app-routing.module.ts

```
import { NgModule } from '@angular/core';
                                        import { RouterModule, Routes } from '@angular/router

redirectUnauthorizedToLogin = () => redirectUnauthorizedTo(['login']);

'./guards/auth.guard';
                      import { AngularFireAuthGuard, hasCustomClaim, redirectUnauthorizedTo, r
{PagesComponent} from './components/pages.component';
                                                 import {LoginComponent} from './component

import {PuntoventaComponent} from './components/puntoventa/puntoventa.component';
                                                                          import {Edit
editcliente.component';
                        import { NewclienteComponent } from './components/newcliente/newclient
condicionventa/condicionventa.component';
                                        import { ProductosComponent } from './components/proc
components/calculoproducto/calculoproducto.component';
                                                  import { CarritoComponent } from './comp
components/venta/venta.component';
                                   import { CuentacorrienteComponent } from './components/cuent
from './components/cabeceras/cabeceras.component';
                                                 import { AnalisiscajaComponent } from './com

import {LoginguardGuard} from './guards/loginguard.guard';
                                                       import {SuperGuard} from './guards/s

{AdminGuard} from './guards/admin.guard';

                                                                          const APP_RO


canActivate:[AngularFireAuthGuard, LoginguardGuard], data: { authGuardPipe: redirectUnauthoriz

component: PuntoventaComponent, data :{titulo:'Punto de Venta'} },
                                                                  { path: 'editcliente', c
{ path: 'newcliente', component: NewclienteComponent, data :{titulo:'Nuevo Cliente'} },
                                                                                  { p
{titulo:'Condicion de Venta'} },
                                     { path: 'productos', component: ProductosComponent, data :
CalculoproductoComponent, data :{titulo:'Calculo de Producto'} },
                                                                 { path: 'carrito', compor
component: VentaComponent, data :{titulo:'Venta'} },
                                                        { path: 'cuentacorriente', component:
'cabeceras', component: CabecerasComponent, data: {titulo: "CC"} },
                                                                    {path: 'analisiscaja',
```

```
              //{ path: 'puntosmedicion', component: PuntosmedicionComponent, data:{titulo: "
'chart/:keygrupo/:keymedicion/:nombregrupo/:nombrepunto', component: ChartComponent, data :{ti
{titulo:'Admin'} ,canActivate: [SuperGuard]},
                                                  //{ path: 'noway', component: NowayComponent, c
WorkingComponent, data :{titulo:'Working'}},
                                                // { path: 'alarms', component: AlarmsComponent,
ManagealarmsComponent, data :{titulo:'Manage Alarms'}},
                                                                  //{ path: 'customimagenes', component
Imagenes'},canActivate: [SuperGuard]},
                                             //{ path: 'configuraciones', component: Configuracione
  //{ path: 'exportar', component: ExportarComponent, data :{titulo:'Exportar'},canActivate: [
ChartanalisisComponent, data :{titulo:'Analisis'}},
                                                           //{ path: 'adminusuarios', component: Adm
[AdminGuard]},
                //{ path: 'dbmanager', component: DbmanagerComponent, data :{titulo:'DB manage
DispositivosComponent, data :{titulo:'Dispositivos Manager'},canActivate: [SuperGuard]}
                                                                                                  // {
{titulo:'Cammesa'} },
                       // { path: 'cammesahora', component: CammesahoraComponent, data :{titul
AdusuariosComponent, data :{titulo:'Usuarios'},  canActivate: [AdminGuard] },
                                                                                  // { path: 'not
{titulo:'Notificaciones'},  canActivate: [AdminGuard] },
                                                          // { path: 'analisis/:cd_et/:distrib
Distribuidor'} },
                    // { path: 'eventos', component: EventosComponent, data :{titulo:'Eventos'}
{titulo:'Scada'} },









{ path: 'login', component: LoginComponent },
                                               { path: '**', pathMatch: 'full', redirectTo: 'log


];
    @NgModule({
                  imports: [RouterModule.forRoot(APP_ROUTES)],
                                                            exports: [RouterModule]
                                                                                  })
                                                                                    export
```

app.component.ts

```
import { Component,OnInit } from '@angular/core';
                                                declare function init_plugins();
                                                                                declare var j



templateUrl: './app.component.html',
                                      styleUrls: ['./app.component.css']
                                                                                 })
                                                                                   export class AppComp



console.log("APP COMPONENTS");
                                     init_plugins();
```

```
import { NgModule } from '@angular/core';
                                        import { BrowserModule } from '@angular/platform-brow
animations';
                        import { DatePipe } from '@angular/common';

dynamicdialog';
               import { DropdownModule } from 'primeng/dropdown';

http';
      import {FormsModule,ReactiveFormsModule } from '@angular/forms';

from './app-routing.module';
                             import { AppComponent } from './app.component';
                                                                    import { LoginCompo
{ initializeApp,provideFirebaseApp } from '@angular/fire/app';
                                                                 import { environment } from '../
fire/auth';
           import { provideDatabase,getDatabase } from '@angular/fire/database';
                                                                          import { provi
{ provideMessaging,getMessaging } from '@angular/fire/messaging';
                                                                 import { provideStorage,getSt
from '@angular/fire/compat/auth-guard';
                                        import { FIREBASE_OPTIONS } from '@angular/fire/compat'

{ PagesComponent } from './components/pages.component';
                                                        import { FooterComponent } from './shar

import { HeaderComponent } from './shared/header/header.component';
                                                          import { SidebarComponent }
{ PuntoventaComponent } from './components/puntoventa/puntoventa.component';

{ ButtonModule } from 'primeng/button';
                                        import { TableModule } from 'primeng/table';
                                                                          import { Ca
from 'primeng/multiselect';

                                                    import { EditclienteComponent } from './
{ NewclienteComponent } from './components/newcliente/newcliente.component';
                                                                    import { Condicion
condicionventa.component';
                          import { ProductosComponent } from './components/productos/productos
calculoproducto/calculoproducto.component';
                                          import { CarritoComponent } from './components/carr
venta/venta.component';
                        import { FilterPipe } from './pipes/filter.pipe';
                                                                     import { Analisispedid
analisispedidos.component';
```

```
                          import { CuentacorrienteComponent } from './components/cuentacorrie
components/cabeceras/cabeceras.component';
                                        import { CondicionventacabComponent } from './compor
{ AnalisiscajaComponent } from './components/analisiscaja/analisiscaja.component';
                                                                                  import { Ped
{ RecibosComponent } from './components/recibos/recibos.component';
                                                                  //import { DataTablesModule


@NgModule({
            declarations: [
                                AppComponent,
                                        LoginComponent,
                                                PagesComponent,
                                                                    FooterC



PuntoventaComponent,
                        EditclienteComponent,
                                        NewclienteComponent,
                                                            CondicionventaCompone


CarritoComponent,
                    VentaComponent,
                                    FilterPipe,
                                            AnalisispedidosComponent,
                                                                    Cuentacorr


    AnalisiscajaComponent,
                                PedidosComponent,
                                        RecibosComponent,
                                                                                ],



MultiSelectModule,
                     //DataTablesModule,
                                        FormsModule,
                                                ReactiveFormsModule,
                                                                    BrowserMo


HttpClientModule,
                    DynamicDialogModule,
                                        DropdownModule,
                                                        provideFirebaseApp(() => initi


provideDatabase(() => getDatabase()),
                                        provideFunctions(() => getFunctions()),
                                                                    provideMes
```

```
  ],
      providers: [  DatePipe, AngularFireAuthGuard, { provide: FIREBASE_OPTIONS, useValue: env


{ }
```

components\analisiscaja\analisiscaja.component.ts

```
import { Component, OnInit } from '@angular/core';
                                                    import { Cabecera } from '../../interfaces/c


{ CargardataService } from '../../services/cargardata.service';
                                                              import { Cliente } from '../../


from '@angular/router';
                        import * as FileSaver from 'file-saver';
                                                              import xlsx from 'xlsx/xlsx';
                                                                                            im


'rxjs'; // Importar catchError
                                import Swal from 'sweetalert2'; // Import SweetAlert2
                                                                                  import { Cr


from '@angular/common';
                        import { SelectItem } from 'primeng/api';
                                                               import { FilterService } from


import { PedidosComponent } from '../pedidos/pedidos.component';
                                                               import { RecibosComponent } fr



header: string;
                }
                 @Component({
                                selector: 'app-analisiscaja',
                                                                templateUrl: './analisiscaja.comp


[FilterService, DialogService]
                                })
                                  export class AnalisiscajaComponent implements OnInit {
                                                                                          cols:


Cabecera;
           public selectedCabeceras: Cabecera[] = [];
                                                          sucursal: SelectItem[];
                                                                                 option2: Selec



selectedSucursal: any;
                         public cabeceras: any;
                                                    public clientes: Cliente[];
                                                                                totalSaldosSelecc


constructor(public dialogService: DialogService, private filterService: FilterService, private
_cargardata: CargardataService, private _router: Router) {
                                                                this.cols = [
                                                                                     { field: 'tip


{ field: 'numero_fac', header: 'N° Factura' },
                                                      { field: 'nombrecliente', header: 'Cliente


'vencimiento', header: 'Vencimiento' },
                                                 { field: 'basico', header: 'Basico' },
```

```
                                                                                { fie
'Bonifica' },
                  { field: 'interes', header: 'Interes' },
                                                          { field: 'saldo', header: 'Sa


this.getClientes();
                      this.sucursal = [
                                              { label: 'Suc. Valle Viejo', value: 2 },
                                                                                    {

    ];
        }
           get selectedColumns(): Column[] {
                                               return this._selectedColumns;
                                                                    }
                                                                       set selected

this._selectedColumns = this.cols.filter((col) => val.includes(col));
                                                                  }
                                                                     showPedidosDialog(su
this.dialogService.open(PedidosComponent, {
                                                data: {
                                                        cod_sucursal: sucursal,



'70%'
         });
               }
                  showRecibosDialog(sucursal: string, comprobante: string) {
                                                                        this.dialogServ


        comprobante: comprobante
                                     },
                                             header: 'Recibos',
                                                                width: '70%'
                                                                             });
                                                                                  }


localStorage.getItem('sucursal');
                                    let sucursal: string = localStorage.getItem('sucursal');


localStorage');
                  // Mostrar notificación al usuario
                                                        this.showNotification('Sucursal no er


this._cargardata.clisucx(sucursal).pipe(
                                                take(1),
                                                   catchError(err => {
                                                                        console

        // Muestra un mensaje al usuario
```

```
                                                // Puedes usar una librería como SweetAlert2 o

        this.showNotification('Hubo un error al obtener los clientes. Por favor, inténtalo nue

evitar errores en la asignación
                                    })
                                          ).subscribe((resp: any) => {
                                                                        if (resp && resp.

console.error('Respuesta inesperada al obtener clientes:', resp);
                                                                    this.showNotificatior

    });
         }
          /*   getClientes() {
                                     let sucursal: string = localStorage.getItem('sucursal');

=> {
          console.log(resp);
                                     this.clientes = resp.mensaje;
                                                              }, (err) => { console.log(e


sobre this.cabeceras y agregar el campo nombre
                                                    this.cabeceras.forEach(cabecera => {
                                                                                          cc

      console.log(cliente);
                                   if (cliente) {
                                                         cabecera.nombrecliente = cliente.nombre




  onSucursalChange(event: any) {
                                       this.selectedSucursal = event.value;
                                                                        console.log(this.s

this._cargardata.cabecerasuc(this.selectedSucursal).subscribe((data: any) => {
                                                                              console.lo

this.integrarNombreClienteaCabecera();
                                                 });
                                                      }
                                                         onSelectionChange(event: any) {
                                                                                          console.l


consultaRecibo() {
                         this.showRecibosDialog(this.selectedSucursal, this.selectedCabeceras[0].

this.showPedidosDialog(this.selectedSucursal, this.selectedCabeceras[0].numero_int.toString())
```

```
this.totalSaldosSeleccionados = this.selectedCabeceras
                                                    .reduce((sum, cabecera) => sum + N

showNotification(message: string) {
                                        Swal.fire({
                                                    icon: 'error',
                                                                title: 'Error',
```

components\analisispedidos\analisispedidos.component.ts

```
import { Component } from '@angular/core';

                                                                                @Component

analisispedidos.component.html',
                                styleUrls: ['./analisispedidos.component.css']
                                                                              })
                                                                                export class
```

