DOCUMENTACIÓN PEC 1 – HTML Y CSS I

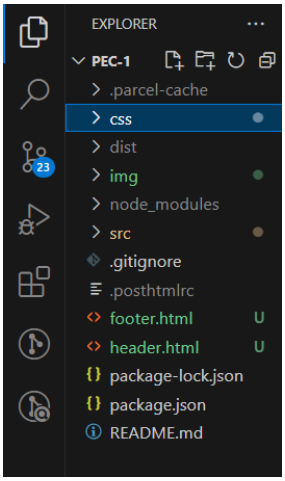
ENLACE DIRECTORIO GITHUB: https://github.com/HelenaR10/PEC-1.git

URL DE LA WEB: https://webbillieilishmania.netlify.app/

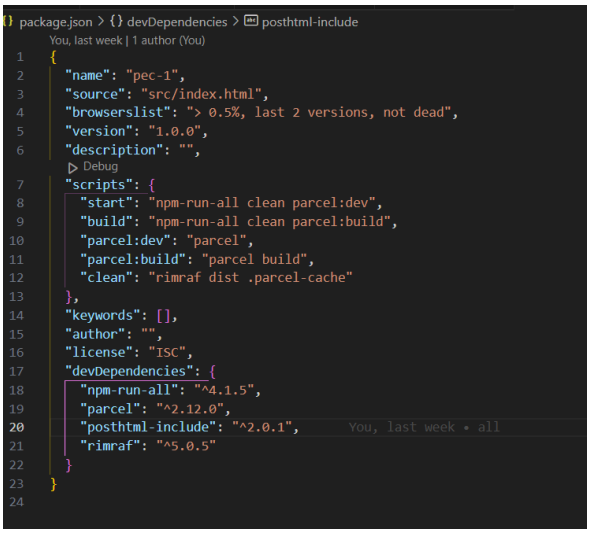
1. **Creación del boilerplate basado en Parcel. Como se ha definido el entorno de desarrollo y producción, y cómo se configura el soporte para navegadores antiguos.**

En este punto he seguido los pasos a realizar de la actividad previa M2. Elegí Parcel como bundler debido a su facilidad de uso y configuración mínima. Otras opciones como Webpack son más potentes pero requieren una configuración más compleja, también decidí utilizar Babel para la transpilación del código JavaScript a una versión compatible con navegadores antiguos, ya que es una herramienta ya incluida en Parcel y ofrece un amplio soporte para transformaciones de código.

En primer lugar, creé una estructura de archivos básica: una carpeta src para guardar los archivos HTML y JS, una carpeta CSS para guardar la hoja de estilos, una carpeta IMG para guardar todas las imágenes y los archivos que van a ser necesarios para Parcel los he dejado en la misma raíz del proyecto.



En segundo lugar, creé un package.json y le añadí las líneas de código propuestas en el ejercicio previo para configurar el entorno de desarrollo y producción.



Utilicé Babel para transpilar el código JavaScript a una versión compatible con navegadores antiguos.

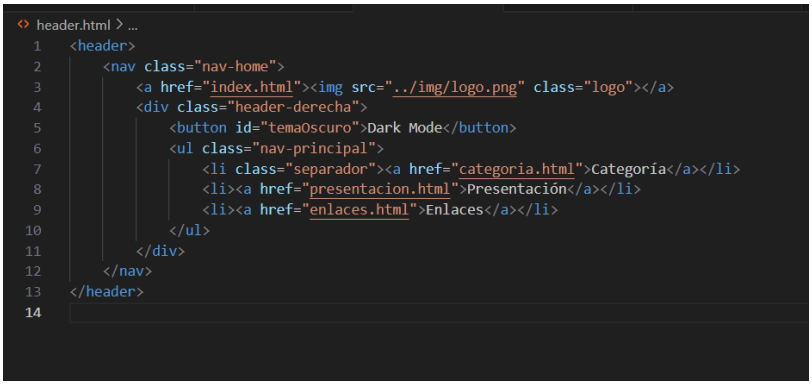
Para la configuración del soporte para navegadores antiguos definí una browserlist para que Parcel la utilice automáticamente y garantice el soporte adecuado para navegadores antiguos.

2. Gestión de dependencias: pre- o postprocesadores y dependencias adicionales.

He utilizado npm como gestor de las dependencias utilizando los comandos npm run start para iniciar un servidor de desarrollo local para el proyecto y npm run build para generar una versión optimizada y lista para producción del proyecto.

En este apartado instalé el plugin de postHTML y creé el archivo `.posthtmlrc.` Después creé un archivo para el footer y otro para el header, ya que en todas las páginas son comunes y los incrusté en todos los index mediante la etiqueta `<include>.`

Creo que este método es muy útil para escribir menos código y que la estructura de ficheros quede de forma organizada.

