App Component.html

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>

<link href='https://fonts.googleapis.com/css?family=Sail' rel='stylesheet'>

</head>

<div class="jumbotron jumbotron-fluid " >

<!-- <h4 class="display-5 text-center">

<img src="../assets/img/food.png" class="img-responsive" width="100px" height="120px">

<b>Fun & Food Restaurant</b></h4> -->

<!-- <div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

</div> -->

<div id="menu-nav " >

<div id="navigation-bar" >

<ul>

<li id="li"><a [routerLink]="['/order']" routerLinkActive="active"><span> <img src="../assets/img/food.png" class="img-responsive" width="80px" height="80px">

<h5 id="titleH5" ><b>Food Restaurant</b></h5>

</span></a></li>

<li><a [routerLink]="['/Home']"><i class="fa fa-home"></i><span>Home</span></a></li>

<li><a [routerLink]="['/order']" routerLinkActive="active"><i class="fa fa-rss" ></i><span>Order</span></a></li>

<li><a [routerLink]="['/Contact']" routerLinkActive="active"><i class="fa fa-book"></i><span>Contact</span></a></li>

<li><a [routerLink]="['/register']" routerLinkActive="active"><i class="fa fa-user"></i> <span>Register</span></a></li>

</ul>

</div>

</div>

<!-- <div class="container-fluid">

<div class="row"> -->

<div class="col-md-12 ">

<!-- <app-order></app-order> -->

<router-outlet></router-outlet>

<!-- </div>

</div> -->

</div>

</div>

App Routing.ts

import { NgModule } from '@angular/core';

import { Routes, RouterModule } from '@angular/router';

import { OrdersComponent } from './orders/orders.component';

import { empty } from 'rxjs';

import { OrderComponent } from './orders/order/order.component';

import { RegisterComponent } from './register/register.component';

import { HomeComponent } from './home/home.component';

import { ContactComponent } from './contact/contact.component';

import { LoginComponent } from './login/login.component';

const routes: Routes = [

// For default Rout

{path:'',redirectTo:'order',pathMatch:'full'},

{ path:'orders',component:OrdersComponent},

{ path:'order',children:[

{path:'',component:OrderComponent},

{path:'edit/:id',component:OrderComponent} ]},

{path:'orders',component:RegisterComponent},

{ path: 'register', component:RegisterComponent},

{ path: 'Home', component:HomeComponent},

{ path: 'Contact', component:ContactComponent},

{ path: 'login', component:LoginComponent},

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

Orders.html

<table class=" table table-hover">

<thead class="thead-light">

<th> # Order Number </th>

<th> Table Number </th>

<th> Payment Method </th>

<th> Grant Total </th>

<th>

<input type="text" name="GTotal" [(ngModel)]=GTotal id="search" placeholder="Enter the Price" class="form-control"/>

</th>

<th><a class="btn btn-outline-success" routerLink="/order" style="float:right"><i class="fa fa-plus-square " ></i> Create New</a></th>

</thead>

<tbody>

<tr \*ngFor="let item of orderList|TotalPipes:GTotal">

<th (click)="openForEdit(item.orderId)">{{item.OrderNumber}}</th>

<th>{{item.Customer}}</th>

<th>{{item.PayMethod}}</th>

<th> {{item.GTotal}}</th>

<th> <a class="btn text-danger" (click)="onOrderDelete(item.orderId)" ><i class="fa fa-trash fa-lg"></i></a></th>

</tr>

</tbody>

</table>

Orders.ts

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

import { OrderService } from '../shared/order.service';

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

import { ToastrService } from 'ngx-toastr';

@Component({

selector: 'app-orders',

templateUrl: './orders.component.html',

styles: []

})

export class OrdersComponent implements OnInit {

orderList;

constructor(private service:OrderService,

private router:Router,

private toastr:ToastrService) { }

ngOnInit() {

this.refreshList();

}

refreshList()

{

this.service.getOrderList().then(res=>this.orderList=res);

}

openForEdit(orderId:number)

{

this.router.navigate(['/order/edit/'+orderId])

}

onOrderDelete(id:number)

{

if(confirm('Are you sure to delete this record')){

this.service.deleteOrder(id).then(res=>{

this.refreshList();

this.toastr.warning("Deleted Successfully"," Food Restaurant");

});

}

}

}

Order.html

<form class="wraper" #form="ngForm" autocomplete="off" \*ngIf="service.formData" (submit)="onSubmit(form)">

<input type="hidden"name="orderId" #orderId="ngModel" [(ngModel)]="service.formData.orderId">

<div class="row">

<div class="col-md-6">

<div class="form-group">

<label>

Order Number

</label>

<div class="input-group">

<div class="input-group-prepend">

<div class="input-group-text">#</div>

</div>

<input name="OrderNumber" #OrderNumber="ngModel" [(ngModel)]="service.formData.OrderNumber" class="form-control"readonly/>

