UNIVERSIDAD TÉCNICA NACIONAL

SEDE CENTRAL

INGENIERÍA DEL SOFTWARE

Aplicaciones Web Utilizando Software Libre

1C2024-G01 (CÓDIGO DEL CURSO)

Investigación 1

PROFESOR:

Nathalie Paniagua López

ELABORADO POR:

Rodrigo Andrey Jiménez Porras

Kimberly Fabiola Vargas Herrera

ALAJUELA 20 DE MARZO DEL 2024.

Reportes en con JsPDF

Links: https://www.npmjs.com/package/jspdf-autotable

Comando:
npm Install jspdf jspdf-autotable

Este comando es para que en el proyecto se nos instale la librerías de jsPDF en las cuales van a venir métodos que nos va a ayudar para diferentes reportes como lo es los listados, las tablas y guardarlos en formato PDF.

Lo primero que vamos a realizar es un servicio para utilizarlo en cada parte del proyecto que lo vayamos a ocupar, esto sirve para no dejarlo solamente para un componente si no que varios componentes pueda utilizarlo y para que no haya necesidad de estar pasando de componente a componente si no que sea un método global.

Comando para el servicio :

ng g service shared /services/impresion –skip test

Ya creado el servicio importamos la librería de JsPDF en el servicio “ImpresionService”:

Ejemplo:

```
import { jsPDF } from "jspdf";

import autoTable from 'jspdf-autotable'
```

Lo primero que vamos a hacer es crear un componente o donde lo vayamos a implementar.

Ya creado el componente se importa el “ImpresionService” y también al contructor:

-Componente

Ejemplo:

```
import { ImpresionService } from
'../../../share/services/impresion.service';
constructor(private httpRequest: HttpRequestService, private router: Router,
private ImpresionService: ImpresionService) {
  }
```

Reporte Tabla

Para hacer un table tenemos que hacer un arreglo el cual va a ser llamado encabezado el cual se va a mostrar con valores estáticos como lo es los campos de cada columna y un cuerpo el cual va a tener una lista de todos los datos que vamos a meter a la tabla.

-Componente
Ejemplo:

```
  onImprimir() {
    const encabezado = ["Id Producto", "Nombre", "Descripcion", "Precio",
"Categoria", "SubCategoria"]
    const cuerpo = this.productos;
    this.ImpresionService.imprimir(encabezado, cuerpo, "Tabla de productos",
true);
  }
```

El servicio de impresión hacemos un método el cual se va a llamar de su preferencia pero en mi caso lo llame ““imprimir ()””.

