ‘name of the component’”(for instance member-list component is created using ‘ng generate c member-list’ command), we can use the same command for creating angular service by replacing c by s (i.e. ng generate s ‘name of service’). So, I created an angular service named ParliamentMemberService for enabling the components to share data. Some relevant pieces of codes are depicted below. The `getPerson()` function is one of the relevant functions in the application that retrieves the detail of the person by calling another two supporting functions: `getParty()` and `getPersonalWebsite()`, both methods take `personId` as a parameter. The function is found in the person-detail component of the application.



```

getPerson() {
    this.id = +this.activatedRoute.snapshot paramMap.get('id')!;
    this.parliamentMemberService.getMembers().subscribe(data =>{
        this.parliamentMembers=data;
        this.selectedPerson =
this.parliamentMembers.find((item:any) =>{
            if(item.personId==this.id) {
                return true;
            }

            return false;
        })
        this.getParty(this.id);
        this.getPersonalWebsite(this.id);

        return this.selectedPerson;
    })
}

```

Fig1. getPerson() function to retrieve the person's details when the name of the member is clicked on a list-member page.

```

public getMembers(): Observable<ParliamentMember[]> {
    return this.http.get(this.url1).pipe(map((members: any)
=> {
        return members.map((member: any) => {
            return {
                personId: member.PersonID,
                name: member.PreferredName,
                photoURL: member.PhotoURL,
                birthDate: member.BirthDate
            }
        })
    }))
}

```

Fig 2. getMembers() function is found in ParliamentMemberService class, responsible to interact with api (json data) and return observable to the subscriber component .

1.2. Deployment

- ❖ Screenshots of the app running from my local machine and tomcat server as well.

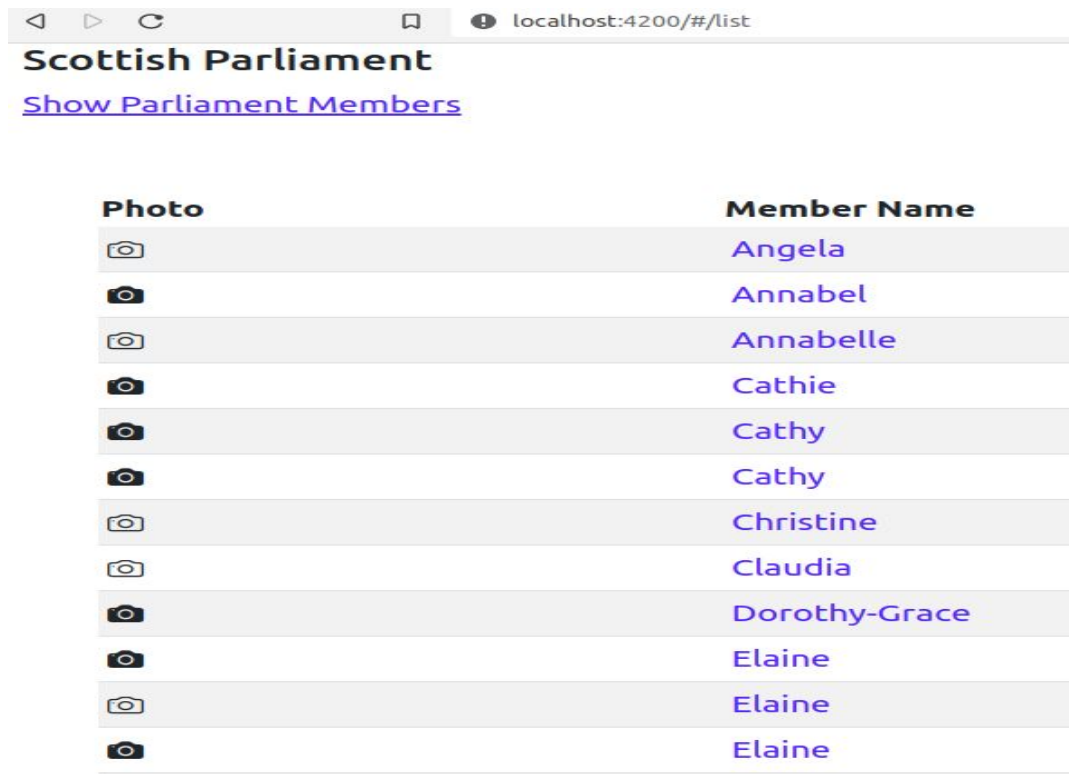














Photo	Member Name
	Angela
	Annabel
	Annabelle
	Cathie
	Cathy
	Cathy
	Christine
	Claudia
	Dorothy-Grace
	Elaine
	Elaine
	Elaine


