Eines HTML y CSS II

PAC2



Jaume Ramon Martínez  
Màster universitari de Desenvolupament de Llocs i Aplicacions Web

1. Consideracions Inicials

L’enunciat de la PAC2 és el següent:  
  
Caldrà que elaboris un lloc web de tenir quatre pàgines (pàgina de portada i tres pàgines interiors) on la temàtica del lloc web ha de ser informació sobre les jornades de seguretat informàtica que organitzeu.   
Tota la informació que no es proporciona amb l’enunciat, així com la línia gràfica, la pots decidir tu segons els gustos o et convingui.

El repertori GitHub on està localitzat el codi font és el següent:

<https://github.com/JaumeRM/WebEinesHTMLiCSS2PAC2>

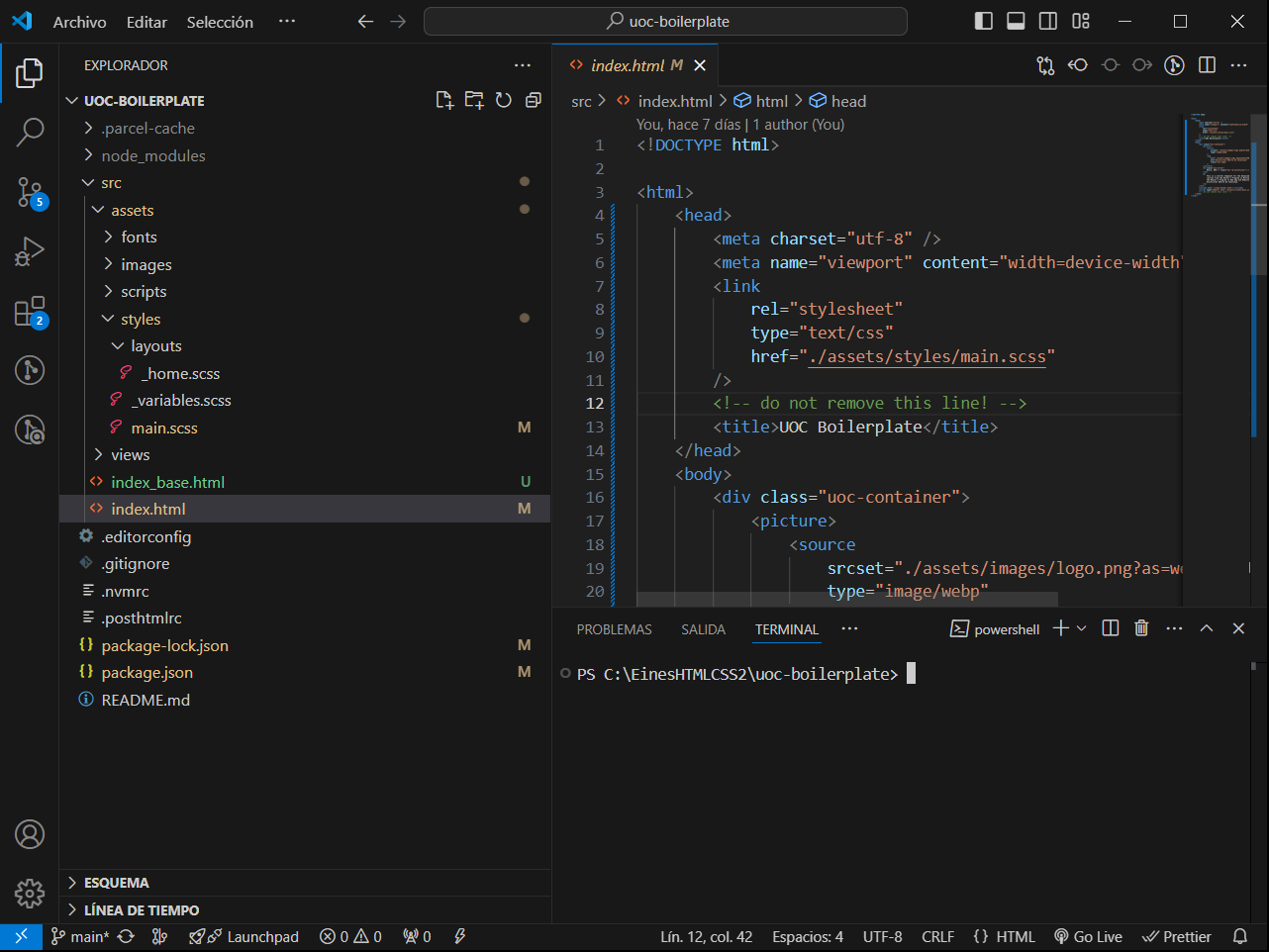
El lloc web publicat és el següent:

<https://jaumer-eines-html-css2-pac2.netlify.app/>

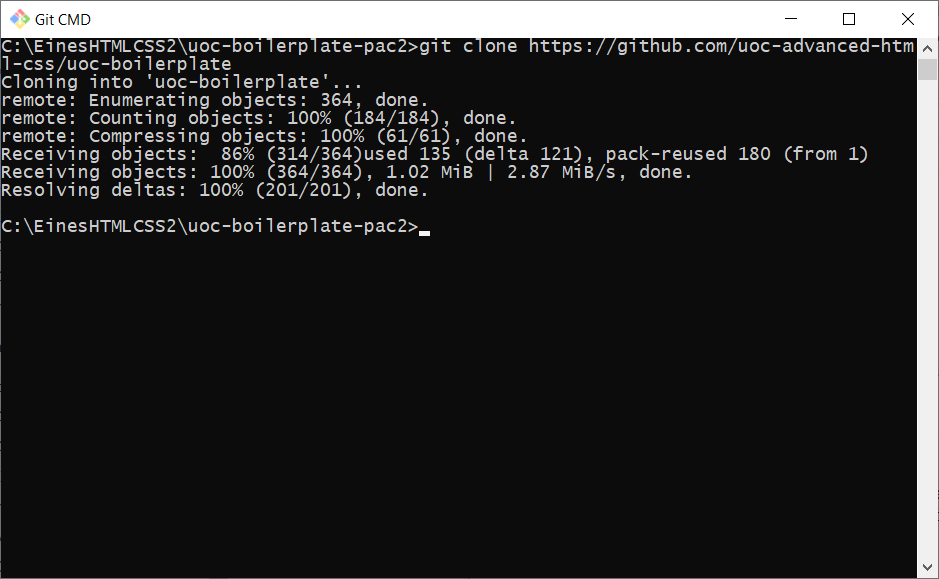
1. Instal·lació i configuració de l’entorn.

En aquest primer mòdul detallarem com s’ha realitzat la instal·lació i configuració de l’entorn de desenvolupament per poder realitzar la PAC2.

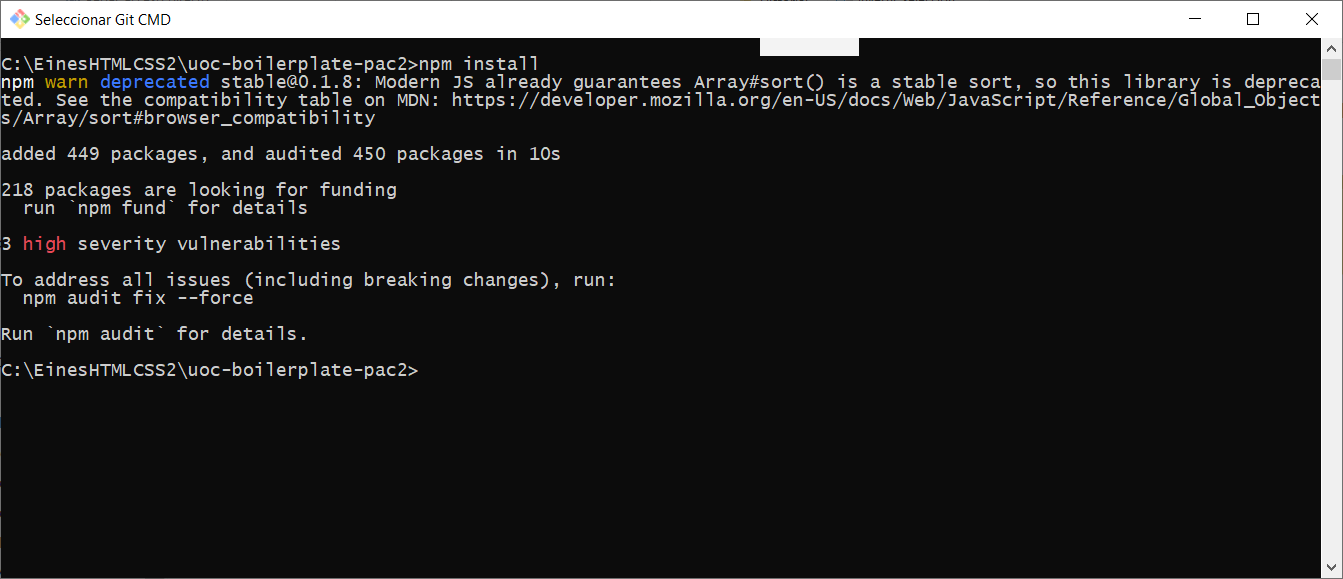
Com a entorn de desenvolupament s’ha escollit mantenir “*Visual Studio Code*”, ja que per la realització de la PAC1 ha estat una bona eina de desenvolupament per dur-la a terme.



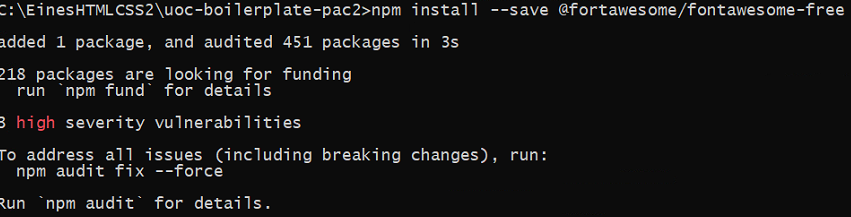
En primer lloc, es realitza la còpia del Boilerplate de la UOC en el repertori local per tal de poder treballar amb ell, aquest cop anomenat “uoc-boilerplate-pac2” per mantenir les diferències amb la PAC1.



