



**Nombre:**

Christian Hernandez

Jessica Ñauta

**Asignatura:**

Gerencia Informática

**Ciclo:**

10° Ciclo

**Carrera:**

Ingenierías de Sistemas

**Docente:**

Ing. Christian Timbi

**Año lectivo:**

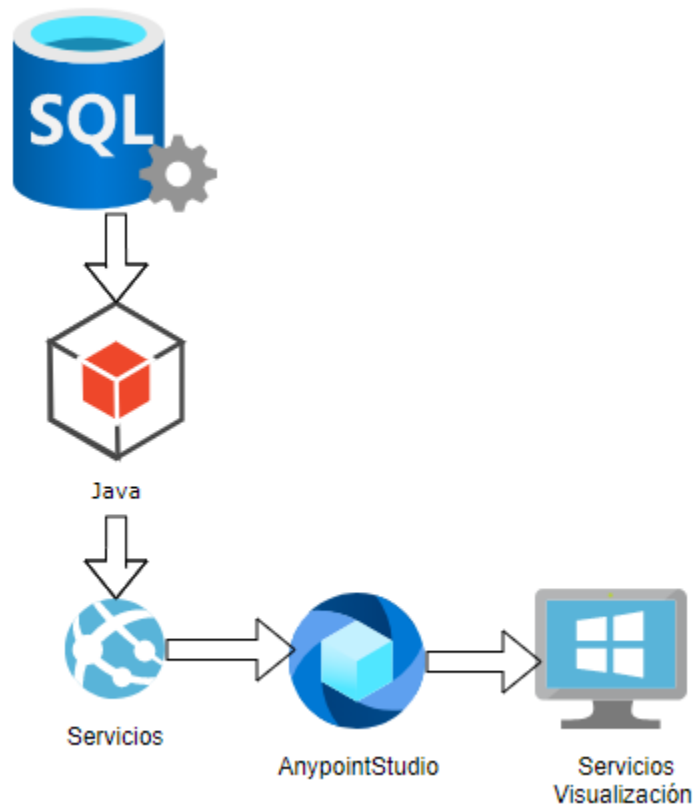
2020-2021

## Informe

Nuestro proyecto está basado en un sistema de recargas móvil en el cual se debe insertar el cliente con su respectivo número de teléfono para luego proceder a realizar la recarga dependiendo de la operadora que tenga el cliente.

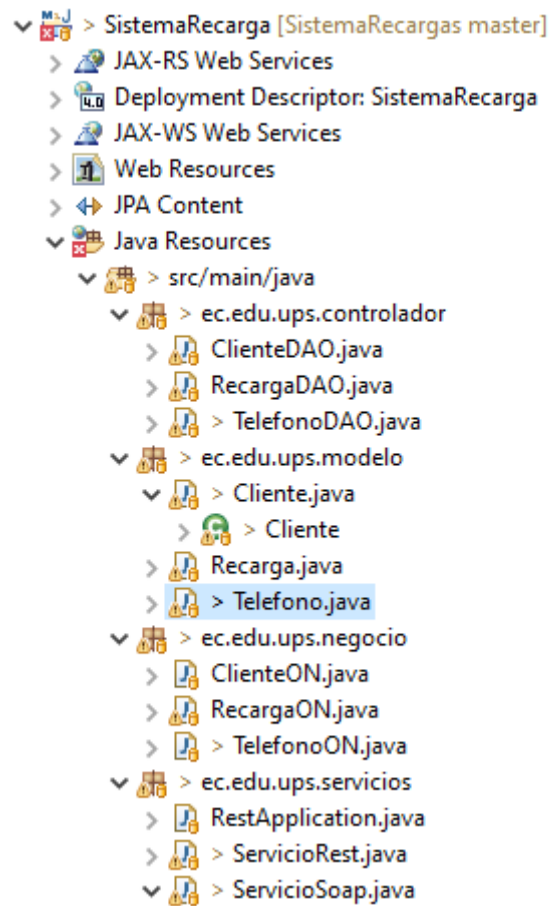
### Arquitectura:

Nuestra arquitectura consta de una base de datos en mySQL, la cual se conecta con nuestro SistemaRecargas realizada en eclipse, dentro del cual están los servicios web SOAP crearCliente, recarga y listar los cuales son consumidos mediante el ESB en nuestro caso usamos la aplicación llamada AnyPoint Studio el cual consume los servicios mediante la herramienta postman instalada en Google Chrome.



