



19-11-2025

Actividad 1

Visualización Interactivas con
datos JSON y D3.js



Leonard José Cuenca Roa
UNIR – UNIVERSIDAD RIOJA DE ESPAÑA

Contenido

Descripción Breve del Desarrollo	2
Desarrollo Frontend	2
Estructura Front.....	2
.....	2
Resultados de pantallas.....	3
Desarrollo Backend.....	4
Estructura Backend	5
.....	5
Interpretación de Datos	6
Insights Detectados.....	6
Análisis de Resultados (La Narrativa).....	6
Conclusiones.....	7

Descripción Breve del Desarrollo

La actividad actual tiene como objetivo demostrar las habilidades adquiridas en el desarrollo de gráficas, utilizando dos de las librerías principales del mercado: Google Charts y D3.js.

Para este proyecto, se generó una arquitectura modular basada en buenas prácticas de desarrollo, buscando fomentar una mayor flexibilidad y escalabilidad.

A continuación, se describirán los puntos más resaltantes del desarrollo. Sin caer en la retórica de explicar cada línea de código, se desglosará para mayor comprensión qué se realizó en el Frontend y en el Backend, detallando tanto el "qué" (la funcionalidad) como el "cómo" (la implementación).

Desarrollo Frontend

Para el desarrollo del Frontend y para su evaluación se mostrará una captura de pantalla que permita visualizar el resultado final:

Estructura Front

```
/mi-dashboard
  node_modules/      <-- Paquetes Funcionales para el ambiente Node.js
  package.json        <-- Lista de Paquetes para la actividad
  package-lock.json
  server.js          <-- Servidor Express (Backend)
  public              <-- Repositorio Principal (Presentación)
    index.html        <-- Plantilla html que permite la visualización
    app.js            <-- Script Interactivo (Orquestador Cliente)
    js
      google-chart.js <-- Lógica para graficar google Chart (Grafico 1)
      d3-chart.js     <-- Lógica para graficar la funcionalidad de grafica deburbuja (Grafico 2)
      util.js         <-- Lógica modular y flexible permitiendo tener cálculos, validaciones, llamados API, para mantener la modularidad y flexibilidad. (Utilidades)
    css
      global.css      <-- Archivo de limpieza CSS
      dash_ventas.css <-- Archivo para generar el estilo deseado visual para la actividad
```

En la imagen actual destaco la organización de los archivos CSS para el desarrollo visual del dashboard. Me apoyo en dos elementos clave:

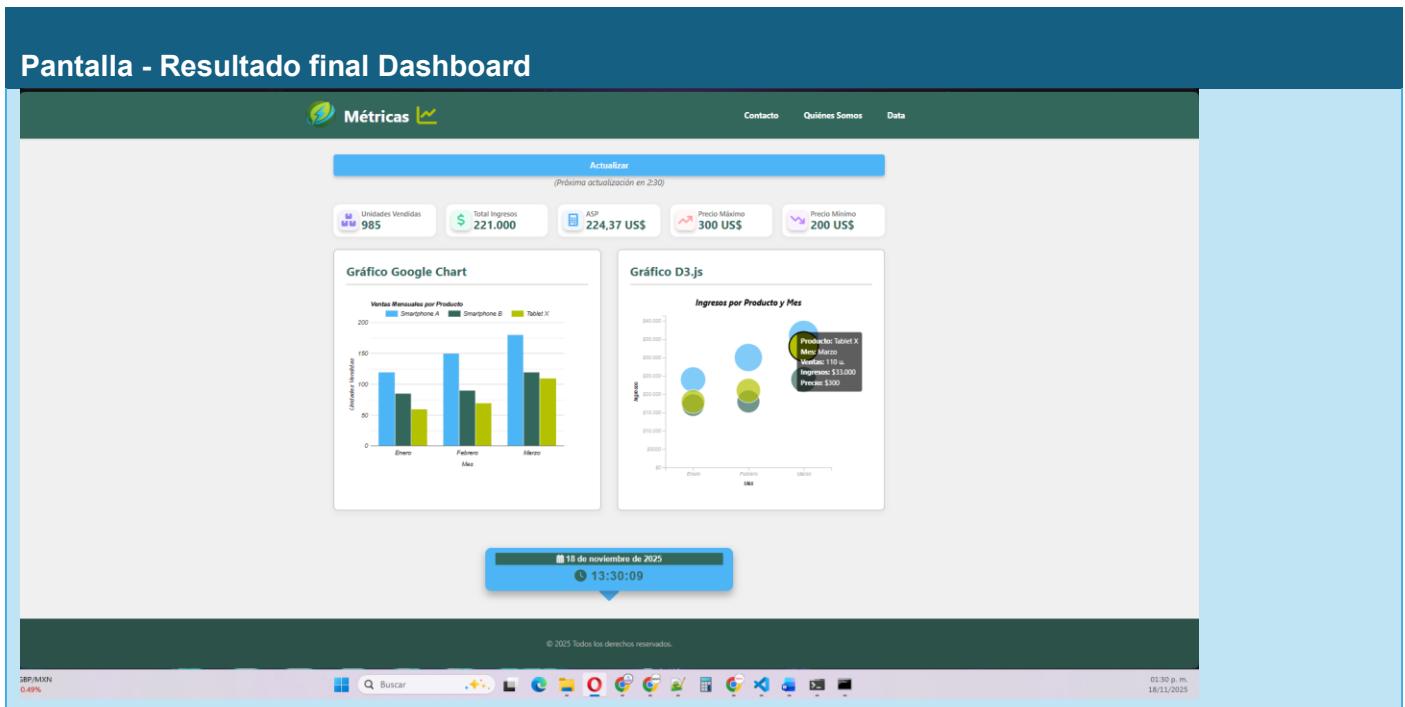
global.css: Este elemento me permite reiniciar las etiquetas HTML con el propósito de tener un control total sobre los estilos nativos, ya que los diferentes tipos de navegadores aplican estilos predeterminados. [\[consultar código fuente aquí\]](#)

