import React from "react"

import Image from "next/image"

import styles from "./FreelancerProfile.module.scss"

import ProfileAvatar from "../../../../../assets/ProfileAvatar.jpg"

import Star from "../../../../ui/svg/star/star"

import Link from "next/link"

import { ScrollMenu, VisibilityContext } from "react-horizontal-scrolling-menu"

import useDrag from "../../../../../utils/usedrag.util"

const FreelancerProfile = ({ item }: any) => {

type scrollVisibilityApiType = React.ContextType<typeof VisibilityContext>

const stars = ["star", "star", "star", "star", "star"]

const { dragStart, dragStop, dragMove, dragging } = useDrag()

const handleDrag =

({ scrollContainer }: scrollVisibilityApiType) =>

(ev: React.MouseEvent) =>

dragMove(ev, (posDiff) => {

if (scrollContainer.current) {

scrollContainer.current.scrollLeft += posDiff

}

})

const getAbout =

item.about.length > 155 ? item.about.slice(0, 105) + "..." : item.about

function onWheel(

apiObj: scrollVisibilityApiType,

ev: React.WheelEvent

): void {

const isThouchpad = Math.abs(ev.deltaX) !== 0 || Math.abs(ev.deltaY) < 15

if (isThouchpad) {

ev.stopPropagation()

return

}

if (ev.deltaY < 0) {

apiObj.scrollNext()

} else if (ev.deltaY > 0) {

apiObj.scrollPrev()

}

}

return (

<div className={styles.freelancer}>

<div className={styles.freelancer\_\_content}>

<div className={styles.freelancer\_\_content\_\_smoke}></div>

<div className={styles.freelancer\_\_content\_\_top}>

<div className={styles.freelancer\_\_content\_\_top\_\_avatar}>

<Image

src={`http://localhost:8080/api/${item.avatarPath}`}

alt={""}

width={80}

height={80}

/>

</div>

<div className={styles.freelancer\_\_content\_\_top\_\_info}>

<div className={styles.freelancer\_\_content\_\_top\_\_info\_\_name}>

<Link href={`/users/${item.id}`}>

{item.name} {item.surname}

</Link>

</div>

<div className={styles.freelancer\_\_content\_\_top\_\_info\_\_ratings}>

<div

className={

styles.freelancer\_\_content\_\_top\_\_info\_\_ratings\_\_stars

}

>

{stars.map((star, i) => (

<Star

key={i}

fill={item.rating >= i + 1 ? "#4db82d" : "#878787"}

w={13}

h={13}

/>

))}

</div>

<div

className={

styles.freelancer\_\_content\_\_top\_\_info\_\_ratings\_\_count

}

>

<p>

{item.rating\_to == 0 || undefined || null

? 0

: item.rating\_to.length}{" "}

отзывов

</p>

</div>

</div>

</div>

</div>

<div className={styles.freelancer\_\_content\_\_middle}>

<div className={styles.freelancer\_\_content\_\_middle\_\_description}>

<p dangerouslySetInnerHTML={{ \_\_html: getAbout }}></p>

</div>

</div>

<div

className={styles.freelancer\_\_content\_\_bottom}

onDragStart={(e) => e.preventDefault()}

>

<div onMouseLeave={dragStop} onDragStart={() => false}>

<ScrollMenu

onMouseDown={() => dragStart}

onMouseUp={() => dragStop}

onMouseMove={handleDrag}

onWheel={onWheel}

>

<div

className={styles.freelancer\_\_content\_\_bottom\_\_tags}

onDragStart={(e) => e.preventDefault()}

>

{item.professionsSelected.length == 0

? ""

: item.professionsSelected.map((user: any) => (

<div

className={

styles.freelancer\_\_content\_\_bottom\_\_tags\_\_tag

}

onDragStart={(e) => e.preventDefault()}

>

<p onDragStart={(e) => e.preventDefault()}>

{user.professions.name}

</p>

</div>

))}

</div>

</ScrollMenu>

</div>

</div>

</div>

</div>

)

}

export default FreelancerProfile

import styles from "./Projects.module.scss"

import Link from "next/link"

import dynamic from "next/dynamic"

const Project = dynamic(() => import("../Projects/project/Project"), {

ssr: false,

})

const Projects = ({ orders }) => {

return (

<div className={styles.projects}>

<div className={styles.projects\_\_title}>

<h3>Последние проекты</h3>

<Link href='/orders'>Все проекты</Link>

</div>

<div className={styles.projects\_\_content}>

{orders

? orders.sort((a, b) => b.views - a.views).slice(0, 6).map((order) => <Project item={order} />)

