PAC 3: M4.258 – Eines HTML i CSS II



Enric Miralles i Selvi

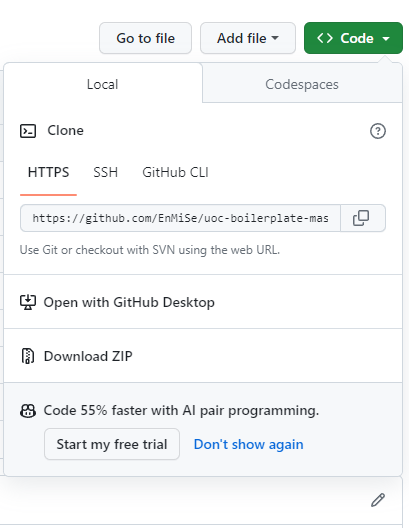
08/01/2024

Link web:

Link repositori:

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 “DownloadZIP” que hi ha a la secció “<> Code” de la pròpia pàgina de Github.



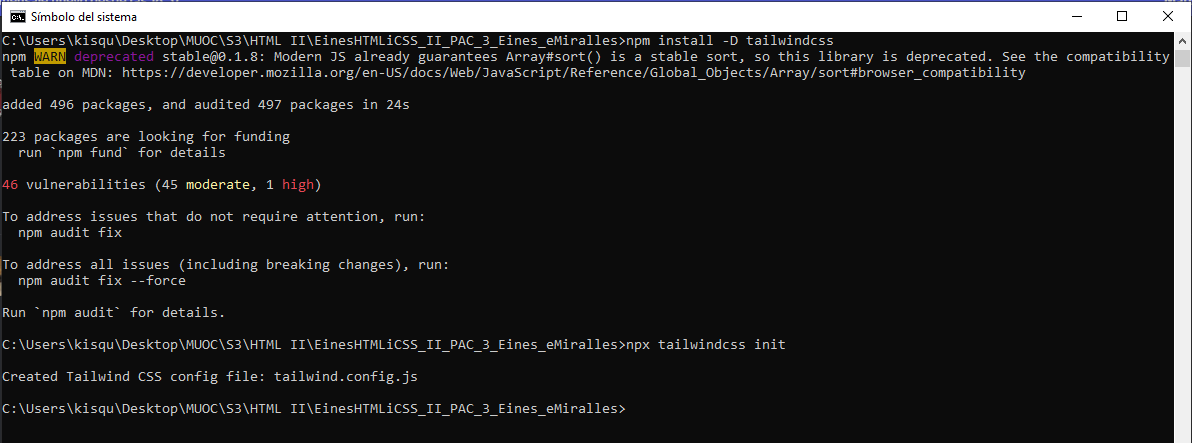
 A continuació, s’ha guardat i descomprimit la carpeta a un directori del dispositiu i s’ha creat un nou repositori mitjançant Github Desktop. Amb un commit inicial, s’ha pujat a Github els fitxers i components del nou projecte.

* 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 l’execució del lloc web en un servidor de desenvolupamentes tira la comanda ‘npm run dev’. Dita comanda, executa el script situat al fitxer ‘package.json’ i 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:

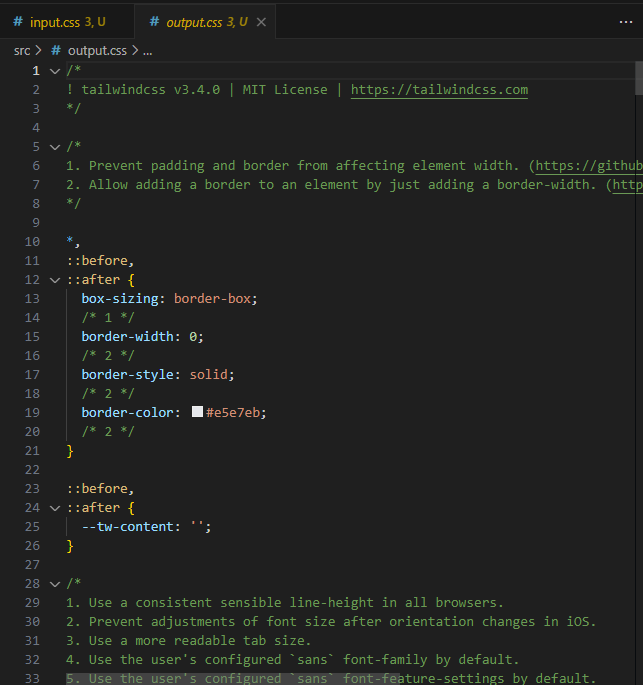


Posar una imatge de src no dist

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.

