

Índice:

CEPA ALMACÉN	2
Web service:	2
Obtener datos:	2
Eliminar	2
Agregar	3
Actualizar	3
CEPA-ALMACÉN	4
Capturas del sistema:	5
Agregar	5
Editar	5
Detalles	6
Consideraciones	6
Web Service:	6
Cliente:	7
Conceptos técnicos:	7
Web Services:	7
Cliente:	8

CEPA ALMACÉN

CEPA ALMACÉN es un proyecto que consta de dos partes principales: **Web Services** y **Cliente**. Su propósito es gestionar productos en una base de datos a través de una interfaz web intuitiva.

Web service:

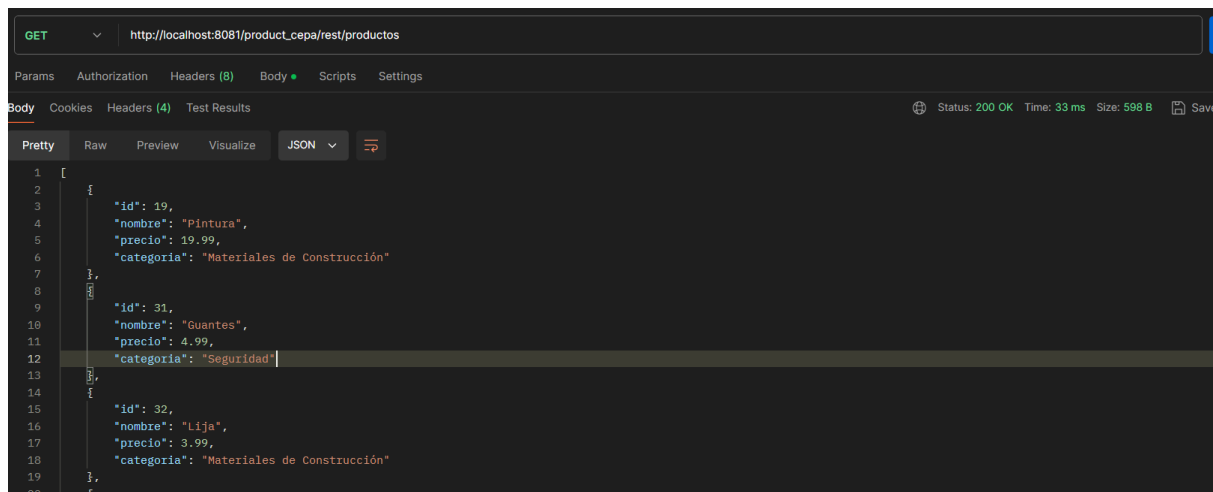
El Web Service de CEPA ALMACÉN está diseñado utilizando REST, que permite la comunicación a través de HTTP. Está construido en Java utilizando JAX-RS (Java API for RESTful Web Services) para proporcionar una API que interactúa con la base de datos mediante el patrón DAO (Data Access Object). La API soporta operaciones CRUD (Crear, Leer, Actualizar y Eliminar) para manejar productos.

Obtener datos:

Método: GET

Descripción: Recupera una lista de todos los productos disponibles

http://localhost:8081/product_cepa/rest/productos



```
GET http://localhost:8081/product_cepa/rest/productos

Status: 200 OK Time: 33 ms Size: 598 B

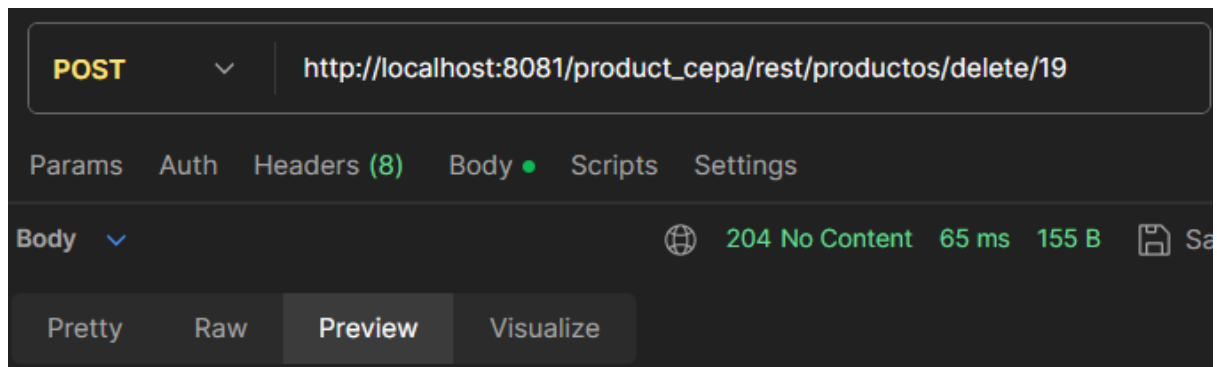
[{"id": 19, "nombre": "Pintura", "precio": 19.99, "categoria": "Materiales de Construcci\u00f3n"}, {"id": 31, "nombre": "Guantes", "precio": 4.99, "categoria": "Seguridad"}, {"id": 32, "nombre": "Lija", "precio": 3.99, "categoria": "Materiales de Construcci\u00f3n"}]
```

Eliminar

Método: Post

Descripción: Elimina un producto específico basado en su ID.

http://localhost:8081/product_cepa/rest/productos/delete/{id}

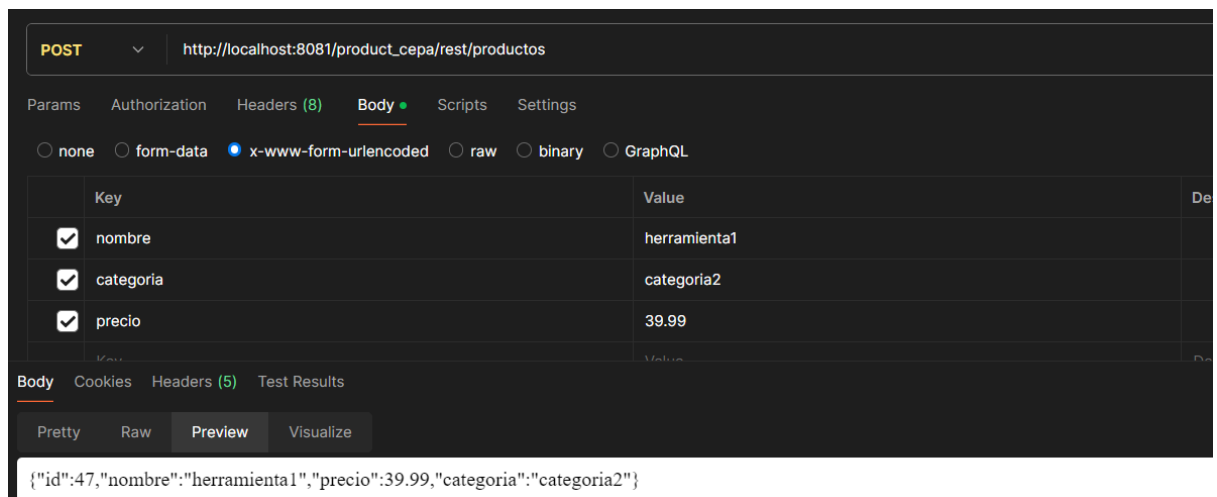


Agregar

Método: POST

Descripción: Añade un nuevo producto a la base de datos.

