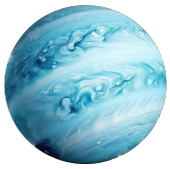


Nebula

Presentación del sistema para la gestión de proyectos.

Link al proyecto

Rodriguez Blanco, Begoña

9 - 12 Diseño Web HTML
SEV Enero 2025 - Tutor: Gustavo Mesa

Información técnica del proyecto

Código:

URL Github https://github.com/BegoRB/proyecto-web-bego-rodriguez.git

URL del dominio: https://berobla.com/

Diseño:

URLs del diseño en Figma: https://www.figma.com/design/U6jGaYu0y7GhQOse7Iu115/Untitled?node-id=0-1&t=7tNtPGhFgpjZebMr-1

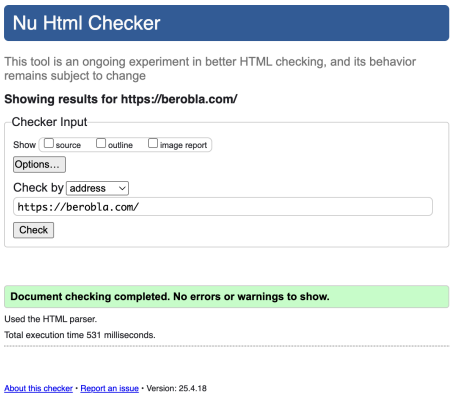
Tecnologías usadas:

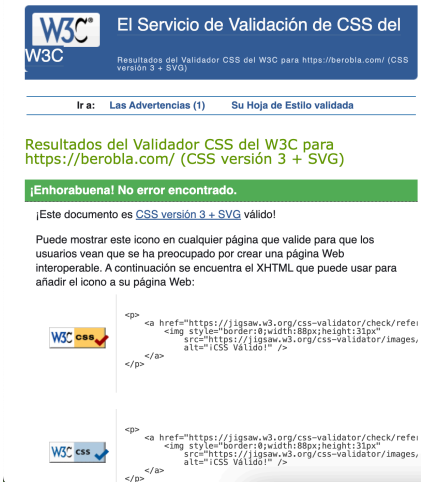
El proyecto ha sido desarrollado en base a las lecciones impartidas en el curso de Diseño Web:

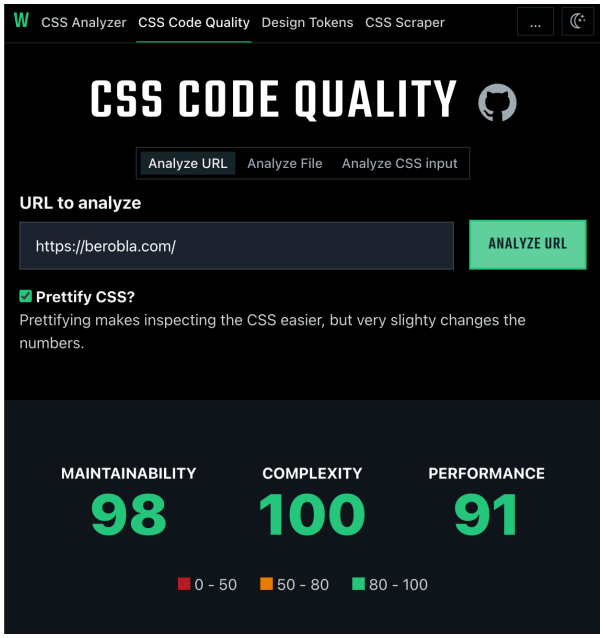
· HTML
· CSS
· Javascript

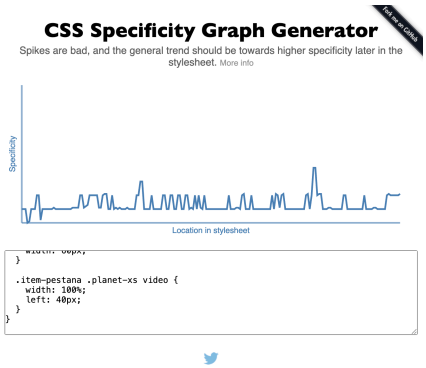
Comprobaciones realizadas

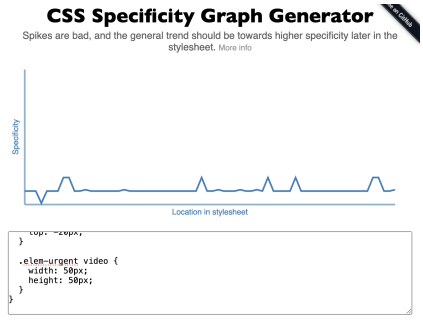
· Semántica con Validator HTML
· Compatibilidad con Validator CSS
· Calidad del CSS con Project Wallace
· Especificidad con CSS Specify Graph
· Rendimiento con Lighthouse

