PAC 3: M4.258 – Eines HTML i CSS II



Enric Miralles i Selvi

08/01/2024

Link web: <https://pac3eineshtml2emirallesse.netlify.app/>

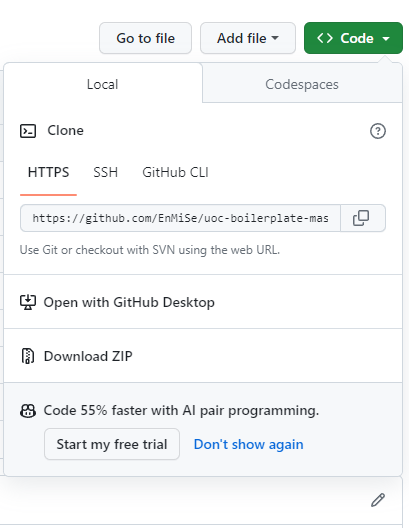
Link repositori: <https://github.com/EnMiSe/EinesHTMLiCSS_II_PAC_3_Eines_eMiralles>

Link web Pac anterior (publicación fallida): <https://pac2eineshtml2emirallesse.netlify.app/>

Link repositori Pac anterior: <https://github.com/EnMiSe/uoc-boilerplate-master_emiralles_PAC2>

1. **Procés de desenvolupament**
   1. **Clonació repositori**

Per començar amb la pràctica s’ha clonat el repositori de Github amb el boilerplate de la UOC al pc mitjançant la opció que ofereix GitHub Web. En aquest cas s’ha triat la opció “Open with Github Desktop” que hi ha a la secció “<> Code” de la pròpia pàgina de Github.



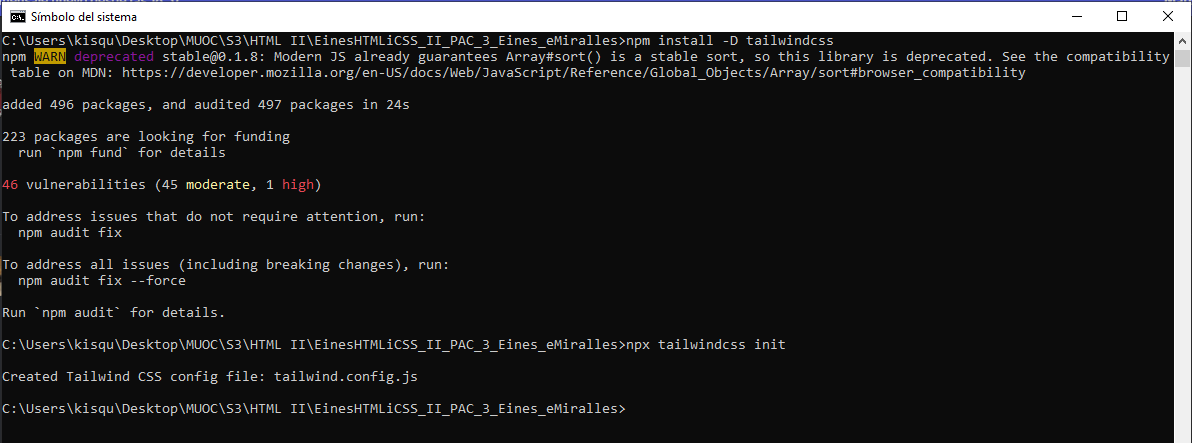
 A continuació, mitjançant Github Desktop s’ha generat un nou repositori localment. S’ha eliminat els fitxers que no són necessaris per a aquesta pràctica i s’han afegits els que han servit de base creats a l’anterior PAC.

Amb un commit inicial, s’ha pujat a Github els fitxers i components del nou projecte.

* 1. **Dependències**

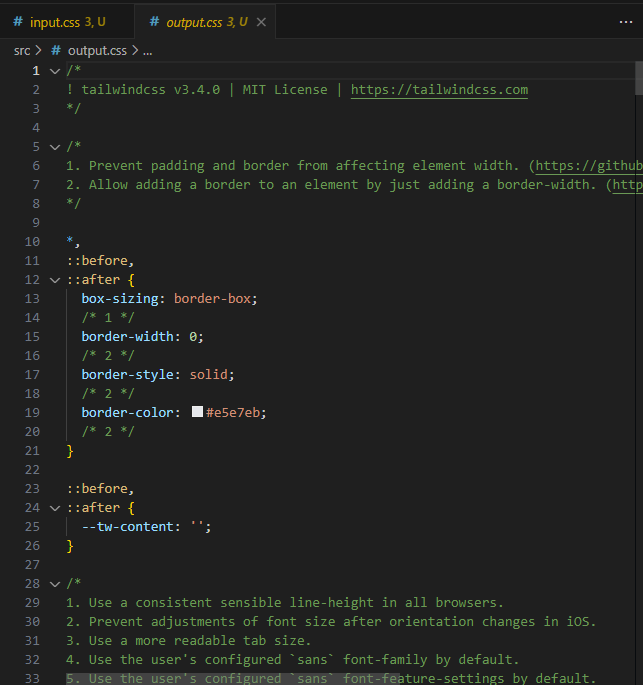
S’instal·la node per mitjà de la comanda ‘npm install’ amb la consola de Windows, un cop situats a la ruta amb la carpeta principal del repositori. Per a la compilació i l’execució del lloc web en un servidor de desenvolupament es fan servir les comandes ‘npm run build’ i ‘npm run dev’. La primera compila el codi del repositori mentre que la segona, executa el script situat al fitxer ‘package.json’ que crearà/modificarà carpetes com “dist” i “parcel” a mesura que s’avanci en el projecte.