</div>

</div>

<div class="form-group">

<label>

Table Number

</label>

<select name="customerId" #customerId="ngModel" [(ngModel)]="service.formData.customerId" class="form-control" [class.is-invalid]="!isValid && service.formData.customerId==0">

<option value="0" >-Select-</option>

<option \*ngFor="let item of customerList" value="{{item.customerId}}">{{item.Name}}</option>

</select>

<!-- <input type="text" name="CustomerId" #CustomerId="ngModel" [(ngModel)]="service.formData.CustomerId" class="form-control" [class.is-invalid]="!isvalid && service.formData.CustomerId==0"> -->

</div>

</div>

<div class="col-md-6">

<div class="form-group">

<label>

Payment Method

</label>

<select name="PayMethod" #PayMethod="ngModel" [(ngModel)]="service.formData.PayMethod" class="form-control">

<option value="" >-Select-</option>

<option value="Cash" >Cash</option>

<option value="Card" >Card</option>

</select>

</div>

<div class="form-group">

<label>

Grand Total

</label>

<div class="input-group">

<div class="input-group-prepend">

<div class="input-group-text">Rs.</div>

</div>

<input name="GTotal" #GTotal="ngModel" [(ngModel)]="service.formData.GTotal" class="form-control" readonly/>

</div>

</div>

</div>

<!--order items table-->

<table class="table tabletable--border-less col-md-12">

<thead class="thead-light col-md-10">

<th>Food</th>

<th>Price</th>

<th>Quantity</th>

<th>Total</th>

<th><a class="btn btn-sm btn-success text-white" (click)="AddOrEditOrderItems(null,service.formData.orderId)"><i class="fa fa-plus"></i>Add Item</a></th>

</thead>

<tbody>

<tr \*ngIf="service.orderItem.length==0" [class.text-danger]="!isValid && service.orderItem.length==0">

<td class="font-italic text-center" colspan="5">

No food Item selected for this Order

</td>

</tr>

<tr \*ngFor="let item of service.orderItem;let i=index;">

<td>{{item.ItemName}}</td>

<td>{{item.Price}}</td>

<td>{{item.quantity}}</td>

<td>{{item.Total}}</td>

<td><a class="btn btn-sm btn-info text-white" (click)="AddOrEditOrderItems(i,service.formData.orderId)"><i class="fa fa-pencil"></i></a></td>

<td><a class="btn btn-sm btn-danger text-white ml-1" (click)="onDeleteOrderItem(item.orderItemId,i)"><i class="fa fa-trash"></i></a></td>

</tr>

</tbody>

</table>

<div class="form-group">

<button type="submit" class="btn btn-dark"><i class="fa fa-database" >Submit</i></button>

<a class="btn btn-outline-dark ml-1" routerLink="/orders"><i class="fa fa-table"></i> View Orders</a>

</div>

</div>

</form>

Order.ts

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

import { OrderService } from 'src/app/shared/order.service';

import { NgForm } from '@angular/forms';

import { MatDialog, MatDialogConfig } from '@angular/material/dialog';

import { OrderItemsComponent } from '../order-items/order-items.component';

// import { currentId } from 'async\_hooks';

import { CustomerService } from 'src/app/shared/customer.service';

import { Customer } from 'src/app/shared/customer.model';

import { ToastrService } from 'ngx-toastr';

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

@Component({

selector: 'app-order',

templateUrl: './order.component.html',

styles: []

})

export class OrderComponent implements OnInit {

customerList:Customer[];

isValid:boolean=true;

constructor(private service:OrderService,

private dialog:MatDialog,

private customerService:CustomerService,

private toastr:ToastrService,

private router:Router,

private currentRoute:ActivatedRoute){}

ngOnInit() {

let orderId=this.currentRoute.snapshot.paramMap.get('id');

if(orderId==null)

this.resetForm();

else

{

this.service.getOrderById(parseInt(orderId)).then(res=>

{

this.service.formData=res.order;

this.service.orderItem=res.orderDetails;

});

}

this.customerService.getCustomerList().then(res=>this.customerList=res as Customer[]);

}

resetForm(form?:NgForm){

if(form=null)

form.resetForm();

this.service.formData={

orderId:null,

OrderNumber:Math.floor(100000+Math.random()\*900000).toString(),

customerId:0,

PayMethod:'',

GTotal:0,

DeletedOrderItemIDs:''

};

this.service.orderItem=[];

}

AddOrEditOrderItems(orderItemIndex,orderId)

{

const dialogConfig=new MatDialogConfig();

dialogConfig.autoFocus=true;

dialogConfig.disableClose=true;

dialogConfig.width="50%";

dialogConfig.data={orderItemIndex,orderId}

this.dialog.open(OrderItemsComponent,dialogConfig).afterClosed().subscribe(res=>{

this.updateGrandTotal();

});

}

onDeleteOrderItem(orderItemId:number,i:number)