### Proyecto en eclipse

Primero creamos un proyecto llamado SistemaRecarga con sus respectivos paquetes y clases



```

SistemaRecarga [SistemaRecargas master]
├── JAX-RS Web Services
├── Deployment Descriptor: SistemaRecarga
├── JAX-WS Web Services
├── Web Resources
├── JPA Content
├── Java Resources
│   └── src/main/java
│       ├── ec.edu.ups.controlador
│       │   ├── ClienteDAO.java
│       │   ├── RecargaDAO.java
│       │   └── TelefonoDAO.java
│       ├── ec.edu.ups.modelo
│       │   ├── Cliente.java
│       │   │   └── Cliente
│       │   ├── Recarga.java
│       │   └── Telefono.java
│       ├── ec.edu.ups.negocio
│       │   ├── ClienteON.java
│       │   ├── RecargaON.java
│       │   └── TelefonoON.java
│       └── ec.edu.ups.servicios
│           ├── RestApplication.java
│           ├── ServicioRest.java
│           └── ServicioSoap.java

```

---

Dentro del paquete modelo creamos las clases Cliente, Recarga y Teléfono con sus respectivos atributos y con los métodos getters y setters.

```

1 package ec.edu.ups.modelo;
2
3 import java.io.Serializable;
4
14
15 @Entity
16 public class Cliente implements Serializable{
17
18     //Atributos de la entidad
19     @Id
20     private String cedula;
21     private String nombre;
22     private String apellido;
23     private String correo;
24
25     /**
26      * Constructor de la clase
27      */
28     public Cliente() {
29
30     }
31     /**
32      * Metodo que permite obtener el atributo cedula
33      * @return El atributo cedula de esta clase
34      */
35     public String getCedula() {
36         return cedula;
37     }
38 }

```

Dentro del paquete creamos las clases ClienteDAO, TelefonoDAO, RecargaDAO, con los métodos para insertar la información en la base de datos, para buscar y listar.

```

1 package ec.edu.ups.controlador;
2
3 import java.util.List;
4
11
12 /**
13  * Esta clase me permite hacer las funciones basicas en una base de datos
14  * utilizando la clase Cliente
15  *
16  * @version 1.0
17  *
18  */
19
20 @Stateless
21 public class ClienteDAO {
22
23     // Atributo de la clase
24     @PersistenceContext(name = "SistemaRecargaPersistenceUnit")
25     private EntityManager em;
26
27     /**
28      * Metodo que permite registrar un cliente en la base de datos
29      *
30      * @param c Cliente que se va a registrar en la base
31      */
32     public void insert(Cliente c) {
33         if (read(c.getCedula()) != null) {
34             em.persist(c);
35         } else {
36             update(c);
37         }
38     }
39 }

```

A continuación, dentro del paquete Negocio creamos las clases ClienteON, RecargaON y TelefonoON y consumimos mediante el inject los métodos que hemos creado en las clases DAO.

```
ClienteDAO.java
1 package ec.edu.ups.controlador;
2
3 import java.util.List;
4
11
12 /**
13  * Esta clase me permite hacer las funciones basicas en una base de datos
14  * utilizando la clase Cliente
15  *
16  * @version 1.0
17  *
18  */
19
20 @Stateless
21 public class ClienteDAO {
22
23     // Atributo de la clase
24     @PersistenceContext(name = "SistemaRecargaPersistenceUnit")
25     private EntityManager em;
26
27     /**
28     * Metodo que permite registrar un cliente en la base de datos
29     *
30     * @param c Cliente que se va a registrar en la base
31     */
32     public void insert(Cliente c) {
33         if (read(c.getCedula()) != null) {
34             em.persist(c);
35         } else {
36             update(c);
37         }
38     }
39 }
```

Ahora procedemos a crear los servicios Web SOAP dentro del paquete servicios.