Paral·lelament, també s’ha instal·lat TailwindCss per tal de elaborar els estils referents a dos pàgines web del projecte. La imatge que es mostra a continuació es una finestra de la línia de comandes on s'ha realitzat la instal·lació i configuració de Tailwind. Els passos realitzats són els següents:



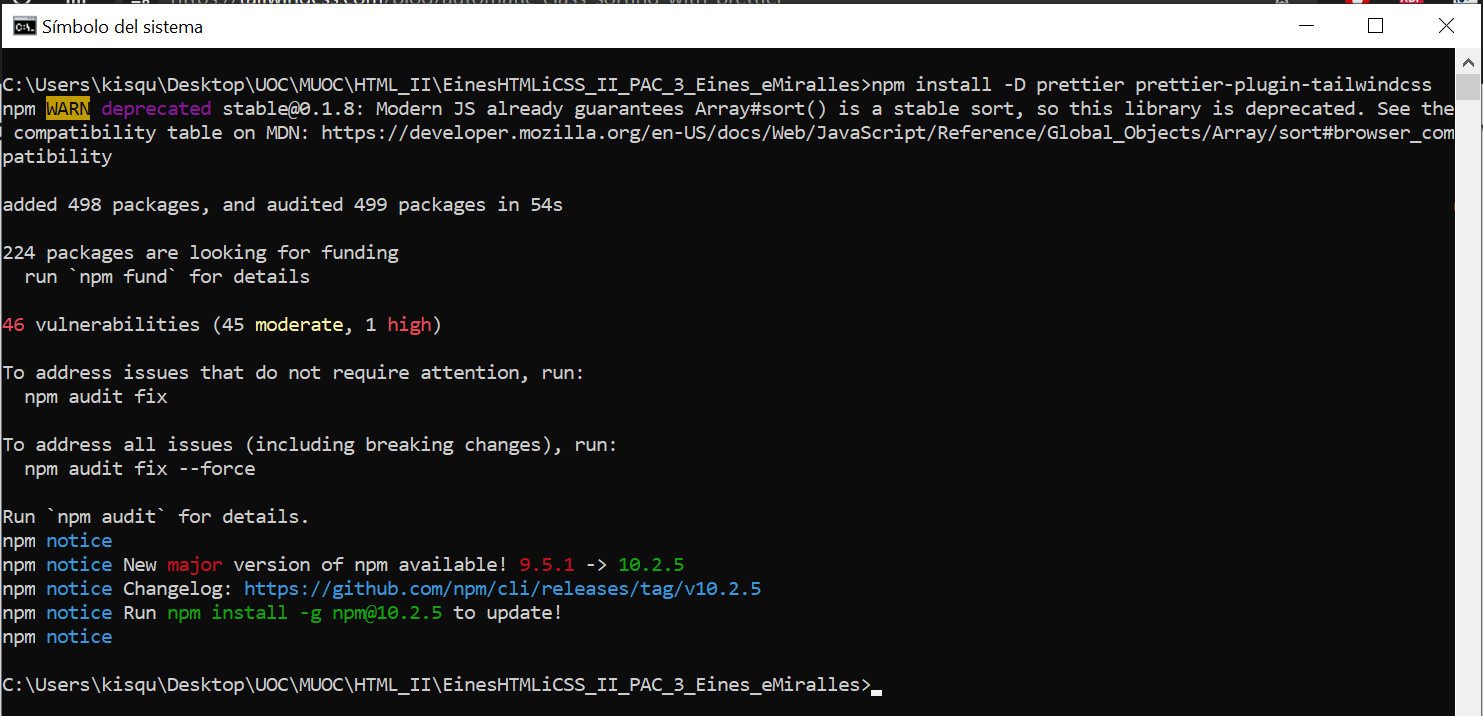
Primer s'ha executat la comanda ‘npm install -D tailwindcss’ per instal·lar la dependència en el projecte. A continuació, s’executa la comanda `npx tailwindcss init` que crea un fitxer de configuració ‘tailwind.config.js’. Aquest s’utilitza per personalitzar la configuració de Tailwind per a les necessitats que requereix el projecte.

