**1.LoginController :-**

package com.abctelecom.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.abctelecom.dto.UserDto;

import com.abctelecom.entity.User;

import com.abctelecom.service.UserService;

@RestController

@CrossOrigin

@RequestMapping("/login")

public class LoginController {

    @Autowired

    private UserService userService;

    @PostMapping("/check")

    public User loginHandler(@RequestBody UserDto *userDto*) {

        System.out.println(userDto.getEmail());

        System.out.println(userService.loginCheck(userDto).getEmail());

        return userService.loginCheck(userDto);

    }

}

**2. TicketController :-**

package com.abctelecom.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.abctelecom.entity.Feedback;

import com.abctelecom.entity.Notes;

import com.abctelecom.entity.Ticket;

import com.abctelecom.service.TicketService;

@RestController

@CrossOrigin

@RequestMapping("/tickets")

public class TicketController {

    private TicketService ticketService;

    @Autowired

    public TicketController(TicketService *ticketService*) {

        this.ticketService = ticketService;

    }

    @PostMapping("/new")

    public String raiseNew(@RequestBody Ticket *ticket*) {

        return ticketService.save(ticket);

    }

    @GetMapping("/")

    public List<Ticket> getAllTickets(){

        return ticketService.getAllTickets();

    }

    @GetMapping("/customer/{id}")

    public List<Ticket> getTicketsByCustomerId(@PathVariable String *id*){

        return ticketService.getTicketsByCustomerId(Long.parseLong(id));

    }

    @GetMapping("/{id}")

    public Ticket getTicketById(@PathVariable String *id*) {

        return ticketService.getTicketById(Long.parseLong(id));

    }

    @GetMapping("/notes/{ticketId}")

    public List<Notes> getNotesByTicketId(@PathVariable String *ticketId*) {

        return ticketService.getNotesByTicketId(Long.parseLong(ticketId));

    }

    @PostMapping("/notes/save")

    public String saveNotes(@RequestBody Notes *notes*) {

        return ticketService.saveNotes(notes);

    }

    @GetMapping("/feedback/{ticketId}")

    public Feedback getFeedbackByTicket(@PathVariable String *ticketId* ) {

        return ticketService.findFeedbackByTicket(Long.parseLong(ticketId));

    }

    @PostMapping("/feedback/submit")

    public String submitFeedback(@RequestBody Feedback *feedback*) {

        return ticketService.saveFeedback(feedback);

    }

}

**3. UserController**

package com.abctelecom.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.abctelecom.dto.CustomerDto;

import com.abctelecom.entity.Customer;

import com.abctelecom.entity.Engineer;

import com.abctelecom.entity.Manager;

import com.abctelecom.service.UserService;

@RestController

@CrossOrigin

@RequestMapping("/users")

public class UserController {

    @Autowired

    private UserService userService;

    @GetMapping("/")

    public ResponseEntity<String> apiTest(){

        return new ResponseEntity<String>("Hi user", HttpStatus.OK);

    }

    @GetMapping("/zipcodes")

    public List<String> getZipcodes(){

        System.out.println(userService.getZipcodes());

        return userService.getZipcodes();

    }

    @PostMapping("/customers/new")

    public String createCustomer(@RequestBody CustomerDto *customerDto*) {

        Customer customer = new Customer(customerDto);

        return userService.addCustomer(customer);

    }

    @PostMapping("/engineers/new")

    public String createEngineer(@RequestBody Engineer *engineer*) {

        return userService.addEngineer(engineer);

    }

    @PostMapping("/managers/new")

    public String createManager(@RequestBody Manager *manager*) {

        return userService.addManager(manager);

    }

    @GetMapping("/customers")

    public List<Customer> getAllCustomers(){

        return userService.getAllCustomers();

    }

    @GetMapping("customers/{id}")

    public Customer getCustomerById(@PathVariable String *id*) {

        return userService.getCustomerById(Long.parseLong(id));

    }

    @GetMapping("customers/findByUserId/{id}")

    public Customer findCustomerByUserId(@PathVariable String *id*) {

        return userService.findCustomerByUserId(Long.parseLong(id));

    }

    @GetMapping("engineers/findByUserId/{id}")

    public Engineer findEngineerByUserId(@PathVariable String *id*) {

        return userService.findEngineerByUserId(Long.parseLong(id));

