Documentation of Doraemon_Explorer_Hub_Phase-2

“Project Overview”:
Build an Angular application, the "Doraemon Explorer Hub," that caters to the curiosity of Doraemon fans by providing insights into their favorite characters, gadgets, and memorable scenes with the navigation featuring to redirecting the default page. Also applying the services and data comes from the httpclient so use the httpclient and web-api features. Applying the error handling whenever it required.By using CRUD Operations create the element , update the element , delete the element,retrieve the element in the components with the different functionalities.

What New concepts we are added in the file??

1..Fetching apis
2..Observables
3..Httpclient
4..In-memory-web-api
5..Error Handling
6..Message Service
7..CRUD Operations

Previous Concepts;

1..Component
2..Details-component
3..Routing
4..Navigation
5..Bindings

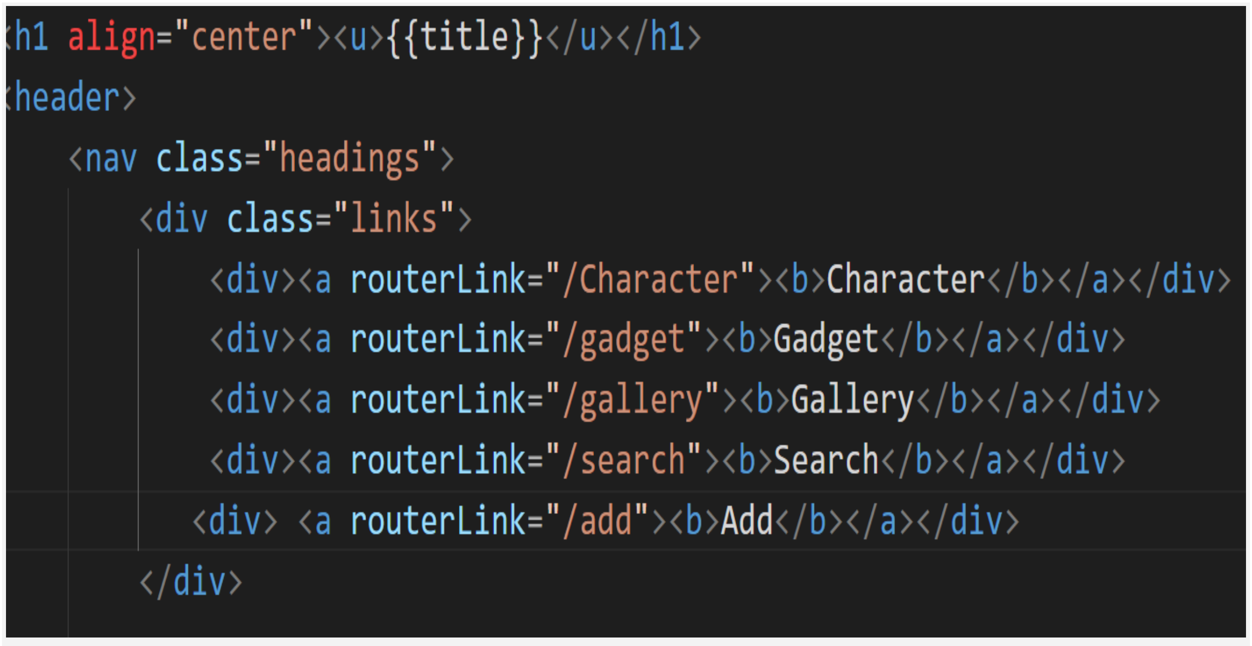
6..Service
7..Interface

TASK1:-Creating the nav bar
Inside the nav bar creating the components what are required .
In terminal use the command:

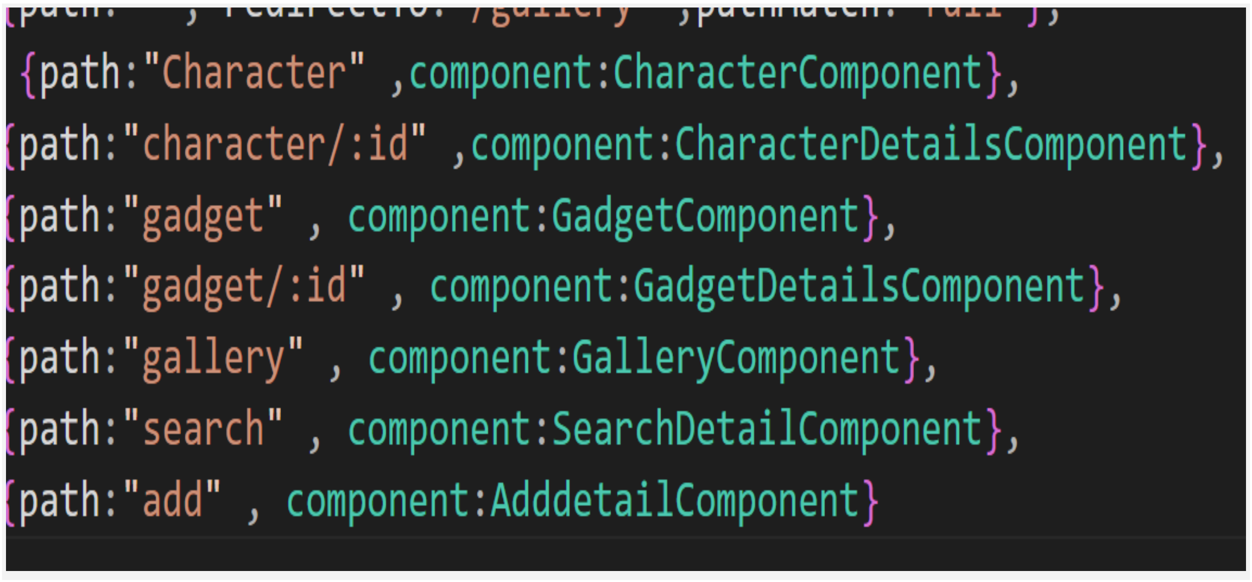
```
→ ng generate component component_name
```

1) **Character component**
2) **Gadget component**
3) **Gallery component**
4) **Add detail component**
5) **Search detail component**
6) **Message detail component**

Also adding the routing of these component in the app.routes.ts file and linked the routes in app.component.html file



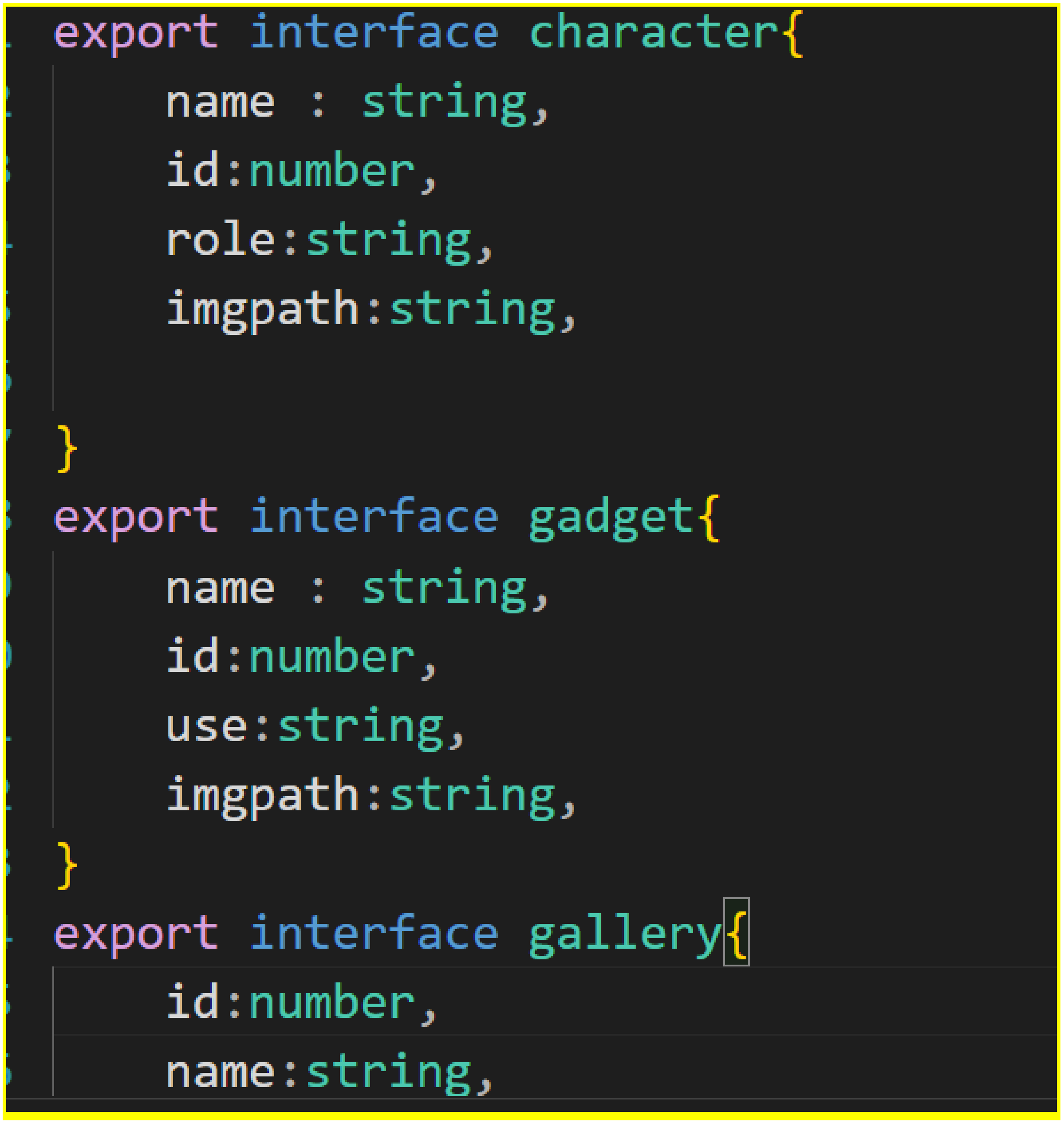
```html
<h1 align="center"><u>{{title}}</u></h1>
<header>
    <nav class="headings">
        <div class="links">
            <div><a routerLink="/Character"><b>Character</b></a></div>
            <div><a routerLink="/gadget"><b>Gadget</b></a></div>
            <div><a routerLink="/gallery"><b>Gallery</b></a></div>
            <div><a routerLink="/search"><b>Search</b></a></div>
          <div> <a routerLink="/add"><b>Add</b></a></div>
        </div>
```



```typescript
 {path:"Character" ,component:CharacterComponent},
{path:"character/:id" ,component:CharacterDetailsComponent},
{path:"gadget" , component:GadgetComponent},
{path:"gadget/:id" , component:GadgetDetailsComponent},
{path:"gallery" , component:GalleryComponent},
{path:"search" , component:SearchDetailComponent},
{path:"add" , component:AdddetailComponent}
```

