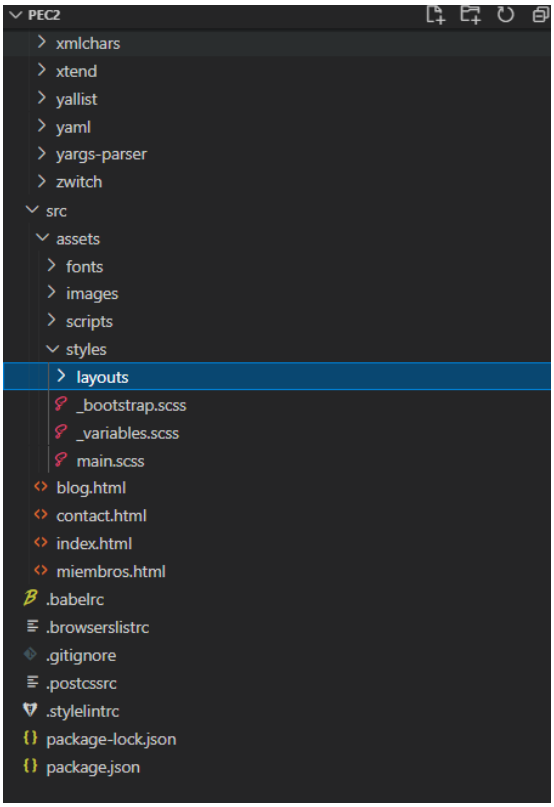
PEC 2 Maquetación con Bootstrap

En esta PEC2 tendremos que crear un sitio web sobre nuestro grupo de música. En esta PEC utilizaremos las herramientas que hemos aprendido a lo largo de los módulos correspondientes a esta PEC, siendo Bootstrap la principal tecnología de esta PEC.
La web constará de cuatro páginas: home, y 3 páginas interiores que serán la página de miembros, blog y contacto.
Para iniciar el desarrollo de este sitio web partiremos del boilerplate de UOC que hemos instalado para la PEC1. Reutilizamos la configuración y realizamos un npm i para instalar todas las dependencias en la que será la carpeta raíz de esta PEC 2.

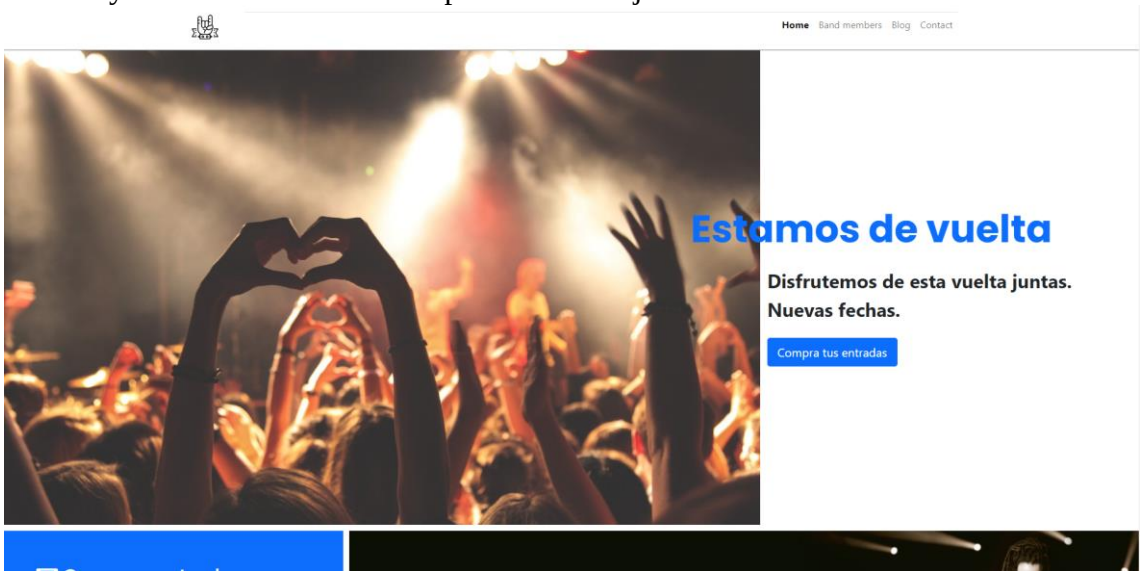


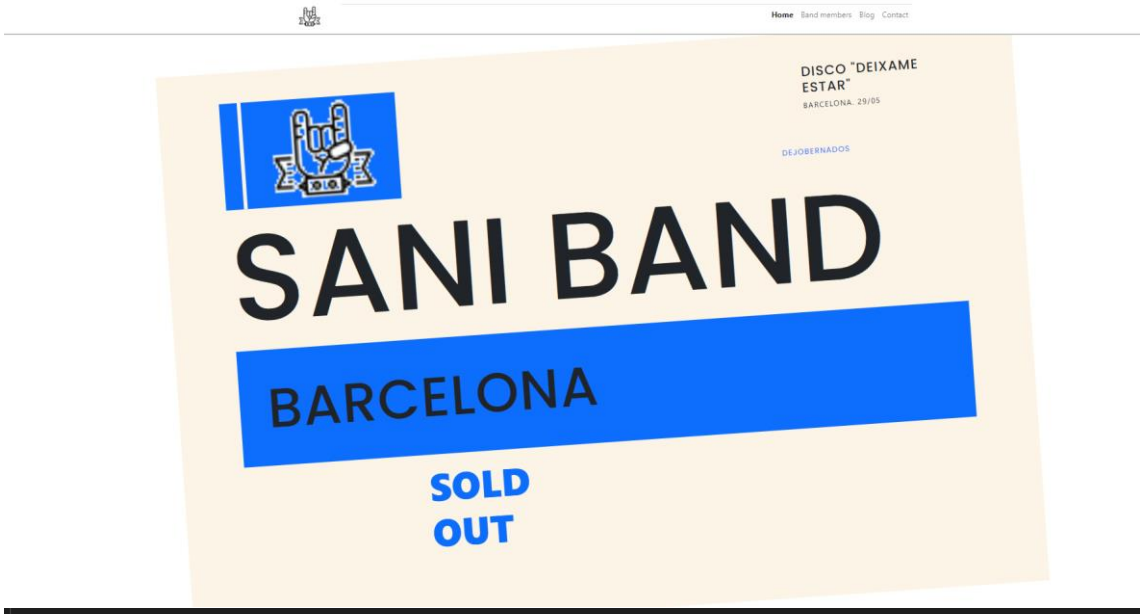
El sitio web debe tener cuatro páginas (página de portada y tres páginas interiores), siguiendo los wireframes proporcionados. La temática del sitio web debe ser información sobre el grupo de música en el que participas

La web que realizaremos tendrá 4 páginas, que serán las siguientes: home, miembros, blog y contacto.