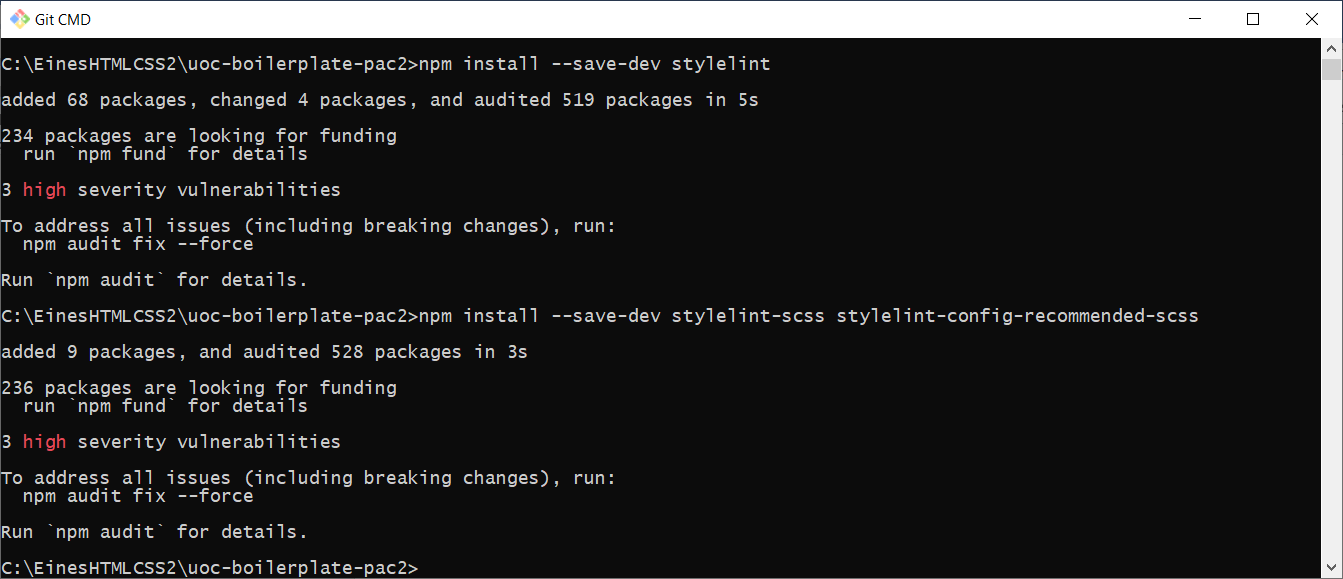
Un cop el tenim, es procedeix a realitzar la instal·lació dels paquets necessaris per fer-lo funcionar mitjançant “npm”.



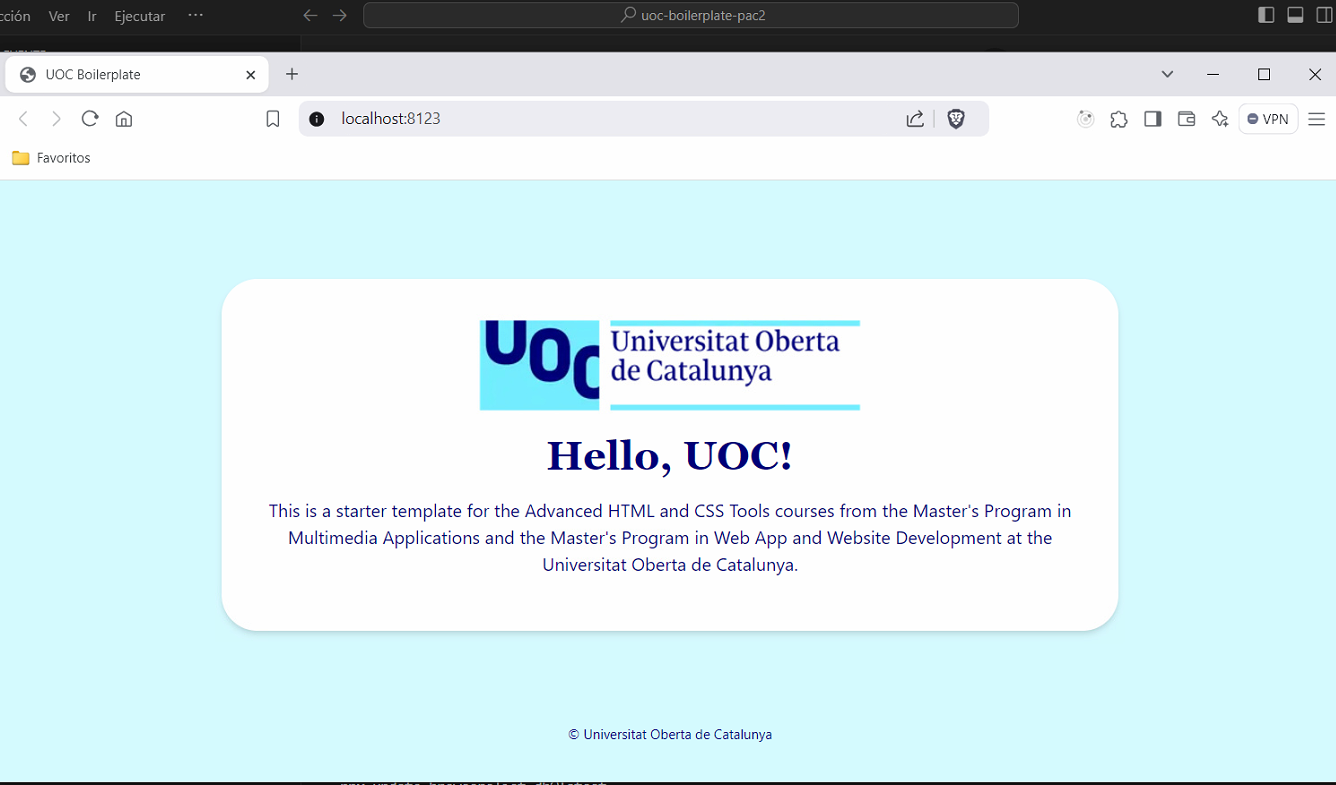
S’instal·la també el complement de “*Font Awesome*”



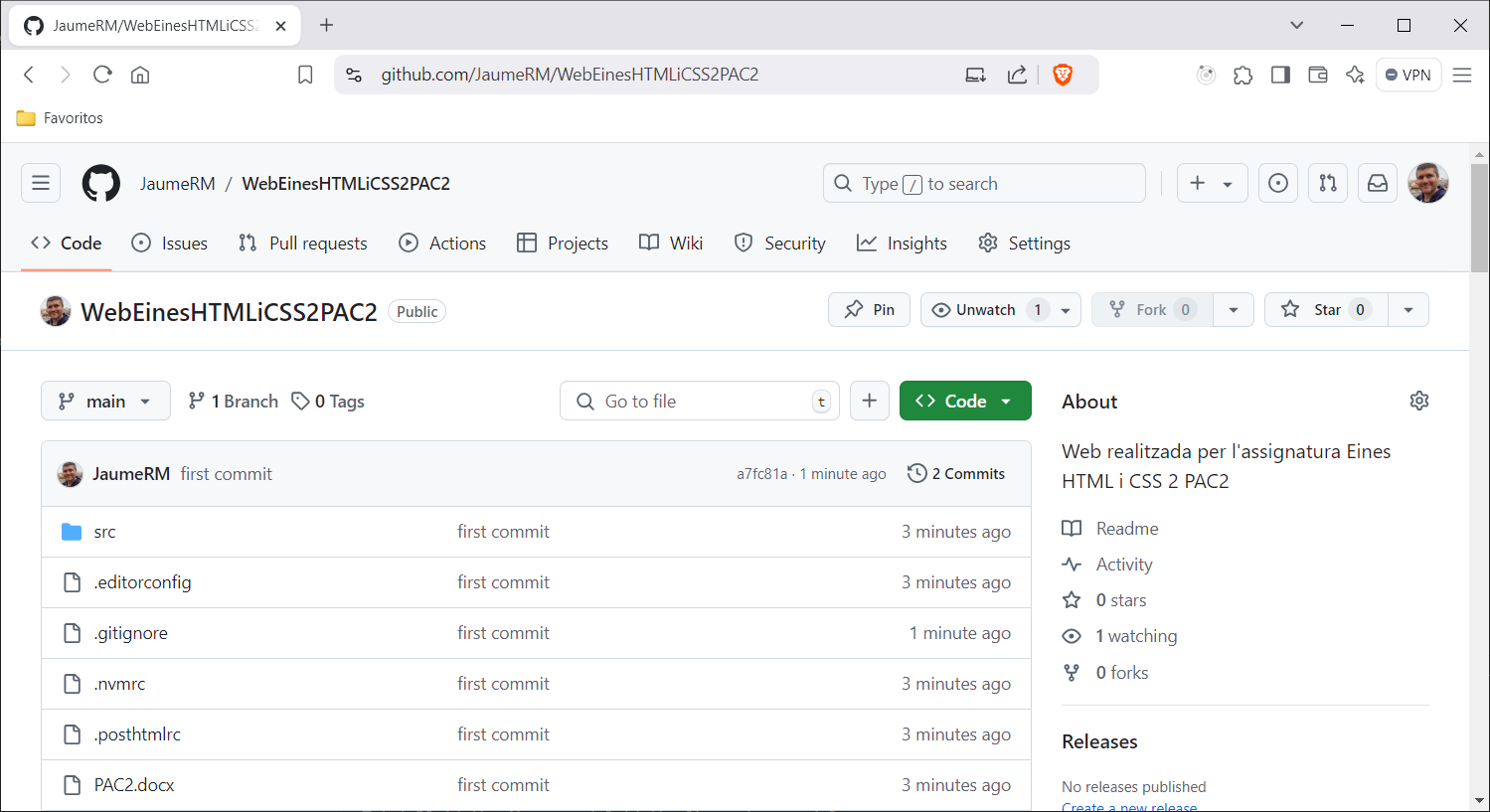
I “*Styleint*” juntament amb la configuració recomanada