components\cabeceras\cabeceras.component.ts

```
import { Component } from '@angular/core';
                                        import { Cabecera } from '../../interfaces/cabecera'


from 'primeng/table';
                     import { CargardataService } from '../../services/cargardata.service';
                                                                                          imp


{ ActivatedRoute, Router, NavigationEnd } from '@angular/router';
                                                                  import * as FileSaver from 'f


'rxjs/operators';
                  import Swal from 'sweetalert2'; // Import SweetAlert2
                                                                        import { CrudService } f


'@angular/common';
                   import { MotomatchBotService } from 'src/app/services/motomatch-bot.service'


from 'pdfmake/build/vfs_fonts';
                                import { Text } from '@angular/compiler';
                                                                          pdfMake.vfs = pdfFonts



cabeceras.component.html',
                             styleUrls: ['./cabeceras.component.css']
                                                                    })
                                                                      export class CabecerasCo


Cabecera;
           public clienteFromCuentaCorriente: any;
                                                    public selectedCabeceras: Cabecera[] = [];


null;
       public importe: number = null;
                                       public tipo: any[] = [];
                                                                filteredTipo: any[] = [];
                                                                                          pu


'';
     public listaPrecio: string = '';
                                       public activaDatos: number;
                                                                  public interes: number = 0



    Dni: 0,
               Numero: 0,
                               Autorizacion: 0
                                                 };
                                                     public cheque = {
                                                                       Banco: '',
                                                                                   Ncuenta:
```

```
ImporteCheque: 0,
                    FechaCheque: null
                                    };
                                        searchText: string;
                                                                public opcionesPagoFlag: bool

FechaCalend: any;
                  public inputOPFlag: boolean = true;
                                                        public puntoVenta_flag: boolean = true

"C"];
       public letraValue: string = "A";
                                          public tipoDoc: string = "FC";
                                                                  public puntoventa: nu


private myRegex = new RegExp('^[0-9]+$');
                                             public numerocomprobante: string;
                                                                    public cliente:


fecha_recibo: any;
                    public recibos: Recibo[] = [];
                                                    public numerocomprobanterecibo: number;
                                                                                          p

any;
      public puntoventaSelectedFormCabecera: any;
                                                   public cabecerasFiltered: any[] = [];
                                                                                       publ


  constructor(private bot: MotomatchBotService, private _crud: CrudService, private activatedR
_router: Router) {
                      this.getNombreSucursal();
                                                }
                                                   ngOnInit(): void {
                                                                        this.clienteFromCuer

    this.clienteFromCuentaCorriente = JSON.parse(this.clienteFromCuentaCorriente);
                                                                                    console.

localStorage.getItem('sucursal');
                                     this._cargardata.cabecerax(sucursal, this.clienteFromCuer

console.log(resp);
                       this.cabeceras = resp.mensaje;
                                                         }, (err) => { console.log(err); });


this._cargardata.tarjcredito().pipe(take(1)).subscribe((resp: any) => {
                                                                        this.tipo = resp.



    //-----------------------------------------------------
                                                          // get vendedores ------------
```

```
    //-----------------------------------------------------
                                                    //get clientes-----------------
JSON.parse(localStorage.getItem('datoscliente'));
                                                //---------------------------------------
sucursal-----------------------------------------
                                                this.sucursal = localStorage.getItem('suc

    //get usuario-------------------------------------------
                                                 this.usuario = localStorage.getI
    //-----------------------------------------------------
                                                }
                                                   getNombreSucursal() {
                                                                        this.
'1') {
            this.sucursalNombre = 'Casa Central';
                                                }
                                                     else if (this.sucursal == '2') {

(this.sucursal == '3') {
                            this.sucursalNombre = 'Suc. Guemes';
                                                                }
                                                                     else if (this.sucur


filterByDay() {
                    const dayOfWeek = new Date().getDay(); // Domingo - 0, Lunes - 1, ..., Sába

'd2', // Lunes
                     2: 'd3', // Martes
                                          3: 'd4', // Miércoles
                                                                4: 'd5', // Jueves


dayFieldMap[dayOfWeek];
                           this.filteredTipo = this.tipo.filter(item => item[dayField] === '1'


this.selectedCabeceras = event.sort((a: any, b: any) => {
                                                    const dateA = new Date(a.emitid

- dateB.getTime();
                       });
                              let selectedCabecerasIniciales = this.selectedCabeceras;
                                                                                    this.
console.log(this.selectedCabecerasIniciales);
                                                this.letraSelectedFormCabecera = this.selecte
```

```
this.selectedCabecerasIniciales[0].puntoventa;
                                                this.opcionesPagoFlag = this.evaluateEventTy


evaluateEventTypes(event: any[]): boolean {
                                                // Recorre cada objeto en el array event
                                                                                for


console.log(item.tipo);
                                // Verifica si la propiedad tipo es diferente de "FC", "ND", "NC"


propiedad tipo no coincide, devuelve falso
                                                this.tipoVal = "EFECTIVO";
                                                                              return fal


devuelve verdadero
                          return true;
                                       }
                                          calculateTotalSum(selectedCabeceras: any[]) {
                                                                                  this.t
parseFloat(cabecera.saldo.toString()), 0);
                                             }
                                                // New function to handle the payment
                                                                             pago() {


Swal.fire({
                     icon: 'error',
                                          title: 'Error',
                                                                  text: 'El saldo total es 0.',



      Swal.fire({
                           icon: 'error',
                                                title: 'Error',
                                                                       text: 'Por favor, ingres


(this.importe > this.totalSum) {
                                       Swal.fire({
                                                                icon: 'error',
                                                                                       title: 'Error',


      });
                 return;
                           }
                                if (this.tipoVal == 'Condicion de Venta') {
                                                                                        Swal.fire({


seleccione una condición de venta.',
                                          });
```

```
                return;
            }
            this.generarSalida();
        }
    a
completos los campos necesarios
    try {
        await this.getNumeroComprobanteCabecera();
= await this.ajuste(this.selectedCabeceras);
        // Extracted filtering logic into a separa
this.idAso();
    const psucursal = await this.generacionPagoPsucursal();
                const cabe
await this.generacionRecibo(this.selectedCabecerasIniciales);
                let pagoCC = {
                c
las cabeceras seleccionadas y los saldos ajustados
                    psucursal: psucursal, // aca tengo
objeto cabecera
                recibo: recibo // aca tengo el objeto recibo
                };
                this.er
generar los datos de pago:', error);
            }
        }
    }
    // Define the filtering function
            async f
paga nada de una cabecera
    this.cabecerasFiltered = [];
        cabeceras.forEach(cabecera => {
Number(cabecera.iva1)) {
        this.cabecerasFiltered.push(cabecera);
            }
        });
        consol

console.log(pagoCC);
    this._cargardata.pagoCabecera(this.sucursal, pagoCC).pipe(take(1)).sub
== "Operación exitosa") {
        this.generarReciboImpreso(pagoCC);
            Swal.fire({

this.incrementarNumeroComprobanteRecibo();
        }
        else {
            Swal.fire({
```

```
                                                                                icon
realizar el pago.',
                        });
                    }
                }
                        , (err) => { console.log(err); });
                                                                            }
                                                                               async g
fechaFormateada = fecha.toLocaleDateString('es-ES', {
                                                        day: '2-digit',
                                                                month: '2-digi


    let year = fecha.getFullYear();
                                    let month = fecha.getMonth() + 1;
                                                            let formattedMonth
(this.cabecerasFiltered.length > 1) {
                                        numero_int = 99999999;
                                                        this.numero_fac = 99999
1 ? this.cabecerasFiltered[0].numero_int : null;
                                                    this.numero_fac = this.cabecerasFiltered
console.log(numero_int);
                            console.log(this.numerocomprobantecabecera + 1);
                                                                //zona de corr
      this.cliente.idcli = 0;
                                }
                                     if (this.cliente.cod_iva == "") {
                                                                this.cliente.cod_


= 0;
        }
             if (this.codTarj == "") {
                                        this.codTarj = "0";
                                                            }
                                                                 let cabecera = { // es
this.numerocomprobanterecibo,//this.numerocomprobantecabecera + 1,
                                                                puntoventa: Number(thi
this.letraSelectedFormCabecera,//this.letraValue,
                                                    numero_fac: this.numero_fac,
                                                                        atipo
Number(this.cliente.idcli),
                                    cod_sucursal: Number(this.sucursal),
                                                                  emitido: fechaForma
```

```
parseFloat((this.importe / 1.21).toFixed(2)),//parseFloat((this.suma/1.21).toFixed(2)),//this.
1.21).toFixed(2)),//parseFloat((this.suma - this.suma/1.21).toFixed(2)),
                                                                    iva2: 0,
                                                                          iv


      saldo: 0,//this.suma,
                                dorigen: false,
                                                  cod_condvta: this.codTarj,
                                                                       cod_iva:
aca hay que ver si se agrega un campo para elegir el nombre del vendedor
                                                                    anulado: false,

es el que se logea?
                          turno: 0,
                                       pfiscal: `${year}${formattedMonth}`,
                                                                        mperc: 0,

formatDate(this.FechaCalend, 'dd/MM/yy', 'en-US'),//this.FechaCalend,//fecha de cierre con caj


id_aso: 0,
               }
                    console.log(cabecera);
                                            return cabecera;
                                                          }
                                                             async getNumeroComprobanteCa
this._cargardata.lastIdnum(this.sucursal).pipe(take(1)).toPromise();
                                                                  console.log('NUMERO SE
= Number(numero['mensaje']);
                                }
                                   async getNumeroComprobanteRecibo() {
                                                                    let numero = await th
console.log('NUMERO SECUENCIAL:' + numero);
                                              this.numerocomprobanterecibo = numero;
                                                                                  }
                                                                                     asyn
this._crud.incrementarNumeroSecuencial('recibo', this.numerocomprobanterecibo + 1).then(() =>

this.numerocomprobanterecibo = 0;
                                       });
                                            }
                                               async generacionPagoPsucursal() {
                                                                        let date = n
1) + "-" + date.getDate();
                                let hora = date.getHours() + ":" + date.getMinutes() + ":" + dat

console.log(Number(this.numerocomprobantecabecera) + 1);
                                                          let codtarj: number = 0;
                                                                              if (th
```

```
Number(this.codTarj);
                        }
                             let numero_int;
                                                  if (this.cabecerasFiltered.length > 1) {

this.cabecerasFiltered.length === 1 ? this.cabecerasFiltered[0].numero_int : null;
                                                                                                  }
                                                                                                       cor


cantidad: 0,//nada
                       precio: 0,//nada
                                                 idcli: this.clienteFromCuentaCorriente.idcli,

      fecha: fecha,
                        hora: hora,
                                               tipoprecio: "",//nada
                                                                                 cod_tar: codtarj,//this.c
necesario
               numerotar: this.tarjeta.Numero,//si es necesario
                                                                                 cod_mov: 0,
                                                                                                 suc_dest

this.tarjeta.Autorizacion,
                                  dni_tar: this.tarjeta.Dni,
                                                                        banco: this.cheque.Banco,


nombre: this.cheque.Nombre,
                                   plaza: this.cheque.Plaza,
                                                                        importeimputar: this.cheque.Im

fechacheque: this.cheque.FechaCheque,
                                              emailop: emailOp,
                                                                          tipodoc: "RC",
                                                                                                 puntover
numerocomprobante: numero_int,
                                        estado: "",
                                                                id_num: this.numerocomprobantecabecera +

console.log(psucursal1);
                             return psucursal1;
                                                  }
                                                                     async generacionRecibo(selectedCabecerasIni

selectedCabecerasIniciales.forEach((cabeceraInicial, index) => {
                                                                                  let importe = 0;
                                                                                                       le

this.currentSaldoArray[index]);
```

```
                                    if (this.currentSaldoArray[index] <= this.importe) {

else {
              importe = this.importe;
                                                saldo = this.currentSaldoArray[index] - this.impo

this.importe -= importe;
                                console.log("currentsaldo:" + this.currentSaldoArray);
                                                                                          let

numero_fac_cabecera = 0;
                                }
                                        else {
                                                    numero_fac_cabecera = Number(this.numero_fa

= {
           recibo: this.numerocomprobanterecibo, //generado con el serial desde firebase

factura de la cabecera fijo para el array
                                                  c_cuota: 0,//fijo
                                                                        fecha: this.fecha_re

importe,//Number(selectedCabecerasIniciales[index].saldo) - selectedCabeceras[index].saldo,//
usuario: this.usuario, //fijo para el array
                                                    observacion: 0,//fijo
                                                                            cod_lugar: '1'

this.letraSelectedFormCabecera,//this.letraValue, //letra de la cabecera fijo para el array
this.puntoventa,// punto de venta de la cabecera fijo para el array
                                                                             recibo_asoc: cabece
        recibo_saldo: saldo,//cabeceraActual.saldo,//---> el saldo para cada array
                                                                                          coc

fec_proceso: this.FechaCalend,//this.FechaCalend,//fecha de cierre con caja puede ser otro dia

        id_fac: this.numerocomprobantecabecera + 1// fijo para el array
                                                                                 };
                                                                                        if (recib



console.log(this.recibos);
                                return this.recibos;
                                                  }
                                                     calculoImporteRecibo(cabecera: any, rem

parseFloat(cabecera.saldo.toString());
                                              if (remainingImporte >= saldoCabecera) {
                                                                                          import
```

```
return importe;
                }
                   ajuste(selectedCabeceras: any[]) {
                                                        this.currentSaldoArray = [];
                                                                                    // V
this.totalSum) {
                      let remainingImporte = this.importe;
                                                            console.log("remainig importe:
y ajustar sus saldos
                          selectedCabeceras.forEach(cabecera => {
                                                                    if (remainingImporte
parseFloat(cabecera.saldo.toString());
                                                  if (currentSaldo <= remainingImporte) {


        } else {
                          cabecera.saldo = currentSaldo - remainingImporte;
                                                                              remaining


        console.log("currentsaldo:" + this.currentSaldoArray);
                                                                 });
                                                                          // Devolver el ar
console.log(selectedCabeceras);
                                      return selectedCabeceras;
                                                               } else {
                                                                           // Si el impor
situación como se prefiera
                                   console.error('El importe es mayor o igual al totalSum, ajuste




this.cabecerasFiltered.forEach(cabecera => {
                                                     cabecera.id_aso = this.numerocomprobantecabe
      cabecera.anumero_com = this.numerocomprobanterecibo;
                                                            cabecera.atipo = "RC";
                                                                                  });


this.totalSum) {
                      this.importe = this.totalSum;
                                                   }
                                                      }
                                                         selectTipo(item: any) {
                                                                                  consol
seleccionado
                this.codTarj = item.cod_tarj;
                                                 this.listaPrecio = item.listaprecio;
                                                                                   this.
```

```
this.abrirFormularioTarj();
                                // aca se llama a la funcion que muestra los prefijos
                                                                                    }


      // aca se llama a la funcion que muestra los prefijos
                                                                  }
                                                                     }
                                                                        abrirFormularioTarj() {


`<input type="text" id="titular" class="swal2-input" placeholder="Titular">
                                                                                  <input
             <input type="number" id="numero" class="swal2-input" placeholder="Número Tarjeta"
input" placeholder="Autorización">`,
                                                showCancelButton: true,
                                                                    confirmButtonText: 'Ace


        const titular = (<HTMLInputElement>document.getElementById('titular')).value;
                                                                                       c
        const numero = (<HTMLInputElement>document.getElementById('numero')).value;
                                                                                     cor
(<HTMLInputElement>document.getElementById('autorizacion')).value;
                                                                          if (!titular || !dni
Swal.showValidationMessage(`Por favor rellene todos los campos`);
                                                                           //return;
                                                                                        }


RegExp("^[0-9]{8}$");
                                let reTitular = new RegExp("^[a-zA-Z ]{1,40}$");
                                                                                let reAut
reNumero.test(numero)) {
                                     Swal.showValidationMessage(`El número de la tarjeta no es vá


          Swal.showValidationMessage(`El DNI no es válido`);
                                                                    //return;
                                                                                 }


Swal.showValidationMessage(`El titular no es válido`);
                                                                 //return;
                                                                              }
                                                                                    if (
Swal.showValidationMessage(`La autorización no es válida`);
                                                                    //return;
                                                                                 }


    }).then((result) => {
```

```
                                if (result.value) {
                                                        this.tarjeta.Titular = result.value.

this.tarjeta.Numero = result.value.numero;
                                                        this.tarjeta.Autorizacion = result.value.aut



      title: 'Ingrese los datos del Cheque',
                                                        html:
                                                                `<input type="text" id="banco"

id="ncuenta" class="swal2-input" placeholder="N° Cuenta">
                                                                   <input type="number" id="nch

type="text" id="nombre" class="swal2-input" placeholder="Nombre">
                                                                         <input type="text" i

type="number" id="importeimputar" class="swal2-input" placeholder="Importe a Imputar">

placeholder="Importe del Cheque">
                                                <input type="text" id="fechacheque" class="swal2-inp

Cambiar el tipo de input a 'date' para activar el datepicker nativo
                                                                        document.getElement

showCancelButton: true,
                              confirmButtonText: 'Aceptar',
                                                                cancelButtonText: 'Cancelar',

(<HTMLInputElement>document.getElementById('banco')).value;
                                                                   const ncuenta = (<HTMLInput

= (<HTMLInputElement>document.getElementById('ncheque')).value;
                                                                       const nombre = (<HTMLIn

= (<HTMLInputElement>document.getElementById('plaza')).value;
                                                                  const importeimputar = (<

        const importecheque = (<HTMLInputElement>document.getElementById('importecheque')).val

(<HTMLInputElement>document.getElementById('fechacheque')).value;

ncheque || !nombre || !plaza || !importeimputar || !importecheque || !fechacheque) {

          //return;
                              }
                                        let reBanco = new RegExp("^[a-zA-Z ]{1,40}$");
                                                                                  let

RegExp("^[0-9]{1,40}$");
                                    let reNombre = new RegExp("^[a-zA-Z ]{1,40}$");
                                                                              let reP

new RegExp("^[0-9]{1,40}$");
                                        let reImporteCheque = new RegExp("^[0-9]{1,40}$");
```

```
{1,40}$");
                if (!reBanco.test(banco)) {
                                                    Swal.showValidationMessage(`El nombre d

reNcuenta.test(ncuenta)) {
                                Swal.showValidationMessage(`El numero de cuenta no es váli

          Swal.showValidationMessage(`El numero de cheque no es válido`);
                                                                          //return;

Swal.showValidationMessage(`El nombre no es válido`);
                                                            //return;
                                                                  }
                                                                          if (!
es válida`);
                     //return;
                                }
                                          if (!reImporteImputar.test(importeimputar)) {
válido`);
                  //return;
                                }
                                        if (!reImporteCheque.test(importecheque)) {

          //return;
                           }
                                Swal.showValidationMessage(`El numero de cheque no es válido`);
                                                                          //return;

Swal.showValidationMessage(`El nombre no es válido`);
                                                            //return;
                                                                  }
                                                                          if (!
es válida`);
                     //return;
                                }
                                          if (!reImporteImputar.test(importeimputar)) {
válido`);
                  //return;
                                }
                                        if (!reImporteCheque.test(importecheque)) {

          //return;
                           }
                                return { banco, ncuenta, ncheque, nombre, plaza, importeim

      if (result.value) {
                                this.cheque.Banco = result.value.banco;
                                                                      this.cheque.Nc
result.value.ncheque;
                           this.cheque.Nombre = result.value.nombre;
                                                                  this.cheque.Plaz
result.value.importeimputar;
                                this.cheque.ImporteCheque = result.value.importecheque;

console.log('Cheque guardado:', this.cheque);
                                                        }
                                                             }).catch((error) => {
                                                                          this.showNo


let missingFields = [];
                           if (this.tipoDoc == "FC") {
                                                    if (!this.FechaCalend) {
                                                                              mi
```

```
missingFields.push('Vendedor');
                                    }
                                         }
                                              if (missingFields.length > 0) {
                                                                                  Swal.fire(

datos!',
                footer: 'Completar: ' + missingFields.join(', ')
                                                                          })
                                                                                  return false;




console.log(event.target.value);
                                     this.tipoDoc = event.target.value;
                                                                                  if (this.tipoDoc ==

      this.letras_flag = true;
                                   }
                                        else if (this.tipoDoc == "NC") {
                                                                                  this.inputOPFlag

      this.letras_flag = false;
                                    }
                                         else if (this.tipoDoc == "NV") {
                                                                                  this.inputOPFlag

      this.letras_flag = false;
                                    }
                                         else if (this.tipoDoc == "ND") {
                                                                                  this.inputOPFlag

      this.letras_flag = false;
                                    }
                                         else if (this.tipoDoc == "PR") {
                                                                                  this.inputOPFlag

      this.letras_flag = false;
                                    }
                                         else if (this.tipoDoc == "CS") {
                                                                                  this.inputOPFlag

      this.letras_flag = false;
                                    }
                                       }
```

```
                                    getVendedores() {
                                                this._cargardata.vendedores().su

console.log(this.vendedores);
                                })
                                    }
                                       validateValue(value: string): boolean {
                                                                    return this

=> {
          const worksheet = xlsx.utils.json_to_sheet(this.cabeceras);
                                                                const workbook = {

excelBuffer: any = xlsx.write(workbook, { bookType: 'xlsx', type: 'array' });
                                                                    this.saveAs


saveAsExcelFile(buffer: any, fileName: string): void {
                                                        let EXCEL_TYPE = 'application/vnd.op

EXCEL_EXTENSION = '.xlsx';
                                const data: Blob = new Blob([buffer], {
                                                                type: EXCEL_TYPE

Date().getTime() + EXCEL_EXTENSION);
                                    }
                                       generarReciboImpreso(pagoCC: any) {
                                                                        // Calcular la

pagoCC.recibo.reduce((sum, recibo) => sum + recibo.importe, 0);
                                                            let cliente = JSON.parse(lo


console.log(pagoCC);
                          let numeroenPlabras = this.numeroAPalabras(totalImporte);
                                                                        let fecha

fechaActual.toISOString().split('T')[0];
                                        console.log(fechaFormateada);
                                                          const tableBody =

parseFloat(item.basico);
                                const iva1 = parseFloat(item.iva1);
                                                              return [
                                                                        item.ti


item.cod_sucursal,
                            item.emitido,
                                        (basico + iva1).toFixed(2),
                                                                    item.saldo


documento
              const documentDefinition = {
                                        background: {
                                                              canvas: [
```

```
{
y: 10, // Posición Y del recuadro
w: 580, // Ancho del recuadro
esquinas (para redondearlas)
lineWidth: 1, // Grosor de la línea
lineCol
fillColor: '#000000', // Color de relleno (ejemplo: gris claro)
},
],
},
```