dash_ventas.css: Puedo generar las reglas de cada estilo para la composición visual del dashboard. Aquí defino clases para estructurar los bloques HTML (header, body y footer), con el fin de mantener un código organizado y homogeneizado. Esto garantiza flexibilidad, escalabilidad y facilidad de mantenimiento, cumpliendo con las buenas prácticas de desarrollo y los requisitos de la actividad. [\[consultar código fuente aquí\]](#)

El index.html (La Presentación)

Sin dejar de lado el index.html, la joya de la corona. Este es el otro orquestador, donde se enlazan los diferentes métodos y llamadas a librerías para generar toda la magia visual, funcionando bajo un ambiente Cliente/Servidor. [\[consultar código fuente aquí\]](#)

Resultados de pantallas



Como se puede apreciar en la captura de pantalla, el desarrollo cumple con los requisitos descritos en la actividad, creando un espacio visual central que contiene dos gráficos clave:

- **Primer Gráfico (Barras):** Representa las ventas del primer trimestre del año, segmentando las ventas de los tres principales productos por mes. Esto permite una comparativa rápida y clara del rendimiento de cada producto a lo largo del trimestre.
- **Segundo Gráfico (Burbujas):** Se implementó para ofrecer interactividad. Al pasar el cursor (hover) sobre el área de cada burbuja, se logra visualizar el detalle específico de la información.

Funcionalidades Adicionales Implementadas:

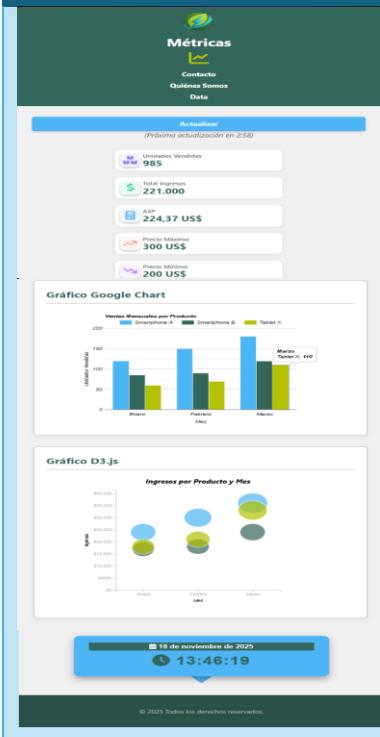
Se agregó un botón que permite la actualización manual de los gráficos en caso de que exista un nuevo valor en el archivo JSON.

Como valor agregado extra, se generó un contador de tiempo regresivo de 5 minutos. Al llegar a cero, el sistema realiza un refresco automático de las gráficas para mostrar los datos más recientes.