Primero creamos un método llamado **creaCliente** en el cual le pasamos los parámetros cédula, nombre, apellido, correo y número de tipo String, también un saldo y saldoAnterior de tipo double, este método permite crear un cliente con sus datos incluyendo el número de teléfono con su respectivo saldo y le guarda en la base de datos.

```
@WebMethod
public String creaCliente(String cedula, String nombre, String apellido, String correo, String numero, double saldo,
    double saldoAnterior) throws Exception {
    Cliente c = new Cliente();
    Telefono t = new Telefono();
    c.setNombre(nombre);
    c.setApellido(apellido);
    c.setCedula(cedula);
    c.setCorreo(correo);
    onCliente.guardarCliente(c);

    t.setCliente(c);
    t.setNumero(numero);
    t.setSaldo(saldo);
    t.setSaldoAnterior(saldoAnterior);

    onTelefono.guardarTelefono(t);

    return "Cliente registrado con éxito";
}
```

Creamos el método llamado **realizarRecarga** en el cual le pasamos como parámetros un número de tipo String, un saldo de tipo double y una operadora de tipo String, este método sirve para realizar la recarga, primero busca el número de teléfono del cliente dentro de la base de datos para ver si está registrado, una vez que encuentra el número procede a hacer la recarga con el valor y el tipo de operadora.

```
@WebMethod
public String realizarRecarga(String numero, double saldo, String operadora) throws Exception {
    Recarga r = new Recarga();
    Telefono t = onTelefono.obtenerTelefono(numero);
    System.out.println("telefono" + t);
    //Telefono t = onTelefono.buscarTelefono(numero);
    //System.out.println("teeeeeeelefono: "+t);
    r.setTelefono(t);
    r.setOperadora(operadora);
    r.setSaldo(saldo);
    onRecarga.guardarRecarga(r);

    if (t.getSaldoAnterior() != 0) {
        t.setSaldoAnterior(t.getSaldoAnterior());
        t.setSaldo(saldo + t.getSaldoAnterior());
        onTelefono.actualizarTelefono(t);
    } else {
        t.setSaldoAnterior(t.getSaldo());
        t.setSaldo(saldo + t.getSaldoAnterior());
        onTelefono.actualizarTelefono(t);
    }

    return "Recarga realizada con éxito";
}
```

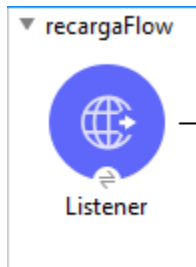
Finalmente creamos el método llamado **getRecargas()** el cual me permite obtener una lista de la base de datos de todos los números de teléfonos con sus clientes y sus respectivos datos.

```
@WebMethod
public ArrayList getRecargas(){
    List datos=new ArrayList();
    List<Recarga> lstRecargas=onRecarga.listaRecargas();
    for (Recarga r : lstRecargas) {
        //datos cliente
        datos.add(r.getTelefono().getCliente().getCedula());
        datos.add(r.getTelefono().getCliente().getNombre());
        datos.add(r.getTelefono().getCliente().getApellido());
        datos.add(r.getTelefono().getNumero());
        datos.add(r.getTelefono().getSaldo());
        datos.add(r.getTelefono().getSaldoAnterior());

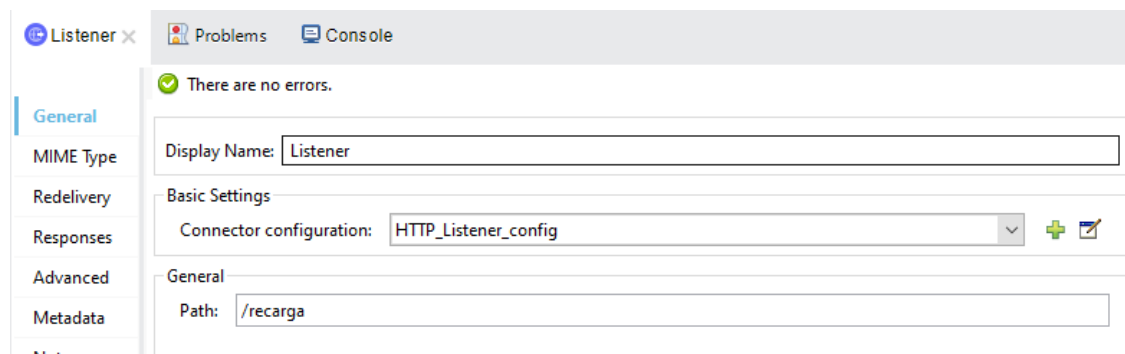
        //saldo de rla recarga
        datos.add(r.getOperadora());
        datos.add(r.getSaldo());
        System.out.println(""+datos);
    }
    return (ArrayList) datos;
}
```

A continuación, debemos de descargarnos la aplicación Anypoint Studio ya que mediante esta consumiremos los servicios de eclipse mediante el ESB.