QSkZJRgABAQEAeAB4AAD/2wBDAAMCAgMCAgMDAwMEAwMEBQgFBQQEBQoHBwYIDAoMDAsKCwsNDhIQDQ4RDgsLEBYQERMUF
BQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBT/wAARCABeAPUDASIAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAE
II0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqDhIWGh
ufo6erx8vP09fb3+Pn6/8QAHwEAAwEBAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL/8QAtREAAgECBAQDBAcFBAQAAQJ3AAE
5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6goOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6w
tFACYopaQ9KAEozUU0yQxtJK4jRVLM7HAUDqSa+B/2sv+CoWj/DyfVPC3wwhg8SeILddk+tSENYWjZwQozmVh7DaCRk0h2
Cnw296VJVNa18eVA5HXZEDk+oyRn2r4y8LeA/jB+3F8TJXR77xXrUQKXWrajIFtNOUt8uCDtXHB2pk8YxX6LfA3/glR8O/
bG+MfxSl3a18RNYhtpGx9l01zaQbTywCqAcjHOWPSp/gT+zN8T/ANqLVnTQLe8uNIhJFzr2tTSfZEH3lKs5JZs5OB+dftD
Z0l0H4OXEmhXOnsjtpWjuLZ7q1X70EbjG0kdu9FmO99jvv2b/hBoX7N/wAJtJ8D2Guf2itoWlnvLq4GZZnwXKgn5VyOFr0
OlrHhnx9Hqc9rqukeLJNQQZkjkhumdJc4KkbcdM8jioG8GeLUUk6B4n2BsO39n3OwJ90YGzqe+efSi4+Rdz+jX+3NO/wCg
N9J4f8AEUUbefpXiGAlizSTW9wu1ePkOV+YDHPfkYrLubua2JWd760cKxRZJ5Y8KeAcNjAyRS5hcvY/pV/tzTuP+Jha89P
Y+/4J22f7SXwwj8a6z471rQ7e4meKCz09BgbQBvBb3z04r6Ft/+CQfgu1Gbf4k+MYZVBAeOZVIBqiT77jmSVcpIrj1Vs07
beH7RV1qPg7xdpS+G/iToyE3djgoswU4ZlVuVYHqOnIxQB9cfjS80wfeA79akpiExTdwBxnn60+uL+LXxV8M/BfwXf+KvP
gq98SfGV9LbfDrT7Pwdo7MyxXV5Gtxduo745VSR2r5y1r9rH40eKpCb74peIpJWOUjjuPIj/h7UrovlZ/QM11Cv3
wDr1hX2varqMzm91TUrnGXdbm+eUkdB949R7danmHyM/oh174yeBfC8Tyat4at4w0WwRPvGa+jGMe2a8k8Zf8ABQj4BeC42a5
Wvwn+xxcGRI7ibftFxMm5txHBJPJ+nSpI2MCubdGhUAOFU5ZVPBB9e5I6DilzXGon62+Kf+Cwfwv0+cp4d8L+JvEsBEYH+uE
HL4aaP4zsdNvNItNSTelpfDEi4OOvce9fjP+xb+yFqP7U3joP7ZHJaedBdJZXv8AUPL+WbDZWKJuhdu5HQGv3E0PQ7Dwvot
Jb/gol+3JqvjXxXqXwz8DahcWPhfS5mFrE52Fsur6hauUOoT5w1urjlVByD3JBr57/ZF/
ZR1v8Aag8fRaRYq8ZaRYY03wpphE2q6sHq8RKf+WKKerNzjPoSelcz+0p8LNe+Dvxm8Vaba4iOeW7ku7W4mJPn27ylw6N0ZsHGR
hLSodL0uzjCBUA3ykDl5G6sx6kn1rsOKanenVpsZhj2owPSlooAgaxtpJPMa3iaT++yAn86iQ5OTlRzUlFAFG40X
TotP0uyttOsYRtjtrSIRog9AAMCtCk20vSgCOXjB7fyr4F/bz+B+qfCvxVpX7SPw0tUj8ReH5hJrNmgwl1D0MhA6kDg/
XPavv1lDfX1qnq2k2esaZc6fe26XNncxtFLDIMq6kYIIoGfBGjf8FhPAdxYwNqfgfxLaXxUF47ZVlQH2dzAWVdVmjLY52MByM024/4LFe
Dd7K0iaXe3BjuLJmBzsOMMAehJBA4x3rxyH/AIJs/H+beB4St7bcGBP2xPlH+beB4St7bcGBP2xPlHAyvP3jjJ+tR7xWh9Kh9KeKP+Cy6qo6tbzL4b+GNy
pX/BLn486hrJ7YfN2Rc8dAp6jr716N4V/4I6E8Nrpl/4SDxzzpelRkYK6bkKK6bC07DPX7wXtRd7DTR8BMqtsG923FWRg
xC+MXhyDxF4F4l1VfAel3SjyLW5ti95NF1DleNgI5HfnNfcPwN/4JoX/4Jo/Cb4alb6xewXHjbPXbdvMiuNYx5ETdikPUG85ya+s
Fj8S79b4YIe5tEaNSOwUACvin9oz9lbx9zDrCweKLJLjSZd1nrtqpa2vqxkpdHVenIupuKBbn5T36V+6uE3wf4Z2Z2/2r4n0
G3xg1ZNN8EeGdQ16ZoziSBD5cXJXKl5cxXBSZkqhn5Tn1aaalbZp/NEr1KbKJ4Y6dr7bGCKKGrYRvFm+eKaVwmFaWmfy+Rqbn
h74es9B8O6Xb6NpFouyC0tUCoo9eOpPqea36Tjrr2TqokKKKKACkZsfX0paY2e1AA0nGQxNxrzH4v/tLfDb4jPxTZ6TPJ
bOP7OPhm18N+GJ4z481tcQyMm9dPgJwZ3GRz1C+4r8e44fE3xY8auwg1Txb4q1Q1qkeCWRSxB1G6ji7gkBhJ6M4cZYNn0r3q3Tw
GfwK8a6j4n0KdVOo6PqdlJA/lZ+aCRiAHbGSrexuPAn/AASp+Np6FqA7uC2xrUQeNatthr2Wx9+OdDoSgbr/BAqS9tCWNT8E
w9cfaNK1KBZU5y0bY+ZG/2lPB+lderbq+Iv8Agmp8HHPD3zgTvG+HH78a7QAQGvDCH2sUjB6/LthdmUZ8OqC/4Tje6uagU6v95hG
Wvgf9urwb+1F8UPGFPGsfh/4dFlfLcaVqSW1xcbfm807gwUdMA6699MMj1puDzgYNJq4H4a6t/wTd/Td/aFbzm78I2+oS
Zc+J3w7imuPE3wz1Szt7Y2Km6t7Bp414yGLIuZMZH+Nf0GCjmhW4RklLRZYz1VlBH69aLF8zF8zP6sF8zP6VmctF8zP6vBvjzxH4AvhN4A
hjivvb9l/8A4KsaszpOoWh/4wxR6nYSR8GVpGsWKQNEQEyOI5aEASLnBfwvrZ+PdhNNc6Fb+H/EQU+RrGmKQNzje
s2+OZvDPiO08y2ZW+x6lENsF7CTnK6C4ld7vD1hdIlYLiFgySKEtvsjSEKGtCvzE/4JNfNftHXs2pah
Pn+1J8SL34afS434rftAeVfKbr1LR3m7M2ABhsUbRiR4N/4j9oKJo3KGVMVYL8gw2ZKYwEMzf6R8P+/ZYZ6TXWdN8CgFNEQ1
pktuzGBraMxlxg7doxkdqhast6Kxo7fTrTxSYNKOKsgCtAFFIBQA6iiGkpALRCClpgFFFFABRRRSYoAW
AAUu+Flh8QP2Zda1J4I11PQSt9a3GwFhg4ZQ3UAjtX1wy18o/wDBS74S74nt1wFqnD5eqhxa7wJQLxnGlQhnD65
BW88TfGC98c3dmG0Pw7C0EFw6sUe4ddvvyMONwHUGv17NJD1uoOopKKok4z4tFjw/8E/A+oeLfjcjgzyRpvTkYGK1rotjg
Dng+a+1GGVrjxHeC2jlTTJKIISRkkeP/wAF/sf618ZJ8ZT8yY3cZGR/9apehSSCs4e9LtWZ8GwsdR0dO4qL8kQ9gkMbTwR/YO
f3On+JdLgWyYwWrPFOiELHrHIrDPwf8P/il4Ae71SLytav9R1R1hF8uAqNyxzV4Lhx1YD
AIR0zxH4u8RQaRY6nCs9nHKCZp1KhzBtCL1ss+OMa0OAhJdqIHmLATEx9PJyZo9/lb9F4RyAbzr5K5iR5S/Z1jmHt32J5gy3NpM9Xs
FvhuS32+YkizzY2kZLZzjkY9uOMUXYtD7z+Hv/BRL14Mf4VvFJFazcZSzX1RL2HtekLR6oxLA9iqMKGpY97Ixd7vz1vH2TWv4A4
jT8OHulu0WOBNYDS7gwG1AzE7w3A9+tfSX7V/8AwT5+K+Ksxwvx6C5+3Qoa0ScfJCKd5dMaXfNkAtrNDzOI9Py1uZ92Avo1YyzL
QJPk46nivgTUP+CfuuaaTqUlhqHxn+HOmnxKmHXbjVC0n9AE2hpcH9GmaTrrZeDMdN3vEdLiztT00a1j4ICvcrvyaTgYvjTxTHF
gqj8AWxjxBfYIJz9gkxwe03ftXzn+11+zDfeONf8G+GoviVv4B0PSPCOhW9imja7qYjkuApAp6W8MIwvHTzIJ65KWCxtFLN+T

AbxJrVvpsXiea2muJFjikurSSOMk8DLEYH419R2t5HeQRTwSpPBMoeOSM5VlIyCD6Yr8TNW/wCCfes6XfTaZf8Axq+Gtte
gL4P+DvD11qMeqXGn6bDbm7hfekm1QBtb+IY6H0pxdyT0Cvnb4vft6fCP4HeOrjwj4q1m6tdZgiWWSOK0d1VWBxyB7V9DN
AJSM8gd8cU27B6n6bfB/40eGfjp4Ng8UeEbxtQ0iZ2jEjLtYMDggg8jHvWX8df2j/A37OPh+01jxxqv9n213OLe3SNDJJI
wBlmz8SaL4w+LvgbVdHvytxaW+m6rGxjuB8rkgndyBz2zXjf7TnwV8VftLfFLUfEWofGr4Z2Olxkw6Tpg13cIIBjkjfgM3
s43ml2fjbWHsbzUgzQW8ELSvtXqxA6D3r59/4Jv/s2f8KR/wCEw8QXvjPw14v+2KluLzw/eC4hhVTuIdsnYeOg4xXiH7VX
wCCmXwM8Ua5p+kadr15PfX86W9vGbN1Du5wBk4HWvfPih8U9C+Dvgm+8V+J7hrLR7FQ08iruYD6CvyK8M/sL6n4T8eaVNd
bI+C3iv9oP4Gjwj4QvdNt7u7mt5pbq+dhEY1IY7dvXP5UXYSSRxS/8FUP2f2f2XK+I71vpYSe3tSn/gqf8AFyT4ivsA4z/Z8
wDgmN8QfG0c0vhr4k+CNfS1YLI+m6hLcCBiM4baxGT15pXY/dP0t+CP7Znw1/aE1nUNN8G6lcXk1hb/AGm5kmt2iSNc9y2
ErSNX8eeGvD+v+KlhsbfWbu9NvDDDk+YnmMd2459fpXg3hP/gmn4y8YzT2vhX4peAtdltgHlh0vUGmeP8A2mCscc+vXPNF
cwrjzblkIzuHRR71N/wAOj/jKpJXXxJ4bzwdxlm25z9RS+Fv7CvjT9nn4b/FSxPXqB8w8M6VqPiDShpdlqV1dNBaxB2G
DvWpru9ndYhJ9jdXvJ2+9LIxHU9eelfVnxO+JehfCDwVf+K/E3k7vW2YQXxLW1GDfEr1bfVnAR8lKa0spoP9nA0fBwjanBHx
Fz4c6jpemRSBYtP1dDJu2YU4LY4POetfUn7ZPibwx+0F8FbzwX4P+KngnTb7UJ0Es2pasiYlOWV5UhO1zQ0/wD4Kc
K3Jx1GPXBopi0NP8A4LGeKxrHj/wJ4T2rKLGzuL1othz+8KjdnPJHl8D3NfEutfCfVdC+G/hHxvLvZbdE8RXV1BZyrnZvhL
wACdF+If/BOfwj4VWUwa3Z28d9YX0sYZwHOh5Pm752ncR0rN6laH5weBfhCF4csW1DVrxmIi8scIoP953953
YKtPA+gRSNc6tqNkmqagRzczPy78n7ueAvYV5N/wTD8D2msfEnx4LyeSK4n8NyadFJu/ynlKq8zzHnjPBqirq1z5Viksf
CjXY9G+IGg3mowaVJHawRT72dhHtUBeM1+MvxO+Etx8K/HOs/zV/hyvbPQgcYrl18PCPLLbYAliCL
+uuagmk6LqF5K6xQ28DybzwFCqT/Svzj/4Jo/sz2/g++b4t6/qf23UYg02yskNZFPmPIzHLMQcD0r67/as+JsHhf8A
Gv6GtPs4tI0u2tIkCQWsKxKxIo6KqqAAPbAr8Mv2FPBaat+1T4ItNnFe2t7p7tvLBVm2ozBeD0JwcdOK/bqTxZZZTTQupjuAHU
wBpD4L+AfhL4a0OP4ieH1fT9JhiMaXYJLiIZwMdzmvyg/as+BM3we+M3iDSDqkeo6hf0XdXTahBJ5ZWVUlOcNzjcPUeprxqLw
vd/0pXdy3ZnWw28nxU+PW8I1xNrfiVtoJJ29tEsUYJydqA5r8lv+Cdf7LNv4i8aW
wAapEvcd4w1aLQfCur6lPzHa2skrfQKTX88Wg+FL342fFs6VpZk1fxJqMuucMzuWHGe+316Zr9s/2xviNi5DOf7M/
j64to5hcSabLDGxRcKWMvy6/4J7+I9p7UrqTQhgSFVeeRlsnPXAGSR83+4JPDjeG9e1fQ76Q76zW3VdW3H
hyGCHVW0NoRmS+3BAxOcbOAxHc9a+If27tQuPGX7V79V3jMlgq2rR2MKyA4ZfMe7V4BB4AyentU2ZfMe8x5H+zj/zj/
wAE19V1e0aO013xhTWdjEuWVZ2HQEmhfavkn9mHxx4R+EnxN3x4tR1L3VNqU8gEpw+d8P/Ko2+9s4NdvI3GcnNdv+094
DXiTxVquuPrOqwm6kGmyCOFAzPBPPCjNGorqzPg/4h+IdM8U/FnXP6xcznajgDnytZt6zzCPQ2vjE6YpyPbxtKGUdeScc/n2xX9C3w
br4B+L7eb4M+CmmFxLL/AGVbo7SSSUdS3Tjj2qokb6/O7/AILBEMm74u6fvxyFp+Hny3Fr+yTiYGDOy4OQfRT+tfSH/BJr
b+1qbx/8AtR+LbJpIo7NE1k0FkHICgk9o37H8Nn4E/ h
Zt8C6TmaVlsVmZkGV3Od5Ay3Tmi2o5M+Yv+CxniiOPUDwf4O+PPwf4G8Nkbhc3zXckZ6FUAHJzwOfSuD/4JX+P4PPX8x4r8V6R
wVW+E7eOvCeg+PtN1BbV9DkNtPaXUWRKknQQeMY5B4NfmLL4fLSM8pgkLbseZhv59DntRfUFZ7n74n9n74n9r74LJhT8TPDwx9j
AIKlfF7QfFn7OPh2Dw3rDvFvrGn6zqoH2i1O+OREB38gjocV+W8nhzbCQY7OQ7SAzWy56dz/njFfTf7UVgdF+DwfwL8I2JQWdDr
aksddufDd7pml22kOsTnUA/753GRjcvJH+ywfrH7LWqaVpniHVtL1e/1hDJELEMDGq4zznJ4v+rev0S/4JW2N+Nn4N+Auc
wD4KeeIpvGX7RbJARBbabYWHWxYoJG6Mwv0BmMeH0mMe7bapqCIOTBXeqIQeMnuT3or2f/AIJQfBDxbaa/dfHfBqqaHrHJY/6A1J

          margin: [0, 0, 80, 0], // izquierda, superior,derecha , inferior
                                                                                      },
                                                                                                {

{ text: 'Vicario Segura 587\n' },
                                                    { text: 'Capital - Catamarca\n' },

'3834-4172012\n' },
                                          { text: 'motomatch01@gmail.com' },
                                                                                       ],

          {
                         text: [
                                                   { canvas: [{ type: 'rect', x: 0, y: 0, w: 10
'\n', style: { fontSize: 40 }, margin: [10, 5, 0, 0] },//{ text: this.letraValue + '\n', style
'this.letra' represente la letra seleccionada en el campo 'letra'
                                                                                    { text: 'RECI

fontSize: 12,
                         },
                                   {
                                                 text: [

```
                                                                { text: 'RECIBO'
+ '\n', alignment: 'right' },
                                                  { text: 'Punto de venta: ' + pagoCC.recibo[0].c_p
              fontSize: 10,
                                      },
                                              ],
                                                      },
                                                               {
                                                                           text: 'Fecha
          margin: [25, 0, 5, 30],
                                                fontSize: 10,
                                                          },
                                                               {
                                                                              canvas: [
              x2: 380, y2: 0,
                                                lineWidth: 2,
                                                                lineColor: '#cccccc' //
margin: [0, 0, 30, 0] // Agregar un margen inferior a la línea
                                                                },
                                                                    {
                                                                                  col
'Recibimos de: ', bold: true },
                                                    { text: cliente.nombre + '\n' },
{ text: 'DNI: ' + cliente.dni + '\n' },
                                                           { text: 'CUIT: ' + cliente.cuit + '\n'
'\n' },
                      ],
                                        fontSize: 10,
                                                                margin: [0, 10, 0, 10],



              type: 'line',
                                     x1: 0, y1: 0,
                                                                    x2: 380, y2: 0,
para la línea
                      }
                                    ],
                                                        margin: [0, 0, 30, 20] // Agregar un margen ir
```

```
widths: ['*'],
                        body: [
                                [
                                        {
                                                text: [
numeroenPlabras + '\n', },
                                { text: '\n' },
                                                { text: 'Rete
{ text: '\n' },
                        { text: 'Neto a Cobrar:          ' + numeroenPlabras + '

                ],
                           style: 'total',
                                                alignment: 'left',


            ]
                    },
                        layout: {
                                        hLineWidth: function (i, node) {

            },
                    vLineWidth: function (i, node) {
                                                return (i === 0 || i =
hLineColor: function (i, node) {
                                        return '#000000';
                                                },
                                                        vLine

          }
                    },
                  // ... Aquí puedes seguir añadiendo más elementos al documento ..

del Pago' + '\n',
                     alignment: 'center',
                                        bold: true,
                                                table: {

              ['Tipo', 'Letra', 'Comprobante', 'Sucursal', 'Emitido', 'Importe', 'Saldo'],


            bold: true,
                            },
                                margin: [0, 0, 0, 30],
                                                },
                                                        {
```

```
            body: [
                                ['TOTAL APLICADO: $' + totalImporte],
                                                                    ],


0, 0, 30], // Añade un margen inferior
                                                },
                                                        {
                                                                text: 'Recibí conforme: \n

          margin: [0, 10, 0, 0],
                                            },
                                                ],
                                                        styles: {
                                                                    header: {
                                                                                fo

        },
                tableExample: {
                                        margin: [0, 5, 0, 5],
                                                                     fontSize: 8,


margin: [0, 10, 0, 0],
                                },
                                        },
                                                defaultStyle: {
                                                                },
                                                                        };
                                                                              Swal.close();
'_RECIBO_' + fechaFormateada + '.pdf');
                                        pdfMake.createPdf(documentDefinition).getBlob((blob
pagoCC.recibo[0].recibo + '_RECIBO_' + fechaFormateada + '.pdf');
                                                                    }, (error: any) => {



    const unidades = ['CERO', 'UNO', 'DOS', 'TRES', 'CUATRO', 'CINCO', 'SEIS', 'SIETE', 'OCHO'
'CATORCE', 'QUINCE', 'DIECISÉIS', 'DIECISIETE', 'DIECIOCHO', 'DIECINUEVE'];
                                                                              const decenas =
'SETENTA', 'OCHENTA', 'NOVENTA'];
                                       const centenas = ['', 'CIEN', 'DOSCIENTOS', 'TRESCIENTOS'
'OCHOCIENTOS', 'NOVECIENTOS'];
                                    if (num < 10) return unidades[num];
                                                                    if (num < 20) return
```

```
return decenas[Math.floor(num / 10)];
                                        return decenas[Math.floor(num / 10)] + ' y ' + unid


return centenas[Math.floor(num / 100)];
                                          return centenas[Math.floor(num / 100)] + ' ' + th


% 1000 === 0) return unidades[Math.floor(num / 1000)] + ' MIL';
                                                                  return unidades[Math.floo


(num < 1000000) {
                      if (num % 1000 === 0) return this.numeroAPalabras(Math.floor(num / 1000
' MIL ' + this.numeroAPalabras(num % 1000);
                                             }
                                                  if (num < 1000000000) {
                                                                        if (num % 100
MILLÓN';
              return this.numeroAPalabras(Math.floor(num / 1000000)) + ' MILLÓN ' + this.numer


showNotification(message: string) {
                                     Swal.fire({
                                                 icon: 'error',
                                                                 title: 'Error',
```

