INVESTIGACION CHART JS

JOSSE FREDDY REYES, JEAN CARLOS ARAUJO, JUAN SEBASTIAN GRANOBLES

QUE ES CHART JS

https://es.wikipedia.org/wiki/Chart.js

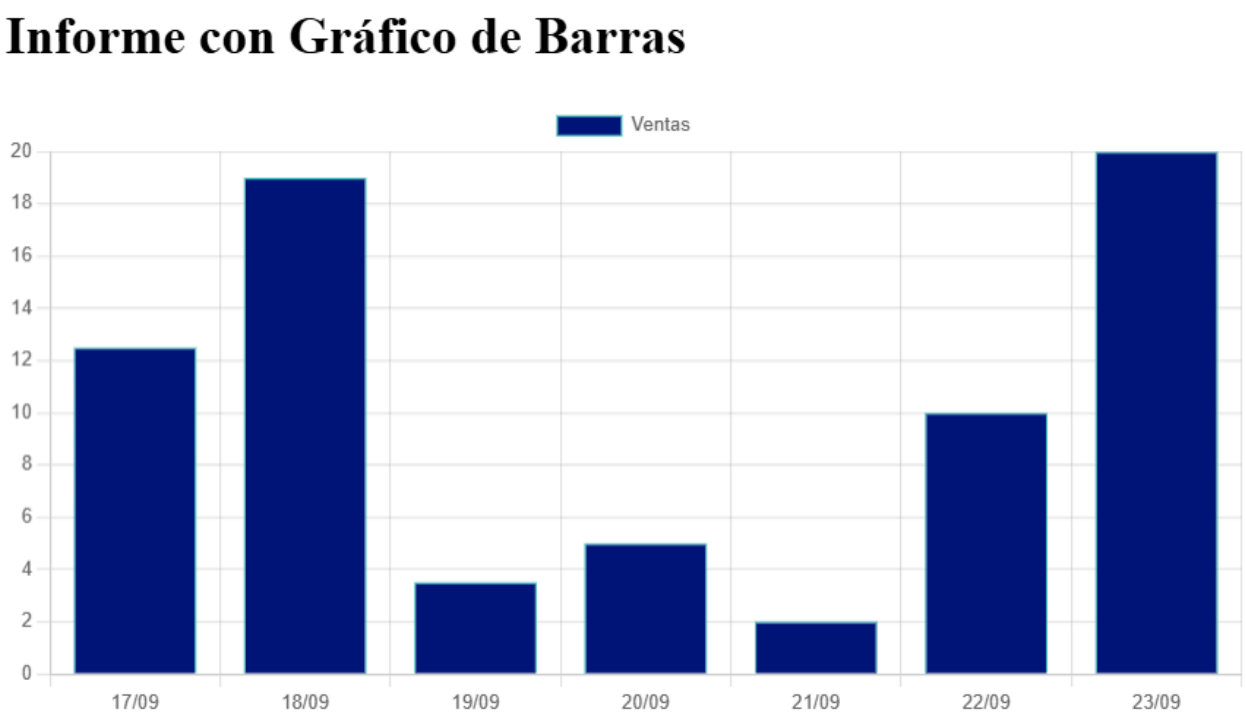
Chart.js es una biblioteca JavaScript gratuita de código abierto para la visualización de datos, Creado por el desarrollador web con sede en Londres **Nick Downie** en **2013**, ahora es mantenido por la comunidad y es la segunda biblioteca de gráficos JS más popular en GitHub por la cantidad de estrellas después de **D3.js**, considerada significativamente más fácil de usar aunque menos personalizable que esta; **Chart.js** admite 8 tipos de gráficos

1. gráficos de barra/
https://es.stackoverflow.com/questions/604033/crear-grafico-de-barras-con-chart-js

```
 <!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Informe con Gráfico de Barras</title>
    <!-- Incluir Chart.js -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <h1>Informe con Gráfico de Barras</h1>

    <!-- Agregar un contenedor para el gráfico -->
    <canvas id="barChart" width="10" height="5"></canvas>
```

```
<script>
    // Datos para el gráfico de barras
    var data = {
        labels: ["17/09", "18/09", "19/09", "20/09", "21/09", "22/09", "23/09"],
        datasets: [{
            label: "Ventas",
            data: [12.50, 19.00, 3.50, 5, 2, 10, 20],
            backgroundColor: 'rgb(4, 24, 124)',
            borderColor: 'rgba(75, 192, 192, 1)',
            borderWidth: 1
        }]
    };

    // Opciones del gráfico de barras
    var options = {
        scales: {
            y: {
                beginAtZero: true
            }
        },
        plugins: {
            datalabels: {
                anchor: 'end',
                align: 'end',
                offset: 4,
                color: 'red',
                font: {
                    weight: 'bold',
                    size: 12
                }
            }
        }
    };

    // Crear el gráfico de barras
```

```
        var ctx = document.getElementById('barChart').getContext('2d');
        var barChart = new Chart(ctx, {
            type: 'bar',
            data: data,
            options: options
        });
    </script>
</body>
</html>
```