{

if(orderItemId!=null)

this.service.formData.DeletedOrderItemIDs+=orderItemId+",";

this.service.orderItem.splice(i,1);

this.updateGrandTotal();

}

updateGrandTotal()

{

this.service.formData.GTotal=this.service.orderItem.reduce((prev,curr)=>{

return prev+curr.Total;

},0);

this.service.formData.GTotal=parseFloat(this.service.formData.GTotal.toFixed(2));

}

validateForm()

{

this.isValid=true;

if(this.service.formData.customerId==0)

this.isValid=false;

else if(this.service.orderItem.length==0)

this.isValid=false;

return this.isValid;

}

onSubmit(form:NgForm)

{

if(this.validateForm())

{

this.service.saveOrUpdateOrder().subscribe(res=>{

this.resetForm();

this.toastr.success('Submitted Successfully','Food Restaurant');

this.router.navigate(['/orders']);

})

}

}

}

Order-Item.html

<h4 class="display-4">Order Food Item</h4>

<hr>

<form #form="ngForm" autocomplete="off" (submit)="onSubmit(form)">

<input type="hidden" name="orderItemId" #orderItemId="ngModel" [(ngModel)]="formData.orderItemId"/>

<input type="hidden" name="orderId" #orderId="ngModel" [(ngModel)]="formData.orderId"/>

<input type="hidden" name="ItemName" #ItemName="ngModel" [(ngModel)]="formData.ItemName"/>

<div class="form-group">

<label>Food Item</label>

<select name="Itemid" #Itemid="ngModel" [(ngModel)]="formData.Itemid" class="form-control" (change)="updatePrice($event.target)" [class.is-invalid]="!isvalid && formData.Itemid==0">

<option value="0" >-Select-</option>

<option \*ngFor="let item of itemList" value="{{item.Itemid}}">{{item.ItemName}}</option>

</select>

</div>

<div class="form-row">

<div class="form-group col-md-6">

<label>Price</label>

<div class="input-group">

<div class="input-group-prepend">

<div class="input-group-text">Rs.</div>

</div>

<input name="Price" #Price="ngModel" [(ngModel)]="formData.Price" class="form-control" readonly/>

</div>

</div>

<div class="form-group col-md-6">

<label>Quantity</label>

<div class="input-group">

<input name="quantity" #quantity="ngModel" [(ngModel)]="formData.quantity" class="form-control" (keyup)="updateTotal()" [class.is-invalid]="!isvalid && formData.quantity==0"/>

</div>

</div>

</div>

<div class="form-group col-md-6">

<label>Total</label>

<div class="input-group">

<div class="input-group-prepend">

<div class="input-group-text">Rs.</div>

</div>

<input name="Total" #Total="ngModel" [(ngModel)]="formData.Total" class="form-control" readonly/>

</div>

</div>

<div class="form-group">

<button type="submit" class="btn btn-dark"><i class="fa fa-database">Submit</i></button>

<button type="button" class="btn btn-outline-dark ml-1" [mat-dialog-close]><i class="fa fa-close" >Close</i></button>

</div>

</form>

Order-item.ts

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

import{MAT\_DIALOG\_DATA,MatDialogRef} from "@angular/material";

import { OrderItem } from 'src/app/shared/order-item.model';

import { ItemService } from 'src/app/shared/item.service';

import { Item } from 'src/app/shared/item.model';

import { NgForm } from '@angular/forms';

import { OrderService } from 'src/app/shared/order.service';

@Component({

selector: 'app-order-items',

templateUrl: './order-items.component.html',

styles: []

})

export class OrderItemsComponent implements OnInit {

formData:OrderItem;

itemList:Item[];

isValid:boolean=true;

constructor(

@Inject(MAT\_DIALOG\_DATA) public data,

public dialogRef:MatDialogRef<OrderItemsComponent>,private itemService:ItemService,

private orderService:OrderService){}

ngOnInit() {

this.itemService.getItemList().then(res=>this.itemList=res as Item[]);

if(this.data.orderItemIndex==null)

this.formData={

orderItemId:null,

orderId:this.data.orderId,

Itemid:0,

ItemName:'',

Price:0,

quantity:0,

Total:0

}

else

this.formData=Object.assign({},this.orderService.orderItem[this.data.orderItemIndex]);

}

updatePrice(ctrl)

{

if(ctrl.selectedIndex==0)

{

this.formData.Price=0;

this.formData.ItemName='';

}else{

this.formData.Price=this.itemList[ctrl.selectedIndex-1].Price;

this.formData.ItemName=this.itemList[ctrl.selectedIndex-1].ItemName;

}

this.updateTotal();

}

updateTotal()

{

this.formData.Total=parseFloat((this.formData.quantity\*this.formData.Price).toFixed(2));

}

onSubmit(form:NgForm)

{

if(this.validateForm(form.value))

