ПРИЛОЖЕНИЕ С

(обязательное)

Листинг программы

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

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { Global } from './\_models/global';

import {BrowserModule} from '@angular/platform-browser';

import { PageNotFoundComponent } from './views/page-not-found/page-not-found.component';

import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

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

import {FormsModule, ReactiveFormsModule} from '@angular/forms';

import {HTTP\_INTERCEPTORS, HttpClient, HttpClientModule} from '@angular/common/http';

import {AuthGuard} from './\_helpers/auth.guard';

import {ErrorInterceptor} from './\_helpers/error.interceptor';

import {JwtInterceptor} from './\_helpers/jwt.interceptor';

import {RoleGuardService} from './\_services/licence-guars.service';

import {SvgModule} from '../svg/svg.module';

import {SignUpComponent} from './views/sign-up/sign-up.component';

import { ModalContentComponent } from './components/modal-content/modal-content.component';

import { ConformEmailComponent } from './views/conform-email/conform-email.component';

import {ToastModule} from './components/toast/toast.module';

import {ErrorService} from './\_services/error.service';

import {InfoService} from './\_services/info.service';

import {SuccessService} from './\_services/success.service';

import { WebinarGuestAuthComponent } from './views/webinar-guest-auth/webinar-guest-auth.component';

import {UserService} from './\_services/user.service';

import {WebinarService} from './\_services/webinar.service';

@NgModule({

declarations: [

AppComponent,

PageNotFoundComponent,

LoginComponent,

SignUpComponent,

ModalContentComponent,

ConformEmailComponent,

WebinarGuestAuthComponent,

],

imports: [

BrowserModule,

AppRoutingModule,

BrowserAnimationsModule,

ReactiveFormsModule,

HttpClientModule,

SvgModule,

FormsModule,

ToastModule

],

providers: [

{provide: HTTP\_INTERCEPTORS, useClass: ErrorInterceptor, multi: true},

{provide: HTTP\_INTERCEPTORS, useClass: JwtInterceptor, multi: true},

JwtInterceptor,

RoleGuardService,

Global,

HttpClient,

AuthGuard,

ErrorService,

InfoService,

SuccessService,

UserService,

WebinarService

],

bootstrap: [AppComponent]

})

export class AppModule { }

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

@Component({

selector: 'app-root',

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

styleUrls: ['./app.component.scss']

})

export class AppComponent {

title = 'Welcome';

}

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

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

import {PageNotFoundComponent} from './views/page-not-found/page-not-found.component';

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

import {

RoleGuardService as RoleGuard

} from './\_services/licence-guars.service';

import {SignUpComponent} from './views/sign-up/sign-up.component';

import {ConformEmailComponent} from './views/conform-email/conform-email.component';

import {WebinarGuestAuthComponent} from './views/webinar-guest-auth/webinar-guest-auth.component';

const routes: Routes = [

{path: '', pathMatch: 'full', redirectTo: 'home/0'},

{

path: 'login',

component: LoginComponent,

},

{

path: 'signup',

component: SignUpComponent,

},

{

path: 'email-ver/:email/:returnUrl',

component: ConformEmailComponent

},

{

path: 'admin',

canActivate: [RoleGuard],

data: {

expectedLicence: 5

},

loadChildren: () => import('./admin/admin.module').then(m => m.AdminModule),

},

{

path: 'home/:id',

loadChildren: () => import('./views/home-page/home-page.module').then(m => m.HomePageModule)

},

{

path: 'webinar/:id',

canActivate: [RoleGuard],

data: {

expectedLicence: 0

},

loadChildren: () => import('./views/webinar-room/webinar-room.module').then(m => m.WebinarRoomModule)

},

{

path: 'webinar/:id/:email/:username',

component: WebinarGuestAuthComponent,

},

{ path: '404', component: PageNotFoundComponent },

{ path: '\*\*', component: PageNotFoundComponent }

];

@NgModule({

imports: [RouterModule.forRoot(routes, { relativeLinkResolution: 'legacy', paramsInheritanceStrategy: 'always'})],

exports: [RouterModule]