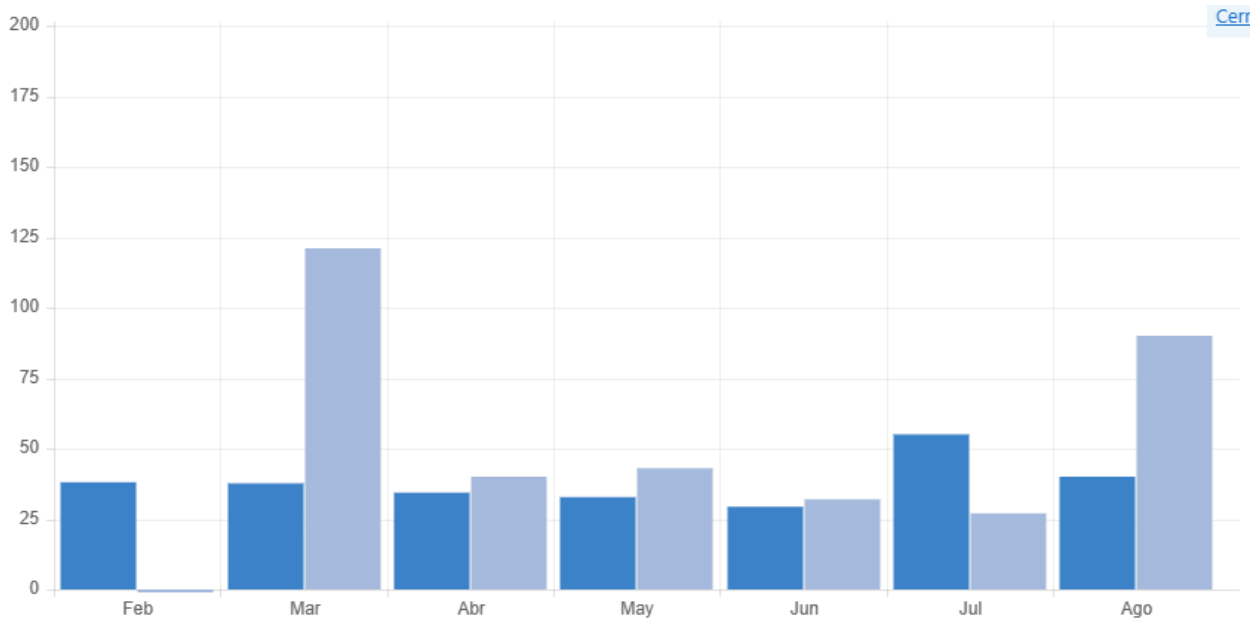
2. gráficos de línea

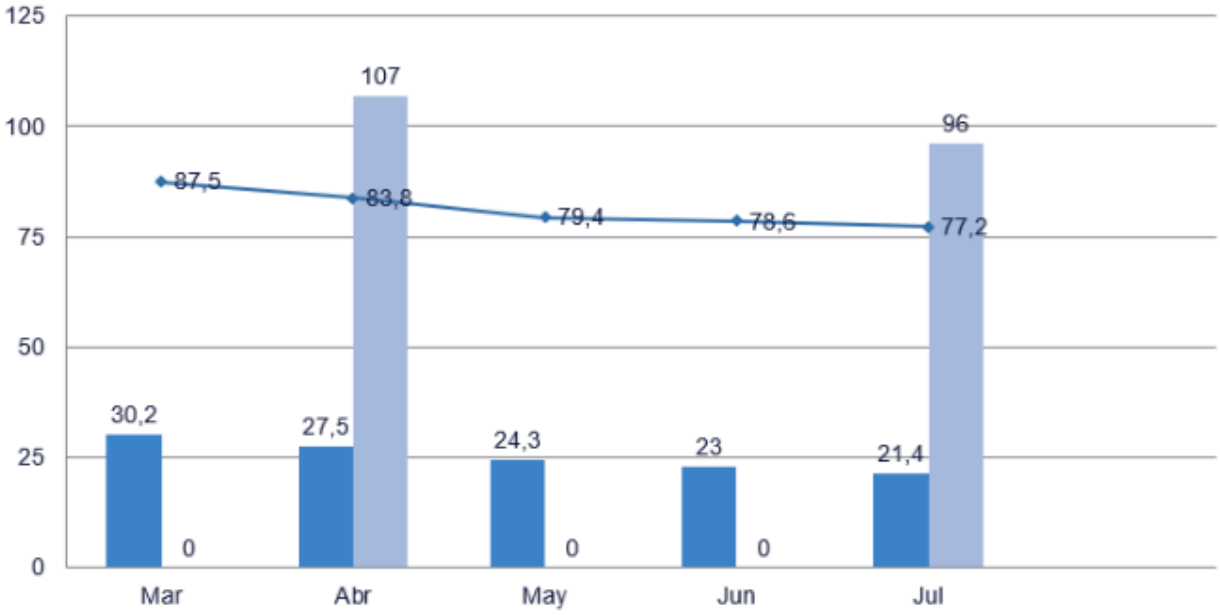
```
/* ChartJS
    * -------
    * Here we will create a few charts using ChartJS
    */

    var areaChartData = {
```

```
    labels  : ['Feb', 'Mar', 'Abr', 'May', 'Jun', 'Jul', 'Ago'],
    datasets: [
     {
      label               : 'Masa grasa',
      fillColor           : 'rgba(63,134,203,1)',
      strokeColor         : 'rgba(63,134,203,1)',
      pointColor          : 'rgba(63,134,203,1)',
      pointStrokeColor    : '#c1c7d1',
      pointHighlightFill  : '#fff',
      pointHighlightStroke: 'rgba(63,134,203,1)',
      data                : [38, 37.6, 34.4, 32.8, 29.3, 55, 40]
     },
     {
      label               : 'Circ cintura',
      fillColor           : '#a6bcdf',
      strokeColor         : '#a6bcdf',
      pointColor          : '#a6bcdf',
      pointStrokeColor    : '#a6bcdf',
      pointHighlightFill  : '#fff',
      pointHighlightStroke: '#a6bcdf',
      data                : [0, 121, 40, 43, 32, 27, 90]
     }
    ]
   }
   //-------------
   //- BAR CHART -
   //-------------
var barChartCanvas                   = document.getElementById("barChart").getCont
   var barChart                      = new Chart(barChartCanvas);
   var barChartData                  = areaChartData
   barChartData.datasets[1].fillColor   = '#a6bcdf'
   barChartData.datasets[1].strokeColor = '#a6bcdf'
   barChartData.datasets[1].pointColor  = '#a6bcdf'
   var barChartOptions               = {
    //Boolean - Whether the scale should start at zero, or an order of magnitude
    scaleBeginAtZero        : true,
```

```
    //Boolean - Whether grid lines are shown across the chart
    scaleShowGridLines      : true,
    //String - Colour of the grid lines
    scaleGridLineColor      : 'rgba(0,0,0,.05)',
    //Number - Width of the grid lines
    scaleGridLineWidth      : 1,
    //Boolean - Whether to show horizontal lines (except X axis)
    scaleShowHorizontalLines: true,
    //Boolean - Whether to show vertical lines (except Y axis)
    scaleShowVerticalLines  : true,
    //Boolean - If there is a stroke on each bar
    barShowStroke           : true,
    //Number - Pixel width of the bar stroke
    barStrokeWidth          : 2,
    //Number - Spacing between each of the X value sets
    barValueSpacing         : 5,
    //Number - Spacing between data sets within X values
    barDatasetSpacing       : 1,
    //String - A legend template
    legendTemplate          : '<ul class="<%=name.toLowerCase()%>-legend"><%
    //Boolean - whether to make the chart responsive
    responsive              : true,
    maintainAspectRatio     : true
  }

  barChartOptions.datasetFill = false
  barChart.Bar(barChartData, barChartOptions)
```