La página de portada: debe estar maquetada con CSS Grid y debes realizar algun diseño interesante, parecido a lo que podría ser un póster promocional del grupo.

Esta página de portada está, tal y como se nos indica en el enunciado, maquetada en CSS Grid. Hemos optado por hacer una especie de banner partido utilizando este método y un contenido más estilo poster más abajo.






```
<div class="container-fluid position-relative">
  <div class="row align-items-center home">
    <div class="col-md-7 col-lg-8 pt-8 px-0"><img class="img-fluid" src="assets/images/concierto.jpg" alt="" />
    </div>
    <div class="col-md-5 col-lg-4 text-center text-md-start pt-5 pt-md-12">
      <h1 class="mb-4 display-3 fw-bold home__title">Estamos de vuelta</h1>
      <p class="mt-3 mb-4 home__subtitle">Disfrutemos de esta vuelta juntas. Nuevas fechas.</p>
      <a class="btn btn-lg btn-primary hover-top" href="#" role="button">Compra tus entradas</a>
    </div>
  </div>
</div>
</section>
```

Hemos decidido trabajar con las columnas de Bootstrap en esta primera parte. Mientras que en la del póster hemos trabajado con clases pero igualmente basándonos en CSS Grid:

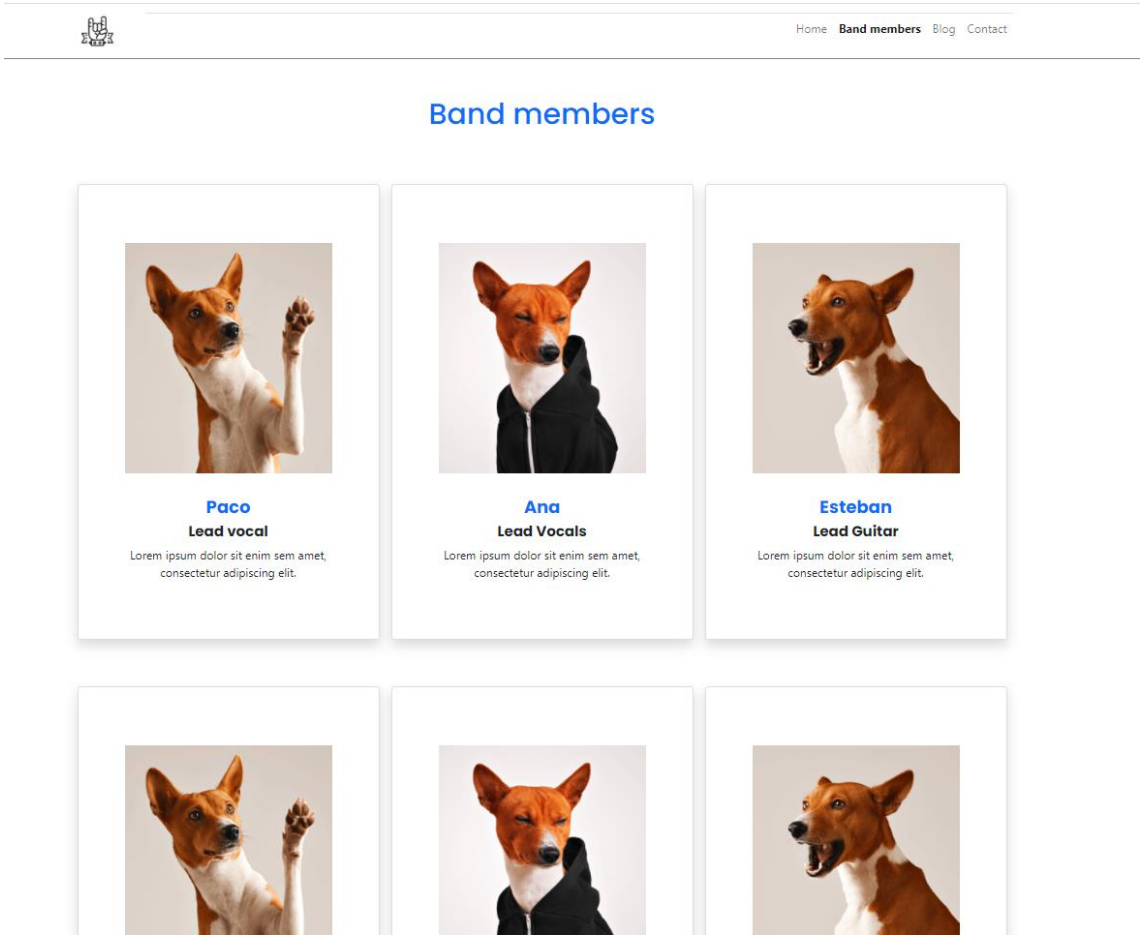
```
<div class="poster__lema">
  <h4>Dejobernados</h4>
</div>

<h1 class="poster__title">Sani Band</h1>

<h2 class="poster__concert">Barcelona</h2>

<div class="poster__soldout">
  <p class="poster__soldout--big">Sold</p>
  <p class="poster__soldout--big">Out</p>
</div>

</div>
</div>
```

Después en nuestros estilos le aplicamos los correspondientes para realizar el poster:

```
.poster__disco {
 grid-column: 3/4;
 display: flex;
 flex-direction: column;
 justify-content: space-between;
 border-left: 2vw solid #fcf4e6;
}

.poster__soldout{
 grid-column:1/2;
 grid-row:5/6;
 color:$band-corporate;
 letter-spacing:0;
 font-weight:800;
 line-height:1;
}
```

Una de las páginas interiores debe incluír una retícula con información sobre los componentes del grupo, maquetada con flex. Para esta retícula, no puedes usar las clases de Bootstrap `.row` o `.col-*`.

En esta página que llamamos members, no hemos utilizado las clases de Bootstrap row ni col. Hemos empleado flex. El resultado ha sido el siguiente:



Para ello hemos utilizado otras herramientas que nos proporciona Bootstrao como son las clases flex como d-flex o el elemento card, con el que están hechos los recuadros.