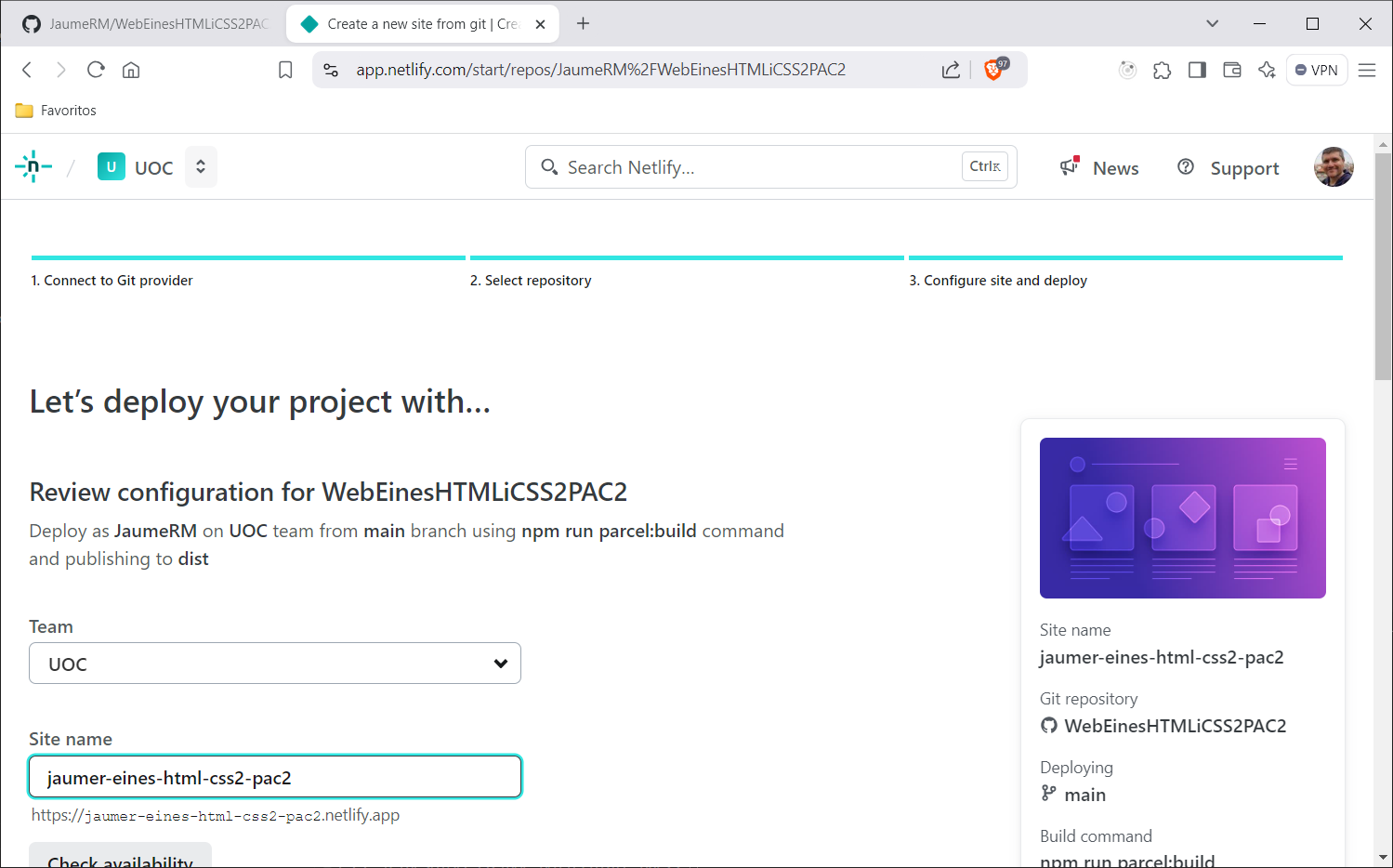
Una vegada es te l’entorn, es realitza una prova per veure la compilació i el funcionament mitjançant l’ordre “npm run dev”



Al tenir aquesta primera versió funcionant, es realitza una pujada del codi font a GitHub, a l’espai personal mitjançant un “git push”

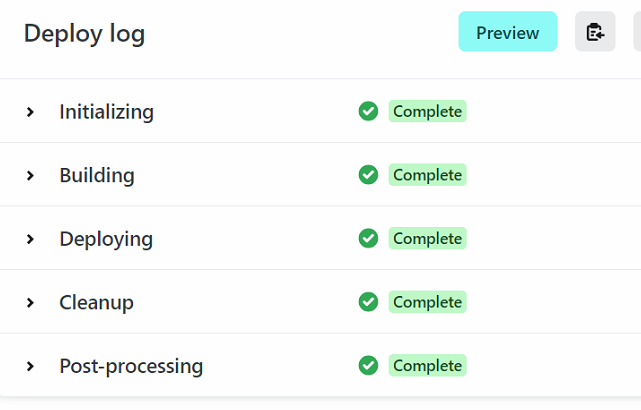


I, un cop es te pujat el projecte a GitHub, es realitza l’enllaç amb Netlify per tal que cada pujada del projecte, es sincronitzi amb el lloc web de producció.

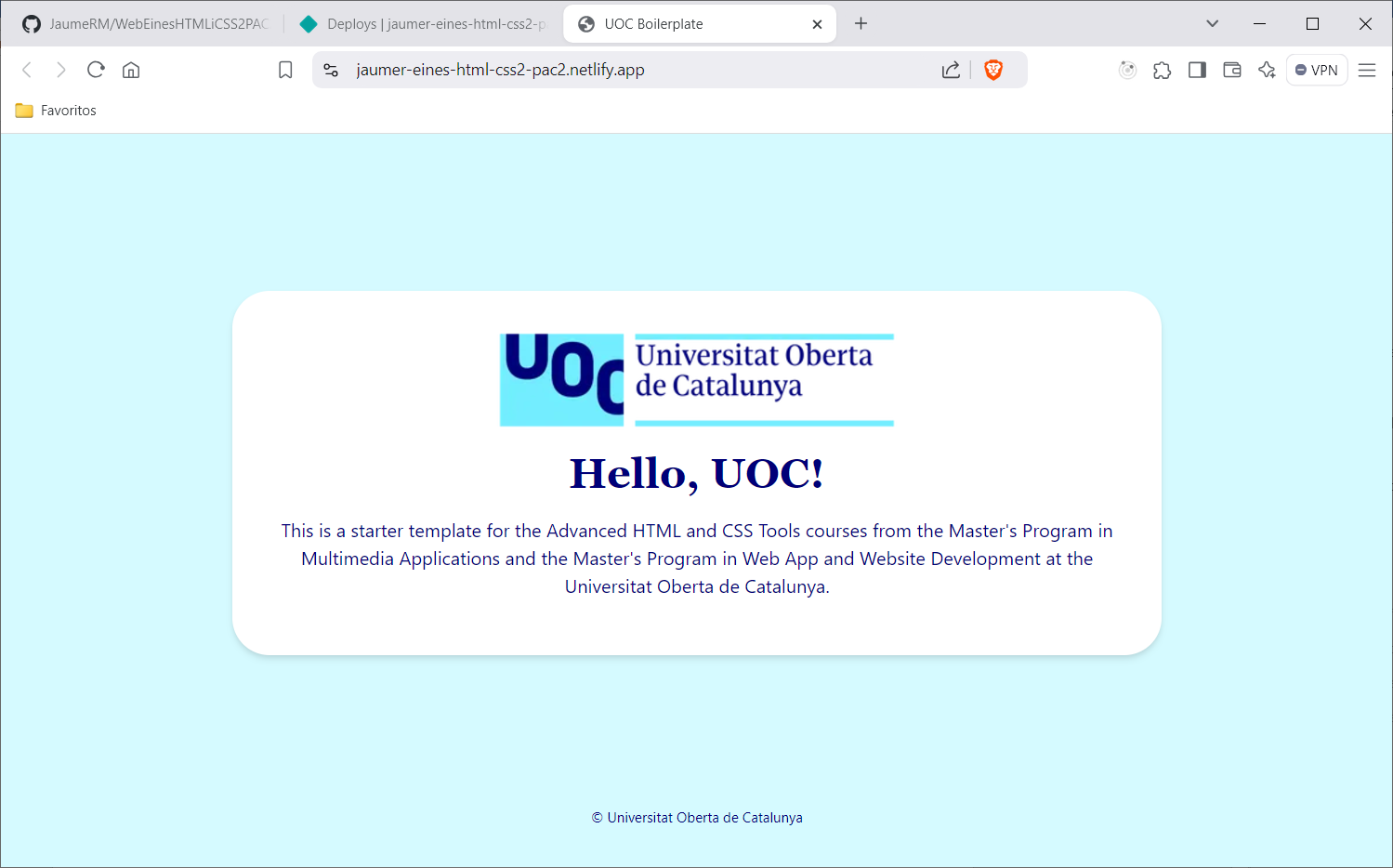


I així, cada cop que es realitza una pujada, es produirà un desplegament.





I, com tot ha anat be, tenim la versió del Boilerplate desplegada.



1. Requeriments i estudi.

En aquest segon mòdul detallarem com s’han realitzat les eleccions per tal de poder realitzar el desenvolupament del lloc web i els motius que han portat a fer-ho.