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/1.1.1/Chart.min.js"></
<script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/1.1.1/Chart.js"></scrip
<canvas id="barChart" style="height:230px"></canvas>
```



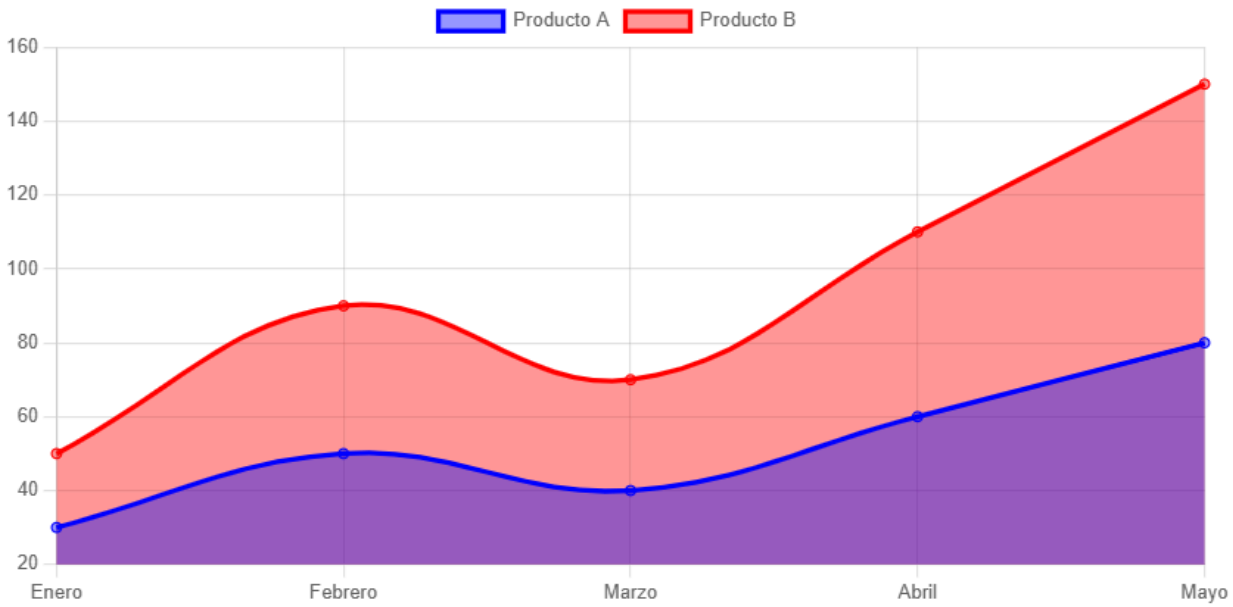


3. área https://chatgpt.com/c/67bb5cf5-89a0-8005-b0ef-8f925354d2e5

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Gráfico de Área Apilado</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <canvas id="graficoAreaApilado"></canvas>
    <script>
        const ctx = document.getElementById('graficoAreaApilado').getContext('2d
        new Chart(ctx, {
            type: 'line',
            data: {
                labels: ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo'],
                datasets: [
                    {
                        label: 'Producto A',
                        data: [30, 50, 40, 60, 80],
                        borderColor: 'blue',
                        backgroundColor: 'rgba(0, 0, 255, 0.4)',
                        fill: true,
                        tension: 0.4,
                        stack: 'grupo1'
                    },
                    {
                        label: 'Producto B',
                        data: [20, 40, 30, 50, 70],
                        borderColor: 'red',
                        backgroundColor: 'rgba(255, 0, 0, 0.4)',
                        fill: true,
                        tension: 0.4,
                        stack: 'grupo1'
                    }
                ]
            },
            options: {
                responsive: true,
                scales: {
                    y: {
```

```
                stacked: true
            }
        }
    }
});
</script>
</body>
</html>
```



4. circular

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Gráfico de Pastel - Chart.js</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
```

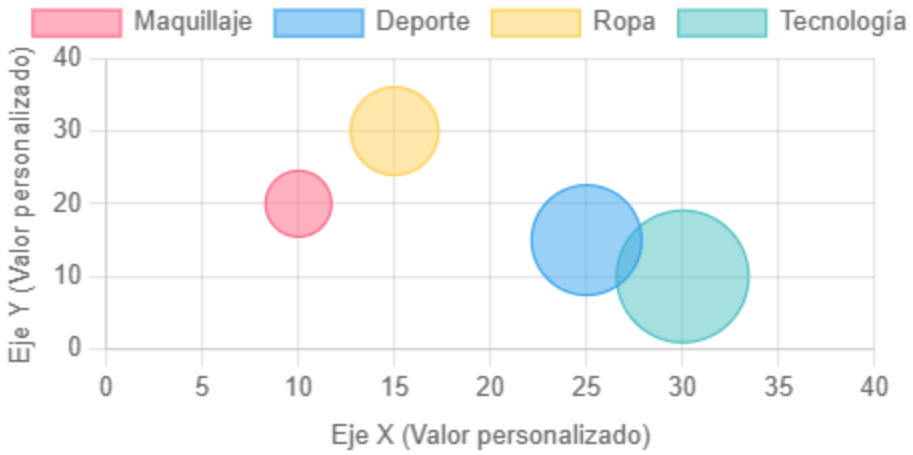
```
    <canvas id="graficoPastel"></canvas>
    <script>
        const ctx = document.getElementById('graficoPastel').getContext('2d');