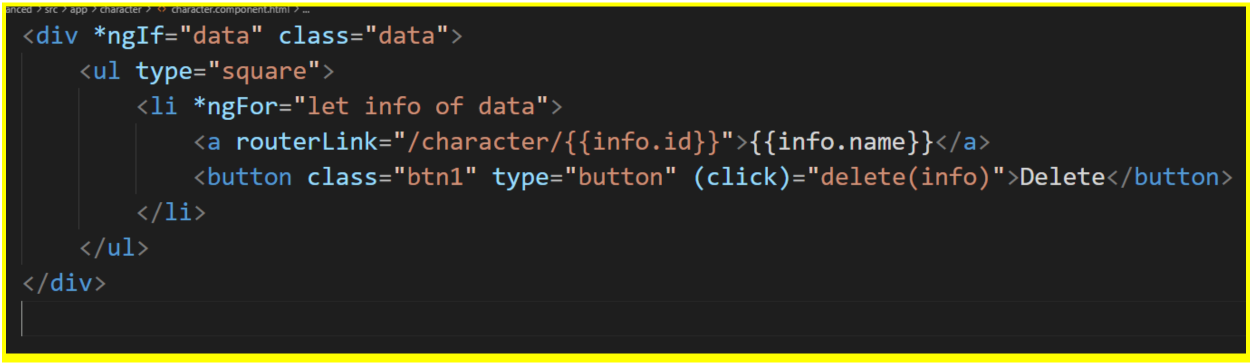
TASK2:- Build up the Character component

Firstofall create a .tsfile in the appcomponent as mydoraemon.ts .In this file creating the interface for the character,gadget and gallery component.



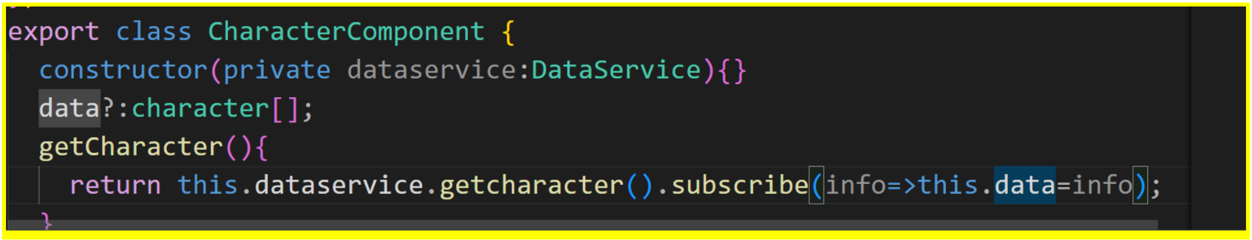
```
export interface character{
    name : string,
    id:number,
    role:string,
    imgpath:string,

}
export interface gadget{
    name : string,
    id:number,
    use:string,
    imgpath:string,
}
export interface gallery{
    id:number,
    name:string,
```

Then in character.component.html
Adding the router link and use the ngFor loop and NgIf condition.


anced > src > app > character > character.component.html > ...

```html
<div *ngIf="data" class="data">
    <ul type="square">
        <li *ngFor="let info of data">
            <a routerLink="/character/{{info.id}}">{{info.name}}</a>
            <button class="btn1" type="button" (click)="delete(info)">Delete</button>
        </li>
    </ul>
</div>
```

In the character.component.ts file
Calling the getcharacter function

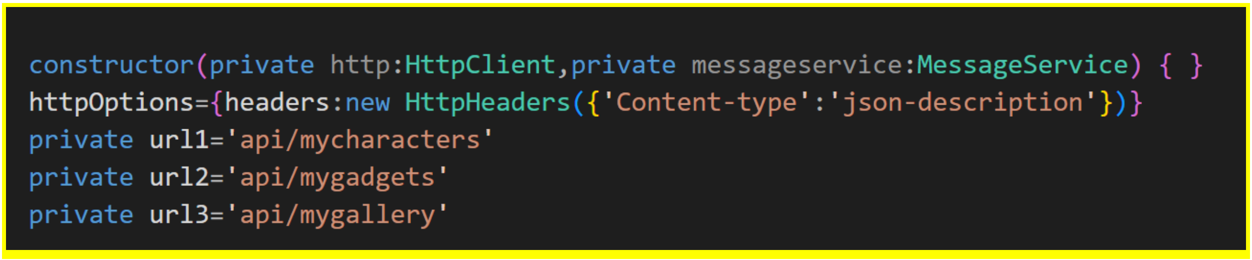


```typescript
export class CharacterComponent {
  constructor(private dataservice:DataService){}
  data?:character[];
  getCharacter(){
    return this.dataservice.getcharacter().subscribe(info=>this.data=info);
  }
```

For using the getcharacter function injects the service
So creating the service named as dataservice
By giving command in the terminal ng generate service service_name

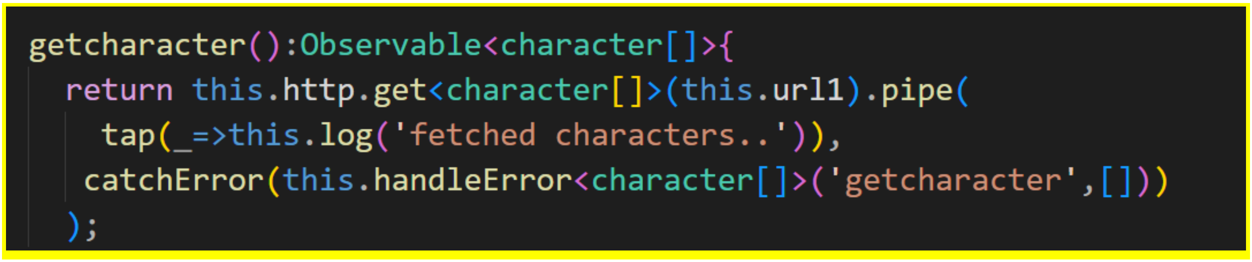
Inside the dataservice.ts file

Firstly fetching the http request and url/api.



```typescript
constructor(private http:HttpClient,private messageservice:MessageService) { }
httpOptions={headers:new HttpHeaders({'Content-type':'json-description'})}
private url1='api/mycharacters'
private url2='api/mygadgets'
private url3='api/mygallery'
```

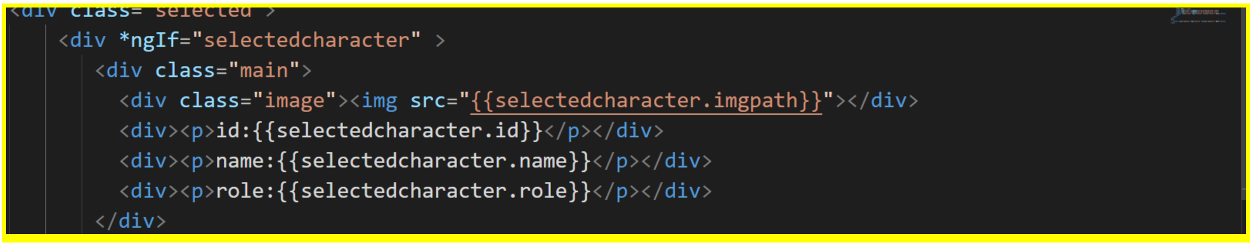
Fetching the character data::



```
getcharacter():Observable<character[]>{
  return this.http.get<character[]>(this.url1).pipe(
    tap(_=>this.log('fetched characters..')),
   catchError(this.handleError<character[]>('getcharacter',[]))
  );
```

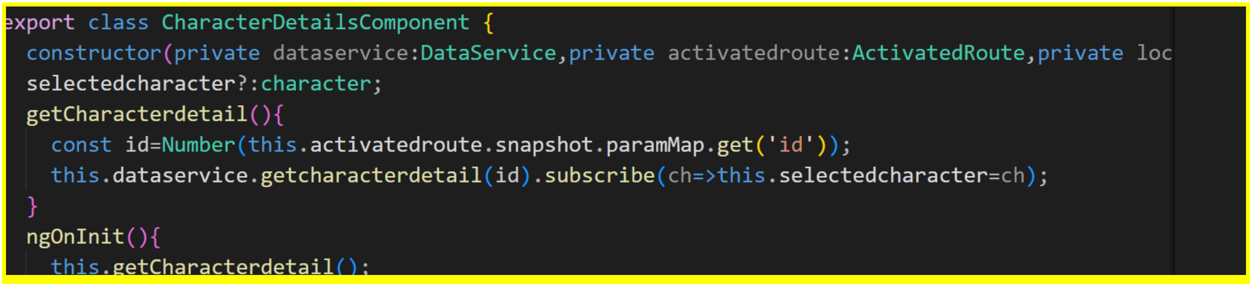
TASK:3 displaying the character details

By generating the new component as character-detail component
In the html file we have to use the functionality displaying the details of each character.