components\calculoproducto\calculoproducto.component.ts

```
import { Component, Inject, OnDestroy } from '@angular/core';
                                                        import { ActivatedRoute, Router }
DynamicDialogConfig } from 'primeng/dynamicdialog';
                                                 import { CarritoService } from 'src/app/ser


templateUrl: './calculoproducto.component.html',
                                                styleUrls: ['./calculoproducto.component.css


clienteFrompuntoVenta: any;
                                                     public producto: any;
                                                                        public cliente:



public codTarj: any
                    public listaPrecio: any;
                                                                   public pedido: any =



'idven': 0,
               'fecha': "",
                                'hora': "",
                                                'tipoprecio': "",
                                                                'cod_tar': 0,
                                                                                'titularta



'nomart': "",
                'nautotar': 0,
                                    'dni_tar': 0,
                                                    'banco': "",
                                                                     'ncuenta': 0,
                                                                                    'ncheque



'importecheque': 0,
                        'fechacheque': '01/01/1900'
                                                };
                                                     public precio: number;
                                                                         public precioT


any;
      public tipoPrecioString: string = '';
                                               constructor(private _carrito: CarritoService, pri
@Inject(DynamicDialogConfig) public config: DynamicDialogConfig) {
```

```
                                                                console.log("constructor

this.config.data.cliente;
                            this.tarjeta = this.config.data.tarjeta;
                                                                    this.cheque = this.co

this.codTarj = this.config.data.codTarj;
                                            this.listaPrecio = this.config.data.listaPrecio;

console.log("producto:" + JSON.stringify(this.producto));
                                                            console.log("cliente:" + JSON.str

JSON.stringify(this.tarjeta));
                                  console.log("cheque:" + JSON.stringify(this.cheque));
                                                                                    cor

console.log("codTarj:" + JSON.stringify(this.codTarj));
                                                        console.log("listaPrecio:" + JSON.s

    //case dependiente de this.listaPrecio en caso de 0 this.producto.precon , si es 1 this.pr
this.producto.prefi3, si es 4 this.producto.prefi4
                                                    switch (this.listaPrecio) {
                                                                            case "0

this.precio = this.producto.precon;
                                          break;
                                                     case "1":
                                                                this.tipoPrecioString

break;
            case "2":
                            this.tipoPrecioString = 'Precio de Tarjeta';
                                                                        this.precio =

= this.producto.prefi3;
                               break;
                                          case "4":
                                                        this.precio = this.producto.prefi4




this.ref.close();
                    }
                       }
                             calcularPrecioTotal(newValue: number) {
                                                                    console.log(newValue)

console.log(this.precioTotal);
                                }
                                      comprar(event: Event) {
```

```
                                                                event.preventDefault();
                                                                                        th

this._carrito.agregarItemCarritoJson('carrito', this.pedido);
                                                                this.precioTotal = 0;
                                                                                    this




es para que se muestre en la factura , presupuesto, etc el nombre del cliente
                                                                                localStorage.

new Date();
                let fecha = date.getFullYear() + "-" + (date.getMonth() + 1) + "-" + date.getDa

date.getSeconds();
                      console.log("PRECIO:" + this.precio);
                                                              //completar los datos del objet

(this.producto.idart != undefined) {
                                          this.pedido.idart = parseInt(this.producto.idart);


= this.producto.nomart;
                            }
                                 this.pedido.cantidad = this.cantidad;
                                                                    this.pedido.precio =

el carrito me va sumado
                             if (this.cliente.idcli != undefined) {
                                                                          this.pedido.idcli = pa

undefined) {
                   this.pedido.idven = this.cliente.cod_ven;
                                                              }
                                                                   if (this.cliente.fecha !=


(this.cliente.hora != undefined) {
                                        this.pedido.hora = hora;
                                                                  }
                                                                       if (this.listaPrecio


if (this.codTarj != undefined) {
                                       this.pedido.cod_tar = parseInt(this.codTarj);
                                                                                        }
                                                                                             if

this.tarjeta.Titular;
                          }
                               if (this.tarjeta.Numero != undefined) {
                                                                            this.pedido.numerot

(this.tarjeta.Autorizacion != undefined) {
                                                    this.pedido.nautotar = parseInt(this.tarjeta.A
```

```
this.pedido.dni_tar = parseInt(this.tarjeta.Dni);
                                                    }
                                                         if (this.cheque.Banco != undefined)

(this.cheque.Ncuenta != undefined) {
                                              this.pedido.ncuenta = parseInt(this.cheque.Ncuenta);

this.pedido.ncheque = parseInt(this.cheque.Ncheque);
                                                      }
                                                           if (this.cheque.Nombre != undefir

(this.cheque.Plaza != undefined) {
                                           this.pedido.plaza = this.cheque.Plaza;
                                                                                  }
                                                                                       if (thi
= parseInt(this.cheque.ImporteImputar);
                                             }
                                                  if (this.cheque.ImporteCheque != undefined) {

    }
         console.log("FECHA CHEQUE:" + this.cheque.FechaCheque);
                                                                  if (this.cheque.FechaChequ
this.pedido.fechacheque = this.cheque.FechaCheque;
                                                      }
                                                           else {
                                                                   this.pedido.fechacheque
```

```
import { Component } from '@angular/core';
                                          //agregar importacion de router para navegacion
                                                                                        impor

app/services/carrito.service';
                              import { SubirdataService } from 'src/app/services/subirdata.ser

cargardata.service';
                     import Swal from 'sweetalert2';
                                                    import { first, take } from 'rxjs/operators

{ set } from '@angular/fire/database';
                                      import { MotomatchBotService } from 'src/app/services/mo

import * as pdfFonts from 'pdfmake/build/vfs_fonts';
                                                     pdfMake.vfs = pdfFonts.pdfMake.vfs;
                                                                                        import

{ formatDate } from '@angular/common';
                                      interface Cliente {
                                                           nombre: string;
                                                                           direccion: string;









selector: 'app-carrito',
                         templateUrl: './carrito.component.html',
                                                                  styleUrls: ['./carrito.com



undefined;
            public FechaCalend: any;
                                      public itemsEnCarrito: any[] = [];
                                                                         public suma: number


  public numerocomprobanteImpresion: string;
                                                 public puntoventa: number = 0;
                                                                                 private myRegex

sucursalNombre: string = '';
                              private indiceTipoDoc: string;
                                                              public inputOPFlag: boolean = tr

boolean = true;
                 public letras: any = ["A", "B", "C"];
                                                         public letraValue: string = "A";
                                                                                          publ

any;
      public usuario: any;

                                                    constructor(private _cargardata: CargardataS
```

```
private _subirdata: SubirdataService, private _carrito: CarritoService, private router: Router

this.calculoTotal();
                        this.getNombreSucursal();
                                                this.getVendedores();
                                                                        this.usuario = l
JSON.parse(localStorage.getItem('datoscliente'));
                                                this.initLetraValue();
                                                                    }
                                                                       getItemsCarrit


      this.itemsEnCarrito = JSON.parse(items);
                                                }
                                                   }
                                                      getVendedores() {
                                                                        this._cargardata.
res.mensaje;
                  console.log(this.vendedores);
                                                 })
                                                    }
                                                       getNombreSucursal() {
                                                                              this.sucurs

      this.sucursalNombre = 'Casa Central';
                                              }
                                                   else if (this.sucursal == '2') {
                                                                                     this

'3') {
             this.sucursalNombre = 'Suc. Guemes';
                                                }
                                                     else if (this.sucursal == '4') {



(this.cliente.cod_iva == 2)//consumidor final
                                                { this.letraValue = "B"; }
                                                                          else if (this.c


else if (this.cliente.cod_iva == 3)//monotributo
                                                {
                                                       this.letraValue = "A";
                                                                              }
                                                                                   else



console.log(event.target.value);
```

```
                                this.tipoDoc = event.target.value;
                                                                    if (this.tipoDoc ==

      this.letras_flag = true;
                                }
                                     else if (this.tipoDoc == "NC") {
                                                                          this.inputOPFlag

      this.letras_flag = false;
                                }
                                     else if (this.tipoDoc == "NV") {
                                                                           this.inputOPFlag

      this.letras_flag = false;
                                }
                                     else if (this.tipoDoc == "ND") {
                                                                           this.inputOPFlag

      this.letras_flag = false;
                                }
                                     else if (this.tipoDoc == "PR") {
                                                                           this.inputOPFlag

      this.letras_flag = false;
                                }
                                     else if (this.tipoDoc == "CS") {
                                                                           this.inputOPFlag

      this.letras_flag = false;
                                }
                                   }
                                     eliminarItem(item: any) {
                                                                 //agregar un sweet alert

seguro?',
               text: "Vas a eliminar un item del carrito!",
                                                              icon: 'warning',
                                                                           showCar

cancelButtonColor: '#d33',
                                confirmButtonText: 'Si, eliminar!'
                                                                 }).then((result) => {


          'El item fue eliminado.',
                                          'success'
                                                              )
                                                                       let index = this.itemsE
```

```
        localStorage.setItem('carrito', JSON.stringify(this.itemsEnCarrito));
                                                                                this._car
del header
                  this.calculoTotal();
                                        }
                                             })
                                                 }

                                                       calculoTotal() {
                                                                            this.suma = 0;

item.precio * item.cantidad;
                                    }
                                       }
                                          async finalizar() {
                                                                 if (this.itemsEnCarrito.length >


          title: 'Enviando...',
                                             allowOutsideClick: false,
                                                                       });
                                                                                  this.indi

console.log('PUNTO VENTA:' + this.puntoventa);
                                                    if (this.tipoDoc == undefined || this.ti
undefined || this.tipoDoc == "" || this.numerocomprobante == undefined || this.numerocomprobar
        {
                   Swal.fire({
                                        icon: 'error',
                                                               title: 'Error..',

campos'
                })
                            return;
                                        }
                                                 else {
                                                                  if (this.tipoDoc == "ND")

this.numerocomprobante = this.numerocomprobante; //numero.toString();
                                                                                      }
                                                                                          else

            this.numerocomprobante = this.numerocomprobante; //numero.toString();
                                                                                           }

= "notacredito";
                              this.numerocomprobante = this.numerocomprobante;//numero.toString(


this.indiceTipoDoc = "devolucion";
                                                                                    this.numerocom

          else if (this.tipoDoc == "PR") {
```

```
                                                    this.indiceTipoDoc = "presupuesto";
this._crud.getNumeroSecuencial('presupuesto').pipe(take(1)).toPromise();
                                                                        console.lo
numero.toString();
                            }
                                        else if (this.tipoDoc == "CS") {
                                                                        this.indice
this._crud.getNumeroSecuencial('consulta').pipe(take(1)).toPromise();
                                                                    console.log('
numero.toString();
                            }
                                        let emailOp = localStorage.getItem('emailOp');


              ...obj,
                                  emailop: emailOp,
                                                        tipodoc: this.tipoDoc,

this.numerocomprobante,
                                    estado: "NP",
                                                    idven: this.vendedoresV,


this.numerocomprobante;
                                localStorage.setItem('carrito', JSON.stringify(result));

localStorage.getItem('sucursal');
                                        let exi = 0;  // ESTO LO HAGO POR QUE NO HAY CORRES

            exi = 3;
                            }
                                    else if (sucursal == "3") {
                                                                exi = 4;



          }
                    this._subirdata.editarStockArtSucxManagedPHP(result, exi).pipe(take(1)).s

          this.agregarPedido(result, sucursal);
                                                    }
                                                          }
                                                               }
                                                                    else {
                                                                        Swal.fire(


items en el carrito!',
                                footer: 'Agregue items al carrito'
                                                                })
                                                                      }
                                                                         }
```

```
                                                                    cabecera(fec
fechasinformato.getFullYear();
                                  let month = fechasinformato.getMonth() + 1;
                                                                  let formatted

    if (this.tipoDoc != "FC") {
                                      numero_fac = 0;
                                                  }
                                                       else {
                                                              numero_fac = Number(t

saldo = this.sumarCuentaCorriente();
                                     let cabecera = {
                                                        tipo: this.tipoDoc,
                                                                          numero_

letra: this.letraValue,
                              numero_fac: numero_fac,
                                                   atipo: this.tipoDoc,
                                                                       anumero_co

this.sucursal,
                     emitido: fecha,
                                      vencimiento: fecha,
                                                          exento: 0,
                                                                  basico: pars
parseFloat((this.suma - this.suma / 1.21).toFixed(2)),
                                                    iva2: 0,
                                                             iva3: 0,
                                                                      bonifi

true,
           cod_condvta: codvent,
                                     cod_iva: this.cliente.cod_iva,
                                                                  cod_vendedor: this.v
nombre del vendedor
                         anulado: false,
                                              cuit: this.cliente.cuit,
                                                                usuario: this.usua

`${year}${formattedMonth}`,
                                   mperc: 0,
                                              imp_int: 0,
                                                                fec_proceso: this.FechaCalend
this.FechaCalend,//fecha de cierre con caja puede ser otro dia   ?
                                                       fec_ultpago: null,



console.log(cabecera);
                          return cabecera;
                                        }
                                           sumarCuentaCorriente(): number {
```

```
                                                                    console.log

this.itemsEnCarrito) {
                          console.log(item);
                                              if (item.cod_tar === 111) {
                                                                    acumula
campo 'valor' que queremos sumar
                                    }
                                         }
                                               return acumulado;
                                                              }
                                                                    getCodVta() {
                                                                          if (

firstCodTar = this.itemsEnCarrito[0].cod_tar;
                                               for (let item of this.itemsEnCarrito) {



return firstCodTar;
                    }
                          agregarProductos() {
                                                 window.history.back();
                                                                 }
                                                                    validateValue(va

agregarPedido(pedido: any, sucursal: any) {
                                              let fecha = new Date();
                                                                 let fechaFormateada

'2-digit',
                year: 'numeric'
                                        });
                                                 let cabecera = this.cabecera(fechaFormateada, fecha)
sucursal).pipe(take(1)).subscribe((data: any) => {
                                                        console.log(data.mensaje);
                                                                          this.i
this.suma);
                 //actualizar indices
                                               if (this.indiceTipoDoc != "") {
                                                                       this._crud.i
parseInt(this.numerocomprobante) + 1).then(() => {
                                                            console.log('Numero secuencial inc



      Swal.fire({
                        icon: 'success',
                                               title: 'Pedido enviado',
                                                                    text: 'El ped
sucursal ' + localStorage.getItem('sucursal')
                                                 })
```

```
                                                    this.itemsEnCarrito = [];
                                                                                loca
this._carrito.actualizarCarrito(); // es para refrescar el numero del carrito del header




= [];
        if (this.tipoDoc == "FC") {
                                        if (!this.FechaCalend) {
                                                                missingFields.push('
missingFields.push('Vendedor');
                                    }
                                      if (!this.puntoventa) {
                                                                 missingFields.push(
missingFields.push('Numero de Comprobante');
                                                }
                                                     }
                                                          else if (this.tipoDoc == "NC" || t
this.numerocomprobante) {
                                  missingFields.push('Número de comprobante');
                                                                          }
                                                                                 if (


    else if (this.tipoDoc == "PR" || this.tipoDoc == "CS") {
                                                                 if (!this.vendedoresV) {


(missingFields.length > 0) {
                                    Swal.fire({
                                                    icon: 'error',
                                                                 title: 'Error..',
missingFields.join(', ')
                                 })
                                        return false;
                                                     }
                                                          else {
                                                                    return true;
                                                                                }
                                                                                    /


imprimir(items: any, numerocomprobante: string, fecha: any, total: any) {
                                                                       //let cliente = J
```

```
try {
       const datosCliente = localStorage.getItem('datoscliente');
                                                                  if (datosCliente) {
                                                                                      clie


'',
         direccion: '',
                           dni: '',
                                      cuit: '',
                                                  tipoiva: ''
                                                               };
                                                                   }
                                                                    } catch (error)

console.error('Error parsing datoscliente from localStorage', error);
                                                                    cliente = {
                                                                                  nombre:




let titulo: string = "";
                           if (this.tipoDoc == "FC") {
                                                       titulo = "FACTURA";
                                                                           }
                                                                                else


else if (this.tipoDoc == "NV") {
                                     titulo = "DEVOLUCION";
                                                            }
                                                                 else if (this.tipoDoc ==


== "PR") {
                titulo = "PRESUPUESTO";
                                       }
                                            else if (this.tipoDoc == "CS") {
                                                                            titulo =


fechaFormateada = fechaActual.toISOString().split('T')[0];
                                                            console.log(fechaFormateada);

item.precio, item.cantidad * item.precio]);
                                               // Definir el contenido del documento
                                                                                    const



            type: 'rect',
                                  x: 10, // Posición X del recuadro
                                                                      y: 10, // Po

            h: 750, // Alto del recuadro
```

```
                                        r: 3, // Radio de las esquinas (para redon

lineColor: '#000000', // Color de la línea (ejemplo: azul)
                                                     fillColor: '#000000', //



    content: [
                {
                        image: "
```