: ""}

</div>

</div>

)

}

export default Projects

import styles from "./Project.module.scss"

import Image from "next/image"

import Eye from "../../../../ui/svg/eye/eye"

import Link from "next/link"

import { useUpdateViewMutation } from "../../../../../store/orders/orders.api"

const Project = ({ item }: any) => {

const getDateOrder = () => {

console.log("item", item)

let a = +new Date()

let b = +new Date(item?.createdAt)

let res = (a - b) / 1000 / 60

let hours = parseInt(res / 60)

let day = parseInt(hours / 24)

if (hours > 24) {

switch (day) {

case 1:

return <p>{day} день назад</p>

break

case 2:

case 3:

case 4:

return <p>{day} дня назад</p>

break

default:

return <p>{day} дней назад</p>

break

}

} else if (res > 60) {

return <p>{hours} час назад</p>

} else {

return <p>{parseInt(res)} минут назад</p>

}

}

const [updateView] = useUpdateViewMutation()

const updateState = async (id) => {

updateView(id)

}

return (

<div className={styles.project} key={item.id}>

<div className={styles.project\_\_content}>

<div className={styles.project\_\_content\_\_top}>

<div className={styles.project\_\_content\_\_top\_\_title}>

<Link onClick={() => updateState(item.id)} href={`/orders/${item.id}`}>{item.title}

</Link>

</div>

</div>

<div className={styles.project\_\_content\_\_middle}>

<div className={styles.project\_\_content\_\_middle\_\_left}>

<div className={styles.project\_\_content\_\_middle\_\_left\_\_price}>

<p>{item.price == 0 ? "Договорная" : item.price + " " + "тг."}</p>

</div>

<div className={styles.project\_\_content\_\_middle\_\_left\_\_feedback}>

<p>{item?.ordersResponse

.length} откликов</p>

</div>

</div>

<div className={styles.project\_\_content\_\_middle\_\_right}>

<div className={styles.project\_\_content\_\_middle\_\_right\_\_added}>

<p>{getDateOrder()}</p>

</div>

<div className={styles.project\_\_content\_\_middle\_\_right\_\_views}>

<div

className={styles.project\_\_content\_\_middle\_\_right\_\_views\_\_ico}

>

<Eye w={20} h={20} />

</div>

<div

className={styles.project\_\_content\_\_middle\_\_right\_\_views\_\_count}

>

<p>{item.views}</p>

</div>

</div>

</div>

</div>

<div className={styles.project\_\_content\_\_bottom}>

<div className={styles.project\_\_content\_\_bottom\_\_tags}>

<div className={styles.project\_\_content\_\_bottom\_\_tags\_\_tag}>

<p>Дизайн сайтов</p>

</div>

<div className={styles.project\_\_content\_\_bottom\_\_tags\_\_tag}>

<p>Дизайн сайтов</p>

</div>

<div className={styles.project\_\_content\_\_bottom\_\_tags\_\_tag}>

<p>Дизайн сайтов</p>

</div>

</div>

</div>

</div>

</div>

)

}

export default Project

import React from "react"

import styles from "./Contests.module.scss"

import Link from "next/link"

import Contest from "./Contest/Contest"

const Contests = () => {

return (

<div className={styles.contests}>

<div className={styles.contests\_\_title}>

<h3>Конкурсы</h3>

<Link href='/'>Все проекты</Link>

</div>

<div className={styles.contests\_\_content}>

{[...new Array(6)].map(() => <Contest />)}

</div>

</div>

)

}

export default Contests

import styles from "./Contest.module.scss"

import Image from "next/image"

import Eye from "../../../../../assets/eye.svg"

import Quest from "../../../../../assets/question.svg"