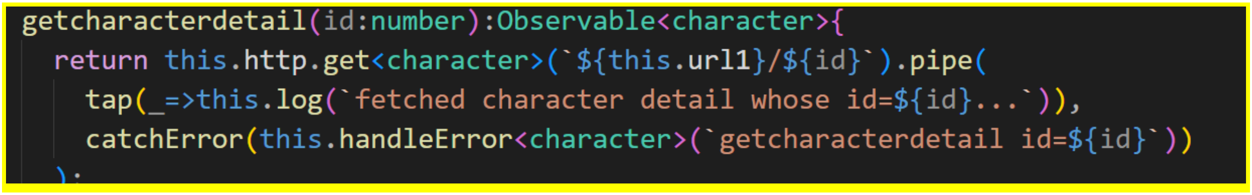
```
<div class="selected">
    <div *ngIf="selectedcharacter" >
      <div class="main">
        <div class="image"><img src="{{selectedcharacter.imgpath}}"></div>
        <div><p>id:{{selectedcharacter.id}}</p></div>
        <div><p>name:{{selectedcharacter.name}}</p></div>
        <div><p>role:{{selectedcharacter.role}}</p></div>
      </div>
```

Importing the activatedroute ,location and service inside the constructor



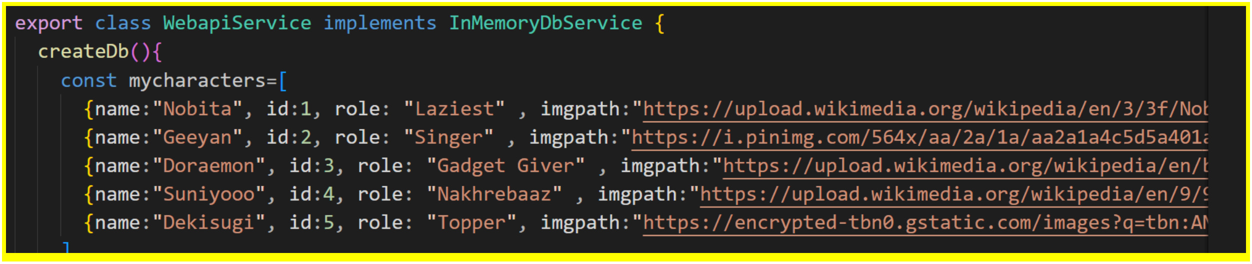
```
xport class CharacterDetailsComponent {
  constructor(private dataservice:DataService,private activatedroute:ActivatedRoute,private loc
  selectedcharacter?:character;
  getCharacterdetail(){
    const id=Number(this.activatedroute.snapshot.paramMap.get('id'));
    this.dataservice.getcharacterdetail(id).subscribe(ch=>this.selectedcharacter=ch);
  }
  ngOnInit(){
    this.getCharacterdetail();
```

HOW DATA COMES FROM THE DATA SERVICE?



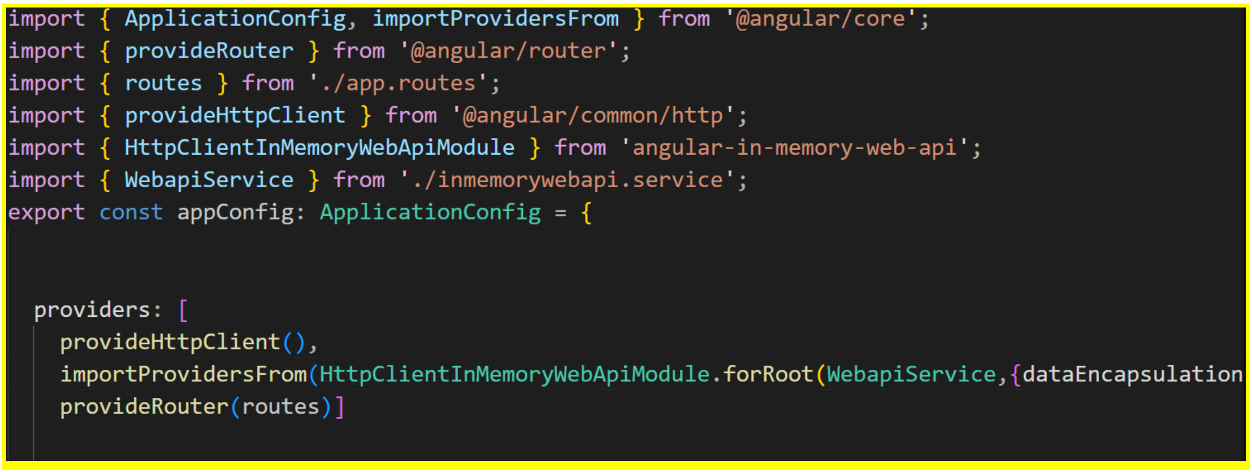
```
getcharacterdetail(id:number):Observable<character>{
  return this.http.get<character>(`${this.url1}/${id}`).pipe(
    tap(_=>this.log(`fetched character detail whose id=${id}...`)),
    catchError(this.handleError<character>(`getcharacterdetail id=${id}`))
  );
```

our entire data is coming for the client side so for this we use a in-memory-web-api service
Inside this service we creating a Database as DB()



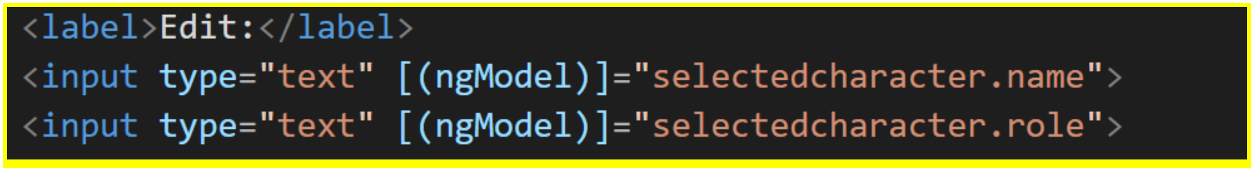
```
export class WebapiService implements InMemoryDbService {
  createDb(){
    const mycharacters=[
      {name:"Nobita", id:1, role: "Laziest" , imgpath:"https://upload.wikimedia.org/wikipedia/en/3/3f/Nob
      {name:"Geeyan", id:2, role: "Singer" , imgpath:"https://i.pinimg.com/564x/aa/2a/1a/aa2a1a4c5d5a401a
      {name:"Doraemon", id:3, role: "Gadget Giver" , imgpath:"https://upload.wikimedia.org/wikipedia/en/b
      {name:"Suniyooo", id:4, role: "Nakhrebaaz" , imgpath:"https://upload.wikimedia.org/wikipedia/en/9/9
      {name:"Dekisugi", id:5, role: "Topper", imgpath:"https://encrypted-tbn0.gstatic.com/images?q=tbn:AN
    ]
```

Importing this Database and the service in the app.config.ts



```
import { ApplicationConfig, importProvidersFrom } from '@angular/core';
import { provideRouter } from '@angular/router';
import { routes } from './app.routes';
import { provideHttpClient } from '@angular/common/http';
import { HttpClientInMemoryWebApiModule } from 'angular-in-memory-web-api';
import { WebapiService } from './inmemorywebapi.service';
export const appConfig: ApplicationConfig = {


  providers: [
    provideHttpClient(),
    importProvidersFrom(HttpClientInMemoryWebApiModule.forRoot(WebapiService,{dataEncapsulation
    provideRouter(routes)]
```

NOTE
In this component we can use two way binding for changing the displayed data by the edit button basically update the data.

