**Solución a problemas CORS (Access-Control-Allow-Origin)**

Para evitar que se presente la necesidad de instalar algún API en el browser para acceder el servidor remoto, se debe de realizar el siguiente procedimiento:

**Tutorial**: <https://www.youtube.com/watch?v=zq48aSVEwbQ>

1. Crea el archivo **proxy.conf.json** en el directorio raíz del proyecto.
2. Agregar la siguiente información:

Ejemplos URL:

[*https://celcer.sri.gob.ec/comprobantes-electronicos-ws/*RecepcionComprobantesOffline?wsdl](https://celcer.sri.gob.ec/comprobantes-electronicos-ws/RecepcionComprobantesOffline?wsdl)

*https://celcer.sri.gob.ec*/comprobantes-electronicos-ws/AutorizacionComprobantesOffline?wsdl

**proxy.conf.json**

{  
 "/ comprobantes-electronicos-ws/\*": {  
 **"target"**: **"** *https://celcer.sri.gob.ec***/"**,  
 **"secure"**: **false**,  
 **"changeOrigin"**: **true** }  
}

Ejemplo URL:

*https://manager.redblu.com.mx:8080/services/*dataservices.asmx

**proxy.conf.json**

{  
 "/services/\*": {  
 **"target"**: **"**https://manager.redblu.com.mx:8080/**"**,  
 **"secure"**: **false**,  
 **"changeOrigin"**: **true** }  
}

Descripción de los cambios en el archivo proxy.conf.json:

1. Se crea un **objeto** con el nombre que va enseguida del dominio en la URL, tal como está marcado en color azul claro.
2. Se indica en la etiqueta **target**, el protocolo (https) y el dominio tal como se marcó en color rojo.
3. El valor de la etiqueta **secure** debe de ser false.
4. En las llamadas al servicio donde se utiliza la URL mencionada en el paso anterior, se debe de eliminar la parte que se indica en la etiqueta target y debe de iniciar a partir del objeto creado:

**Ejemplo**

**Antes**: 'https://manager.redblu.com.mx:8080/services/dataservices.asmx';

**Después**: '/services/dataservices.asmx';

1. Modificar en el archivo **package.json** en la línea que se inicia el servidor:

*Antes: "start": "ng serve”,*

*Después: "start": "ng serve* ***--proxy-config proxy.conf.json****",*

1. En caso de estar el servicio iniciado, se debe detener e iniciarlo nuevamente de la siguiente manera:

*npm start*

**Migrar aplicación 2 o 4 a 5**

1. Eliminar o renombrar el directorio node\_modules de la raíz del proyecto.
2. Ejecutar el siguiente comando en la raíz del proyecto:

npm install @angular/animations@next @angular/common@next @angular/compiler@next @angular/compiler-cli@next @angular/core@next @angular/forms@next @angular/http@next @angular/platform-browser@next @angular/platform-browser-dynamic@next @angular/platform-server@next @angular/router@next typescript@2.4.2 rxjs@^5.5.3

1. Ejecutar el siguiente comando en la raíz del proyecto:

npm install

**ComboBox angular 4**

**Para mostrar el primer elemento de la lista como valor inicial**

<**select type="string" (ngModelChange)="pAtmSeleccionado()"**><**option \*ngFor="let *atm* of ipATMs; let *i* = index" [selected]="*i* == 0"**

**[ngValue]="*atm*"**>{{***atm***}}</**option**>  
</**select**>

**Para mostrar un texto como valor inicial**

<**select type="string" [(ngModel)]="atmSeleccionado" (ngModelChange)="pAtmSeleccionado()"**>  
 <**option value="" disabled**>Seleccione ATM</**option**>  
 <**option \*ngFor="let *atm* of ipATMs" [ngValue]="*atm*"**>{{***atm***}}</**option**>  
</**select**>

**Agrega imagen con jquery**

// Antes de insertar la imagen, se verifica si no se ha agregado previamente.

if ( $('#btnExpExel').length == 0) {  
 $('.data-table-header').append('<input id="btnExpExel" type=image src="assets/img/office\_excel.png" width="40" height="35" (click)="exportaJournal2Excel()">');  
 //$('.data-table-header').append('<button id="btnExpExel" (click)="exportaJournal2Excel($event)">Export</button>');  
  
 //<button (click)="saveFile()">Export</button>  
} else {  
 //this.exportToExcel();  
}

**Flujo de la pantalla Resumen de Movtos**

constructor

inicializaVariables

ngOnInit

resumenInicialOperaciones

mResumenOperaciones

mRretirosPorHora

mResumenPorBanco

pResumenDepositos

parametrosConsulta

pDatosDelJournal

\_soapService

obtenNumeroDePaginasLog

obtenNumeroDePaginasLog

obtenDatosJournal

obtenDatosLog

inicializaVariables

veficaHoraUlimaOperacion

incrementaBanco

incrementaErrorBanco

mResumenOperaciones

mRretirosPorHora

mResumenPorBanco

pResumenDepositos

calculaDownTime

veficaHoraUlimaOperacion

incrementaErrorBanco

ErroresPorBanco

mResumenOperaciones

pResumenDepositos

mResumenPorBanco

mRretirosPorHora

obtenNumeroDePaginasLog

obtenDatosJournal

pDenominacionesBilletes

**Recuperar la información de las últimas páginas del log después de la última consulta en caso de ser el mismo ATM, las mismas fechas y horas.**