const Contest = (): JSX.Element => {

return (

<div className={styles.contest}>

<div className={styles.contest\_\_content}>

<div className={styles.contest\_\_content\_\_top}>

<div className={styles.contest\_\_content\_\_top\_\_title}>

<p>Разработать Мобильное приложение для интернет магазина</p>

</div>

</div>

<div className={styles.contest\_\_content\_\_middle}>

<div className={styles.contest\_\_content\_\_middle\_\_left}>

<div className={styles.contest\_\_content\_\_middle\_\_left\_\_price}>

<p>Договорная</p>

</div>

<div className={styles.contest\_\_content\_\_middle\_\_left\_\_feedback}>

<p>25 откликов</p>

</div>

</div>

<div className={styles.contest\_\_content\_\_middle\_\_right}>

<div className={styles.contest\_\_content\_\_middle\_\_right\_\_ends}>

<p>Через день</p>

<div

className={styles.contest\_\_content\_\_middle\_\_right\_\_ends\_\_quest}

>

<Image src={Quest} alt={""} />

</div>

</div>

<div className={styles.contest\_\_content\_\_middle\_\_right\_\_views}>

<div

className={styles.contest\_\_content\_\_middle\_\_right\_\_views\_\_ico}

>

<Image src={Eye} alt={""} width={19} height={14} />

</div>

<div

className={styles.contest\_\_content\_\_middle\_\_right\_\_views\_\_count}

>

<p>232</p>

</div>

</div>

</div>

</div>

<div className={styles.contest\_\_content\_\_bottom}>

<div className={styles.contest\_\_content\_\_bottom\_\_tags}>

<div className={styles.contest\_\_content\_\_bottom\_\_tags\_\_tag}>

<p>Дизайн сайтов</p>

</div>

<div className={styles.contest\_\_content\_\_bottom\_\_tags\_\_tag}>

<p>Дизайн сайтов</p>

</div>

<div className={styles.contest\_\_content\_\_bottom\_\_tags\_\_tag}>

<p>Дизайн сайтов</p>

</div>

</div>

</div>

</div>

</div>

)

}

export default Contest

import React, { FC, ReactNode } from "react"

import styles from "./Layout.module.scss"

import { ILayout } from "./layout.interface"

import Footer from "./footer/Footer"

import { NextPageAuth } from "../../providers/private.route.interface"

import dynamic from "next/dynamic"

const Header = dynamic(() => import("./header/Header"), {

ssr: false,

})

const Layout = ({ children }: ILayout): JSX.Element => {

return (

<div className={styles.layout}>

<Header />

<div>{children}</div>

<Footer />

</div>

)

}

export default Layout

import { createSlice } from "@reduxjs/toolkit"

import { getStoreLocal, removeTokensStorage } from "../../utils/useLocal"

import { register, login, checkAuth } from "./auth.actions"

const initialState = {

user: getStoreLocal("user"),

isLoading: true,

}

export const authSlice = createSlice({

name: "auth",

initialState,

reducers: {

logout: (state) => {

state.user = null

removeTokensStorage()

localStorage.removeItem('user')

},

},

extraReducers: (builder) => {

builder

.addCase(register.pending, (state) => {

state.isLoading = true

})

.addCase(register.fulfilled, (state, { payload }) => {

state.isLoading = false

})

.addCase(register.rejected, (state) => {

state.isLoading = false

state.user = null

})

.addCase(login.pending, (state) => {

state.isLoading = true

})

.addCase(login.fulfilled, (state, { payload }) => {

state.isLoading = false

state.user = payload.user

})

.addCase(login.rejected, (state) => {

state.isLoading = false

state.user = null

})

.addCase(checkAuth.fulfilled, (state, { payload }) => {

state.user = payload.user

})

},

})

export const authReducer = authSlice.reducer

export const { logout } = authSlice.actions

import { createAsyncThunk } from "@reduxjs/toolkit"

import { AuthInterface } from "./auth.interface"

import { IAuthForm } from "../../components/ui/auth-form/auth-form.interface"

import { toast } from "react-toastify"

import "react-toastify/dist/ReactToastify.css"

import { AuthService } from "../../services/auth.service"

export const register: any = createAsyncThunk<AuthInterface, IAuthForm>(

"auth/register",

async ({ email, password, roleId }, thunkAPI) => {

try {

const response = await AuthService.register(email, password, roleId)

toast.success('Регистрация", "Успешна !', {

position: toast.POSITION.BOTTOM\_RIGHT,

})

return response.data

} catch (e) {

toast.error(`Ошибка", ${e}!`, {

position: toast.POSITION.BOTTOM\_RIGHT,

})

return thunkAPI.rejectWithValue(e)

}

}

)

export const login: any = createAsyncThunk<AuthInterface, IAuthForm>(

"auth/login",

async ({ email, password }, thunkAPI) => {

try {

const response = await AuthService.login(email, password)

toast.success("Вы прошли авторизацию", {

position: toast.POSITION.BOTTOM\_RIGHT,

})

return response.data

} catch (e) {

toast.error(`Ошибка", ${e}!`, {

position: toast.POSITION.BOTTOM\_RIGHT,

})

return thunkAPI.rejectWithValue(e)

