



UNIVERSIDAD VERACRUZANA

FACULTAD DE ESTADÍSTICA E INFORMÁTICA

**E.E TECNOLOGÍAS PARA LA INTEGRACIÓN DE
SOLUCIONES**

DOCENTE: ROJANO CACERES JOSE RAFAEL

**IMPLEMENTACIÓN DE MICROSERVICIOS PARA
TIENDAS DE COMERCIO ELECTRÓNICO**

EQUIPO 7

INTEGRANTES:

- **CRISTINO MORALES DIANA LAURA**
- **FLORES SANTAMARIA ANDRES**
- **SOSA MARTINEZ MANUEL ANTONIO**

Índice de contenido

Contenido

Índice de contenido	1
Introducción.....	3
Motivación, por qué se desarrolla el proyecto	3
Problemática	4
Solución.....	4
Costos, realizar una estimación de costos del proyecto, sueldos, contrataciones, recursos.....	5
Diagrama de despliegue donde se identifican los servicios	7
En el diagrama a continuación se muestra una visión general de la arquitectura con la que se compondrá nuestro aplicativo.	7
Documentación del API SOAP y REST	9
Microservicios y funciones del sistema	9
Métodos para implementar.....	9
Microservicio de catalogo	9
Microservicio Carrito.....	10
Microservicio Pedido	11
Función	11
Descripción	11
El usuario incluye productos en su compra.	11
El usuario puede eliminar su pedido.	11
El usuario ve el listado de su pedido.....	11
XML schema XSD, WSDL	11
EndPoints	16
Parámetros de recepción.....	21
Parámetros devueltos.....	21
Forma de ejecución de los contenedores.....	22
Proyecto publicado en GitHub.....	23
Despliegue en heroku	23
Pruebas	23
Microservicio-pedido	23
Microservicio-catalogo.....	25
Microservicio-carrito.....	28
Conclusiones	29

Introducción

Durante la experiencia educativa Tecnologías para la Integración de Soluciones aprendimos a crear servicios web de tipo SOAP y REST. Para evidenciar dichos conocimientos, para nuestro proyecto final decidimos implementar un servicio web para una tienda en línea.

El propósito de este proyecto consiste en implementar el back-end de tiendas de tipo comercio electrónico basándose en el estilo arquitectónico de microservicios y que dará lugar a un sistema compuesto por un conjunto de servicios web independientes que se irán desplegando según las necesidades que se tengan. Con este enfoque se favorecerá la escalabilidad y el rendimiento y la gestión de la propia aplicación final.

El objetivo final es construir la base funcional de una tienda en línea que podrá ser usado por cualquier aplicación web o móvil relativo a esta área.

La construcción de los microservicios se llevará a cabo usando el modelo contrato primero utilizando herramientas como el framework de Spring boot, tecnologías como Maven, Spring JPA, la plataforma clever cloud para la base de datos y los servicios se probarán con wizzler y postman. Otras herramientas para manejar el repositorio son Git, Docker y se desplegará en heroku.

El sistema debe ser capaz de gestionar los datos relativos a los usuarios/clientes login, productos, compras y pedidos. Los microservicios con los que se compondrá el sistema este “Gestionar el catálogo de productos”, “Gestionar el carrito de la compra” y “Gestión de pedidos”.

En este documento se plasma la documentación de la implementación para este servicio, dando a conocer a mayor detalle la problemática y la solución para esta. A su vez, se mostrará con mayor detalle la funcionalidad de estos microservicios.

Motivación, por qué se desarrolla el proyecto

Lo que nos motivo a realizar este servicio es poder crear una API que ayude a dar un mejor servicio y control de los productos que se vende dentro de ella. El sistema es capaz de gestionar los datos relativos a los usuarios/clientes login, productos, compras y pedidos. Se plantea crear microservicios que ayude a realizar las actividades en un menor tiempo de manera eficiente,

Problemática

Nuestra problemática está enfocada en una tienda en línea, en la que debido al aumento de ventas requería de un mejor sistema para agilizar la administración de los productos, así como también permitir que los clientes tengan mayor comodidad para realizar la compra.

La API o servicio web le permitirá a cualquier plataforma de comercio electrónico integrar fácilmente funcionalidades básicas para la experiencia de compras en línea de los usuarios como la administración de los productos(identificación de artículos, descripciones, información de precios) así como la administración de la compra(gestión de pedidos y pagos).

Solución

Para nuestra propuesta de solución para la problemática antes mencionada es crear un API con 3 microservicios los cuales son los siguientes:

Para SOAP se utilizó:

- Microservicio de catálogo que tiene las siguientes funcionalidades:
 - ✓ Listado de productos
 - ✓ Actualización de un producto en el catálogo
 - ✓ Inclusión de un producto en el catálogo
 - ✓ Eliminación de un producto en el catálogo
 - ✓ Listado de categorías disponibles
 - ✓ Inclusión de una categoría
 - ✓ Listado de productos por categoría

Para REST se utilizó:

- Microservicio de pedidos que tiene las siguientes funcionalidades:
 - ✓ Incluir pedidos
 - ✓ Eliminar un pedido
 - ✓ Listar un pedido
- Microservicio de carrito que tiene las siguientes funcionalidades:
 - ✓ Añadir productos al carrito
 - ✓ Quitar productos del carrito
 - ✓ Listar productos del carrito

- ✓ Confirmar compra

Costos, realizar una estimación de costos del proyecto, sueldos, contrataciones, recursos.