    }

    @GetMapping("managers/findByUserId/{id}")

    public Manager findBYUserId(@PathVariable String *id*) {

        return userService.findManagerByUserId(Long.parseLong(id));

    }

    @GetMapping("engineers/{id}")

    public Engineer getEngineerById(@PathVariable String *id*) {

        return userService.getEngineerById(Long.parseLong(id));

    }

    @GetMapping("managers/{id}")

    public Manager getManagerById(@PathVariable String *id*) {

        return userService.getManagerById(Long.parseLong(id));

    }

    @GetMapping("/engineers")

    public List<Engineer> getAllEngineers(){

        return userService.getAllEngineers();

    }

    @GetMapping("engineers/byZipcode/{zipcode}")

    public List<Engineer> getEngineersByZipcode(@PathVariable String *zipcode*){

        return userService.getEngineersByZipcode(zipcode);

    }

    @GetMapping("/managers")

    public List<Manager> getAllManagers(){

        return userService.getAllManagers();

    }

    @DeleteMapping("/customers/{id}")

    public String deleteCustomer(@PathVariable String *id*) {

        return userService.deleteCustomer(Long.parseLong(id));

    }

}

**LoginComponent:-**

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

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

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

import { UserDto } from 'src/app/common/user-dto';

import { LoginService } from 'src/app/services/login.service';

import { UserService } from 'src/app/services/user.service';

@Component({

  selector: 'app-login',

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

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

})

export class LoginComponent implements OnInit {

  loginForm:FormGroup;

  constructor(

    private *formBulder*:FormBuilder,

    private *loginService*:LoginService,

     private *userService*:UserService,

     private *router*:Router

  ) { }

  ngOnInit(): void {

    this.loginForm=this.formBulder.group({

      email:new FormControl('',[Validators.required,Validators.email]),

        password:new FormControl('',Validators.required)

    });

  }

  loginhandler(){

    let userDto: UserDto=this.loginForm.value;

    this.loginService.checkLogin(userDto).subscribe({

      next:*response*=>{

        if(response!=null){

          alert( `Logged in successfully! User Name: ${response.fullName}`);

          if(response.role=='customer'){

            this.userService.getCustomerByUserId(response.id).subscribe(*data*=>{

              this.userService.loggedInCustomer=data;

              this.router.navigateByUrl("ticket-list");

            })

          }

         else if(response.role=='engineer'){

            this.userService.getEngineerByUserId(response.id).subscribe(*data*=>{

              this.userService.loggedInEngineer=data;

              this.router.navigateByUrl("ticket-list");

            })

          }

          else if(response.role=='manager'){

            this.userService.getManagerByUserId(response.id).subscribe(*data*=>{

              this.userService.loggedInManager=data;

              this.router.navigateByUrl("ticket-list");

            })

          }

          else if(response.role=='admin'){

            this.userService.loggedInCustomer;

            this.loginService.loggedInUser.subscribe(*data*=>{

              console.log(data);

              this.router.navigateByUrl("home");

            })

          }

          else{

              alert("something went wrong");

          }

        }

      },

      error:*err*=>{

        // alert(`There was an error: ${err.message}`);

        alert("incorrect credentials")

      }

    })

  }

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**TicketComponent:-**

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

import { tick } from '@angular/core/testing';

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

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

import { timestamp } from 'rxjs/operators';

import { Customer } from 'src/app/common/customer';

import { Engineer } from 'src/app/common/engineer';

import { Feedback } from 'src/app/common/feedback';

import { Manager } from 'src/app/common/manager';

import { Notes } from 'src/app/common/notes';

import { Ticket } from 'src/app/common/ticket';

import { User } from 'src/app/common/user';

import { LoginService } from 'src/app/services/login.service';

import { TicketService } from 'src/app/services/ticket.service';

