

CS2032 - Cloud Computing (Ciclo 2024-2)

Serverless

Semana 10 - Taller 1: Sistema de Seguridad

ELABORADO POR: GERALDO COLCHADO

Con apoyo de Asistentes de Cátedra y Laboratorio:

- Ana Accilio (ana.accilio@utec.edu.pe)
- Sofía García (sofia.garcia@utec.edu.pe)

Contenido

Sistema de Seguridad

1. **Objetivo del taller 1**
2. Conocimientos Previos
3. Ejercicio 1: Sistema de Seguridad
4. Ejercicio 2: Ejercicio Propuesto
5. Cierre

Serverless

Objetivo del Taller 1

- Implementar un Sistema de Seguridad Serverless para proteger el acceso a las apis

Contenido

Sistema de Seguridad

1. Objetivo del taller 1
2. **Arquitectura de Referencia**
3. Ejercicio 1: Sistema de Seguridad
4. Ejercicio 2: Ejercicio Propuesto
5. Cierre

Serverless

Arquitectura de Referencia

Reference architecture



RESTful microservices

1. **Customers** leverage your microservices by making HTTP API calls. Ideally, your consumers should have a tightly bound service contract to your API to achieve consistent expectations of service levels and change control.
2. **Amazon API Gateway** hosts RESTful HTTP requests and responses to customers. In this scenario, API Gateway provides **built-in authorization**, throttling, security, fault tolerance, request and response mapping, and performance optimizations.
3. **AWS Lambda** contains the **business logic** to process incoming API calls and use DynamoDB as a persistent storage.
4. **Amazon DynamoDB** **persistently stores microservices data and scales based on demand**. Since microservices are often designed to do one thing well, a schemaless NoSQL data store is regularly incorporated.

Contenido

Sistema de Seguridad

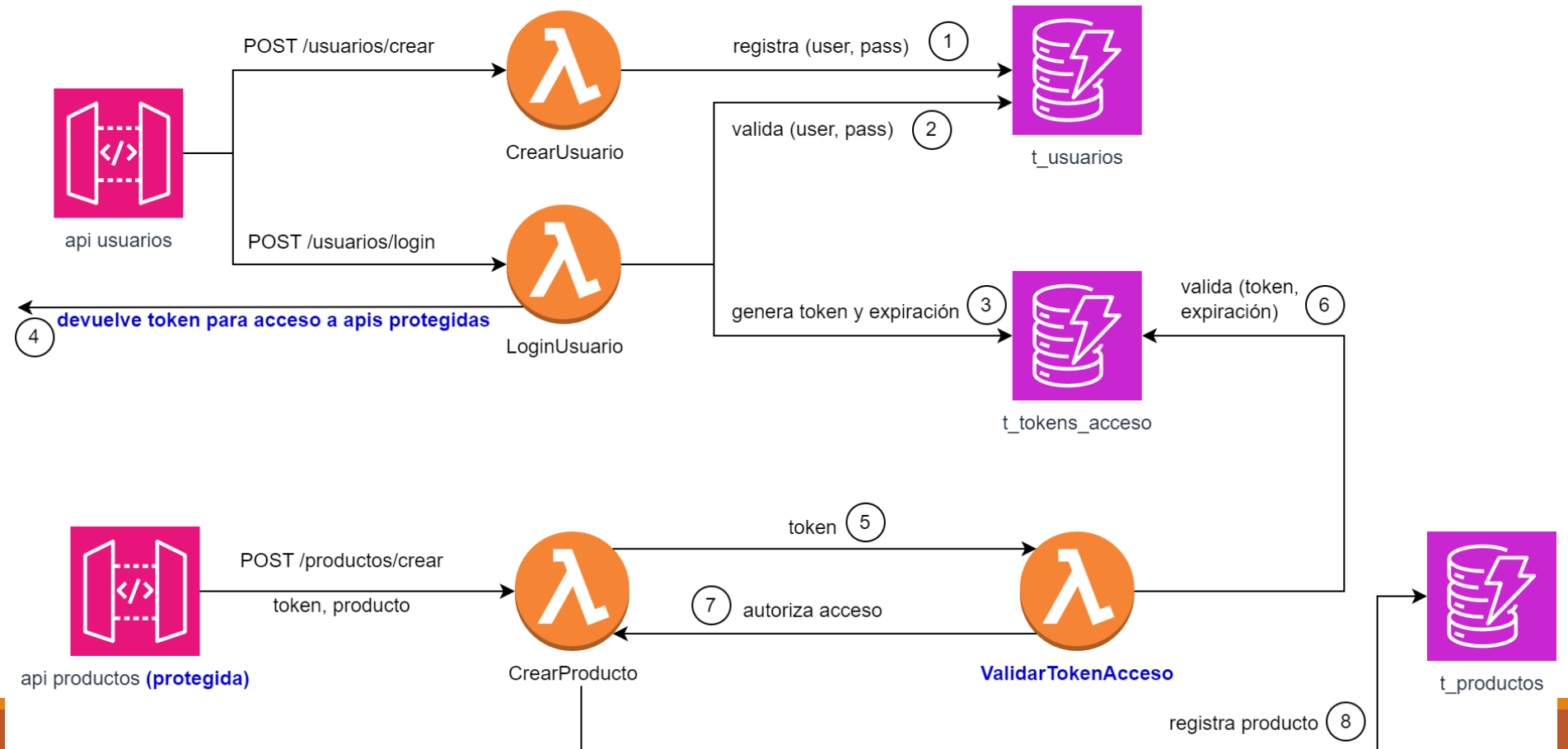
1. Objetivo del taller 1
2. Arquitectura de Referencia
3. **Ejercicio 1: Sistema de Seguridad**
4. Ejercicio 2: Ejercicio Propuesto
5. Cierre