{

if(this.data.orderItemIndex==null)

this.orderService.orderItem.push(form.value);

else

this.orderService.orderItem[this.data.orderItemIndex]=form.value;

this.dialogRef.close();

}

}

validateForm(formData:OrderItem)

{

this.isValid=true;

if(formData.Itemid==0)

this.isValid=false;

else if(formData.quantity==0)

this.isValid=false;

return this.isValid;

}

}

Login.html

            <div class="card"><br/><h3 id="title"><b>Login</b></h3><br/>

              <div class="card-body">

                  <div class="glowBox">

<div class="bg-img" class="col-md-6 offset-md-3" >

<div class="container-fluid">

<form #form="ngForm" (submit)="login(form)" autocomplete="off" id="form">

<div class="form-group" [class.has-error]="Email.invalid && Email.touched"> <label>Email Id \*:</label>

<input type="text" name="Email" placeholder="Email" #Email="ngModel" [(ngModel)]="this.service.formData.Email" class="form-control" required Email >

<span class="help-block" \*ngIf="Email.errors?.required && Email.touched"> Email is Required</span>

</div>

<div class="form-group" [class.has-error]="Password.invalid && Password.touched"> <label>Password \*:</label>

<input type="password" name="Password" placeholder="Password" #Password="ngModel" [(ngModel)]="this.service.formData.Password" #Password="ngModel"

class="form-control" required><span class="help-block" \*ngIf="Password.errors?.required && Password.touched">

Password is Required

</span>

</div>

<div class="form-group">

<a [routerLink]="['/register']" routerLinkActive="active" ><b>New user? Create your Account</b></a>

</div>

<div class="form-group">

<button type="submit" class="btn btn-lg btn-block"> <b>Login</b> </button>

</div>

</form>

          </div>

          </div>

        </div>

      </div>

</div>

Login.ts

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

import { NgForm } from '@angular/forms';

import { RegisterService } from '../shared/register.service';

import { Register} from '../shared/register.model';

import { ToastrService } from 'ngx-toastr';

import { Route, Router } from '@angular/router';

@Component({

selector: 'app-login',

templateUrl: './login.component.html',

styleUrls: ['./login.component.css']

})

export class LoginComponent implements OnInit {

// public Register:loginarray[];

loginarray: Register[];

constructor(private service:RegisterService,private toastr:ToastrService,private router:Router) { }

ngOnInit() {

this.service.refreshList();

this.service.formData={

rId:null,

firstName:null,

lastName:null,

Email :null,

Password:null,

cPassword:null

}

}

login(form:NgForm)

{

console.log(form.value.Email);

console.log(form.value.Password);

this.loginarray=this.service.list.filter(e=>e.Email==form.value.Email && e.Password==form.value.Password);

if(this.loginarray.length==0)

{

this.toastr.success("Something went wrong","invalid User");

}

else

{

this.toastr.success("Login Successfully","Valid User");

this.router.navigate(['/order']);

}

}

}

Login.css

/\* <!-- Bootstrap Scripts --> \*/

html,

body,

header {

height:100%;

  width: 150%;

margin-top:1px;height:20px;

  font-family: "Open Sans", sans-serif;

  transition: all 0.3s ease-in-out;

  line-height: 1.15;

  -webkit-text-size-adjust: 100%;

  -ms-text-size-adjust: 100%;

  /\* -ms-overflow-style: scrollbar; \*/

  -webkit-tap-highlight-color: transparent;border:1px solid red;

}

#title{

text-align: center;

}

textarea:focus,

input:focus {

  outline: none;

}

/\* .bgImg {

  background-image: url("https://wallpapers.wallhaven.cc/wallpapers/full/wallhaven-371678.jpg");

  background-size: cover;

  display: block;

  filter: grayscale(75%);

  -webkit-filter: grayscale(75%);

  height: 100%;

  left: 0;

  position: fixed;

  right: 0;

  z-index: -999;

} \*/

#containerSize

{

  width:600px;border:2px solid black;

  /\* padding: 15px 15px 15px 15px; \*/

border-radius: 10px;

}

.card {

/\* margin-left:290px; \*/

  width: 100%;margin-top: 10px;margin-bottom: 100px;

  /\* height: 61%; \*/

  /\* margin-top: 10px; \*/

  /\* margin-bottom: 150px; \*/

  /\* margin-top: 100px; \*/

  /\* padding: 0px 15px 10px 15px; \*/

  background-color: rgba(255, 255, 255, 0.8);

  -webkit-box-shadow: 5px 6px 20px -5px rgba(0, 0, 0, 0.75);

  -moz-box-shadow: 5px 6px 2px -5px rgba(0, 0, 0, 0.75);

  box-shadow: 5px 6px 20px -5px rgba(0, 0, 0, 0.75);

  /\* border-radius: 6px !important; \*/

}

.glowBox {

  background: linear-gradient(135deg, #ce9ffc, #7367f0);

  margin-top: -40px;

  padding: 20px 0;

  width: 100%;

  margin-bottom: 15px;

