HTML

<div class="container mt-3" \*ngIf = "oneway.length">

    <div class='parent d-flex grid-parent'>

        <div class='child flex-fill'>

            <h2>{{oneway[0].fromLocation}} <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-arrow-right" viewBox="0 0 16 16">

                <path fill-rule="evenodd" d="M1 8a.5.5 0 0 1 .5-.5h11.793l-3.147-3.146a.5.5 0 0 1 .708-.708l4 4a.5.5 0 0 1 0 .708l-4 4a.5.5 0 0 1-.708-.708L13.293 8.5H1.5A.5.5 0 0 1 1 8z"/>

              </svg> {{oneway[0].toLocation}} </h2>

            <h3>{{oneway[0].departureDate.slice(0,10)}}</h3>

            <div class="box d-flex" \*ngFor="let o of oneway;let i = index;"

                [ngClass]="OWboolArray[i]? 'select': 'unselect'">

                <div id="details\_{{i}}" (click)="OWSelect(i,o)" class="box flex-fill">

                    {{o.flightNumber}}

                    <h4>{{o.departureTime}} -------------------- {{o.arrivalTime}}</h4>

                </div>

                <button id="0\_{{i}}" [disabled]="!(OWboolArray[i] && (o.availableEconomySeats >= pCount))" (click)="OWpriceSelect($event)" [ngClass]="OWbtnTrigger[0] && OWboolArray[i]?'btn btn-success': 'btn btn-primary'" [value]="o.economyPrice" class="box flex-fill">Economy Class: ₹ {{o.economyPrice}}</button>

                <button id="1\_{{i}}" [disabled]="!(OWboolArray[i] && (o.availableBusinessSeats >= pCount))" (click)="OWpriceSelect($event)" [ngClass]="OWbtnTrigger[1] && OWboolArray[i]?'btn btn-success': 'btn btn-primary'" [value]="o.businessPrice" class="box flex-fill">Business Class: ₹ {{o.businessPrice}}</button>

            </div>

        </div>

        <div class='child flex-fill' \*ngIf="flag && twoway.length" >

            <h2>{{twoway[0].fromLocation}} <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-arrow-right" viewBox="0 0 16 16">

                <path fill-rule="evenodd" d="M1 8a.5.5 0 0 1 .5-.5h11.793l-3.147-3.146a.5.5 0 0 1 .708-.708l4 4a.5.5 0 0 1 0 .708l-4 4a.5.5 0 0 1-.708-.708L13.293 8.5H1.5A.5.5 0 0 1 1 8z"/>

              </svg>  {{twoway[0].toLocation}} </h2>

            <h3>{{twoway[0].departureDate.slice(0,10)}}</h3>

            <div class="box d-flex" \*ngFor="let t of twoway; let i = index;" [ngClass]="RTboolArray[i]? 'select': 'unselect'">

                <div id="details\_{{i}}" (click)="RTSelect(i,t)" class="box flex-fill">

                    {{t.flightNumber}}

                    <h4>{{t.departureTime}} -------------------- {{t.arrivalTime}}</h4>

                </div>

                <button id="0\_{{i}}" [disabled]="!(RTboolArray[i] && (t.availableEconomySeats >= pCount))" (click)="RTpriceSelect($event)" [ngClass]="RTbtnTrigger[0] && RTboolArray[i]?'btn btn-success': 'btn btn-primary'" [value]="t.economyPrice" class="box flex-fill">Economy Class: ₹ {{t.economyPrice}}</button>

                <button id="1\_{{i}}" [disabled]="!(RTboolArray[i] && (t.availableBusinessSeats >= pCount))" (click)="RTpriceSelect($event)" [ngClass]="RTbtnTrigger[1] && RTboolArray[i]?'btn btn-success': 'btn btn-primary'" [value]="t.businessPrice" class="box flex-fill">Business Class: ₹ {{t.businessPrice}}</button>

            </div>

        </div>

    </div>

</div>

<div class="my-3 mx-auto w-50 d-grid gap-2" \*ngIf="!flag">

    <button class="btn btn-primary" [disabled]="!(OWselect)" (click)="Continue()">Continue</button>

</div>

<div class="my-3 mx-auto w-50 d-grid gap-2" \*ngIf="flag">

    <button class="btn btn-primary" [disabled]="!(OWselect && RTselect)"(click)="Continue()">Continue</button>

</div>

TS file

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

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

import { Seats } from 'src/app/models/seats';

import { SeatService } from 'src/app/services/seats.service';