**Requeriments**

Donat l’enunciat de la PAC, aquesta hauria de complir els següents requeriments:

* El lloc web ha de tenir quatre pàgines (pàgina de portada i tres pàgines interiors):
  + La pàgina de portada: ha d’estar maquetada amb CSS Grid i has de realitzar algun disseny interessant, semblant al què podria ser un poster promocional de les jornades.
  + Una de les pàgines interiors ha d'incloure una retícula amb informació sobre els participants, maquetada amb flex.
  + Una de les pàgines interiors ha de seguir un format d’article i incloure un mínim de 4 paràgrafs de text i 2 fotografies.
  + La tercera pàgina interior és de contingut completament lliure.
* El lloc web ha de ser responsive i s’ha de poder visualitzar correctament des de qualsevol dispositiu modern (telèfon, tauleta, ordinador…)..
* Cal partir de UOC Boilerplate per al teu desenvolupament.
* Cal que facis una tria d’una o més de les metodologies i guies d’estil estudiades al mòdul 2 i l’apliquis al teu desenvolupament.
* Cal usar Bootstrap 5, carregat a UOC Boilerplate i personalitzat amb Sass com s’indica als materials, i usar un mínim de 4 components diferents.
* Cal utilizar container queries a les 4 pàgines. A més, has d’utilitzar de manera coherent capes de cascada a dues de les quatre pàgines.
* A banda, cal incorporar i usar alguna altra dependència externa, com ara FontAwesome o qualsevol altra que consideris interessant.
* A més d’aquesta dependència, cal que hi instal·lis Stylelint i en personalitzis la configuració perquè apliqui els criteris d’estils escollits, amb plugins i normes.
* Cal utilitzar necessàriament les següents característiques de Sass: variables, imbricació (nesting), funcions, parcials i importació.
* Un cop finalitzat el desenvolupament, cal que publiquis el codi a GitHub i facis un deployment a Netlify, com s’explica a la guia del mòdul 1

**Guies d’Estil**

Un cop llegits els continguts teòrics de la PAC, se’ns demana seguir una guia d’estil per a fer el projecte.

A l’hora de segur el projecte, s’intentarà seguir **la guia d’estil de Mark Otto** definida a https://codeguide.co/.

El motiu d’escollir aquesta guia és que ha estat, un cop llegida, la més senzilla de seguir, al estar tot en una pàgina, i la que te punts més lògics i estructurats, sobretot a l’hora de repassar els continguts a la guia, per veure si s’han fet correctament.

**Metodologia**

De la mateixa manera que amb la guia d’estil, un cop llegits els continguts teòrics de la PAC, escollim una metodologia.

La escollida en aquest cas ha estat la **Metodologia BEM**.

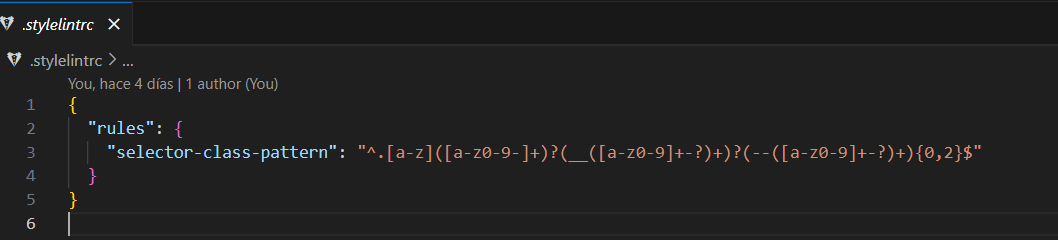
El motiu d’aquesta elecció ha estat que, d’entre les opcions, m’ha semblat la més lògica a l’hora de nombrar i organitzar les classes al CSS.

Per exemple, a l’hora de nombrar les classes, queda clar a simple vista la seva funció, ja que una classe anomenada “*button\_\_label—error*” indica clarament que fa referència la etiqueta d’un element botó quan hi ha un error.

Alguns dels avantatges, els quals comparteixo, es poden llegir en aquest article:  
<https://medium.com/@techathoncert/css-bem-concept-and-best-practices-dce695cd3530>

Llavors, un cop escollida la metodologia, adaptem el complement Stylelint del boilerplate a dita configuració.

Per tant, afegim al fitxer de configuració la següent línia:

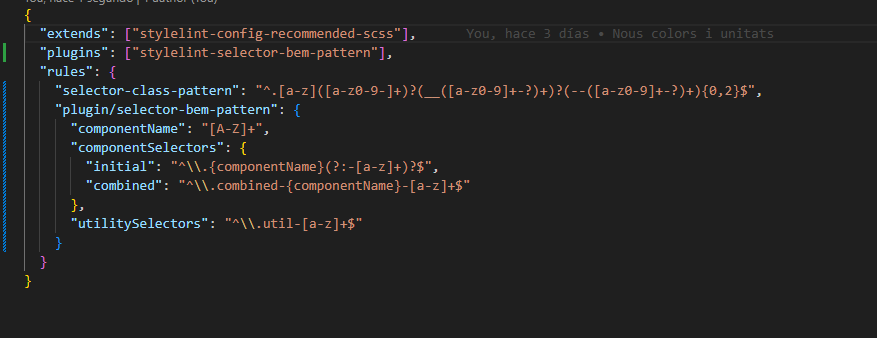


A part de la configuració estàndard, es decideix cercar un plugin de Stylelint específic de BEM, amb lo que es descarrega el següent:

<https://www.npmjs.com/package/stylelint-selector-bem-patter>



I es modifica el fitxer de configuració per tal que faci també una inspecció amb el nou plugin.



I, tal i com s’aconsella al contingut teòric, afegim que a l’hora de fer el *build*, realitzi també una comprovació de Styleint



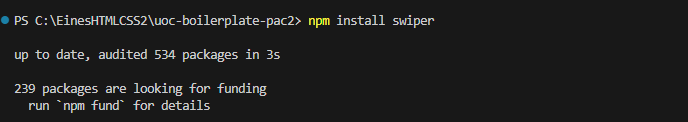
**Dependències externes**

A l’enunciat de la PAC, se’ns demana utilitzar altres dependències externes.

Per defecte, en l’enunciat proposa la ja instal·lada “*include*” i la que s’instal·la “*font awesome*”, a part d’ella farem servir una altra anomenada “*swiper*”.

<https://swiperjs.com/get-started>

La idea d’aquesta dependència externa, coneguda per un missatge del fòrum de la primera PAC, és la de crear un carrusel d’imatges que s’integrarà en la pàgina de lliure contingut.

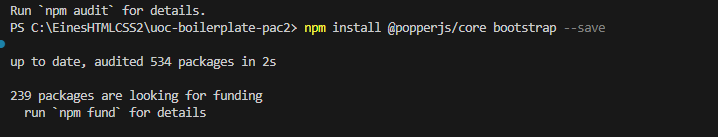


L’ús d’aquesta dependència externa es veure més detallat en la part de “Desenvolupament” d’aquesta memòria.

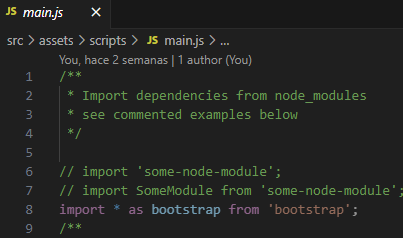
També s’ha utilitzat, tot i ja venir instal·lada amb Parcel, el mòdul Sharp, per poder fer gestió d’imatges.

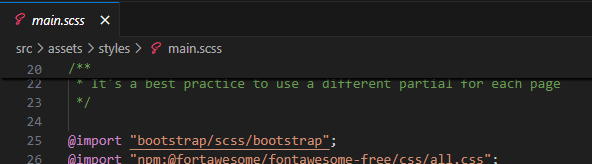
**Bootstrap**

Tal i com indica l’enunciat s’ha de fer servir Bootstrap per realitzar el desenvolupament, per tant es realitza la instal·lació seguint la documentació.



Un cop fet, s’integra al projecte tot seguint les indicacions de la documentació realitzant les importacions als fitxers escaients.





Amb lo que ja tenim tot llest per implementar Bootstrap on sigui necessari.

**Disseny Responsive**

A l’hora de fer el disseny responsive de la aplicació, hauríem de seguir els requeriments de la aplicació:

* Índex: CSS Grid.
* Speakers: Flex
* Sobre: Bootstrap
* Inscripció: Bootstrap

Donat que pel primer i pel segon tenim definits com ha de ser el disseny responsive, però per les altres dos pàgines no, es decideix d’utilitzar les eines “*row*” i “*col-xx*” que ens ofereix bootstrap,   
El motiu de fer servir aquest sistema per és que, amb [projectes personals anteriors](https://www.eldoblatge.com), ja havia treballat, amb lo que és una bona solució i ja coneguda per realitzar el disseny responsive.

El disseny hauria de contenir un capçal, un contingut i un peu de pàgina.

Dintre del capçal estaria contingut el logo i un navegador a les altres pàgines, mentre que el footer hi haurà una informació de l’autoria així com un navegador en forma de *selectbox*, ambdós responsive que s’adaptaran l’espai horitzontal de la pantalla.

Al tenir un disseny amb wireframes, es realitza una mica de disseny de la versió per dispositius mòbils, però donat que tenim els wireframes inicials en forma de la versió Desktop, es decideix treballar amb un enfocament “*Desktop First*”.  
  
A part del motiu anterior, també donat que, en la experiència de desenvolupament que disposo, al haver treballat gairebé sempre en aquest enfocament, és més còmode començar amb “*Desktop First*” per fer després l’adaptació a “*Mobile*”.

Per tant, el disseny (realitzat amb **FIGMA**) dels Wireframes amb versió mòbil quedaria de la següent manera:

*Pàgina d’inici*

Sense disseny en grid i amb els components un a sobre de l’altre



*Pàgina de Speakers*

Amb el disseny en flex i amb els components un a sobre de l’altre quan l’amplada no permeti tenir-los al costat



*Pàgina de l’article (sobre nosaltres)*

Amb el disseny en bootstrap, aprofitant el row i el col per tal que faci el disseny automàticament



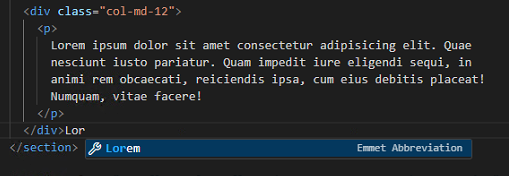
*Pàgina lliure (inscripció a cursos)*

Amb el disseny en bootstrap, aprofitant el *row* i el *col* per tal que faci el disseny automàticament



**Maquetació**

Per tal de fer la maquetació de les pàgines web, en un primer lloc, a fi d’emplenar el contingut s’utilitzarà el conegut text “Lorem Ipsum”, ja que Visual Studio Code proporciona la utilitat que escrivint “Lorem” emplena automàticament de contingut amb dit text.