  -webkit-box-shadow: 0 5px 20px 0 rgba(0, 0, 0, 0.2),

    0 13px 24px -11px rgba(204, 200, 240, 0.6);

  box-shadow: 0 5px 20px 0 rgba(243, 229, 229, 0.2),

    0 13px 24px -11px rgba(64, 58, 124, 0.6);

  border-radius: 16px;

}

.btn-block

{

border:1px solid black;

}

.desc {

  color: #888888;

  font-size: 14px;

  margin: 0 0 10px;

}

.form {

  display: block;

  width: 100%;height:100%;

  padding: 0.4375rem 0;

  font-size: 1rem;

  line-height: 1.5;

  color: #495057;

  background-color: transparent;

  background-clip: padding-box;

  border: 0;

  border-bottom: 1px solid #d2d2d2;

  border-radius: 0;

  -webkit-box-shadow: none;

  box-shadow: none;

  -webkit-transition: border-color 0.15s ease-in-out,

    -webkit-box-shadow 0.15s ease-in-out;

  transition: border-color 0.15s ease-in-out,

    -webkit-box-shadow 0.15s ease-in-out;

  transition: border-color 0.15s ease-in-out, box-shadow 0.15s ease-in-out;

  transition: border-color 0.15s ease-in-out, box-shadow 0.15s ease-in-out,

    -webkit-box-shadow 0.15s ease-in-out;

  padding-bottom: 7px;

  margin: 27px 0 0 0;

}

input::-webkit-input-placeholder {

  color: #aaa;

}

input:focus::-webkit-input-placeholder {

  color: #847dd3;

}

.form:focus + .underline {

  transform: scale(1);

}

.underline {

  background-color: #817ad3;

  display: inline-block;

  height: 2px;

  left: 0px;

  margin-top: -4px;

  position: relative;

  top: -18px;

  -webkit-transform: scale(0, 1);

  transform: scale(0, 1);

  -webkit-transition: all 0.25s linear;

  transition: all 0.25s linear;

  width: 100%;

}

.btn.btn-primary.btn-link {

  background-color: transparent;

  color: #c3bff8;

  -webkit-box-shadow: none;

  box-shadow: none;

  border: none;

  cursor: pointer;

  text-decoration: none;

  padding: 1.125em 2.25em;

font-size: 1em;

line-height: 1.5;

border-radius: .2em;

}

Register.html

            <div class="card"><br/><h3 id="title"><b>Registration</b></h3><br/>

              <div class="card-body">

                  <div class="glowBox">

<div class="bg-img" class="col-md-6 offset-md-3" >

<div class="container-fluid" >

<form #form="ngForm" (submit)="OnSubmit(form)" autocomplete="off" id="form">

<div class="form-group">

<label>Register Id:</label>

<input value="1"class="form-control" placeholder="Register Id" id="rId" name="rId" [(ngModel)]="this.service.formData.rId" #rId="ngModel" disabled>

</div>

<!-- <div class="form-group">

<label> First Name \*:</label>

<input class="form-control" id="firstName" name="firstName" [(ngModel)]="this.service.formData.firstName" #firstName="ngModel">

</div> -->

<div class="form-group" [class.has-error]="firstName.invalid && firstName.touched" [class.has-success]="name.valid" >

<label for="firstName">First Name \*:</label>

<input required type="text" name="firstName" [(ngModel)]="this.service.formData.firstName"

#firstName="ngModel" id="firstName" class="form-control"placeholder="First Name" >

<span class="help-block" \*ngIf="firstName.invalid && firstName.touched">

Name is Required

</span>

</div>

<div class="form-group">

<label> Last Name:</label>

<input class="form-control" placeholder="Last Name"id="lastName" name="lastName" [(ngModel)]="this.service.formData.lastName" #lastName="ngModel">

</div>

<div class="form-group" [class.has-error]="Email.invalid && Email.touched"> <label>Email Id \*:</label>

<input type="text" name="Email" placeholder="Email" [(ngModel)]="this.service.formData.Email" #Email="ngModel"

class="form-control" required email><span class="help-block" \*ngIf="Email.errors?.required && Email.touched">

Email is Required

</span>

</div>

<!-- <div class="form-group">

<label> Password \*</label>

<input type="password"placeholder="Password" class="form-control" name="Password" [(ngModel)]="this.service.formData.Password" #Password="ngModel">

</div> -->

<div class="form-group" [class.has-error]="Password.invalid && Password.touched"> <label>Password \*:</label>

<input type="password" name="Password" placeholder="Password" [(ngModel)]="this.service.formData.Password" #Password="ngModel"

class="form-control" required><span class="help-block" \*ngIf="Password.errors?.required && Password.touched">

Password is Required

</span>

</div>

<div class="form-group" [class.has-error]="cPassword.invalid && cPassword.touched">