Una de las páginas interiores debe seguir un formato de artículo e incluír un mínimo de 4 párrafos de texto y 2 fotografías. Puede ser una página que contenga información genérica del festival, una página del estilo noticia o nota de prensa, o cualquier otra opción que cumpla los requisitos descritos. A parte de la maquetación de la página deberás aplicar estilos a algunos elementos HTML que habitualmente podrían formar parte de una página de este tipo, como `blockquote` o listas (como se puede ver en el wireframe).

Esta página la hemos concebido como una noticia de un medio, en este caso es la review de nuestro último disco. Con el consiguiente resultado:

Hemos utilizado también etiquetas HTML semánticas propias de este tipo de artículos.

Etiquetas como <address> para indicar el autor de un post.

También utilizamos las etiquetas de figure y un pie de página para la imagen. Para ello utilizamos la etiqueta <figure> y <figurecaption>

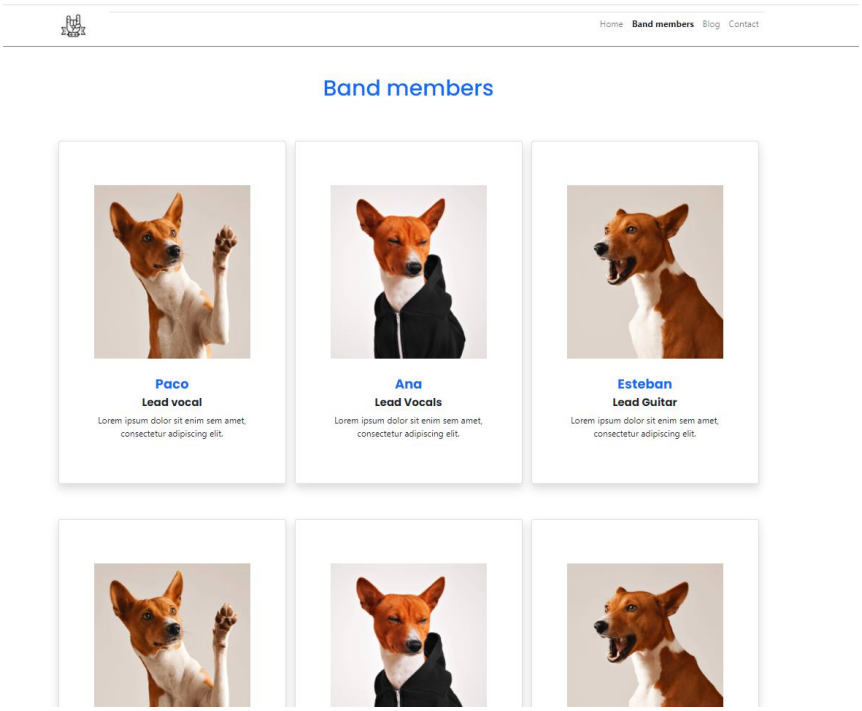
```
<div class="col-sm-12 col-xl-12 pt-8 px-0">
    <figure>
        <img class="img-fluid" src="assets/images/concierto.jpg" alt="" />
        <figcaption>Foto de nuestro último concierto | Fuck COVID</figcaption>
    </figure>
</div>
```

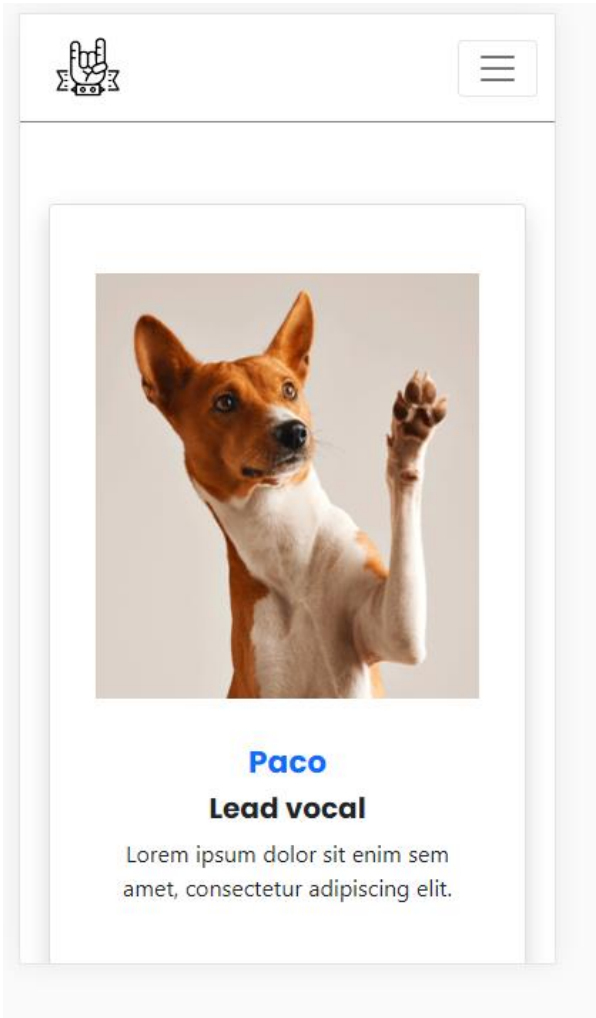
Dejando para el final una pequeña review de otro medio y citándolo utilizando blockquote:

```
    <blockquote
        cite='https://www.mindies.es/2019/12/camellos-presentan-calle-para-siempre-en-madrid-de-la-mano-de-gures/'>
        <p>
            Si ya has ido antes a uno de sus conciertos ya sabes lo que te esperas. <strong>Momentos
                de euforia colectiva</strong> donde el ingenio de sus letras, movidas por el
            humorístico pero veraz análisis social, es capaz de provocar tumultos de personas de lo
            más sudorosas. Dejándonos con la completa sensación de cómo sus nuevos temas son
            coreados con la misma energía que los anteriormente conocidos, os dejamos con la crónica
            del concierto a través de la lente de Joaquín Gómez Ruiz.
        </p>
    </blockquote>
</div>
```

La página debe ser responsive y se debe poder visualizar correctamente desde cualquier dispositivo moderno

En este caso haremos esta web responsive. Para ello utilizaremos medias querys para medir el ancho de la pantalla desde la que se visualiza nuestra página y así el estilo se adapta. Por ejemplo:



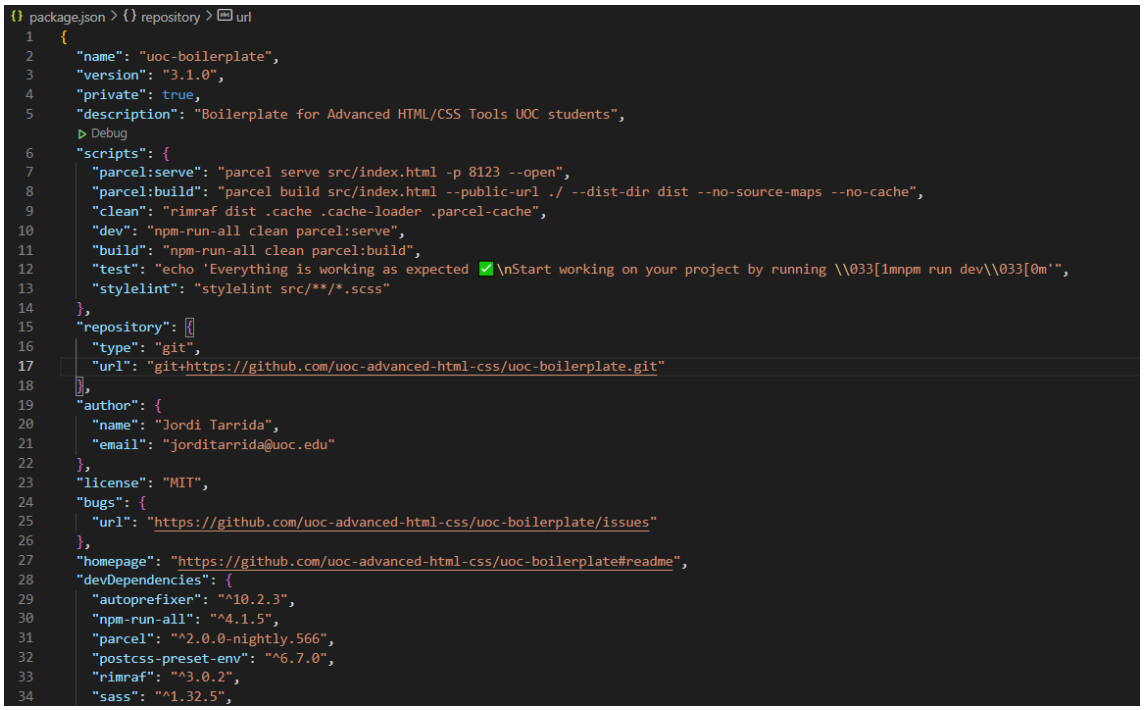


Las imágenes de los perros son: https://www.freepik.es/foto-gratis/adorable-perro-basenji-bostezando-o-hablando-aislado-blanco_11829505.htm

Nuestras páginas se van adaptando al tamaño de la pantalla que este disponible.

Debes partir de UOC Boilerplate para tu desarrollo. Asegúrate de estar usando la última versión (en el momento de escribir este enunciado, la última versión es la 3.1.0).

Tal y como hemos comentado al principio de este documento hemos partido del boilerplate que hemos instalado para las Actividades de los módulos 1 y 2: Lo comprobamos echando un vistazo a nuestro packaje.json:



Debes añadir alguna dependencia externa a UOC Boilerplate.

En este apartado instalaremos una dependencia externa a UOC Boilerplate que nos vendrá muy bien.

No seremos muy originales pero instalaremos FontAwesome que nos puede venir muy bien para los enlaces a las redes sociales.

Para ello lo que haremos en la carpeta raíz de nuestro proyecto será ejecutar el siguiente comando:

```
\Máster de desarrollo web\Herramientas_HTML_CSS_II> npm install --save @fortawesome/fontawesome-free
```

```
+ @fortawesome/fontawesome-free@5.15.3
updated 1 package and audited 1398 packages in 18.705s

202 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

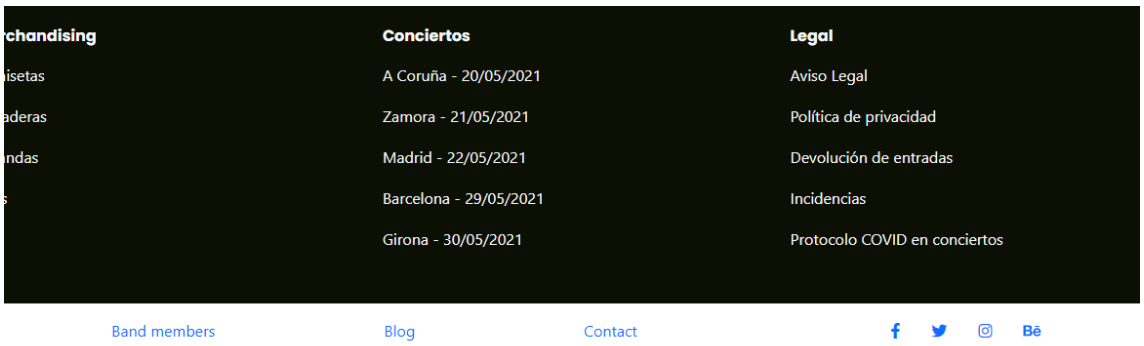
Veamos cómo se ha añadido como dependencia en nuestro packaje.json

```
  },
  "dependencies": {
    "@fortawesome/fontawesome-free": "^5.15.3"
  }
}
```

Para que nos funcione perfectamente añadimos a nuestro archivo main.scss los estilos de la dependencia. Para ello importamos los archivos:

```
 */
@import "@fortawesome/fontawesome-free/scss/fontawesome.scss";
@import "@fortawesome/fontawesome-free/css/all.css";
// @import "some-node-module";
// @import "@some-company/some-node-module";