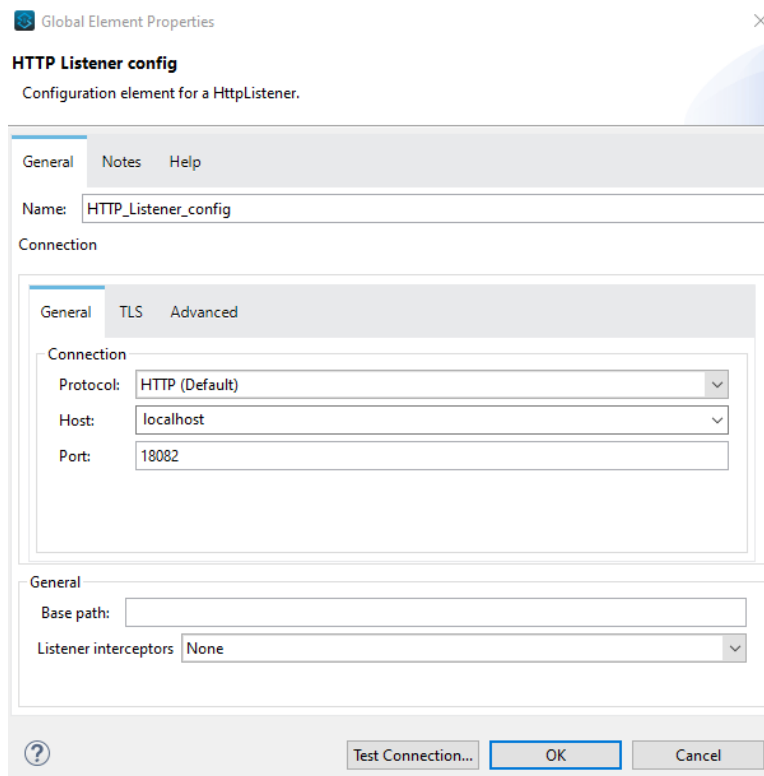
Primero creamos un proyecto llamado recargas en el cual arrastraremos el componente listener que es el escucha HTTP es una fuente de eventos que le permite configurar un servidor HTTP y desencadenar flujos cuando se reciben solicitudes HTTP.



Dentro de las configuraciones de listener debemos poner el path y crear una configuración para el conector.



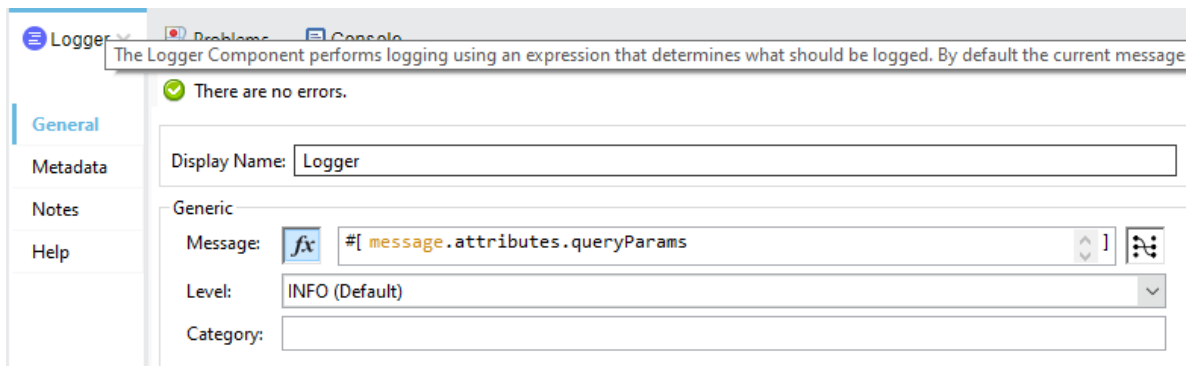
En la configuración para el conector debemos de poner el localhost y el número de puerto en el que queremos ejecutar el servicio.