@Component({

  selector: 'app-ticket',

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

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

})

export class TicketComponent implements OnInit {

  complaintForm: FormGroup;

  loggedInUser:User=null;

  loggedInCustomer:Customer=new Customer();

  loggedInEngineer:Engineer;

  loggedInmanagre:Manager;

  serviceType:string='';

  assignedTo:string='';

  ticket:Ticket=null;

  allEngineers:Engineer[];

  engineers:Engineer[];

  notes:Notes[]=null;

  lastkeydown1: number = 0;

  subscription: any;

  feedback:Feedback=null;

  constructor(

    private *formBuilder*: FormBuilder,

    private *ticketService*:TicketService,

    private *loginService*:LoginService,

    private *route*:ActivatedRoute

  ) { }

// End of constructor

  ngOnInit(): void {

    this.complaintForm = this.formBuilder.group({

      ticket:this.formBuilder.group({

        id:[''],

        mobile:[''],

        serviceType:[''],

        issueType:[''],

        description:[''],

        zipcode:[''],

        address:[''],

        createdOn:[''],

        updatedOn:[''],

        status:[''],

        // notes:[''],

        customer:this.formBuilder.group({

          id:[''],

          serviceType:new FormControl("",[Validators.required]),

          address:new FormControl("",[Validators.required]),

          zipcode:new FormControl("",[Validators.required]),

          user:this.formBuilder.group({

              id:[''],

              fullName: new FormControl("",[Validators.required]),

              role:new FormControl("",[Validators.required]),

              email:new FormControl("",[Validators.required,Validators.email]),

              phone:new FormControl("",[Validators.required]),

              password:new FormControl("",[Validators.required])

          })

        }),

        // engineer:['']

        engineer:this.formBuilder.group({

            id:[''],

            zipcode:new FormControl("",[Validators.required]),

            user:this.formBuilder.group({

                id:[''],

                fullName: new FormControl("",[Validators.required]),

                role:new FormControl("",[Validators.required]),

                email:new FormControl("",[Validators.required,Validators.email]),

                phone:new FormControl("",[Validators.required]),

                password:new FormControl("",[Validators.required])

            })

        }),

      }),

      notes:this.formBuilder.group({

        id:[''],

        ticket:[''],

        comments:[''],

        createdOn:[''],

        updatedBy:[''],

        isCustomer:['']

      })

    });

    this.loginService.loggedInUser.subscribe(*data*=>{

      this.loggedInUser=data;

      if(this.route.snapshot.paramMap.has('id')){

        this.route.paramMap.subscribe(()=>{

          this.getTicketDetails();

        })

      }

      else if(this.loggedInUser.role="customer"){

        console.log("line 103");

        this.getTicketDetailsByCustomer();

      }

    });

  }

  // End ngOnInit

  getTicketDetails() {

    //  "+" prefixed here will cast string to a number

    this.ticketService.getTicketById(+this.route.snapshot.paramMap.get('id')).subscribe(*data*=>{

      this.ticket=data;

      this.ticketService.getFeedback(data.id).subscribe(*data*=>{

        this.feedback=data;

      })

     this.ticketService.getNotesByTicketId(data.id).subscribe(*data*=>{

      this.notes=data;

     });

      console.log(this.ticket.serviceType+" at line 121");

      this.serviceType=this.ticket.serviceType;

      if(this.ticket.engineer!=null){

        this.assignedTo=this.ticket.engineer.user.email;

        this.complaintForm.get('ticket').get('engineer').patchValue(this.ticket.engineer);

        console.log("At line 136===>"+this.assignedTo);

      }

      this.complaintForm.get('ticket').patchValue({

        id:this.ticket.id,

        mobile:this.ticket.mobile,

        serviceType:this.ticket.serviceType,

        issueType:this.ticket.issueType,

        description:this.ticket.description,

        zipcode:this.ticket.zipcode,

        address:this.ticket.address,

        createdOn:this.ticket.createdOn,

        updatedOn:this.ticket.updatedOn,

        status:this.ticket.status,

        customer:this.ticket.customer

      });

      this.complaintForm.get('ticket').disable();

      if(this.loggedInUser.role=='engineer'){

        this.loginService.loggedInEngineer.subscribe(*data*=>{

          this.loggedInEngineer=data;

          if(this.ticket.engineer.id==data.id){

            this.complaintForm.get('ticket').get('status').enable({onlySelf:true});

          }

        });

      }

      else if(this.loggedInUser.role=='manager'){

        console.log("line 153");

        this.loginService.loggedInManager.subscribe(*data*=>{

          this.loggedInmanagre=data;

          console.log("line 157");

          this.complaintForm.get('ticket').get('engineer').enable({onlySelf:true});

          console.log("Enabled Engineer");