})

export class AppRoutingModule {

}

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

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

import {WebinarRoomComponent} from './webinar-room.component';

import {WebinarRoomRoutingModule} from './webinar-room.routing.module';

import {MediaPlayerModule} from '../../components/media-player/media-player.module';

import {CommentsModule} from '../../components/comments/comments.module';

import {WebinarService} from '../../\_services/webinar.service';

@NgModule({

imports: [

CommonModule,

MediaPlayerModule,

CommentsModule,

WebinarRoomRoutingModule

],

declarations: [

WebinarRoomComponent

],

providers: [

WebinarService

]

})

export class WebinarRoomModule{

}

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

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

import {WebinarService} from '../../\_services/webinar.service';

@Component({

selector: 'app-webinar-room',

templateUrl: './webinar-room.component.html',

styleUrls: ['./webinar-room.component.scss']

})

export class WebinarRoomComponent implements OnInit {

roomNom: any;

webinar: any;

constructor(

private route: ActivatedRoute,

private router: Router,

private webinarService: WebinarService

) {

this.route.params.subscribe(params => {

this.roomNom = params.id;

this.webinarService.getWebinarById(this.roomNom).subscribe(body => {

if (body.success) {

this.webinar = body.data;

} else {

router.navigate(['404']);

}

});

});

}

ngOnInit(): void {

}

}

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

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

import {WebinarService} from '../../\_services/webinar.service';

import {UserService} from '../../\_services/user.service';

import {ErrorService} from '../../\_services/error.service';

import {InfoService} from '../../\_services/info.service';

@Component({

selector: 'app-webinar-guest-auth',

templateUrl: './webinar-guest-auth.component.html',

styleUrls: ['./webinar-guest-auth.component.scss']

})

export class WebinarGuestAuthComponent implements OnInit {

id: number = 0;

constructor(

private userService: UserService,

private route: ActivatedRoute,

private router: Router,

private error: ErrorService,

private info: InfoService,

private webinarService: WebinarService

) {

this.route.params.subscribe(params => {

const roomNom = params.id;

this.id = roomNom;

this.webinarService.getWebinarById(roomNom).subscribe(body => {

if (body.success) {

userService.guestCheck({id: params.id, email: decodeURIComponent(params.email)}).subscribe((response) => {

localStorage.removeItem('guest');

if (response.success) {

localStorage.setItem('guest', 'true');

localStorage.setItem('email', response.data.email);

localStorage.setItem('userName', response.data.userName);

}

},

e => {this.error.errorChange(e); },

() => {

if (localStorage.getItem('guest') === 'true') {

this.info.infoChange('Вы вошли под именем ' + localStorage.getItem('userName'));

router.navigate(['webinar', params.id]);

} else {

this.error.errorChange('К сожалению эта ссылка не действительна');

router.navigate(['404']);

}

}

);

} else {

router.navigate(['404']);

}

});

});

}

ngOnInit(): void {

}

}

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

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

import {RegUser} from '../../\_models/user';

import {AuthService} from '../../\_services/auth.service';

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

@Component({

selector: 'app-sign-up',

templateUrl: './sign-up.component.html',

styleUrls: ['./sign-up.component.scss']

})

export class SignUpComponent implements OnInit {

submitted = false;

public reguser: RegUser = {

emailreg: '',

passwordreg: '',

};

public loginForm: FormGroup;

private returnUrl: string;

public loading;

constructor(private authService: AuthService,

private fb: FormBuilder,

private router: Router,

private route: ActivatedRoute

) {

}

ngOnInit(): void {

if (this.authService.isLoggedIn()) {

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

}

// creating logIn form

this.loginForm = this.fb.group({

// input login validator

email: ['', [Validators.required, Validators.pattern('^\\w+([\\.-]?\\w+)\*@\\w+([\\.-]?\\w+)\*(\\.\\w{2,3})+$')]],

// input password validator

password: ['', [Validators.required]],

// input name

name: ['', [Validators.required, Validators.pattern('^[a-zA-ZА-Яа-я\\s]+$')]]

});

this.returnUrl = this.route.snapshot.queryParams[`returnUrl`] || '/';