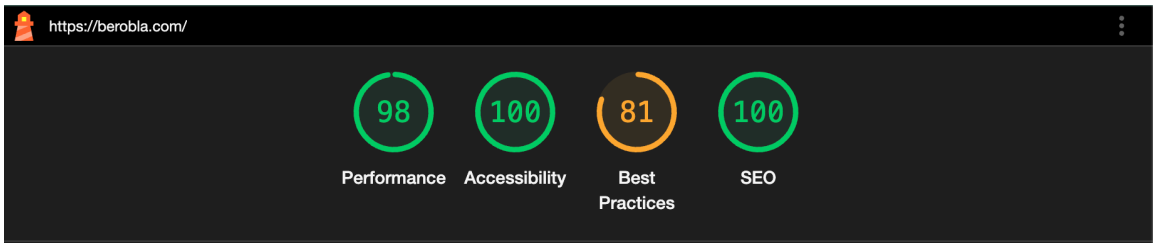












Introducción

Nebula es un proyecto destinado a la gestión de proyectos.

Este proyecto se creó a causa de la gran cantidad de personas afectadas por el exceso de responsabilidades diarias en su trabajo y la falta de herramientas que ayuden a minimizar esa presión .

La mayoría de las herramientas tradicionales presentan la información de forma plana y poco visual, lo que termina siendo más una carga que una ayuda.

En general, una persona puede rendir mejor cuando sus tareas no se presentan como bloques de texto en el centro de su planificación. Este proyecto surge con la idea de ofrecer una alternativa más ligera y visual.

En lugar de mostrar listas infinitas de texto, se utiliza una metáfora espacial donde las tareas orbitan alrededor del objetivo final, lo que permite priorizar de forma gamificada, reduciendo la sensación de carga mental.

Entre sus principales funcionalidades, incluye la creación y organización de tareas, visualización de estas mediante sistemas de órbitas y un enfoque caracteristico para destacar las tareas urgentes.

El proyecto enfoca principalmente la presentación de dicha plataforma a través de una demo a la cual podremos acceder en el footer de la web.

Link al proyecto

Descripción del proyecto

Nebula es una plataforma de gestión de tareas, creada para resolver el caos y el estrés del día a día en el mundo laboral. Inspirada en la frase “La galaxia es inmensa, pero nunca caótica”, mostramos el principal objetivo, ayudar a los usuarios a ordenar ese desorden y organizar su trabajo de manera más visual y tranquila.

Se ha desarrollado con HTML y CSS, JavaScript, permitiendo gestionar las tareas de forma intuitiva, donde orbitan alrededor del objetivo central, dándole una sensación de claridad y orden.

Secciones Principales:

Página de Inicio: La página inicial intenta demostrar como Nebula es capaz de armonizar el espacio, a través de 3 frames principales.
Se ha querido mostrar la idea principal, las órbitas.

Contenido detallado: Aquí se explica cómo funciona Nebula, profundizando en las funcionalidades. Mostramos cuáles son sus elementos principales y cómo se pueden asignar prioridades.

Demo: Tras la presentación inicial, los usuarios pueden explorar Nebula a través de una demo interactiva, ubicada en el footer de cada página. Esta demo permite experimentar en tiempo real cómo se gestionan y organizan las tareas.

Gestión de Tareas: Los usuarios pueden añadir, editar y eliminar tareas en la tabla de contenido ubicada en esta página.

Funcionalidades Principales:

Visualización de Tareas: Las tareas se representan de manera visual como planetas que orbitan alrededor del sol, representando así el objetivo final.

Enfoque en Tareas Urgentes Las tareas urgentes se destacan en el sistema mediante un elemento posicionado cerca de su tarea “padre”, personalizado con un movimiento de latido, permitiendo que el usuario identifique rápidamente y la resuelva antes que otras.

Sistema de Edición El sistema permite crear, eliminar y modificar tareas, cambiando sus fechas limite, su prioridad (Urgente / No urgente) y estado actual (En curso / Pendiente / Finalizado).