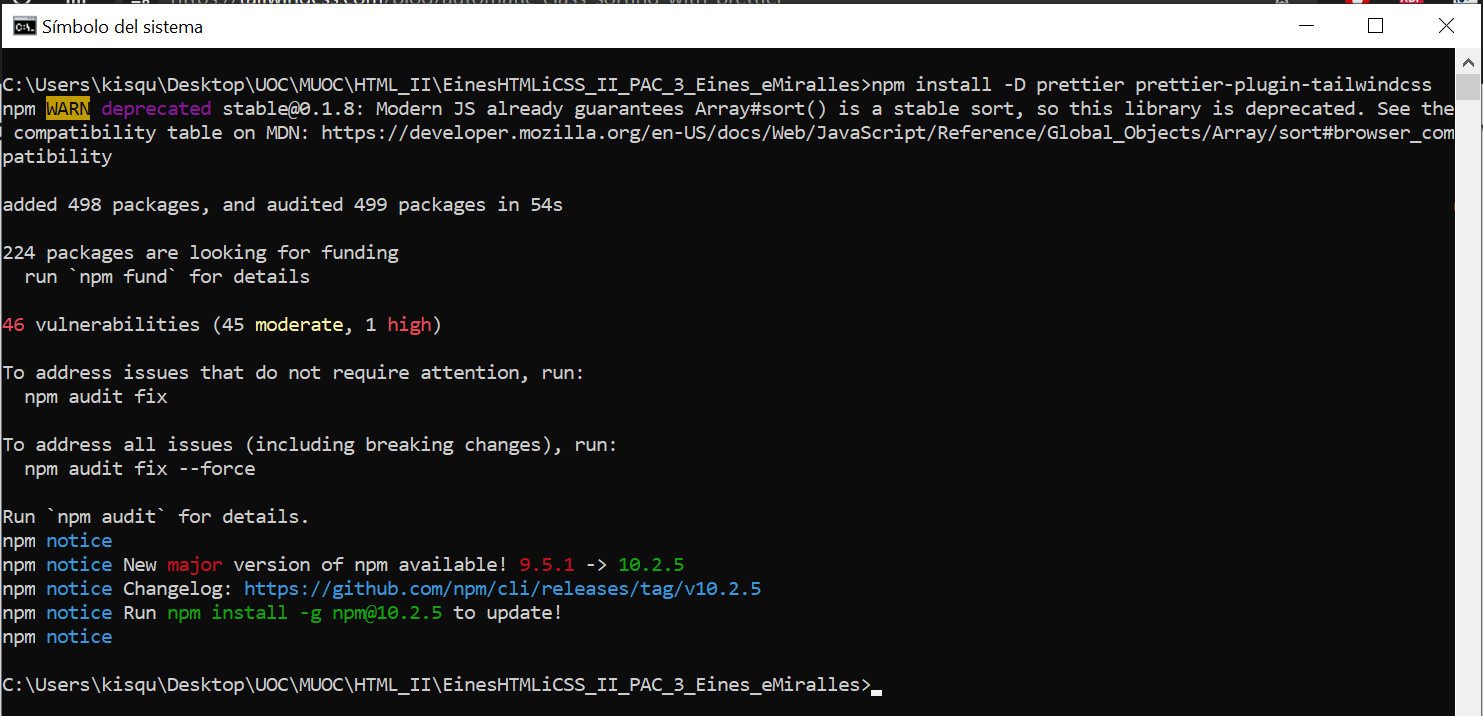
També s’ha ‘npx tailwindcss -i ./src/input.css -o ./src/output.css –watch’. Aquesta comanda indica a Tailwind CSS que es processi l'arxiu CSS d'entrada `input.css` i que generi l'arxiu CSS de sortida `output.css`. L'opció “—watch” s'utilitza perquè Tailwind segueixi observant els arxius d'entrada per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