FRUVDA8XGBYUGBIUFRT/2wBDAQMEBAUEBQkFBQkUDQsNFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
ECAwQFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQAAAF9AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkK
h4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8
ECAxEEBSExBhJBUQdhcRMiMoEIFEKRobHBCSMzUvAVYnLRChYkNOEl8RcYGRomJygpKjU2Nzg5OkNERUZHSElKU1RVVldY
wsPExcbHyMnK0tPU1dbX2Nna4uPk5ebn6Onq8vP09fb3+Pn6/9oADAMBAAIRAxEAPwD9UuaMUtFACYopaQ9KAEozUU0yQx
DyfVPC3wwhg8SeILddk+tSENYWjZwQozmVh7DaCRk0h2ufZ3xL+LXhD4QaDJrXjHxDZaBpycebeTBSx9FHUn2Ar4B+NX/A
jB+3F8TJXR77xXrUQKXWrajIFtNOUt8uCDtXHB2pk8YxX6LfA3/glR8O/A8cGoePbubx9rvyO8c2YrGNx/diH3hn+9S1Y9
ANqLVnTQLe8uNIhJFzr2tTSfZEH3lKs5JZs5OB+dftD42/ZT+E/xA8O2uhat4I0r+zLadbiOGzgW32uDkcrjIPcd64L9tD
wAJtJ8D2Guf2itoWlnvLq4GZZnwXKgn5VyOFr07+3NNCknUbXHr56/41/OlrHhnx9Hqc9rqukeLJNQQZkjkhumdJc4Kkbo
wBf8aP7c04/8xC1/wC/6/41/N9J4f8AEUUbefpXiGAlizSTW9wu1ePkOV+YDHPfkYrLubua2JWd760cKxRZJ5Y8KeAcNjA
Y+/4J22f7SXwwj8a6z471rQ7e4meKCz09BgbQBvBb3z04r6Ft/+CQfgu1Gbf4k+MYZVBAeOZVIBqiT77jmSVcpIrj1Vs07
beH7RV1qPg7xdpS+G/iToyE3djgoswU4ZlVuVYHqOnIxQB9cfjS80wfeA79akpiExTdwBxnn60+uL+LXxV8M/BfwXf+KvF
gq98SfGV9LbfDrT7Pwdo7MyxXV5Gtxduo745VSR2r5y1r9rH40eKpCb74peIpJWOUjjuPIjQMM5AU/h7UrovlZ/QM11Cv3
wDr1hX2varqMzm91TUrnGXdbm+eUkdB949R7danmHyM/oh174yeBfC8Tyat4wWRPvGa+jGMe2a8k8Zf8ABQj4BeC42a5
Wvwn+xxcGRI7ibftFxMm5txHBJPJ+nSpI2MCubdGhUAOFU5ZVPBB9e5I6DilzXGon62+Kf+wv0+cp4d8L+JvEsYH+uE
HL4aaP4zsdNvNItNSTelpfDEi4OOvce9fjP+xb+yFqP7U3juP7ZHJaeBdJZXv8AUPL+WbDZWKJuhdu5HQGv3E0PQ7Dwvot
Jb/gol+3JqvjXXxXqXwz8DahcWPhfS5fsur6hauUOoT5w1w1urjlVByD3JBr57/ZF/
ZR1v8Aag8fRaRYY03wpphWTXNVI+WKKerNzjPoSelcz+0p8+Dvxm8VaD4hs7iOeW7ku7W4mJPn27ylw6N0ZsHGR
hLSodL0uzjCBUA3ykDl5G6sx6kn1rsOKanenVpsZhj2owPSlooAgaxtpJPMa3iaT++++UBP509reJs5U2aoI9pFR
TotP0uyttOsYRtjtrSIRog9AAMCtCk20vSgCOXjB7fyr4F/bz+B+qfCvFvxVpX7TSPw0tUj8ReH5hJrNmgwl1D0M
XPavv1lDfX1qnq2k2esaZc6fe26XNrcxNNFLFADxxYwNqfgfxLaxuF47VmjLxxF47dvVmjLY52MByM024/4LFe
Dd7K0iaXe3BjuLJmBzsOMMAehJB4x4x3rxyH/AIJs/H+beB4St/bGP2xPl...ddfyBBb1VwYMHRanOmJk2Wm/d...
pX/BLn486hKrS2Gh6Wd7Nte9JRc8dAp6jr716N4V/4I6+Nrpl/4SDxzpelRkYK6bC07DPX7wXtRd7DTR8BMqtsG923FWRg
xC+MXhyDxF4l1VfAel3SjyLW5ti95NF1DleNgI5HfnNfcPwN/4Jo/Cb4Oalb6xewXHjPXbdvMiuNYx5ETdikPIUg85ya+s
Fj8S79b4YIe5tEaNSOwUAcGvin9oz9lbx9+zDrCweKLLjSZJd1nrtqpa2l5xhifuHbn5T36V+6+ufE3wf4Z2/2r4n0jTu
G3xg1ZNN8EeGdQ16ZoziSBD5cXJKl5D8qkE5+hr7x+BP/BI28uriz1X4X4sa3HHbK3mPoOkE7jnBKvL745xX6N+B7Xwlp+kx
h74es9B8O6Xb6NpFouyC0tUCoo9eOpPqea36TjrTqokKKKKACkZsfX0paY2e1AA0nGQNxrzH4v/tLfDb4Fwo3jPxTZ6TPJ
bOP7OPhm18N+GJ4z481tcQyMm9dPgJwZ3GRz1C+4r8e44fE3xY8auwg1Txb4auNxb4o1WcNtVGuZZ53LAkNk/KACYxjioZSjc/Sf4
GfwK8a6j4n0KdVOo6PqdlJA/lZ+aCRiAHbGSrexruPAn/AASp+N3xV+vSWvn1AWbypRdjjGX3+V5xc8ICMsfzeRC+a29c/wCCCS
w9cfaNK1KBZU5y0bY+ZG/2lPB+derbq+Iv8zgmp8Hfi38GPH+kjwGPH+kjS9D+0Jc6an7DGr7K8cEhS4Ckxf4fvbW8HtkRG
Wvgf9urwb+1F8UPhSh4/4dRHTfAPlfLcaVqSW1xcsIAyVSOXOxf9nrXhXqPBzXO5/ZvF07+i/k2I2h3x2jG0Eufe9vHsDp3sWSMzR
Zc+J3w7imuPE3wz1SztyY2Km6t7BtpT3H4bL1WEvH7P5vvbvBjvjxjHn4H4AvhN4d1/UdA/YmL2I8V90gFm7I2+oS1xX/vvW3
hjivvb9l/8A4KsagRrePa1RNYoSMIV8/4wzR6nYSMIV8TWKBWi5AVpkHBHqRyO9fYR/MmGzRiNNfHwvzjfF0GvyScn+E3xjm3
s2+OZvDPiO08y2ZZW+x6lENsF7TnK9cN0t29ahIYLiFgySKEGSn8zE/4NnFtHXs2pahkHxSfFpZZSPKOKcwPzM/KWlVasi
Pn+1J8SL34rftAeoL+0f2Lhf8/ksffPcnzSMKmAY4qpOHNWfg1zfR6Kk0nhxPF3B+Rl7oAI49Pr3kVR6kDuYYXiQ4MhKEnKmrmR
pktuzGBraMxlxg7doxkdgqhastG6Kxo7fTrTxSYNKOKsgCtAFFIBQA6iikpALRSClpgFFFFABRRRQAUUUAFFFFABRRSYoAW
AAUu+Flh8QP2Zda1J4I11PQSt9a3GwFhg4ZQ3UAjtX1wy18o/wDBS74gWn3lnxDaSuv2zWGSxtoS21MmI/AGkxo/MP
BW88TfGC98c3dmG0Pw7C0EFw6sDVyGLWdV7NJDluOopKKok4z4tfFjw/8ADLw/ZAAx+7W/8oxHoH8k7pOBgm7fbhLw5AqNPW
Dng+a+1GGVrjxHeC2jljTJKISRkkeH/P/wAF+6f8FzZp...LSYEEEPIN7/8Kgk8tNQR6qHP+ZGwX4teH
f3On+JdLgWyYwWrPFOiELHIrDPUeo/tWfEr/hb3pwfwf8P/il4A0e71SLytav9Ly2b+V7hRIk6eHAj2Z...YAD
AIR0zxH4u8RQaRY6nCs9nHKCZp1KhhtQDPQ14uv/AAVP+ABxnxFfDIB5sJOP0/zmvlr9rf8AZn1T42fHbWNdX4v/AA6s9P
FvhuS32+YkizzY2kZLZzjkY9uOMUXYtD7z+Hv/BRL4H4I+JA3+HAxrVZHeHtM8Vi21G8cRwLqEEDwLi56KGYYyTxW78cv21vhd+zv4k
jT8OHulu0WOBNYDS7gwG1AzE7w3A9+tfSX7V/8AwT5+Kvx6Ay+LS+JtK1zQrXTYNptdPtGPtRcPJ5wEScyNzjJYt07Yo8wduh7C
QJPk46nivgTUP+CfuuaTqUhqnHxn+HOmXnBfhqHxm2TOmXtu5ia3uNaYOjkcqVL7s/730/AMEL/jDdCK4h8gUHeJIpYlAnhml2sh7g5
gqj8AWxjxBfYIJz9gkxweO3ftXzen+11+zDfeONf8G+Goviv4B0WWSPSLWY68Bxmsm...VKHYE4cNOGnDjZlzq3v...
AbxJrVvpsXiea2muJFjikurSSOMk8DLEYH419R2t5HeQRTwSpPBMoeOSM5VlIyCD6Yr8TNW/wCCfes6XfTaZf8Axq+Gtte
gL4P+DvD11qMeqXGn6bDbm7hfekm1QBtb+IY6H0pxdyT0Cvnb4vft6fCP4HeOrjwj4q1m6tdZgiWWSOK0d1VWBxyB7V9DN
AJSM8gd8cU27B6n6bfB/40eGfjp4Ng8UEbxtQ0iZ2jEjLtYMDggg8jHvWX8df2j/A37OGCan9...
wBlmz8SaL4w+LvgbVdHvytxaW+m6rGxjuB8rkgndyBz2zXjf7TnwV8VftLFLUfEWofGr4Z20lxkw6Tpg13cIIBjkjfgM3
s43ml2fjbWHsbzUgzQW8ELSvtXqxA6D3r59/4Jv/s2f8KR/wCEw8QXvjp14v+2KluLzw/eC4hhVTuIdsnYeOg4xXiH7VX
wCCmXwM8Ua5p+kadr15PqN/kadr15PfX86W9vGbN1Du5wBk4HWvfih8U9C+sL6n4T8eaVNJ+Dgm+8V+J7hrLR7FQ08iruVNc
bI+C3iv9oP4Gjwj4QvDvNt7u7mt5pbq+dhEY1IY7dvXP5UXSSRxS/8FUP+tSn/gqf8AFyT4A4ivsA4z/Z8
wDgmN8QfG0c0vhr4k+CNfS1YLI+m6hLcBiM4baxbaP/dP0t+CP7Znw1/ne4kLB/aE1nUNN8G6lcXbM8...

ErSNX8eeGvD+v+KlhsbfWbu9NvDDDk+YnmMd2459fpXg3hP/gmn4y8YzT2vhX4peAtdltgHlh0vUGmeP8A2mCscc+vXPNF
cwrjzblkIzuHRR71N/wAOj/jKpJXxJ4bzwdxlm25z9efoeK9S+Fv7CvjT9nn4b/FSfxB4w8M6VqPiDShpdlqV1dNBaxB2G
DvWpru9ndYhJ9jdXvJ2+9LIxHU9eelfVnxO+JehfCHwTqfizxNdGz0XTk8y4mVC5AzjgDk1+ZP7I/7L+j/A342aL4x8W/
Fz4c6jpemRSBYtP1dDJu2YU4LY4POetfUn7ZPibwx+0F8FbzwX4P+KngnTb7UJ0Es2pasiIYlOWVSrdelUhO1zQ0/wD4Ko
K3Jx1GPXBopi0NP8A4LGeKxrHj/wJ4T2rKLGzuL1othz+8KjdnPJHl8D3NfEutfCfVdC+G/hHxvLZbdE8RXV1BZyrnZvhL
wACdF+If/BOfwj4VWUwa3Z28d9YX0sYZILh5Pm752ncR0rN6laH5weBfhLqPjzRfFWuWVkq6F4csW1DVrxmIi8scIoP953
YKtPA+gRSNc6tqNkmqagRzczPy78n7ueAvYV5N/wTD8D2msfEnx4LyeSK4n8NyadFJEu/ynlKq8nzHnjPBqirq1z5Viksf
CjXY9G+IGg3mowaVJHawRT72dhHtUBeM1+MvxO+Etx8K/HOs+FZdQg1JtPuWhF15O3zV/hyvbPQgcYrl18PCPLlbYAliCL
+uuagmk6LqF5K6xQ28DybzwFCqT/Svzj/4Jo/sz2/g++b4t6/qf23UJY2g02yskOyBZFPmPIzHLMQcD0r67/as+JsHhf8A
Gv6GtPs4tI0u2tIkCQWsKxIo6KqqAAPbAr8Mv2FPBaat+1T4IivZFe2t7p7tvLBVm2ozBeD0JwcdOK/bqTxZZTQupjuAHU
wBpD4L+AfhL4a0OP4ieH1fT9JhiMaXYJLiIZwMdzmvyg/as+BM3we+M3iDSDqkeo6fdXTahBJ5ZWVUlPqxqLw
vd/0pXdy3ZnWw28nxU+PW8I1xNrfiVtoJJ3qZuAf95R17Yr+hzS9Oh0fTrWxtty29tEsMeTW+dsEsUYJydqjA5r8lv+Cdf7LNv4i8aWf
wAapEvcd4w1aLQfCur6lPzHa2skrfQKTX88Wg+FL342/GJrTSrdv7Q1/WHkiCruIEspbK+wVsZ7KZ9ok/Mn7ZNMH8CdBjCDF/+Gu
j64to5hcSabLDGxRcKWGMnmvy6/4J7+GYLz9qvqvwU12xkTT45b0quRgh+SFVeeRlsnPXFDHGSR83+JPDjeG9e1fQ76zWT3vdN
hyGCHVW0NoRmS+3BAxOcbOAxHc9a+If27tQuPGX7V3jMlgq2rR2MKgpA4AVRt4B4Ayenk6ZfMe8/Cfx5H+zj/
wAE19V1e0aO013fTWdjEuWVixMZZQcEAKGOfavkn9mHxx4R+Enxj+B0Cqd+mX7094+0Mpj7nC0KoiB2ZHSj7ZNPMMf/+09+
DXiTxVquuPrOqwm6kGmyCOFAzFQoBPPCjNGorqzPg/4h+IdM8U/FnXXPE2mwTWGnajrJvoYpY8PbxtKGUdeScc/n2xX9C3w
br4B+L7eb4M+CmmFxLL/AGVbo7MASSEUdS3Tjj2qokbo/O7/AILBeHRxU8IeHHKtFp+ny3OFy4OQfRT+tfSH/BJ
b+1qbx/8AtR+LblJpIo7NE01FkHICgk9D90k1+o37H8Nn4E/
Zt8C6TmaVlsVmZk3O5y3Tmi2o5M+Yv+CxniiiOPwf4G8Nk7bhc3zXckZ6FUAHJzOfSuD/4JX+PPh98A/Cnxg8XeLfEu
wVW+E7eOvvCeg+PtN1BbV9DkNtPaXUWRKknQqeMY5B4NfmLL4fLSM8pgkLbseZHv59DntRfUFZ7n74dt9f9r74LJhT8TPDw9
AIKlfF7QfFn7OPh2Dw3rFvrGn6zqoH2i1O+OREB38gjococV+W8nhbzCQY7OQ7SAzWy56dz/njFf7UVgdF+DfwL8I2JQWd
aksddufDd7pml22kOsTnUA/753GRjcWV+05+yfrH7LVlNT+e9ySfulkAAJNcNdtJMzvo9+/45+XzbHABfLnUE2aHdF1NSNn4N+Aud
wD4KeeIpvGX7RbJARBbabYRJb34YRwx+cgZi5LbjjGHJdRqbmaW+VpqG0bnhGqoNqHJs+/2uDmwiXPqOLuMPgx2ubCGIdJfSbYUuF5yU


```
margin: [0, 0, 80, 0], // izquierda, superior,derecha , inferior
                },
                        {

{ text: 'Vicario Segura 587\n' },
                        { text: 'Capital - Catamarca\n' },

'3834-4172012\n' },
                { text: 'motomatch01@gmail.com' },
                        ],

        {
                text: [
                        { canvas: [{ type: 'rect', x: 0, y: 0, w: 10
this.letraValue + '\n', style: { fontSize: 40 }, margin: [10, 5, 0, 0] },//{ text: this.letraV
Asegúrate de que 'this.letra' represente la letra seleccionada en el campo 'letra'

            { text: 'COMO FACTURA' }
                        ],
                                alignment: 'center',

            { text: titulo + '\n' },
                        { text: 'N° 0000 -' + numerocomprobant
' + this.puntoventa + '\n' },
                ],
```

```
                                                    alignment: 'right',
                                                                        fo

'Fecha: ' + fecha,
                            alignment: 'right',
                                                  margin: [25, 0, 5, 30],
                                                                        font

            {
                        type: 'line',
                                                x1: 0, y1: 0,
                                                                    x2: 380, y2:

Color gris claro para la línea
                                    }
                                                ],
                                                        margin: [0, 0, 30, 0] // Agre

            {
                        text: [
                                                { text: 'Sres: ' + cliente.nombre + '\n' },

                { text: 'DNI: ' + cliente.dni + '\n' },
                                                                { text: 'CUIT: ' + clie

cliente.tipoiva + '\n' },
                                    ],
                                                fontSize: 10,
                                                                    margin: [0,

canvas: [
                    {
                                type: 'line',
                                                x1: 0, y1: 0,
                                                                        x2:

'#cccccc' // Color gris claro para la línea
                                                }
                                                    ],
                                                            margin: [0, 0, 3

Aquí puedes seguir añadiendo más elementos al documento ...
                                                    {
                                                            style: 'tableExa
```

```
'15%'],
                body: [
                                ['Cant./Lts.', 'DETALLE', 'P.Unitario', 'Total'],

        },
                {
                        style: 'tableExample',
                                table: {
                                        widths: ['*']

            // ... Añade más filas según sea necesario ...
                                ],
                                        bold: tr

{
        header: {
                        fontSize: 10,
                                bold: true,
                                        margin: [2, 0, 0, 10],

         fontSize: 8,
                        },
                                total: {
                                        bold: true,
                                                fontSize: 8,

      },
            };
                // cerrar el loading
                                Swal.close();
                                        // Crear el PDF
                                                pdfMake.createP
'_' + fechaFormateada + '.pdf');
                                pdfMake.createPdf(documentDefinition).getBlob((blob) => {
this.numerocomprobanteImpresion + '_' + titulo + '_' + fechaFormateada + '.pdf');
                                                }, (error

showNotification(message: string) {
                                Swal.fire({
                                        icon: 'error',
                                                title: 'Error',
```