Fig 3. List page running from <http://localhost:4200>

localhost:4200/#/list/detail/1735

Scottish Parliament

[Show Parliament Members](#)

Person detail



Name: [Angela](#)

Date of birth: [15-07-1970](#)

Party: [Scottish National Party](#)

Website:
<http://www.parliament.scot/msps/currentmsps/angela-constance-msp.aspx>

Fig 4. Person detail page running from http://localhost:4200

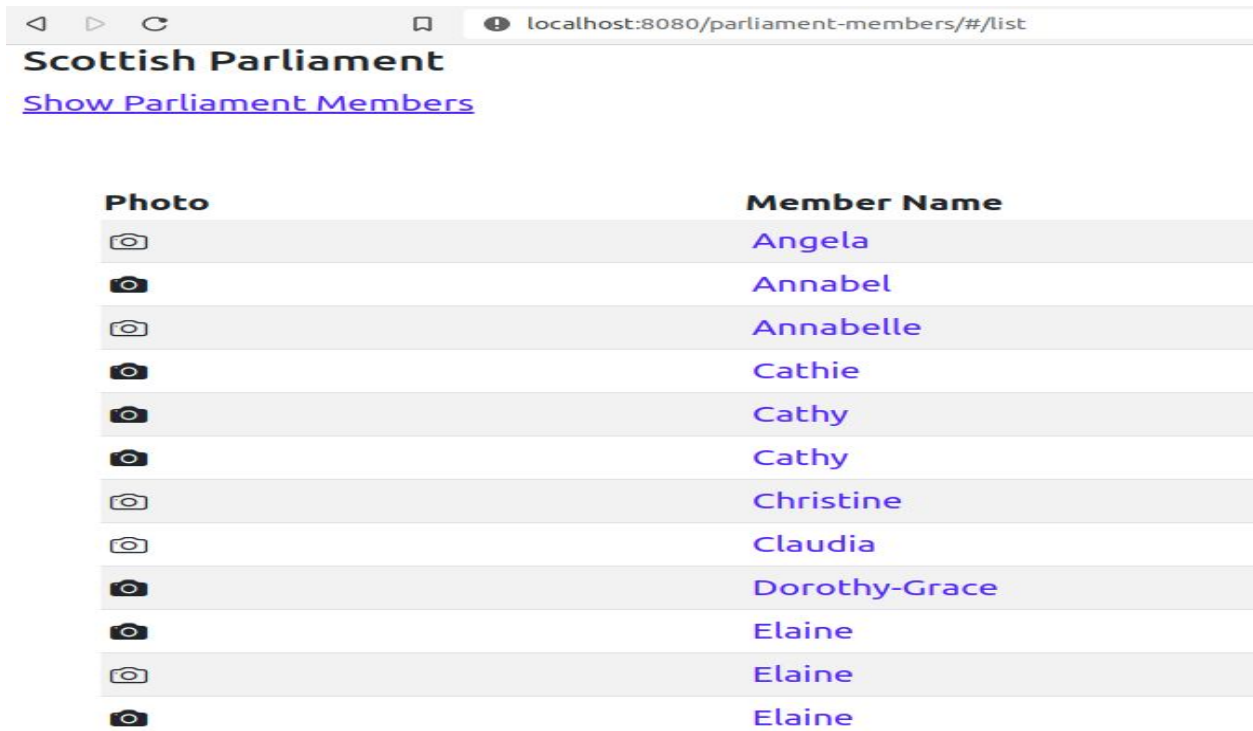














Photo	Member Name
	Angela
	Annabel
	Annabelle
	Cathie
	Cathy
	Cathy
	Christine
	Claudia
	Dorothy-Grace
	Elaine
	Elaine
	Elaine

Fig 5. List page running from tomcat server(<http://localhost:8080>)

