План:

Створення grid

Розмітка

Підготовка графіки

Оформлення елементів

Оформлення контенту

Анімація інтерактивних елементів

Оптимізація

Захист перегляду сторінки через пароль що надається лише в резюме (доопрацювання роутингу та модулю вітання)

PageNotFoundComponent (<https://angular.io/guide/router>) path: '\*\*', component: NotFoundComponent

Розгорнуто проект Angular

Підключено Bootstrap, jQuery, popper.js

**npm install bootstrap**

Інтегровано стилі Bootstrap та додаткові елементи до angular.json:

"styles": [

"src/styles.css",

"./node\_modules/bootstrap/dist/css/bootstrap.min.css"

],

"scripts":[

"./node\_modules/jquery/dist/jquery.min.js",

"./node\_modules/bootstrap/dist/js/bootstrap.min.js"

]

Перевірено коректність роботи

Створено модулі додатку: header, content, footer

Створено складові компоненти модулів

У модулі content створюємо content-routing.module, прописані шляхи переходів до окремих компонент, внесені данні до глобального app-routing.module

Створено модуль Sequre, додано secure-routing.module, та компоненту access-check

В компоненті access-check створено базову розмітку на AngularMaterial (встановлено та підключено усі його складові ) з формою уведення коду

До класу компоненти access-check додано значення паролю з environments, створено реактивну форму, додано ErrorStateMatcher (**працює лише з AngularMaterial** ), додано метод getAccess що здійснює порівняння базового паролю із введеним.

Метод getAccess у випадку коректного ведення паролю зберігає до LocalStorage повідомлення, якщо введено невірний пароль обмежується кількість спроб введення.

В розмітку додано поля що інформують про некоректність введення не валідного паролю, та про обов’язковість заповнення поля, спрацьовують по даним валідатора форми:

*\*ngIf*="passwordForm.get('password').errors?.required"

Для виведення інтерактивних повідомлень підключено **primeng** підключено стилів angular.json, **(**<https://www.primefaces.org/primeng/#/toast>**),** налаштовано функціонал інтерактивний повідомлень.

Створено глобальний сервіс ***contro-access.service.ts*** перевірки доступу, після підтвердження доступу видається дозвіл (true/false)

У класі сервісу створено два методи перший для отримання значення з локального сховища через get

get getAccess() {

*return* localStorage.getItem('mlp\_client\_token');

}

Та другий метод що надає данні про доступ

Додано захист роутигну: ng g guard guards/auth -> CanActive

Модуль захисту:

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

*import* { ActivatedRouteSnapshot, RouterStateSnapshot, UrlTree, CanActivate, Router } *from* '@angular/router';

*import* { ControAccessService } *from* '../global-services/contro-access.service';

@Injectable({

providedIn: 'root'

})

*export* class AuthGuard implements CanActivate {

constructor(

private access: ControAccessService,

private router: Router

) {}

canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot): boolean {

*if* (*this*.access.keeper()) {

*return* true;

} *else* {

*this*.router.navigate([`/secure`]);

*return* false;

}

}

}

Написання Хедеру:

В модулі хедеру прописуємо експорт компоненти хедеру

Та обов’язково імпортуємо RouterModule та реєструємо в imports

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

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

*import* { SiteHeaderComponent } *from* './site-header/site-header.component';

*import* { RouterModule } *from* '@angular/router';

@NgModule({

declarations: [SiteHeaderComponent],

imports: [

CommonModule,

RouterModule

],

exports: [ SiteHeaderComponent ]

})

*export* class HeaderModule { }

Додано модуль хедеру до загального app.module.ts

Створюємо базову розмітку хедеру

Додано компоненту хедеру в розмітку app.component.ts

Додано навігацію з routerLink

Створено розмітку розділу Skills

Додано посилання на розділ проектів через routerLink="/content/projects"

Створено розмітку для розділу Projects

Заблоковано відображення хедеру на початковій сторінці sequre:

В app.component.html

<ng-container *\*ngIf*="showHeader">

<app-site-header></app-site-header>

</ng-container>

App.component.ts:

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

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

*import* { ControlAccessService } *from* './global-services/control-access.service';

@Component({

selector: 'app-root',

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

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

})

*export* class AppComponent implements OnInit {

public title = 'portfolio';

public showHeader: boolean;

constructor(

private access: ControlAccessService,

private router: Router

) {}

ngOnInit(): void {

*this*.router.events.subscribe((event) => {

*if* (event instanceof NavigationEnd) {

*this*.showHeader = *this*.access.keeper();

}

});

}

}

Тригер на куруванням відображення хедеру спрацьовує у сервісі ***ControlAccessService, компонента app*** відстежує кожну подію переходу навігації та під час завершення навігації контролює дозвіл на доступ.

