API Contabilidad

Referencia
Versión 1.0

Octubre 2019

API Contabilidad

¿Qué es?

Es un servicio del sistema de administración y finanzas diseñado para el correcto manejo de su base de datos.

Esta API permitirá a los módulos de Brokerage, Cross dock y Almacén enviar la información de los viajes, pedidos y/o almacenajes para ser incluidos en el sistema de facturación tanto de ***Cuentas por Cobrar*** como de ***Cuentas por Pagar***. Además, este regresara a cada uno un identificador para ubicar en sus respectivas tablas el elemento del cual se está hablando.

Propósito

El propósito de este documento es orientar a los desarrolladores de los sistemas que utilizaran esta API para que envíen la información que se necesita de forma correcta y, además, reciban la información devuelta por cada actualización a la base de datos.

Utilización de la API

Para la utilización de la API en los sistemas es necesario efectuar una función de llamada (Ajax o Fetch) con la URL establecida [INSERTE URL AQUI]. Se requiere que se envié un conjunto de datos dependiendo de la acción que se necesite realizar.

Tipo de datos

Campo	**Tipo**	**Descripción**
Folio	VarChar	Folio del viaje, almacenaje y/o pedido.
NombreCortoCliente	VarChar	Nombre corto del cliente al que se le cobrara.
NombreCortoProveedor	VarChar	Nombre corto del transportista al que se le pagara.
FechaDescarga	VarChar	Fecha en la que se descargó el viaje.
Moneda	VarChar	Tipo de moneda manejada por el cliente y/o transportista.
CostoSubtotal	Numeric/Float	Costo del viaje o almacenaje, sin tener en consideración el IVA ni la retención, truncado a 5 decimales.
CostoIVA	Numeric/Float	Costo del IVA total del viaje o almacenaje, truncado a 5 decimales.
CostoRetencion	Numeric/Float	Costo de la retención total del viaje o almacenaje, truncado a 5 decimales.
CostoTotal	Numeric/Float	Costo total de viaje tomando en consideración el IVA y la retención, truncado a 5 decimales.
PrecioSubtotal	Numeric/Float	Precio del viaje o almacenaje, sin tener en consideración el IVA ni la retención, truncado a 5 decimales.
PrecioIVA	Numeric/Float	Precio del IVA total del viaje o almacenaje, truncado a 5 decimales.
PrecioRetencion	Numeric/Float	Precio de la retención total del viaje o almacenaje, truncado a 5 decimales.
PrecioTotal	Numeric/Float	Precio total de viaje tomando en consideración el IVA y la retención, truncado a 5 decimales.
Status	VarChar	Estado del viaje actual. Para cancelar un viaje, basta con cambiar al estado “Cancelado”.
IsFacturaCliente	Bit/Boolean	Bandera que indica si la factura está dirigida al cliente.
IsFacturaProveedor	Bit/Boolean	Bandera que indica si la factura está dirigida al proveedor.
IsEvidenciaFisica	Bit/Boolean	Bandera que indica si ya se entregó evidencia física del viaje.
IsEvidenciaDigital	Bit/Boolean	Bandera que indica si toda la evidencia digital del viaje está en el sistema.

Si el sistema que se está tratando no tiene alguno de estos campos o no es necesario para su facturación, este puede no incluirse en el JSON de parámetros y se inicializara automáticamente como ***null.***

Ejemplo de llamada Fetch

```
var jParams = {
 "Folio": "XDD000068",
 "NombreCortoCliente": "Rafael",
 "NombreCortoProveedor": "Barcel",
 //"FechaDescarga": '2019-10-02 20:54:56',
 "Moneda": "Peso",
 "CostoSubtotal": 5000,
 "CostoIVA": 500,
 "CostoRetencion": 120,
 "CostoTotal": 6000,
 "PrecioSubtotal": 0,
 "PrecioIVA": 0,
 "PrecioRetencion": 0,
 "PrecioTotal": 0,
 "Status": "InRoute",
 "IsFacturaCliente": true,
 "IsFacturaProveedor": false,
 "IsEvidenciaFisica": false,
 "IsEvidenciaDigital": false,
 "IDConcepto": 1,
 "IDCliente": 0,
 "IDProveedor": 0,
 "Proyecto": 'XD',
};
```

Este es un ejemplo del JSON jParams el cual será enviado a la API por medio del Fetch. Este tiene todos los datos que se insertaran en la base de datos como un registro nuevo. Nótese que los datos no obligatorios (como FechaDescarga en el caso de almacén) pueden omitirse y serán inicializados con null. No obstante, si este dato omitido es un numérico/Flotante, este se inicializara en 0.

POST

El método POST se encarga de insertar un registro en la base de datos con los parámetros especificados en el jParams, en el cual no se pueden repetir folios. Si esto llegase a ocurrir, el sistema regresara una respuesta ***HTTP 400 BAD REQUEST***.

```
fetch('http://127.0.0.1:8000/api/', {
 method: 'POST',
 body: JSON.stringify(jParams),
 headers: {
  "Content-Type": "application/json; charset=utf-8",
}
```

Para acceder a esta función solo es necesario hacer el fetch al URL principal y enviar jParams para su inserción.

PUT

El método PUT se utiliza cuando se necesita actualizar la información de algún registro. Los campos que se necesitan actualizar se especifican en el jParams con la misma estructura que el POST.

```
fetch('http://127.0.0.1:8000/api/bkg000056/', {
 method: 'PUT',
 body: JSON.stringify(jParams),
 headers: {
  "Content-Type": "application/json; charset=utf-8",
}
```

Para acceder a esta función necesitamos especificar el folio del viaje/almacenaje que se va a editar, utilizar el método PUT y enviar el jParams con los campos que se cambiaran.

PATCH

El método PATCH se implementa cuando solo se necesita actualizar 1 campo del registro. De misma forma en jParams solo se especifica el campo a actualizar.

```
fetch('http://127.0.0.1:8000/api/bkg000056/', {
 method: 'PATCH',
 body: JSON.stringify(jParams),
 headers: {
  "Content-Type": "application/json; charset=utf-8",
}
```

La estructura del fetch es la misma que en PUT, solamente se cambia el método por PATCH.