components\condicionventa\condicionventa.component.ts

```
import { Component } from '@angular/core';
                                        import { CargardataService } from '../../services/ca

import Swal from 'sweetalert2';
                                import * as FileSaver from 'file-saver';
                                                                        import xlsx from 'xlsx/

router';
        import { FilterPipe } from 'src/app/pipes/filter.pipe';
                                                                  import { filter } from 'rxjs/op

componente calculoproducto
                            import { CalculoproductoComponent } from '../calculoproducto/calculo

'primeng/dynamicdialog';
                           import { ChangeDetectorRef } from '@angular/core';


condicionventa',
                  templateUrl: './condicionventa.component.html',
                                                                  styleUrls: ['./condicionver


CondicionventaComponent {
                             public tipoVal: string = 'Condicion de Venta';
                                                                       public codTarj: str


  public tipo: any[] = [];
                              searchText: string;
                                                  ref: DynamicDialogRef | undefined;
                                                                                  public pr

boolean = false;
                  public prefi3: boolean = false;
                                                    public prefi4: boolean = false;
                                                                                  public tar




  };
      public cheque = {
                             Banco: '',
                                                Ncuenta: '',
                                                                    Ncheque: '',
                                                                                         Nombre: '',
                                                                                                              Plaza:




  public productos: Producto[];
                                    public productoElejido: Producto;
```

```
                                                                    public clienteFrompuntoVen

boolean = false;
                  public previousUrl: string = "";
                                                          filteredTipo: any[] = [];

cdr: ChangeDetectorRef, private router: Router, private activatedRoute: ActivatedRoute, privat

    this.clienteFrompuntoVenta = this.activatedRoute.snapshot.queryParamMap.get('cliente');

    this._cargardata.tarjcredito().pipe(take(1)).subscribe((resp: any) => {
                                                                          this.tipo = r

this.filterByDay();
                       });
                            }
                               filterByDay() {
                                                 const dayOfWeek = new Date().getDay(); // Do

'd1', // Domingo
                     1: 'd2', // Lunes
                                          2: 'd3', // Martes
                                                                   3: 'd4', // Miércoles



    };
          const dayField = dayFieldMap[dayOfWeek];
                                                          this.filteredTipo = this.tipo.filter(ite



      this.ref.close();
                           }
                              }
                                    ngOnInit() {
                                                                 }
                                                                        selectTipo(item: any) {

this.tipoVal = item.tarjeta; // Almacena el centro seleccionado
                                                                  this.codTarj = item.cod_tar

item.activadatos;
                     this.listaPrecioF(); // aca se llama a la funcion que muestra los prefijo

      // aca se llama a la funcion que muestra los prefijos
                                                              }
                                                                   else if (this.activaDatos
```

```
funcion que muestra los prefijos
                                    }
                                         else {
                                                    this._cargardata.artsucursal().pipe(take(
this.productos = [...resp.mensaje];
                                      // Forzar la detección de cambios
                                                                                this.cdr.d



      title: 'Ingrese los datos de la tarjeta',
                                                    html: `<input type="text" id="titular" cl
id="dni" class="swal2-input" placeholder="DNI">
                                                         <input type="number" id="numero" cla
type="number" id="autorizacion" class="swal2-input" placeholder="Autorización">`,
                                                                                showCar

cancelButtonText: 'Cancelar',
                                    preConfirm: () => {
                                                               const titular = (<HTMLInputEleme
(<HTMLInputElement>document.getElementById('dni')).value;
                                                                  const numero = (<HTMLInputEle
autorizacion = (<HTMLInputElement>document.getElementById('autorizacion')).value;
                                                                                    if (!
Swal.showValidationMessage(`Por favor rellene todos los campos`);
                                                                        //return;
                                                                                       }

RegExp("^[0-9]{8}$");
                                let reTitular = new RegExp("^[a-zA-Z ]{1,40}$");
                                                                                let reAut
reNumero.test(numero)) {
                                     Swal.showValidationMessage(`El número de la tarjeta no es vá


          Swal.showValidationMessage(`El DNI no es válido`);
                                                                   //return;
                                                                          }

Swal.showValidationMessage(`El titular no es válido`);
                                                              //return;
                                                                    }
                                                                        if (
Swal.showValidationMessage(`La autorización no es válida`);
                                                                   //return;
                                                                          }
```

```
    }).then((result) => {
                                if (result.value) {
                                                        this.tarjeta.Titular = result.value.
this.tarjeta.Numero = result.value.numero;
                                                    this.tarjeta.Autorizacion = result.value.aut
        this._cargardata.artsucursal().pipe(take(1)).subscribe((resp: any) => {
                                                                                    conso
          // Forzar la detección de cambios
                                                      this.cdr.detectChanges();
                                                                          });



'Ingrese los datos del Cheque',
                                      html:
                                                    `<input type="text" id="banco" class="swal2-
class="swal2-input" placeholder="N° Cuenta">
                                                     <input type="number" id="ncheque" class="sw
id="nombre" class="swal2-input" placeholder="Nombre">
                                                          <input type="text" id="plaza" clas
id="importeimputar" class="swal2-input" placeholder="Importe a Imputar">
                                                                                <input type="nu
Cheque">
                <input type="text" id="fechacheque" class="swal2-input" placeholder="Fecha del

'date' para activar el datepicker nativo
                                                  document.getElementById('fechacheque').setAttr


confirmButtonText: 'Aceptar',
                                    cancelButtonText: 'Cancelar',
                                                                    preConfirm: () => {
(<HTMLInputElement>document.getElementById('banco')).value;
                                                                const ncuenta = (<HTMLInput
= (<HTMLInputElement>document.getElementById('ncheque')).value;
                                                                    const nombre = (<HTMLIn
= (<HTMLInputElement>document.getElementById('plaza')).value;
                                                                const importeimputar = (<
        const importecheque = (<HTMLInputElement>document.getElementById('importecheque')).val
(<HTMLInputElement>document.getElementById('fechacheque')).value;
                                                                      if (!banco || !ncuent
|| !fechacheque) {
```

```
                            Swal.showValidationMessage(`Por favor rellene todos los campos`);

{1,40}$");
                let reNcuenta = new RegExp("^[0-9]{1,40}$");
                                                                    let reNcheque = new RegE
{1,40}$");
                let rePlaza = new RegExp("^[a-zA-Z ]{1,40}$");
                                                                      let reImporteImputar =
RegExp("^[0-9]{1,40}$");
                                    let reFechaCheque = new RegExp("^\\d{2}/\\d{2}/\\d{4}$");//("^
        if (!reBanco.test(banco)) {
                                                Swal.showValidationMessage(`El nombre del banco r

reNcuenta.test(ncuenta)) {
                                        Swal.showValidationMessage(`El numero de cuenta no es váli


          Swal.showValidationMessage(`El numero de cheque no es válido`);
                                                                              //return;

Swal.showValidationMessage(`El nombre no es válido`);
                                                                  //return;
                                                                          }
                                                                                  if (!
es válida`);
                        //return;
                                    }
                                                  if (!reImporteImputar.test(importeimputar)) {
válido`);
                    //return;
                                        }
                                                  if (!reImporteCheque.test(importecheque)) {

          //return;
                                }
                                        return { banco, ncuenta, ncheque, nombre, plaza, importeim

      if (result.value) {
                                    this.cheque.Banco = result.value.banco;
                                                                        this.cheque.Nc
result.value.ncheque;
                             this.cheque.Nombre = result.value.nombre;
                                                                    this.cheque.Plaz
result.value.importeimputar;
                                        this.cheque.ImporteCheque = result.value.importecheque;
```

```
console.log('Cheque guardado:', this.cheque);//console.log('Tarjeta guardada:', this.tarjeta);

any) => {
                  console.log(resp.mensaje);
                                                this.productos = [...resp.mensaje];




listaPrecioF() {
                   if (this.listaPrecio == "0") {
                                                 this.prefi0 = true;
                                                                        this.prefi1 =


this.prefi4 = false;
                         }
                              else if (this.listaPrecio == "1") {
                                                                  this.prefi0 = false;


false;
            this.prefi4 = false;
                                    }
                                         else if (this.listaPrecio == "2") {
                                                                           this.prefi0


this.prefi3 = false;
                          this.prefi4 = false;
                                                }
                                                     else if (this.listaPrecio == "3") {


false;
            this.prefi3 = true;
                                       this.prefi4 = false;
                                                           }
                                                                else if (this.listaPrecio ==


this.prefi2 = false;
                          this.prefi3 = false;
                                                   this.prefi4 = true;
                                                                       }
                                                                          }
                                                                             selectProduct


producto,
               cliente: this.clienteFrompuntoVenta,
                                                    tarjeta: this.tarjeta,
                                                                            cheque: t
```

```
      listaPrecio: this.listaPrecio,
                                        };
                                              this.ref = this.dialogService.open(Calculoproduc


      {
               producto: producto,
                                          cliente: this.clienteFrompuntoVenta,
                                                                                  tarjeta:


        codTarj: this.codTarj,
                                       listaPrecio: this.listaPrecio,
                                                                          },
                                                                                 contentStyle


    });
         }
            exportExcel() {
                                  import('xlsx').then((xlsx) => {
                                                                        const worksheet = xlsx.uti
{ data: worksheet }, SheetNames: ['data'] };
                                                    const excelBuffer: any = xlsx.write(workbook
this.saveAsExcelFile(excelBuffer, 'products');
                                                    });
                                                         }
                                                            saveAsExcelFile(buffer: any, fileNam
officedocument.spreadsheetml.sheet;charset=UTF-8';
                                                        let EXCEL_EXTENSION = '.xlsx';
                                                                                          const


FileSaver.saveAs(data, fileName + '_export_' + new Date().getTime() + EXCEL_EXTENSION);
                                                                                           }
                                                                                            }
```

c

```
import { Component } from '@angular/core';

                                                                    @Component

condicionventacab.component.html',
                                    styleUrls: ['./condicionventacab.component.css']
                                                                                    })
                                                                      export c

}
```

components\cuentacorriente\cuentacorriente.component.ts

```
import { Component, OnInit, ViewChild } from '@angular/core';
                                                        import { Table } from 'primeng/ta

cargardata.service';
                    import { Cliente } from '../../interfaces/cliente';
                                                                  import { ActivatedRoute

saver';
       import xlsx from 'xlsx/xlsx';
                                    import { first, take } from 'rxjs/operators'
                                                                    import Swal fr



'app-cuentacorriente',
                         templateUrl: './cuentacorriente.component.html',
                                                                  styleUrls: ['./cuent



public clientes: Cliente[];
                              public clienteElejido: Cliente;


Router) { }
              ngOnInit(): void {
                                    //let sucursal: string = localStorage.getItem('sucursal');


if (sucursal) {
                      this._cargardata.clisucx(sucursal).pipe(take(1)).subscribe((resp: any) =>


      }, (err) => {
                            console.log(err);
                                                 this.showNotification('Error al cargar los



this.showNotification('Sucursal no encontrada en localStorage');//console.log('Sucursal no enc



console.log(cliente);
                           this._router.navigate(['components/cabeceras'], { queryParams: { clie


import('xlsx').then((xlsx) => {
                                     const worksheet = xlsx.utils.json_to_sheet(this.clientes)

['data'] };
                  const excelBuffer: any = xlsx.write(workbook, { bookType: 'xlsx', type: 'arra



saveAsExcelFile(buffer: any, fileName: string): void {
                                                            let EXCEL_TYPE = 'application/vnd.op

EXCEL_EXTENSION = '.xlsx';
```

```
                                const data: Blob = new Blob([buffer], {
                                                                        type: EXCEL_TYPE

Date().getTime() + EXCEL_EXTENSION);
                                    }
                                          showNotification(message: string) {
                                                                              Swal.fire({



confirmButtonText: 'Aceptar'
                                });
                                     }
                                      }
```

components\editcliente\editcliente.component.ts

```
import { Component, OnInit } from '@angular/core';
                                                    import { ActivatedRoute, Router } from '@ang

ReactiveFormsModule, AbstractControl, ValidatorFn, AbstractControlOptions, FormBuilder, FormGr

import { SubirdataService } from 'src/app/services/subirdata.service';
                                                                    import { debounceTime }


@Component({
               selector: 'app-editcliente',
                                             templateUrl: './editcliente.component.html',
                                                                                          styl


EditclienteComponent implements OnInit {
                                          public editarclienteForm!: FormGroup;
                                                                               public client


boolean;
          public cuitFlag: boolean;
                                       public dniFlag: boolean;
                                                                 public telefonoFlag: boolean;



  constructor(private subirdata: SubirdataService, private router: Router, private activatedRo

    this.sucursal = localStorage.getItem('sucursal');
                                                          this.clienteFrompuntoVenta = this.act

this.clienteFrompuntoVenta = JSON.parse(this.clienteFrompuntoVenta);
                                                                    console.log(this.clier



    this.monitorFormChanges();
                                }
                                   ngOnInit(): void {
                                                       }
                                                          inicializarForm() {
                                                                               this.editarcl


cuit: new FormControl(''),
                                dni: new FormControl(''),
                                                           telefono: new FormControl(''),


      ingresos_br: new FormControl(''),
                                             });
                                                 /*    this.editarclienteForm = this.fb.group({

Z0-9\sñÑ]{2,40}){1}$/)]], // Nombre debe tener mínimo 2 caracteres
                                                                                 direccion: ['', [Valid

{1}$/)]], // Dirección debe tener mínimo 2 caracteres
                                                            cuit: ['', [Validators.required, Va

dni: ['', [Validators.required, Validators.pattern(/^\d{8}$/)]], // DNI debe ser 8 dígitos
```

```
this.clienteFrompuntoVenta.telefono || 0, Validators.pattern(/^(0|[0-9]{5,15}){1}$/)]], // Tel

      ingresos_br: ['', Validators.required]
                                                }); */
                                                        }
                                                           cargarDatosForm() {
                                                                                this.editarc
FormControl(this.clienteFrompuntoVenta.nombre.trim(), Validators.compose([Validators.required,

cuit: new FormControl(this.clienteFrompuntoVenta.cuit, Validators.compose([Validators.required

FormControl(this.clienteFrompuntoVenta.dni, Validators.compose([Validators.required,
                                                                                        Vali

FormControl(this.clienteFrompuntoVenta.telefono || 0, Validators.compose([
                                                                                Validators.p

FormControl(this.clienteFrompuntoVenta.direccion.trim(), Validators.compose([Validators.requir

      ])),
                tipoiva: new FormControl(this.clienteFrompuntoVenta.tipoiva),
                                                                                 ingresos_br


onSelectionChange(event: any) {
                                      const selectedValue = event.target.value;
                                                                        console.log(se

this.editarclienteForm.controls['cuit'].setValue(0);
                                                                }
                                                                     else {
                                                                                this.editarclienteFor

selectedValue
               }
                  guardar(form: FormGroup) {
                                                if (form.valid)  // si el formulario es validc

console.log("TIPO IVA:" + form.value.tipoiva);
                                                    if (form.value.tipoiva == "Excento") {

"Monotributo") {
                        cod_iva = 3;
                                                }
                                                       else if (form.value.tipoiva == "Responsable I
```

```
(form.value.tipoiva == "Consumidor Final") {
                                                    cod_iva = 2;
                                                                }
                                                                        let date = new Da
+ "-" + date.getDate();//let fecha= date.getDate() + "/"+ (date.getMonth()+ 1) + "/" +date.get
":" + date.getSeconds();
                                let editadoCliente =
                                                            {
                                                                    "cliente": parseInt(this.clie

"direccion": form.value.direccion,
                                                "dni": parseInt(form.value.dni),
                                                                                    "cuit": form
parseInt(this.clienteFrompuntoVenta.cod_iva),
                                                 "cod_ven": parseInt(this.clienteFrompunto
parseInt(this.clienteFrompuntoVenta.cod_zona),
                                                  "tipoiva": form.value.tipoiva,//this.cli
this.clienteFrompuntoVenta.vendedor,
                                                "zona": this.clienteFrompuntoVenta.zona,//"",


        "idcli": parseInt(this.clienteFrompuntoVenta.idcli),
                                                                    "id_cli": parseInt(this.cl


        "ingresos_br": form.value.ingresos_br,
                                                                    "n_sucursal": this.clienteFrompuntoVenta
parseInt(this.clienteFrompuntoVenta.id_suc),
                                                                "estado_act": this.clienteFrompuntoVenta.e


== "eliminar") {
                            this.subirdata.eliminarCliente(editadoCliente, this.sucursal).subscrib


            title: 'Eliminando...',
                                                timer: 300,
                                                                        didOpen: () => {



            if (result.dismiss === Swal.DismissReason.timer) {
                                                                                    console.log('I was


          })
                    });
                                }
                                        else if (this.accion == "guardar") {
                                                                                        if (editadoCli

          Swal.fire({
```

```
                                title: 'ERROR',
                                                    text: 'Se requiere un cuit para es
true,
                confirmButtonColor: '#3085d6',
                                                    cancelButtonColor: '#d33',


        this.subirdata.editarDatosClientes(editadoCliente, this.sucursal).subscribe((data: a

            title: 'Guardando...',
                                                timer: 300,
                                                                didOpen: () => {

          }).then((result) => {
                                            if (result.dismiss === Swal.DismissReason.timer

              window.history.back();
                                                }
                                                        })
                                                                });
                                                                        }



    Swal.fire({
                        title: 'ERROR',
                                                text: 'Verifique los datos ingresados, hay cam

      showCancelButton: true,
                                        confirmButtonColor: '#3085d6',
                                                                cancelButtonColor

console.log("invalido");
                            // Marca los campos como tocados para que se muestren los errore

form.get(control).markAsTouched();
                                        }
                                             }
                                                }
                                                    monitorFormChanges() {
                                                                Object.keys(this.

this.editarclienteForm.get(field);
                                        control.valueChanges.pipe(debounceTime(1000)).subscrib

      this.nombreFlag = this.editarclienteForm.controls['nombre'].invalid;
                                                                this.cuitF
```

```
this.dniFlag = this.editarclienteForm.controls['dni'].invalid;
                                                                this.telefonoFlag = this

this.direccionFlag = this.editarclienteForm.controls['direccion'].invalid;
                                                                        });
                                                                    });
                                                                        }
                                                                         }
```

```
import { Component, OnInit } from '@angular/core';
                                                    import { NgForm } from '@angular/forms';
                                                                                                impo
forms';
       import { ActivatedRoute, Router } from '@angular/router';
                                                                 import { LoginService } from '


{ CryptoService } from '../../services/crypto.service';
                                                          import { CrudService } from '../../serv



@Component({
              selector: 'app-login',
                                      templateUrl: './login.component.html',
                                                                              styleUrls: ['./l



forma: FormGroup;
                   public validacion: boolean = false;
                                                        public mail: string;
                                                                              public check: bo



errorhttp: string;
                    public spinerbotoningresar: boolean = true;
                                                                  public sucursal: string;
                                                                                               cor
_crypto: CryptoService, private _crud: CrudService) {



(localStorage.getItem('sddccasdf') != null) {
                                                         let check: string = localStorage.getItem('s



      else {
                    this.check = false;
                                                }
                                                         }
                                                              //login con datos guardados----------
localStorage.getItem("sddeeasdf") !== null) {
                                                         let email: any = localStorage.getItem('sddc
localStorage.getItem('sddeeasdf');//password
                                                        email = this._crypto.decrypt(email);
                                                                                                pa
pass).then((resp) => {
                                if (resp == true) {
                                                                    this.router.navigate(['./components



      email: new FormControl(null, [Validators.required, Validators.email]),// primer valor es
```

```
Validators.required)
                        });
                             }
                                ingresar(f: any) {
                                                    if (this.sucursal == null) {
                                                                                Swal.f

sucursal',
                  icon: 'error',
                                        showCancelButton: true,
                                                                    confirmButtonText: 'OK'



this.sucursal);
                     this.checkPromise().then(() => {
                                                                 console.log("email:" + f.value.em

online----------------------------------
                                                    this._login.SignIn(f.value.email, f.value.passw

(resp == true) {
                              this._crud.getListSnapFilter('usuarios/cliente', 'email', f.value.

localStorage.setItem('usernameOp', resp2[0].payload.val().username);
                                                                                localStorage

localStorage.setItem('sddffasdf', this._crypto.encrypt(resp2[0].payload.val().nivel));// sddff

resp2[0].key);//sddggasdf userkey
                                                    f.reset();
                                                                    this.router.navigate(['

dashboard']);
                           });
                                        if (this.check == true) {
                                                                            localStorage.se

               localStorage.setItem('sddddasdf', this._crypto.encrypt(f.value.email));// sdddda

this._crypto.encrypt(f.value.password));// sddeeasdf:pass, cdkcdck:true
                                                                                }

sddccasdf:check, cdkcdck:true
                                                  localStorage.setItem('sddddasdf', this._crypto.encr

localStorage.setItem('sddeeasdf', this._crypto.encrypt(null));// sddeeasdf:pass, cdkcdck:true



localStorage.setItem('sddddasdf', this._crypto.encrypt(null));// sddddasdf:mail, cdkcdck:true

this._crypto.encrypt(null));// sddeeasdf:pass, cdkcdck:true
                                                                    f.reset();
                                                                                        S

usuario incorrectos',
                                     icon: 'error',
                                                             showCancelButton: true,
```

```
'#d33',
                    confirmButtonText: 'OK',
                                                });
                                                    }
                                                            })
                                                                    });



      title: 'Input email address',
                                        input: 'email',
                                                    inputLabel: 'Your email address'


{
       this._login.ForgotPassword(email);
                                            }
                                               }
                                                     checkPromise() {
                                                          let promise = new Pr

localStorage.setItem('sddccasdf', "true");
                                               }
                                                      else {
                                                           localStorage.setItem('sdd



  onSelectionChange(event: any) {
                                     this.sucursal = event.target.value;
                                                                          }
                                                                           }
```

