Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento Ciencias de la Computación
IIC3103 – Taller de integración
Prof: A. Tagle

Descripción de sistemas

1 SISTEMA DE ÓRDENES DE COMPRA

Api REST

1.1 URLS

La ruta de conexión provisoria a la API es la siguiente:

- Desarrollo: http://mare.ing.puc.cl/oc
- Producción: http://moto.ing.puc.cl/oc

Esta url no posee ningún tipo de validación de seguridad por lo que será cambiada.

Incluir en todas las llamadas el header 'Content-Type: application/json'.

1.2 MODELO DE DATOS

1.2.1 Orden de compra

- Id (string)
- Fecha creación (datetime)
- Canal (ftp, b2c, b2b)
- Proveedor (string)
- Cliente (string)
- Sku (string)
- Cantidad (int)
- Cantidad despachada (int)
- Precio unitario (int)
- Fecha entrega (datetime)
- Fechas despachos (datetime[])
- Estado (creada, aceptada, rechazada, finalizada, anulada)
- Motivo rechazo (string, opcional)
- Motivo anulación (string, opcional)
- Notas (string)
- Id Factura (string)

1.3 MÉTODOS

1.3.1 Crear orden de compra

- Descripción: Crea una orden de compra de cierta cantidad de un producto. La orden de compra especifica un grupo proveedor del producto y un grupo cliente, que es quien genera la orden.
- Url: PUT /crear
- Parámetros

    - canal (string): Canal por el que se realizó la compra (b2b o b2c)
    - cantidad (int): Cantidad de productos a comprar.
    - sku (string): Sku del producto a comprar
    - cliente (string): Id de grupo que solicita el producto
    - proveedor (string): Id de grupo que provee el producto
    - precioUnitario (int): Precio unitario convenido entre proveedor y comprador para la transacción
    - fechaEntrega (int): Fecha solicitada para la entrega de productos en milisegundos a partir del 01/01/1970 UTC (formato Unix).
    - notas (string): campo para insertar notas a las órdenes
- Retorno
    - Orden de compra o error

1.3.2 Recepcionar orden de compra

- Descripción: Método que permite a un proveedor aceptar una orden de compra creada por un cliente.
- Url: POST /recepcionar/:id
- Parámetros
    - id (string): Id de la orden de compra a recepcionar
- Retorno
    - Orden de compra o error

1.3.3 Rechazar orden de compra

- Descripción: Método que permite a un proveedor rechazar una orden de compra creada por un cliente.
- Url: POST /rechazar/:id
- Parámetros
    - id (string): Id de la orden de compra a rechazar
    - rechazo (string): Motivo por el cual se está rechazando la orden
- Retorno
    - Orden de compra o error

1.3.4 Anular orden de compra

- Descripción: Método que permite a un cliente anular una orden de compra ya emitida por el mismo.
- Url: DELETE /anular/:id
- Parámetros
    - id (string): Id de la orden de compra a anular
    - anulación (string): Motivo por el cual se está anulando la orden
- Retorno
    - Orden de compra o error

1.3.5 Obtener orden de compra

- Descripción: Método que permite a un cliente obtener una orden de compra específica que él ha generado.
- Url: GET /obtener/:id
- Parámetros
    - id (string): Id de la orden de compra a consultar
- Retorno
    - Orden de compra o error

1.3.6 Despachar producto

- Descripción: Método que permite marcar los productos despachados de una orden de compra
- Url:
- Parámetros
    - id (string): Id de la orden de compra
- Retorno
    - Orden de compra o error

2 SISTEMA BANCO

2.1 URLS

La ruta de conexión provisoria a la API es la siguiente:

- Desarrollo: http://mare.ing.puc.cl/banco
- Producción: http://moto.ing.puc.cl/banco

Esta url no posee ningún tipo de validación de seguridad por lo que será cambiada.

Incluir en todas las llamadas el header 'Content-Type: application/json'.

2.2 MODELO

2.2.1 Transacción

Una transacción representa un movimiento de dinero desde una cuenta de origen a una cuenta de destino.

- Id (string)
- Fecha creación (datetime)
- Cuenta origen (string)
- Cuenta destino (string)
- monto (double)

2.2.2 Saldo

El saldo es el resumen de una cuenta. Este se actualiza cada 10 minutos de acuerdo a las transacciones recibidas.