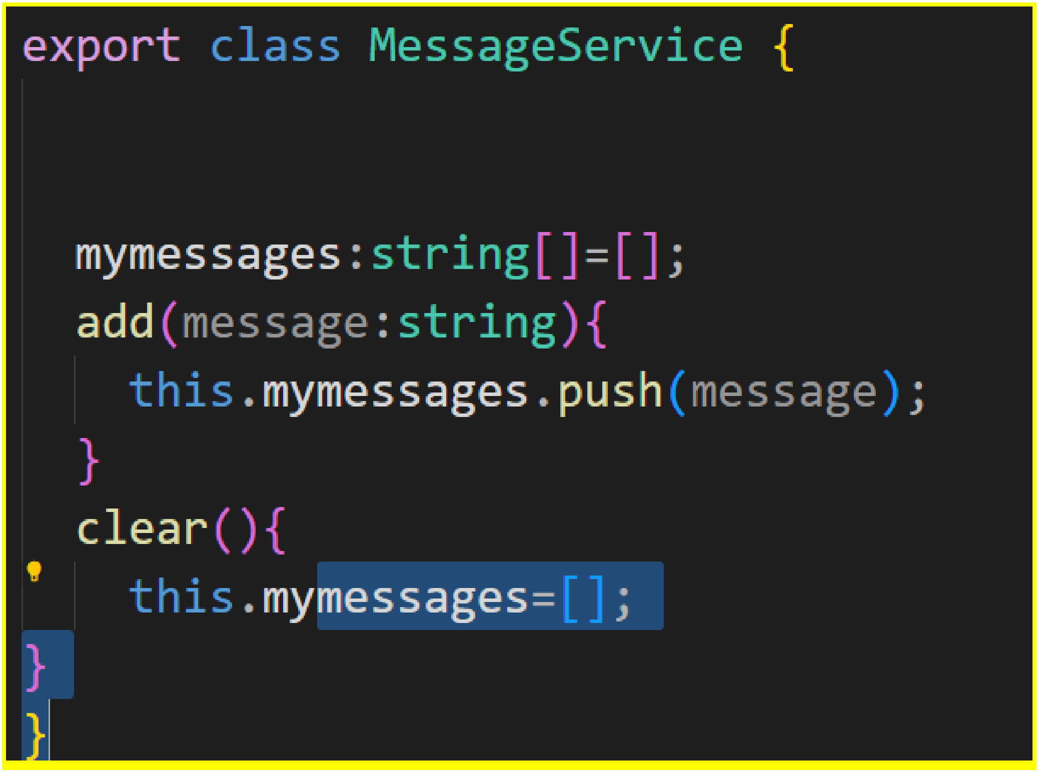
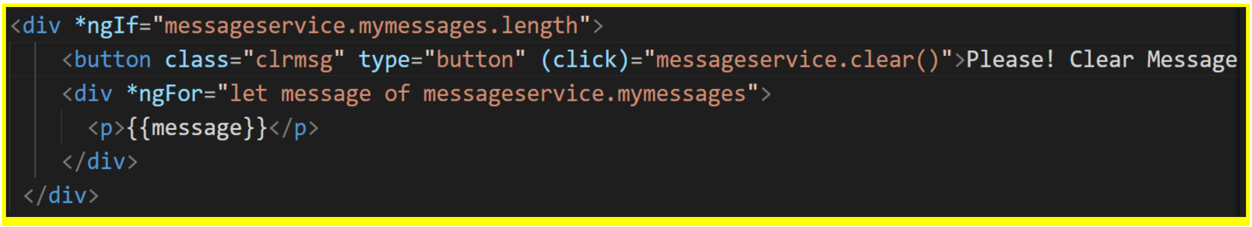
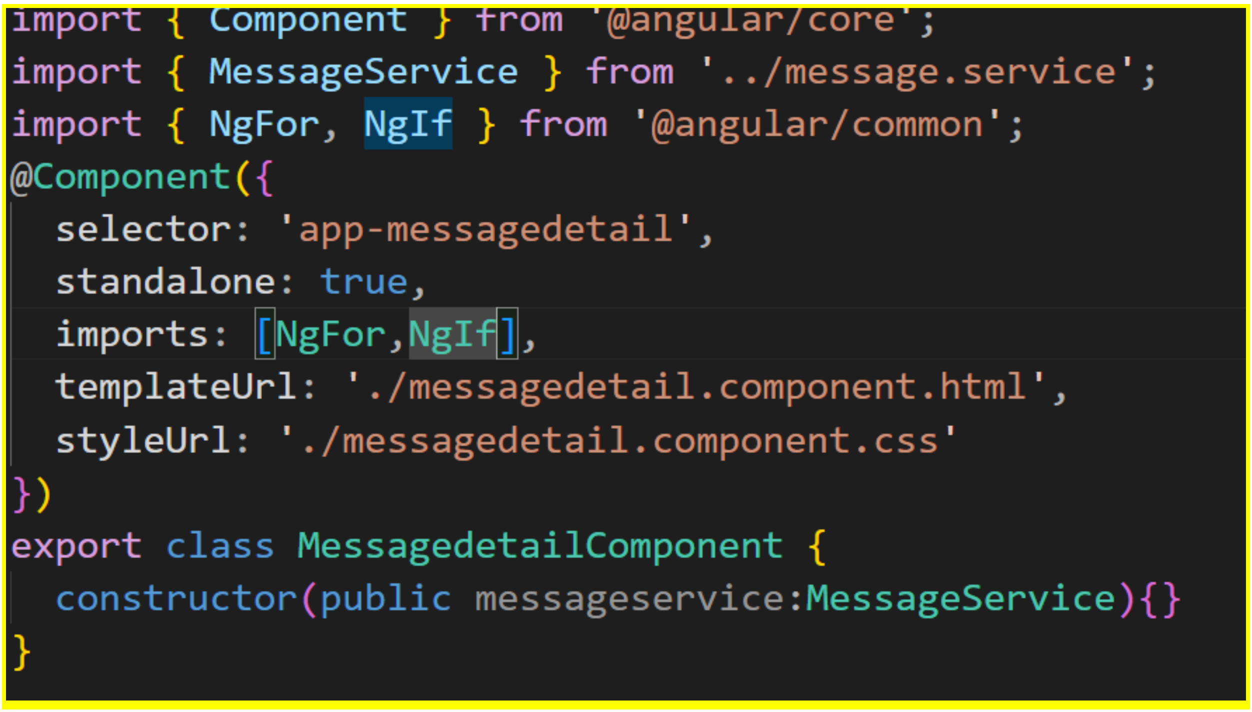


```
<label>Edit:</label>
<input type="text" [(ngModel)]="selectedcharacter.name">
<input type="text" [(ngModel)]="selectedcharacter.role">
```

TASK4;

For showing and clearing the message when we click on the character name and showing is detail,for this we have to use the message component and message service
It is common for component but calling is necessary inside the ts file of that component.

messageservice.ts


```
export class MessageService {


  mymessages:string[]=[];
  add(message:string){
    this.mymessages.push(message);
  }
  clear(){
    this.mymessages=[];
}
}
```

```
<div *ngIf="messageservice.mymessages.length">
    <button class="clrmsg" type="button" (click)="messageservice.clear()">Please! Clear Message
    <div *ngFor="let message of messageservice.mymessages">
      <p>{{message}}</p>
    </div>
 </div>
```

```
import { Component } from '@angular/core';
import { MessageService } from '../message.service';
import { NgFor, NgIf } from '@angular/common';
@Component({
  selector: 'app-messagedetail',
  standalone: true,
  imports: [NgFor,NgIf],
  templateUrl: './messagedetail.component.html',
  styleUrl: './messagedetail.component.css'
})
export class MessagedetailComponent {
  constructor(public messageservice:MessageService){}
}
```

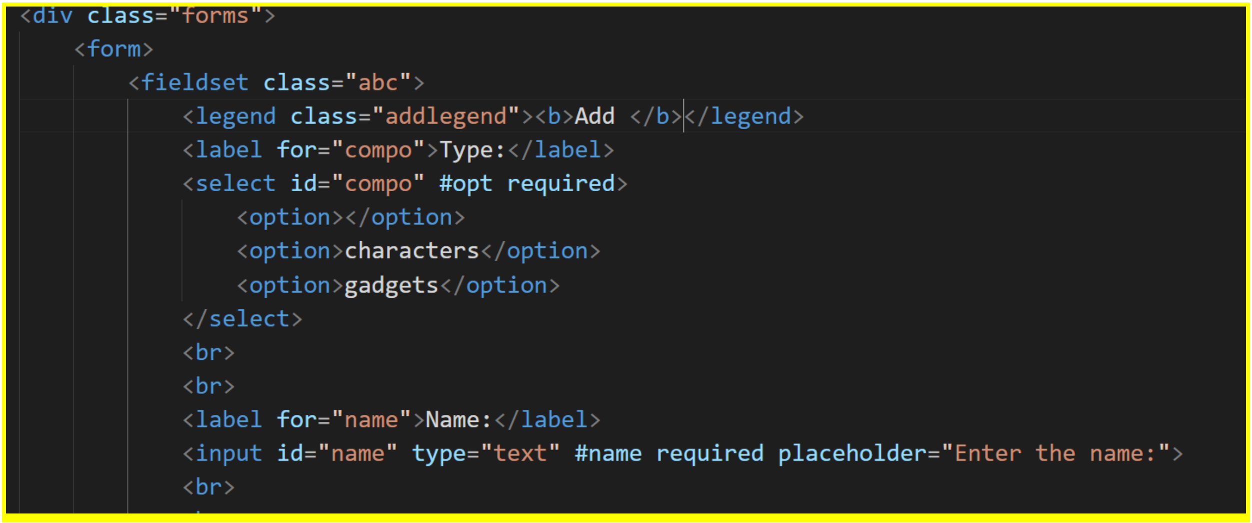
Messagedetail.component.html and messagedetail.component.ts file

TASK 5:

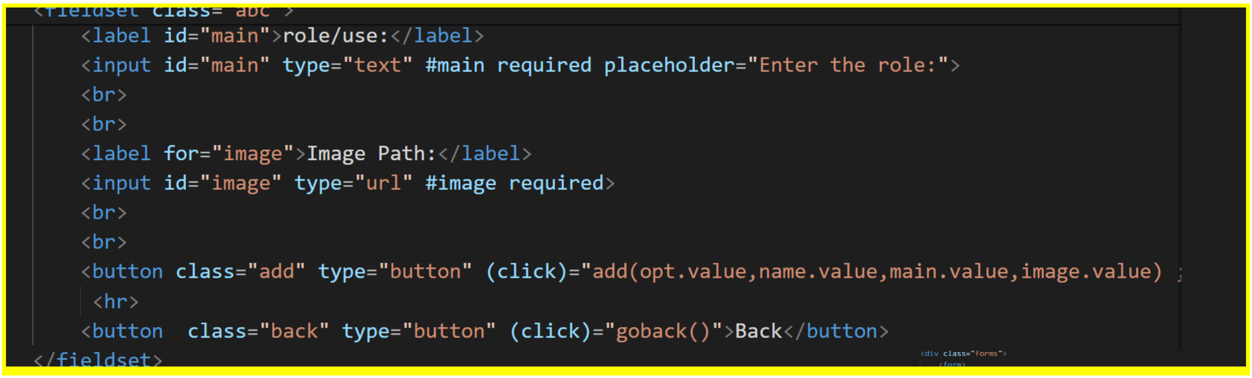
HOW TO ADD DETAILS IN THE ALL COMPONENT

ADD-DETAIL COMPONENT

ADDDETAILCOMPONENT.html file

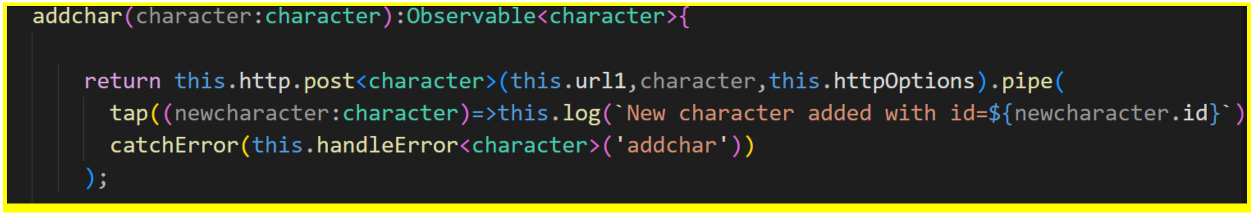


```html
<div class="forms">
    <form>
        <fieldset class="abc">
            <legend class="addlegend"><b>Add </b></legend>
            <label for="compo">Type:</label>
            <select id="compo" #opt required>
                <option></option>
                <option>characters</option>
                <option>gadgets</option>
            </select>
            <br>
            <br>
            <label for="name">Name:</label>
            <input id="name" type="text" #name required placeholder="Enter the name:">
            <br>
```