També s’ha emprat ‘npx tailwindcss -i ./src/input.css -o ./src/output.css –watch’. Aquesta comanda indica a Tailwind CSS que es processi l'arxiu CSS d'entrada `input.css` i que generi l'arxiu CSS de sortida `output.css`. L'opció “—watch” s'utilitza perquè Tailwind segueixi observant els arxius d'entrada per a canvis, i reconstrueixi l'arxiu de sortida automàticament. Aquest correspn al fitxer ‘output.css’ on es guarden i es generen els estils:

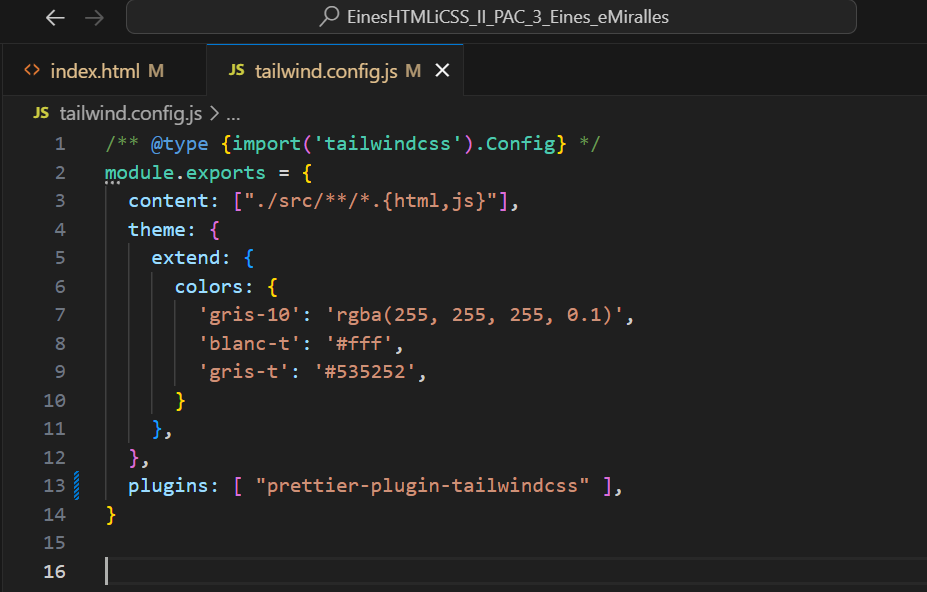


Prettier:

Amb la finalitat de mantenir una estructura ordenada de les classes utilitzades per atomic i Tailwind, s’ha instal·lat també un pluggin de prettier.



Posteriorment s’afegeix el plugin a l’arxiu de configuració de Tailwind:



Cada cop que es dona format al document, prettier reorganitzarà el codi html de manera que sigui llegible i entenedora per a l’usuari per tal de poder treballar seguint les directrius de atomic i Tailwind.

* 1. **Contingut web**

La idea principal es refer dues pàgines d’un lloc web promocional d’un club de tenis creat per un grup d’amics. L’objectiu es treballar amb la dependència Tailwind i fer que quedin de manera similar a les pàgines creades a la entrega anterior. Convinat amb l’ús d’atomic, es generen classes descriptives on s’apliquen els estils per mitjà de tailwind. Es una manera senzilla i ordenada de d’organitzar estils reutilitzables per als diferents components generats a partir del codi html. Les dues pàgines corresponen a la de “Artcile” i “Participants”.

En general, totes les pàgines contenen un “header” i un “footer” compartit. Al header s'assegura que la pàgina sigui responsive amb la metaetiqueta viewport, que controla com es visualitza el contingut en dispositius mòbils. El “link” a la fulla d'estils output.css indica que els estils CSS estan situats al fitxer que tailwind va actualitzant segons les classes que es va afegint al codi html.

El cos del document, indicat per l'element <body>, té un fons negre i text blanc aplicant “atomic” a les utilitats de Tailwind bg-black i text-white respectivament. La tipografia general de la pàgina està establerta com a sans-serif amb la classe font-sans.

A la secció del “header” se situa la icona del club de tenis, a més del nom de la entitat i un menú de navegació per a navegar per les diferents pàgines del lloc web. Dins de l'encapçalament, la imatge del logotip i el títol es posen al costat amb flex items-center. La mida de la imatge s'ajusta en funció de la mida de la pantalla amb w-1/6 per defecte i md:w-16 per pantalles mitjanes a més grans. El títol té un espaiament adicional a l'esquerra amb pl-2.

El menú de navegació usa pr-5 per afegir padding a la dreta. Els enllaços de navegació estan dins d'una llista horitzontal que gestiona l'espaiat entre elements amb space-x-4. Els enllaços canvien de color quan es passa per sobre amb hover:text-blue-800 i hover:text-red-700, seguint el estil vermell i blau que es troba al logotip.

Mirar de posar un mínim de 2 components amb posthtml.

**Pàgina principal (Index): Article**

Pàgina centrada en un article. El contingut principal, <main>, està emmarcat amb un ampli espai superior per separar-lo de l'encapçalament. Conté un <article> que serveix de contenidor per al contingut textual, amb diversos paràgrafs <p> que descriuen l'esdeveniment, el club i altres detalls relacionats amb el tennis.