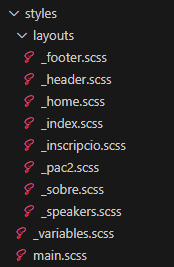


1. Desenvolupament.

En aquest tercer mòdul detallarem alguns aspectes sobre com s’ha realitzat el desenvolupament de la pàgina web, tot i que en algunes coses no entrarem en detall, ens centrarem en les especificitats dels requeriments de la PAC.

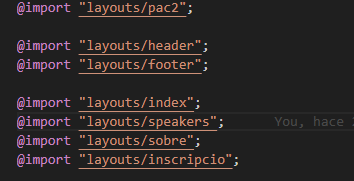
**Creació dels CSS**

A l’hora de crear els CSS, s’ha utilitzat la estructura del Boilerplate facilitat i, per tal de mantenir l’històric, enlloc d’utilitzar el fitxer “\_home.scss” s’ha crear una nova estructura de fitxers dintre de la carpeta “layouts”, on es posarà un fitxer per cada pàgina específica, més un fitxer anomenat *“\_pac2.scss*” per les declaracions que s’utilitzen a totes les pàgines i un altre fitxer anomenat “\_variables.scss” per tal de definir les variables que s’utilitzaran en el codis CSS.



El motiu de fer-ho així i no en un fitxer gran amb tot inclòs com es va fer en la PAC1, és que al ser un projecte més gran, ja es requereix més contingut i, per tant, seguint la filosofia de “divideix i venceràs”, fent la separació en diferents fitxers parcials ajuda a la localització i encapsulació del codi a fi de poder tenir un bon manteniment.

Llavors, dintre del “mains.css”, que s’utilitza dintre de les pàgines “\*.html”, es crida amb un @import als fitxers.

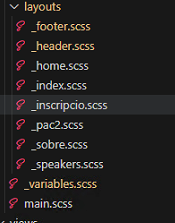


**Aplicació de la guia d’estil**

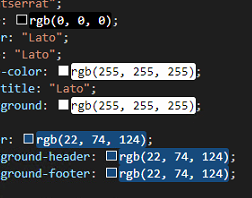
En el moment de realitzar el desenvolupament del SCSS s’ha intentat seguir la guia d’estil escollida.

A mode d’exemple alguns detalls:

* Divisió del CSS en categories, per exemple General, Header, Footer, etc.



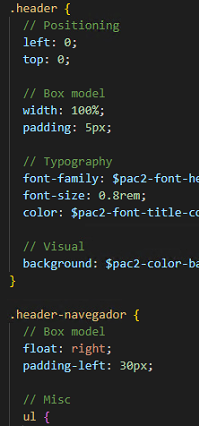
* Utilització de colors en RGB



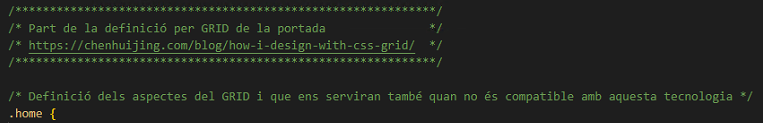
* Espai de 2 columnes



* Ordre de les declaracions (Positioning / Box model / Typographic / Visual / Misc)



* Comentaris indicatius i amb font d’on ha pogut treure el codi en casos que s’ha reutilitzat.

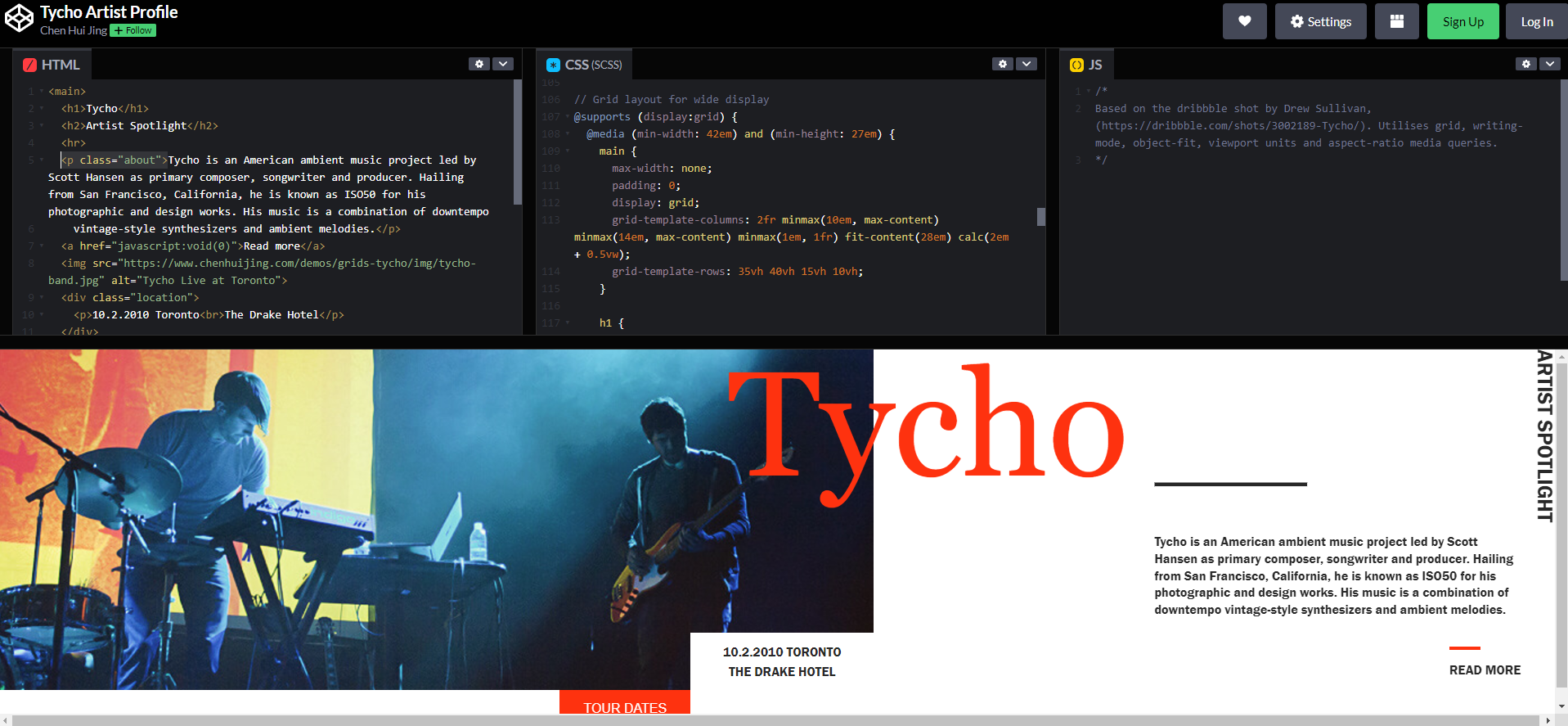


Aquestes han estat unes quantes de les guies seguides aplicades que estan detallades al lloc web de codeguide.co

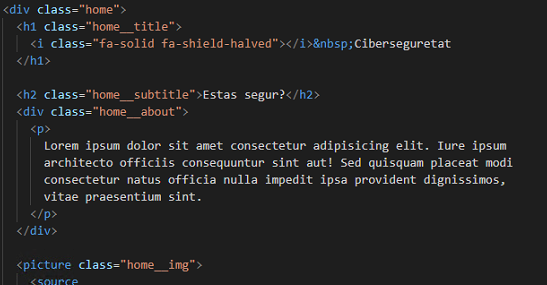
Ara be, algunes com “*Media query placement*”, que indica que s’ha de posar els media queries a prop de la regla, no s’han seguit i s’han posat al final, ja que crec que el projecte no és tan gran com per haver-la de seguir.

**Pàgina d’Índex amb CSS GRID**

Un dels requeriments de la aplicació ha estat la creació d’una pàgina d’índex amb GRID i per fer-ho ens hem basat i adaptat l’exemple facilitat a la documentació <https://chenhuijing.com/blog/how-i-design-with-css-grid/>

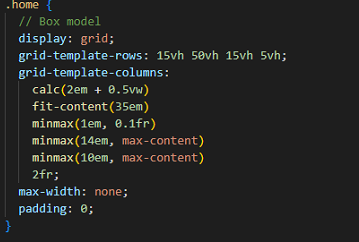


En primer lloc, tal i com indica la documentació, s’ha adaptat els diferents elements, però s’han renombrat i adaptat els elements a classes, per exemple la classe del exemple “<class="about">” s’ha renombrat a <class=”home\_\_about”> o el títol, que només utilitza <h1>a <class=”home\_\_title”> quedant de la següent manera

