**1- Explica los principales pros y contras de Grid, Flex y floats en cuanto al maquetado de layouts web. Indica para qué es mejor cada una de estas técnicas.**

PROS

* Grid Layout: Hace lo mismo que Flex Layout pero aparte es bidireccional, es decir que podemos alinear nuestros componentes vertical y horizontalmente adaptándose a el ancho de nuestra pantalla.
* Flex Layout: Podemos alinear los componentes tanto en filas o columnas más fácil que con antiguas propiedades como float o position y además se adaptan al tamaño de pantalla.
* Floats: En su día fue la primera forma para adaptar nuestra maquetación web y alinear componentes en columnas, pero actualmente ya casi no se usa con la llegada de las otras dos alternativas.

CONTRAS

* Grid Layout: Quizás funciona peor al alinear solo en una dirección.
* Flex Layout: Solo es unidireccional, solo se puede alinear o en filas o columnas, pero no en ambas.
* Floats: es una manera muy rudimentaria de alinear componentes en columnas que cualquier de las otras dos maneras resuelve mejor y más sencillo de utilizar.

**2- Explica en breves palabras qué es el diseño responsivo (responsive design) y enumera al menos cuatro características y/o técnicas que ayudan a realizar un diseño responsivo. Explica brevemente qué se entiende por “mobile first”.**

El diseño web responsive es capaz de adaptarse a cualquier tipo de pantalla desde donde se visualice la página web, desde imágenes, textos y cualquier componente de la web y todo esto programado en un solo sitio web, no tenemos dos versiones como se hacía antiguamente.

CARACTERISTICAS:

* Único desarrollo web
* Layouts, componentes y imágenes fluidas que se adaptan
* Reduce tiempo de desarrollo
* Mas fácil de entender nuestro código al ser más limpio y una única versión
* Evita contenidos duplicados

TECNICAS:

* Utilizando Media Querys
* Implementando algún framework como Bootstrap

**MOBILE FIRST:**

Es el nuevo paso en el diseño web, se refiere a que tengas en cuenta primero el diseño para móviles antes que los de escritorio ya que todo está evolucionando hacia los dispositivos móviles.

Una vez diseñado para móviles seria escalarla y aumentar el tamaño para cualquier otro dispositivo como tablets o escritorio.

Sería el paso inverso a responsive web design.

**Explicación de la estructura de carpetas y Diseño**

La web consta de una estructura de 3 carpetas:

-CSS: se almacenarán todos los documentos css de nuestra web.

-IMG: se almacenarán todas las imágenes de nuestra web.

-Objetivos: Se almacenarán todas las templates html de cada objetivo

La web contiene todos los html de cada pagina que aparece en el menú al mismo nivel que el index.html.

En cuento al diseño, es una web responsive basada en flex y grid Box juntos con mediaquerys con 4 tamaños, donde se va adaptando según los pixeles de pantalla.

Desktop: > 990 px

MD: >850px && < 990px

SM: >767px && < 850px

XS: < 767px

**Explicación de las entidades HTML y CSS utilizadas**

1. Sobre las entidades HTML: debe explicarse el uso de las etiquetas elegidas.

He intentado utilizar las etiquetas más modernas de HTML5 que como estudiamos en la parte teórica son más semánticas y se entienden mejor como, por ejemplo:  
**<header>:** para definir la cabecera de la web o cabeceras de artículos etc..  
**<footer>:** para definir el pie de la web

También he utilizado otras que son más comunes:

**<h1>**-**<h3>** para títulos, listas.

**<ul>** para listas.

**<span>** para etiquetas.

**<div>** para separar contenidos.

**<p>** para crear parrafos con contenido.

**<table>** para crear una tabla con datos

**<form>** para crear un formulario.

**<section**> para secciones de la web claramente diferenciadas

**<article>** en este caso para almacenar los objetivos de la web

1. Sobre el CSS: debe explicarse todo el CSS utilizado, incluyendo cuando se ha utilizado y por qué y para qué se aplican los selectores y propiedades definidos.

:root {  
 --main-bg-color: #306291;  
 --dark-bg-color: rgba(69, 141, 209, 0.5);  
 --light-bg-color: rgba(69, 141, 209, 0.2);  
 --text-color: #1f1f11;  
 --text-light-color: #666666;  
 --contrast-color: #D9645D;  
 --header-font: "Roboto Slab";  
 --main-font: "Roboto";  
}

Lo utilizo para declarar unas “variables” y asignarles un valor que permanecerá en todo el documento y se podrá aplicar a cualquier propiedad con el uso de var().

header > .flex-container{  
 display: flex;  
 flex-direction: row;  
 align-items: center;  
 justify-content: center;  
}

Lo utilizo para asignarle al header de la web un flexBox del 100% en una row centrando todos sus elementos.

.flex-item {  
 width: auto;  
 margin: 10px;  
 text-align: center;  
 line-height: 75px;  
 font-size: 15px;  
}

Lo utilizo para asignarle a todos los elementos dentro de un flex box la anchura que el mismo ocupe centrarlo y darle algunas propiedades de diseño.

.responsive{  
 display: none;  
}

Lo utilizo para ocultar los elementos que contengan esta clase en versión responsive.

#content > .flex-container{  
 display: flex;  
 flex-direction: row;  
 align-items: start;  
 justify-content: center;  
}

#content > .flex-container > .flex-item{  
 width: 60%;  
 line-height: 25px;  
}

Utilizo todas estas clases para definir como serán los estilos del espacio donde ira el contenido del index de la web con un flexbox en modo file centrando sus elelementos y dándole propiedades de diseño.

footer{  
 width: 100%;  
 font-size: 16px;  
 background-color: var(--light-bg-color);  
 color: var(--text-light-color);  
 border-top: 6px solid transparent;  
 border-image: linear-gradient(to right, #AFD8FB 0%, var(--main-bg-color) 100%);  
 border-image-slice: 1;  
}

Lo utilizo para darle un estilo al footer de la web como color de texto y de fondo y un borde superior con degradado.