El fitxer ‘output.css’ on es guarden i es generen els estils es el següent:

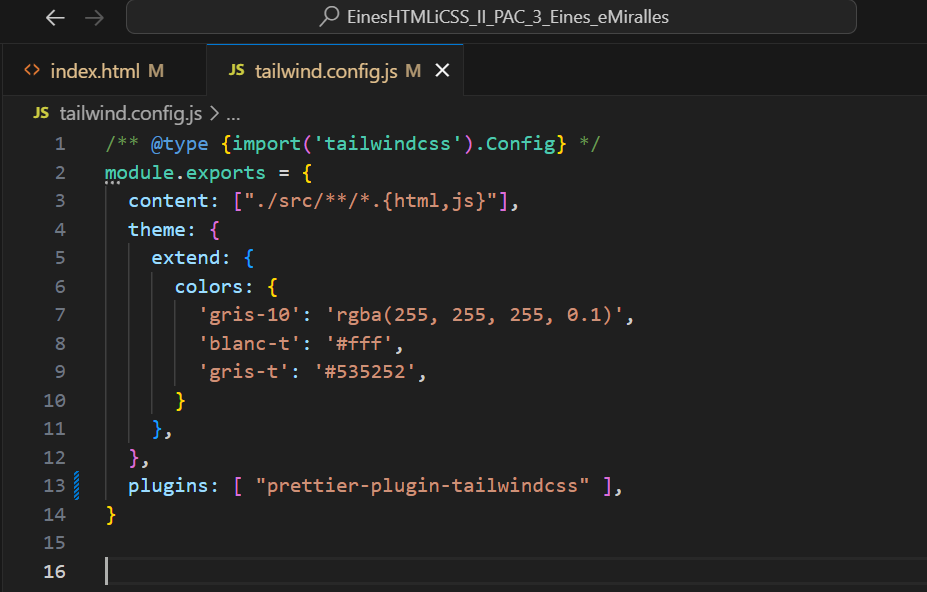


Prettier:

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



Posteriorment s’afegeix el plugin a l’arxiu de configuració de 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.

Les dues pàgines corresponen a la de “Artcile” i “Participants”.

En general, totes les pàgines contenen un “header” i un “footer” compartit.

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. Al menú de navegació se li ha afegit un color en l’estat hover per a cada enllaç. Un blau per a la pàgina article i un vermell a la pàgina de participants, per a seguir amb l’estil de colors usats al logotip.

* **Pàgina interior 1: 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.

La pàgina està estructurada de manera clara amb l'ús de classes BEM (Block Element Modifier) per a l'estilització CSS, permetent una organització i manteniment eficients dels estils. La pàgina també està dissenyada per ser responsiva, adaptant-se a diferents mides de pantalla.

* **Pàgina interior 2: Article**

Pàgina centrada en un article. Conté un <article> 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.

* 1. **Entorn de desenvolupament**
* **Ús de Grid i @supports**
* **Pseudoclasses funcionals**

S’ha utilitzat pseudoclasses a la pàgina de l’article. En aquest cas es fa servir dues pseudoclasses funcionals: :has() i :where().

En el cas de la classe :has() s'utilitza en la regla & :has(> strong) { color: blue; } dins de .general\_\_art &\_\_p. Aquesta pseudoclasse selecciona tots els elements <p> que contenen directament un element <strong> com a fill (> denota un fill directe). Si un element <p> compleix aquesta condició, el color del seu text canviarà a blau.

* 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 la minificació de codi, processament de SCSS o LESS, entre altres.

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

Al fitxer “output.css” s’ha 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.

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”. Pel format mòbil només es necessari donar un format on tot es disposi de manera vertical en una columna, ja sigui amb grid o flex-box.

diferents dispositius. Les pseudoclasses :has() i :where() són utilitzades per a aplicar els estils condicionals.

* 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.

En el cas de la pàgina participants s’ha optat per mostrar els competidors registrats en un format fitxa. Com el disseny implementat per a aquesta secció està fet de manera responsiva, no ha sigut necessari adaptar-la pels diferents formats de resolució de pantalla.