<label>Confirm Password \*</label>

<input type="password" required placeholder="Confirm Password" class="form-control" name="cPassword" [(ngModel)]="this.service.formData.cPassword" #cPassword="ngModel">

<span class="help-block" \*ngIf="cPassword.errors?.required && cPassword.touched">

Confirm Password is Required

</span>

</div>

<div class="form-group">

<a [routerLink]="['/login']" routerLinkActive="active" ><b>Already have a Account?</b></a>

</div>

<div class="form-group">

<button type="submit" class="btn btn-lg btn-block"> <b>Register</b> </button>

</div>

</form>

          </div>

          </div>

        </div>

      </div>

</div>

Register . ts

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

// import { Register } from '../shared/register.model';

// import { ToastrService } from 'ngx-toastr';

// import { RegisterService } from '../shared/register.service';

// import { NgForm } from '@angular/forms';

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

import { RegisterService } from '../shared/register.service';

import { NgForm, NgFormSelectorWarning } from '@angular/forms';

import { MockNgModuleResolver } from '@angular/compiler/testing';

import {HttpClient} from '@angular/common/http';

import { Register } from '../shared/register.model';

import { ToastrService } from 'ngx-toastr';

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

@Component({

selector: 'app-register',

templateUrl: './register.component.html',

styleUrls: ['./register.component.css']

})

export class RegisterComponent implements OnInit {

public RId:number[];

public result:any;

registerid:number;

reg:Register;

constructor(private service:RegisterService,private toastr:ToastrService,private router:Router) {

this.service.formData={

rId:null,

firstName:null,

lastName:null,

Email:null,

Password:null,

cPassword:null

}

this.result=this.service.list;

}

ngOnInit(){

this.resetForm();

}

OnSubmit(form:NgForm)

{

{{debugger}}

console.log(form.value.rId);

console.log(form.value.firstName);

console.log(form.value.lastName);

console.log(form.value.Email);

console.log(form.value.Password);

console.log(form.value.cPassword);

// this.RId=form.value.rId;

// this.reg=this.service.list.find(e=>e.rId==this.registerid);

if(this.reg==null)

// if(form.value.eid=this.employeeid)

{

this.InsertRecord(form);

this.router.navigate(['/login']);

}

else

{

this.UpdateRecord(form);

}

}

InsertRecord(form:NgForm)

{

this.service.postRegister(form.value).subscribe(res=>{

this.toastr.success("Inserted Successfully","You are Registered Successfully");

this.resetForm(form)

this.service.refreshList();

});

}

UpdateRecord(form:NgForm)

{

this.service.putRegister(form.value).subscribe(res=>{

this.toastr.warning("Updated Successfully","You are Registered Successfully");

this.resetForm(form)

this.service.refreshList();

});

}

resetForm(form?:NgForm)

{

if(form!=null)

{

form.resetForm();

this.service.formData={

rId:null,

firstName:null,

lastName:null,

Email:null,

Password:null,

cPassword:null

}

}

}

}

Environment.ts

export const environment = {

production: false,

// production: false,

apiURL:'http://localhost:60453/api'

};

Home.html

<!DOCTYPE html>

<html>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<style>

body {font-family: "Times New Roman", Georgia, Serif;}

h1, h2, h3, h4, h5, h6 {

font-family: "Playfair Display";

letter-spacing: 5px;

}

</style>

<body>

<!-- Navbar (sit on top) -->

<!-- <div class="w3-top">

<div class="w3-bar w3-white w3-padding w3-card" style="letter-spacing:4px;">

<a href="#home" class="w3-bar-item w3-button">Gourmet au Catering</a>

Right-sided navbar links. Hide them on small screens -->

<!-- <div class="w3-right w3-hide-small">

<a href="#about" class="w3-bar-item w3-button">About</a>

<a href="#menu" class="w3-bar-item w3-button">Menu</a>

<a href="#contact" class="w3-bar-item w3-button">Contact</a>

</div>

</div>

</div> -->

<!-- Header -->

<header class="w3-display-container w3-content w3-wide" style="max-width:1600px;min-width:500px" id="home">

<img class="Container-fluid" src="../../assets/img/home1.jpg" alt="Hamburger Catering" width="1600" height="800">

<div class="w3-display-bottomleft w3-padding-large w3-opacity">

<h1 class="w3-xxlarge">Food Restaurant</h1>

</div>

</header>

<!-- Page content -->

<div class="w3-content" style="max-width:1100px">

<!-- About Section -->

<div class="w3-row w3-padding-64" id="about">

<div class="w3-col m6 w3-padding-large w3-hide-small">

<img src="../../assets/img/home2.jpg" class="w3-round w3-image w3-opacity-min" alt="Table Setting" width="600" height="750">