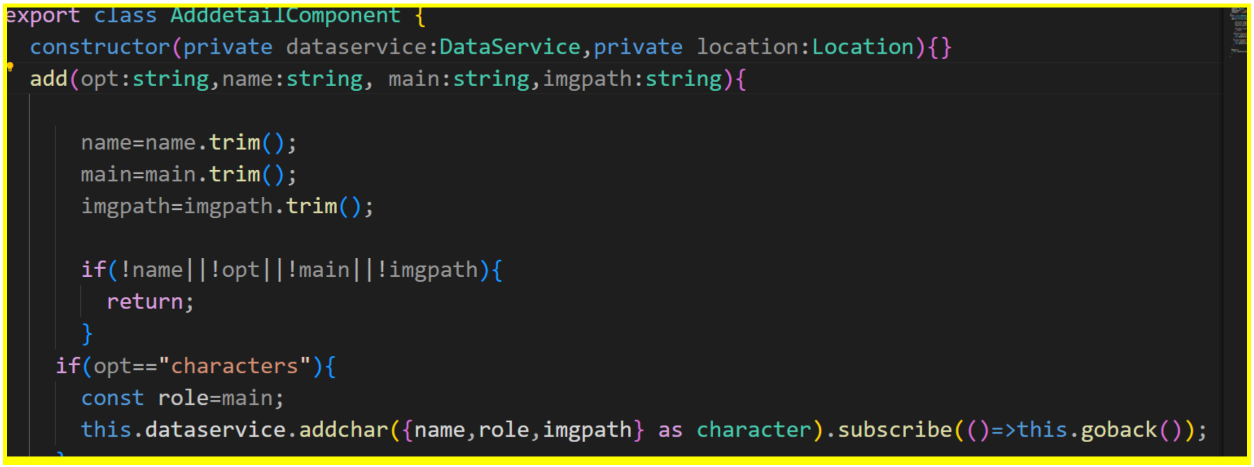
```html
  <fieldset class="abc">
      <label id="main">role/use:</label>
      <input id="main" type="text" #main required placeholder="Enter the role:">
      <br>
      <br>
      <label for="image">Image Path:</label>
      <input id="image" type="url" #image required>
      <br>
      <br>
      <button class="add" type="button" (click)="add(opt.value,name.value,main.value,image.value)
       <hr>
      <button  class="back" type="button" (click)="goback()">Back</button>
  </fieldset>
```

In the adddetailcomponent.ts file we call the addchar function from the dataservice



```
addchar(character:character):Observable<character>{

    return this.http.post<character>(this.url1,character,this.httpOptions).pipe(
      tap((newcharacter:character)=>this.log(`New character added with id=${newcharacter.id}`)
      catchError(this.handleError<character>('addchar'))
    );
```

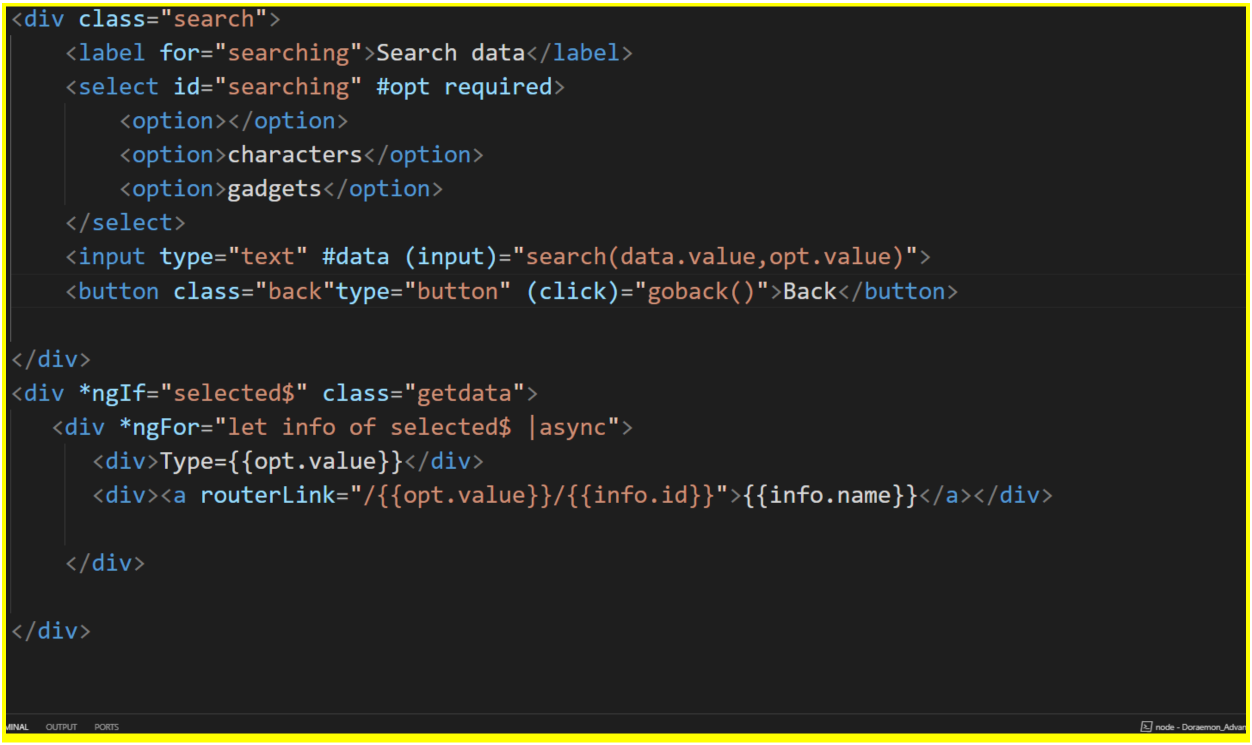
It is the entire functionality for the adddetailcomponent.ts We create this for the option where we want to add the data then what data we added so we can add the dta in the form format .



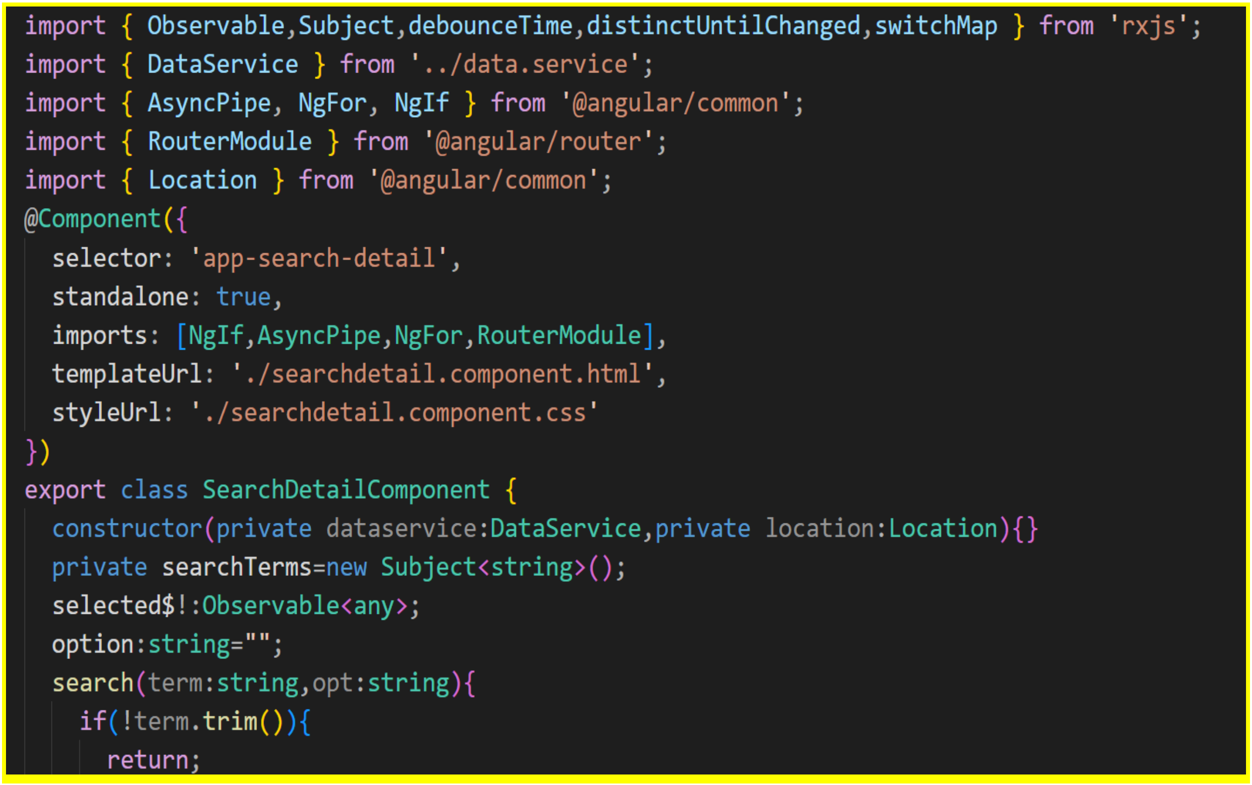
```
export class AdddetailComponent {
  constructor(private dataservice:DataService,private location:Location){}
  add(opt:string,name:string, main:string,imgpath:string){

      name=name.trim();
      main=main.trim();
      imgpath=imgpath.trim();

      if(!name||!opt||!main||!imgpath){
        return;
      }
    if(opt=="characters"){
      const role=main;
      this.dataservice.addchar({name,role,imgpath} as character).subscribe(()=>this.goback());
```