A continuación, se muestran los recursos que se utilizaron para el desarrollo de los microservicios antes mencionados.

- **Visual studio code:** En cuanto a la codificación se utilizó visual studio code ya que es gratuito y de código abierto.



- **Heroku:** Para hacer el despliegue de la aplicación de los microservicios se utilizó esta aplicación por su versión gratuita.



- **Clever cloud:** utilizamos esta base de datos por su versión gratuita.



- **Docker:** es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software.



- **GitHub:** es un sitio web y un servicio en la nube que ayuda a los desarrolladores a almacenar y administrar su código, al igual que llevar un registro y control de cualquier cambio sobre este código y es gratuito.

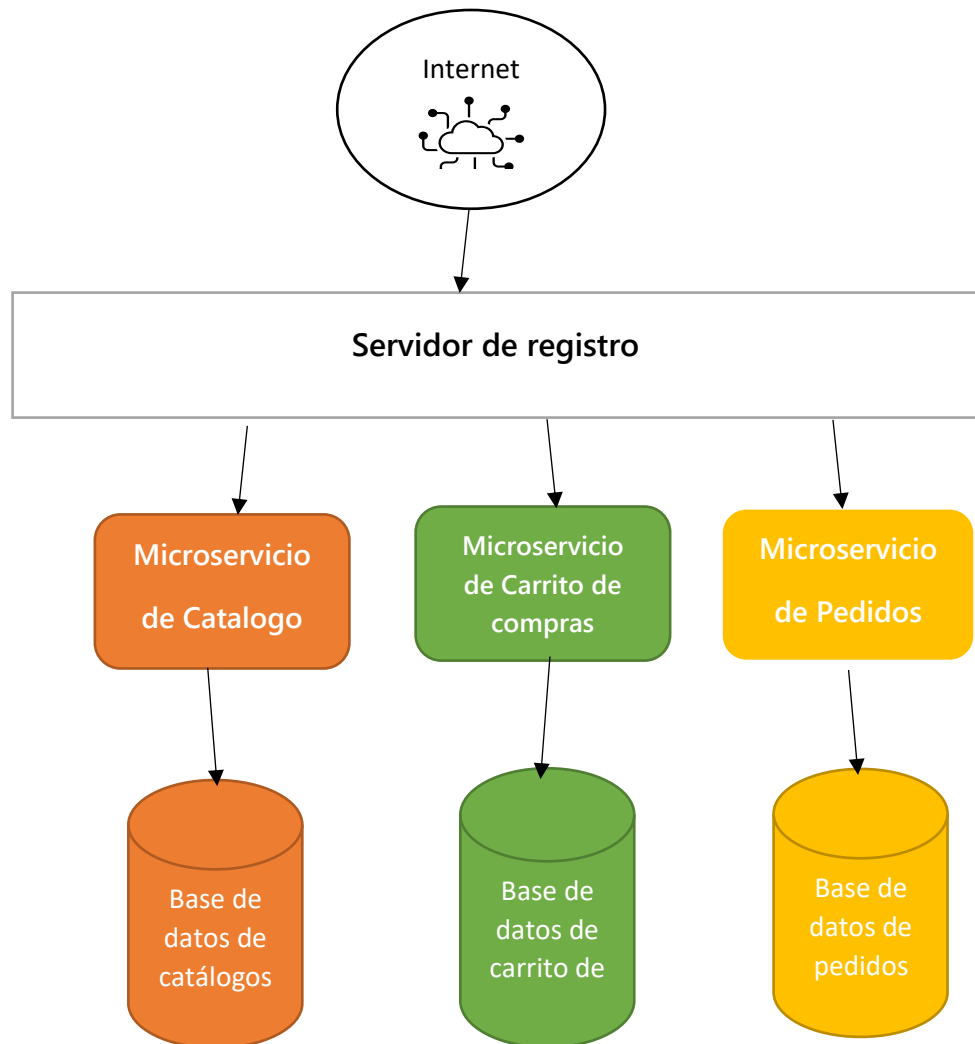


- Para el desarrollo de estos microservicios lo está realizando el equipo.