}

get f(): any {

return this.loginForm.controls;

}

isInvalid( attr: string): any {

return (this.loginForm.get(attr).touched

|| this.loginForm.get(attr).dirty

|| this.submitted)

&& this.loginForm.get(attr).errors;

}

getErrors(attr: string): any {

return this.loginForm.get(attr).errors;

}

// sending logIn form

onSubmit(): void {

this.submitted = true;

if (this.loginForm.invalid) {

return;

}

const formData = this.loginForm.value;

this.loading = true;

this.authService.registerUser(formData).subscribe(

result => {

if (result.success) {

this.router.navigate(['email-ver', formData.email, this.returnUrl]);

} else {

alert(result.message);

}

this.loading = false;

});

}

}

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

@Component({

selector: 'app-page-not-found',

templateUrl: './page-not-found.component.html',

styleUrls: ['./page-not-found.component.scss']

})

export class PageNotFoundComponent implements OnInit {

constructor() { }

ngOnInit(): void {

}

}

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

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

import {RegUser, User} from '../../\_models/user';

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

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

import {AuthService} from '../../\_services/auth.service';

import {ModalContentComponent} from '../../components/modal-content/modal-content.component';

import {NgbModal} from '@ng-bootstrap/ng-bootstrap';

@Component({

selector: 'app-login',

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

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

})

export class LoginComponent implements OnInit {

submitted = false;

public reguser: RegUser = {

emailreg: '',

passwordreg: '',

};

public loginForm: FormGroup;

private returnUrl: string;

public loading;

constructor(private authService: AuthService,

private fb: FormBuilder,

private router: Router,

private route: ActivatedRoute,

private modalService: NgbModal,

) {

}

ngOnInit(): void {

if (this.authService.isLoggedIn()) {

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

}

// creating logIn form

this.loginForm = this.fb.group({

// input login validator

email: ['', [Validators.required, Validators.pattern('^\\w+([\\.-]?\\w+)\*@\\w+([\\.-]?\\w+)\*(\\.\\w{2,3})+$')]],

// input password validator

password: ['', [Validators.required]],

// input remember me checkbox

RME: ['0']

});

this.returnUrl = this.route.snapshot.queryParams[`returnUrl`] || '/';

}

get f(): any {

return this.loginForm.controls;

}

isInvalid( attr: string): any {

return (this.loginForm.get(attr).touched

|| this.loginForm.get(attr).dirty

|| this.submitted)

&& this.loginForm.get(attr).errors;

}

getErrors(attr: string): any {

return this.loginForm.get(attr).errors;

}

// sending logIn form

onSubmit(): void {

this.submitted = true;

if (this.loginForm.invalid) {

return;

}

const formData = this.loginForm.value;

this.loading = true;

this.authService.loginUser(formData).subscribe( (result) => {

if (result.success) {

this.router.navigate([this.returnUrl]);

} else {

const modalRef = this.modalService.open(ModalContentComponent);

modalRef.componentInstance.name = 'Неверный логин или пароль';

}

this.loading = false;

});

}

}

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

import {HomePageComponent} from './home-page.component';

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

import {WEBINAR\_PROVIDERS} from './home-page.providers';

import {WebinarService} from '../../\_services/webinar.service';

import {SearchService} from '../../\_services/search.service';

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

import {HomePageRoutingModule} from './home-page.routing.module';

import {SliderModule} from '../../components/slider/slider-module';

import {NavbarModule} from '../../components/navbar/navbar.module';

import {FormModule} from '../../components/form/form.module';

import {SpikerModule} from '../../components/spiker/spiker.module';

import {ProfitModule} from '../../components/profit/profit.module';

import {ProgrammModule} from '../../components/programm/programm.module';

import {DropdownItemModule} from '../../components/dropdown-item/dropdown-item.module';

@NgModule({

imports: [

CommonModule,

SliderModule,

NavbarModule,

SpikerModule,

FormModule,

ProfitModule,

ProgrammModule,

HttpClientModule,

HomePageRoutingModule,

DropdownItemModule,

],

declarations: [

HomePageComponent,

],

providers: [

WEBINAR\_PROVIDERS,

SearchService,

WebinarService,

]