/**
```

Podemos ver cómo queda en nuestra web:

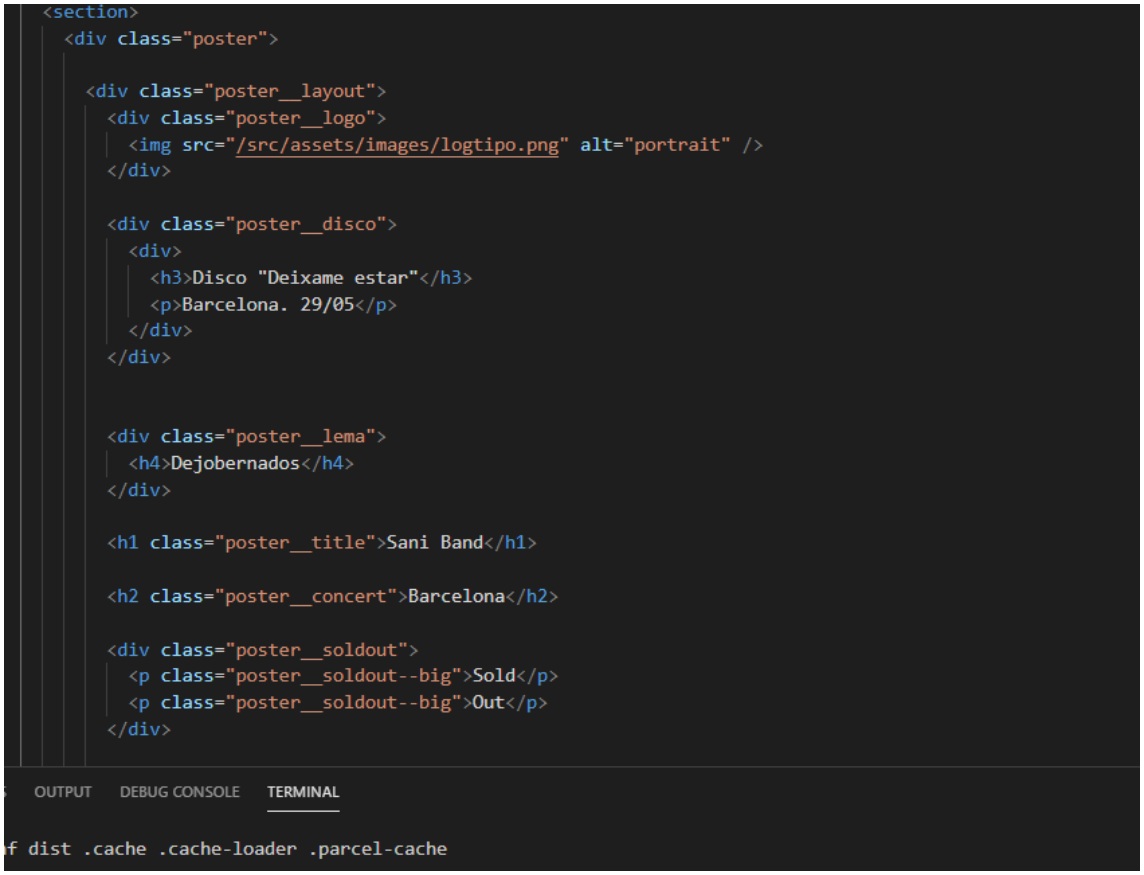


Debes escoger una o más de las metodologías y guías de estilo estudiadas en el módulo 2 y aplicarla en tu desarrollo.

Tal y como hemos comentado también más arriba utilizaremos para este trabajo la metodología BEM: block, element, modifier.

Para ello hemos seguido y aplicado clases en base a este estilo.

Por ejemplo, en nuestro código tenemos una clase que e section como bloque y después title como elemento:




```
<section>
  <div class="poster">

    <div class="poster__layout">
      <div class="poster__logo">
        <img src="/src/assets/images/logtipo.png" alt="portrait" />
      </div>

      <div class="poster__disco">
        <div>
          <h3>Disco "Deixame estar"</h3>
          <p>Barcelona. 29/05</p>
        </div>
      </div>


      <div class="poster__lema">
        <h4>Dejobernados</h4>
      </div>

      <h1 class="poster__title">Sani Band</h1>

      <h2 class="poster__concert">Barcelona</h2>

      <div class="poster__soldout">
        <p class="poster__soldout--big">Sold</p>
        <p class="poster__soldout--big">Out</p>
      </div>
```

La razón de haber elegido esta metodología es porque me parece una metodología intuitiva ya que va desde algo genético como es un bloque, pasando por un elemento y llegando al modificador.

También me parece que estoy más familiarizado con este tipo de estilo y, por ello, me he decantado por él. También me he encontrado con que es una metodología extendida y hay buena documentación.

Para comprobar que hemos seleccionado bien el naming utilizamos Stylelint. Configuramos stylelint para que se adapte a nuestra metodología. Nos descargamos una

configuración básica y añadimos a las reglas el patrón de naming: "selector-class-pattern": "^.[a-z]([a-z0-9-]+)?(__([a-z0-9]+-?)+)?(--([a-z0-9]+-?)+){0,2}$"

Debes usar Bootstrap 5, cargado en UOC Boilerplate y personalizado con Sass como se indica en los materiales, y usar un mínimo de 4 componentes distintos.

Para esta práctica era necesario el uso de Bootstrap 5. En esta práctica hemos seguido la guía que utilizamos en el módulo 4.1 perteneciente a Bootstrap para instalarlo.

Hemos utilizado el siguiente comando: `npm install @popperjs/core bootstrap@next --save`.

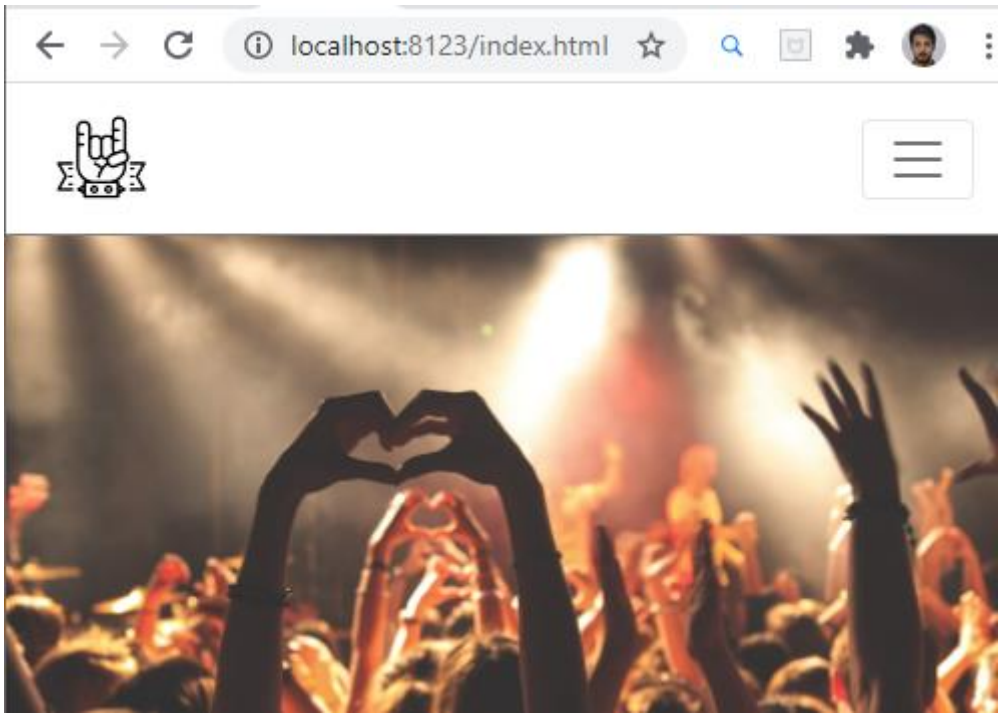
Después hemos importado las dependencias en los lugares correspondientes.