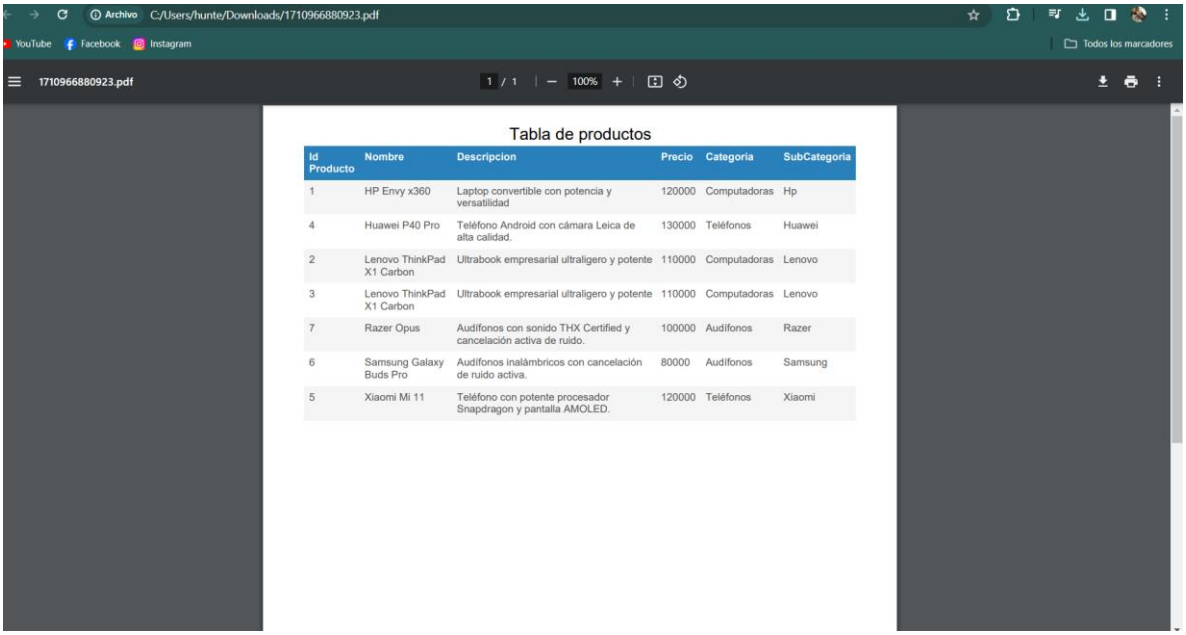
-ImpresionServicio
Ejemplo:

```
  imprimir(encabezado: string[], cuerpo: Array<any>, titulo: string,
guardar?: boolean) {
    const doc = new jsPDF({
      orientation: "portrait",
      unit: "px",
      format: 'letter'
    });
    const productos = cuerpo.map(prod => {
      return [
        prod.id,
        prod.nombre,
        prod.descripcion,
        prod.precio,
        prod.categoria.nombre,
        prod.subCategorias.nombre
      ]
    });

    doc.text(titulo, doc.internal.pageSize.width / 2, 25, { align: 'center'
});
    autoTable(doc, {
      head: [encabezado],
      body: productos,
    })

    if (guardar) {
      const hoy = new Date();
      doc.save(hoy.getDate() + hoy.getMonth() + hoy.getFullYear() +
hoy.getTime() + '.pdf')
    }
  }
```

Imagen representativa:



Id Producto	Nombre	Descripcion	Precio	Categoria	SubCategoria
1	HP Envy x360	Laptop convertible con potencia y versatilidad	120000	Computadoras	Hp
4	Huawei P40 Pro	Teléfono Android con cámara Leica de alta calidad.	130000	Teléfonos	Huawei
2	Lenovo ThinkPad X1 Carbon	Ultrabook empresarial ultraligero y potente	110000	Computadoras	Lenovo
3	Lenovo ThinkPad X1 Carbon	Ultrabook empresarial ultraligero y potente	110000	Computadoras	Lenovo
7	Razer Opus	Audífonos con sonido THX Certified y cancelación activa de ruido.	100000	Audífonos	Razer
6	Samsung Galaxy Buds Pro	Audífonos inalámbricos con cancelación de ruido activa.	80000	Audífonos	Samsung
5	Xiaomi Mi 11	Teléfono con potente procesador Snapdragon y pantalla AMOLED.	120000	Teléfonos	Xiaomi

Reporte Listado

Para el listado repetimos el mismo proceso que la tabla a excepción de enviarle un encabezado y concatenar en una misma fila los valores que necesitamos.

-Componente

Ejemplo:

```
  onImprimirListado() {
    const cuerpo = this.productos;
    console.log(cuerpo);
    this.ImpresionService.imprimirListado(cuerpo, "Listado de productos",
true);
  }
```

En el servicio hacemos un método llamado “imprimirListado()” el cual va a concatenar los valor que necesitamos sin la necesidad de tener una tabla.

-ImpresionService

Ejemplo:

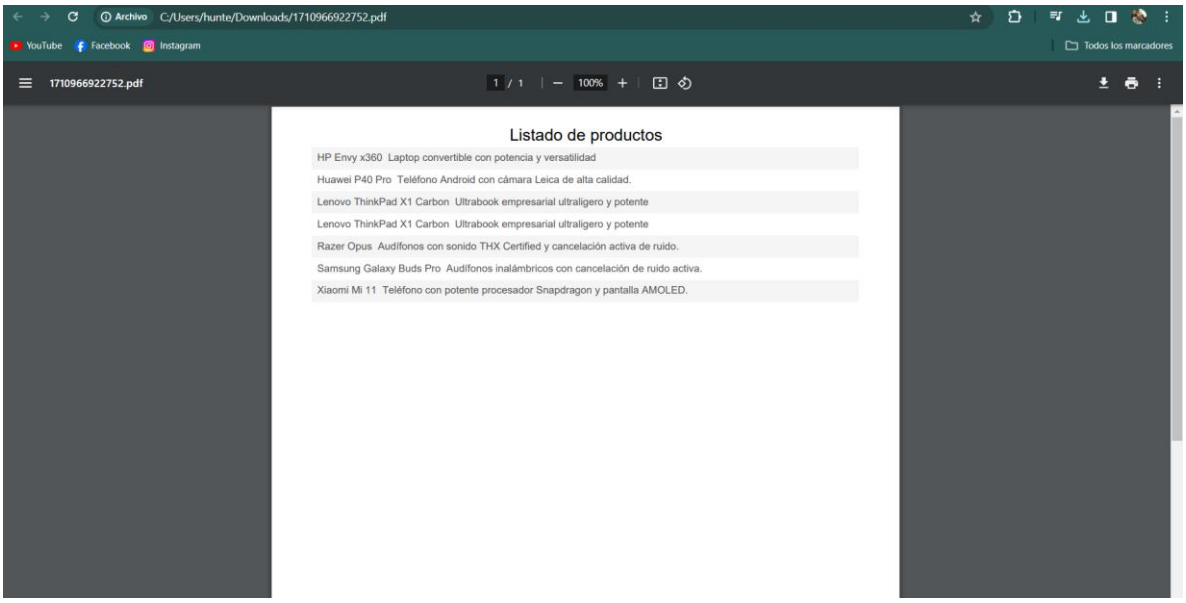
```
 imprimirListado(cuerpo: Array<any>, titulo: string, guardar?: boolean) {
    const doc = new jsPDF({
      orientation: "portrait",
      unit: "px",
      format: 'letter'
    });
    const productos = cuerpo.map(prod => {
      return [
        prod.nombre + "   " + prod.descripcion,

      ]
    });
    doc.text(titulo, doc.internal.pageSize.width / 2, 25, { align: 'center'
});
    autoTable(doc, {

      body: productos,
    })
    if (guardar) {
      const hoy = new Date();
      doc.save(hoy.getDate() + hoy.getMonth() + hoy.getFullYear() +
hoy.getTime() + '.pdf')
    }

}
```

Imagen representativa:



Reporte de grafico:

Lo primero en hacer es instalar el “Chart” en el proyecto para poder implementarlo y que podamos utilizarlo.

Comando:

npm i chart.js

Previamente necesitamos tener un componente o servicio ya creado el cual nos permita implementar y utilizar el “Chart”.

Se importa el Chart al proyecto:

```
import { Chart, ChartType } from 'chart.js/auto'
```

Se construye un metodo “ngOnInit()” el cual implemente este código y va a ser el responsable de cargar el grafico, el siguiente código le va a dar el formato al grafico, los campos y data que va a utilizar.

Ejemplo:

```
 const data = {
      labels: ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio',
'Julio', 'Agosto', 'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'],
      datasets: [{
        label: 'Grafico de Clientes Registrados',
        data: this.datosGrafico,
        backgroundColor: [
          'rgba(255, 99, 132, 0.2)',
          'rgba(255, 159, 64, 0.2)',
          'rgba(255, 205, 86, 0.2)',
          'rgba(75, 192, 192, 0.2)',
          'rgba(54, 162, 235, 0.2)',
          'rgba(153, 102, 255, 0.2)',
          'rgba(201, 203, 207, 0.2)',
          'rgba(220, 53, 69, 0.2)',
          'rgba(255, 127, 14, 0.2)',
          'rgba(221, 160, 221, 0.2)',
          'rgba(137, 54, 177, 0.2)', 'rgba(105, 105, 105, 0.2)'
        ],
        borderColor: [
          'rgb(255, 99, 132)',
          'rgb(255, 159, 64)',
```

```
          'rgb(255, 205, 86)',
          'rgb(75, 192, 192)',
          'rgb(54, 162, 235)',
          'rgb(153, 102, 255)',
          'rgb(201, 203, 207)'
        ],
        borderWidth: 1
      }]
    };
```

Acá construimos el grafico con esa data o los estilos que vaya a tener el grafico.