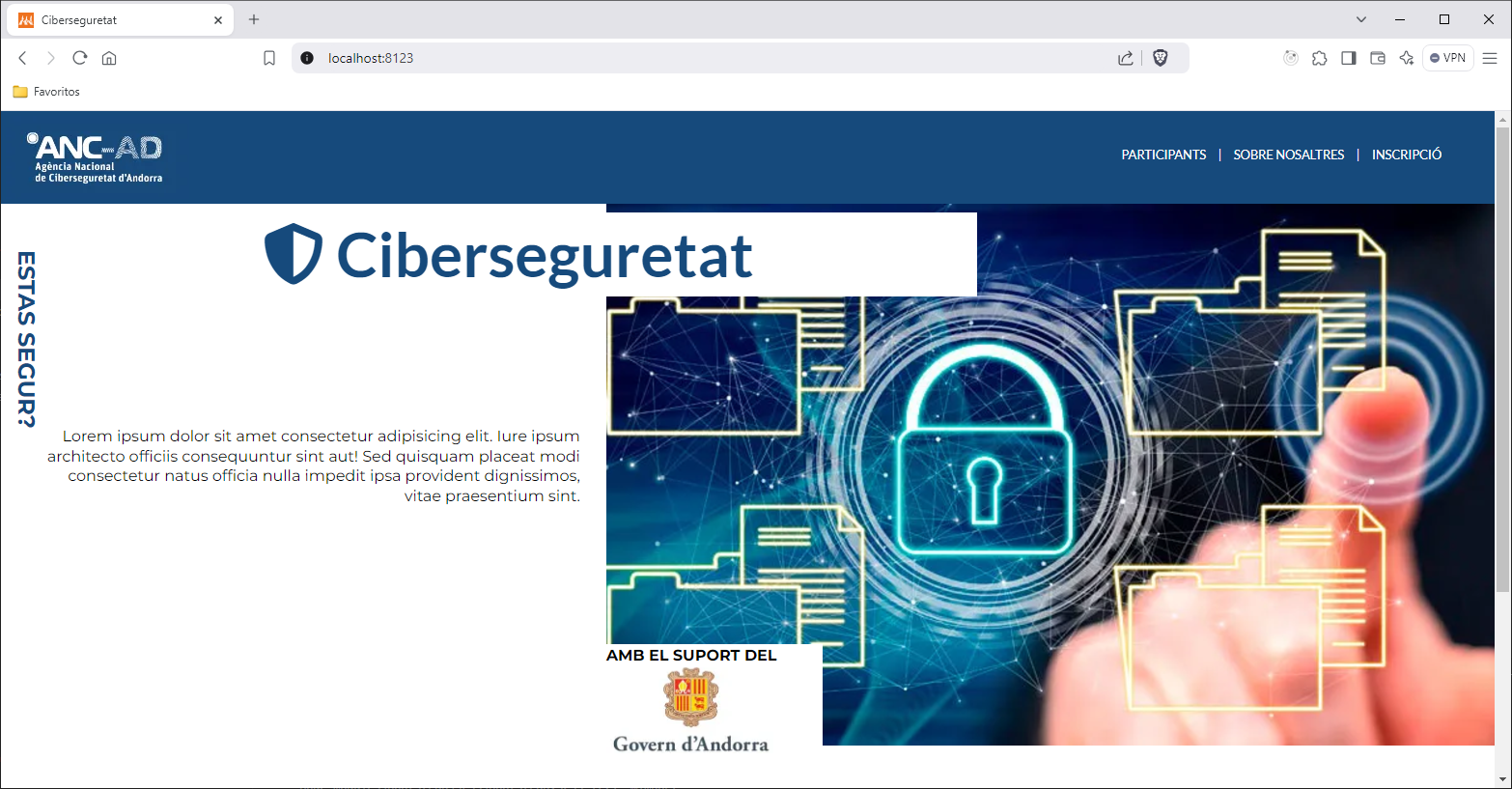


Això ens permet controlar mitjançant classes el contingut en dispositiu mòbil o no compatible amb GRID.

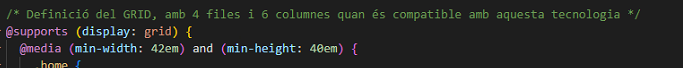
El següent pas ha estat adaptar-ho al GRID, i per això, continuant basant-nos el l’exemple facilitat, s’adapta per tal que tingui 4 files adaptables amb VH i 6 columnes també adaptables,



Llavors, mitjançant amb bastanta prova i error amb el CSS i el codi HTML, s’aconsegueix crear la portada amb GRID i els seus elements.



També, definim dos aspectes importants en aquesta línia, i és que s’utilitzi el GRID en cas que sigui compatible i mitjançant un @media quan tingui una amplada i alçada suficients.



Per tant, per fer una prova, recuperem un antic dispositiu amb Windows XP i una versió antiga del Firefox incompatible amb GRID per observar com es veuria.





Amb lo que podrem observar que, tot i no utilitza el GRID, es veu correctament.

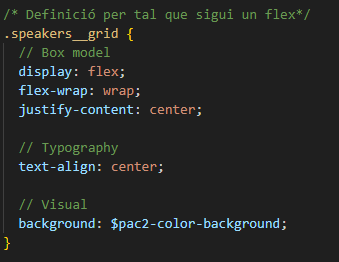
**Pàgina de “Speakers” amb CSS GRID**

Un altre dels requeriments de la aplicació és la que ha d'incloure una retícula amb informació sobre els participants maquetada amb flex.

Per fer-ho principalment ens hem basat en la documentació facilitada per MDN sobre com fer una maquetació amb flex.  
<https://developer.mozilla.org/en-US/docs/Web/CSS/flex-wrap>

A part també buscant exemples sobre com fer una estructura de flex i “cards”  
<https://codepen.io/mcraiganthony/pen/NxGxqm>

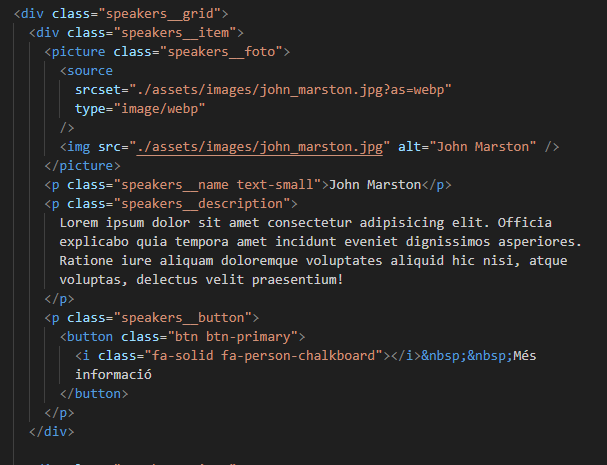
Per tant, un cop definit com ho fem, es realitza la implementació, primerament amb el CSS



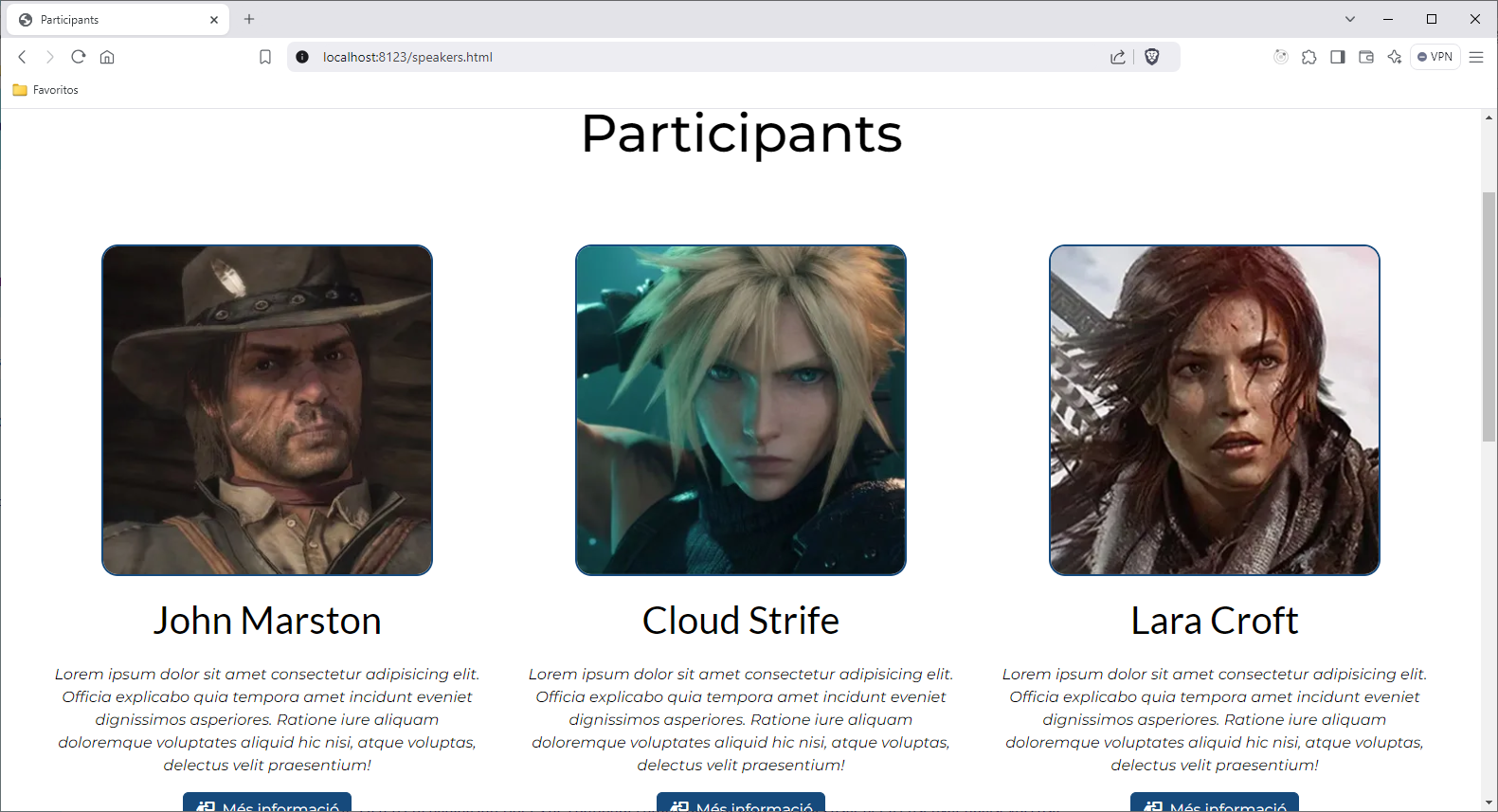
I després amb els items que contindrà el flex.



I que serà implementat en el codi HTML



I, un cop implementat, també ajustant les coses mitjançant prova i error, el resultat és el següent:



I també és pot comprovar que, amb la versió mòbil, es compleix el disseny responsive.