També inclou imatges relacionades amb el tennis i un mapa, així com un <blockquote> per destacar una cita sobre el tennis. Al final de l'article, hi ha una llista amb ítems <li> que proporcionen informació addicional sobre el club i els esdeveniments, com dates d'inscripció i disponibilitat de classes.

Pel que fa als estils, s'utilitzen classes de Tailwind per estilitzar el contingut de manera que siguin responsives i visualment atractives.

Dins de l'encapçalament, es troba una estructura de Flexbox que canvia entre columna i fila depenent de la mida de la pantalla. Això s'aconsegueix amb les classes flex, flex-col, items-center, md:flex-row, i justify-between. El padding superior i inferior s'ajusta amb les classes pt-32 i pb-8. El color de fons, bg-gris-10, defineix un to de gris específic, que correspon una classe personalitzada dins del fitxer tailwind.config.ts.

El contingut principal està emmarcat per un ample espai superior amb pt-44. Dins de l'article, el contingut està centrat tant vertical com horitzontalment amb flex, flex-col, mx-auto, items-center, i justify-center. La grandària de la font i l'ajust del text es controla amb text-lg i text-justify, respectivament, mentre que el padding horitzontal es gestiona amb px-24.

Per tal de minimitzar el codi resultant, s’ha reduït la classe corresponent a <article> i <header> en duess classes anomenades “article” i “header” respectivament, on s’hi apliquen tots els paràmetres mencionats anteriorment fent ús de @apply. Aquests estils es poden trobar dins del fitxer apply.css, dins de la secció @layer components.

Les imatges dins de l'article tenen mides relatives definides amb w-3/6 i w-2/6, ajustant-se a la mida del contenidor. Els paràgrafs i les seccions dins de l'article tenen espaiat consistent amb classes com pb-3, pt-4, i pb-4.

Al final de l'article, es troba una llista que destaca informació clau amb un fons gris, personalitzat amb bg-gris-t.

* **Pàgina interior: Participants**

Pàgina centrada en mostrar un llistat dels participants del torneig de tenis. Conté dues seccions <section> que engloben articles <article> amb informació sobre diversos participants. Cada article inclou una imatge del participant, un títol amb el seu nom i un paràgraf de descripció o estadístiques. Aquests articles estan organitzats en dues seccions diferents dins de l'element `<main>`, que aporta un padding a tots els costats.

Cada secció `<section>` utilitza Flexbox per a organitzar els seus fills, els articles `<article>`, que s'ajusten i s'envolten per adaptar-se a l'amplada de la pantalla. Els articles estan centrats dins de la secció gràcies a les classes `justify-center` i separats entre si per un gap definit per `gap-4`. A més, la primera secció té un marge inferior `mb-10`, proporcionant separació de qualsevol contingut que pugui venir després. Per als articles s’ha utilitzat la classe “targeta participant”, creades a partir de @apply com s’ha esmentat anteriorment.

Dins de cada article, hi ha un contenidor `<div>` amb un fons gris `bg-gris-10`, ombres `shadow-lg`, vores arrodonides `rounded` i ocultació del contingut desbordat `overflow-hidden`. Aquest contenidor inclou una imatge que ocupa l'espai de la seva contenidora, mantenint les proporcions amb `object-cover`.

A continuació, cada contenidor de l'article conté un altre `<div>` amb un padding en totes direccions `p-4`. Dins d'aquest, hi ha un encapçalament `<h3>` que està centrat dins del seu contenidor pare amb les classes `justify-center` i `items-center`, utilitzant una font semi-negreta `font-semibold` i un tamany de font més gran `text-lg`. Després hi ha un paràgraf `<p>` amb el text en blanc `text-white` que proporciona una descripció o estadístiques del participant.

Al final del `<main>`, hi ha un contenidor `<div>` que centra els seus elements en l'eix horitzontal i vertical amb `justify-center` i `items-center`, respectivament, i té un marge superior `mt-10`. Dins d'aquest, hi ha un botó que, quan s'hi fa clic, ha d'activar un menú desplegable. El botó està estilitzat amb un fons blau fosc `bg-blue-800`, text en blanc `text-white`, vores arrodonides `rounded`, i classes de focus per millorar l'accessibilitat `focus:outline-none` i `focus:ring`.

El menú desplegable associat amb el botó està inicialment ocult `hidden` i es posiciona absolutament `absolute` amb respecte al seu contenidor pare, amb un marge superior `mt-2`, una amplada fixa `w-40`, vores arrodonides `rounded-md`, ombra `shadow-lg`, fons blanc `bg-white`, i un anell decoratiu `ring-1 ring-black ring-opacity-5`. Conté enllaços `<a>` que permeten l'ordenació dels participants segons diversos criteris, i cada enllaç té un padding `px-4 py-2`, un tamany de font petit `text-sm`, un color de font gris `text-gray-700`, i un canvi de color de fons en interacció `hover:bg-gray-100`.

El codi JavaScript al final del `<main>` defineix la funció `toggleDropdown()` que s'encarrega de commutar la visibilitat del menú desplegable en resposta als clics de l'usuari, mostrant-lo si està ocult i ocultant-lo si està visible. A més, hi ha un gestor d'esdeveniments que escolta els clics en qualsevol lloc de la finestra i tanca qualsevol menú desplegable obert si el clic es fa fora d'un botó de desplegable.