components\ne

```
import { Component, OnInit } from '@angular/core';
                                                    import { ActivatedRoute, Router } from '@ang

ReactiveFormsModule, AbstractControl, ValidatorFn, AbstractControlOptions, FormBuilder, FormGr

import { SubirdataService } from 'src/app/services/subirdata.service';
                                                                    import { debounceTime, t

app/services/cargardata.service';
                                  import { first, take } from 'rxjs/operators';



templateUrl: './newcliente.component.html',
                                             styleUrls: ['./newcliente.component.css']
                                                                                      })
                                                                                        export


public clienteFrompuntoVenta: any;
                                      public sucursal: any;
                                                           public nombreFlag: boolean;
                                                                                        public


boolean;
          public direccionFlag: boolean;
                                               public cod_ivaFlag: boolean;
                                                                     public vendedores: any


nombreVendedor: any;

                                               constructor(private _cargardata: CargardataService,
activatedRoute: ActivatedRoute, private fb: FormBuilder) {
                                                                  this.cargarForm();
                                                                                  this.monit



this.nuevoclienteForm = this.fb.group({
                                             nombre: new FormControl('', Validators.compose([V

{1}$/)
            ])),
                       cuit: new FormControl('', Validators.compose([Validators.required,


FormControl('', Validators.compose([Validators.required,
                                                                  Validators.pattern(/^([0-9]{8}){


        Validators.pattern(/^(0|[0-9]{5,15}){1}$/)
                                                          ])),
                                                                       direccion: new FormControl('

^([a-zA-Z0-9°\.\-_\s,/ñÑªº]{2,60}){1}$/)
```

```
                                                ])),
                                                    cod_iva: new FormControl('', Validator
    },);
          }
             onSelectionVendedorChange(event: any) {
                                                    this.codigoVendedor = this.vendedores[
this.vendedores[event.target.value].vendedor;
                                                 }
                                                       onSelectionChange(event: any) {
                                                                                        const
if (selectedValue == "2") {
                                  this.nuevoclienteForm.controls['cuit'].setValue(0);
                                                                                        }
                                                                                             e


    // Realiza acciones basadas en selectedValue
                                                 }
                                                    guardar(form: FormGroup) {
                                                                                  if (form.va
= ["", "Responsable Inscripto", "Consumidor Final", "Monotributo", "Excento"];
                                                                                   let indexn
genera un numero entre 10000 y 9999999 incluidos
                                                     let sucursal: any = localStorage.getItem
Math.floor((Math.random() * 99999) + 10000);
                                                 let date = new Date();
                                                                        let fecha = date
let fecha= date.getDate() + "/"+ (date.getMonth()+ 1) + "/" +date.getFullYear();
                                                                                     let hora

      let nuevoCliente =
                                 {
                                          "cliente": (sucursal * 100000) + nuevoclirandom,//parse
"direccion": form.value.direccion,
                                             "dni": parseInt(form.value.dni),
                                                                          "cuit": form
"cod_ven": this.codigoVendedor,//this.vendedor.cod_ven,//parseInt(form.value.cod_ven),
ivaArray[form.value.cod_iva],// ""
                                             "vendedor": this.nombreVendedor,//this.vendedor.vend
form.value.telefono,
                               "estado": "",//"",
                                             "idcli": indexnuevocli,//parseInt(form.v
parseInt(form.value.id_cli),
                                    "fecha": fecha,
                                                  "hora": hora,
                                                                      "ingresos_br":
```

```
"id_suc": indexnuevocli,
                                "estado_act": ""
                                                    }
                                                            if (nuevoCliente.cuit == 0 && nue

title: 'ERROR',
                        text: 'Se requiere un cuit para este tipo de IVA',
                                                                                icon: 'er

'#3085d6',
                    cancelButtonColor: '#d33',
                                                        cancelButtonText: 'Cancelar'
                                                                                        })


this.subirdata.subirDatosClientes(nuevoCliente, sucursal).subscribe((data: any) => {

'Guardando...',
                            timer: 300,
                                                    didOpen: () => {
                                                                                Swal.showLoadi

(result.dismiss === Swal.DismissReason.timer) {
                                                                console.log('I was closed by the


        });
                }
                     }
                          else {
                                    this.monitorFormChanges();
                                                                    //mostrar en consola qu

console.log(form);
                        Swal.fire({
                                        title: 'ERROR',
                                                                text: 'Verifique los datos i
        icon: 'error',
                                showCancelButton: true,
                                                                confirmButtonColor: '#3085d6',


      console.log("invalido");
                                    // Marca los campos como tocados para que se muestren los

form.get(control).markAsTouched();
                                     }
                                          }
```

```
                                                    }
                                                       cargarVendedor() {
                                                                                this._cargardata.vend

console.log(resp);
                          this.vendedores = resp.mensaje;
                                                              });
                                                                   }
                                                                                    monitorFormChanges() {

const control = this.nuevoclienteForm.get(field);
                                                                control.valueChanges.pipe(debounceTime(

cambió a: `, value);
                               this.nombreFlag = this.nuevoclienteForm.controls['nombre'].invalid

        this.dniFlag = this.nuevoclienteForm.controls['dni'].invalid;
                                                                                        this.telefonoFlag

this.direccionFlag = this.nuevoclienteForm.controls['direccion'].invalid;
                                                                                          });
                                                                                                 });
                                                                                                      }
                                                                                                       }
```

```
import { Component, OnInit } from '@angular/core';
                                                    //import { CargardataService } from '../serv

declare function init_plugins();

                                                                  @Component({
                                                                               selector: 'app-p

pages.component.css']
                      })
                        export class PagesComponent implements OnInit {



this._cargardata.tarjcredito().pipe(take(1)).subscribe((resp:any)=>{
                                                                                    console.log(resp.mer

JSON.stringify(resp.mensaje));
                                      localStorage.setItem('lastSelectedValue', JSON.stringify({



console.log("PAGES COMPONENTS");
                                      init_plugins();
                                                        }
                                                         }
```

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

'./pedidos.component';

                                        describe('PedidosComponent', () => {
                                                                let componer


  beforeEach(async () => {
                              await TestBed.configureTestingModule({
                                                                  declarations: [ Pedi


    fixture = TestBed.createComponent(PedidosComponent);
                                                        component = fixture.componentInsta



    expect(component).toBeTruthy();
                                     });
                                        });
```

components\pedidos\pedidos.component.t

```
import { Component, OnInit } from '@angular/core';
                                                    import { DynamicDialogConfig, DynamicDialogR

'src/app/services/cargardata.service';
                                         import Swal from 'sweetalert2';



pedidos.component.html',
                           styleUrls: ['./pedidos.component.css']
                                                                    })
                                                                      export class PedidosComponen



constructor(
                 private cargarDataService: CargardataService,
                                                                       public ref: DynamicDialogRef,




    const { cod_sucursal, comprobante } = this.config.data;
                                                                  this.cargarDataService.pedidoxC

console.log(data);
                         this.pedidos = data.mensaje;
                                                      }, (error) => {
                                                                          this.showNotifica



showNotification(message: string) {
                                          Swal.fire({
                                                      icon: 'error',
                                                                        title: 'Error',
```

components\productos\productos.component.ts

```
import { Component } from '@angular/core';
                                        import Swal from 'sweetalert2';
                                                                    @Component({
                                                                                  selecto


styleUrls: ['./productos.component.css']
                                      })
                                        export class ProductosComponent {
                                                                     public estado: stri




// Función para abrir el modal
                                async openModal() {
                                                        const { value } = await Swal.fire({
                                                                                             titl


      // Supongamos que tienes los siguientes estados
                                                             'estado1': 'Estado 1',
                                                                                  'estado


true,
         confirmButtonText: 'OK',
                                      cancelButtonText: 'Cancelar'
                                                                 })
                                                                         if (value) {
                                                                                      thi
```

components\puntoventa\puntoventa.component.ts

```
import { Component, OnInit, ViewChild } from '@angular/core';
                                                        import { Table } from 'primeng/ta

cargardata.service';
                    import { Cliente } from '../../interfaces/cliente';
                                                                  import { ActivatedRoute

saver';
       import xlsx from 'xlsx/xlsx';
                                    import { first, take } from 'rxjs/operators'
                                                                        import Swal fr



'app-puntoventa',
                   templateUrl: './puntoventa.component.html',
                                                                  styleUrls: ['./puntoventa.comp



  public clientes: Cliente[];
                                public clienteElejido: Cliente;
                                                                  constructor(private _cargardat

    let sucursal: string = localStorage.getItem('sucursal');
                                                                  this._cargardata.clisucx(sucur

      this.clientes = resp.mensaje;
                                       }, (err) => {
                                                          this.showNotification('Error al car



     });
          }
             selectCliente(cliente) {
                                          console.log(cliente);
                                                            this._router.navigate(['comp
JSON.stringify(cliente) } });
                                }
                                   editCliente(cliente) {
                                                           console.log(cliente);
                                                                                  this._rou
JSON.stringify(cliente) } });
                                }
                                   exportExcel() {
                                                       import('xlsx').then((xlsx) => {
                                                                                        cons

workbook = { Sheets: { data: worksheet }, SheetNames: ['data'] };
                                                                  const excelBuffer: any

this.saveAsExcelFile(excelBuffer, 'products');
                                                  });
                                                       }
                                                          saveAsExcelFile(buffer: any, fileNam

officedocument.spreadsheetml.sheet;charset=UTF-8';
                                                           let EXCEL_EXTENSION = '.xlsx';
```

```
                                                                                const
FileSaver.saveAs(data, fileName + '_export_' + new Date().getTime() + EXCEL_EXTENSION);
                                                                                 }
                                                                                    sh


      title: 'Error',
                          text: message,
                                               confirmButtonText: 'Aceptar'
                                                                    });
                                                                         }
                                                                          }
```

compo

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

'./recibos.component';

                                        describe('RecibosComponent', () => {
                                                                              let componer


  beforeEach(async () => {
                               await TestBed.configureTestingModule({
                                                                      declarations: [ Reci


    fixture = TestBed.createComponent(RecibosComponent);
                                                           component = fixture.componentInsta



    expect(component).toBeTruthy();
                                     });
                                        });
```

components\recibos\recibos.component.ts

```
import { Component, OnInit } from '@angular/core';
                                                    import { DynamicDialogConfig, DynamicDialogR

'src/app/services/cargardata.service';
                                           @Component({
                                                    selector: 'app-recibos',
                                                                                templateUrl: './


export class RecibosComponent implements OnInit {
                                                    recibos: any[] = [];
                                                                                errorMessage: string


CargardataService,
                       public ref: DynamicDialogRef,
                                                        public config: DynamicDialogConfig
                                                                                                ) {

this.config.data;
                     this.cargarDataService.reciboxComprobante(cod_sucursal, comprobante).subs


this.errorMessage = data.mensaje;
                                                console.error(this.errorMessage);
                                                                                } else {



        this.errorMessage = 'Ocurrió un error al cargar los datos.';
                                                                                console.error(this
```

components\venta\venta.component.ts

```
import { Component } from '@angular/core';
                                        import {ActivatedRoute, Router} from '@angular/route



selector: 'app-venta',
                        templateUrl: './venta.component.html',
                                                              styleUrls: ['./venta.component



carritoArray: any[] = [];
                           constructor(private activatedRoute: ActivatedRoute)
                                                                                {
                                                                                   this.carritoS

this.carritoArray = JSON.parse(this.carritoString);
                                                    console.log(this.carritoArray);
                                                                                    }
                                                                                     }
```

config

```
/* Đ
   export const Urllogin="https://apachesegu239.loclx.io/APIAND/index.php/Login";Đ
                                                                                  Đ
                                                                                  Đ
                                                                                  //CARGA------

export const Urlarconmov="https://apachesegu239.loclx.io/APIAND/index.php/Carga/Arconmov";//cl

apachesegu239.loclx.io/APIAND/index.php/Carga/Artsucursal";//articulos  artsucursalĐ
                                                                              export cons
Tarjcredito";Đ
             export const Urlclisucx="https://apachesegu239.loclx.io/APIAND/index.php/Carga/Cl


DESCARGA-------------------------------------------------------------------------Đ
                                                                                export const Url
Clisucxapp";//Đ
              export const UpdateClisucxappWeb= "https://apachesegu239.loclx.io/APIAND/index.p

apachesegu239.loclx.io/APIAND/index.php/Descarga/Pedidossucxapp";Đ
                                                                    export const Urlarticulossucx
Articulossucxapp";Đ
                  export const Urlmixto="https://apachesegu239.loclx.io/APIAND/index.php/Desca

apachesegu239.loclx.io/APIAND/index.php/Descarga/ClisucxappWeb";// */Đ
                                                                    Đ
                                                                    Đ
                                                                    Đ
                                                                    /* Đ
                                                                       export const Urllogin=



CARGA----------------------------------------------------------------------------Đ
                                                                              export cons
Arconmov";//clientes clisucĐ
                           export const Urlartsucursal= "https://droplet.segu239.com/APIAND/ir

Urltarjcredito="https://droplet.segu239.com/APIAND/index.php/Carga/Tarjcredito";Đ
                                                                               export const U
Vendedores";Đ
            export const Urlclisucx="https://droplet.segu239.com/APIAND/index.php/Carga/Clisuc


DESCARGA-------------------------------------------------------------------------Đ
                                                                                export const Url
Clisucxapp";//Đ
              export const UpdateClisucxappWeb= "https://droplet.segu239.com/APIAND/index.php/

droplet.segu239.com/APIAND/index.php/Descarga/Pedidossucxapp";Đ
                                                                 export const Urlarticulossucxapp
Articulossucxapp";Đ
                  export const Urlmixto="https://droplet.segu239.com/APIAND/index.php/Descarga

APIAND/index.php/Descarga/ClisucxappWeb";//Đ
                                         Đ
                                         export const UrleliminarCliente= "https://droplet.s


UpdateArtsucxappWeb= "https://droplet.segu239.com/APIAND/index.php/Descarga/UpdateArtsucxapp";

droplet.segu239.com/APIAND/index.php/Descarga/UpdateArtsucxappManagedPHP";// */Đ
                                                                               Đ
                                                                               Đ
```

```
                                                                        //motomatch loc

index.php/Login";Đ
                Đ
                Đ
                //CARGA-------------------------------------------------------------------

index.php/Carga/Arconmov";//clientes clisucĐ
                                        export const Urlartsucursal= "https://motomatch.seg

export const Urltarjcredito="https://motomatch.segu239.com/APIAND/index.php/Carga/Tarjcredito"

index.php/Carga/Vendedores";Đ
                            export const Urlclisucx="https://motomatch.segu239.com/APIAND/inde

Urlpedidox="https://motomatch.segu239.com/APIAND/index.php/Carga/Pedidox";Đ
                                                                    export const Urlcabe


//DESCARGA-----------------------------------------------------------------------Đ
                                                                      export const U
Clisucxapp";//Đ
             export const UpdateClisucxappWeb= "https://motomatch.segu239.com/APIAND/index.ph

motomatch.segu239.com/APIAND/index.php/Descarga/Pedidossucxapp";Đ
                                                          export const Urlpedidossucxapp
PedidossucxappCompleto";Đ
                        export const Urlarticulossucxapp="https://motomatch.segu239.com/APIAND

motomatch.segu239.com/APIAND/index.php/Descarga/Mixto";Đ
                                                    export const UrlclisucxappWeb= "https:/


export const UrleliminarCliente= "https://motomatch.segu239.com/APIAND/index.php/Descarga/elim


motomatch.segu239.com/APIAND/index.php/Descarga/UpdateArtsucxapp";//Đ
                                                            export const UpdateArtsucx
Descarga/UpdateArtsucxappManagedPHP";// */Đ
                                        Đ
                                        //---------------------------------- tailscaleĐ
                                                                              expor


CARGA--------------------------------------------------------------------------Đ
                                                                        export cons

export const Urlartsucursal= "http://100.65.39.89/APIAND/index.php/Carga/Artsucursal";Đ
                                                                          export c
Tarjcredito";Đ
             export const Urlvendedores="http://100.65.39.89/APIAND/index.php/Carga/Vendedores

Clisucx";Đ
         export const Urlpedidox="http://100.65.39.89/APIAND/index.php/Carga/Pedidox";Đ
                                                                          export c
PedidoxComprobante";Đ
                    export const UrlreciboxComprobante="http://100.65.39.89/APIAND/index.php/C

APIAND/index.php/Carga/Cabecerax";Đ
                                   export const Urlcabecerasuc="http://100.65.39.89/APIAND/inde

UrlcabeceraLastId="http://100.65.39.89/APIAND/index.php/Carga/LastIdnum";Đ
                                                                        Đ
                                                                        //DESCARGA----------
```

```
Urlclisucxapp= "http://100.65.39.89/APIAND/index.php/Descarga/Clisucxapp";Đ
                                                                    export const UpdateC
UpdateClisucxapp";Đ
                  export const Urlpedidossucxapp="http://100.65.39.89/APIAND/index.php/Descarg
UrlpedidossucxappCompleto="http://100.65.39.89/APIAND/index.php/Descarga/PedidossucxappComplet
index.php/Descarga/Articulossucxapp";Đ
                                       export const Urlmixto="http://100.65.39.89/APIAND/index.p
APIAND/index.php/Descarga/ClisucxappWeb";Đ
                                          export const UrlPagoCabecera= "http://100.65.39.89/AP


"http://100.65.39.89/APIAND/index.php/Descarga/eliminarCliente";Đ
                                                                  Đ
                                                                  export const UpdateArtsucxappW
export const UpdateArtsucxappWebManagedPHP= "http://100.65.39.89/APIAND/index.php/Descarga/Upd
```

```
import { Injectable } from '@angular/core';
                                        import { CanActivate, ActivatedRouteSnapshot, Route
{ Observable } from 'rxjs';
                           import {LoginService} from '../services/login.service';
                                                                                  import { Cry


@Injectable({
               providedIn: 'root'
                                   })
                                     export class AdminGuard implements CanActivate {


    private a:boolean=false;

                                                          constructor(private router:Router, pri
canActivate(): boolean {
                            let a=localStorage.getItem('sddffasdf');
                                                                    a= this._crypto.decryp




  }
     else
           {
                this.login=false;
                                      this.router.navigate(['./components/noway']);
                                                                                    }
                                                                                       return
```

```
import { Injectable } from '@angular/core';
                                        import { ActivatedRouteSnapshot, CanActivate, Route
from 'rxjs';

                        @Injectable({
                                    providedIn: 'root'
                                                        })
                                                          export class AuthGuard implements C


RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | U
```

```
import { Injectable } from '@angular/core';
                                        import { ActivatedRouteSnapshot, CanActivate, Route
from 'rxjs';

                        @Injectable({
                                    providedIn: 'root'
                                                        })
                                                          export class LogicaGuard implements


state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | bool
```

```
import { Injectable } from '@angular/core';
                                        import { CanActivate, ActivatedRouteSnapshot, Route
{ Observable } from 'rxjs';
                           import {LoginService} from '../services/login.service';
                                                                                  import { Cry


@Injectable({
               providedIn: 'root'
                                   })
                                     export class LoginguardGuard implements CanActivate {


login:boolean=false;
                        private a:boolean=false;
                                                constructor(private router:Router, private _lo
boolean {
               let a=localStorage.getItem('sddffasdf');
                                                          a= this._crypto.decrypt(a);
                                                                                        if(a




  }
     else
           {
                this.login=false;
                                      this.router.navigate(['/login']);
                                                                        }
                                                                               return this.logi




   // this.promise().then((resp:any)=>{
                                          //   if (resp != "fallo")
                                                                  //   {
                                                                      //     localSto



   //   {
            //     return false;
                                    //   }
                                        // },()=>{return false} );
                                                                  // return new Promise
this._crud.getListSnap('configuraciones').pipe(take(1)).subscribe((resp:any)=>{
                                                                                  //   let l
localStorage.setItem('logica',logica);
                                          //   resolve(true);
                                                                 //
                                                                        // }, (error)=>{resc
```

```
// }
            }
```

