

Tarea 02
Arquitectura de Sistemas
ICF244

Profesor Juan Calderón Maureira

Carolina Contreras
Camila Molina

Noviembre, 2021

Aplicación que se integrará:

Según lo propuesto en el TO BE de la Fase D de nuestro trabajo anterior (relacionado con Archimate), el sistema de ERP se encargará (entre otras cosas) de informar a los encargados de bodega, jefe de bodega y los proveedores sobre el stock de bodegas, esto mediante las aplicaciones “orden de stock” y “control de inventario” que dependerán de que se actualice correctamente el stock. En este trabajo nos centraremos en la aplicación de “control de inventario” ya que esta aplicación permitirá tener un control claro y un listado de todo lo que hay y no en bodega. Esta aplicación tendrá como métodos “actualizar stock”, “eliminar stock” y “crear stock”, este último en el caso que llegué un nuevo inventario que no existía antes. Para esto usaremos los servicios que provee Amazon Web Service.

Diagramas:

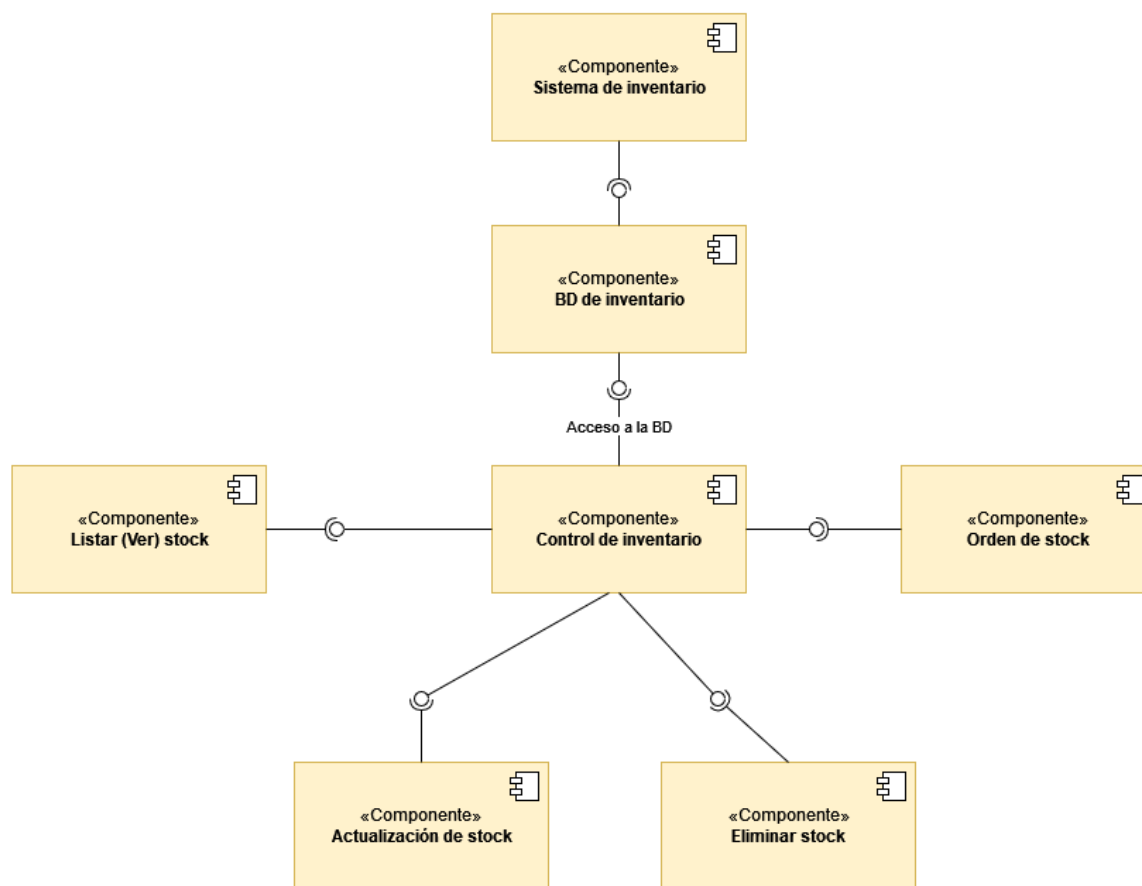


Diagrama 1: “Diagrama de componentes de inventario”