        });

        // populate Engineers for auto complete textbox

        this.ticketService.getEngineersByZipcode(this.ticket.zipcode).subscribe(*data*=>{

          this.allEngineers=data;

          this.engineers=data;

        })

      }

      else if(this.loggedInUser.role=='customer'){

        this.loginService.loggedInCustomer.subscribe(*data*=>this.loggedInCustomer=data);

      }

      console.log(this.ticket.serviceType+" at line 125");

      // this.complaintForm.get('ticket').get('serviceType').setValue(ticket.serviceType);

    })

  }

  getTicketDetailsByCustomer() {

    console.log("line 110");

    this.loginService.loggedInUser.subscribe(*data*=>{

      console.log("line 113");

      this.loggedInUser=data;

        let ticketFrom=this.complaintForm.get('ticket');

        this.loginService.loggedInCustomer.subscribe(*data*=>{

          this.loggedInCustomer=data;

          // assign all the customer details to ticket in order to auto fill the form

          console.log("line 119");

          ticketFrom.get('customer').get('user').get("fullName").setValue(data.user.fullName);

          ticketFrom.get('customer').get('user').get("fullName").disable();

          ticketFrom.get('mobile').setValue(data.user.phone);

          ticketFrom.get('address').setValue(data.address);

          ticketFrom.get('zipcode').setValue(data.zipcode);

        })

    })

  }

  // Auto complete list function

  getAutoComplList(*$event*) {

    let engineerEmail = this.complaintForm.get('ticket').get('engineer').get('user').get('email').value;

    // this.engineers = [];

    // if (engineerEmail.length > 2) {

    //   if ($event.timeStamp+200> 200) {

    //     this.engineers = this.searchFromArray(this.allEngineers, engineerEmail);

    //   }

    // }

    console.log(engineerEmail+" Selected Engineers");

  }

  searchFromArray(*arr*:Engineer[], *input*:string) {

    let matches:Engineer[] = [], i:number;

    for (i = 0; i < arr.length; i++) {

      if (arr[i].user.email.match(input)) {

        matches.push(arr[i]);

      }

    }

    return matches;

  };

  submitHandle(){

    if(this.ticket!=null){

      console.log(this.complaintForm.get('ticket').get('engineer').get('id').value);

      if(this.loggedInUser.role=='manager'){

        let engineer:Engineer=new Engineer();

        engineer.id=this.complaintForm.get('ticket').get('engineer').get('id').value;

        this.ticket.engineer=engineer;

        this.updateTicket(this.ticket);

      }

      else if(this.loggedInUser.role=='engineer'){

        this.ticket.status=this.complaintForm.get('ticket').get('status').value;

        this.updateTicket(this.ticket);

      }

      let comments:string='';

      comments =this.complaintForm.get('notes').get('comments').value;

      console.log(comments+"====>Comments");

      if((comments.trim().length>0)){

        let notes:Notes = new Notes();

        notes.comments=comments;

        notes.isCustomer=(this.loggedInUser.role=='customer');

        notes.updatedBy=this.loggedInUser.email;

        notes.ticket=this.ticket;

        this.ticketService.saveNotes(notes).subscribe({

          next:*response*=>{

            alert(`Ticket# ${JSON.stringify(response)} updated successfully`)

          },

          error:*err*=>{

            alert(`There was an error: ${err.message}`);

            console.log(err.message);

          }

        })

      }

    }

    else{

      let ticket:Ticket = this.complaintForm.controls['ticket'].value;

      ticket.customer=this.loggedInCustomer;

      ticket.engineer=null;

      console.log(ticket);

      this.ticketService.raiseNewTicket(ticket).subscribe({

        next:*response*=>{

          alert( `Your ticket has been submitted! Ticket number: ${JSON.stringify(response)}`);

        },

        error:*err*=>{

          alert(`There was an error: ${err.message}`);

          console.log(err.message);

        }

      });

    }

  }

  updateTicket(*ticket*: Ticket) {

    this.ticketService.raiseNewTicket(ticket).subscribe({

      next:*response*=>{

        alert(`Ticket# ${JSON.stringify(response)} updated successfully`)

      },

      error:*err*=>{

        alert(`There was an error: ${err.message}`);

        console.log(err.message);

      }

    })

  }

}