TASK 6:
SEARCHING THE DATA IN THE COMPONENTS

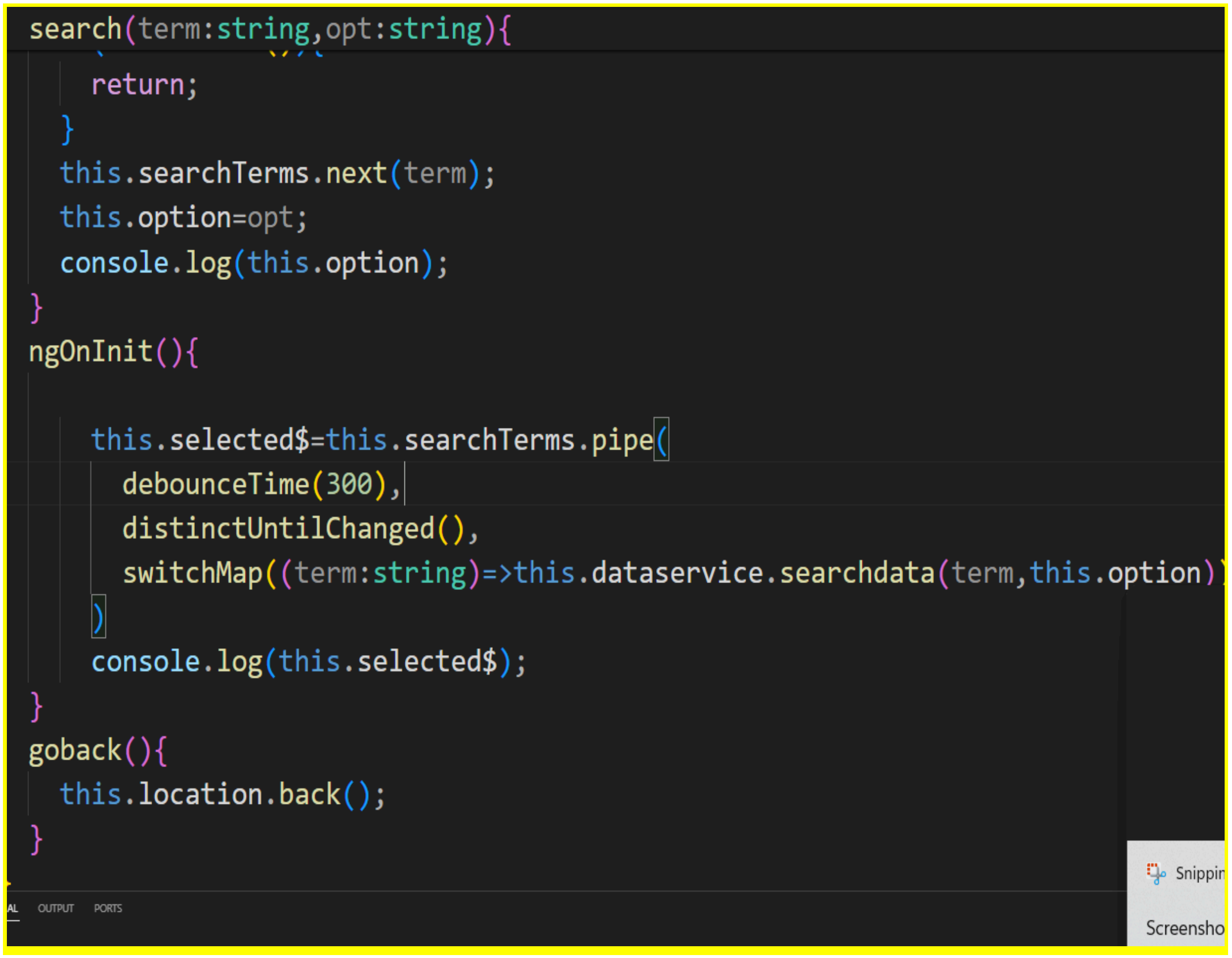
SERACHDETAIL COMPONENT AND THE SERVICE



```
<div class="search">
    <label for="searching">Search data</label>
    <select id="searching" #opt required>
        <option></option>
        <option>characters</option>
        <option>gadgets</option>
    </select>
    <input type="text" #data (input)="search(data.value,opt.value)">
    <button class="back"type="button" (click)="goback()">Back</button>

</div>
<div *ngIf="selected$" class="getdata">
   <div *ngFor="let info of selected$ |async">
      <div>Type={{opt.value}}</div>
      <div><a routerLink="/{{opt.value}}/{{info.id}}">{{info.name}}</a></div>

    </div>

</div>
```

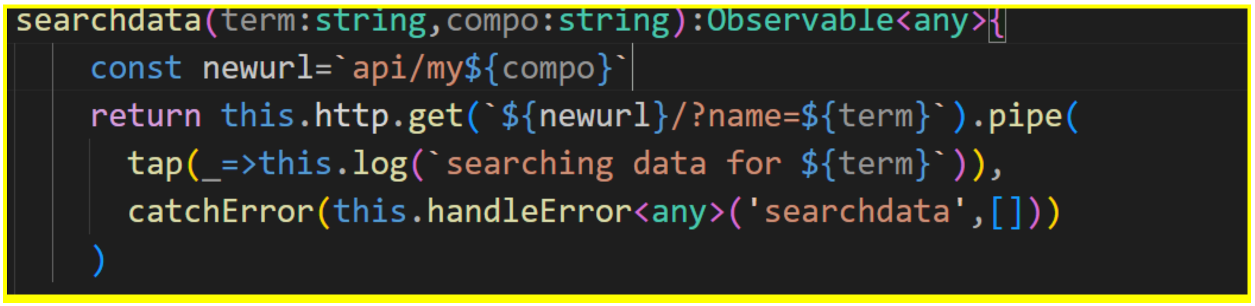


```
import { Observable,Subject,debounceTime,distinctUntilChanged,switchMap } from 'rxjs';
import { DataService } from '../data.service';
import { AsyncPipe, NgFor, NgIf } from '@angular/common';
import { RouterModule } from '@angular/router';
import { Location } from '@angular/common';
@Component({
  selector: 'app-search-detail',
  standalone: true,
  imports: [NgIf,AsyncPipe,NgFor,RouterModule],
  templateUrl: './searchdetail.component.html',
  styleUrl: './searchdetail.component.css'
})
export class SearchDetailComponent {
  constructor(private dataservice:DataService,private location:Location){}
  private searchTerms=new Subject<string>();
  selected$!:Observable<any>;
  option:string="";
  search(term:string,opt:string){
    if(!term.trim()){
      return;
```



```
search(term:string,opt:string){
    return;
  }
  this.searchTerms.next(term);
  this.option=opt;
  console.log(this.option);
}
ngOnInit(){

    this.selected$=this.searchTerms.pipe(
      debounceTime(300),
      distinctUntilChanged(),
      switchMap((term:string)=>this.dataservice.searchdata(term,this.option)
    )
    console.log(this.selected$);
}
goback(){
  this.location.back();
}
```

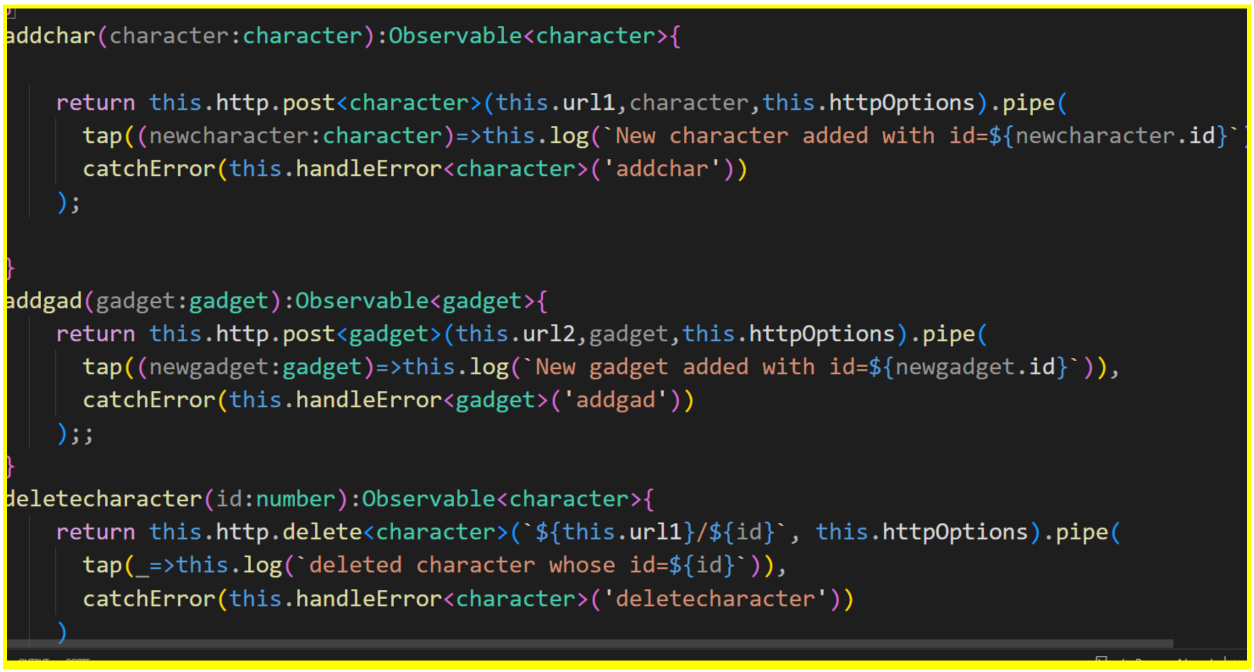
CALLING THE FUNCTION IN DATASERVICE.TS
And handles the errorhandling as well



```
searchdata(term:string,compo:string):Observable<any>{
    const newurl=`api/my${compo}`
    return this.http.get(`${newurl}/?name=${term}`).pipe(
      tap(_=>this.log(`searching data for ${term}`)),
      catchError(this.handleError<any>('searchdata',[]))
    )
```

TASK 7: CRUD Operations
CREATE
RETRIEVE
UPDATE
DELETE

ALL of these operations are performed on both gadget component as well as character component with the help of bindings and dependency injection. We call the functions in data service as dataservice is linked with all of the components and subcomponents.