@Component({

  selector: 'app-user-seat-selection',

  templateUrl: './user-seat-selection.component.html',

  styleUrls: ['./user-seat-selection.component.css']

})

export class UserSeatSelectionComponent implements OnInit {

    seats : Seats [];

    seatIndex : number;

    seatTogglers = new Array<boolean>().fill(false);

    element? : HTMLDivElement;

    temp : any;

    seatNoPush : string;

    seatIdPush : number;

    selectedSeats : string[] =[];

    selectedSeatsId : number[] =[];

    seatCount : number = 0;

    classBool : boolean;

    availableBool : boolean = false;

    freezeCheck : number = 0;

    OWflightNumber : string;

    isReturn : boolean;

    RTflightNumber : string;

    tripName : string;

    OWselectedSeats : string[] =[];

    OWselectedSeatsId : number[] =[];

    OWclassBool : boolean;

    skip : boolean = false;

    constructor(private service : SeatService,private route : ActivatedRoute, private router : Router) {

    }

    // SEAT SELECTION METHOD

    // Business Seats Allocation Method

    SeatB(event : any){

      this.seatIndex = Number(event.target.id);

      if(this.seatCount != 0 || this.seatTogglers[this.seatIndex] == true){

        if(this.seats[this.seatIndex].isAvailable){

          this.seatTogglers[this.seatIndex] = !this.seatTogglers[this.seatIndex];

          this.temp = document.getElementById(event.target.id);

          this.element = this.temp;

          if(this.seatTogglers[this.seatIndex]){

            this.element?.setAttribute("class",'seat selected');

            this.seatNoPush = event.target.attributes.name.nodeValue;

            this.seatIdPush = this.seats[this.seatIndex].seatId

            this.selectedSeatsId.push(this.seatIdPush);

            this.selectedSeats.push(this.seatNoPush);

            this.seatCount -= 1;

          }

          else{

            this.element?.setAttribute("class",'seat');

            this.seatNoPush = event.target.attributes.name.nodeValue;

            this.seatIdPush = this.seats[this.seatIndex].seatId

            this.selectedSeatsId.splice(this.selectedSeatsId.indexOf(this.seatIdPush),1)

            this.selectedSeats.splice(this.selectedSeats.indexOf(this.seatNoPush),1)

            this.seatCount += 1;

          }

        }

      }

      console.log(this.selectedSeats)

      console.log(this.selectedSeatsId)

    }

    // Economy Seats Allocation Method

    SeatE(event : any){

      this.seatIndex = Number(event.target.id);

      if(this.seatCount != 0 || this.seatTogglers[this.seatIndex] == true){

        if(this.seats[this.seatIndex].isAvailable){

          this.seatTogglers[this.seatIndex] = !this.seatTogglers[this.seatIndex];

          this.temp = document.getElementById(event.target.id);

          this.element = this.temp;

          if(this.seatTogglers[this.seatIndex]){

            this.element?.setAttribute("class",'seat selected');

            this.seatNoPush = event.target.attributes.name.nodeValue;

            this.seatIdPush = this.seats[this.seatIndex].seatId

            this.selectedSeatsId.push(this.seatIdPush);

            this.selectedSeats.push(this.seatNoPush);

            this.seatCount -= 1;

          }

          else{

            this.element?.setAttribute("class",'seat');

            this.seatNoPush = event.target.attributes.name.nodeValue;

            this.seatIdPush = this.seats[this.seatIndex].seatId

            this.selectedSeatsId.splice(this.selectedSeatsId.indexOf(this.seatIdPush),1)

            this.selectedSeats.splice(this.selectedSeats.indexOf(this.seatNoPush),1)

            this.seatCount += 1;

          }

        }

      }

      console.log(this.selectedSeats)

      console.log(this.selectedSeatsId)

    }

    Submit(){

      if(!this.skip){

        if(this.isReturn){

          this.OWselectedSeats = this.selectedSeats;

          this.OWselectedSeatsId = this.selectedSeatsId;

          this.OWclassBool =this.classBool;

          this.availableBool = true;

          this.skip = true;

          this.tripName = "Round Trip";

          this.classBool = Boolean(JSON.parse(localStorage.getItem('RTflightType')!));

          this.RTflightNumber = JSON.parse(localStorage.getItem('roundtripDetails')!)['flightNumber'];

          this.seatCount = Number(JSON.parse(localStorage.getItem('passengerList')!).length);

          this.seatTogglers.forEach((value,index) =>{

            if(value == true)

            value = false;

            this.temp = document.getElementById(index.toString());

            this.element = this.temp;

            this.element?.setAttribute("class",'seat');