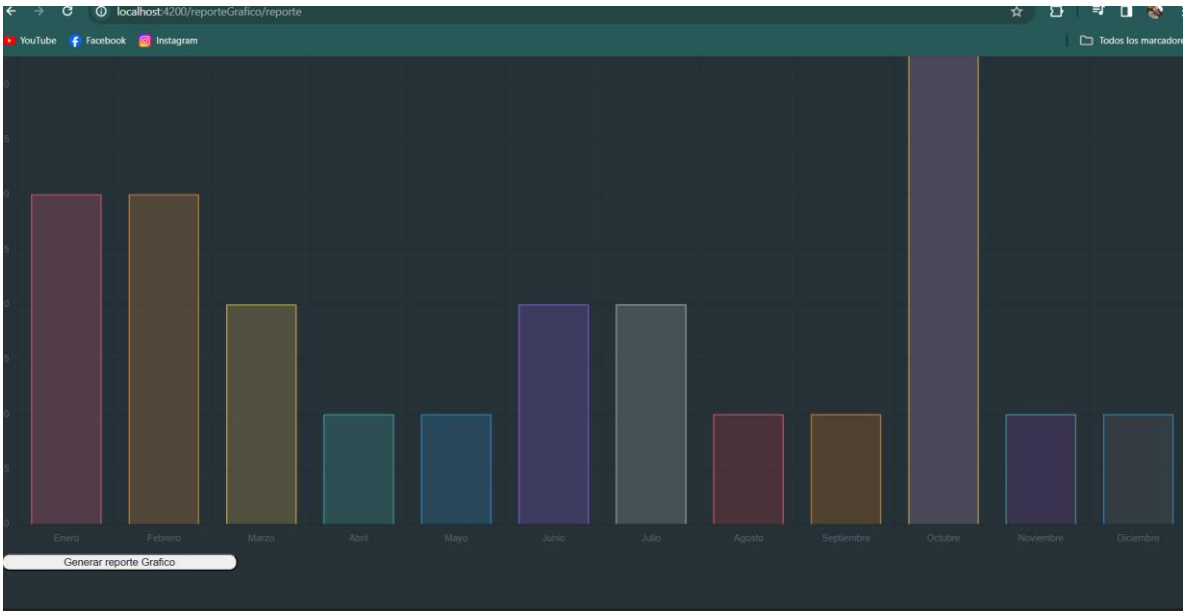
```
    // Creamos la gráfica
    this.chart = new Chart("chart", {
      type: 'bar' as ChartType, // tipo de la gráfica
      data: data, // datos
      options: { // opciones de la gráfica
        scales: {
          y: {
            beginAtZero: true
          }
        }
      },
    });
```

Ya cuando tenemos esto tenemos que implementar una línea en el html:

Ejemplo:

```
<div>
    <canvas id="chart">{{chart}} </canvas>
</div>
```

Imagen representativa:



Guardar Grafico PDF

Anteriormente habíamos instalado el JsPDF, solamente es de importarlo y utilizarlo, pero también se instala el Canvas2

Comando:

npm i html2canvas

Importamos el JsPDF y el Canvas2:

```
import { jsPDF } from "jspdf";
import html2canvas from 'html2canvas';
```

Hacemos un método “Descargar()” para llamarlo al HTML y que guarde el reporte en formato PDF.

Ejemplo:

```
descargar() {
  html2canvas(document.body).then(canvas => {
    const contentDataURL = canvas.toDataURL('imagen/png')
    let pdf = new jsPDF('p', 'px', 'a4');

    pdf.addImage(contentDataURL, 'PNG', 0, 0, 450, 250)
    pdf.save(this.title + '.pdf');


  })
}
```