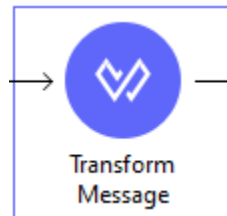
Ahora arrastramos el componente llamado Logger que agrega un registrador en cualquier lugar de un flujo y puede configurarlo para registrar una cadena que especifique, la salida de una expresión de DataWeave que escriba o cualquier combinación de cadenas y expresiones.



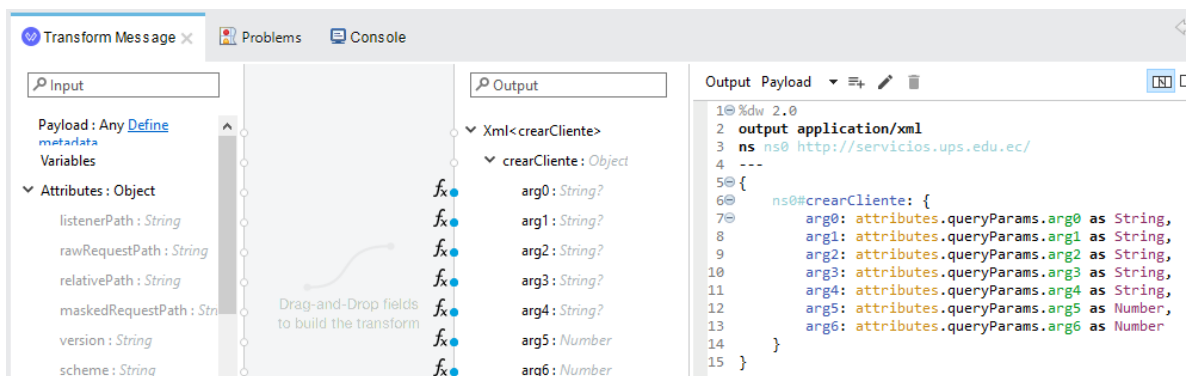
Debemos de enviar como sms los atributos para poder agregar parámetros de entrada.



A continuación, arrastramos el componente Transform Message el cual permite realizar transformaciones sobre los datos de entrada que recibe. Puede escribir explícitamente una transformación en lenguaje DataWeave, o puede usar la interfaz de usuario para construirla arrastrando y soltando elementos.

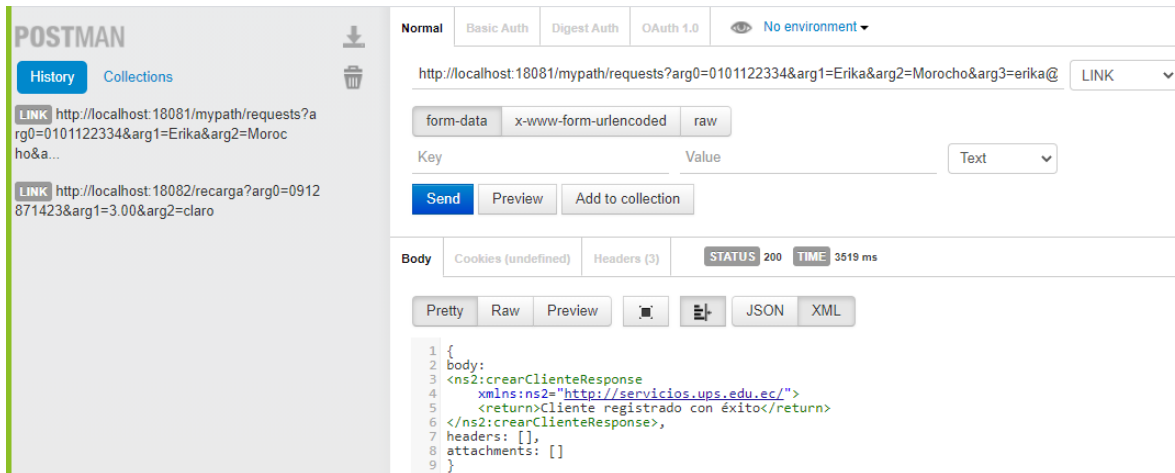


Aquí debemos de irnos a la parte que dice Attributes y arrastrar queryParams a cada uno de nuestras variables.





Debemos crear el cliente con su respectivo número de celular y saldo pasando el enlace del localhost con su respectivo puerto en postman y veremos el siguiente resultado que el servicio es consumido y los datos son guardados en la BD



Finalmente arrastramos el componente llamado web service consume el cual consume un servicio web SOAP de una aplicación Mule para adquirir datos de una fuente externa.



