Harmony IT Solution

Angular

Tahaluf Training Center 2022

Harmony IT Solution

1. Hits API (Update)
2. Hits API (Delete)
3. Hits API (Search)

Harmony IT Solution

Objective

The Objective of this lecture

- Understand the HTTP put and how to update any row in any table.
- Understand the ng-template and the viewChild decorator.
- Dealing with the delete function and how we can search for a specific row and get their information.

Hits API (Update)

Exercise

Retrieve all courses from the database as a table in the Manage course component.

How to deal with the update function

Step one: Add a new function in the home service that hits API to the Update function.

```
updateCourse(body:any){
  body.imagename=this.display_image;
  debugger
  this.http.put('https://localhost:44320/api/course/',
  body).subscribe((res)=>{
  this.toastr.success('updated Successfully :)');
  },err=>{
  this.toastr.error(err.status,err.message);
  })}
```

How to deal with the update function

Step two: Add a form group in an HTML file in the Manage course component to receive the new data for a specific row.

```
updateform : FormGroup = new FormGroup({
courseid:new FormControl(),
coursename: new FormControl() ,
price:new FormControl() ,
startdate:new FormControl() ,
enddate:new FormControl() ,
imagename:new FormControl()
})
```

How to deal with the update function

Step three: Add a new button in the HTML file inside the table in the Mange course component and send the previous data as a parameter.

```
<button (click)="openUpdateDailog(course.courseid,course.coursename,course.price,
course.startdate,course.enddate,course.imagename)">Update</button>
```

How to deal with the update function

Step four: define an object to receive previous data.

```
previous_data:any={};
openUpdateDailog(courseid1:any, coursename1:any,
price1:any,startdate1:any,enddate1:any, imagename1:any)
{
  this.previous_data={
    courseid:courseid1,
    coursename:coursename1,
    price:price1,
    startdate:startdate1,
    enddate:enddate1,
    imagename:imagename1
  }
  console.log(this.previous_data);
  this.updateform.controls['courseid'].setValue(this.previous_data.courseid);
```

How to deal with the update function

Step five: Add the template of the update form in the HTML file of the Manage course component.

Note: Use the ng template which is used to render HTML within Angular templates. Although, it is not rendered directly to the DOM.

The <ng-template> is an Angular element for rendering HTML. It is never displayed directly.

The <ng-template> is an Angular element for rendering HTML. It is never displayed directly.

Harmony IT Solution

How to deal with the update function

```
<ng-template #callUpdateDailog>
    <form class="example-form" [formGroup]="updateform" >
        <mat-form-field class="example-full-width" appearance="fill">
          <mat-label>Course Name </mat-label>
          <input type="text" matInput formControlName="coursename"
          [(ngModel)]="previous_data.coursename" >
          </mat-form-field>
            <mat-form-field class="example-full-width" appearance="fill">
            <mat-label>Course price </mat-label>
            <input type="number" matInput formControlName="price"
            [(ngModel)]="previous_data.price" >
          </mat-form-field>
          <mat-form-field class="example-full-width" appearance="fill">
            <mat-label>Start Date </mat-label>
            <input type="date" matInput formControlName="startdate"
            [(ngModel)]="previous_data.startdate" >
          </mat-form-field>
```

How to deal with the update function

```
<mat-form-field class="example-full-width" appearance="fill">
  <mat-label>End Date </mat-label>
  <input type="text" matInput formControlName="enddate"
  [(ngModel)]="previous_data.enddate" >
</mat-form-field>

 <div class="example-container">
  <input type="file" #file formControlName="imagename"
  [(ngModel)]="previous_data.imagename"
   (change)="uploadFile(file.files)">
  </div>
```

The <ng-template> is an Angular element for rendering HTML. It is never displayed directly.

Harmony IT Solution

How to deal with the update function

@ViewChild: Property decorator for configuring the view query. Whenever a change is detected, the change detector looks for the first element or directive matching the selector in the view DOM. As soon as a new child matches the selector in the view DOM, the property is updated.

The <ng-template> is an Angular element for rendering HTML. It is never displayed directly.