```
@import "/node_modules/bootstrap/scss/bootstrap";
@import "@fortawesome/fontawesome-free/css/all.css";
// @import "some-node-module";
// @import "@some-company/some-node-module";

/**
 * Import layouts
 * It's a best practice to use a different partial for eac
 */

@import "layouts/home";
@import "layouts/members";
@import "layouts/blog";
@import "layouts/contacto";
@import "layouts/header";
@import "layouts/main-content";
@import "layouts/sections";
@import "layouts/responsive";

@import "bootstrap";

/**
```

Nos hemos importado Bootstrap en nuestro archivo main para poder coger todos los estilos.

Podemos ver que en nuestro código al aplicar los estilos propios de Bootstrap se están aplicando:

```
<main class="main" id="top">
  <nav class="navbar navbar-expand-lg navbar-light fixed-top main_header"
    data-navbar-on-scroll="data-navbar-on-scroll">
    <div class="container"><a class="navbar-brand d-flex align-items-center fw-bold fs-2" href="#"><img
          class="d-inline-block me-3" src="assets/images/logtipo.png" alt="" /></a>
      <button class="navbar-toggler collapsed" type="button" data-bs-toggle="collapse"
        data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false"
        aria-label="Toggle navigation"><span class="navbar-toggler-icon"></span></button>
      <div class="collapse navbar-collapse border-top border-lg-0 mt-4 mt-lg-0" id="navbarSupportedContent">
        <ul class="navbar-nav ms-auto pt-2 pt-lg-0">
          <li class="nav-item"><a class="nav-link fw-bold active" aria-current="page" href="#">Home</a></li>
          <li class="nav-item"><a class="nav-link" href="miembros.html">Band members</a></li>
          <li class="nav-item"><a class="nav-link" href="blog.html">Blog</a></li>
          <li class="nav-item"><a class="nav-link" href="contact.html">Contact</a></li>
        </ul>
      </div>
    </div>
  </nav>
```



Además de esta dependencia, debes instalar Stylelint y personalizar su configuración para que aplique los criterios de estilos elegidos, con plugins y reglas. Cuando ejecutes la orden para validarlo no debe aparecer ningún error.

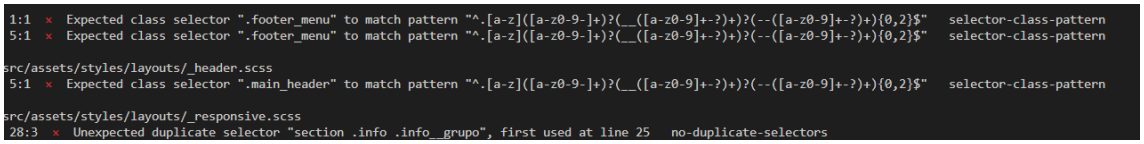
Instalamos entonces para esta práctica Stylelint