})

export class HomePageModule{

}

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

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

import {AuthService} from '../../\_services/auth.service';

import {SearchService} from '../../\_services/search.service';

import {SuccessService} from '../../\_services/success.service';

import {ErrorService} from '../../\_services/error.service';

import {WebinarService} from '../../\_services/webinar.service';

import {Common} from '../../\_helpers/common.helper';

@Component({

selector: 'app-home-page',

templateUrl: './home-page.component.html',

styleUrls: ['./home-page.component.scss' ],

})

export class HomePageComponent implements OnInit, AfterViewInit {

public webinar: any;

id = 0;

constructor(

private element: ElementRef,

private route: ActivatedRoute,

private authService: AuthService,

private searchService: SearchService,

private success: SuccessService,

private error: ErrorService,

private router: Router,

private webinarService: WebinarService

) {

this.route.params.subscribe(params => {

this.id = +params.id;

if (this.id !== 0) {

this.webinarService.getWebinarById(this.id).subscribe(body => {

if (body.success) {

this.webinar = body.data;

} else {

router.navigate(['404']);

}

},

e => {

error.errorChange('К сожалению, этого вебинара не существует.\n Перенапраыляем на главную...');

router.navigate(['home', '0']);

});

} else {

this.webinar = {

id: 0,

};

}

});

}

program = [

{title: 'Урок 1', text: 'Что такое MICE?'},

{title: 'Урок 2', text: 'Деловая переписка. Правила Разговора по телефону.'},

{title: 'Урок 3', text: 'Расстановка в зале - SET UP'},

{title: 'Урок 4', text: 'Этапы организации мероприятия.'},

{title: 'Урок 5', text: 'Шаблоны заявок.'},

{title: 'Урок 6', text: 'Смета - Самый важный документ!'},

{title: 'Урок 7', text: 'Программа мероприятия.'},

{title: 'Урок 8', text: 'Как выбрать место Проведения для мероприятия.'},

{title: 'Урок 9', text: 'Ценообразование.'},

{title: 'Урок 10', text: 'Отношения, поставщики, команда. Настройки и ответственность.'},

{title: 'Урок 11', text: 'Авиабилеты.'},

{title: 'Урок 12', text: 'Логистика.'},

{title: 'Урок 13', text: 'Оборудование на мероприятии.'},

{title: 'Урок 14', text: 'Кэйтеринг.'},

{title: 'Урок 15', text: 'Зарубежные мероприятия, поездки. Партнеры за рубежом.'},

{title: 'Урок 16', text: 'Разылекательные мероприятия.'},

{title: 'Урок 17', text: 'Тимбилдинги.'},

{title: 'Урок 18', text: 'Разбор мероприятий.'},

{title: 'Урок 19', text: 'Онлайн мероприятия.'},

];

spikier = 'assets/spikier\_1.jpg';

spikierText = 'Почему я создала школу event-менеджера?<br>- Организации мероприятий не учат в учебных заведениях, хотя - это современная, востребованная профессия. <br>- Сейчас, все больше людей и компаний доверяют профессионалам организацию своего мероприятия. <br>- Я хочу создать больше профессионалов в этом секторе профессий! ';

navBarElements = [];

slides = [

{ background: 'slider-back-1.jpg',

title: 'Хочешь научиться профессионально организовывать мероприятия? ',

subtitle: 'Скорее записывайся на мой курс по подготовке организатора мероприятий в event школу.'

},

{ background: 'slider-back-2.jpg',

title: 'Конференции, семинары, банкеты, свадьбы, дни рождения...',

subtitle: 'orem ipsum dolor sit amet, consectetur adipisicing'

},

{ background: 'slider-back-3.jpg',

title: 'Организации мероприятий не учат в учебных заведениях, хотя - это современная, востребованная профессия. ',

subtitle: 'Сейчас, все больше людей и компаний доверяют профессионалам организацию своего мероприятия. '