Harmony IT Solution

How to deal with the update function

In the managecourse.component.ts

```
@ViewChild('callUpdateDailog') callUpdateDailog! :TemplateRef<any>
```

Then complete the openUpdateDailog.

```
previous_data:any={};
  openUpdateDailog(courseid1:any, coursename1:any,
  price1:any,startdate1:any,enddate1:any, imagename1:any)
  {
    this.previous_data={
      courseid:courseid1,
      coursename:coursename1,
      price:price1,
      startdate:startdate1,
      enddate:enddate1,
      imagename:imagename1
    }
    console.log(this.previous_data);
    this.updateForm.controls['courseid'].setValue(this.previous_data.courseid)
    this.dialog.open(this.callUpdateDailog);
  }
```

How to deal with the update function

Step six: Add a new button in the update form

```
<button (click)="updateCourse()">Update </button>
</form>
</ng-template>
```

How to deal with the update function

Step seven: Implement the update course function in the typescript file

```
updateCourse(){
  this.home.updateCourse(this.updateform.value);
}
```

Harmony IT Solution

Hits API (Delete)

How to deal with the update function

Step one: Add a new function in the home service that hits API to the Delete function.

```
deleteCourse(id:number){
  this.http.delete('https://localhost:44320/api/course/'+id).subscribe((resp)=>{
    this.toastr.success('Deleted Successfully ');
  },err=>{
    this.toastr.error(err.message , err.status)
  });
}
```

How to deal with the Delete function

Step two: Add a new button called Delete in the HTML file inside the table in the Mange course component and send the id as a parameter.

```
<td><button (click)="openDeleteDailog(course.courseid)">Delete</button></td>
```

How to deal with the Delete function

Step three: Add the template of the delete form in the HTML file of the Manage course component.

```
<ng-template #callDeleteDailog>
<h2>Are you sure you want to delete this item?</h2>
<button mat-button mat-dialog-close="yes">Yes</button>
<button mat-button mat-dialog-close="no">No</button>
</ng-template>
```

How to deal with the Delete function

And define a ViewChild properties In the managecourse.component.ts.

```
@ViewChild('callDeleteDailog') callDeleteDailog! :TemplateRef<any>
```

How to deal with the Delete function

Step four: Implement the openDelete function In the managecourse.component.ts

```
openDeleteDailog(id:number){
 const dialogRef=this.dialog.open(this.callDeleteDailog);
 dialogRef.afterClosed().subscribe((result)=>{
   if(result!=undefined){
     if(result=='yes')
     this.home.deleteCourse(id);
   else if(result=='no')
   console.log('Thank you');
   }
 })
}
```

Hits API (Search)

How to deal with the Search function

Step one: Add a new function in the home service that hits API to the Get course by name function.

```
searchCourse(data:any)
{
this.http.post('https://localhost:44320/api/course/GetCourseByName',data)
.subscribe((res)=>{
console.log(res);
this.course=[res];
},err=>{
this.toastr.error('something error');
})
}
```

How to deal with the Search function

Step two: Add the template of the search form in the HTML file of the Manage course component.

```
<form class="form-inline" style="margin-top: 3%;">
    <div class="form-group mx-sm-3 mb-2">
    <label for="inputText" class="sr-`
    only">Course Name</label>
    <input type="text" class="form-control"
    id="inputText" placeholder="Course name"
    [value]="coursename" (change)="inputValue($event)">
    </div>
    <button type="submit"
    class="btn btn-success"
    (click)="search()">Search</button>
    </form>
```

How to deal with the Search function

Step three: Implement the search and InputValue functions In the managecourse.component.ts