- Id (string)
- Cuenta (string)
- monto (double)

2.3 MÉTODOS

2.3.1 Transferir

- Descripción: Transfiere un determinado monto desde una cuenta de origen a una cuenta de destino. Una vez realizada la transferencia, el monto queda inmediatamente liberado para su uso en otras transacciones.
- Url: PUT /trx
- Parámetros
    - monto (int): Monto de la transacción
    - origen (string): Cuenta a la que se le descontará el monto

    - destino (string): Cuenta a la que se le abonará el monto
- Retorno
    - Transacción o error

2.3.2 Obtener transacción

- Descripción: Método que permite obtener un transacción
- Url: GET /trx/:id
- Parámetros
    - id (string): Id de la transacción
- Retorno
    - Transacción o error

2.3.3 Obtener cartola

- Descripción: Método que permite una lista de transacciones entre una fecha de inicio y una fecha de fin.
- Url: POST /cartola
- Parámetros
    - fechaInicio (int): Fecha en formato unix (milisegundos a partir de 01/01/1970 UTC)
    - fechaFin (int): Fecha en formato unix (milisegundos a partir de 01/01/1970 UTC)
    - id: id de la cuenta que se está consultando
    - limit (int, opcional): límite de resultados que se enviarán
- Retorno
    - Lista de Transacciones y conteo de resultados o error

2.3.4 Obtener cuenta

- Descripción: Método que permite obtener la cuenta y el saldo de esta. El saldo se actualiza cada 10 minutos.
- Url: GET /banco/cuenta/:id
- Parámetros
    - id (string): id de cuenta que se está obteniendo
- Retorno
    - Saldo o error

2.4 PORTAL WEB PARA VENTAS EN LÍNEA

El banco pone a su disposición un portal tipo “webpay” para la realización de transferencias en línea para ventas de productos y servicios.

Para utilizar el portal, el grupo debe generar una boleta y llamar a los servicios que se detallan a continuación. Luego, el banco, a partir de la información de la boleta, obtiene los montos de la transacción a realizar y la cuenta a la cual se le depositará. La cuenta del cliente se obtiene mediante una autenticación del usuario que está pagando.

Para acceder a este portal web, una vez que se genera una boleta, el sitio donde se está generando la venta, deberá redirigir al usuario a la url:

- Desarrollo
    - http://url/banco/pagoenlinea?callbackUrl=URL_OK&cancelUrl=URL_FAIL&boletaId=ID
- Producción
    - http://url/banco/pagoenlinea?callbackUrl=URL_OK&cancelUrl=URL_FAIL&boletaId=ID

Si la transacción se realiza de manera exitosa, el usuario será redirigido a URL_OK. En caso de error o en el caso que el mismo usuario cancela la transacción, se redirigirá a URL_FAIL.

Las urls que se envían como parámetro deberán estar debidamente codificadas como uriComponent[1] para evitar problemas de malinterpretación de los parámetros. Es decir, que si se quiere redirigir al usuario al sitio *http://www.emol.com*, el parámetro deberá ser enviado como *http%3A%2F%2Fwww.emol.com*

Descripción de parámetros:

- URL_OK: corresponde a la url a la que se redirigirá en caso de éxito. El caso de éxito corresponde a aquel en que el usuario fue correctamente autenticado y la transferencia respectiva fue correctamente realizada. Por último, la boleta se marca como pagada.
- URL_FAIL: corresponde a la url a la que se redirigirá al usuario en caso de error o de cancelación de la transacción. En el caso de cancelación del usuario, la boleta se marca como rechazada y no se realiza ninguna transferencia.
- ID: Corresponde al id de boleta generada en el sistema de facturas.

[1] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/encodeURIComponent

3 SISTEMA FACTURAS

3.1 URLS

La ruta de conexión provisoria a la API es la siguiente:

- Desarrollo: http://mare.ing.puc.cl/facturas
- Producción: http://moto.ing.puc.cl/facturas

Esta url no posee ningún tipo de validación de seguridad por lo que será cambiada.

Incluir en todas las llamadas el header 'Content-Type: application/json'.