Aquesta estructura completa ofereix una interfície d'usuari interactiva i visualment ordenada que utilitza Tailwind CSS per aplicar estils de manera eficient i mantenible.

* 1. **Entorn de desenvolupament**

Per a la realització d’aquest projecte s’ha utilitzat VSCode com a editor de text, a més de node amb els plugins Prettier, PostCss i Tailwind com a eines d’ajuda per a l’escriptura del codi i les seves respectives dependències.

Amb el terminal de comandes, s’ha generat i compilat el codi per mitjà de npm run build i npm run dev. D’aquesta manera, des d’un servidor local s’ha pogut anat veient i retocant el codi per tal de complir els objectius estipulats al enunciat de la PAC. Paral·lelament, amb la comanda anteriorment esmentada “npx tailwindcss -i ./src/input.css -o ./src/output.css –watch” s’han anat generant els canvis al fitxer output.css per a configurar els estils de la pàgina web.

Els components afegits al projecte s’han generat amb PostHTML (explicar)

Per anar guardant el projecte s’ha fet servir GitHub web per a la gestió del repositori i GitHub Desktop per a poder anar treballant i gestionant els avenços des de diferents dispositius de treball.

* 1. **Compilació**

Per a compilar el codi del projecte s’executa la comanda “npm run dev” a la consola de windows. Aquesta comanda inicia un procés d’execució del fitxer Package.json.

En general, npm run dev està destinada a iniciar un servidor de desenvolupament local o a executar l'aplicació en un mode que faciliti el desenvolupament. Això pot incloure característiques com la recàrrega en calent (hot-reloading), on els canvis en el codi font són detectats automàticament i l'aplicació es recarrega o recompila sense necessitat de reiniciar manualment el servidor.

D’aquesta manera, es llança un Servidor Local que permet al desenvolupador veure els canvis en temps real mentre treballen en l'aplicació.

Per altra banda també es compila l'aplicació en un mode optimizat per a depuració i desenvolupament i s’executen tasques automatitzades, com per exemple la minificació de codi.

1. **Metodologia i guia d’estils**
   1. **Metodologia aplicada**

Als fitxers “output.css” i “apply.css” s’han implementat tots els estils referents a colors, disposició dels elements a la web, així com els estils de les imatges generats a partir de la modificació del fitxer “html” de les dues pàgines. També, al fitxer tailwind.config.js s’ha implementat una sèrie de colors que no venen per defecte per tal de seguir amb els estils creats en l’activitat anterior.

S’ha seguit la metodologia “Mobile-first” per tal d’adaptar tant tots els elements de les pàgines web, com el títol i menú de navegació situat al “header”.

Responent a les preguntes plantejades en l’enunciat, les diferències entre l'enfocament CSS semàntic i l'enfocament d'utilitats (com Atomic CSS o Tailwind CSS, utilitzats en aquest projecte) són força significatives.

L'enfocament CSS semàntic utilitza noms de classes basats en la funció o el contingut i agrupa múltiples propietats d'estil, resultant en menys classes per element i facilitant la personalització. No obstant això, això pot conduir a estils duplicats i complicar el manteniment en projectes grans. D'altra banda, el CSS d'utilitats es basa en classes atòmiques que representen propietats individuals, incrementant la reusabilitat i promocionant la coherència de disseny, tot i que pot fer que l'HTML sigui menys llegible i requereixi una major comprensió de com combinar aquestes classes.

L'ús de llibreries de components contrasta amb les llibreries d'utilitats en que les primeres ofereixen components predefinits amb menys flexibilitat de personalització, mentre que les segones permeten una construcció més granular i directa dels components. La selecció de classes i components a extreure es basa en factors com la reusabilitat, la complexitat i la necessitat de coherència en el disseny.

Quines classes i components has decidit extreure i per què?

* 1. **Disseny i Estil aplicat**

Per al disseny del lloc web s’ha seguit amb l’estil de les pàgines originals, un fons negre que ressalti l’interior on es mostra el contingut rellevant. El color de la font (blanc), s’ha escollit per a crear un contrast amb el fons, seguint les tonalitats anteriorment mencionades. Els colors grisos utilitzats, tant al header com de background color a la pàgina de l’article, serveixen per a crear un contrast entre el fons negre i un element que es vol ressaltar, sense perdre el fil del color emprat pel fons de la web.

En el cas de la pàgina participants s’ha optat per mostrar els competidors registrats en un format fitxa. El disseny implementat per a aquesta secció està fet de manera responsiva, adaptant els formats de resolució de pantalla per a mòbil primer. Per exemple, en formats inferiors a 768px, el menú de navegació es disposa en columna per sota del títol. També les imatges i caixes que conforment el llistat de participants, es disposen en files de 4, 2 o verticalment segons l’espai disponible de l’ample de pantalla.

El contenidor principal <main> té un padding aplicat a tots els costats amb la classe p-10. Aquesta classe crea un espai uniforme al voltant del contingut dins del <main>.