Dentro de este componente nos dirigimos a la opción de Connector configuration y se abrirá la ventana que se muestra a continuación, dentro de la cual debemos de poner el wsdl que obtenemos del servicio SOAP de nuestro proyecto de sistemaRecargas creada en eclipse, los otros datos se cargaran automáticamente.

Global Element Properties

## Web Service Consumer Config

Default configuration

General   Advanced   Notes   Help

Name:

Connection

General   Security   Transport   Advanced

Soap version:  SOAP11 (Default) ▼

Mtom enabled:  ▼

Encoding:

Connection

WSDL location:   ...

Service:   ▼ ⚙

Port:   ▼

Address:   ▼

?

OK Cancel

En la parte de connector configuration escogemos la configuración creada anteriormente y en operation escogemos el método con el cual vamos a trabajar.

Consume x Problems Console

General

Advanced

Error Mapping

Metadata

Notes

Help

✓ There are no errors.

Display Name:

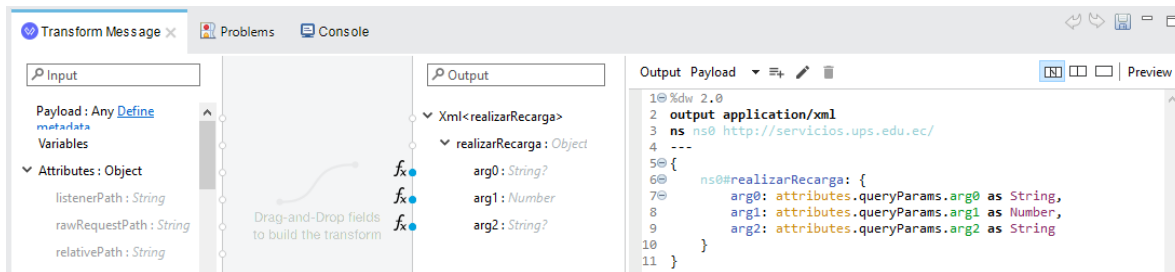
Basic Settings

Connector configuration:  ▼ + ✎

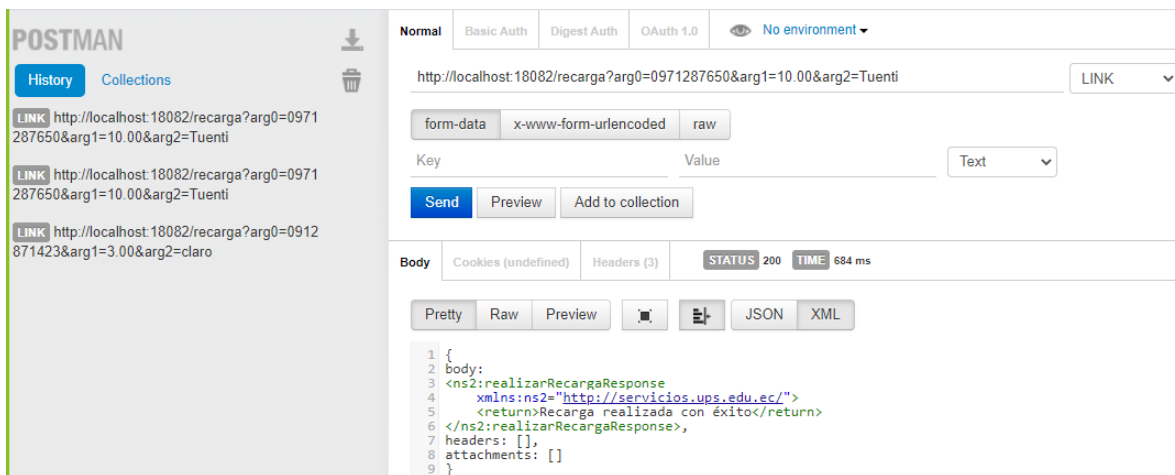
General

Operation:   ▼ ⚙

Ahora para realizar la recarga hacemos lo mismo que el paso anterior la única diferencia es en el componente de Transform Message ya que aquí primero se manda a buscar el número de teléfono en la base de datos para realizar la recarga.