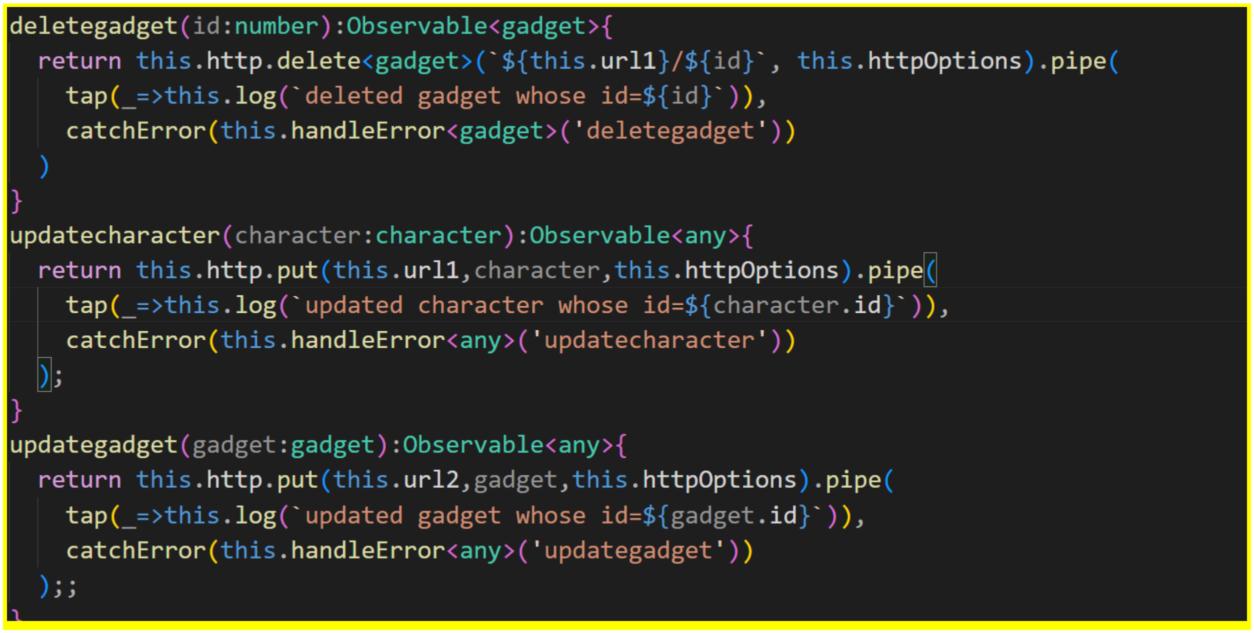
```
addchar(character:character):Observable<character>{

    return this.http.post<character>(this.url1,character,this.httpOptions).pipe(
      tap((newcharacter:character)=>this.log(`New character added with id=${newcharacter.id}`
      catchError(this.handleError<character>('addchar'))
    );

}
addgad(gadget:gadget):Observable<gadget>{
    return this.http.post<gadget>(this.url2,gadget,this.httpOptions).pipe(
      tap((newgadget:gadget)=>this.log(`New gadget added with id=${newgadget.id}`)),
      catchError(this.handleError<gadget>('addgad'))
    );;
}
deletecharacter(id:number):Observable<character>{
    return this.http.delete<character>(`${this.url1}/${id}`, this.httpOptions).pipe(
      tap(_=>this.log(`deleted character whose id=${id}`)),
      catchError(this.handleError<character>('deletecharacter'))
    )
```

Addchar is for the character component where we can add the character’s name if we want to add any name with the intialzing parameters.for adding the data we used post method.

Similarly, delete the character's name if we want by click on delete button . Delete button perform the event binding and condition. It is declared in the html file.

Same for GADGET Component

Update character is used for update the information whatever is displayed for update we using put method.
In the both component gadget and character i.e data is changeable.

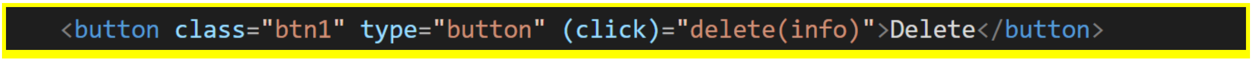


```
deletegadget(id:number):Observable<gadget>{
  return this.http.delete<gadget>(`${this.url1}/${id}`, this.httpOptions).pipe(
    tap(_=>this.log(`deleted gadget whose id=${id}`)),
    catchError(this.handleError<gadget>('deletegadget'))
  )
}
updatecharacter(character:character):Observable<any>{
  return this.http.put(this.url1,character,this.httpOptions).pipe(
    tap(_=>this.log(`updated character whose id=${character.id}`)),
    catchError(this.handleError<any>('updatecharacter'))
  );
}
updategadget(gadget:gadget):Observable<any>{
  return this.http.put(this.url2,gadget,this.httpOptions).pipe(
    tap(_=>this.log(`updated gadget whose id=${gadget.id}`)),
    catchError(this.handleError<any>('updategadget'))
  );;
```

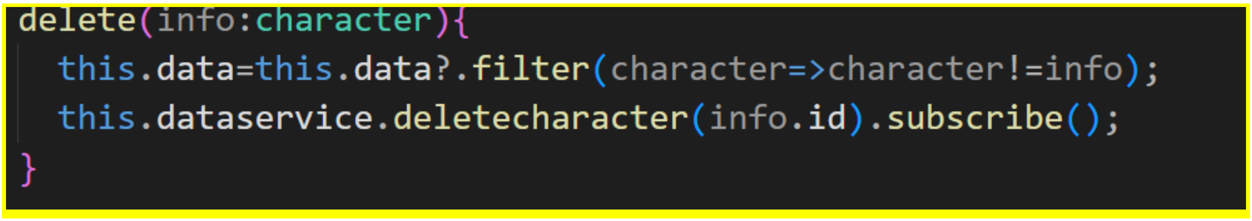
TASK 8:

Creating the delete button,back button and save button.

Delete Button: deleting the detail data which is display in the screen.

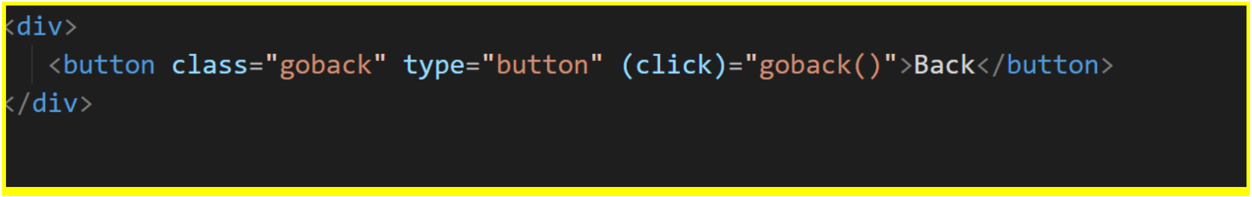


```
  <button class="btn1" type="button" (click)="delete(info)">Delete</button>
```

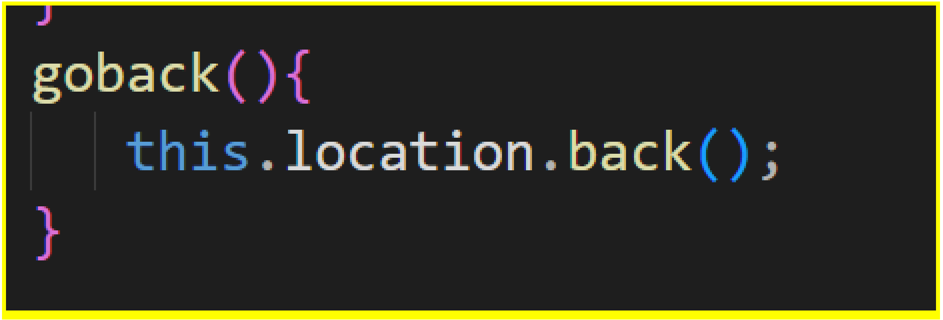


```
delete(info:character){
  this.data=this.data?.filter(character=>character!=info);
  this.dataservice.deletecharacter(info.id).subscribe();
}
```

Back Button:for redirecting to previous page
FOr this we used the LOCATION directive

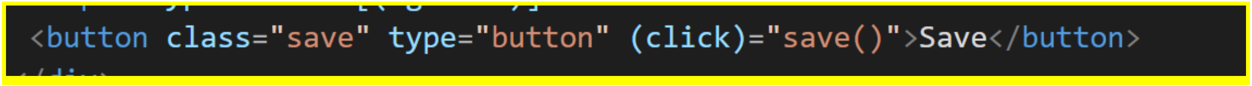


```
<div>
  <button class="goback" type="button" (click)="goback()">Back</button>
</div>
```