        new Chart(ctx, {
            type: 'pie', // Gráfico de pastel
            data: {
                labels: ['Maquillaje', 'Deporte', 'Ropa', 'Tecnología'],
                datasets: [{
                    label: 'Preferencias de Compra',
                    data: [10, 35, 20, 50], // Valores más variados
                    backgroundColor: ['#FF6384', '#36A2EB', '#FFCE56', '#4BC0C0'],
                    hoverOffset: 10
                }]
            },
            options: {
                responsive: true
            }
        });
    </script>
</body>
</html>
```

5. burbuja

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Gráfico de Burbuja - Chart.js</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <canvas id="graficoBurbuja"></canvas>
    <script>
        const ctx = document.getElementById('graficoBurbuja').getContext('2d');
```

```
new Chart(ctx, {
    type: 'bubble', // Tipo de gráfico de burbuja
    data: {
        datasets: [
            {
                label: 'Maquillaje',
                data: [{ x: 10, y: 20, r: 15 }], // r: tamaño de la burbuja
                backgroundColor: 'rgba(255, 99, 132, 0.5)',
                borderColor: 'rgba(255, 99, 132, 1)'
            },
            {
                label: 'Deporte',
                data: [{ x: 25, y: 15, r: 25 }],
                backgroundColor: 'rgba(54, 162, 235, 0.5)',
                borderColor: 'rgba(54, 162, 235, 1)'
            },
            {
                label: 'Ropa',
                data: [{ x: 15, y: 30, r: 20 }],
                backgroundColor: 'rgba(255, 206, 86, 0.5)',
                borderColor: 'rgba(255, 206, 86, 1)'
            },
            {
                label: 'Tecnología',
                data: [{ x: 30, y: 10, r: 30 }],
                backgroundColor: 'rgba(75, 192, 192, 0.5)',
                borderColor: 'rgba(75, 192, 192, 1)'
            }
        ]
    },
    options: {
        responsive: true,
        scales: {
            x: {
                title: { display: true, text: 'Eje X (Valor personalizado)' },
```

```
                min: 0,
                max: 40
            },
            y: {
                title: { display: true, text: 'Eje Y (Valor personalizado)' },
                min: 0,
                max: 40
            }
        }
    }
});
</script>
</body>
</html>
```