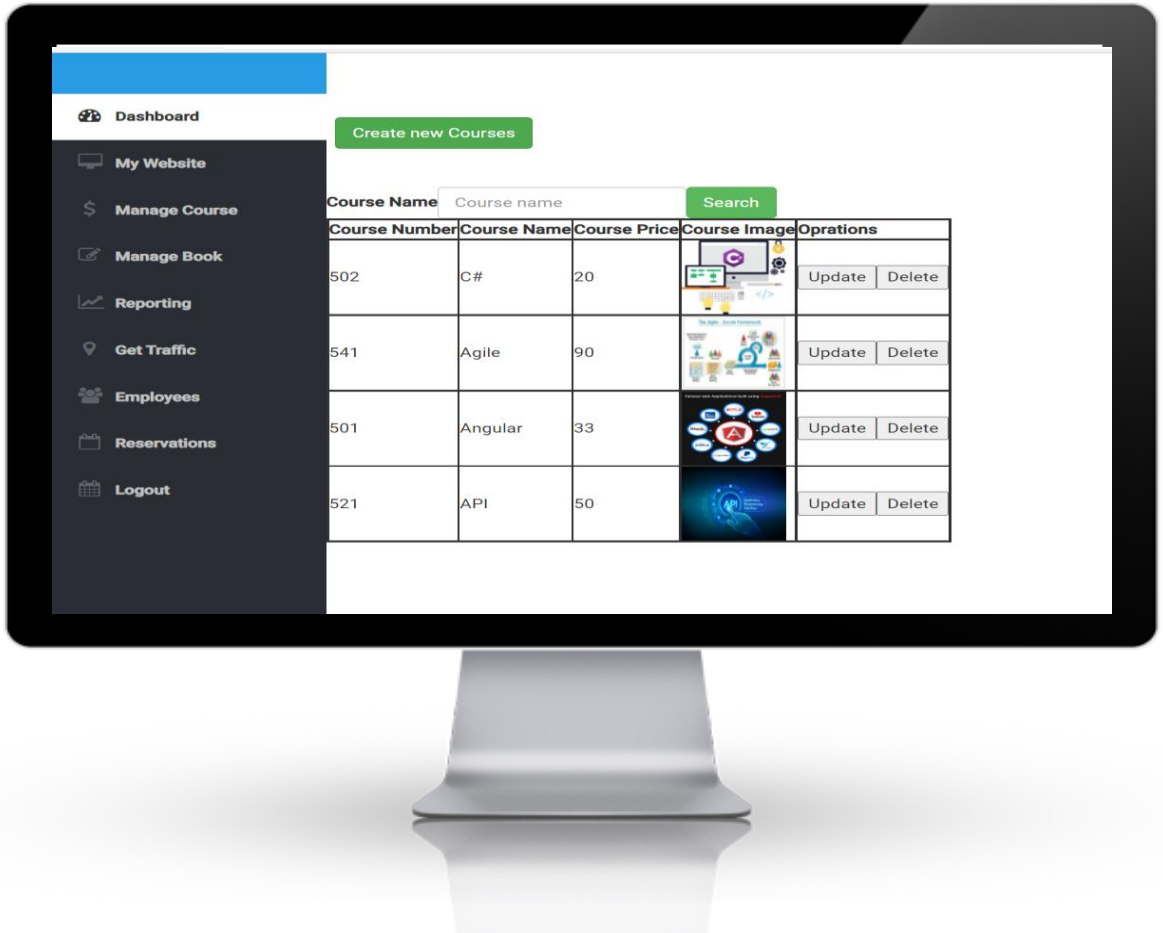
```
coursename:any='';
inputValue(ev:any){
  this.coursename=ev.target.value;
  console.log(ev.target.value);

}
search(){
  const courseobj=
  {
  coursename:this.coursename.toString()
  };
  debugger;
  this.home.searchCourse(courseobj);
  }
```

Harmony IT Solution

How to deal with the Search function

The Result:



References

[1] Angular, “Angular,” *Angular.io*, 2019. https://angular.io/

[2] “Complete Angular Tutorial For Beginners,” *TekTutorialsHub*. https://www.tektutorialshub.com/angular-tutorial/

[3]“npm | build amazing things,” *Npmjs.com*, 2019. https://www.npmjs.com/

[4]“Angular Tutorial for Beginners | Simplilearn,” *Simplilearn.com*. https://www.simplilearn.com/tutorials/angular-tutorial (accessed Aug. 19, 2022).

Harmony IT Solution