}

];

pros = [

{

header: 'Что будет на курсе?',

body: 'На этом курсе будут затронуты все аспекты организации мероприятия: <br>- вы пройдете все этапы организации, начиная от получения запроса, подбора места проведения и заканчивая контролем мероприятия, <br>- вы научитесь грамотно составлять сметы, <br>- научитесь понимать весь процесс от начала и до конца; <br>- узнаете что такое логистика и как правильно все организовать,<br>- научитесь составлять программы, <br>- научитесь правильно вести переговоры как с клиентами, так и с поставщиками, <br>- узнаете о том что такое онлайн мероприятия, как их можно организовать, <br> и еще много полезного. '

},

{

header: 'Какие бонусы выполучите?',

body: 'Чек-листы, образцы смет, образцы программ и другие важные документы, необходимые организатору мероприятий.'

},

{

header: 'Для кого подойдет этот курс?',

body: '- для тех, кто хочет открыть для себя мир event индустрии.<br>- для тех, кто просто хочет научиться ремеслу организатора мероприятий.<br>- для всех, кто хочет знать как это все происходит изнутри.<br>- для тех, кто хочет сам контролировать процесс организации своих мероприятий.<br>- для тех, кто любит общаться, посещать разные места.<br>- для тех, кто не знает куда деть свою врожденную ответственность, креативность, энергию; <br>- для тех, кому нравятся нестандартные решения и для тех, кто не любит сидеть на месте.<br>- для тех, кто хочет расширить свой круг знакомств и свой кругозор.<br>- для всех, кто хочет сменить сферу деятельности, обрести новую профессию и зарабатывать на этом деньги.'

},

{

header: 'Что вы будете уметь после окончания курса?',

body: 'Сможете организовать любое мероприятие:<br> ● от собственного дня рождения, до свадьбы любимой племянницы, <br> ● от небольшого семинара до масштабной конференции.<br> Узнаете много креативных идей, сможете легко считать и составлять сметы.<br> Научитесь быстро договариваться и грамотно контролировать поставщиков.<br> Спокойно решать экстренные ситуации'

},

];

ngOnInit(): void {

this.navBarElements.push(

{

name: 'Главная',

routerLink: '/home/' + this.id,

fragment: 'top'

},

{

name: 'О нас',

routerLink: '/home/' + this.id,

fragment: 'about'

});

if (this.id !== 0) {

this.navBarElements.push({

name: 'Зарегистрироваться',

routerLink: '/home/' + this.id,

fragment: 'signup',

});

}

}

ngAfterViewInit(): void {

this.route.fragment.subscribe((fragment: string) => {

const el = this.element.nativeElement.querySelector( fragment ? '#' + fragment : '#top');

console.log(fragment ? '#' + fragment : '#top');

el.scrollIntoView();

});

}

exit(): void {

this.authService.logoutUser();

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

}

isLoggedIn(): string {

return this.authService.isLoggedIn() ? 'Выход( ' + localStorage.getItem('userName') + ' )' : 'Вход';

}

onSubmin(form: any): void {

this.searchService.sendPotentialListener(form).subscribe((api: any) => {

if (api.success) {

this.success.successChange('Вы удачно записались на вебинар');

} else {

this.error.errorChange(('Что-то пошло не так, попробуйте позже'));

}

});

}

getTime(startTime: number): string {

return Common.timeConverter(+startTime);

}

openWhatsappModal(): void {

window.open('https://wa.me/77071170179/?text=%D0%94%D0%BE%D0%B1%D1%80%D1%8B%D0%B9%20%D0%B4%D0%B5%D0%BD%D1%8C%2C%20%20%D0%BC%D0%BE%D0%B3%D1%83%20%D1%8F%20%D1%83%D0%B7%D0%BD%D0%B0%D1%82%D1%8C%20%D0%BE%20%D0%BA%D1%83%D1%80%D1%81%D0%B5%20%20%20%D0%BF%D0%BE%D0%B4%D1%80%D0%BE%D0%B1%D0%BD%D0%B5%D0%B5%3F%20', '\_blank');

}

openTG(): void {

window.open('https://t.me/event\_school1', '\_blank');

}

}

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

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

import {AuthService} from '../../\_services/auth.service';

import {InfoService} from '../../\_services/info.service';