#content > .flex-container > .sidebar{  
 display: flex;  
 flex-direction: column;  
 width:30%;  
}

#content > .flex-container > .sidebar >img{  
 width:inherit;  
 margin: auto;  
 margin-left:10%;  
}

.sidebarMenu {  
 margin-left:10%;  
 padding-top: 20px;  
}

.sidebarMenu a {  
 padding-bottom: 10px;  
 text-decoration: none;  
 font-size: 20px;  
 color: black;  
 display: block;  
}

Utilizo todas estas clases para asignarle al sidebar que se coloca a la izquierda de la web un flexbox en modo columna y darle propiedades de diseño a todos sus elementos.

#content > .container > .grid-container{  
 display: grid;  
 grid-template-columns: repeat(3,33.33%);  
 grid-row-gap:25px;  
 padding-bottom: 30px;  
}

#content > .container > .grid-container > a > img{  
 width: 100%;  
}

#content > .container > .grid-container > .titular {  
 padding-left: 20px;  
}

#content > .container > .grid-container > .titular > a {  
 text-decoration: none;  
 color: black;  
}

#content > .container > .grid-container > .titular > a >h2{  
 text-align: left;  
}

#content > .container > .grid-container > .titular > p{  
 text-align: justify;  
}

#content > .container > .grid-container > .titular > p > a.info{  
 text-decoration: underline;  
 color:blue;  
 text-align: left;  
}

Utilizo todas estas clases para definir el grid layout de los objetivos y darles propiedades de diseño a todos sus elementos

form > fieldset{  
 background-color:var(--light-bg-color);  
 border-radius: 10px;  
 border: none;  
 padding: 30px;  
 margin-bottom: 30px;  
}

form > fieldset > p{  
 padding: 5px;  
}

form > fieldset > i{  
 color: var(--text-light-color);  
}

form > fieldset > p > label{  
 width:40%;  
 display: inline-block;  
}

form > fieldset > p > input,form > fieldset > p > select{  
 border: 1px solid var(--dark-bg-color);  
 width:50%;  
 height: 25px;  
 border-radius: 3px;  
 outline: none;  
}

form > fieldset > p > input:focus,  
form > fieldset > p > select:focus{  
 border: 1px solid var(--contrast-color);  
}

form > fieldset > fieldset{  
 width:60%;  
 padding: 10px;  
 border: 1px solid var(--dark-bg-color);  
 border-radius: 5px;  
}

.fieldsetNumber{  
 color: var(--main-bg-color);  
}

.sendButton{  
 font-size: 22px;  
 background-color: var(--contrast-color);  
 color:white;  
 text-align: center;  
 padding: 10px;  
 width: 35%;  
 display: block;  
 margin: auto;  
 margin-top: 20px;  
 border-radius: 5px;  
}

Utilizo todas estas propiedades para darle estilo al formulario como que tenga border redondeados, color de fondo, color de texto etc..

ul.breadcrumb {  
 padding: 10px 16px;  
 list-style: none;  
 background-color: #eee;  
 margin-bottom: 20px;  
}

ul.breadcrumb li {  
 display: inline;  
 font-size: 18px;  
}

ul.breadcrumb li+li:before {  
 padding: 8px;  
 color: black;  
 content: "/\00a0";  
}

ul.breadcrumb li a {  
 color: #0275d8;  
 text-decoration: none;  
}

ul.breadcrumb li a:hover {  
 color: #01447e;  
}

Utilizo todas estas clases para crear el diseño del breadcrumb de la web para seguir las rutas por donde va pasando el usuario.

footer {  
 background-color: #222;  
 color: #fff;  
}

footer > .super-flex-container {  
 display: flex;  
 flex-direction: row;  
 align-items: center;  
 justify-content: center;  
}

footer > .flex-container{  
 display: flex;  
 flex-direction: row;  
 justify-content: flex-start;  
}

footer > .super-flex-container > .flex-container-left{  
 display: flex;  
 flex-direction: row;  
 width: 50%;  
 justify-content: flex-start;  
}

footer > .super-flex-container > .flex-container-right{  
 display: flex;  
 flex-direction: row;  
 width: 50%;  
 justify-content: flex-end;  
}

footer > .super-flex-container > .flex-container-left > .flex-item >a{  
 text-decoration: none;  
 color:White  
}

footer > .super-flex-container > .flex-container-right > .flex-item > a{  
 text-decoration: none;  
 color:White  
}

.go-top {

padding-top: 5px;  
 cursor: pointer;  
 left: 5%;  
 bottom: 0;  
 width: 40px;  
 height: 30px;  
 background: #563d7c;  
 position: fixed;  
 z-index: 2;  
 text-align: center;  
}

Estos son todas las clases que utilizo para darle un estilo al footer y que todos sus elementos se coloquen en un flex box en estilo “fila”

/\* md \*/

@media (max-width: 990px) {}

Utilizo esta media query para cambiar el diseño si la web se visualiza en dispositivos más pequeños de 990px y más grande de 850px

/\* sm \*/

@media (max-width: 850px) {}

Utilizo esta media query para cambiar el diseño si la web se visualiza en dispositivos más pequeños de 850 y más grande de 767px

/\* xs \*/

@media (max-width: 767px) {}

Utilizo esta media query para cambiar el diseño si la web se visualiza en dispositivos más pequeños 767px