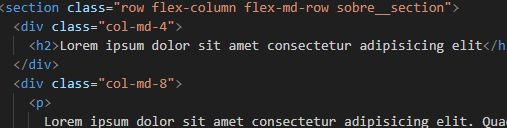


**Pàgina de l’article “Sobre Nosaltres”**

El següent requeriment ha estat la creació d’una pàgina en forma d’article amb diferents elements HTML, utilització de pseudoclases i semblant al wireframe facilitat.

A l’hora de fer la pàgina responsive, s’ha escollit utilitzar Bootstrap pels motius especificats a l’apartat 2 de la documentació, utilitzant *row* i *col-xx*.

Per exemple aquí es veu la utilització de una *row* amb flex i després dos columnes de diferent grandària que després, a l’hora de fer la gestió responsive, es farà automàticament.



*Elements HTML i CSS especials*

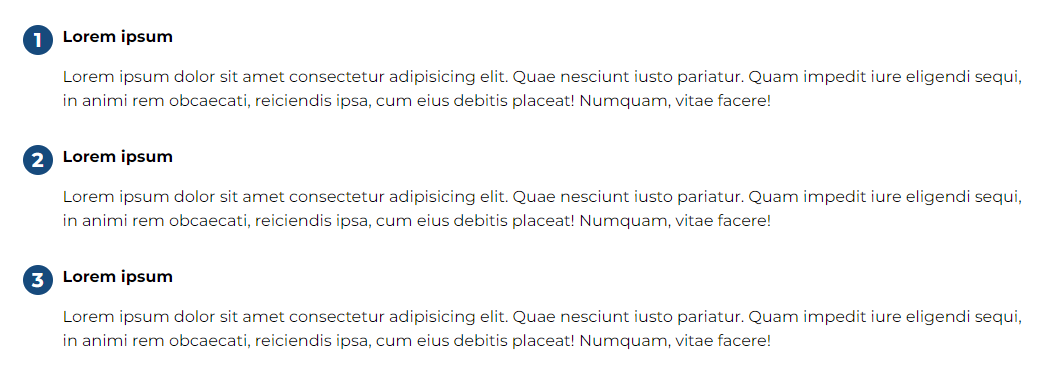
En primer lloc, parlarem una mica dels elements especials que s’han utilitzat.

El primer d’ells és, seguint la especificació del wireframe, una llista amb números estilitzats, que es va trobar un exemple a <https://blog.logrocket.com/styling-numbered-lists-with-css-counters/>

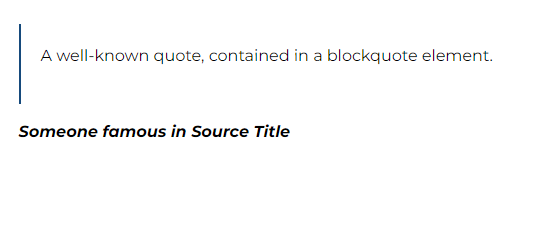
Es va implementar amb l’ús d’un comptador que s’anava incrementant i després donar-li un aspecte visual amb un cercle.



Amb el resultat següent que compleix el wireframe



Un altre element ha estat un blockquote al qual se li ha aplicat una mica d’estil per CSS amb elements apresos de la assignatura realitzada HTML i CSS.



De la mateixa manera s’han aplicat diferents estils CSS als altres elements, com per exemple afegir una línia al final, conversió d’imatges a webp, personalització d’elements, inclusió d’un hero, etc. molts dels quals han estat utilitzats en altres PAC d’altres assignatures anteriors.

*Utilització de pseudoclases*

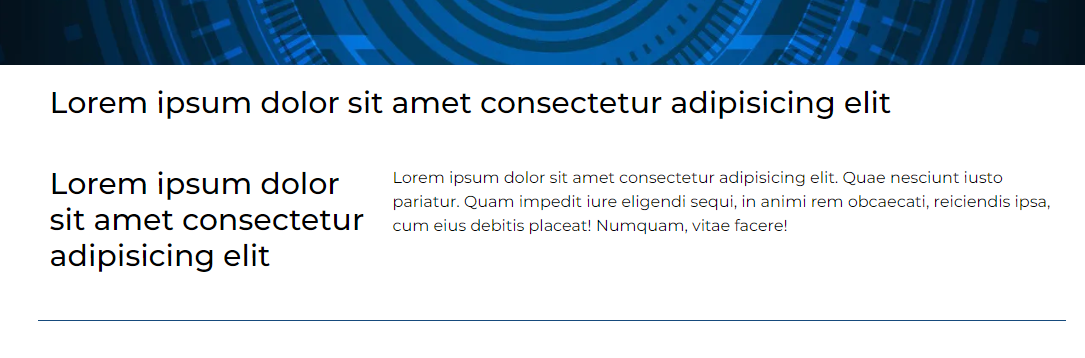
Un altre dels requeriments de la pàgina ha estat la utilització de pseudoclases.

En primer lloc veurem com hem utilitzat la pseudoclase “:is()”

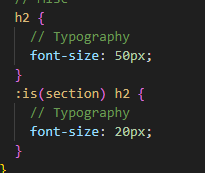
Si observem el codi CSS, veiem que hem definit que els “h2” tinguin una grandària de lletra de 30px.

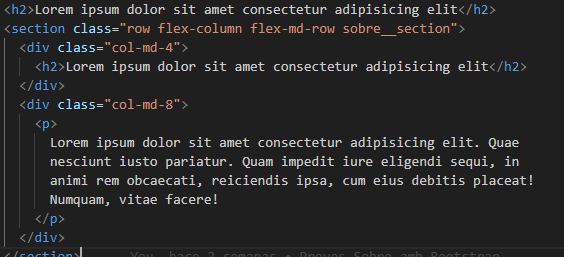


Amb el següent resultat:



Mitjançant “is()” podem fer que si aquest h2 està dintre de un “section”, aquest se li apliqui un codi CSS específic:



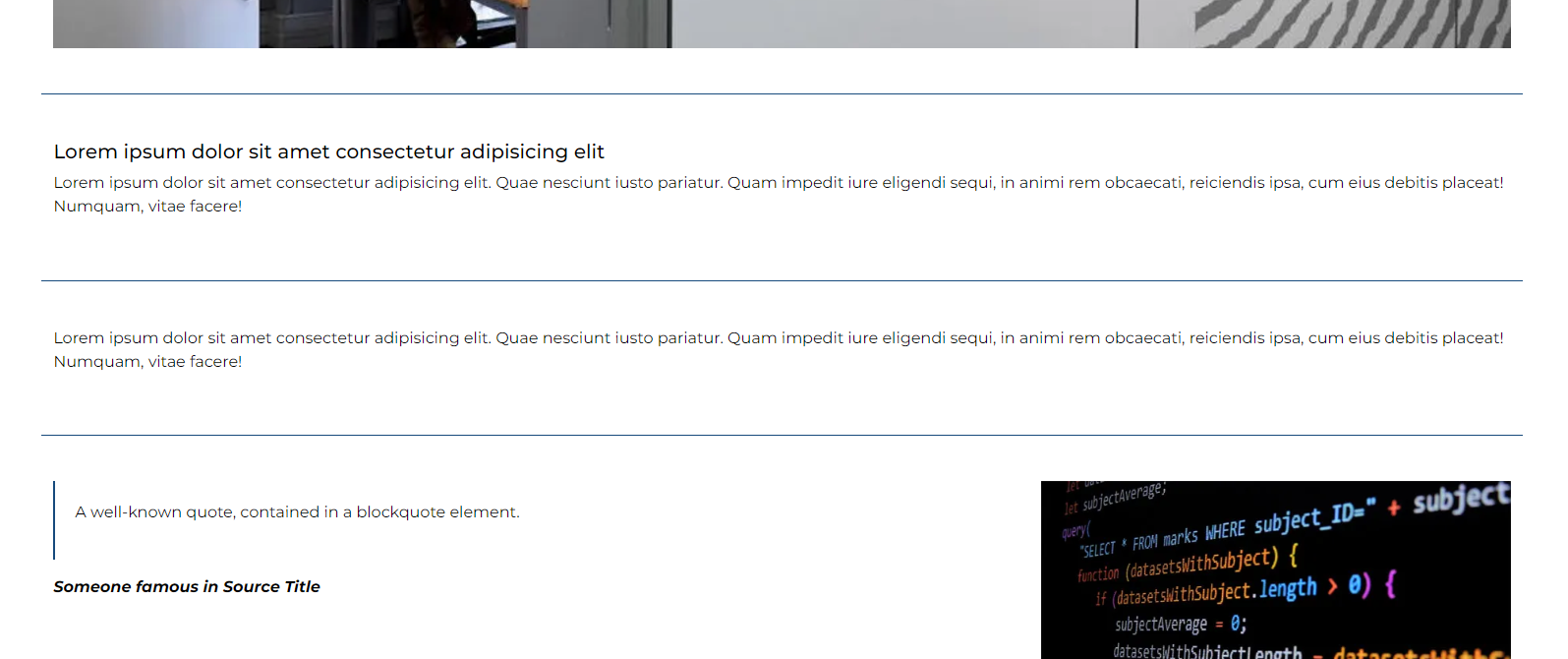


Amb aquest codi fem que si el h2 està a dintre de una section, la grandària de la lletra sigui diferent més petita, mentre que la que està fora del section, te la grandària definida.

