# Editemployee.html

<h2>Edit Employee</h2>

<form [formGroup]="edit\_emp" (ngSubmit)="onSubmit(edit\_emp.value)">

    <div>

        <label for="emp\_name"><b>Name</b></label><br>

        <input type="name" formControlName="name" id="emp\_name">

        <div class="error" \*ngIf="edit\_emp.controls['name'].errors?.minlength">Minimum length of name is 4.</div>

        <div class="error" \*ngIf="edit\_emp.controls['name'].errors?.maxlength">Maximum length of name is 20.</div>

        <div class="error" \*ngIf="edit\_emp.controls['name'].invalid && edit\_emp.controls['name'].touched">Enter valid name.</div>

        <br><br>

    </div>

    <div>

        <label for="emp\_salary"><b>Salary</b></label><br>

        <input type="number" formControlName="salary" id="emp\_salary">

        <div class="error" \*ngIf="edit\_emp.controls['salary'].errors?.min">Minimum salary should be 5000/-</div>

        <div class="error" \*ngIf="edit\_emp.controls['salary'].errors?.max">Maximum salary should be 50000/-</div>

        <div class="error" \*ngIf="edit\_emp.controls['salary'].invalid && edit\_emp.controls['salary'].touched">Enter valid salary.</div>

        <br><br>

    </div>

    <div>

        <b>Is Permanent or not?</b><br>

        <input type="radio" formControlName="permanent" id="emp\_true" [value]="true">

        <label for="emp\_true">Yes</label>

        <input type="radio" formControlName="permanent" id="emp\_false" [value]="false">

        <label for="emp\_false">No</label>

        <div class="error" \*ngIf="edit\_emp.controls['permanent'].invalid && edit\_emp.controls['permanent'].touched">Permanent field is required.</div>

        <br><br>

    </div>

    <div>

        <h4>Department</h4>

        <select formControlName="department">

        <option \*ngFor="let department of departments" [ngValue]="department" [selected]="department.id === employee.department.id">{{department.name}}</option>

        </select>

        <br><br>

    </div>

    <div>

        <h4>Additional Skills</h4>

        <div \*ngFor="let item of skills;let i=index">

            <input [value]="skills[i].name" type="checkbox" (change)="onCheckBoxChange($event)">{{skills[i].name}}

        </div>

        <br>

    </div>

    <button type="submit" [disabled]="!edit\_emp.valid">SAVE</button>

</form>

<div class="block">

    <strong>Employee Model</strong><br>

    <pre>{{employee | json }}</pre>

</div>

<div class="block" style="margin-right: 50px;">

    <strong>Angular Generated Form Model</strong><br>

    <pre>{{edit\_emp.value | json}}</pre>

</div>

# EditEmployee.ts

import { Component, OnInit } from '@angular/core';

import { Employee } from 'Models/Employee';

import { Department } from 'Models/Department';

import { Skills } from 'Models/Skills';

import { FormGroup, FormControl, Validators, FormArray, FormBuilder } from '@angular/forms';

import { ActivatedRoute } from '@angular/router';

import { EmployeeService } from '../employee.service';

@Component({

  selector: 'app-edit-emp-reactive',

  templateUrl: './edit-emp-reactive.component.html',

  styleUrls: ['./edit-emp-reactive.component.css']

})

export class EditEmpReactiveComponent implements OnInit {

  //Creating array for including departments.

  departments:Department[]=[];

  //Creating array for including skills.

  skills:Skills[]=[];

  //Creating an object to store employee details.

  employee:any={};

  edit\_emp: FormGroup;

  constructor(public fb:FormBuilder,private route:ActivatedRoute,private obj:EmployeeService)

  {

    //Fetching out the URL'S parameter.

    const employeeId=Number(this.route.snapshot.paramMap.get('id'));

    //Fetching the employee having id same as employeeId.

    this.employee=this.obj.getEmployee(employeeId);

     this.edit\_emp=this.fb.group(

      {

        name:[this.employee.name,[Validators.required,Validators.minLength(4),Validators.maxLength(20)]],

        salary:[this.employee.salary,[Validators.required,Validators.min(5000),Validators.max(50000)]],

        permanent:[this.employee.permanent,[Validators.required]],

        department:[this.employee.department,[Validators.required]],

        empskills:new FormArray([],[Validators.required])

      });

      this.skills=

      [

        {id:4,name:"Kotlin"},

        {id:5,name:"AI"},

        {id:6,name:"ML"},

        {id:7,name:"IOT"},

        {id:8,name:"Node JS"},

        {id:9,name:"Graphics Design"},

        {id:10,name:"FireBase"},

        {id:11,name:"Flutter"}

      ];

  }

  ngOnInit(): void

  {

    this.departments=

    [

      {id:1,name:"PayRoll"},

      {id:2,name:"Internal"},

      {id:3,name:"HR"}

    ];

    //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Note\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

    //Fetch the employee having requisite id inside constructor.

    //Because if you fetch here, then the form data won't be populated with correct employee details.

    //The above happens because we are initialising the form variable inside constructor.

  }

  //Method to be executed on submission of form for displaying form details.

  onSubmit(data:any):void

  {

    console.log(data);

  }

  //Method to record checkboxes choices.

  onCheckBoxChange(e:any):void

  {

    const temp=<FormArray> this.edit\_emp.get('empskills');

    //If a checkbox(e.target) is checked, push its value to temp array.

    if(e.target.checked)

    temp.push(new FormControl(e.target.value));

    //If a checkbox(e.target) is unchecked, pop its value from temp array.

    else

    {

      let i:number=0;

      temp.controls.forEach((item)=>{

        if(item.value==e.target.value)

        {

          temp.removeAt(i);

          return;

        }

        i++;

      });

    }

  }

}

# Output:

# 