          })

          this.selectedSeatsId = [];

          this.selectedSeats = [];

          this.getSeats(this.RTflightNumber!);

        }

      }

      else{

        this.updateSelectedSeats(this.OWselectedSeatsId);

        this.updateSelectedSeats(this.selectedSeatsId);

        this.updateAvailableSeatsInDb(this.RTflightNumber!);

        this.updateAvailableSeatsInDb(this.OWflightNumber);

        localStorage.setItem('OWseatsSelected',JSON.stringify(this.OWselectedSeats));

        localStorage.setItem('OWseatsSelectedId',JSON.stringify(this.OWselectedSeatsId));

        localStorage.setItem('OWclassBool',JSON.stringify(this.OWclassBool));

        localStorage.setItem('RTseatsSelected',JSON.stringify(this.selectedSeats));

        localStorage.setItem('RTseatsSelectedId',JSON.stringify(this.selectedSeatsId));

        localStorage.setItem('RTclassBool',JSON.stringify(this.classBool));

        this.navigate();

      }

      if(!this.availableBool){

        this.updateSelectedSeats(this.selectedSeatsId);

        this.updateAvailableSeatsInDb(this.OWflightNumber);

        localStorage.setItem('OWseatsSelected',JSON.stringify(this.selectedSeats));

        localStorage.setItem('OWseatsSelectedId',JSON.stringify(this.selectedSeatsId));

        localStorage.setItem('OWclassBool',JSON.stringify(this.classBool));

        this.navigate();

      }

    }

    navigate(){

      if(this.route.parent?.routeConfig?.path?.localeCompare("home") == 0){

        this.router.navigate(["/loginPage","1"])

      }

      else {

        this.router.navigate(["/payment"]);

      }

    }

    ngOnInit(): void {

      if(this.route.parent?.routeConfig?.path == "userDashboard"){

        if(sessionStorage.getItem('LoginCheckCode') != "User"){

          this.router.navigate(['/loginPage/flightSearch'])

        }

      }

      this.seatCount = Number(JSON.parse(localStorage.getItem('passengerList')!).length)

      this.classBool = Boolean(JSON.parse(localStorage.getItem('OWflightType')!))

      this.freezeCheck = this.seatCount;

      this.OWflightNumber = JSON.parse(localStorage.getItem('onewayDetails')!)['flightNumber']

      this.isReturn = JSON.parse(localStorage.getItem('flagTripType')!);

      this.tripName = "One Way"

      this.getSeats(this.OWflightNumber);

    }

    // API METHODS

    getSeats(flightNumber : string){

      this.service.getAllSeats(flightNumber).subscribe( data =>

        {

          this.seats = data as Seats[];

          console.log(data);

        },

        err => {

        console.log(err);

        });

    }

    updateSelectedSeats(selectedSeatsId : number[]){

      this.service.updateSeats(selectedSeatsId).subscribe(data =>

        {

          console.log(data);

        },(err) =>{

          console.log(err)

        }

      );

    }

    updateAvailableSeatsInDb(flightNumber : string){

      this.service.availbleSeatsUpdate(flightNumber,this.classBool,this.freezeCheck).subscribe(data =>

        {

          console.log(data);

        },(err) =>{

          console.log(err)

        }

      );

    }

  }

User seat Service

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

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

import { Users } from "../models/users";

@Injectable({providedIn:"root"})

export class UserServices{

    constructor(private http:HttpClient){}

    readonly url="http://localhost:13972/api/User";

    getUser()

    {

        return this.http.get(this.url);

    }

    insertUser(User1:Users)

    {

        return this.http.post(this.url,User1);

    }

    sendRegistrationNumber(email : string){

        return this.http.get(`${this.url}/sendRegId?email=${email}`,{responseType : "text"})

    }

    getBookingsForAUser(email : string){

        return this.http.get(`${this.url}/getBookings?email=${email}`)

    }

    userVerification(useremail : string, userpassword: string){

        return this.http.get(`${this.url}/login?email=${useremail}&password=${userpassword}`);

      }

    sendOTP(regId : number,email : string){

        return this.http.get(`${this.url}/forgotpassword?regId=${regId}&email=${email}`);

    }

    updatePassword(newpassword : string,email : string){

        return this.http.put(`${this.url}/updatepassword?newpassword=${newpassword}&email=${email}`,null,{responseType : "text"});

    }

    getUserID(email : string){

        return this.http.get(`${this.url}/getUserId?email=${email}`);

    }

}