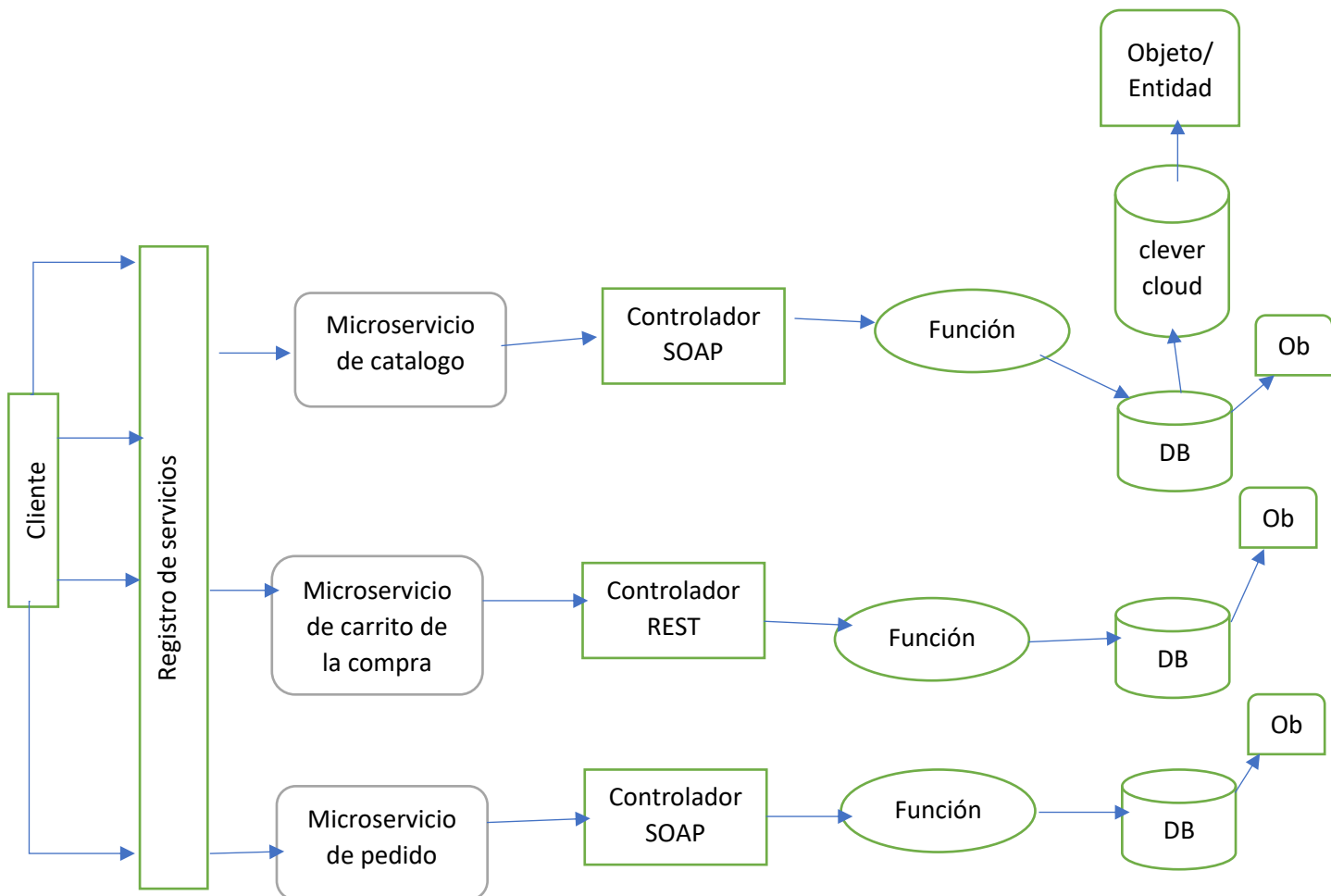
Diagrama de despliegue donde se identifican los servicios

En el diagrama a continuación se muestra una visión general de la arquitectura con la que se compondrá nuestro aplicativo.



Cada microservicio contará con su base de datos y estará alojado en un servidor de registro para posteriormente ser consumida por clientes.

En el siguiente diagrama se muestra en más detalle el diseño de los distintos microservicios planteados. Se puede observar que cada microservicio constara de un controlador que se encargara de recibir las peticiones sobre cada uno de los recursos llamando al servicio encargado según la acción solicitada. El servicio hará uso de una base de datos propia del microservicio para obtener o aplicar los datos de la solicitud.



Documentación del API SOAP y REST

Microservicios y funciones del sistema

Microservicio	Funcionalidades
Catálogo	Listado de productos
	Inclusión de un producto en el catalogo
	Actualización de un producto en el catalogo
	Eliminación de un producto en el catalogo
	Listado de categorías disponibles
	Inclusión de una categoría
	Listado de productos por categoría
Carrito	Incluir productos al carrito
	Eliminación de contenido del carrito
	Listado del contenido del carrito
Pedidos	Incluir un pedido
	Eliminar un pedido
	Listar un pedido

Métodos para implementar

Microservicio de catalogo

Incluye funcionalidades relacionada con la gestión de los productos y las categorías a las que van asociados permitiendo operaciones create, read, update y delete (CRUD).

Este microservicio estará implementado con la arquitectura SOAP.

<<Interface>>

Servicio Catalogo

POST + nuevoProducto (productold:int, nombre: string, descripción: string, precio: double, categoriald: int): Producto

GET + listaProductos (): List <Producto>

PUT + editarProducto ((productold:int, nombre: string, descripción: string, precio: double): Producto

DELETE + eliminarProducto (productold:int): Producto

POST + nuevaCategoria (categoriald: int, nombre: string) : Categoría

GET + listaProductosCategoria (productold: int, categoriald: int) : Producto, categoría

GET + listarCategorias (categoriald:int, nombre: string): categoría

Función	Descripción
Añadir producto	Se podrá incluir un nuevo producto en el sistema incluyendo nombre, descripción, precio y categoría a la que pertenece
Consultar productos	El microservicio deberá ser capaz de proporcionar una lista con todos los productos disponibles
Modificación producto	Se podrán cambiar los datos de cualquier producto
Borrado de producto	Se podrá borrar cualquier producto por medio de su Id
Añadir categoría	Se podrá añadir una nueva categoría
Consultar productos por categoría	El microservicio debe ser capaz de devolver los productos de una categoría
Listar categorías	El microservicio deberá ser capaz de devolver las categorías existentes

Microservicio Carrito

Este microservicio estará dedicado a gestionar exclusivamente el carrito de la compra de cada usuario. Estos pueden añadir los productos que quieran comprar al carrito, ver el contenido de este, eliminar su contenido o poder validarlo para crear un pedido.