</div>

<div class="w3-col m6 w3-padding-large">

<h1 class="w3-center">About Food Restaurant</h1><br>

<h5 class="w3-center">Tradition since 1889</h5>

<p class="w3-large">A restaurant , or an eatery, is a business which prepares and serves food and drinks to customers in exchange for money. Meals are generally served and eaten on the premises, but many restaurants also offer take-out and food delivery services. Restaurants vary greatly in appearance and offerings, including a wide variety of cuisines and service models ranging from inexpensive fast food restaurants and cafeterias to mid-priced family restaurants, to high-priced luxury establishments.</p>

</div>

</div>

<hr>

<!-- Menu Section -->

<div class="w3-row w3-padding-64" id="menu">

<div class="w3-col l6 w3-padding-large">

<h1 class="w3-center">Our Menu</h1><br>

<h4>Bread Basket</h4>

<p class="w3-text-grey">Assortment of fresh baked fruit breads and muffins 5.50</p><br>

<h4>Honey Almond Granola with Fruits</h4>

<p class="w3-text-grey">Natural cereal of honey toasted oats, raisins, almonds and dates 7.00</p><br>

<h4>Belgian Waffle</h4>

<p class="w3-text-grey">Vanilla flavored batter with malted flour 7.50</p><br>

<h4>Scrambled eggs</h4>

<p class="w3-text-grey">Scrambled eggs, roasted red pepper and garlic, with green onions 7.50</p><br>

<h4>Blueberry Pancakes</h4>

<p class="w3-text-grey">With syrup, butter and lots of berries 8.50</p>

</div>

<div class="w3-col l6 w3-padding-large">

<img src="../../assets/img/home3.jpg" class="w3-round w3-image w3-opacity-min" alt="Menu" style="width:100%">

</div>

</div>

<hr>

<!-- Contact Section -->

<div class="w3-container w3-padding-64" id="contact">

<h1>Contact</h1><br>

<p>We offer full-service catering for any event, large or small. We understand your needs and we will cater the food to satisfy the biggerst criteria of them all, both look and taste. Do not hesitate to contact us.</p>

<p class="w3-text-blue-grey w3-large"><b>Food Restaurant Service, 42nd Living St, 43043 India, NY</b></p>

<p>You can also contact us by phone 00553123-2323 or email foodSupport@example.com.</p>

<!-- <form action="/action\_page.php" target="\_blank">

<p><input class="w3-input w3-padding-16" type="text" placeholder="Name" required name="Name"></p>

<p><input class="w3-input w3-padding-16" type="number" placeholder="How many people" required name="People"></p>

<p><input class="w3-input w3-padding-16" type="datetime-local" placeholder="Date and time" required name="date" value="2017-11-16T20:00"></p>

<p><input class="w3-input w3-padding-16" type="text" placeholder="Message \ Special requirements" required name="Message"></p>

<p><button class="w3-button w3-light-grey w3-section" type="submit">SEND MESSAGE</button></p>

</form> -->

</div>

<!-- End page content -->

</div>

<!-- Footer -->

<footer class="w3-center w3-light-grey w3-padding-32">

<p>Powered by <a [routerLink]="['/home']" title="Food Restaurant" target="\_blank" class="w3-hover-text-green">Food Restaurant</a></p>

</footer>

</body>

</html>

Contact.html

<div class="container col-md-6 col-md-offset-1">

<div class="row col-md-6" >

<address id="address">

<strong>Support:</strong> <p>foodSupport@example.com</p><br />

<strong>Issues:</strong> <p>foodHotel@example.com</p>

</address>

</div>

<div class="row col-md-6" >

<address id="address1">

<strong>Phone Number:</strong> <p>9999-9999-8899</p><br />

<strong>Issues:</strong> <p>foodHotel@example.com</p>

</address>

</div>

</div>

Contact.css

#address{

margin-top: 30px;

font-size:20px;float:right;margin-left:70%;

}

#address1{

float:left;

font-size:20px;margin-left:70%;

}

.container{

background-color: rgb(248, 237, 239);

padding-bottom: 100px;

border-radius: 20px;

}

Modal For customer used as Table Number

export class Customer {

customerId:number;

Name:string;

}

Order-item Modal

export class OrderItem {

orderItemId:number;

orderId:number;

Itemid :number;

quantity:number;

ItemName:string;

Price:number;

Total:number;

}

Order.modal

export class Order {

orderId:number;

OrderNumber:string;

customerId:number;

PayMethod :string;

GTotal:number;

DeletedOrderItemIDs:string;//For delete the record

}

Register.Modal

export class Register {

rId:number;

firstName:string;

lastName:string;

Email :string;

Password:string;

cPassword:string;//For delete the record

}

Total-pipes.ts

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

import {Pipe, PipeTransform} from '@angular/core';

import { Order } from './order.model';

@Pipe ({

name : 'TotalPipes'

})

// @Component({

// selector: 'app-total-pipes',

// templateUrl: './total-pipes.component.html',

// styleUrls: ['./total-pipes.component.css']