Se agregó también un resumen de métricas clave, una forma práctica de resaltar cantidades importantes. Para este caso, se resaltaron las siguientes métricas:

- Total de Unidades Vendidas
- Total de Ingresos
- ASP (Precio Promedio de Venta)
- Precio Máximo
- Precio Mínimo

Pantalla - Resultado Responsivo



Como se puede apreciar en la segunda imagen, se realiza la validación de que el dashboard también cumpla con su potencial de adaptabilidad o mejor conocido diseño responsive, ofreciendo la oportunidad de que el dashboard pueda ser consumido en dispositivos móviles como celulares o tabletas, sin perder la funcionalidad principal.

Desarrollo Backend

Para el desarrollo del Backend empleé la tecnología Node.js. Esta plataforma me permitió utilizar JavaScript de forma Full Stack, abarcando tanto el ambiente del servidor como el ambiente del cliente. Específicamente, utilicé el entorno del servidor para la validación del JSON y para el despliegue de un **API Endpoint** que será consumido desde el cliente. En el cliente, se desarrolló la interfaz de usuario y toda la lógica de presentación con HTML, CSS y JavaScript.

Estructura Backend

```
/mi-dashboard
  └── node_modules/      <-- Paquetes Funcionales para el ambiente Node.js
  ├── package.json        <-- Lista de Paquetes para la actividad
  └── package-lock.json
  └── server.js          <-- Servidor Express (Backend)
  └── public              <-- Repositorio Principal (Presentación)
    ├── index.html         <-- Plantilla html que permite la visualización
    ├── app.js              <-- Script Interactivo (Orquestador Cliente)
    └── /js
      └── google-chart.js  <-- Lógica para graficar google Chart (Grafico 1)
      └── d3-chart.js       <-- Lógica para graficar la funcionalidad de grafica de burbuja (Grafico 2)
      └── util.js           <-- Lógica modular y flexible permitiendo tener calculos, validaciones, llamados API, para mantener la modularidad y flexibilidad. (Utilidades)
    └── /css
      └── global.css        <-- Archivo de limpieza css
      └── dash_venta.css     <-- Archivo para generar el estilo deseado visual para la actividad
```

El server.js (Backend)

El archivo `server.js`, utilizando el paquete Express, es fundamental para configurar y crear el servidor que permite desplegar y operar todo el entorno de la aplicación. El archivo JSON con los datos se encuentra alojado en una sección del repositorio. Simplemente tomamos ese JSON, lo validamos, y luego generamos un API *Endpoint* para que los datos puedan ser consumidos por el *frontend*. [\[consultar código fuente aquí\]](#)

El app.js (Orquestador Cliente)

El archivo `app.js` actúa como el orquestador principal (*main*) del lado del cliente. Desde aquí puedo importar las librerías necesarias, así como mis propios módulos (*models*). Esto me permite cumplir con las bases de desarrollo para crear un *dashboard* que sea flexible, incremental y escalable, asegurando el uso de buenas prácticas y evitando el código espagueti. [\[consultar código fuente aquí\]](#)

Los Módulos de Gráficas (google-chart.js y d3-chart.js)

Generé módulos separados que se almacenan en el repositorio JS. Uno, llamado `google-chart.js`, contiene la lógica para mostrar la gráfica de barras. El otro archivo, `d3-chart.js`, contiene la lógica para mostrar las gráficas de burbuja (utilizando la librería D3.js). [\[consultar código fuente aquí\]](#)

El Módulo de Utilidades (util.js)

También generé el módulo `util.js`. Este archivo contiene los métodos que me permiten consumir el API *Endpoint* para obtener los datos. Posteriormente, estos datos se envían como parámetros a los destinos necesarios, que en este caso son los módulos para el gráfico de barras y el gráfico de burbujas. Además, incluye métodos de validación y cálculos de totales. [\[consultar código fuente aquí\]](#)

El index.html (La Presentación)

Sin dejar de lado el `index.html`, la joya de la corona. Este es el otro orquestador, donde se enlazan los diferentes métodos y llamadas a librerías para generar toda la magia visual, funcionando bajo un ambiente Cliente/Servidor. [\[consultar código fuente aquí\]](#)

Interpretación de Datos

Para este análisis, se generó un dashboard que mide el rendimiento de ventas del primer trimestre Enero, febrero, marzo. Se desea evaluar el éxito, por lo que se realizó un planteamiento hipotético todo con el fin educativo de dar una narrativa a los datos y poder aplicar el conocimiento técnico que es el uso de las gráficas.