Debemos crear la recarga pasando el enlace del localhost con su respectivo puerto en postman y veremos el siguiente resultado que el servicio es consumido y los datos son guardados en la BD



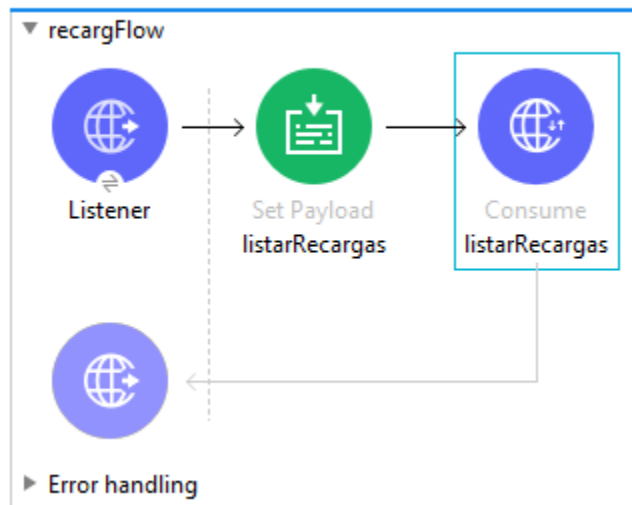
En la consola de eclipse podemos observar que los datos son agregados a BD.

```

16:36:14,121 INFO [stdout] (default task-1) into
16:36:14,122 INFO [stdout] (default task-1) Recarga
16:36:14,122 INFO [stdout] (default task-1) (operadora, saldo, idTelefono)
16:36:14,122 INFO [stdout] (default task-1) values
16:36:14,122 INFO [stdout] (default task-1) (?, ?, ?)
16:36:14,131 INFO [stdout] (default task-1) Hibernate:
16:36:14,131 INFO [stdout] (default task-1) select
16:36:14,132 INFO [stdout] (default task-1) telefono0_.idTefono as idTefono1_2_0_,
16:36:14,132 INFO [stdout] (default task-1) telefono0_.cedula as cedula5_2_0_,
16:36:14,132 INFO [stdout] (default task-1) telefono0_.numero as numero2_2_0_,
16:36:14,132 INFO [stdout] (default task-1) telefono0_.saldo as saldo3_2_0_,
16:36:14,132 INFO [stdout] (default task-1) telefono0_.saldoAnterior as saldoAnt4_2_0_
16:36:14,132 INFO [stdout] (default task-1) from
16:36:14,132 INFO [stdout] (default task-1) Telefono telefono0_
16:36:14,132 INFO [stdout] (default task-1) where

```

Ahora para el listar realizamos el mismo procedimiento anterior.



Arrastramos el componente Payload que permite actualizar la carga útil del mensaje.

listarRecargas

Problems

Console

Progress

There are no errors.

General

MIME Type

Metadata

Notes

Display Name:

listarRecargas

Settings

Value:

fx

#[ payload

Resultados del Listar

http://localhost:8081

No Environment

LINK

http://localhost:8081/sopare

Params

Send

Authorization

Headers

Body

Pre-request Script

Tests

Type

No Auth

Body

Cookies

Headers (3)

Test Results

Status

Pretty

Raw

Preview

XML

```
1 {
2   body:
3   <ns:getRecargasResponse xmlns:ns2="http://servicios.ups.edu.ec/">
4     <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">0106393291</return>
5     <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">cris</return>
6     <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">hernandez</return>
7     <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">07345439234</return>
8     <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">20.0</return>
9     <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
10    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">c1</return>
11    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
12    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">0106393291</return>
13    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">cris</return>
14    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">hernandez</return>
15    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">07345439234</return>
16    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">20.0</return>
17    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
18    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">c1</return>
19    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
20    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">0106393291</return>
21    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">cris</return>
22    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">hernandez</return>
23    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">07345439234</return>
24    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">20.0</return>
25    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
26    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">c1</return>
27    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
28    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">0106393291</return>
29    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">cris</return>
30    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">hernandez</return>
31    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">07345439234</return>
32    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">20.0</return>
33    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
34    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">c1</return>
35    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
36    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">0106393291</return>
37    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">cris</return>
38    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">hernandez</return>
39    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">07345439234</return>
40    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">20.0</return>
41    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:double">10.0</return>
42    <return xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:string">c1</return>
```