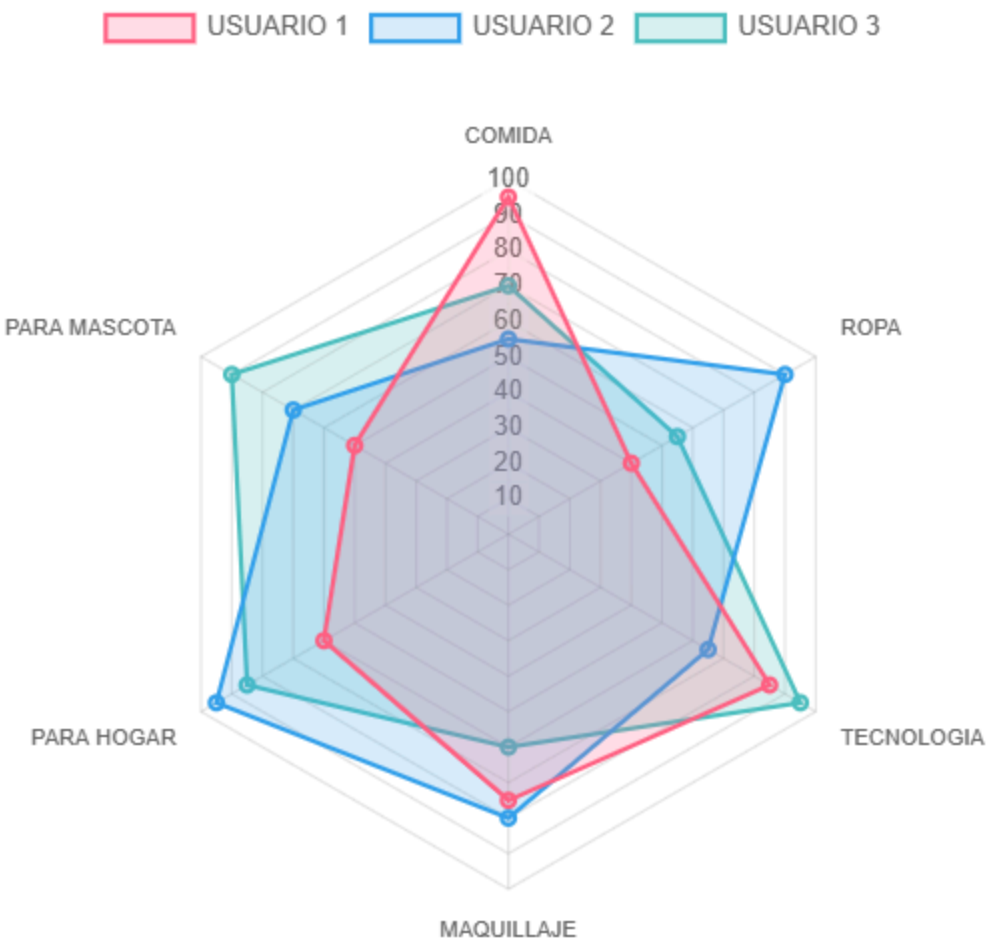


6. radar https://chatgpt.com/c/67bb6581-e2a8-8005-8d0b-9218d11c0043

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Gráfico de Radar con Chart.js</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>

    <canvas id="radarChart"></canvas>

    <script>
        const ctx = document.getElementById('radarChart').getContext('2d');

        const data = {
            labels: ['COMIDA', 'ROPA', 'TECNOLOGIA', 'MAQUILLAJE', 'PARA HOGAR',
            datasets: [{
                label: 'USUARIO 1',
                data: [95, 40, 85, 75, 60, 50], // Valores más variados
                backgroundColor: 'rgba(255, 99, 132, 0.2)',
                borderColor: 'rgba(255, 99, 132, 1)',
                borderWidth: 2
            }, {
                label: 'USUARIO 2',
                data: [55, 90, 65, 80, 95, 70],
                backgroundColor: 'rgba(54, 162, 235, 0.2)',
                borderColor: 'rgba(54, 162, 235, 1)',
                borderWidth: 2
            }, {
                label: 'USUARIO 3',
                data: [70, 55, 95, 60, 85, 90],
                backgroundColor: 'rgba(75, 192, 192, 0.2)',
                borderColor: 'rgba(75, 192, 192, 1)',
                borderWidth: 2
            }]
        };

        const options = {
            responsive: true,
```

```
            scales: {
                r: {
                    beginAtZero: true,
                    suggestedMin: 0,
                    suggestedMax: 100
                }
            }
        };

        new Chart(ctx, {
            type: 'radar',