Tipografías

Tipografía primaria / Títulos: Lato
Tipografía secundaria / Párrafos: Poppins

Tamaño extra grande: 3rem
Tamaño grande: 2rem
Tamaño medio: 1.5rem
Tamaño pequeño: 1.25rem
Tamaño base: 1rem
Tamaño extra pequeño: 0.75rem

Pesos de fuente:

Bold: 700
Semibold: 600
Regular: 400
Light: 300

Colores de texto:

Texto base: Negro (rgb(0, 0, 0))
Texto claro: Blanco (rgb(255, 255, 255))

Botones

Botón añadir btn--add :
Color de fondo: #826270
Hover: #8f375f

Botón prioridad btn--priority.select-prioridad:
Color de fondo: #f1f1f1
Color del texto: #535252

Botón estado btn--priority.select-estado:
Color de fondo: #dff1ff
Color del texto: #00576f

H1 Texto Galaxia Bold

H1 Texto Galaxia SemiBold

H2 Texto Galaxia Bold

H2 Texto Galaxia SemiBold

H3 Texto Galaxia Bold

H3 Texto Galaxia SemiBold

H3 Texto Galaxia SemiBold

Párrafo Galaxia Bold

Párrafo Galaxia SemiBold

Texto pequeño Galaxia Bold

Texto pequeño SemiBold

btn—add

select-prioridad

urge | no urge

select-estado

en curso | finalizado | pendiente

“Urge”: fondo #ffc6d5, texto #660b23
“No urge”: fondo #f1f1f1, texto #535252

“En curso”: fondo #dff1ff, texto #00576f
“Finalizado”: fondo #c9ffd9, texto #0c441d
“Pendiente”: fondo #ffddf4, texto #440c26

Link de Figma para ver el proceso de creación

Tecnologías y Metodologías

Durante el desarrollo del proyecto utilicé **Visual Studio Code** como editor, acompañado de varios plugin:

- **Prettier**, código limpio y consistente.
- **Live Server**, visualizar los cambios.
- **BEM Helper**, para la metodología BEM.
- **px to rem & rpx & vw (cssrem),** atajo para convertir unidades (px -> rem).

Mi experiencia personal

Una de las partes más difíciles fue llegar a las conclusiones para crear ciertos eventos y condicionales que implicaban varios elementos a la vez. Tenía muy clara la idea del resultado que quería lograr, pero no sabía cómo plasmarlo desde cero, en una pantalla en blanco.

Empecé a crear wireframes de la idea principal, pasé el diseño a Figma y a partir de ahí fui construyendo las conexiones necesarias que en mi mente tenían sentido para hacer las condicionales. Relacioné los elementos en papel, probé ideas, y corregí cuando algo no funcionaba como esperaba.
Uno de los desafíos más grandes fue hacer que funcionara bien la función para añadir una tarea nueva; finalmente, tuve que pedir ayuda en las tutorías para solucionarlo.

Descripción del problema:
Con la estructura base, podemos añadir una subtarea sin problema.
Al añadir una tarea principal nueva, se crea correctamente pero al querer añadir otra subtarea en el contenedor base, se nos generan dos subtareas en el contenedor base.

Si decidimos añadir otra tarea principal nueva, y volvemos a añadir una subtarea en el contenedor base, se generan tres subtareas nuevas en vez de una.
Y así sucesivamente: la cantidad de subtareas generadas en el contenedor base aumentaba en proporción al número de tareas principales creadas previamente.

El problema se ubicaba en el botón principal base, ya que le estaba llamando a la función repetidas veces.
Es decir, cada vez que se añadía una tarea al contenedor, se estaba asignando una nueva acción al botón base, lo que provocaba un bucle: cuantas más tareas se creaban, más veces se ejecutaba la acción, provocando que se anidasen múltiples tareas a la tabla base.

La solución era añadir un string con un número específico a cada tarea nueva que se crease, asi solo se ejecuta la acción principal en el botón base, la (1) : agregarEventoAddTarea(1).

Otra parte especialmente compleja fue conseguir que los planetas girasen en órbita alrededor del sol. Durante las clases me costaba entender el concepto de perspectivas 3D y los ejes X, Y y Z, y no sabía cómo llevar eso al código.

Tuve que investigar bastante por internet hasta que encontré un canal de YouTube (link en la bibliografía) que explicaba cómo funcionaban los movimientos circulares. A partir de ahí, pude entender cómo buscar la operación en código para rotar un elemento en círculo y cómo animarlo para que simulara el movimiento que buscaba.