1. Obtener el número de páginas y registros.
2. Verificar si las fechas, horas y el atm son los mismos que la consulta anterior, en caso de no ser así, continuar con el paso 3, de lo contrario, continuar con los pasos de este mismo punto.
   1. Concatenar la fecha, hora y el atm actual y dejarlos en la variable *paramsConsultaActual*.
   2. Comparar el valor del paso anterior con el valor de la variable *paramsConsultaAnterior.*
   3. Si los datos anteriores son idénticos, se debe de realizar una nueva consulta de los datos del Journal con los mismos parámetros.
   4. Si no son iguales, se debe de hacer una consulta con los nuevos parámetros.
3. Si los parámetros de la consulta anterior no son iguales, se debe de poner el valor cero a la variable *gNumPagsConsAnterior.*
4. Consultar los datos del Journal.
   1. Realizar la consulta del Journal utilizando como página inicial, el valor de la variable *gNumPagsConsAnterior.*
   2. En el for en el que se realiza la consulta del Journal, si el valor del índice es menor al de la última página, se concatenara el buffer en el arreglo global *gPaginasCompletasJournal* y debe de conservarse para no recuperar la información de estas páginas cuando se trate de los mismos parámetros.
   3. Si el índice del for es igual al del total de las páginas, se guarda el resultado de la consulta en la variable *gUltimaPaginaJournal* y se concatenara al arreglo que contendrá todos los datos a mostrar.
   4. Al terminar el for, concatenara el valor de las variables *gPaginasCompletasJournal + gUltimaPaginaJournal* y las mostrara en el grid.
5. Guardar en la variable *gNumPagsConsAnterior* el total de páginas.

**Noviembre 27**

**Página de iconos para el dashboard:** [**http://fa2png.io/r/stroke-7/**](http://fa2png.io/r/stroke-7/)

1. No se están inicializando los datos del retiro y del downtime del depósito después de la ETV.
2. Después de realizar el retiro de la ETV, empezar a calcular el downtime; debe de ser el evento de la ETV cuando se retira el efectivo, ya que puede intervenir el equipo para solucionar problemas, pero la hora para el downtime debe de ser cuando retira efectivo.
3. La consulta de operaciones está mostrando información en la pantalla “Resumen” sin que este activo el ATM (ej. 10.40.2.4).
4. En la ventana “Estatus de ATMs”, incluir el estatus de los cassetteros; este estatus se debe de realizar, consultando el log de hardware de los últimos 2 minutos con referencia a la última actualización del estatus de los ATMs.

Consultar el log de hardware con los siguientes criterios:

1. Iniciar el Journal en la última página.
2. Incluir el botón [Export Excel] a la pantalla del Journal.
3. Ordenar las columnas del journal a exportar igual que el actual.
4. Verificar con Robert, que información requiere en la pantalla de aclaraciones.
5. En el journal, mostrar en los siguientes colores los renglones de la siguiente manera:

Código de respuesta diferente de cero: rojo

Retiro OK: Verde

Consulta OK: Azul

Reverso: Naranja

1. Pantalla con información de Totales por Tienda.

**Dispositivos**

<data name="Human\_AFD" xml:space="preserve">

<value>Dispensador de Billetes</value>

</data>

<data name="Human\_BAR" xml:space="preserve">

<value>Lectora código de barras</value>

</data>

<data name="Human\_CDM" xml:space="preserve">

<value>Dispensador de Monedas</value>

</data>

<data name="Human\_CHK" xml:space="preserve">

<value>Lectora de Cheques</value>

</data>

<data name="Human\_DEP" xml:space="preserve">

<value>Aceptador de Billetes</value>

</data>

<data name="Human\_DeviceBus" xml:space="preserve">

<value>Bus de Dispositivos</value>

</data>

<data name="Human\_EPP" xml:space="preserve">

<value>Pin Pad</value>

</data>

<data name="Human\_ICM" xml:space="preserve">

<value>Lectora de Tarjetas</value>

</data>

<data name="Human\_PTR" xml:space="preserve">

<value>Impresora</value>

</data>

<data name="Human\_TIO" xml:space="preserve">

<value>Sensores</value>

</data>

<data name="Human\_Unknown" xml:space="preserve">

<value>Desconocido</value>

</data>

<data name="Human\_VDM" xml:space="preserve">

<value>Módulo del Fabricante</value>

</data>

case DeviceType.AFD:

return Labels.Human\_AFD + " [" + value + "] ";

case DeviceType.BAR:

return Labels.Human\_BAR + " [" + value + "] ";

case DeviceType.CDM:

return Labels.Human\_CDM + " [" + value + "] ";

case DeviceType.CHK:

return Labels.Human\_CHK + " [" + value + "] ";

case DeviceType.DEP:

return Labels.Human\_DEP + " [" + value + "] ";

case DeviceType.DeviceBus:

return Labels.Human\_DeviceBus + " [" + value + "] ";

case DeviceType.EPP:

return Labels.Human\_EPP + " [" + value + "] ";

case DeviceType.ICM:

return Labels.Human\_ICM + " [" + value + "] ";

case DeviceType.PTR:

return Labels.Human\_PTR + " [" + value + "] ";

case DeviceType.TIO:

return Labels.Human\_TIO + " [" + value + "] ";

case DeviceType.Unknown:

return Labels.Human\_Unknown + " [" + value + "] ";

case DeviceType.VDM:

return Labels.Human\_VDM + " [" + value + "] ";

default:

return value.ToString();