Este microservicio será implementado con la arquitectura REST

Función	Descripción
Añadir producto al carrito	El microservicio añade el producto al carrito del usuario solicitante
Eliminar contenido del carrito	El usuario podrá vaciar el contenido de su carrito
Listar productos del carrito	Cada usuario podrá observar el contenido de su carrito
Confirmar compra carrito	El microservicio permitirá la confirmación de la compra del carrito, supondrá una generación automática de un pedido

<<Interface>>

Servicio Carrito

POST + añadirProductoCarrito (productold:int, cantidad:int): Orden, Producto

GET + verCarrito (): List <Producto>

DELETE + quitarProductoCarrito ()

POST + confirmarOrden ()

Microservicio Pedido

Este microservicio estará dedicado a gestionar exclusivamente los pedido de la compra de cada usuario. Estos puedes agregar pedidos, borrar pedidos y listar pedidos.

Este microservicio estará implementado con la arquitectura REST

Función	Descripción
Incluir un pedido	El usuario incluye productos en su compra.
Eliminar un pedido	El usuario puede eliminar su pedido.
Listar un pedido	El usuario ve el listado de su pedido

XML schema XSD, WSDL

Se creará un archivo XML a partir de un esquema XML (XSD) validado con el convertidor en línea <https://www.liquid-technologies.com/online-xsd-to-xml-convert>

Pedido

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="https://tis.uv.mx/pedido"
  xmlns:tns="https://tis.uv.mx/pedido"
  elementFormDefault="qualified">
  <xs:element name="agregarPedidoRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="pedidoId" type="xs:int" />
        <xs:element name="usuarioId" type="xs:int" />
        <xs:element name="direccion" type="xs:string" />
        <xs:element name="fecha" type="xs:string" />
        <xs:element name="total" type="xs:double" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="agregarPedidoResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="respuesta" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="listarPedidoRequest" />
  <xs:element name="listarPedidoResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="pedido" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
```

```
</xs:sequence>
    <xs:element name="pedido" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="pedidoId" type="xs:int" />
                <xs:element name="usuarioId" type="xs:int" />
                <xs:element name="direccion" type="xs:string" />
                <xs:element name="fecha" type="xs:string" />
                <xs:element name="total" type="xs:double" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="borrarPedidoRequest">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="pedidoId" type="xs:int" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="borrarPedidoResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="respuesta" type="xs:boolean" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

Catalogo

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetNamespace="https://tis.uv.mx/catalogo" xmlns:tns="https://tis.uv.mx/catalogo" elementFormDefault="qualified">
  <xs:element name="agregarCategoriaRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="categoriaId" type="xs:int" />
        <xs:element name="nombre" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="agregarCategoriaResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="respuesta" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="listarCategoriasRequest" />
  <xs:element name="listarCategoriasResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="categorias" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="categoriaId" type="xs:int" />
              <xs:element name="nombre" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```
<xs:element name="agregarProductoRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="productoId" type="xs:int" />
      <xs:element name="nombre" type="xs:string" />
      <xs:element name="descripcion" type="xs:string" />
      <xs:element name="precio" type="xs:double" />
      <xs:element name="categoriaId" type="xs:int" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="agregarProductoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="respuesta" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="buscarProductosRequest" />
<xs:element name="buscarProductosResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="productos" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="productoId" type="xs:int" />
            <xs:element name="categoria" type="xs:string" />
            <xs:element name="nombre" type="xs:string" />
            <xs:element name="descripcion" type="xs:string" />
            <xs:element name="precio" type="xs:double" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="buscarProductosCategoriaRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="categoriaId" type="xs:int" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="buscarProductosCategoriaResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="productosCategoria" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="categoria" type="xs:string" />
            <xs:element name="productoId" type="xs:int" />
            <xs:element name="nombre" type="xs:string" />
            <xs:element name="descripcion" type="xs:string" />
            <xs:element name="precio" type="xs:double" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```


Endpoints

El servicio de catalogo se encarga de mostrar los productos disponibles y listarlos según la petición recibida, asimismo permite gestionar los distintos productos y categorías, pudiendo añadirlos, editarlos o eliminarlos. Para este microservicio se tienen los siguientes endpoints para las solicitudes:

- AgregarCategoriaRequest
- listarCategoriRequest
- buscarProductoCategoriaRequest
- agregarProductoRequest
- buscarProductoRequest
- modificarProductoRequest
- borrarProductoRequest

```
@Endpoint
public class CatalogoEndPoint {

    @Autowired
    Icategoria ic;
    @Autowired
    Iproducto ip;

    @PayloadRoot(localPart = "agregarCategoriaRequest", namespace = "https://tis.uv.mx/catalogo")
    @ResponsePayload
    public AgregarCategoriaResponse agregarCategoria(@RequestPayload AgregarCategoriaRequest petition){
        AgregarCategoriaResponse respuesta = new AgregarCategoriaResponse();

        Categoria categoria = new Categoria();
        categoria.setId(petition.getCategoriaId());
        categoria.setNombre(petition.getNombre());
        ic.save(categoria);
        respuesta.setRespuesta("categoria agregada correctamente");

        return respuesta;
    }
}
```

```

@PayloadRoot(localPart = "listarCategoriasRequest", namespace = "https://tis.uv.mx/catalogo")
@ResponsePayload
public ListarCategoriasResponse listarCategorias(){
    ListarCategoriasResponse res = new ListarCategoriasResponse();
    Iterable<Categoria> lista = ic.findAll();

    for(Categoria o : lista){
        ListarCategoriasResponse.Categorias c = new ListarCategoriasResponse.Categorias();
        c.setCategoriaId(o.getId());
        c.setNombre(o.getNombre());
        res.getCategorias().add(c);
    }

    return res;
}

@PayloadRoot(localPart = "agregarProductoRequest", namespace = "https://tis.uv.mx/catalogo")
@ResponsePayload
public AgregarProductoResponse agregarProducto(@RequestPayload AgregarProductoRequest petition){
    AgregarProductoResponse res = new AgregarProductoResponse();

    Producto producto = new Producto();
    producto.setId(petition.getProductoId());
    producto.setNombre(petition.getNombre());
    producto.setDescripcion(petition.getDescripcion());
    producto.setPrecio(petition.getPrecio());
    producto.setCategoriaId(petition.getCategoriaId());
    ip.save(producto);
    res.setRespuesta("producto agregado");
}