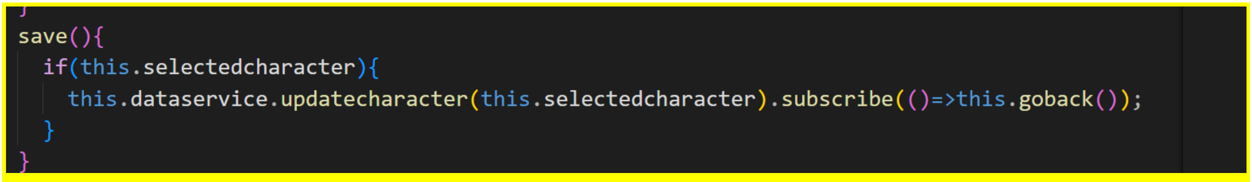


```
goback(){
  this.location.back();
}
```

SAVE BUTTON: for saving the information/data.



```
 <button class="save" type="button" (click)="save()">Save</button>
```



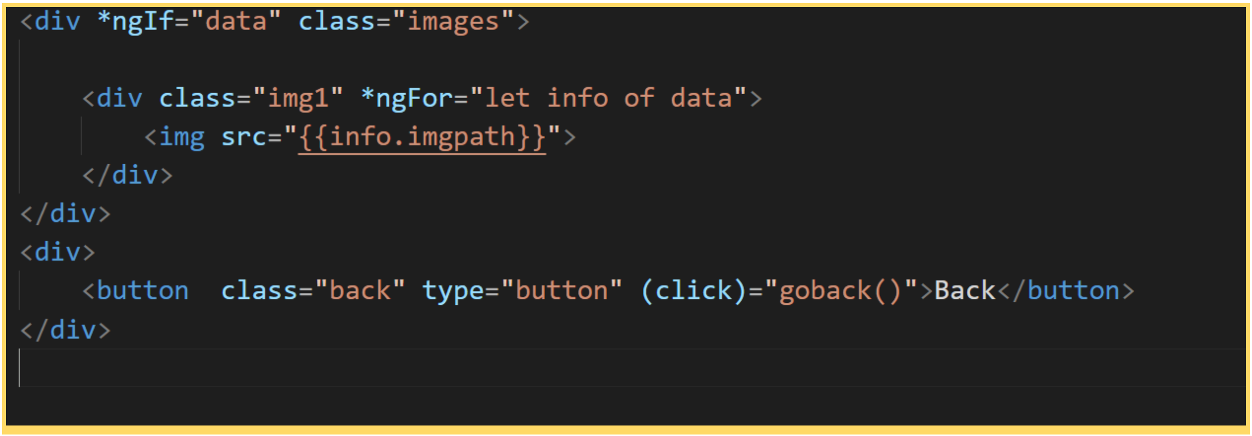
```
save(){
  if(this.selectedcharacter){
    this.dataservice.updatecharacter(this.selectedcharacter).subscribe(()=>this.goback());
  }
}
```

Same process is done for the gadget component and gallery component.
For gadget component we have the gadget and gadget details but for gallery component there is no detailing of gallery component.

Gadget component functioning is showing the list of gadgets and after that entire detail of the gadget with the image of that gadget.
And We can add,delete ,update and change the gadget data.

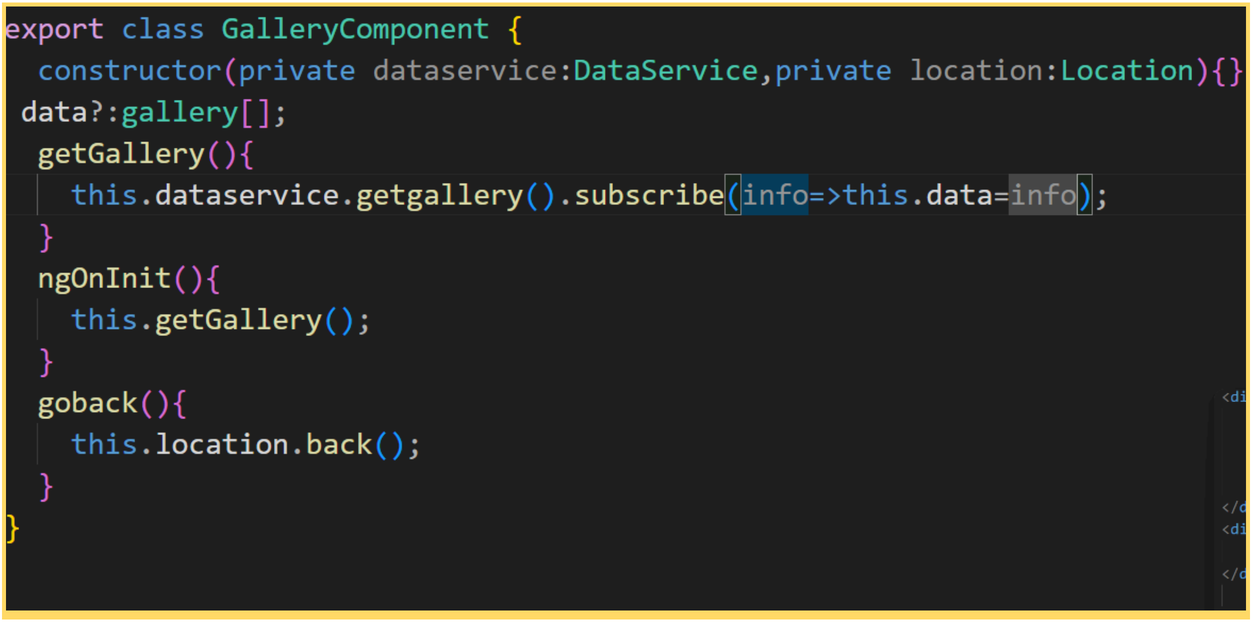
Gallery component is displaying the images of the doreamon simply.
By default we redirect to the gallery.

Gallery Component.html file



```
<div *ngIf="data" class="images">

    <div class="img1" *ngFor="let info of data">
        <img src="{{info.imgpath}}">
    </div>
</div>
<div>
    <button  class="back" type="button" (click)="goback()">Back</button>
</div>
```

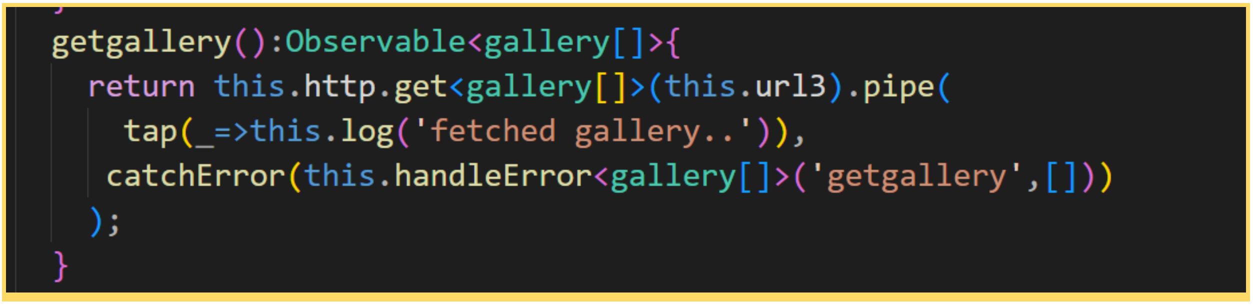
Galley Component.ts file



```
export class GalleryComponent {
  constructor(private dataservice:DataService,private location:Location){}
 data?:gallery[];
  getGallery(){
    this.dataservice.getgallery().subscribe(info=>this.data=info);
  }
  ngOnInit(){
    this.getGallery();
  }
  goback(){
    this.location.back();
  }
}
```

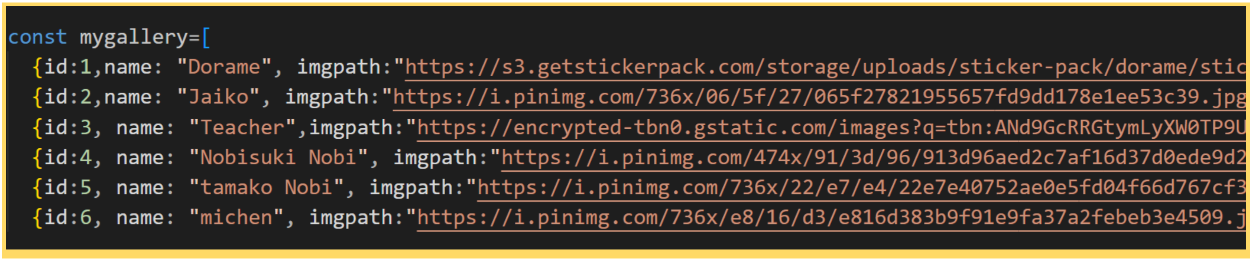
dataservice.ts

Getgallery function:



```
  getgallery():Observable<gallery[]>{
    return this.http.get<gallery[]>(this.url3).pipe(
      tap(_=>this.log('fetched gallery..')),
     catchError(this.handleError<gallery[]>('getgallery',[]))
    );
  }
```

inmemorywebapi.ts : array data



```
const mygallery=[
  {id:1,name: "Dorame", imgpath:"https://s3.getstickerpack.com/storage/uploads/sticker-pack/dorame/stic
  {id:2,name: "Jaiko", imgpath:"https://i.pinimg.com/736x/06/5f/27/065f27821955657fd9dd178e1ee53c39.jpg
  {id:3, name: "Teacher",imgpath:"https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRRGtymLyXW0TP9U
  {id:4, name: "Nobisuki Nobi", imgpath:"https://i.pinimg.com/474x/91/3d/96/913d96aed2c7af16d37d0ede9d2
  {id:5, name: "tamako Nobi", imgpath:"https://i.pinimg.com/736x/22/e7/e4/22e7e40752ae0e5fd04f66d767cf3
  {id:6, name: "michen", imgpath:"https://i.pinimg.com/736x/e8/16/d3/e816d383b9f91e9fa37a2febeb3e4509.j
```