}

}

)

export const checkAuth = createAsyncThunk(

"auth/check-auth",

async (\_, thunkAPI) => {

try {

const response = await AuthService.getNewTokens()

return response.data

} catch (error) {

return thunkAPI.rejectWithValue(error)

}

}

)

import { Module } from '@nestjs/common'

import { AppController } from './app.controller'

import { AppService } from './app.service'

import { ConfigModule, ConfigService } from '@nestjs/config'

import { TypeOrmModule } from '@nestjs/typeorm'

import { UsersModule } from './users/users.module'

import { AuthModule } from './auth/auth.module'

import { RolesModule } from './roles/roles.module'

import { JwtStrategy } from './auth/stratigies/jwt.strategy'

import { SkillsModule } from './skills/skills.module'

import { SkillsSelectedModule } from './skills-selected/skills-selected.module'

import { GendersModule } from './genders/genders.module'

import { LanguagesModule } from './languages/languages.module'

import { CountriesModule } from './countries/countries.module'

import { OrdersModule } from './orders/orders.module'

import { RatingsModule } from './ratings/ratings.module'

import { ProfessionsModule } from './professions/professions.module'

import { FilesModule } from './files/files.module'

import { MulterModule } from '@nestjs/platform-express'

import { CategoriesModule } from './categories/categories.module'

import { OrdersResponseModule } from './orders-response/orders-response.module'

@Module({

imports: [

MulterModule.register({

dest: './uploads',

}),

ConfigModule.forRoot(),

TypeOrmModule.forRootAsync({

imports: [ConfigModule],

inject: [ConfigService],

useFactory: async () => ({

type: 'postgres',

host: 'localhost',

port: 5432,

username: 'postgres',

password: 'root',

database: 'freelance',

autoLoadEntities: true,

synchronize: true,

}),

}),

UsersModule,

AuthModule,

RolesModule,

SkillsModule,

SkillsSelectedModule,

GendersModule,

LanguagesModule,

CountriesModule,

OrdersModule,

RatingsModule,

ProfessionsModule,

FilesModule,

CategoriesModule,

OrdersResponseModule

],

controllers: [AppController],

providers: [AppService],

})

export class AppModule { }

import { BadRequestException, Injectable, NotFoundException } from '@nestjs/common'

import { InjectRepository } from '@nestjs/typeorm'

import { UsersModel } from './users.model'

import { Repository } from 'typeorm'

import { UsersDto } from './dto/users.dto'

import { genSalt, hash } from 'bcryptjs'

@Injectable()

export class UsersService {

constructor(@InjectRepository(UsersModel) private readonly usersModel: Repository<UsersModel>) { }

async getAll() {

return await this.usersModel.find({ relations: { rating\_to: true } })

}

async getById(id: number) {

const user = await this.usersModel.findOne({

where: { id },

relations: {

countries: true,

roles: true,

genders: true,

languages: true,

skillsSelected: { skills: true },

rating\_to: { owners: true },

rating\_from: true,

professionsSelected: { professions: true },

city: true,

},

})

if (!user) throw new BadRequestException('Пользователь не найден')

return user

}

async updateDescription(id: number, dto: UsersDto) {

const user = await this.getById(id)

user.about = dto.about

user.title = dto.title

await this.usersModel.save(user)

return

}

async updateRole(id: number, dto: UsersDto) {

const user = await this.getById(id)

user.roles = dto.roleId

await this.usersModel.save(user)

return

}

async updateProfile(id: number, dto: UsersDto) {

const user = await this.getById(id)

user.name = dto.name

user.surname = dto.surname

user.countries = dto.countriesId

user.city = dto.cityId

await this.usersModel.save(user)

return

}

async updatePhoto(id: number, dto: any) {

const user = await this.usersModel.findOneBy({ id })

user.avatarPath = dto.avatarPath

await this.usersModel.save(user)

return

}

async getAllFreelancers() {

const freelancers = await this.usersModel.find({

where: {

roles: {

name: 'freelancer',

},

},

relations: {

rating\_to: true,

city: true,

countries: true,

professionsSelected: {

professions: true,

},

},

})

return freelancers.sort((a, b) => b.rating - a.rating)