```

```

@PayloadRoot(localPart = "buscarProductosCategoriaRequest", namespace = "https://tis.uv.mx/catalogo")
@ResponsePayload
public BuscarProductosCategoriaResponse buscarProductosCategoria(@RequestPayload BuscarProductosCategoriaRequest petition){
    BuscarProductosCategoriaResponse res = new BuscarProductosCategoriaResponse();

    Iterable<Producto> lc = ip.findAll();
    for(Producto o : lc){
        BuscarProductosCategoriaResponse.ProductosCategoria proCat = new BuscarProductosCategoriaResponse.ProductosCategoria();
        Optional<Categoria> opt = ic.findById(petition.getCategoriaId());
        if(o.getCategoriaId() == opt.get().getId()){
            proCat.setCategoria(opt.get().getNombre());
            proCat.setProductoId(o.getId());
            proCat.setNombre(o.getNombre());
            proCat.setDescripcion(o.getDescripcion());
            proCat.setPrecio(o.getPrecio());
            res.getProductosCategoria().add(proCat);
        }
    }
    return res;
}
}

```

Servicio de pedido se encarga de realizar las compras que realiza el usuario

- Catalogo-microservicios-catalogo/categorías
- Catalogo-microservicios-catalogo/productos
- Catalogo-microservicios-catalogo/producto/categoriaId

```
public class PedidoEndPoint{
    @Autowired
    Ipedido ipedido;

    @PayloadRoot(localPart = "agregarPedidoRequest", namespace = "https://tis.uv.mx/pedido")
    @ResponsePayload
    public AgregarPedidoResponse agregarPedido(@RequestPayload AgregarPedidoRequest petition){
        AgregarPedidoResponse respuesta = new AgregarPedidoResponse();

        Pedido pedido = new Pedido();
        pedido.setPedidoId(petition.getPedidoId());
        pedido.setUsuarioId(petition.getUsuarioId());
        pedido.setDireccion(petition.getDireccion());
        pedido.setFecha(petition.getFecha());
        pedido.setTotal(petition.getTotal());
        ipedido.save(pedido);
        respuesta.setRespuesta(pedido.toString());

        return respuesta;
    }

    @PayloadRoot(localPart = "listarPedidoRequest", namespace = "https://tis.uv.mx/pedido")
    @ResponsePayload
    public ListarPedidoResponse listarPedido(){
        ListarPedidoResponse res = new ListarPedidoResponse();
        Iterable<Pedido> lista = ipedido.findAll();

        for(Pedido o : lista){
            ListarPedidoResponse.Pedido p = new ListarPedidoResponse.Pedido();
```

```

@PayloadRoot(localPart = "listarPedidoRequest", namespace = "https://tis.uv.mx/pedido")
@ResponsePayload
public ListarPedidoResponse listarPedido(){
    ListarPedidoResponse res = new ListarPedidoResponse();
    Iterable<Pedido> lista = ipedido.findAll();

    for(Pedido o : lista){
        ListarPedidoResponse.Pedido p = new ListarPedidoResponse.Pedido();
        p.setPedidoId(o.getPedidoId());
        p.setUsuarioId(o.getUsuarioId());
        p.setDireccion(o.getDireccion());
        p.setFecha(o.getFecha());
        p.setTotal(o.getTotal());
        res.getPedido().add(p);
    }

    return res;
}

@PayloadRoot(localPart = "borrarPedidoRequest", namespace = "https://tis.uv.mx/pedido")
@ResponsePayload
public BorrارPedidoResponse borrarPedido(@RequestPayload BorrارPedidoRequest peticion){
    BorrارPedidoResponse respuesta = new BorrارPedidoResponse();

    ipedido.deleteById(peticion.getPedidoId());
    respuesta.setRespuesta(true);

    return respuesta;
}

```

```
<xs:element name="modificarProductoRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="productoId" type="xs:int" />
      <xs:element name="nombre" type="xs:string" />
      <xs:element name="descripcion" type="xs:string" />
      <xs:element name="precio" type="xs:double" />
      <xs:element name="categoriaId" type="xs:int" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="modificarProductoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="respuesta" type="xs:boolean" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="borrarProductoRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="productoId" type="xs:int" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="borrarProductoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="respuesta" type="xs:boolean" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Parámetros de recepción

Pedido parámetros Request:

- agregarPedido
- borrarPedido

Catalogo parámetros Request:

- agregarCategoria
- agregarProducto
- buscarProductosCategoria
- modificarProducto
- borrarProducto

Carrito endpoints

POST /api/añadir

GET /api/carrito

Delete /api/carrito/producto

Parámetros devueltos

Pedido parámetros Response:

- agregarPedido
- borrarPedido
- listarPedido

Catalogo parámetros Response:

- listarCategorias
- buscarProductos
- buscarProductosCategoria
- agregarCategoria
- agregarProducto
- borrarProducto

Forma de ejecución de los contenedores

Dockerfile microservicio catalogo