            data: data,
            options: options
        });
    </script>

</body>
</html>
```

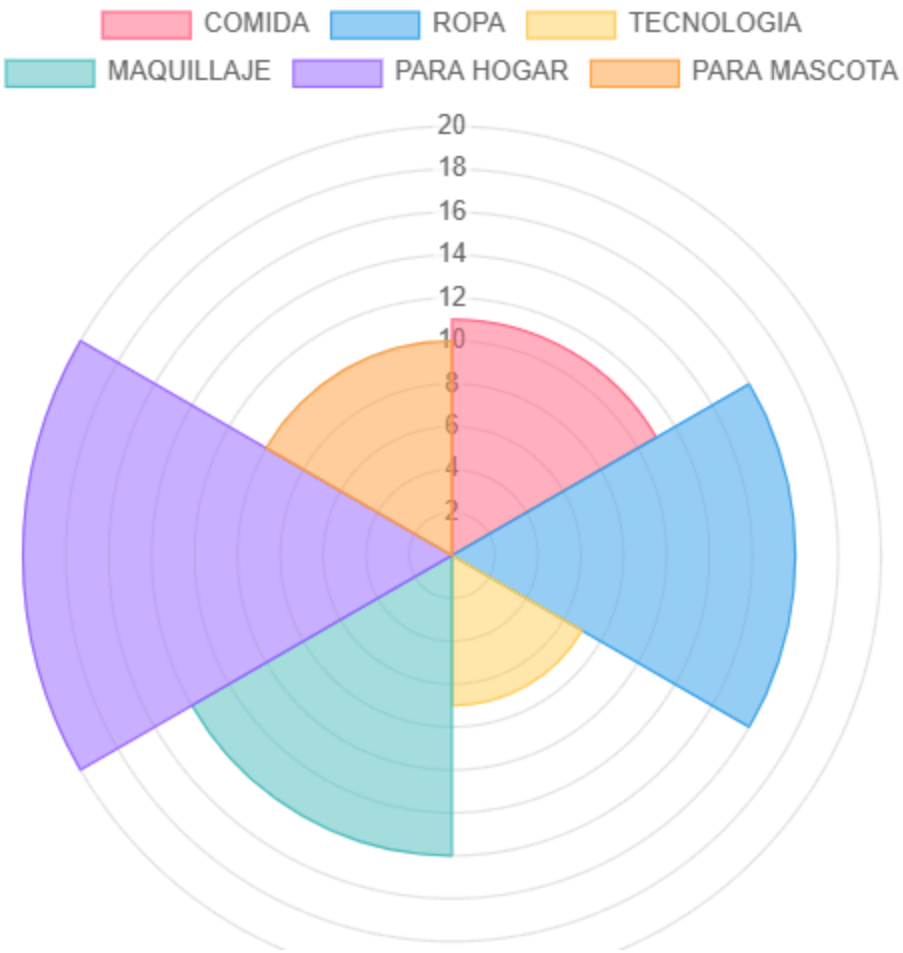


7. polar

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Gráfico Polar en Chart.js</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <canvas id="miGraficoPolar"></canvas>

    <script>
```

```
const ctx = document.getElementById('miGraficoPolar').getContext('2d');

new Chart(ctx, {
    type: 'polarArea',
    data: {
        labels: ['COMIDA', 'ROPA', 'TECNOLOGIA', 'MAQUILLAJE', 'PARA HOGA
        datasets: [{
            label: 'Mi conjunto de datos',
            data: [11, 16, 7, 14, 20, 10], // Valores de cada sector
            backgroundColor: [
                'rgba(255, 99, 132, 0.5)',
                'rgba(54, 162, 235, 0.5)',
                'rgba(255, 206, 86, 0.5)',
                'rgba(75, 192, 192, 0.5)',
                'rgba(153, 102, 255, 0.5)',
                'rgba(255, 159, 64, 0.5)'
            ],
            borderColor: [
                'rgba(255, 99, 132, 1)',
                'rgba(54, 162, 235, 1)',
                'rgba(255, 206, 86, 1)',
                'rgba(75, 192, 192, 1)',
                'rgba(153, 102, 255, 1)',
                'rgba(255, 159, 64, 1)'
            ],
            borderWidth: 1
        }]
    },
    options: {
        responsive: true,
        plugins: {
            legend: {
                position: 'top',
            }
        },
        scales: {
```

```
            r: { // Escala radial
                beginAtZero: true
            }
        }
    }
});
</script>
</body>
</html>
```