http://localhost:8081/product_cepa/rest/productos

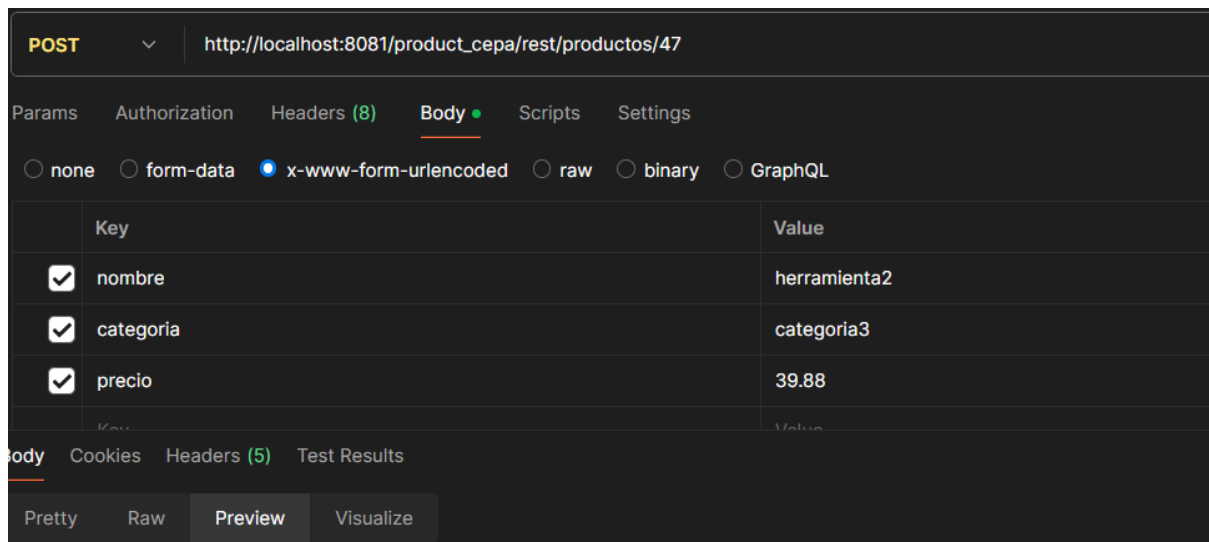


Actualizar

Método: Post

Descripción: Actualiza la información de un producto existente basado en su ID.

http://localhost:8081/product_cepa/rest/productos/{id}

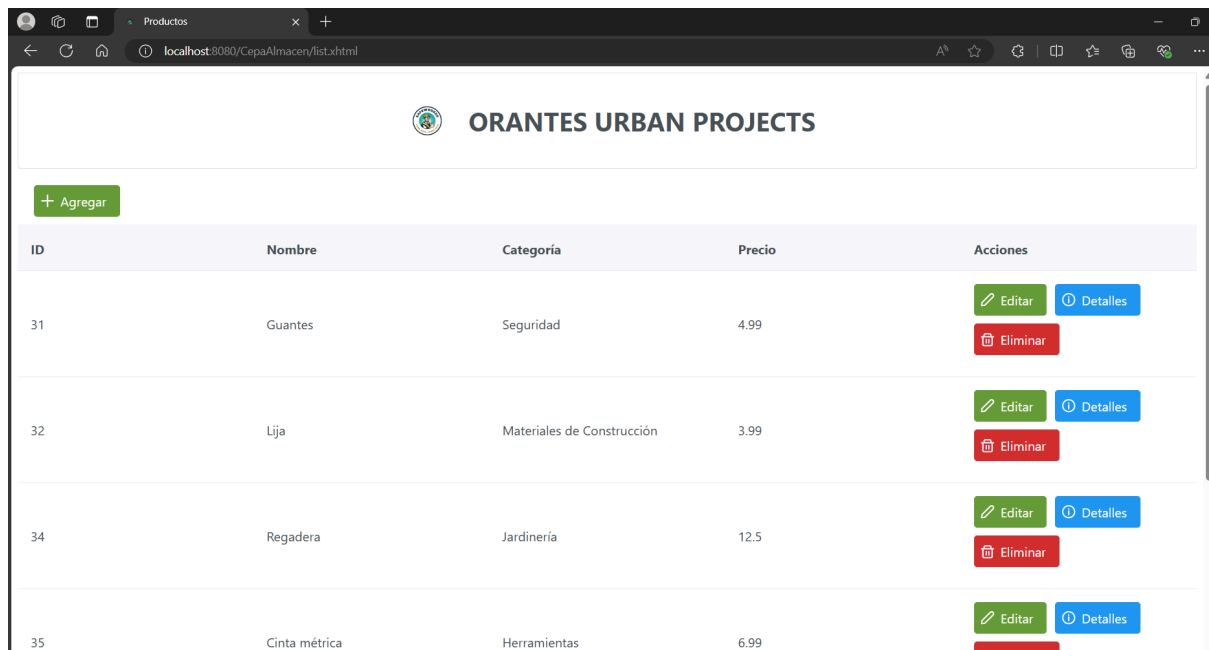


```
{ "id":47,"nombre":"herramienta2","precio":39.88,"categoria":"categoria3" }
```

CEPA-ALMACEN

Permite diversas operaciones CRUD para un conjunto de productos (herramientas) haciendo uso del web services rest.