```
from rrojano/jdk8
workdir /app
#CMD ["/app/script.sh"]
add app/Catalogo-0.0.1-SNAPSHOT.jar /app/Catalogo-0.0.1-SNAPSHOT.jar
#add script.sh /app/script.sh
#run chmod 755 /app/script.sh
CMD java -jar -Dserver.port=$PORT Catalogo-0.0.1-SNAPSHOT.jar
```

Script ejecutar.sh de microservicio catalogo

```
#!/bin/bash
#!/bin/sh
/usr/bin/java -jar -Dserver.port=$PORT Catalogo-0.0.1-SNAPSHOT.jar
```

Dockerfile microservicio carrito

```
from rrojano/jdk8
workdir /app
add app/carrito-0.0.1-SNAPSHOT.jar /app/carrito-0.0.1-SNAPSHOT.jar
CMD java -jar -Dserver.port=$PORT carrito-0.0.1-SNAPSHOT.jar
```

Dockerfile microservicio Pedido

```
from rrojano/jdk8
workdir /app
#expose 8080
#cmd ["/app/script.sh"]
add microservicio-pedido/app/tis-0.0.1-SNAPSHOT.jar /app/tis-0.0.1-SNAPSHOT.jar
#add microservicio-pedido/script.sh /app/script.sh
#run chmod 755 /app/script.sh
cmd java -jar -Dserver.port=$PORT tis-0.0.1-SNAPSHOT.jar
```

Proyecto publicado en GitHub

<https://github.com/ManiiySosa/proyecto78935equipo7>

Despliegue en heroku

Microservicio-catalogo

<https://microservicio-catalogo-01.herokuapp.com/api/catalogo.wSDL>

Microservicio-carrito

<https://microservicio-carrito.herokuapp.com/api/>

Microservicio-pedido

<https://microserviciops.herokuapp.com/ws/pedido.wSDL>

Pruebas

Microservicio-pedido

Agregar pedido



```
POST https://microserviciops.herokuapp.com/443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    ns1:agregarPedidoResponse xmlns:ns1="https://t1s.uv.mx/pedido"
      <ns2:respuestaPedido [ pedido= 2, usuario= 2, direccion= calle1deseptiembre, fecha=09-06-2022, total=60.0 ] />
    </ns2:respuestaPedido>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```


Listar pedido

```
POST https://microserviciops.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:listarPedidoResponse xmlns:ns2="https://tis.uv.mx/pedido">
      <ns2:pedido>
        <ns2:pedidoId>3</ns2:pedidoId>
        <ns2:usuarioId>1</ns2:usuarioId>
        <ns2:direccion>bulvar san cristobal</ns2:direccion>
        <ns2:fecha>04-06-2022</ns2:fecha>
        <ns2:total>0.0</ns2:total>
      </ns2:pedido>
      <ns2:pedido>
        <ns2:pedidoId>5</ns2:pedidoId>
        <ns2:usuarioId>5</ns2:usuarioId>
        <ns2:direccion>hola</ns2:direccion>
        <ns2:fecha>06-06-2022</ns2:fecha>
        <ns2:total>20.0</ns2:total>
      </ns2:pedido>
      <ns2:pedido>
        <ns2:pedidoId>7</ns2:pedidoId>
        <ns2:usuarioId>1</ns2:usuarioId>
        <ns2:direccion>calle 13 de septiembre</ns2:direccion>
        <ns2:fecha>08-06-2022</ns2:fecha>
        <ns2:total>50.0</ns2:total>
      </ns2:pedido>
      <ns2:pedido>
        <ns2:pedidoId>8</ns2:pedidoId>
        <ns2:usuarioId>2</ns2:usuarioId>
        <ns2:direccion>calle 13 de septiembre</ns2:direccion>
        <ns2:fecha>09-06-2022</ns2:fecha>
        <ns2:total>60.0</ns2:total>
      </ns2:pedido>
    </ns2:listarPedidoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Borrar pedido

```
POST https://microserviciops.herokuapp.com:443/ws
:Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/"
  <Body>
    <borrarPedidoRequest xmlns="https://tis.uv.mx/pedido">
      <pedidoId>[int]</pedidoId>
    </borrarPedidoRequest>
  </Body>
:/Envelope
```

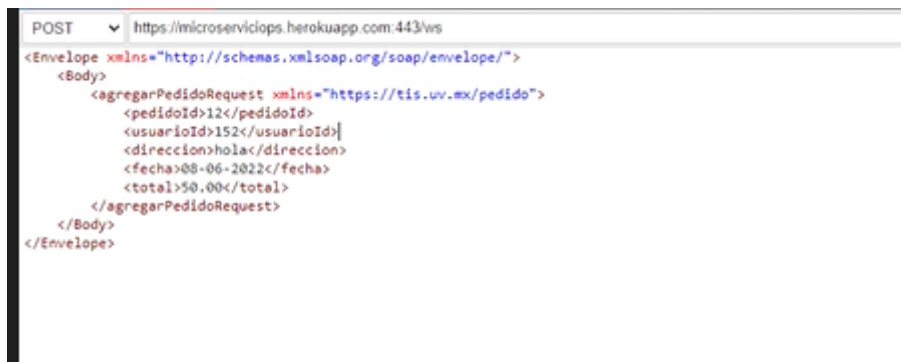
Microservicio-catalogo

Agregar categoría