Partimos de la base de las siguientes operaciones (todas las referencias en la bibliografía):

Observamos cómo deberíamos tener el ángulo del elemento que se desplazarà:
Math.cos(values[i] * Math.PI / 180)
Math.sin(values[i] * Math.PI / 180)

Sabemos que debemos multiplicar las coordenadas por la distancia del elemento central:
x = center.x + Math.cos(angle) * offset
y = center.y + Math.sin(angle) * offset

El ángulo debe estar en radianes, por lo que convertimos así:
radians = degrees * Math.PI / 180

Resultado (Usamos eje X y eje Z) :
const x = Math.cos(ángulo del planeta * Math.PI / 180) * distancia del sol;
const z = Math.sin(ángulo del planeta * Math.PI / 180) * distancia del sol;

Queremos aplicar 3D al elemento, al eje X y Z, para darle profundidad y movimiento con transform: translate;

elemento.style.transform = `translateX(${x}px) translateZ(${z}px)`;

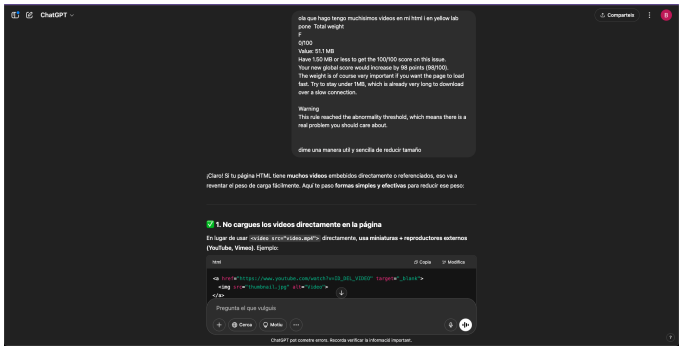
Además, la intención era que fuese a una velocidad determinada, unos 20 segundos por vuelta. Para ello debemos saber a cuantos frames funciona el método requestAnimation-Frame: (60 FPS).
Lo calculamos entonces de la siguiente manera:

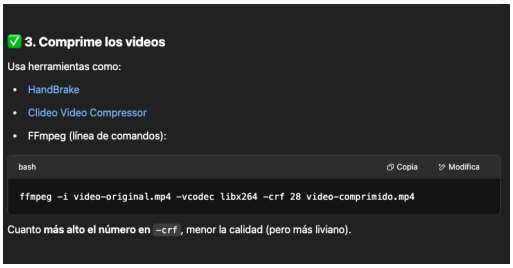
Girar 360° en 20 segundos (60 FPS por Segundo) => 360/20s = 18 / 60 fps = 0.3
ángulo del planeta += 0.3;

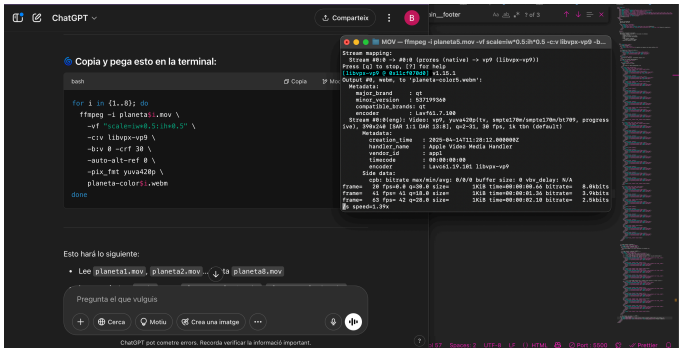
Aplicamos posteriormente diferentes ángulos a los planetas, para que no se solapen las capas.

Lo único en que me ayudó este problema, fue sentir que podría hacer cualquier otra cosa del proyecto, ya que me llevó muchos días conseguirlo y estoy muy orgullosa del resultado.

En las comprobaciones finales, tuve el problema de que los videos de los planetas ocupaban demasiado, **tuve que recurrir a la IA** para buscar una solución, ya que las plataformas online no me daban buen resultado y no tuve el tiempo necesario para aprender a hacerlo con algun programa de edición de video, asi que opté por usar el recurso de comprimir archivos mediante FFmpeg en la terminal de mi Mac.







Bibliografía

Link al proyecto

Documentación oficial

https://developer.mozilla.org/en-US/

Figma

https://www.figma.com/

Teoria y cálculo para senos y cosenos para crear órbitas:

- https://www.adammarcwilliams.co.uk/animating-javascript-trigonometry/
- https://stackoverflow.com/questions/32078052/using-math-pi-how-to-set-my-own-value-of-angles-in-each-element
- https://www.youtube.com/watch?v=ZnZHdta97K4&ab_channel=FlippingPhysics

Foro para media queries en JS

https://forum.elhacker.net/desarrollo-web/solucionado-porque-en-un-window-matchmedia-me-funciona-y-en-el-otro-no/

```
// !!MEDIA QUERIES!!
function actualizarDistancia() {
  if (window.matchMedia("(max-width:
768px)").matches) {
    // Pantalla pequeña
    distanciaDelSol = 150;
  } else {
    // Pantalla normal o grande
    distanciaDelSol = 280;
  }
```