En caso de usar el puerto 8080: <http://localhost:8080/CepaAlmacen/list.xhtml>



Capturas del sistema:

Agregar

Permite agregar un nuevo producto.

Nombre: *	<input type="text"/>
Categoría: *	<input type="text"/>
Precio: *	<input type="text" value="0.00"/>

✓ Guardar

✕ Cancelar

Editar

Permite editar un producto.

Editar Producto

ID:	<input type="text" value="31"/>
Nombre: *	<input type="text" value="Guantes"/>
Categoría: *	<input type="text" value="Seguridad"/>
Precio: *	<input type="text" value="4.99"/>

+ Guardar Cambios

Cancelar

Detalles

Permite ver los detalles de un producto.

Detalles del Producto	
ID:	31
Nombre:	Guantes
Categoría:	Seguridad
Precio:	\$4.99

Volver

Consideraciones

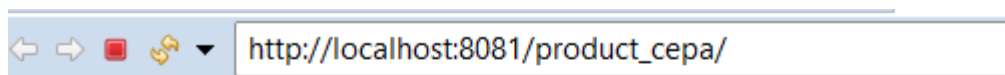
Web Service:

- **Entorno de Desarrollo:**
 - **IDE Utilizado:** Eclipse
 - **Servidor de Aplicaciones:** Apache Tomcat
- **Configuración Recomendada:**
 - **Puerto del Servidor:** Configure Tomcat para que utilice un puerto diferente al del cliente para evitar conflictos. Se recomienda utilizar el puerto **8081** para el Web Service.
- **Versión de Java:**
 - **JDK:** Utilice **JDK 17** para desarrollar el Web Service.
- **Base de Datos:**

- **Base de Datos:** Asegúrese de importar correctamente la base de datos denominada “**Productos**”. Esta base de datos es esencial para gestionar la información de los productos en el Web Service.
- **Funcionalidad del Web Service:**
 - El Web Service permite realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre productos utilizando una API RESTful.

Cliente:

- **Entorno de Desarrollo:**
 - **IDE Utilizado:** NetBeans
 - **Servidor de Aplicaciones:** Payara Server 6.2024.4
- **Versión de Java:**
 - **JDK:** Utilice **JDK 17** para desarrollar el cliente.
- **Configuración Recomendada:**
 - Asegúrese de que el cliente y el servidor de aplicaciones utilicen configuraciones y puertos diferentes para evitar conflictos de red.
- **Funcionalidad del Cliente:**
 - El cliente interactúa con el Web Service para gestionar los productos. Utiliza JavaServer Faces (JSF) y PrimeFaces para proporcionar una interfaz de usuario rica y dinámica.



Hello World!

Conceptos técnicos:

Web Services:

JAX-RS (Java API for RESTful Web Services):

- Anotaciones clave: `@Path`, `@GET`, `@POST`, `@Produces`, `@PathParam`, `@FormParam`.
- Propósito: Define rutas, métodos HTTP, y formato de respuesta para el servicio web RESTful.

JDBC (Java Database Connectivity):

- Clases: `Connection`, `Statement`, `PreparedStatement`, `ResultSet`.
- Facilita la conexión y la interacción con bases de datos SQL, incluyendo la ejecución de consultas y el manejo de resultados.

DAO (Data Access Object) Pattern:

- Clase `ProductoDAO`: Maneja operaciones CRUD (crear, leer, actualizar, eliminar) para la entidad `Producto`.

Cliente:

JavaServer Faces (JSF): Para la creación de interfaces web dinámicas.

PrimeFaces: Biblioteca de componentes UI para JSF, proporcionando una rica interfaz de usuario.

JAX-RS (Java API for RESTful Web Services): Para la comunicación con un servicio RESTful.