// })

export class TotalPipesComponent implements OnInit {

transform(total:Order[],gtotal:number) {

if(!total || !gtotal)

{

return total;

}

return total.filter(x=>x.GTotal>=gtotal);

}

constructor() { }

ngOnInit() {

}

}

Service For All Modal

import { Injectable } from '@angular/core';

import { HttpClient } from '@angular/common/http';

import { environment } from 'src/environments/environment';

@Injectable({

providedIn: 'root'

})

export class CustomerService {

constructor(private http:HttpClient) {

}

getCustomerList()

{

return this.http.get(environment.apiURL+'/Customers').toPromise()

}

}

import { Injectable } from '@angular/core';

// import { HttpClient } from 'selenium-webdriver/http';

import { environment } from 'src/environments/environment';

import{HttpClient} from '@angular/common/http';

@Injectable({

providedIn: 'root'

})

export class ItemService {

constructor(private http:HttpClient) { }

getItemList()

{

return this.http.get(environment.apiURL+'/Item').toPromise()

}

}

Style.css

/\* You can add global styles to this file, and also import other style files \*/

@import "~@angular/material/prebuilt-themes/indigo-pink.css";

.jumbotron{

padding-top:0px;

background: #fff;

}

.form-control,.input-group-text{

border:1px solid #0080ff;

}

table.table-hover tbody tr:hover{

cursor:pointer;

}

#orderForm

{

margin-left:300px;

margin-right: 300px;

}

#search{

width:140px;height:25px

}

.wraper{

width: 85%;

margin: 0 auto;

}

#menu-nav {

max-width: 100%;

max-height: 1500px;

margin: 50px auto;

background-color: #ecf0f1;

border-radius: 4px;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.3); font: normal 300 18px "Roboto", sans-serif;background: #e4c3de;

}

#navigation-bar ul {

border-radius: 4px;

overflow: hidden;

}

#navigation-bar li {

float: left;

width: 20%;

}

#navigation-bar li:hover a::before {

right: 0;

left: 0;

}

#navigation-bar a {

color: #0a0a0a;

font-size: 20px;

text-decoration: none;

text-align: center;

padding: 20px 0;

display: block;

position: relative;

transition: all 0.3s ease;

}

#navigation-bar a:hover {

color: #e74c3c;

}

#navigation-bar a::before {

content: "";

transition: all 0.4s ease-in-out;

position: absolute;

right: 50%;

left: 50%;

bottom: 0;

height: 2px;

background-color: #e74c3c;

}

#navigation-bar a:hover::before {

right: 0;

left: 0;

}

li {

list-style: none;

}

Index.html

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>AngularRestaurant</title>

<base href="/">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="icon" type="image/x-icon" href="favicon.ico">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

<link rel= "stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">

</head>

<body>

<app-root></app-root>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

</body>

</html>

Register .service

import { Injectable } from '@angular/core';

import {HttpClient, HttpHeaders, HttpParams} from '@angular/common/http';

import { Observable } from 'rxjs/internal/Observable';

import { getNumberOfCurrencyDigits } from '@angular/common';

import { Register } from './register.model';

import { environment } from 'src/environments/environment';

@Injectable({

providedIn: 'root'

})

export class RegisterService {

formData:Register;

list:Register[];

constructor(private http:HttpClient) { }

postRegister(formData:Register)

{

return this.http.post(environment.apiURL+'/Registers',formData);

}

refreshList():void

{

this.http.get(environment.apiURL+'/Registers')

.toPromise().then(res=>this.list=res as Register[]);

}

putRegister(formData:Register)

{

const headers = new HttpHeaders().set('content-type', 'application/json');{{debugger}}

const params = new HttpParams().set('ID', formData.rId.toString());

{

return this.http.put<Register>(environment.apiURL+'/Registers/'+formData.rId,formData,{headers,params});

}

}

deleteRegister(id:number)

{

return this.http.delete(environment.apiURL+'/Registers/'+id);

}

}

Order.service

import { Injectable } from '@angular/core';

import { Order } from './order.model';

import { OrderItemsComponent } from '../orders/order-items/order-items.component';

import { OrderItem } from './order-item.model';

import { HttpClient } from '@angular/common/http';

import { environment } from 'src/environments/environment';

@Injectable({

providedIn: 'root'

})

export class OrderService {

formData:Order;

orderItem:OrderItem[];

constructor(private http:HttpClient) { }

saveOrUpdateOrder()

{

var body =

{

...this.formData,

OrderItems:this.orderItem

};

return this.http.post(environment.apiURL+'/Ordertbls',body);

}

getOrderList()

{

return this.http.get(environment.apiURL+'/Ordertbls').toPromise()

}

getOrderById(id:number):any

{

return this.http.get(environment.apiURL+'/Ordertbls/'+id).toPromise()

}

deleteOrder(id:number)

{

return this.http.delete(environment.apiURL+'/Ordertbls/'+id).toPromise()

}

}