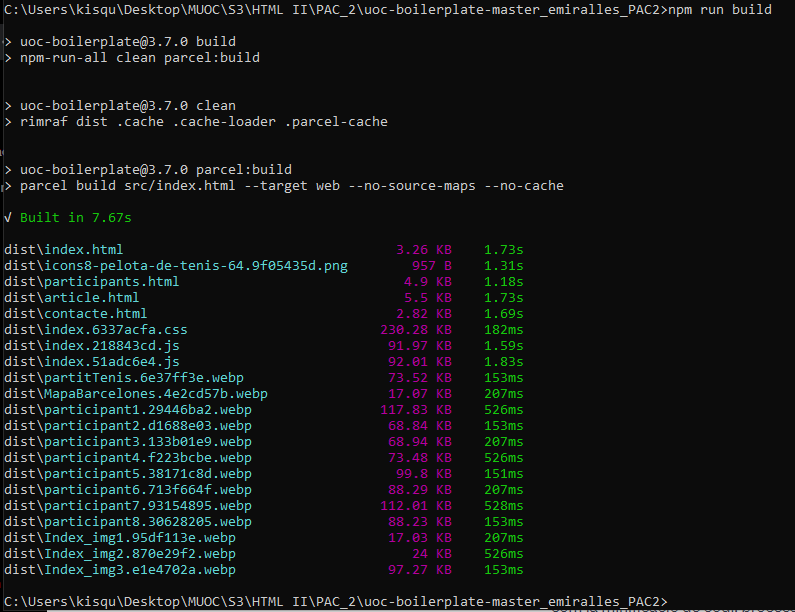
Les diferències entre l'enfocament CSS semàntic i l'enfocament d'utilitats (com Atomic CSS o Tailwind CSS) són significatives tant en el procés de desenvolupament com en l'estructura del codi.

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.

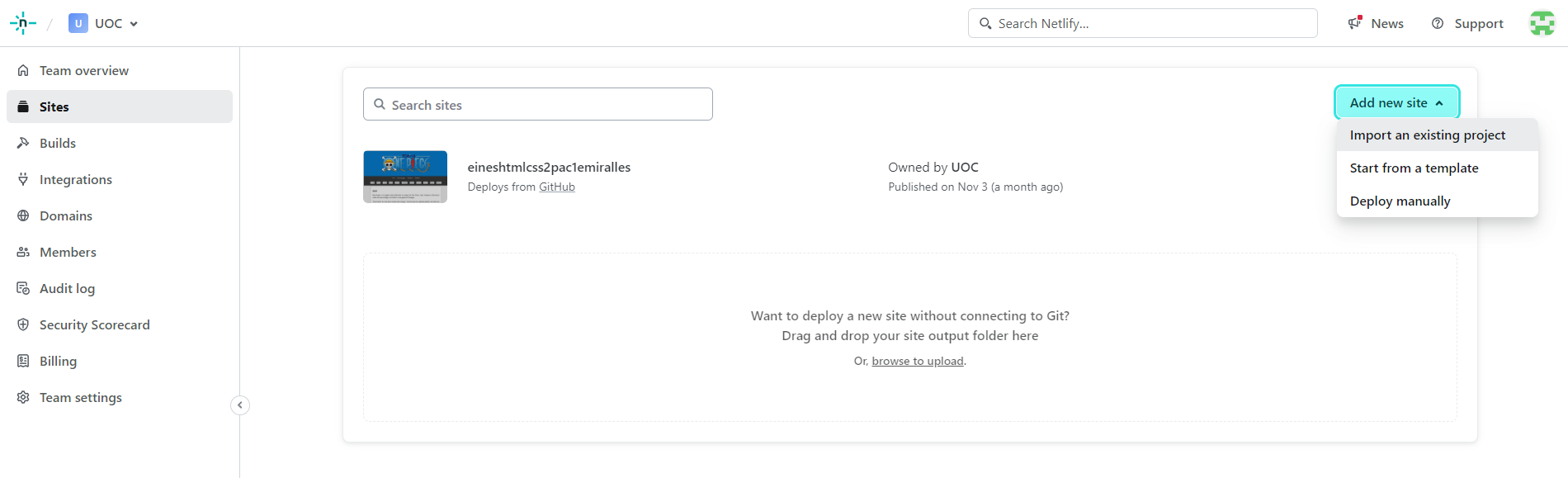
L'ús de llibreries de components contrasta amb les llibreries d'utilitats en que les primeres ofereixen components predefinits amb menys flexibilitat de personalització, mentre que les segones permeten una construcció més granular i directa dels components. La selecció de classes i components a extreure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