3.2 MODELO

3.2.1 Factura

- Id (string)
- Fecha creación (datetime)
- Proveedor (string)
- Cliente (string)
- Valor bruto (int)
- Iva (int)
- Valor total (int)
- Estado pago (pendiente, pagado, anulada, rechazada)
- Fecha pago (datetime)
- Id orden de compra (string)
- Motivo rechazo (string, opcional)
- Motivo anulación (string, opcional)

3.3 MÉTODOS

3.3.1 Emitir factura

- Descripción: Método que permite emitir una factura a partir de una orden de compra
- Url: PUT /
- Parámetros
    - oc (string): Id de la orden de compra en la cual se basará la factura
- Retorno
    - Factura o error

3.3.2 Obtener factura

- Descripción: Método que permite a un proveedor obtener una factura
- Url: GET /:id
- Parámetros

    - id (string): Id de la factura a consultar
- Retorno
    - Factura o error

3.3.3 Pagar factura

- Descripción: Método que permite a un proveedor marcar una factura como pagada
- Url: POST /pay
- Parámetros
    - id (string): Id de la factura
- Retorno
    - Factura o error

3.3.4 Rechazar factura

- Descripción: Método que permite a un cliente rechazar una factura creada por un proveedor.
- Url: POST /reject
- Parámetros
    - id (string): Id de la factura a rechazar
    - motivo: Motivo por el cual se está rechazando la factura
- Retorno
    - Factura o error

3.3.5 Anular factura

- Descripción: Método que permite a un proveedor anular una factura ya emitida por el mismo.
- Url: POST /cancel
- Parámetros
    - id (string): Id de la factura a anular
    - motivo: Motivo por el cual se está anulando la factura
- Retorno
    - Factura o error

3.3.6 Crear boleta

- Descripción: Método que permite a un proveedor crear una boleta para un cliente final.
- Url: PUT /boleta
- Parámetros
    - proveedor (id): Id de grupo de grupo que genera la boleta
    - cliente (string): Id de cliente (asignado por la empresa proveedora)
    - total (int): Monto total de la transacción
- Retorno
    - Factura o error

4 SISTEMA BODEGA

El sistema de gestión de stocks o sistema de bodega es un rudimentario sistema que permite apoyar el proceso de administración de las bodegas de la empresa.

Este sistema tiene algunos servicios web que permiten realizar operaciones sobre los productos almacenados en la bodega.

Hasta el momento la única gran falencia de este sistema es que, como una simplificación realizada por los desarrolladores, los espacios utilizados y libres de cada almacén se calculan en relación al número de productos y no en relación al volumen de estos. Dado que estos casos son excepcionales, son manejados internamente por los operarios de la bodega.

4.1 URLS

La ruta de conexión a la API es la siguiente:

- Desarrollo: http://integracion-2016-dev.herokuapp.com/bodega
- Producción: http://integracion-2016-prod.herokuapp.com/bodega

Incluir en todas las llamadas el header 'Content-Type: application/json'.

4.2 MODELO

4.2.1 Almacén

- Id (string)
- espacioUtilizado (int)
- espacioTotal (int)
- recepción (boolean)
- despacho (boolean)
- pulmón (boolean)

4.2.2 Producto

- Id (id de la instancia del producto en la bodega)
- Sku (string)
- AlmacenId (string)
- Costos (double) costos asociados al producto (fabricación más costos de bodega que se vayan acumulando)

4.3 MÉTODOS

4.3.1 getAlmacenes (GET)

- Descripción: Entrega información sobre los almacenes de la bodega solicitada.

- Url: /almacenes
- Parámetros:
- Retorno:
    - Almacenes[]
- Authorization: GET (llamada no tiene parámetros)

4.3.2 getSkusWithStock (GET)

- Descripción: Entrega una lista de skus que tienen stock en un almacén
- Url: /skusWithStock
- Parámetros:
    - almacenId: Id de almacén a consultar
- Retorno
    - Arreglo de objetos:

```
{
  _id : sku,   // sku
  total : int   // total de stock disponible para el stock
}
```

- Authorization: GETalmacenId (reemplazar con valor del parámetro)

4.3.3 getStock (GET)