```
"devDependencies": {
  "autoprefixer": "^10.2.3",
  "npm-run-all": "^4.1.5",
  "parcel": "^2.0.0-nightly.566",
  "postcss-preset-env": "^6.7.0",
  "rimraf": "^3.0.2",
  "sass": "^1.32.5",
  "sharp": "^0.27.1",
  "stylelint": "^13.12.0",
  "stylelint-config-recommended-scss": "^4.2.0",
  "stylelint-scss": "^3.19.0"
},
"dependencies": {
  "@fortawesome/fontawesome-free": "^5.15.3",
  "@popperjs/core": "^2.9.2",
  "bootstrap": "^5.0.0-beta3"
}
```

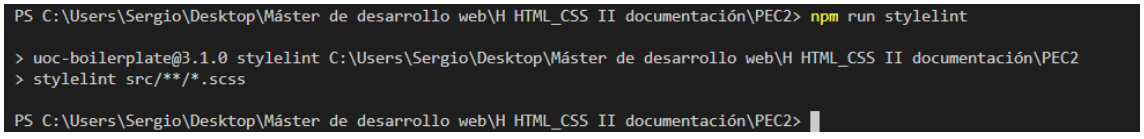

Podemos ver cómo contamos con dichas dependencias en nuestro packaje.json Lo siguiente que haremos será empezar a seleccionar la metodología que vamos a utilizar en nuestra PEC. Utilizaremos la metodología BEM.

Configuramos stylelint para que se adapte a nuestra metodología. Añadimos a las reglas el patrón de naming: "selector-class-pattern": "^.[a-z]([a-z0-9-]+)?(__([a-z0-9]+-?)+)?(--([a-z0-9]+-?)+){0,2}$"

Que como podemos observar función al utilizar el comando npm run stylelint y nos ofrece fallos de estilo en nuestro código. Si solucionamos estos problemas nos dará un código limpio.



Tras revisar los errores que nos da la herramienta, debería no haber ningún error.



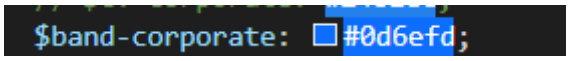
Debes usar necesariamente las siguientes características de Sass: variables, anidado, funciones, parciales e importación.

Mostraremos ahora cómo hemos trabajado con Sass. Iremos mostrando que hemos hecho aprovechando las diferencias características de Sass

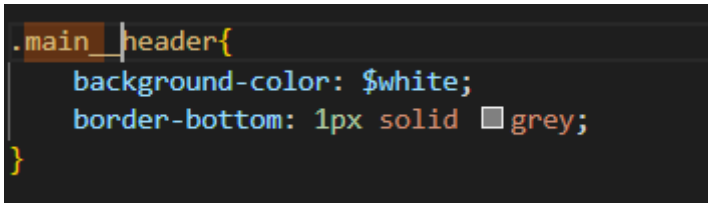
Variables

Para utilizar las variables tenemos un fichero scss de variables que después importaremos en el archivo principal de sass.

Hemos utilizado pocas variables pero las que hemos utilizado están ahí:

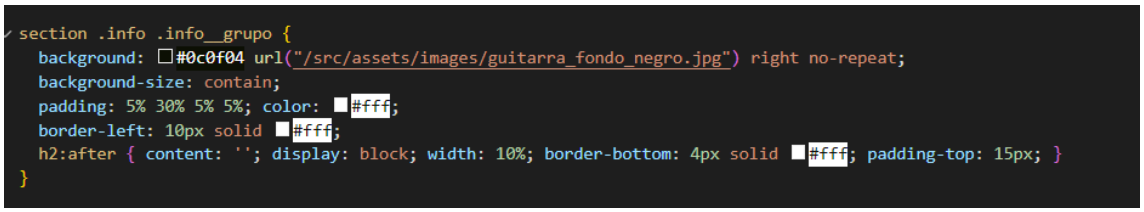


Estas variables después las hemos ido añadiendo a las distintas propiedades en el resto de hojas de estilo. Por ejemplo:



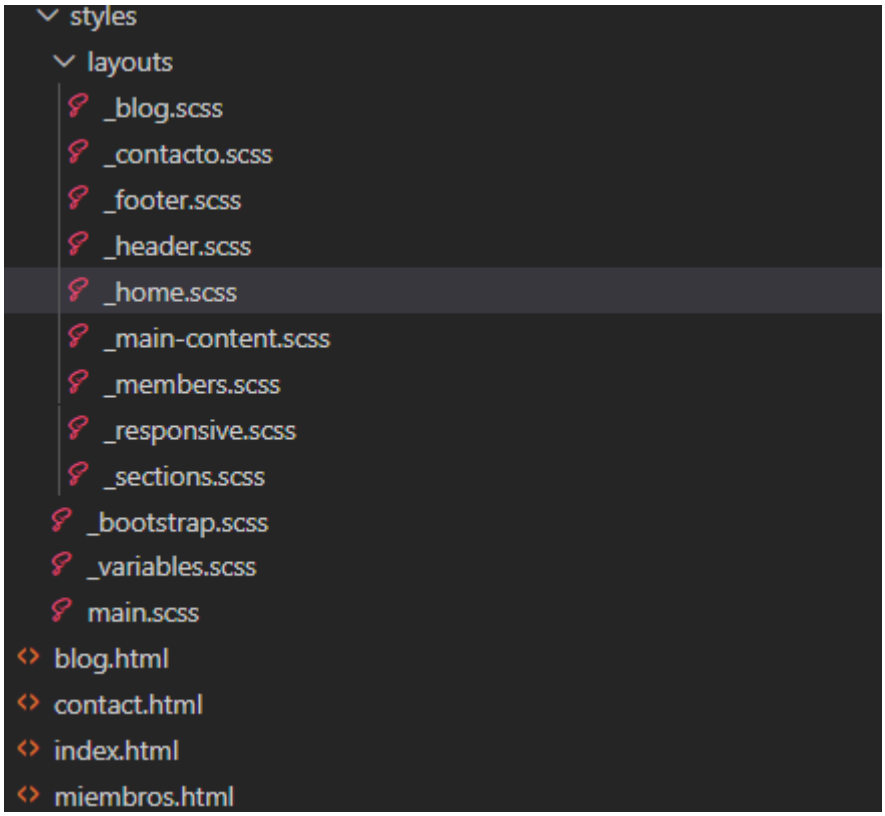
Anidado

Esta característica de Sass nos permite simplificar el código css y tener más claro qué depende de qué. Hemos utilizado el anidado para el estado :after.



Parciales e importación

Para esta parte hemos decidido dividir nuestro código en distintas partes para que nos quede un código más claro, legible y con un mayor mantenimiento. Para ello hemos dividido las hojas de estilo de sass en distintas partes como pueden ser el header o las queries en el archivo responsiveness

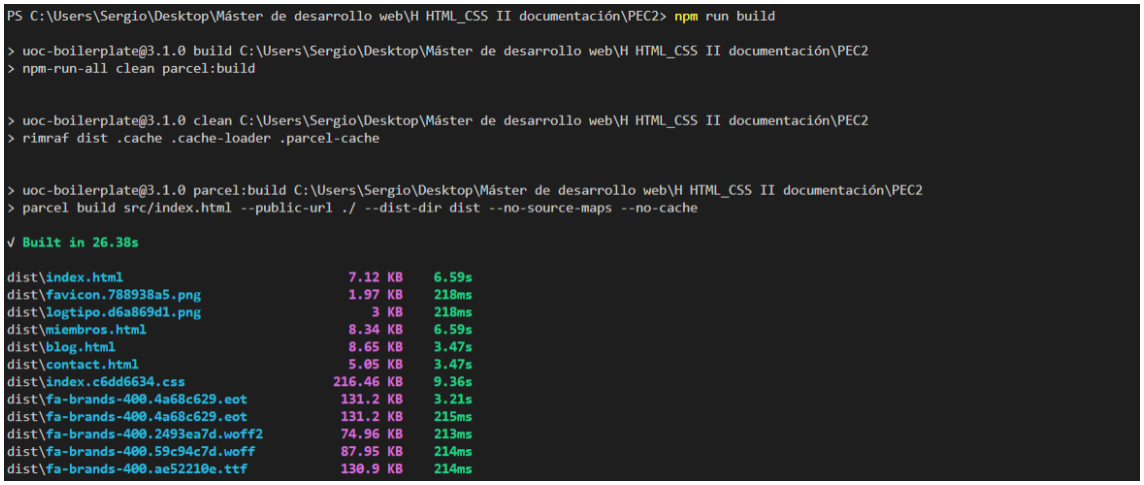


Para indicarle a Sass que son parciales a estos archivos les hemos añadido una _ al inicio del archivo para que no se compilen y se importen en el archivo principal. En nuestro archivo principal, que es main.scss que sí se compilará a css añadimos @import de todas las partes así como la hoja de estilos de la dependencia Font Awesome que hemos instalado