1. **Deployment a Netlify**

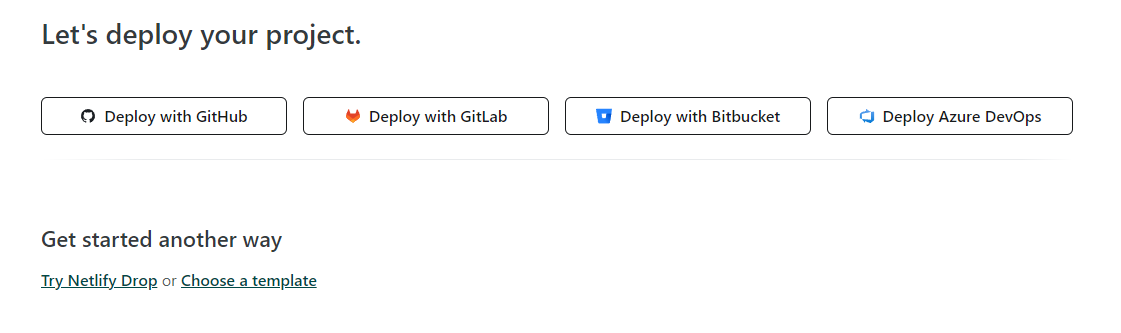
Primer s’executa la comanda npm run build al terminal.



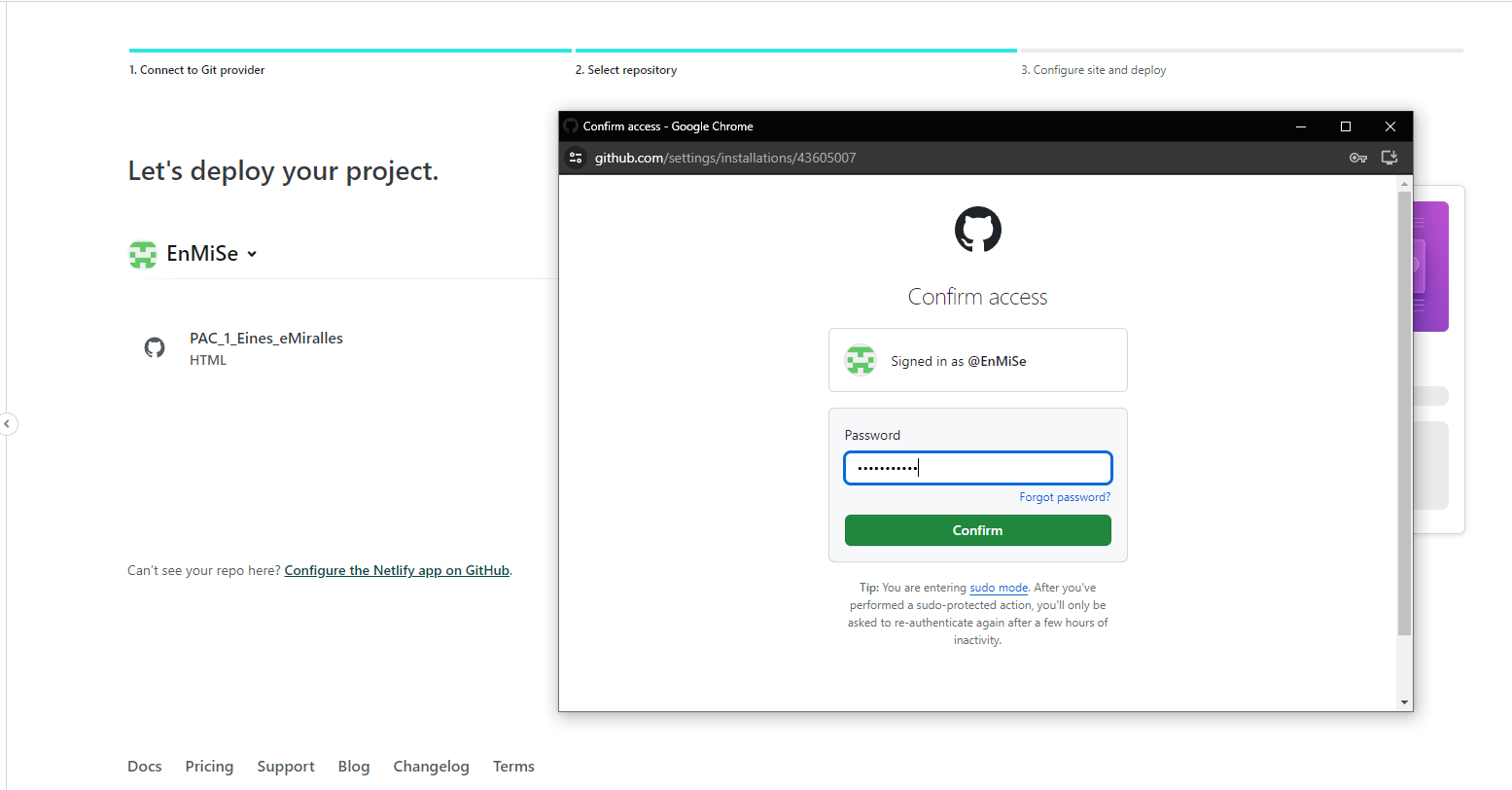
Dins de la pàgina web de Netlify, un cop fet el login, es va al apartat de “Sites” on hi trobarem els projectes realitzats per l’usuari. Alla s’accedeix al desplegable de “Add new site” > “Import an existing project”.