- Descripción: Devuelve todos los productos que se encuentran guardados en un almacén
- Url: /stock
- Parámetros:
    - almacenId: Id de almacén a consultar
    - sku: Sku de producto a consultar
    - limit (opcional): límite de productos a entregar, 100 por defecto, máximo 200
- Retorno:
    - Productos[]
- Authorization: GETalmacenIdsku (reemplazar con valores de parámetros)

4.3.4 moverStock (POST)

- Descripción: Mueve un producto de un almacén a otro, siempre y cuando haya espacio en el almacén de destino
- Url: /moveStock
- Parámetros
    - productoId
    - almacenId: Almacén de destino
- Retorno:
    - Producto
- Errores:

    - Producto no encontrado
    - Almacén no encontrado
    - Traspaso no realizado debido a falta de espacio
- Authorization: POSTproductoIdalmacenId (reemplazar con valores de parámetros)

4.3.5 moverStockBodega (POST)

- Descripción: Mueve un producto de una bodega a otra, siempre y cuando haya espacio en el almacén de recepción de la bodega de destino
- Url: /moveStockBodega
- Parámetros
    - producotId
    - almacenId: Almacén de recepción de la bodega del grupo de destino
    - oc: Id de orden de compra
    - precio (int): precio al que se vendió el producto
- Retorno:
    - Producto
- Errores:
    - Producto no encontrado
    - Bodega no encontrada
    - Traspaso no realizado debido a falta de espacio
    - Producto no se encuentra en bodega de despacho
- Authorization: POSTproductoIdalmacenId (reemplazar con valores de parámetros)

4.3.6 despacharStock (DELETE)

- Descripción: Envía un stock a un cliente desde la bodega de despacho
- Url: /stock
- Parámetros
    - productId
    - dirección: String
    - precio: int, precio al que se vendió el producto
    - oc: string, id de la orden de compra para el cual se está despachando
- Retorno:
    - Boolean, true si despacho se cursó correctamente, false en otro caso
- Errores:
    - Producto no encontrado
- Authorization: DELETEproductoIddireccionpreciopedidoId (reemplazar con valores de parámetros)

4.3.7 producirStock (PUT)

Para realizar esta acción, primero se debe pagar la producción de acuerdo a las tablas de costos, y disponer de las materias primas necesarias en el almacén de despacho.

En caso de que no se haya pagado el precio correcto o que no existan suficientes materias primas, no se realizará la producción.

En el momento de entregar la producción solicitada, se despachará al almacén de recepción. En caso de no haber espacio suficiente en el almacén de despacho, se procederá a almacenar los productos en la bodega pulmón.

- Descripción: Solicitud de producir stock de un nuevo producto.
- Url: /fabrica/fabricar
- Parámetros
    - sku: string
    - trxId: string
    - cantidad: int
- Retorno:
    - Detalles de pedido
- Errores:
    - Pago incorrecto
    - Materias primas insuficientes
- Authorization: PUTskucantidadtrxId (reemplazar con valores de parámetros)

4.3.8 getCuentaFabrica (GET)

Provee el número de cuenta de la fábrica para pagar la producción

- Descripción: Entrega número de cuenta de la fábrica
- Url: /fabrica/getCuenta
- Parámetros
- Retorno:
    - Número de cuenta
- Errores:
- Authorization: GET

4.4 SEGURIDAD

Esta API cuenta con seguridad HMAC. Todos los request a los servicios deben llevar el siguiente encabezado:

Authorization: INTEGRACION grupoX:hash

Donde X es el número de grupo y hash es un hash generado utilizando la llave privada de cada grupo.

Para cada uno de los métodos disponibles, el hash debe ser generado utilizando el tipo de request (GET, POST, DELETE) y los valores concatenados de los parámetros de la llamada (en el orden indicado en la descripción de cada servicio). Luego, se debe generar hash HMAC-SHA1 codificado en base 64.

Ejemplo

Grupo 1 realiza request GET con parámetros almacenId=534960ccc88ee69029cd3fb2. En este caso la clave privada del grupo es acbd12345

String para realizar hash: GET534960ccc88ee69029cd3fb2

Hash HMAC-SHA1 en base 64 generado: audZHU593D+EdOhUzRLE9bfF8bI=

Header a agregar al request:

Authorization: INTEGRACION grupo1:audZHU593D+EdOhUzRLE9bfF8bI=