@Component({

selector: 'app-conform-email',

templateUrl: './conform-email.component.html',

styleUrls: ['./conform-email.component.scss']

})

export class ConformEmailComponent implements OnInit {

email: string;

code: string;

private returnUrl: any;

constructor(private route: ActivatedRoute, private authService: AuthService, private router: Router, private info: InfoService) {

this.route.paramMap.subscribe(

(params: ParamMap) => {

this.email = params.get('email');

});

this.returnUrl = this.route.snapshot.queryParams[`returnUrl`] || '/';

}

ngOnInit(): void {

}

async sendCode(): Promise<void> {

console.log(this.code);

this.authService.conformEmail(this.code).subscribe((data: any) => {

console.log(data);

if (data.success) {

this.router.navigate([this.returnUrl]);

} else {

this.info.infoChange(data.message);

}

});

}

}

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

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

import {AdminComponent} from './admin.component';

import {RoutingMap} from './routing-map';

const routes: Routes = [

{

path: '',

component: AdminComponent,

children: [

{

path: RoutingMap.scheduleWebinar.routeName,

component: RoutingMap.scheduleWebinar.child,

},

{

path: RoutingMap.allScheduledWebinars.routeName,

component: RoutingMap.allScheduledWebinars.child,

},

{

path: RoutingMap.users.routeName,

component: RoutingMap.users.child,

},

]

}

];

@NgModule({

imports: [RouterModule.forChild(routes)],

exports: [RouterModule]

})

export class AdminRoutingModule { }

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

import {AdminComponent} from './admin.component';

import {AdminRoutingModule} from './admin.routing.module';

import {SideMenuComponent} from './side-menu/side-menu.component';

import {SideMenuItemComponent} from './side-menu/side-menu-item/side-menu-item.component';

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

import {ScheduleWebinarComponent} from './admin-views/schedule-webinar/schedule-webinar.component';

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

import {AllScheduledWebinarsComponent} from './admin-views/all-scheduled-webinars/all-scheduled-webinars.component';

import {SvgModule} from '../../svg/svg.module';

import {WebinarRowComponent} from './admin-views/all-scheduled-webinars/webinar-row/webinar-row.component';

import {UsersComponent} from './admin-views/users/users.component';

import {UserRowComponent} from './admin-views/users/user-row/user-row.component';

import {NgbDatepickerModule, NgbTooltipModule} from '@ng-bootstrap/ng-bootstrap';

import {WebinarService} from '../\_services/webinar.service';

import {ErrorService} from '../\_services/error.service';

import {InfoService} from '../\_services/info.service';

import {NgbDateRuParserFormatter} from '../\_helpers/ngb-date-ru-parser-formatter';

import {ToastModule} from '../components/toast/toast.module';

import {UserService} from '../\_services/user.service';

@NgModule({

declarations: [

AdminComponent,

SideMenuComponent,

SideMenuItemComponent,

ScheduleWebinarComponent,

WebinarRowComponent,

AllScheduledWebinarsComponent,

UsersComponent,

UserRowComponent,

],

imports: [

AdminRoutingModule,

CommonModule,

ReactiveFormsModule,

SvgModule,

NgbTooltipModule,

NgbDatepickerModule,

ToastModule

],

exports: [],

providers: [WebinarService, ErrorService, InfoService, NgbDateRuParserFormatter, UserService]

})

export class AdminModule {

}

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

@Component({

selector: 'app-admin',

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

styleUrls: ['./admin.component.scss']

})

export class AdminComponent implements OnInit {

constructor() { }

ngOnInit(): void {

}

}

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

import {expand} from 'rxjs/operators';

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

import {WebinarService} from '../../../../\_services/webinar.service';

import {NgbCalendar, NgbDate, NgbModal} from '@ng-bootstrap/ng-bootstrap';

import {ModalContentComponent} from '../../../../components/modal-content/modal-content.component';

import {InfoService} from '../../../../\_services/info.service';

import {ErrorService} from '../../../../\_services/error.service';

import {Common} from '../../../../\_helpers/common.helper';

import {NgbDateRuParserFormatter} from '../../../../\_helpers/ngb-date-ru-parser-formatter';