```
extension://oebpmncolmhiapingjaagmapififiakb/editor.html#wsdl=http
POST https://microservicio-catalogo-01.herokuapp.com:443/api
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:agregarCategoriaResponse xmlns:ns2="https://tis.uv.mx/catalogo">
      <ns2:respuesta>categoria agregada correctamente</ns2:respuesta>
    </ns2:agregarCategoriaResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

En este apartado se agrega un pedido



```
POST https://microserviciops.herokuapp.com:443/ws
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <agregarPedidoRequest xmlns="https://tis.uv.mx/pedido">
      <pedidoId>12</pedidoId>
      <usuarioId>152</usuarioId>
      <direccion>hola</direccion>
      <fecha>08-06-2022</fecha>
      <total>50.00</total>
    </agregarPedidoRequest>
  </Body>
</Envelope>
```

Se visualiza el pedido que anteriormente se agregó con éxito



```
extension://oebpmncolmhiapingjaagmapififiakb/editor.html#wsdl=http
POST https://microservicio-catalogo-01.herokuapp.com:443/api
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:agregarCategoriaResponse xmlns:ns2="https://tis.uv.mx/catalogo">
      <ns2:respuesta>categoria agregada correctamente</ns2:respuesta>
    </ns2:agregarCategoriaResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Aquí se muestra el listado