```scss
@import "variables";

/**
 * Import npm dependencies
 * see: https://v2.parceljs.org/features/module-resolution/
 * see commented examples below
 */
@import "/node_modules/bootstrap/scss/bootstrap";
@import "@fortawesome/fontawesome-free/css/all.css";
// @import "some-node-module";
// @import "@some-company/some-node-module";

/**
 * Import layouts
 * It's a best practice to use a different partial for each page
 */

@import "layouts/home";
@import "layouts/members";
@import "layouts/blog";
@import "layouts/contacto";
@import "layouts/header";
@import "layouts/main-content";
@import "layouts/sections";
@import "layouts/responsive";

@import "bootstrap";

/**
 * Do NOT include anything else in this file!
 */
```

Una vez finalices el desarrollo debes publicar el código en GitHub y realizar un deployment en Netlify,

Es ahora el momento de subir nuestro código a nuestro repositorio de Github para posteriormente subirlo al servidor de Netlify. Lo primero que haremos será hacer el npm run build para preparar el código que subiremos a "producción".



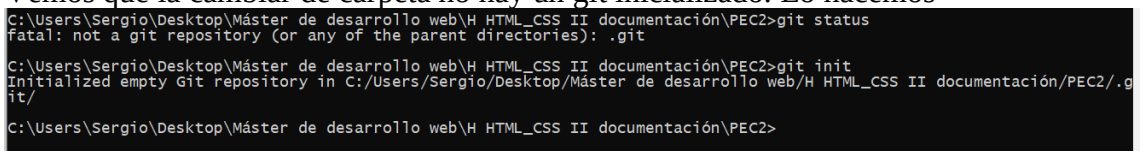
Vemos como lo ha hecho y podremos apreciar cambios en nuestra carpeta dist. Lo primero será tener ese repositorio en Github creado.

En nuestro caso utilizaremos el que hemos usado para hacer las prácticas de los Módulos 1 y 2 y para la PEC1:

https://github.com/ssanisidro/HTML_CSS_PEEC2

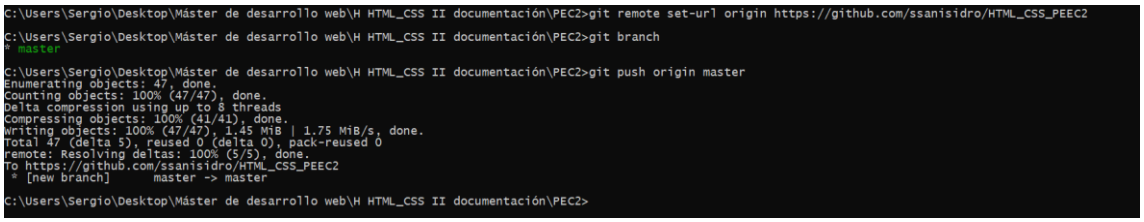
Vamos entonces a hacer la subida por comandos con GIT CMD. Vamos a nuestra carpeta y vamos a ver qué estatus tiene con git status

Vemos que la cambiar de carpeta no hay un git inicializado. Lo hacemos



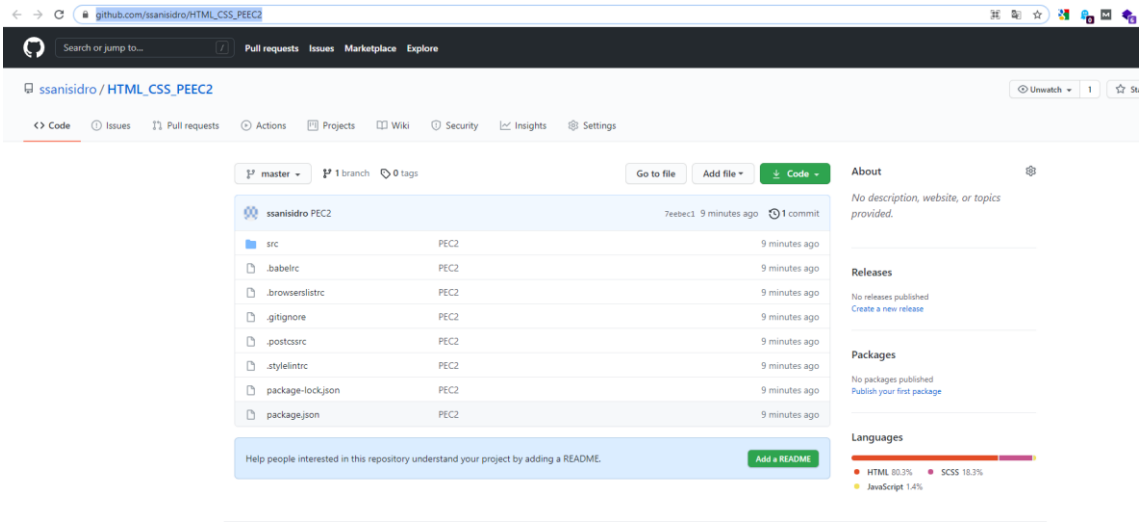
Vamos a añadir el contenido con el comando git add -A

Y ahora hacemos el commit que llamaremos PEC2



```
C:\Users\Sergio\Desktop\Máster de desarrollo web\H HTML_CSS II documentación\PEC2>git add -A
warning: LF will be replaced by CRLF in .babelrc.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in .browserslistrc.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in .gitignore.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in .postcssrc.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package-lock.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/assets/scripts/main.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/assets/styles/_variables.scss.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/assets/styles/layouts/_home.scss.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/assets/styles/main.scss.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/index.html.
The file will have its original line endings in your working directory

C:\Users\Sergio\Desktop\Máster de desarrollo web\H HTML_CSS II documentación\PEC2>git commit -m "PEC2"
[master (root-commit) 7eebec1] PEC2
 38 files changed, 14876 insertions(+)
 create mode 100644 .babelrc
```

Comprobamos que estamos en la rama de master y hacemos el git push para subir el contenidoComprobamos que se ha subido a nuestro repositorio:



https://github.com/ssanisidro/HTML_CSS_PEEC2

Vamos ahora a ver Netifly.

Como ya lo teníamos conectado por los módulos 1 y 2 ahora se ha actualizado solo y ha cogido la carpeta dist de nuestro repositorio para desplegar el sitio web. Nuestro CV está disponible en el siguiente enlace:

https://laughing-tesla-818a16.netlify.app