Dins d'aquest contenidor principal, hi ha dues seccions <section> que utilitzen Flexbox per alinear els <article>. Utilitzen les classes flex, flex-wrap, i justify-center per crear una disposició flexible que centra els articles horitzontalment i permet que els elements s'envoltin en línies múltiples quan s'esgota l'espai horitzontal. La classe gap-4 afegeix un espai de 16 píxels entre els elements de la graella, i mb-10 afegeix un marge inferior a la primera secció.

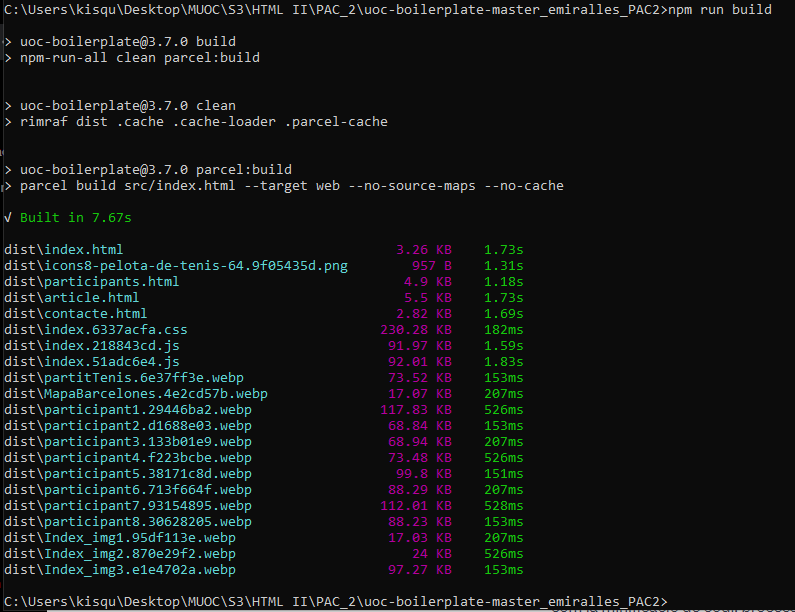
Cada <article> representa un participant. La classe p-2 afegeix un padding de 8 píxels, flex-auto permet que l'article es redimensioni dins de la disposició flex, w-1/2, que estableix l'amplada de l'article a la meitat de l'amplada del contenidor per a dispositius més petits. La etiqueta md:w-1/6 canvia aquesta amplada a 1/6 en pantalles de mida mitjana cap amunt. La classe border-4 afegeix un borde de 4 píxels d'ample, i border-gris-10 que pertany al color personalitzat.

Dins de cada article, hi ha un contenidor <div> amb un fons gris, una ombra gran, vores arrodonides i un overflow ocult, aconseguint això amb les classes bg-gris-10, shadow-lg, rounded, i overflow-hidden. Aquest contenidor envolta una imatge <img> amb la classe object-cover, la qual assegura que la imatge cobreixi l'espai disponible sense deformar-se. A continuació, hi ha un altre contenidor <div> que conté un encapçalament <h3> i un paràgraf <p>, amb estils de text centrats i blancs.

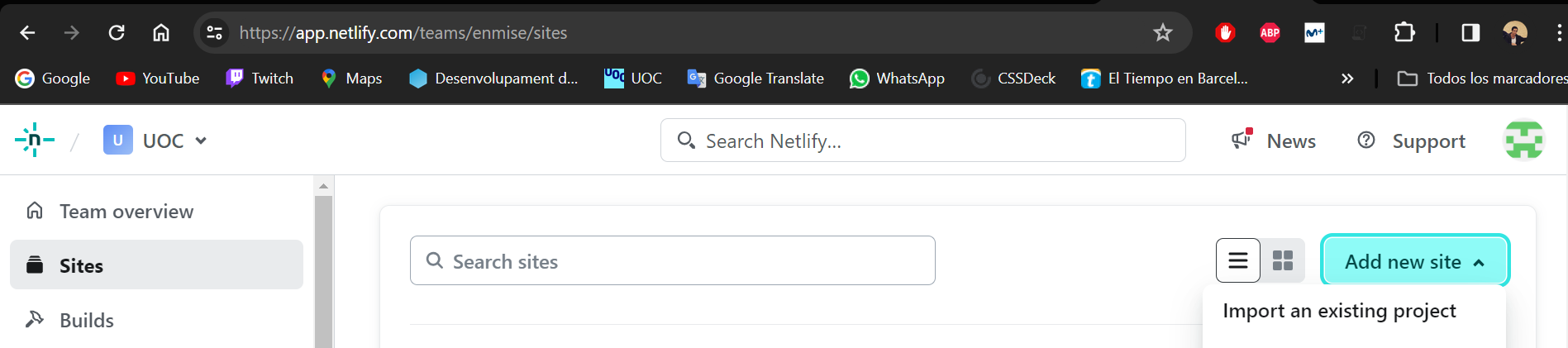
Finalment, hi ha un botó <button> que està destinat a mostrar un menú desplegable quan s'hi fa click. Està estilitzat amb un fons blau fosc seguint els colors del logotip, text blanc i vores arrodonides. El menú desplegable <div> que conté els enllaços <a> està inicialment ocult amb la classe hidden i es posiciona sobre altres continguts amb absolute. Aquest menú desplegable també té un estil amb vores arrodonides, ombra gran, i un anell al voltant per a destacar-se del contingut de fons.