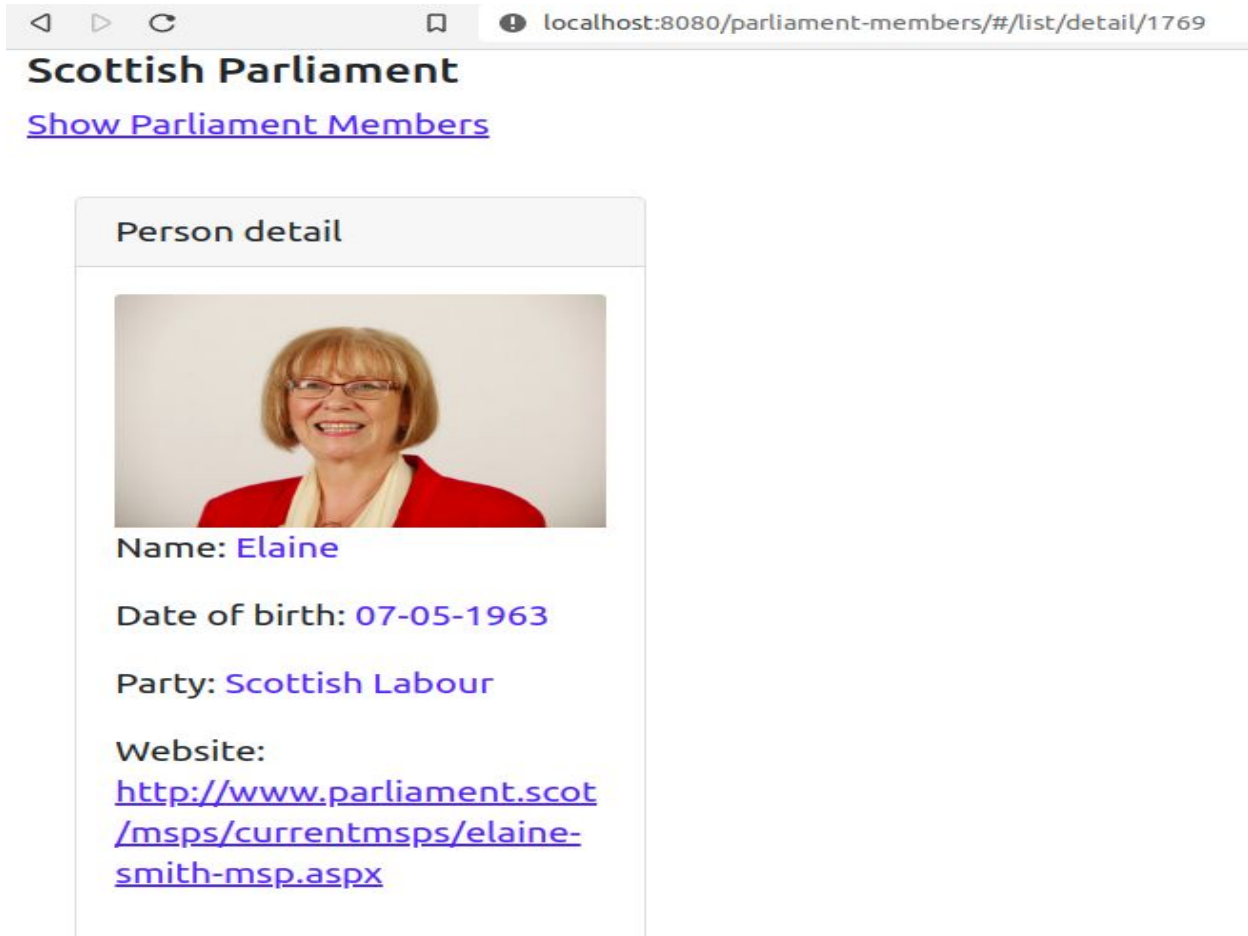


Fig 6. Person detail page running from tomcat server

2.3. Comment

When I refresh the second page (Example: <http://localhost:8080/parliament-members/list/detail/1735>), I was encountering a 404 error, page not found. However, I fixed the problem by using the hashBang approach. The approach is as follows, add the following import in the app.module.ts

```
import {HashLocationStrategy, LocationStrategy} from '@angular/common';
```

And then add { provide:LocationStrategy, useClass:HashLocationStrategy} in the providers array of the @NgModule directives. Finally, I did not focus on the style of the pages and the like, I only focused on the functionality of the given task.