Змінено тему AngularMaterial, відредаговано відображення інпуту.

Перевірено підключення ***jQuery:***

Npm I jquery

Add to angular.json -> scriprs: jqiery.min.js

In component: declare const $: any; or ***import \* as $ from “jquery”***

У класі компоненти де використовується карусель прописати:

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

declare const $: any;

На ініціалізації:

ngAfterViewInit(): void {

    $(document).ready(

      () => $('.owl-carousel').owlCarousel({

        items: 3,

        loop: true,

        center: true,

        nav: true,

        navText: ['&larr;', '&rarr;'],

        autoplay: true,

        autoplayTimeout: 5000,

        autoplayHoverPause: true,

        autoplaySpeed: 1200,

        smartSpeed: 750,

        responsive: {

          0: {

            items: 1

          },

          768: {

            items: 2

          },

          1024: {

            items: 3

          }

        }

      })

    );

  }

Для того щоб вставка даних в карусель працювала правильно з ngFor потрібно ініціалізувати її на етапі:  ***ngAfterViewInit***

Яке представлення в каруселі буде

Налаштовано відображення зображень в каруселі:

*.imaje-wrapper* {

overflow: hidden;

height: 380px;

}

*.imaje-wrapper* img {

display: block;

border-radius: 5px;

height: 100%;

object-fit: cover;

}

Кольорова гамма сайту:

Фони тла:

#313131

#2f2f2f

Кольори шрифтів

#ead4b2 стандарт

#ffc353 активний стан

#fff малкий тескст

#cbb593 – текс в тіні

Фони блоків

#e9d2af – основний

#e1c292 – додатковий для виділення блоку

Створено компоненту футеру, в модулі прописаний експорт для використання по усьому прокту, у app.module.ts заімпортовано модуль футру.

На сторінці з проектами виникла проблема адаптивності розмітки оскільки не знаю які розміри задати на карусель і для яких брейкпоінтів.

Вирішено: переробити хедер без аватару, і на блок з проектами встановити висоту через calc(100vh- (hh+hf)).

Додано на хедер та футер:

box-shadow: 0 0 10px 5px rgba(104, 102, 102, 0.8);

Зроблено адаптивність сторінки проектів, додано функціонал відображення та приховання навігації

Відкориговано маршрутизацію згідно до вимог документації

***Анімація сторінок розділу content:***

1. App.module.ts

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

imports: [

…………………………………………………

BrowserAnimationsModule,

………………………………………………………

AppRoutingModule,

],

1. Додано об’єкти параметрів у дочірній модуль маршрутизації

content-routing.module.ts

{path: 'skills', component: SkillsComponent, data: {animation: 'first'} },

{path: 'projects', component: EducationalProjectsComponent, data: {animation: 'second'} },

1. Створено виділений файл анімації app.animation.ts

Всередину імпортовано елементи анімації ангулару та експортовано змінну анімації

*export* const slideAnimation = trigger('routeAnimation', [

1. До файлу app.component.ts

*import* { RouterOutlet } *from* '@angular/router';

*import* { slideAnimation } *from* './app.animation';

та метод обробки даних з шаблону

getAnimationData(outlet: RouterOutlet) {

*return* outlet && outlet.activatedRouteData &&

outlet.activatedRouteData['animation'];

}

1. До файлу app.component.html додано обгортку на <router-outlet>

<div *[@routeAnimation]*="getAnimationData(routerOutlet)">

<router-outlet *#routerOutlet*="outlet"></router-outlet>

</div>

Project hover Overlay:

<div *class*="container">

<img *src*="some Img.png" *alt*="Avatar" *class*="image">

<div *class*="overlay">

<div *class*="text">Hello World</div>

</div>

</div>

Доповнено анімацію проекту

Введено описи проектів.

Відредаговано стилі та перевірена адаптивність сторінки проектів.

На сторінці вмінь змінено шаблон відображення, перший варіант дизайну з градієнтним розділювачем між зображенням та інформативним блоком.

Зролено сторінку про себе.

Виправлено стилі елементів що збились.

Перевірена адаптивність сторінки вмінь

Змінено header, адаптивність, відкорегованостилі.

Змінено футер, модифіковано сторінку умінь для телефонів, виправлено шрифти та блоки на сторінці про себе.