1. **Deployment a Netlify**

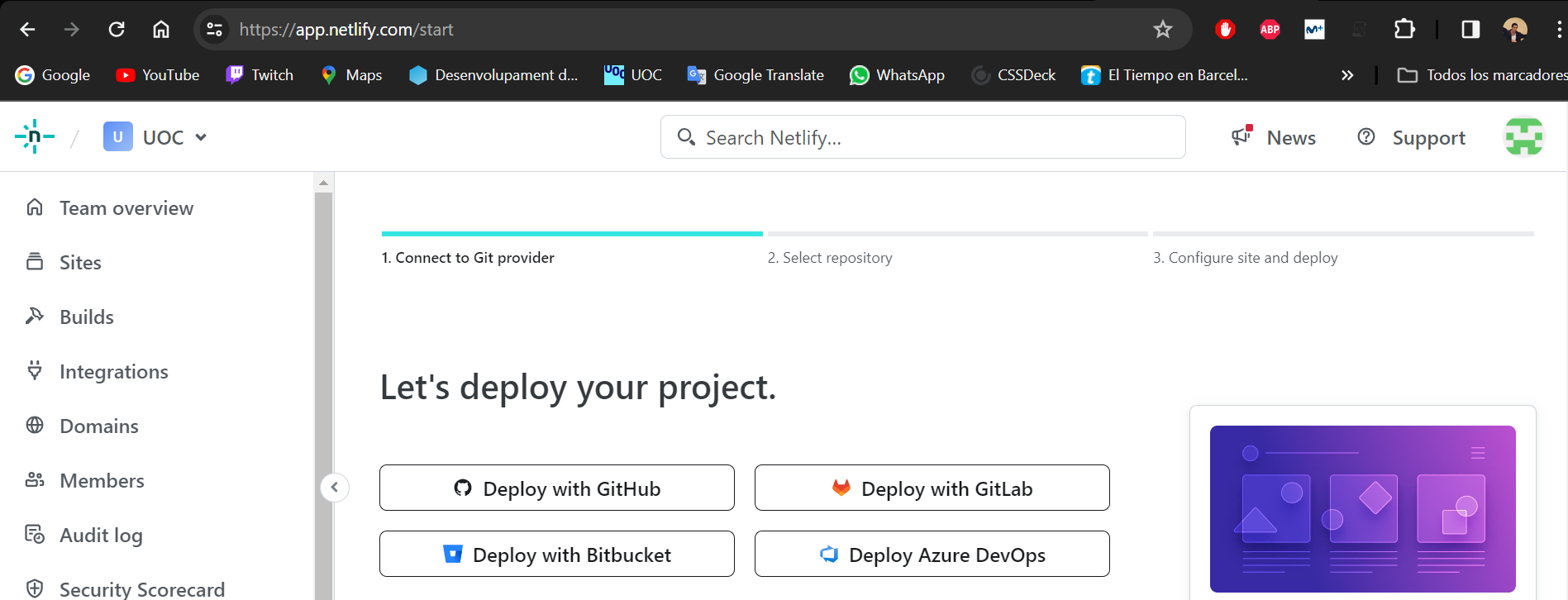
Primer s’executa la comanda npm run build al terminal i npm run dev per tal de compilar tots els recursos necessaris dins del directori dist.



Dins de la pàgina web de Netlify, un cop fet el login, es va al apartat de “Sites” on hi trobarem els projectes realitzats per l’usuari. Alla s’accedeix al desplegable de “Add new site” > “Import an existing project”.