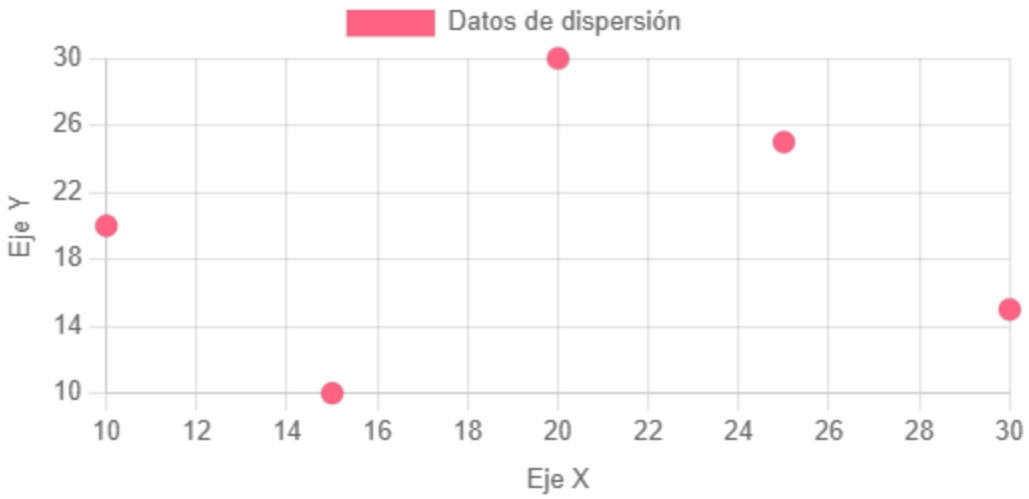


8. dispersión

```
<!DOCTYPE html>
<html lang="es">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Gráfico de Dispersión</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <canvas id="scatterChart"></canvas>
    <script>
        const ctx = document.getElementById('scatterChart').getContext('2d');

        const data = {
            datasets: [{
                label: 'Datos de dispersión',
                data: [
                    { x: 10, y: 20 },
                    { x: 15, y: 10 },
                    { x: 20, y: 30 },
                    { x: 25, y: 25 },
                    { x: 30, y: 15 }
                ],
                backgroundColor: 'rgb(255, 99, 132)', // Color de los puntos
                pointRadius: 5 // Tamaño de los puntos
            }]
        };

        new Chart(ctx, {
            type: 'scatter',
            data: data,
            options: {
                scales: {
                    x: {
                        type: 'linear',
                        position: 'bottom',
                        title: { display: true, text: 'Eje X' }
                    },
```

```
            y: {
                title: { display: true, text: 'Eje Y' }
            }
        }
    }
});
</script>
</body>
</html>
```