}

async updateView(login: string) {

const user = await this.usersModel.findOne({

where: { login },

relations: {

countries: true,

roles: true,

genders: true,

languages: true,

skillsSelected: { skills: true },

rating\_to: { owners: true },

rating\_from: true,

professionsSelected: { professions: true },

city: true,

},

})

if (!user) throw new BadRequestException('Пользователь не найден')

user.view++

return await this.usersModel.save(user)

}

}

import React from "react"

import Image from "next/image"

import styles from "./FreelancerProfile.module.scss"

import ProfileAvatar from "../../../../../assets/ProfileAvatar.jpg"

import Star from "../../../../ui/svg/star/star"

import Link from "next/link"

import { ScrollMenu, VisibilityContext } from "react-horizontal-scrolling-menu"

import useDrag from "../../../../../utils/usedrag.util"

const FreelancerProfile = ({ item }: any) => {

type scrollVisibilityApiType = React.ContextType<typeof VisibilityContext>

const stars = ["star", "star", "star", "star", "star"]

const { dragStart, dragStop, dragMove, dragging } = useDrag()

const handleDrag =

({ scrollContainer }: scrollVisibilityApiType) =>

(ev: React.MouseEvent) =>

dragMove(ev, (posDiff) => {

if (scrollContainer.current) {

scrollContainer.current.scrollLeft += posDiff

}

})

const getAbout =

item.about.length > 155 ? item.about.slice(0, 105) + "..." : item.about

function onWheel(

apiObj: scrollVisibilityApiType,

ev: React.WheelEvent

): void {

const isThouchpad = Math.abs(ev.deltaX) !== 0 || Math.abs(ev.deltaY) < 15

if (isThouchpad) {

ev.stopPropagation()

return

}

if (ev.deltaY < 0) {

apiObj.scrollNext()

} else if (ev.deltaY > 0) {

apiObj.scrollPrev()

}

}

return (

<div className={styles.freelancer}>

<div className={styles.freelancer\_\_content}>

<div className={styles.freelancer\_\_content\_\_smoke}></div>

<div className={styles.freelancer\_\_content\_\_top}>

<div className={styles.freelancer\_\_content\_\_top\_\_avatar}>

<Image

src={`http://localhost:8080/api/${item.avatarPath}`}

alt={""}

width={80}

height={80}

/>

</div>

<div className={styles.freelancer\_\_content\_\_top\_\_info}>

<div className={styles.freelancer\_\_content\_\_top\_\_info\_\_name}>

<Link href={`/users/${item.id}`}>

{item.name} {item.surname}

</Link>

</div>

<div className={styles.freelancer\_\_content\_\_top\_\_info\_\_ratings}>

<div

className={

styles.freelancer\_\_content\_\_top\_\_info\_\_ratings\_\_stars

}

>

{stars.map((star, i) => (

<Star

key={i}

fill={item.rating >= i + 1 ? "#4db82d" : "#878787"}

w={13}

h={13}

/>

))}

</div>

<div

className={

styles.freelancer\_\_content\_\_top\_\_info\_\_ratings\_\_count

}

>

<p>

{item.rating\_to == 0 || undefined || null

? 0

: item.rating\_to.length}{" "}

отзывов

</p>

</div>

</div>

</div>

</div>

<div className={styles.freelancer\_\_content\_\_middle}>

<div className={styles.freelancer\_\_content\_\_middle\_\_description}>

<p dangerouslySetInnerHTML={{ \_\_html: getAbout }}></p>

</div>

</div>

<div

className={styles.freelancer\_\_content\_\_bottom}

onDragStart={(e) => e.preventDefault()}

>

<div onMouseLeave={dragStop} onDragStart={() => false}>

<ScrollMenu

onMouseDown={() => dragStart}

onMouseUp={() => dragStop}

onMouseMove={handleDrag}

onWheel={onWheel}

>

<div

className={styles.freelancer\_\_content\_\_bottom\_\_tags}

onDragStart={(e) => e.preventDefault()}

>

{item.professionsSelected.length == 0

? ""

: item.professionsSelected.map((user: any) => (

<div

className={

styles.freelancer\_\_content\_\_bottom\_\_tags\_\_tag

}

onDragStart={(e) => e.preventDefault()}

>

<p onDragStart={(e) => e.preventDefault()}>

{user.professions.name}

</p>

</div>

))}

</div>

</ScrollMenu>

</div>

</div>

</div>

</div>

)

}

export default FreelancerProfile