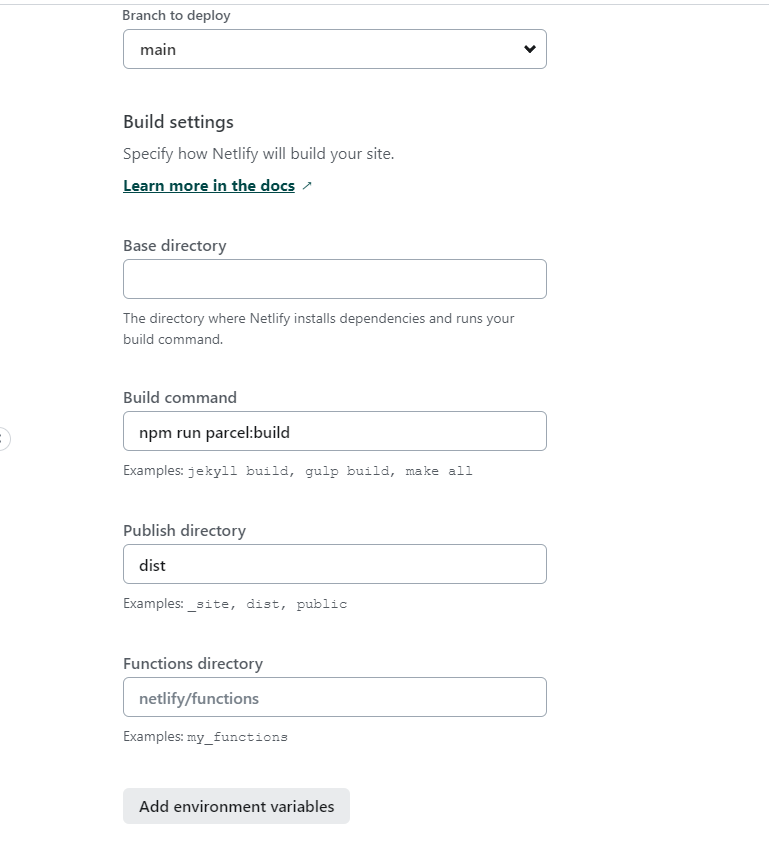


Posteriorment se selecciona la opció “Deploy with GitHub” on se situen tots els projectes vinculats al usuari de GitHub.

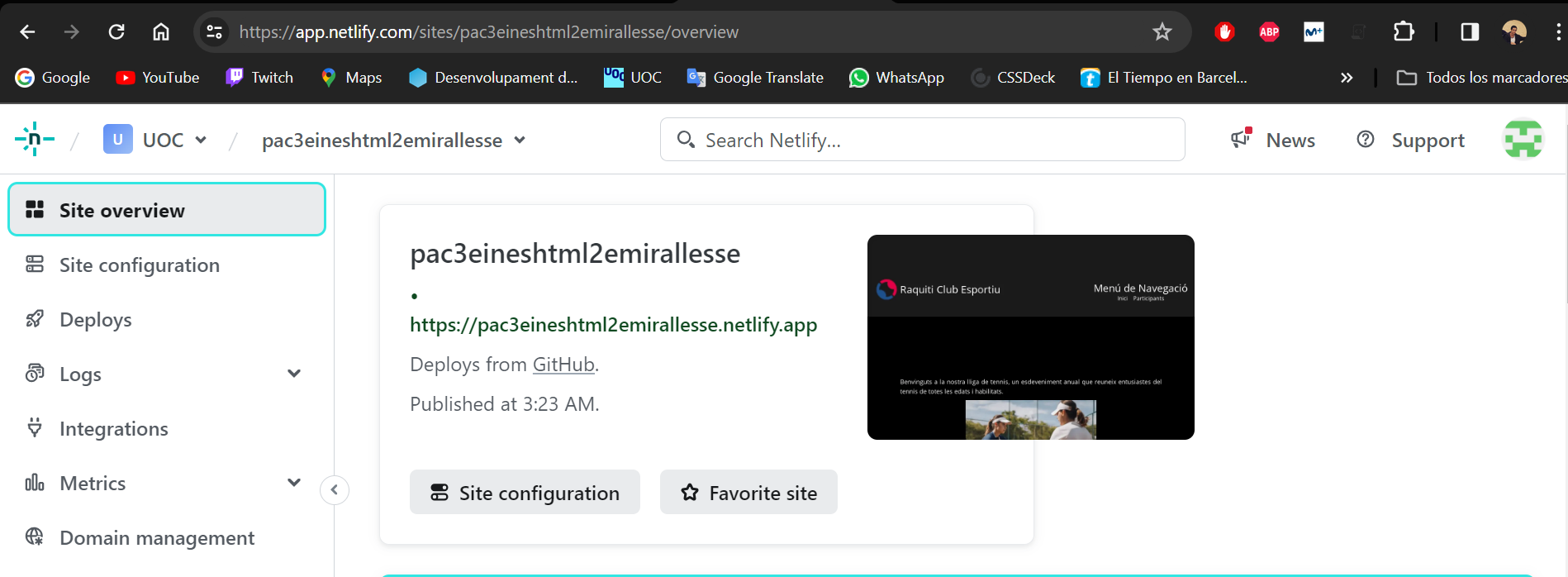


Es dona els permisos a Netlify per a que accepti tots els repositoris existents al compte de GitHub.

Es configura el Deployment en la següent pantalla.



Finalment, un cop fet el Deployment, es canvia la URL dins de l’apartat “site configuration”:



Finalment s’accedeix al link de la web per a observar-ne el resultat un cop finalitzat.



En aquest cas, sembla que l’apartat on hi ha el script de la pàgina participants, no opera correctament un cop fet el Deployment. En servidor local, opera correctament.

1. **Referències**

**General:**

* **Header 🡪 Logo tenis:** <https://iconos8.es/icons>

**Participants:**

* **Generador de persones:** https://this-person-does-not-exist.com/es

**Article:**

* **Generador d’imatges:** https://gencraft.com/
* **Mapa Barcelonès:** https://www.enciclopedia.cat/gran-enciclopedia-catalana/el-barcelones