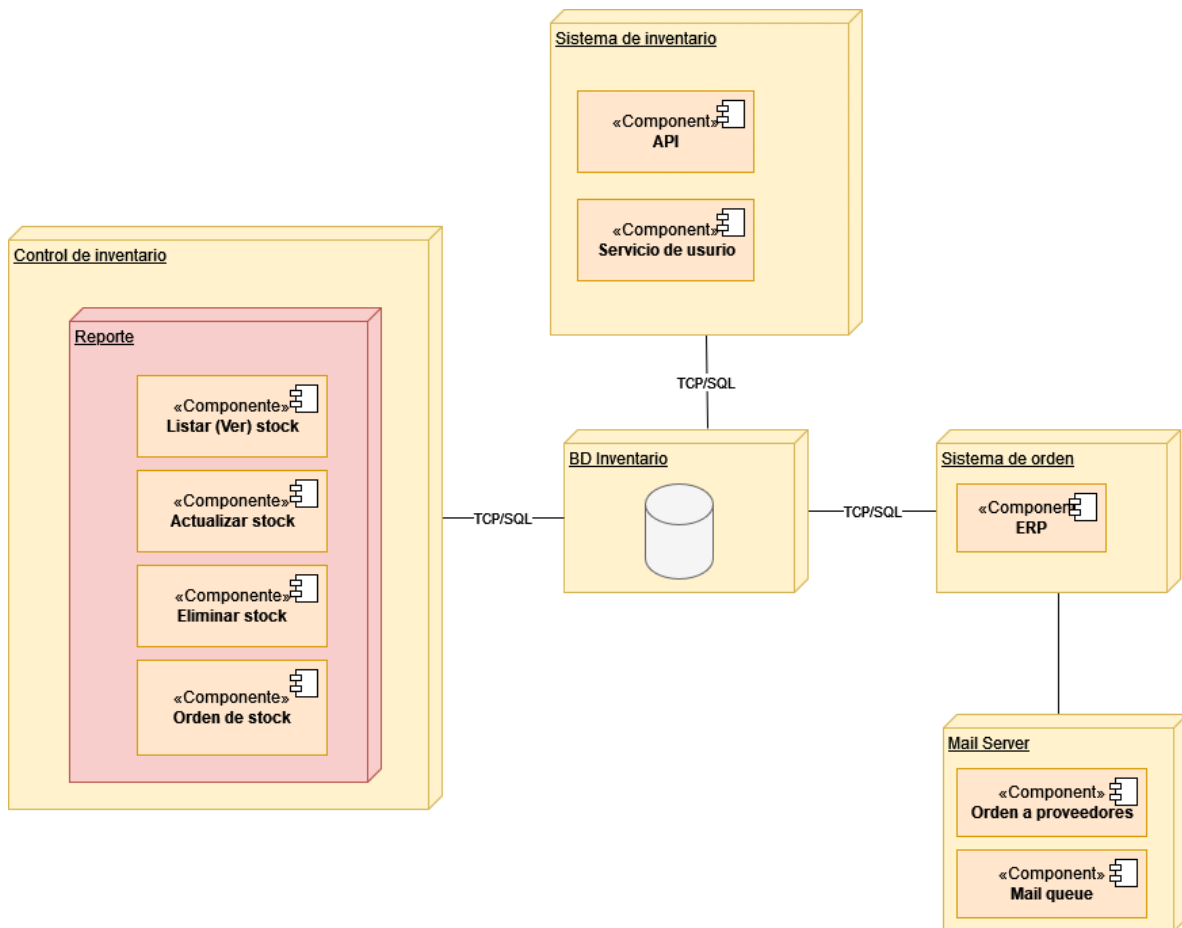


Diagrama 2: “Diagrama de deploy de inventario”

Funcionamiento:

super Stage Editor

Invoke URL: <https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super>

Settings | Logs/Tracing | Stage Variables | SDK Generation | Export | Deployment History | Documentation History | Canary

Cache Settings

Enable API cache ☐

Default Method Throttling

Choose the default throttling level for the methods in this stage. Each method in this stage will respect these rate and burst settings. Your current account level throttling rate is **10000** requests per second with a burst of **5000** requests. [Read more about API Gateway throttling](#)

Enable throttling ☒ ⓘ

Rate requests per second

Burst requests

Web Application Firewall (WAF) [Learn more.](#)

Al usar el link: <https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super>

Y usarlo en Postman con los distintos métodos se verán los cambios en DynamoDB.

The screenshot displays two Postman requests. The first is a POST request to `https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super/product` with a JSON body containing product details. The response shows a successful save operation. The second is a GET request to `https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super/products` which returns a list of products in a table format.

POST Request:

URL: `https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super/product`

Method: POST

Body (JSON):

```
1 {
2   "productId": "3",
3   "nombreProducto": "Neo QLED 85",
4   "precio": 5990990,
5   "marca": "Samsung",
6   "stock": 10
7 }
```

Response (JSON):

```
1 {
2   "Operation": "SAVE",
3   "Message": "SUCCESS",
4   "Item": {
5     "productId": "3",
6     "nombreProducto": "Neo QLED 85",
7     "precio": 5990990,
8     "marca": "Samsung",
9     "stock": 10
10  }
11 }
```

GET Request:

URL: `https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super/products`

Method: GET

Query Params:

KEY	VALUE	DESCRIPTION
Key	Value	Description

DELETE Request:

URL: `https://yfzjcs18k7.execute-api.us-east-2.amazonaws.com/super/product`

Method: DELETE

Body (JSON):

```
1 {
2   "productId": "3"
3 }
```

inventory

Expand to query or scan items.

View table details

Items returned (4)

↺

Actions ▼

Create item

<

1

>

⚙

✖

<input type="checkbox"/>	productId ▲	marca ▼	nombre... ▼	precio ▼	stock ▼
<input type="checkbox"/>	0	LG	TV	100000	50
<input type="checkbox"/>	1	SONY	TV	990990	100
<input type="checkbox"/>	2	Apple	Macbook Pr...	3990990	20
<input type="checkbox"/>	3	Samsung	Neo QLED 85	5990990	10

Items returned (4)

↺

Actions ▼

Create item

<

1

>

⚙

✖

<input type="checkbox"/>	productId ▲	marca ▼	nombre... ▼	precio ▼	stock ▼
<input type="checkbox"/>	0	LG	TV	100000	50
<input type="checkbox"/>	1	SONY	TV	990990	100
<input type="checkbox"/>	2	Apple	Macbook Pr...	3990990	20

Repositorio con video explicativo, los códigos y documento:

<https://github.com/CamiiMolina/Tarea2>