Serverless

Ejercicio 1: Sistema de Seguridad

Implementar este Sistema de Seguridad para **proteger** la ejecución de **apis** con un **token** con **expiración**

Diagrama de Arquitectura de Solución de Sistema de Seguridad



Serverless

Ejercicio 1: Sistema de Seguridad

Paso 1: Crear tablas en DynamoDB: t_usuarios, t_tokens_acceso, t_productos

Nombre ▲	Estado	Clave de partición	Clave de ordenación
t_productos	✓ Activo	tenant_id (S)	producto_id (S)
t_tokens_acceso	✓ Activo	token (S)	-
t_usuarios	✓ Activo	user_id (S)	-

Serverless

Ejercicio 1: Sistema de Seguridad

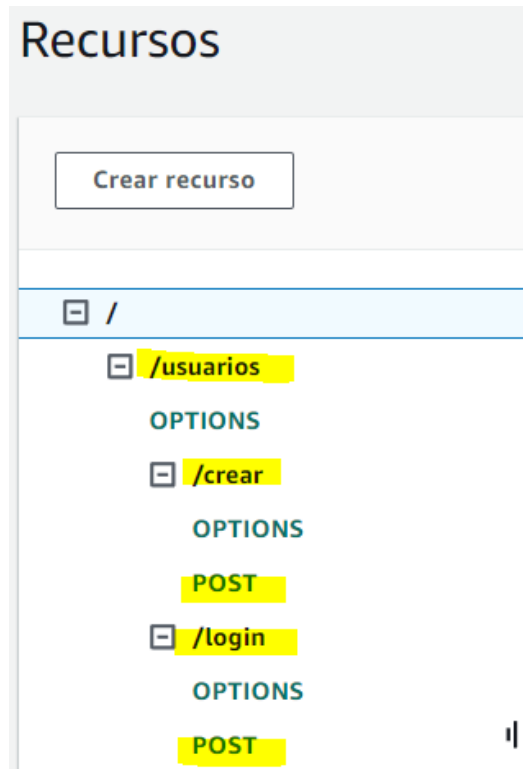
Paso 2: Crear lambdas: CrearUsuario, LoginUsuario, ValidarTokenAcceso, CrearProducto

Nombre de la función ▼	Descripción ▼	Tipo de paquete ▼	Tiempo de ejecución
LoginUsuario	-	Zip	Python 3.12
ValidarTokenAcceso	-	Zip	Python 3.12
CrearProducto	-	Zip	Python 3.12
CrearUsuario	-	Zip	Python 3.12

Serverless

Ejercicio 1: Sistema de Seguridad

Paso 3: Crear Api Gateway usuarios



Serverless

Ejercicio 1: Sistema de Seguridad

Paso 4: Probar api usuarios: Crear un usuario

user_id (Cadena)	password
gcolchado@utec.edu.pe	8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c92

t_usuarios

POST ▼ <https://x00hp4q99a.execute-api.us-west-2.amazonaws.com/prod/usuarios/crear>