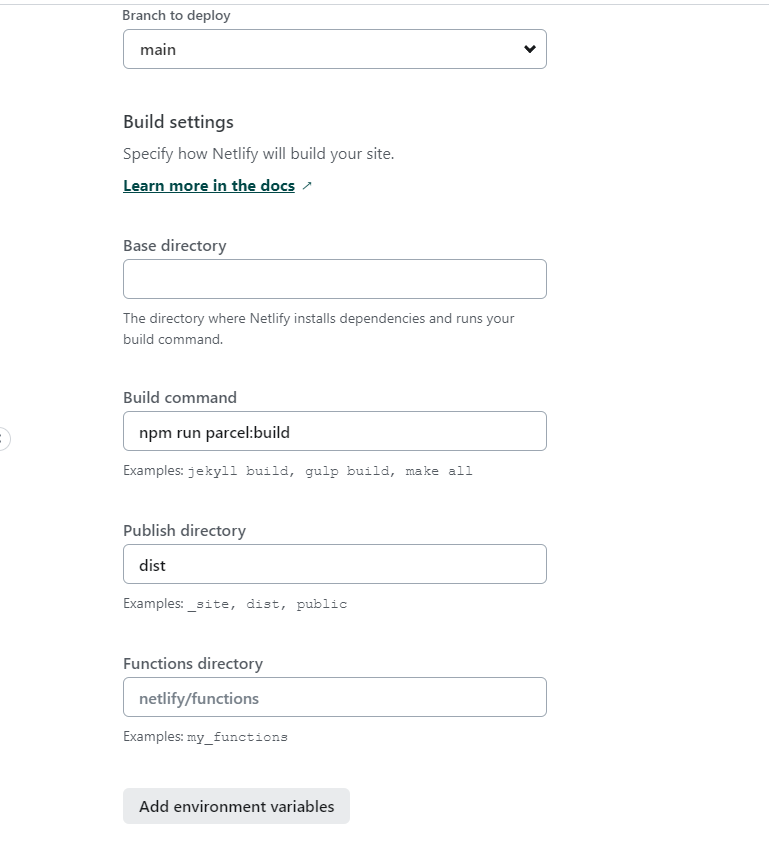
Posteriorment se selecciona la opció “Deploy with GitHub” on se situen tots els projectes vinculats al usuari de GitHub.