```
POST https://microserviciops.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2: listarPedidoResponse xmlns:ns2="https://tis.uv.mx/pedido">
      <ns2:pedido>
        <ns2:pedidoId>3</ns2:pedidoId>
        <ns2:usuarioId>1</ns2:usuarioId>
        <ns2:direccion>buirvarsancristo</ns2:direccion>
        <ns2:fecha>04-06-2022</ns2:fecha>
        <ns2:total>0.0</ns2:total>
      </ns2:pedido>
      <ns2:pedido>
        <ns2:pedidoId>5</ns2:pedidoId>
        <ns2:usuarioId>5</ns2:usuarioId>
        <ns2:direccion>hola</ns2:direccion>
        <ns2:fecha>06-06-2022</ns2:fecha>
        <ns2:total>20.0</ns2:total>
      </ns2:pedido>
      <ns2:pedido>
        <ns2:pedidoId>7</ns2:pedidoId>
        <ns2:usuarioId>1</ns2:usuarioId>
        <ns2:direccion>calle13deseptiembre</ns2:direccion>
        <ns2:fecha>08-06-2022</ns2:fecha>
        <ns2:total>50.0</ns2:total>
      </ns2:pedido>
      <ns2:pedido>
        <ns2:pedidoId>8</ns2:pedidoId>
        <ns2:usuarioId>2</ns2:usuarioId>
        <ns2:direccion>calle13deseptiembre</ns2:direccion>
        <ns2:fecha>09-06-2022</ns2:fecha>
        <ns2:total>60.0</ns2:total>
      </ns2:pedido>
    </ns2: listarPedidoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Buscar producto

```
POST https://microservicio-catalogo-01.herokuapp.com:443/api
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2: buscarProductosResponse xmlns:ns2="https://tis.uv.mx/catalogo">
      <ns2: productos>
        <ns2: productoId>1</ns2: productoId>
        <ns2: categoria>camisas</ns2: categoria>
        <ns2: nombre>camisa rayas</ns2: nombre>
        <ns2: descripcion>camisa de rayas color gris</ns2: descripcion>
        <ns2: precio>49.9</ns2: precio>
      </ns2: productos>
      <ns2: productos>
        <ns2: productoId>4</ns2: productoId>
        <ns2: categoria>sudaderas</ns2: categoria>
        <ns2: nombre>tomy</ns2: nombre>
        <ns2: descripcion>sudadera negra</ns2: descripcion>
        <ns2: precio>50.0</ns2: precio>
      </ns2: productos>
      <ns2: productos>
        <ns2: productoId>5</ns2: productoId>
        <ns2: categoria>camisas</ns2: categoria>
        <ns2: nombre>camine rayas</ns2: nombre>
        <ns2: descripcion>camisa rayas color roja</ns2: descripcion>
        <ns2: precio>100.0</ns2: precio>
      </ns2: productos>
      <ns2: productos>
        <ns2: productoId>6</ns2: productoId>
        <ns2: categoria>Camisas manga corta</ns2: categoria>
        <ns2: nombre>camisa polo</ns2: nombre>
        <ns2: descripcion>camisa manga corta azul</ns2: descripcion>
        <ns2: precio>59.9</ns2: precio>
      </ns2: productos>
    </ns2: buscarProductosResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Buscar productos por categorías

extension://oebpmncolmhiapingjaagmapififiakb/editor.html#wsl=htps%3A%
POST https://microservicio-catalogo-01.herokuapp.com:443/api
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Header/>
 <SOAP-ENV:Body>
 <ns2:buscarProductosCategoriaResponse xmlns:ns2="https://tis.uv.mx/catalogo">
 <ns2:productosCategoria>
 <ns2:categoria>Camisas manga corta</ns2:categoria>
 <ns2:productoId>6</ns2:productoId>
 <ns2:nombre>camisa polo </ns2:nombre>
 <ns2:descripcion>camisa manga corta azul</ns2:descripcion>
 <ns2:precio>59.9</ns2:precio>
 </ns2:productosCategoria>
 </ns2:buscarProductosCategoriaResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Microservicio-carrito

The screenshot shows the Postman interface with a POST request to the endpoint `https://microservicio-carrito.herokuapp.com/api/añadir`. The request body is a JSON object: `{ "productoId": "camisa adidas", "cantidad": 2, "precio": 49.9 }`. The response is a JSON object: `{ "productoId": "camisa adidas", "cantidad": 2, "precio": 49.9, "total": 99.8, "id": 10 }`. Annotations point to the endpoint and the response body.

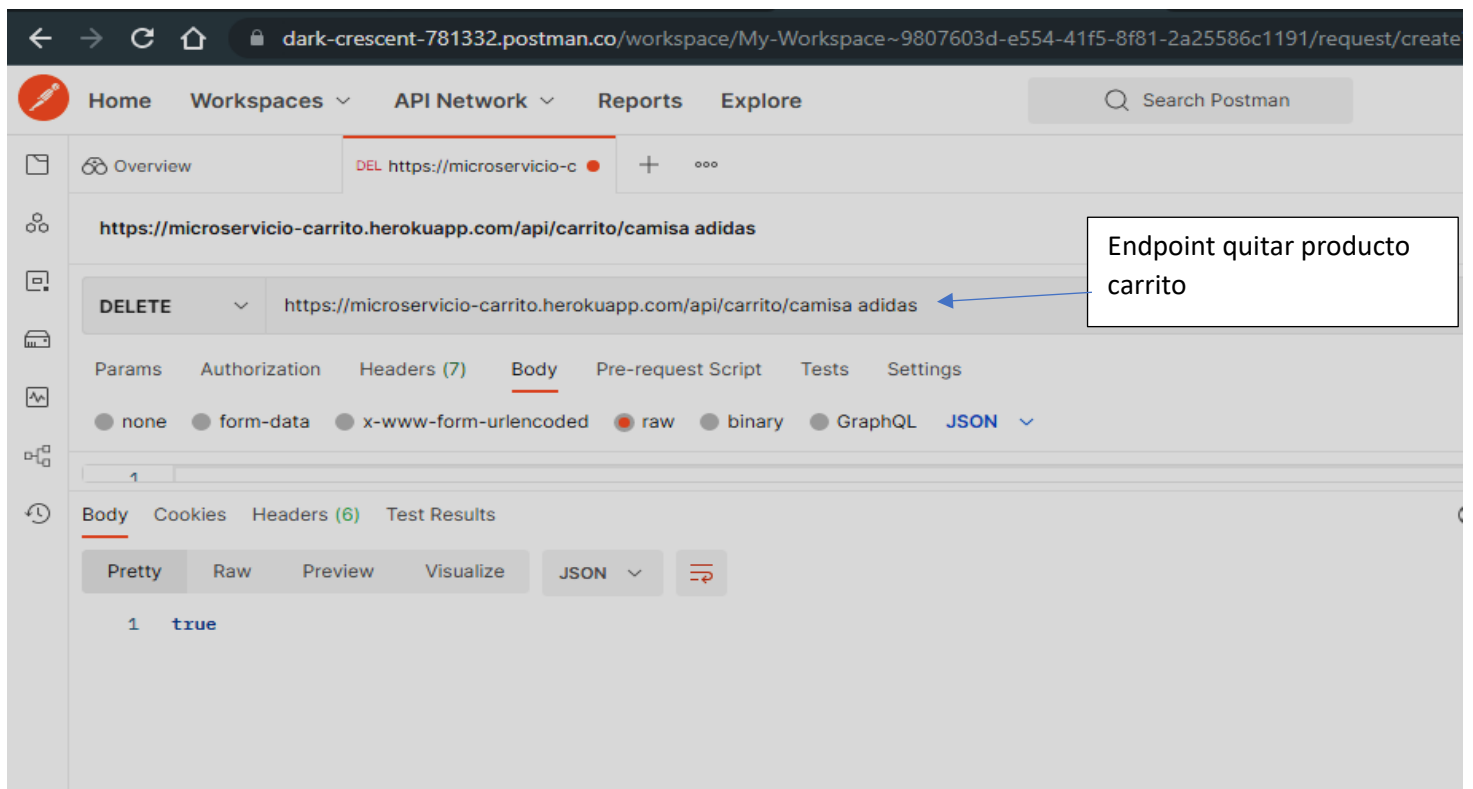
Endpoint añadir producto al carrito

Producto añadido al carrito

The screenshot shows the Postman interface with a GET request to the endpoint `https://microservicio-carrito.herokuapp.com/api/carrito`. The response is a JSON array of two objects: `[{ "productoId": "camisa adidas", "cantidad": 2, "precio": 49.9, "total": 99.8, "id": 10 }, { "productoId": "sudadera calvin", "cantidad": 1, "precio": 200.0, "total": 200.0, "id": 11 }]`. Annotations point to the endpoint and the response body.

Endpoint ver el carrito

Devuelve los productos en el carrito



Conclusiones

Tras la elaboración de este proyecto se ha logrado mejorar la forma de trabajo adquirida durante las prácticas y actividades realizadas en la experiencia educativa, fortaleciendo la capacidad de poder afrontar la implementación de un proyecto de microservicios ya sean SOAP o REST con diferentes tecnologías. Con lo anterior mencionado tratamos de crear unos microservicios los cuales nos fueran funcionales para la creación de tiendas virtuales sin la necesidad de tener que programar más de lo necesario. Y facilitar la creación de estos métodos, aparte de llevar experiencias nuevas las cuales nos ayudan a nuestra formación académica.