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

import {SuccessService} from '../../../../\_services/success.service';

@Component({

selector: 'app-webinar-row',

templateUrl: './webinar-row.component.html',

styleUrls: ['./webinar-row.component.scss']

})

export class WebinarRowComponent implements OnInit {

@Input() odd: boolean;

@Input() webinar: { id: number, chatroomId: number, start\_time: number, url: string, name: string }

= {id: 0, start\_time: Date.now(), chatroomId: 0, name: '', url: '' };

@Input() last: boolean;

@Output() changeWebinar = new EventEmitter();

expand: boolean;

form: FormGroup;

deleted: boolean;

hoveredDate?: NgbDate;

minDate: NgbDate;

maxDate: NgbDate;

startDate: NgbDate;

constructor(

private webinarService: WebinarService,

private modalService: NgbModal,

private info: InfoService,

private success: SuccessService,

private error: ErrorService,

private calendar: NgbCalendar,

public dateFormatter: NgbDateRuParserFormatter) {

this.minDate = calendar.getPrev(calendar.getToday(), 'm', 6);

this.maxDate = calendar.getNext(calendar.getToday(), 'm', 6);

}

ngOnInit(): void {

const jsStartDate = new Date(+this.webinar.start\_time);

this.startDate = new NgbDate(jsStartDate.getFullYear(), jsStartDate.getMonth(), jsStartDate.getDate());

this.form = new FormGroup({

name: new FormControl(this.webinar.name),

roomId: new FormControl(this.webinar.chatroomId),

startTime: new FormControl(this.startDate),

// this.webinar.start\_time)),

url: new FormControl(this.webinar.url),

});

}

edit(): void {

this.expand = !this.expand;

}

async askIfDelete(): Promise<void> {

const modalRef = this.modalService.open(ModalContentComponent);

modalRef.componentInstance.name = 'Вы действительно хотите удалить? Это действие нельзя отменить';

const res = await modalRef.result;

if (res === 'Ok') {

this.delete();

}

}

delete(): void {

this.webinarService.deleteWebinar(this.webinar.id).subscribe(data => {

if (data.success) {

this.success.successChange('Запись удалена');

} else {

this.error.errorChange(data.message);

}

});

this.deleted = true;

}

async submit(): Promise<void> {

const body: {id: number, name: string, start\_time: number, url: string, chatroomId: number} = {

id: this.webinar.id,

name: this.f.name.value,

chatroomId: this.f.roomId.value,

start\_time: Common.TimestampFromUTC({date: this.f.startTime.value, time: {hour: 0, min: 0}}),

url: this.f.url.value

};

this.webinarService.changeWebinar(body).subscribe(async (data) => {

console.log(data);

if (data.success) {

this.success.successChange('Успешно изменено');

} else {

this.error.errorChange(data.message);

}

},

error => {},

() => this.changeWebinar.emit(this.webinar.id));

}

getTime(startTime: number): string {

return Common.timeConverter(startTime);

}

get f(): {[p: string]: AbstractControl} { return this.form.controls; }

onDateSelection(date: NgbDate, target: any, dateType: string): void {

this.f.startTime.setValue(date);

this.startDate = date;

target.close();

}

isInvalid(attr: string): any {

return (this.form.get(attr)?.touched

|| this.form.get(attr)?.dirty)

&& this.form.get(attr)?.errors;

}

getErrors(attr: string): any {

return this.form.get(attr)?.errors;

}

validateInput(currentValue: NgbDate, input: string): NgbDate | null {

const parsed = this.dateFormatter.parse(input);

return (parsed && this.calendar.isValid(NgbDate.from(parsed))) ? NgbDate.from(parsed) : currentValue;

}

isHovered(date: NgbDate, target: any, minDate: NgbDate, maxDate: NgbDate): any {

return this.hoveredDate && !target && !date.before(minDate)

&& !date.after(maxDate);

}

isDisabled(date: NgbDate, minDate: NgbDate, maxDate: NgbDate): boolean {

return date.before(minDate) || date.after(maxDate);

}

}