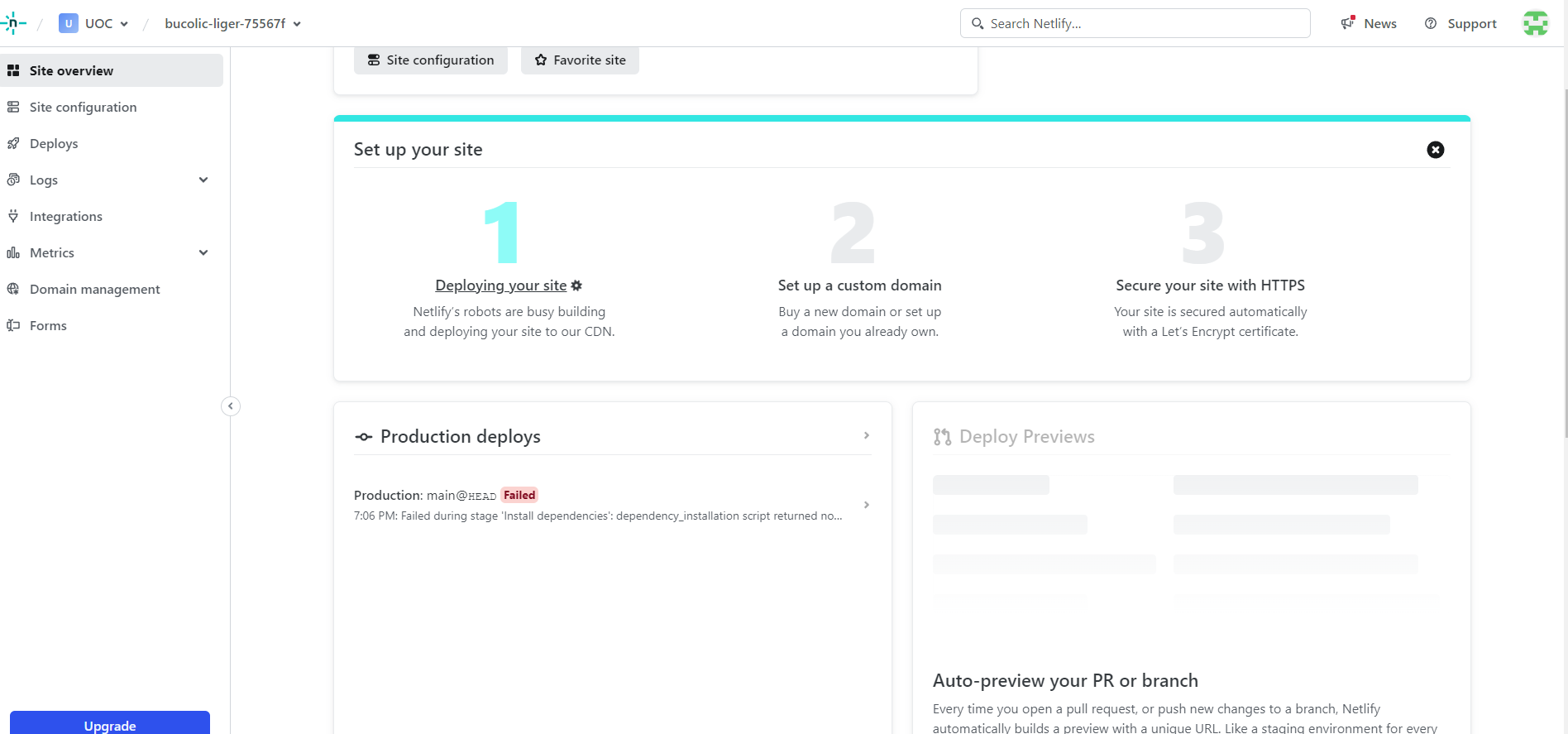
Es dona els permisos a Netlify per a que accepti tots els repositoris existents al compte de GitHub.



Es configura el Deployment en la següent pantalla.



Després d’uns quants intents en la configuració de node i subdependencies que podrien afectar a la publicació del lloc web, no s’aconsegueix fer el deployment. S’ha revisat la versió de node usada (9.5.0) i les compatibilitats de shark que podrien afectar al procés sense obtenir un resultat positiu



1. **Referencies**

**General:**

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

**Index:**

* **Index\_Img 1:** https://weezevent.com/es/blog/organizar-torneo-dia-mundial-del-tenis/
* **Index\_Img 2:** <https://www.freepik.es/vector-premium/tennis-logo-template-vector-deporte-activo-campeonato-torneos-tenis_25881866.htm>
* **Index\_Img 3:** <https://pixabay.com/es/photos/tenis-pelota-de-tenis-vez-pelota-1381230/>

**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