USOS DE CHART JS EN E-COMMERCE Y/O UN MARKETPLACE

el chart.js utilizado para mostrar graficas se podría utilizar para demostrar a los usuarios sus mayores interese de compra en la pagina y a nosotros como desarrolladores de la pagina, ayudaría a que el algoritmo aprenda de los gustos de los usuarios para recomendarle mas productos del gusto del usuario

CUÁL ES LA RECETA PARA HACER UN CHAT

1. Receta para desarrollar un chat con Chart.js

Para crear un chat con visualización de datos en tiempo real usando Chart.js, se siguen los siguientes pasos:

Paso 1: Configurar el entorno

- Instalar Node.js y npm.
- Crear un proyecto con npm init -y.
- Instalar las dependencias necesarias: npm install express socket.io chart.js.

Paso 2: Configurar el servidor con Express y Socket.IO

Crear un archivo server.js con el siguiente código:

```
const express = require('express');
const http = require('http');
const socketIo = require('socket.io');

const app = express();
const server = http.createServer(app);
const io = socketIo(server);

app.use(express.static('public'));

io.on('connection', (socket) => {
    console.log('Nuevo usuario conectado');
    socket.on('message', (data) => {
        io.emit('message', data);
    });
});

server.listen(3000, () => console.log('Servidor en http://localhost:3000'));
```

Paso 3: Crear la interfaz web

En la carpeta public, crear un archivo index.html:

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Chat con Chart.js</title>
    <script src="/socket.io/socket.io.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <input type="text" id="message" placeholder="Escribe un mensaje...">
    <button onclick="sendMessage()">Enviar</button>
    <canvas id="myChart"></canvas>
    <script src="script.js"></script>
</body>
</html>
```

Paso 4: Configurar el frontend con Chart.js

Crear public/script.js para manejar los mensajes y la actualización de gráficos:

```
const socket = io();
const ctx = document.getElementById('myChart').getContext('2d');
let messagesCount = [];
let labels = [];

let chart = new Chart(ctx, {
    type: 'bar',
    data: {
        labels: labels,
        datasets: [{
            label: 'Mensajes por usuario',
            data: messagesCount,
            backgroundColor: 'blue'
        }]
    },
    options: {
        responsive: true
    }
});

function sendMessage() {
    let message = document.getElementById('message').value;
    socket.emit('message', { user: 'Usuario', text: message });
}

socket.on('message', (data) => {
    let index = labels.indexOf(data.user);
    if (index === -1) {
        labels.push(data.user);
        messagesCount.push(1);
    } else {
        messagesCount[index] += 1;
    }
    chart.update();
});
```

CUÁL ES LA DIFICULTAD

La dificultad de usar Chart.js puede variar dependiendo de tu nivel de experiencia con JavaScript y la complejidad del gráfico que deseas crear. Sin embargo, en general, Chart.js es considerada una biblioteca fácil de aprender y usar ya que no depende de otras biblioteca y es muy versátil a la hora de hacer gráficos.

Implementar un chat con visualización en tiempo real mediante Chart.js presenta algunos desafíos:

- **Gestión de rendimiento**: La actualización constante de los gráficos puede afectar el rendimiento si se manejan grandes volúmenes de datos.
- **Sincronización de datos**: Garantizar que los datos del gráfico se actualicen correctamente sin retrasos o inconsistencias.
- **Escalabilidad**: Manejar múltiples conexiones en tiempo real sin afectar la estabilidad del sistema.
- **Manejo de eventos**: Coordinar la recepción y visualización de datos en tiempo real de manera eficiente.

Uso de Chart.js para la visualización de datos en un chat

Chart.js puede utilizarse para representar gráficamente la actividad del chat, como la cantidad de mensajes enviados por usuario, la frecuencia de mensajes en un período de tiempo o estadísticas de interacción.

Características clave de Chart.js

- Permite la creación de gráficos dinámicos y personalizables.
- Compatible con múltiples tipos de gráficos como barras, líneas y tortas.
- Soporte para animaciones y actualizaciones en tiempo real.

tecnologías complementarias

- **js**: Servidor backend para gestionar la comunicación entre clientes.
- **MongoDB/Firebase**: Almacén de datos para guardar mensajes y estadísticas.
- **js**: Alternativa para gráficos interactivos más avanzados.