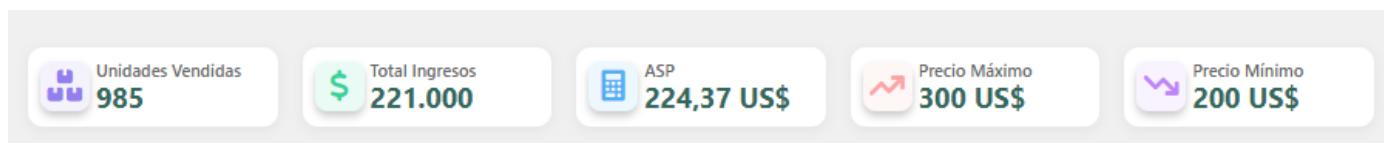
Insights Detectados

Se estableció dos objetivos (KPIs) supuestos para este trimestre:

- **Objetivo de Ingresos:** Alcanzar una meta de ingresos totales de \$125.000.
- **Objetivo de Eficiencia:** Mantener un Ingreso Promedio por Venta (ASP) superior a \$220.

Análisis de Resultados (La Narrativa)

Los resultados del primer trimestre son muy positivos y superan ambos objetivos supuestos. **Se alcanzó un total de ingresos de \$128.000**, superando nuestra meta de **\$125.000**. Esto se logró a través de un volumen total de **575 unidades vendidas**. Más importante aún, nuestro **ASP (Ingreso Promedio por Venta)** se situó en **\$222,61 US\$**. Esto no solo supera nuestro objetivo de **\$220**, sino que también nos indica que estamos logrando una mezcla de productos saludable y constantes que la mayoría de nuestros clientes tienen la confianza de adquirir por su excelente calidad, vendiendo eficientemente nuestros artículos de alta y media gama.



Las gráficas de Google Chart y D3.js nos muestran cómo logramos esos números, el producto estrella **Smartphone A**. El Gráfico **Google Chart (Ventas Mensuales)** muestra claramente que el **Smartphone A** es nuestro producto líder en volumen.



El gráfico de burbujas (**Ingresos**) de D3.js confirma que también es nuestro mayor generador de ingresos. Se observó un crecimiento de enero a febrero, impulsado principalmente por un aumento en las ventas del **Smartphone A** y una ligera subida en la **Tablet X**.

La Importancia de esta mezcla de productos es que se tiene un precio máximo y un mínimo, aunque la **Tablet X** es nuestro precio máximo de \$300 es la que menos unidades vende, su contribución es vital para mantener nuestro **ASP** por encima del objetivo. El **Smartphone B** es el precio mínimo de \$200 actúa como una base de volumen estable.

Conclusiones

En conclusión, el análisis realizado con el dashboard, basado en datos hipotéticos del primer trimestre de ventas, y que tienen fines de prueba, demuestra un resultado exitoso. No solo se cumplió la meta de ingresos brutos establecida, sino que se logró con eficiencia al **superar el objetivo** de Precio Promedio de Venta (ASP). El **Smartphone A** se consolida como el motor del crecimiento, mientras que la **Tablet X** es clave para mantener una rentabilidad promedio

Adicionalmente, proponemos llevar a cabo estudios complementarios clave, como encuestas de satisfacción del cliente, análisis de sentimiento de la marca y sus productos, y una validación exhaustiva de la actividad en redes sociales. El objetivo es identificar con precisión las áreas de mejora para estar mejor preparados en el **próximo trimestre** y conseguir **un aumento en las ventas** de aquellos productos con bajo desempeño. Todo este trabajo se abordará con un enfoque riguroso, listo para aplicar el conocimiento adquirido y generar una ventaja competitiva.

Sin duda, saber emplear estos conocimientos técnicos para construir un ambiente **CLIENTE/SERVIDOR**, utilizando **HTML, CSS y JAVASCRIPT**, es fundamental para diseñar dashboard visuales e interactivos. Esta base técnica no solo facilita la interacción con los datos, sino que también es un elemento esencial para lograr una mayor precisión al **crear la narrativa** y la **historia de los datos** mejor conocido del **English Data Storytelling**. Como se enfatiza en esta unidad, una imagen bien definida, presentada y aterrizada, tiene más valor que mil palabras.