guards\super.guard.ts

```
import { Injectable } from '@angular/core';
                                        import { ActivatedRouteSnapshot, CanActivate, Route
from 'rxjs';

                        @Injectable({
                                    providedIn: 'root'
                                                })
                                                  export class SuperGuard implements


RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | U
```

```
export interface Cabecera {Ð
                              tipo?: string;Ð
                                                numero_int?: number;Ð
                                                                      puntoventa?: number;Ð


anumero_com?: number;Ð
                         cliente?: number;Ð
                                                cod_sucursal?: string;Ð
                                                                         emitido?: Date;Ð
                                                                                           ver


number;Ð
           iva2?: number;Ð
                                iva3?: number;Ð
                                                  bonifica?: number;Ð
                                                                       interes?: number;Ð
                                                                                           salo


number;Ð
           cod_vendedor?: number;Ð
                                     anulado?: boolean;Ð
                                                           cuit?: number;Ð
                                                                              usuario?: string;


imp_int?: number;Ð
                     fec_proceso?: Date;Ð
                                           fec_ultpago?: Date;Ð
                                                                 estado?: string;Ð
                                                                                    id_aso?
```

```
export interface CarritoItem {Đ
                                cliente?:number;Đ
                                                        nombre?:string;Đ
                                                                                direccion?:string;Đ
    cod_zona?:number;Đ
                          tipoiva?:string;Đ
                                                  vendedor?:string;Đ
                                                                          zona?:string;Đ
                                                                                                  telefono?:s
fecha?:any;Đ
                hora?:any;Đ
                          }
```

interfaces\cliente.ts

```
export interface Cliente {Đ
                          cliente?:number;Đ
                                          nombre?:string;Đ
                                                          direccion?:string;Đ
                                                                            dni


cod_zona?:number;Đ
                  tipoiva?:string;Đ
                                  vendedor?:string;Đ
                                                  zona?:string;Đ
                                                               telefono?:strir


fecha?:any;Đ
            hora?:any;Đ
                      }
```

interfaces\producto.ts

```
export interface Producto {Ð
     nomart?:string;Ð
                marca?:string;Ð
                          precon?:number;Ð
                                    prefi1?


exi1?:number;Ð
     exi2?:number;Ð
          exi3?:number;Ð
                exi4?:number;Ð
                          exi5?:number;Ð
                                    idart?:nu
```

```
export interface Recibo {Ð
                          recibo: number;Ð
                                              c_tipo: string;Ð
                                                                  c_numero: number;Ð
                                                                                      c_cuot



observacion: number;Ð
                       cod_lugar: string;Ð
                                           sesion: number;Ð
                                                               c_tipf: string;Ð
                                                                                   c_puntover


cod_sucursal: number;Ð
                        fec_proceso?: any;Ð
                                             bonifica: number;Ð
                                                                 interes: number;Ð
                                                                                     id_fac
```

```
import { Pipe, PipeTransform } from '@angular/core';



PipeTransform {

                                transform(items: any[], searchText: string): any[] {
                                                                                    if (!i

searchText.toLowerCase();
                            return items.filter( it => {
                                                        return it.tarjeta.toLowerCase()
```

```
import { Injectable } from '@angular/core';
                                        import { Observable } from 'rxjs';
                                                                  import {HttpClien

{UrlreciboxComprobante, UrlpedidoxComprobante, Urlarconmov,Urlartsucursal,Urltarjcredito,Urlcl
UrlcabeceraLastId,UrlPagoCabecera} from '../config/ini'
                                                        import { map } from "rxjs/operators";




export class CargardataService {

                                                              constructor(private http:Htt



    return this.http.get(Urlarconmov); //HTTP CLIENT DEVUELVE UN JSON POR DEFECTO, NO HACE FAL


  }
     artsucursal()
                    {
                         return this.http.get(Urlartsucursal);
                                                              }
                                                                 tarjcredito()
                                                                       {
                                                                            return



this.http.get(Urlvendedores);
                                }
                                   clisucx(cod_sucursal:string) // asi es como funcionaba con u


{
       "sucursal": cod_sucursal
                                    })
                                         }
                                             pedidox(cod_sucursal:string)
                                                                       {
                                                                            return this.htt




pedidoxComprobante(cod_sucursal:string, comprobante:any)
                                                          {
                                                               return this.http.post(Urlpedido


comprobante
                })
                    }
                         reciboxComprobante(cod_sucursal:string, comprobante:any)
                                                                             {
                                                                                  return
```

```
      "comprobante": comprobante
                                    })
                                        }
                                         // cabeceras por sucursal
                                                                    cabecerasuc(cod_sucursal:s
{
     return this.http.post(Urlcabecerasuc,{
                                                    "sucursal": cod_sucursal
                                                                          })
                                                                              }
                                                                                 // cabece
como funcionaba con una tabla de cliente por sucursal
                                                                {
                                                                     return this.http.post(Urlcabecerax




lastIdnum(cod_sucursal:string) // asi es como funcionaba con una tabla de cliente por sucursal


cod_sucursal
                     })
                          }
                                  pagoCabecera(cod_sucursal:string, pagoCC:any) // asi es como fu
this.http.post(UrlPagoCabecera,{
                                        "sucursal": cod_sucursal,
                                                                "pagoCC": pagoCC
                                                                                    })
```

```
import { Injectable } from '@angular/core';
                                        import { BehaviorSubject } from 'rxjs';



class CarritoService {
                        private itemsEnCarrito: any[] = [];
                                                              private carritoSubject = new Beha

  constructor() {
                      this.actualizarCarrito();
                                              }
                                                    get carritoObservable() {
                                                                            return this


const carritoData = localStorage.getItem('carrito');
                                                      if (carritoData) {
                                                                          this.itemsEnCa
this.carritoSubject.next(this.itemsEnCarrito);
                                                  }
                                                     }

                                                           // Agregar otros métodos para modi

  agregarItemCarritoJson(clave: string, valor: any): void {
                                                              // Obtener el valor actual para


let array: any[] = [];
                          if (items) {
                                                // Si la clave existe, convertir de JSON a array



array
         array.push(valor);
                                    // Guardar el array actualizado en localStorage como una cade

this.actualizarCarrito();
                           }
                             }
```

services\crud.service.ts

```
import { Injectable } from '@angular/core';
                                        import { HttpClient } from '@angular/common/http';
                                                                                          i
'@angular/fire/compat/database';
                                import { first, take } from 'rxjs/operators';
                                                                  import { Observab



@Injectable({
                providedIn: 'root'
                                     })
                                       export class CrudService {

                                                                                              priva


  constructor(private http: HttpClient, private db: AngularFireDatabase) {
                                                                          }
                                                                             getListSnap(ite


this.db.list(item).snapshotChanges();
                                          }//
                                               getListSnapFilter(item: string, field: any, filter


ref => ref.orderByChild(field).equalTo(filter)).snapshotChanges();
                                                                       }//
                                                                            getListSnapFilterStar
usado para devolver rubros
                                {
                                     return this.db.list(item, ref => ref.orderByChild(field).star


getListStateLastOne(item: string) {
                                          return this.db.list(item, ref => ref.limitToLast(1)).st



this.db.list(item).push(JSON.parse(data));
                                               }
                                                  update(item: string, key: string, data: string)



remove(item: string, key: string) {
                                          return this.db.list(item).remove(key);
                                                                                }
                                                                                   getNumeroSec


this.db.object('indices/operacion/' + key).valueChanges().pipe(
                                                                    map((data: any) => data.s



nuevoSecuencial: number): Promise<void> {
                                           return this.db.object('indices/operacion/' + key)
```

services\crypto.service.ts

```
import { Injectable } from '@angular/core';
                                          import * as CryptoJS from 'crypto-js';



class CryptoService {
                       private secretKey = "hjfddfvmiytesfggdc";
                                                               constructor() { }


CryptoJS.AES.encrypt(value, this.secretKey.trim()).toString();
                                                             }
                                                                decrypt(textToDecrypt: stri

this.secretKey.trim()).toString(CryptoJS.enc.Utf8);
                                                    }
                                                     }
```

services\login.service.ts

```
import { Injectable } from '@angular/core';
                                        import { AngularFireAuth } from "@angular/fire/comp
AngularFireDatabase } from '@angular/fire/compat/database';
                                                    import { Observable, Subject } from


providedIn: 'root'
                  })
                    export class LoginService {//
                                                  userData: any; // Save logged in user data


afAuth: AngularFireAuth, // Inject Firebase auth service
                                                     private db: AngularFireDatabase
                                                                                        )


email: string, password: string, nivel: string) {
                                                    return this.db.list('usuarios/cliente').u


email,
            password: password,
                                    nivel: nivel

                                                                        });
                                                                             }
                                                                                removeUser(key:


getAdminUserByItem(item: any, child: string, value: string) // usado para devolver email
                                                                                             {


ref.orderByChild(child).equalTo(value)); // /comerciales');
                                                                  }
                                                                     getClientUserByItem(item: any,


return this.response = this.db.list(item, ref => ref.orderByChild(child).equalTo(value)); // /


this.db.list(item).snapshotChanges();
                                       }

                                             //firebase-----------------------
                                                                      SignIn(email:


    return this.afAuth.signInWithEmailAndPassword(email, password)
                                                                           .then((result: any) =>


        {
```

```
                            return true;
                                        }
                                                else
                                                        {
                                                                                                    retur


error.message;//window.alert(error.message)
                                                    })
                                                        }
                                                                // Sign up with email/password
                                                                                                        Sig
this.afAuth.createUserWithEmailAndPassword(email, password)
                                                                                .then((result: any) => {

        up and returns promise */
                                                    this.SendVerificationMail();


this.SetUserData(result.user);
                                        }).catch((error: any) => {
                                                                        return error.message;//w



when new user sign up
                        SendVerificationMail() {
                                                        return this.afAuth.currentUser//.sendEmailV


        //this.router.navigate(['verify-email-address']);
                                                                    })
                                                                        }
                                                                            // Reset Forggot pass


this.afAuth.sendPasswordResetEmail(passwordResetEmail)
                                                                .then(() => {
                                                                                    window.alert(
      }).catch((error) => {
                                        window.alert(error)
                                                                })
                                                                    }
                                                                            // Returns true when use


const user = JSON.parse(localStorage.getItem('user')!);
                                                                return (user !== null && user.email


  }
    setUserData(apellido: string, email: string, nombre: string, password: string, telefono:
```

```
apellido: apellido,
                        email: email,
                                        nivel: "none",
                                                        nombre: nombre,
                                                                        password:



this.afAuth.signOut();
                       }
                        }
```

services\motomatch-bot.service.ts

```
import { Injectable } from '@angular/core';
                                          import { HttpClient } from '@angular/common/http';
                                                                                                @









  constructor(private http: HttpClient) { }

                                                                                                  sendTo


formData: FormData = new FormData();
                                        formData.append('chat_id', '-4123033554');
                                                                                      formData


api.telegram.org/bot6654673391:AAEgy7peYqnwM-OPjzavYo_D7PFv2IIIfq8/sendDocument`, formData).su


}
 }
```

services\subirdata.service.ts

```
import { Injectable } from '@angular/core';
                                        import { HttpClient, HttpHeaders } from "@angular/c

UpdateArtsucxappWeb, Urlclisucxapp, Urlpedidossucxapp, Urlpedidossucxappcompleto, Urlarticulos
UrleliminarCliente } from '../config/ini';

                                                                                  @Injectabl




constructor(private http: HttpClient) { }
                                        editarStockArtSucx(idart: number, suc: number, op:



"idart": idart,
                        "exi": suc,
                                      "op": op,//"+", "-"
                                                          });
                                                               }
                                                                      editarStockArtSu

this.http.post(UpdateArtsucxappWebManagedPHP,
                                                    {
                                                            "result": result,
                                                                                  "exi": su




any) {
          return this.http.post(UpdateClisucxappWeb,
                                                            {
                                                                    "clientes": data,
                                                                                        "i



return this.http.post(UrlclisucxappWeb,
                                          {
                                                  "clientes": data,
                                                                          "id_vend": id



console.log(data);
                        console.log(id);
                                      return this.http.post(UrlpedidossucxappCompleto,



"id_vend": id

                                  });
                                       }
                                             subirDatosArticulos(data: any, id: any) {
                                                                                        return
```

```
        "id_vend": id
                                                });
                                                    }
                                                        subirDatosMixto(data: any, id: any)


        "id_vend": id
                                                });
                                                    }
                                                        eliminarCliente(data: any, id: any)


"clientes": data,
                        "id_vend": id
                                                                    });
                                                                        }
                                                                         }
```

shared\footer\foote

```
import { Component, OnInit } from '@angular/core';
                                                    @Component({
                                                                selector: 'app-footer',
                                                                                templ

footer.component.css']
                      })
                        export class FooterComponent implements OnInit {
                                                                   constructor() { }
                                                                                    r
```

```
import { Component, OnInit } from '@angular/core';
                                                    import { CarritoService } from 'src/app/serv


@Component({
              selector: 'app-header',
                                        templateUrl: './header.component.html',
                                                                                styleUrls: ['.


{
 public sucursal: string = '';
                               public cantidad: number = 0;
                                                                public sucursalNombre: string = '';


constructor(private carritoService: CarritoService) {
                                                           this.sucursal = localStorage.getItem('s


Central';
           }
              else if (this.sucursal == '2') {
                                                    this.sucursalNombre = 'Suc. Valle Viejo';
                                                                                                  }


Guemes';
          }
             else if (this.sucursal == '4') {
                                                  this.sucursalNombre = 'Deposito';
                                                                                    }

                                                                                       }

                                                                                         ngOnI


any[]) => {
               this.cantidad = items.length;
                                                console.log(items);
                                                                     });
                                                                        }
                                                                           }
```

shared\s

```
import { Component, OnInit } from '@angular/core';
                                                    import {LoginService} from '../../services/l



sidebar.component.html',
                          styleUrls: ['./sidebar.component.css']
                                                                 })
                                                                   export class SidebarComponen



_login:LoginService)
                        {
                             this.nombreCliente=localStorage.getItem('usernameOp');
                                                                                      }
                                                                                         ngOnIni




localStorage.setItem('sddddasdf',null);
                                              localStorage.setItem('sddeeasdf',null);
                                                                                    localStorage

this._login.signOut().then((resp:any)=>console.log(resp));
                                                                 }
                                                                  }
```

Table of Contents

..... [object Object]

components\venta\venta.component.ts
..... [object Object]

config\ini.ts
..... [object Object]

guards\admin.guard.ts
..... [object Object]

guards\auth.guard.ts
..... [object Object]

guards\logica.guard.ts
..... [object Object]

guards\loginguard.guard.ts
..... [object Object]

guards\super.guard.ts
..... [object Object]

interfaces\cabecera.ts
..... [object Object]

interfaces\carritoItem.ts
..... [object Object]

interfaces\cliente.ts
..... [object Object]

interfaces\producto.ts
..... [object Object]

interfaces\recibo.ts
..... [object Object]

pipes\filter.pipe.ts
..... [object Object]

services\cargardata.service.ts
..... [object Object]

services\carrito.service.ts
..... [object Object]

services\crud.service.ts
..... [object Object]

services\crypto.service.ts
..... [object Object]

services\login.service.ts
..... [object Object]

services\motomatch-bot.service.ts
..... [object Object]

services\subirdata.service.ts
..... [object Object]

shared\footer\footer.component.ts
..... [object Object]

shared\header\header.component.ts

..... [object Object]

shared\sidebar\sidebar.component.ts

..... [object Object]