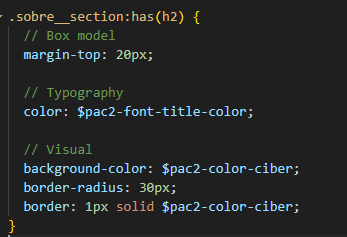


La següent classe utilitzada és “:has()”

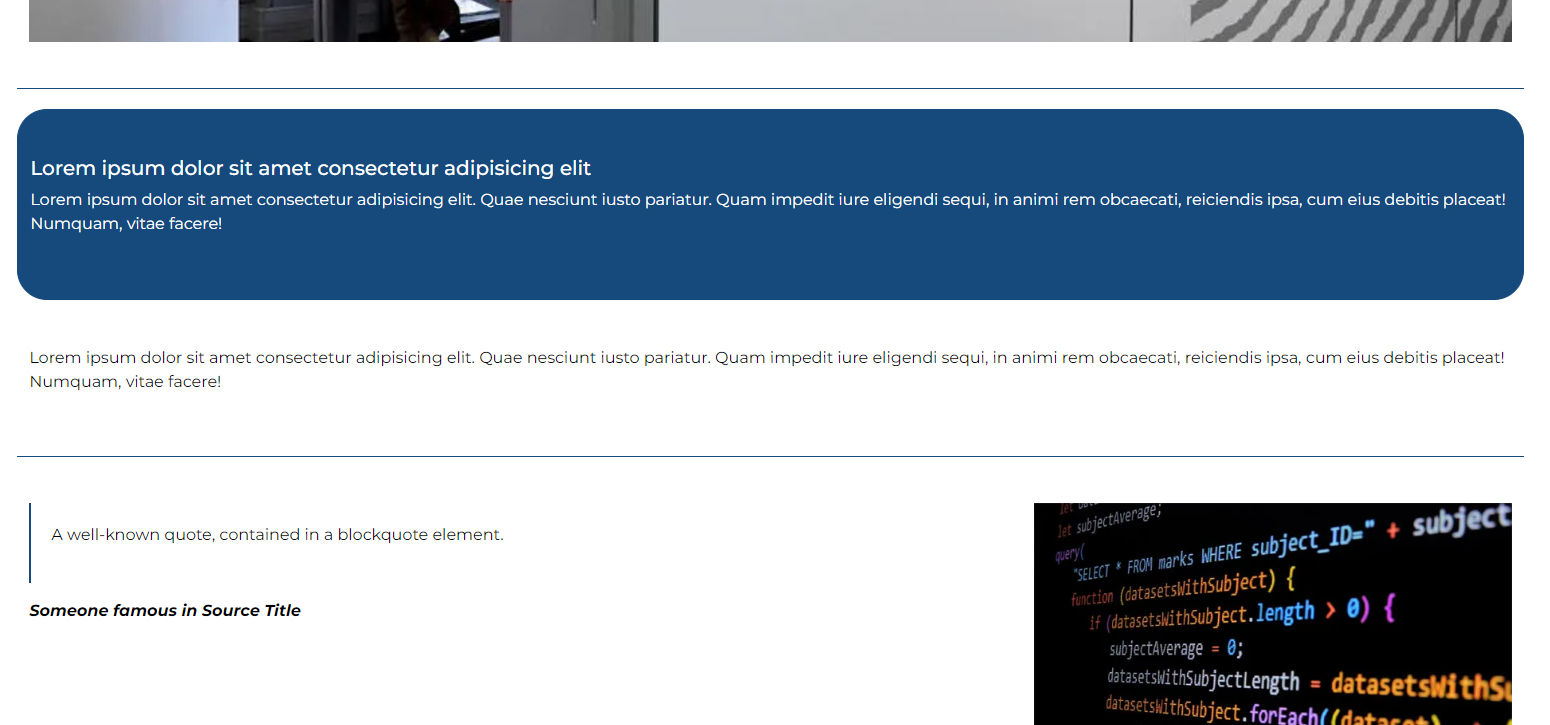
Amb el codi actual es pot veure que hi ha algunes seccions que tenen un element h2.



Mitjançant “:has()”, podem fer que si te aquest element, se li apliqui algun estil.

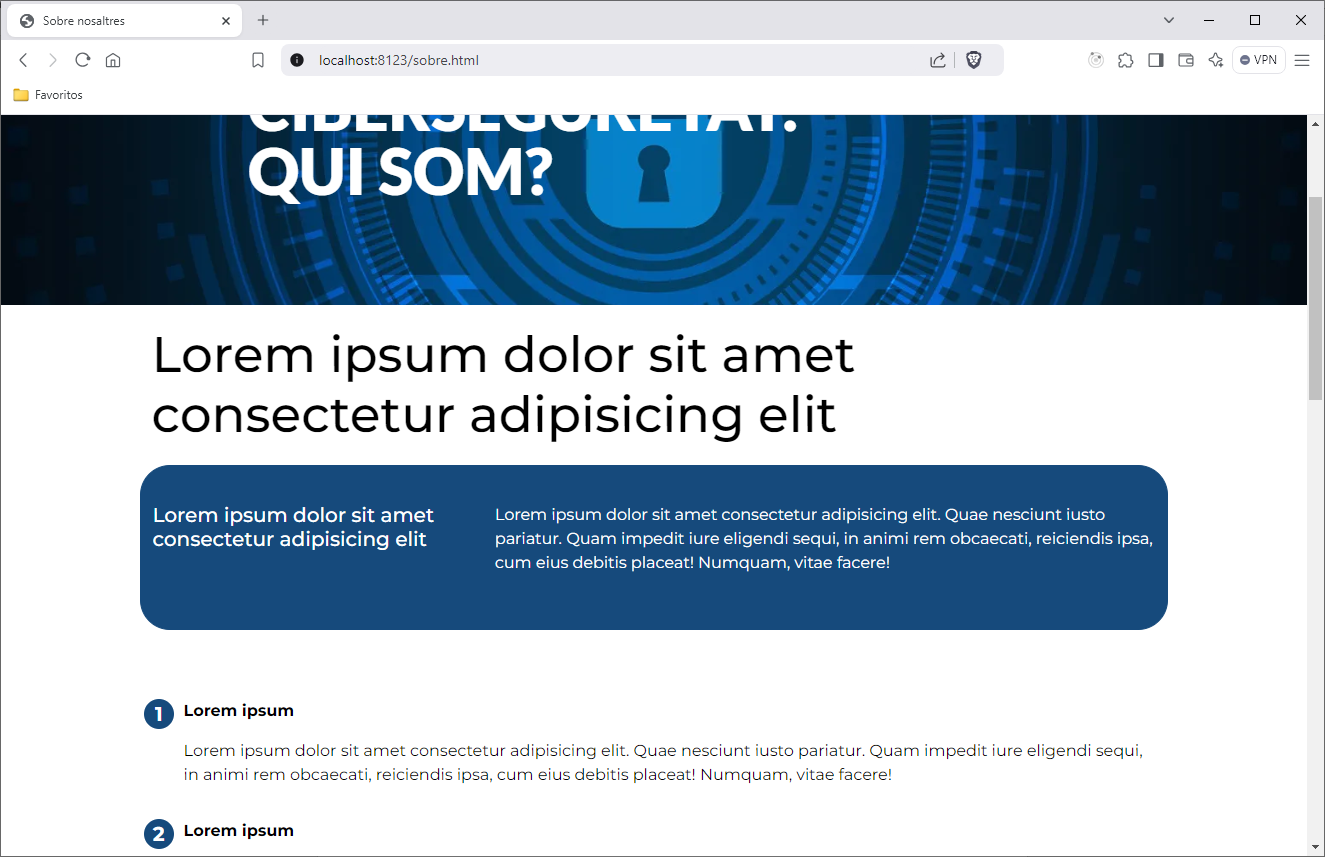


Aplicant aquesta modificació, podem veure que els llocs que tenen la classe “.sobre\_\_section” si tenen el element h2, se li apliqui un estil especial.

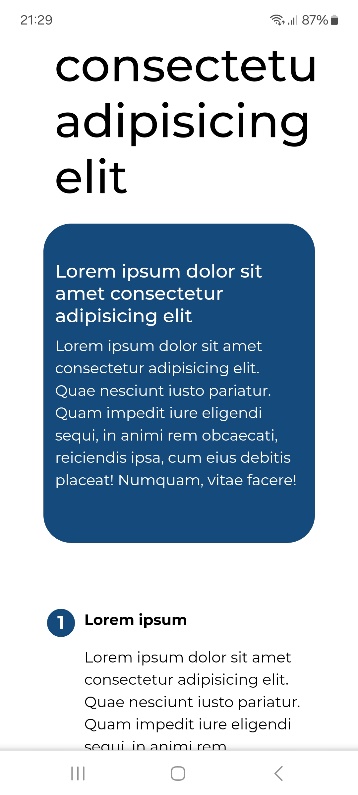


*Resultat final*

Tal i com s’ha dit anteriorment, amb la utilització del bootstrap s’aconsegueix fer una gestió del responsive amb el següent resultat.



I per la versió mòbil



**Pàgina lliure (inscripció a cursos)**

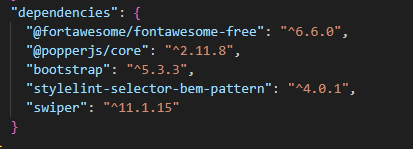
Per la darrera pàgina el contingut és completament lliure, però s’ha d’utilitzar les noves unitats i colors.

En aquesta pàgina s’ha decidit fer una pàgina amb un formulari d’inscripció als cursos, i per fer-ho s’utilitzarà la dependència externa “*swiper*” i els formularis Netlify.

A l’igual que la pàgina anterior, també s’utilitzarà el Bootstrap per fer la gestió del responsive.

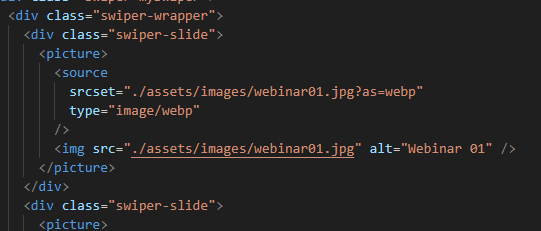
*Dependència externa “swiper”*

Un cop s’ha instal·lat i configurat al Boilerplate seguint les instruccions definides a <https://swiperjs.com/get-started#install-from-npm> es procedeix a la recerca del carrusel d’imatges que millor es pugui adaptar.



Finalment s’escull el “Infinte Loop” ( <https://swiperjs.com/demos#infinite-loop> ), ja que proporciona elements de navegació, tant seqüencial com accessos directes, amb un avanç automàtic.

Per tant, s’estudia el codi i s’adapta al projecte, afegint tant al CSS com al HTML els elements necessaris per fer-lo funcionar.







I el resultat final és el següent:



*Formulari Netlify*