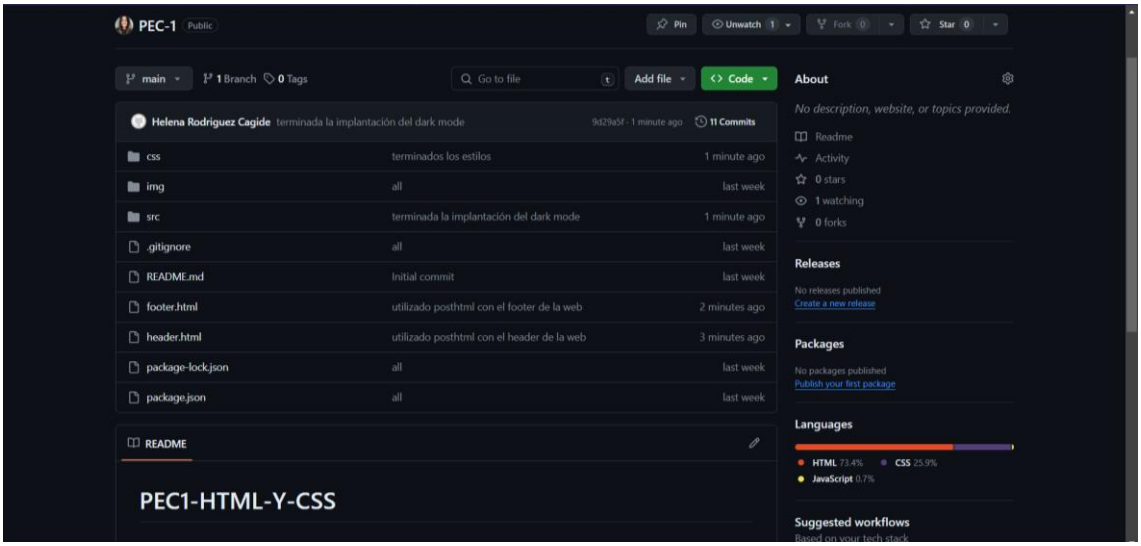


```
    <link href="https://fonts.googleapis.com/css2?family=IBM+Plex+Sans
</head>
<body>
    <include src="header.html"></include>
    <main>
        <div class="home-principal">
            <h1>BillieEilishManía</h1>
            <h3>La web para los más fans de Billie Eilish</h3>
        </div>
```

Además, con el comando npm run build también se ejecuta autoprefixer ya que viene por defecto con Parcel, el cual agrega los prefijos necesarios según la configuración especificada en el browserlist.

3. Creación del repositorio Git

En primer lugar, creé un repositorio en GitHub y me lo descargué en mi equipo. Después, una vez realizados todos los cambios y commits hice un Git push para añadir todas las actualizaciones ignorando la carpeta de node-modules y parcel caché.



4. Adecuación a la temática y estructura de la práctica.

Se han creado en total cinco documentos HTMT:

En cuanto a la temática, he decidido crear la web sobre la famosa cantante Billie Eilish, ya que es una artista que me gusta y que creo que por su estilo podía dar juego en el diseño de la web.

Se ha seguido la estructura propuesta en el ejercicio:

- **Index:** en él tenemos la portada de la web donde se muestra un diseño minimalista con los álbumes de la artista enlazados a Spotify.
- **Categoría:** donde se muestran algunas noticias relevantes y curiosidades de la cantante. Solo una de ellas (la biografía) se enlaza correctamente con la página Detalle. Las demás noticias también enlazan con esta, pero no se muestra la información correspondiente.

- **Detalle:** donde se muestra la noticia detallada de la página categoría.
- **Presentación:** en ella se hace una breve explicación sobre el contenido de la web. Este texto se ha generado por medio de una IA.
- **Enlaces:** en ella se muestran los enlaces de donde se han sacado la información e imágenes que forman la web.
  Todas estas páginas tienen el header y el footer en común. El header contiene la navegación que enlaza con todas las páginas excepto la de detalle. Y el footer contiene enlaces vacíos hacia las políticas y recursos legales.

También, se añadió una funcionalidad con JS para que se pueda cambiar el diseño a tema oscuro o claro, según la preferencia del usuario.

5. Diseño responsive, complejidad y estética

En el diseño, intenté seguir un estilo minimalista con imágenes redondeadas y utilicé tres colores (blanco, negro y verde en el tema claro y negro, gris y verde en el tema oscuro). Implementé estilos CSS utilizando flexbox, grid para crear un diseño responsive que se adapte a diferentes tamaños de pantalla y, además, utilicé media queries para aplicar estilos específicos según el tamaño del dispositivo.

```
/*SOLO TABLET*/
@media (min-width: 768px) and (max-width: 1024px) {
  /*PORTADA*/
  .home-principal {
    height: 300px;
  }
  .home-principal h1 {
    font-size: 25px;
  }
  .home-principal h3 {
    font-size: 15px;
  }
  .albumes {
    grid-template-columns: repeat(2, 1fr);
  }
  .noticias {
    padding-top: 30px;
  }       You, 17 minutes ago • terminados los estilos
}

/*SOLO MÓVIL*/
@media (max-width: 767px) {
  /*PORTADA*/
  .home-principal {
    height: 250px;
  }
  .home-principal h1 {
    font-size: 14px;
  }
  .home-principal h3 {
    font-size: 8px;
  }
  .albumes {
    grid-template-columns: repeat(1, 1fr);
  }
```

6. Semántica y accesibilidad

Para la semántica y accesibilidad del sitio, he procurado que todo el HTML tenga una estructura semántica adecuada utilizando etiquetas HTML apropiadas como `<header>, <nav>, <main>, <article>`, etc.

Además, en las imágenes he utilizado atributos alt para mejorar su accesibilidad.

7. Publicación a internet

Para la publicación en internet del proyecto, utilicé la plataforma Netlify. Vinculé mi cuenta de GitHub con esta y añadí mi directorio para poder subirlo y obtener una url proporcionada por Netlify.