Repositorio GitHub
https://github.com/AlbertViSi/PAC2Eines
Netlify
https://app.netlify.com/sites/pac2/overview

Preparacion entorno

Mediante el comando de git clone se realiza un clon de UOC boilerplate (https://github.com/uoc-advanced-html-css/uoc-boilerplate). A continuación se realiza otro clon de un repositorio vacío donde se subirá el proyecto para, finalmente, copiar los archivos del clon de UOC boilerplate, salvo el archivo .git, para moverlo a la carpeta del clon del repositorio vacío.
Una vez preparado lo anterior se realiza, mediante el comando “npm install”, la instalación de los node_modules del boilerplate, a continuacion “npm init stylelint” y preparar la configuracion de Stylelint, basada en la configuracion recomendada y las reglas extras de no dejar bloques vacíos y los selectores con patrones de letras y numeros.
Finalmente queda añadir “<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">” para añadir font awesome y ya estaria el trabajo previo finalizado.

Desarrollo
Se ha empezado toqueteando el código básico de bloques de cabecera, body y pie de página, se ha modificado el código SCSS para que esta tenga el formato básico.
Para la cabecera se ha creado otro preset similar al del pie de página, de la misma forma se ha creado otro preset para las información de contacto. Al mismo tiempo se ha creado otro preset para el encabezado, compartido en todas las páginas.
Respecto a los grids para la versión móvil se ha decidido convertir todos esos a un grid de 1 sola columna, permitiendo así que, de ser el caso que se contenga un grid ordenado de múltiples filas y columnas no se verá tan afectado el código.
En todo momento el código se ha desarrollado primero pensando en ordenador de sobremesa, pero manteniendo una atención y convirtiendo

los elementos que no quedaban correctamente adaptados para móvil a posterior.

Ordenamiento clases SCSS

Para el estilo de CSS se ha decidido ordenar las clases usando un ordenamiento alfabético para mejor facilidad a la hora de encontrar una clase cuando esa deba ser modificada para, por ejemplo, aplicar los criterios para que sea responsive, con la generación de subclases para las clases específicas de puntos especiales concretos (un grid muy específico por ejemplo), y luego clases muy generales para normas que puedan tener usos en múltiples partes de las diferentes páginas (por ejemplo grow: 1;).

Se había planteado también ordenar por página HTML en la que está cada clase, pero al evaluarse eso genera problemas de posicionamiento en clases que se encuentren en múltiples páginas, así como en clases que puedan estar dentro de elementos como el :is() quedando la duda de donde estarían, por lo que se descartó.

También cabe mencionar que se han mantenido algunos nombres del boilerplate, por ejemplo uoc-title y se ha seguido un poco con ese modelo de nombramiento por simple comodidad, pero las propiedades que venían por defecto se han modificado según ha ido conveniendo.
Los elementos que empiecen por :is() están situados al final del documento de scss. Si un elemento empieza por otro nombre pero contiene el :is(), por ejemplo ".example :is(ex1, ex2)", este elemento estaría junto a los demás .example

Body index (Página principal)

Para la portada, la versión con soporte para grid se ha ejecutado un layout más bien simple con múltiples filas y columnas y se ha puesto el texto pertinente en la fila y columna que interesaba.
Para la versión sin grid se ha optado por un simple desplazamiento del primer elemento. Para esta versión la idea principal era crear una tabla,

pero las tablas són, no solo muy rígidas, sino que además no se recomiendan, por lo que se ha descartado la idea y se ha optado por un sistema más simple pero con un resultado final similar a la versión con el grid.

Body pageone (Quienes Somos?)
Para los integrantes se ha usado un sistema similar que anteriormente había realizado, creando elementos dentro de un flex para añadir los integrantes, los cuales serían los ítems del flex.
El código no es nada complicado, quizás lo más extraño podría ser el hecho de añadir varios ítems vacíos al final, esto se ha hecho por una simple razón, se ha creado el flex para que este ocupe todo el tamaño disponible vacío, esto provoca que la última fila del flex sea absurdamente grande, por lo que se deben añadir elementos fantasmas para que estos ocupen el espacio disponible y no provoquen que el último sea muy grande, pero a la vez, al ser elementos fantasmas estos no tienen altura, siendo prácticamente indetectables.
Para los encargados se han usado personajes ficticios, es evidente, y el texto se ha generado mediante IA, aunque se ha modificado ligeramente la base ha sido mediante IA.

Body pagetwo (Que ofrecemos?)
El texto generado ha sido generado por IA.
El diseño en si es muy simple, una imagen al final de la explicación para amenizar un poco la página y una imagen inicial con el título, siendo esta como fondo de pantalla pero sin ser muy invasiva. Esto ha permitido el uso del comando :has() aunque siendo sinceros se podría haber solucionado con una clase, aunque ha quedado bien y no se ha necesitado añadir una clase completa para esto en esta ocasión.
El uso del comando :is() / :where(), debido a la simplicidad de esta página, no he encontrado una forma lógica de añadirlo, pero por otro lado se ha añadido en los pie y cabecera y también en la página de la portada para compensar, siendo ahí mucho más lógico y útil de añadir ahorrando scss que se podría considerar duplicado.

Body pagethree (Eventos)
La información de esta última página se ha generado con IA.

Los eventos se han catalogado en cajas que se les ha dado diferentes colores con el metodo de HSL. Se ha escogido que los colores sean muy claros (aumentando el LIGHTNESS) para que no sea tan agresivo a la vista. Se ha escogido que sean un conjunto de tres colores que se van repitiendo.
Finalmente se ha creado una pequeña animación, creando cajas que se expanden cuando pasas el cursor por encima para poder tener la información colapsada y, si al visitante le interesa, poder ver los detalles del curso particular.
En un principio se había planteado que se pudiera abrir al hacer click, pero no se ha podido configurar, y al usar focus no reaccionaba, por lo que se ha terminado descartando.

Algunos problemas con el Deploy en Netlify
Durante el deploy ha habido problemas debido a que, algunas de las imágenes en el código tenían el código con su nombre en minúsculas, mientras que las imágenes tenían el nombre en mayúsculas. Eso ha generado un problema que no he detectado durante las pruebas con el npm run dev, pero ha sido relativamente sencillo de solucionar, aunque ha tomado un poco de tiempo al tener que hacer más veces commit.