Params Authorization Headers (7) **Body** ● Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1 {
2   "user_id": "gcolchado@utec.edu.pe",
3   "password": "123456"
4 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize **JSON** ▼ ≡

```
1 {
2   "statusCode": 200,
3   "body": {
4     "message": "User registered successfully",
5     "user_id": "gcolchado@utec.edu.pe"
6   }
7 }
```

Serverless

Ejercicio 1: Sistema de Seguridad

Paso 5: Probar api usuarios: Login usuario

POST ▼ https://x00hp4q99a.execute-api.us-west-2.amazonaws.com/prod/usuarios/login

Params Authorization Headers (7) **Body** ● Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1 {  
2   "user_id": "gcolchado@utec.edu.pe",  
3   "password": "123456"  
4 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize **JSON** ▼ ≡

```
1 {  
2   "statusCode": 200,  
3   "token": "ae975067-bdd4-4e39-89fe-272c8d499ed8"  
4 }
```

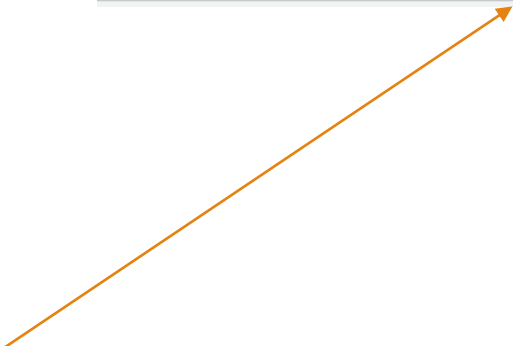
token (Cadena) ▼

expires

[ae975067-bdd4-4e39-89fe-272c8d499ed8](#)

2024-06-24 01:53:04

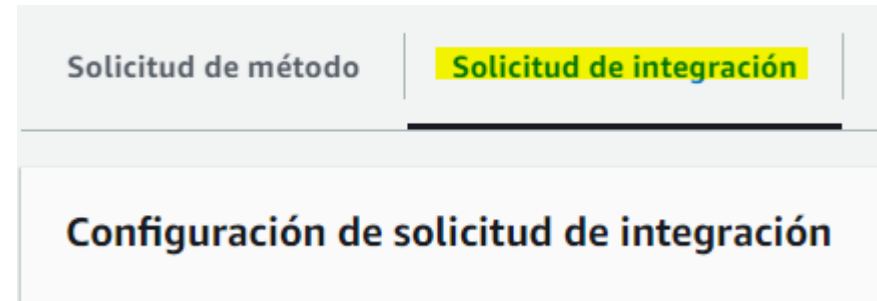
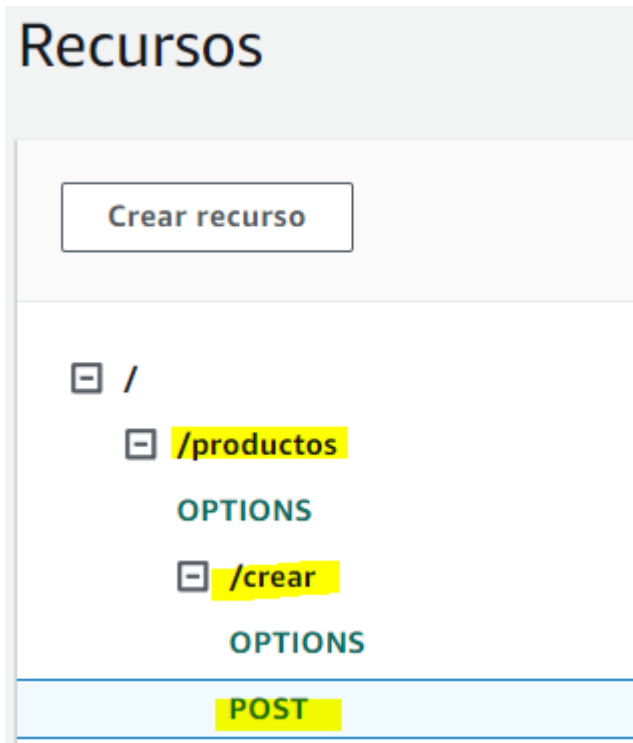
t_tokens_acceso



Serverless

Ejercicio 1: Sistema de Seguridad

Paso 6: Crear Api Gateway productos



▼ application/json

```
1  {
2    "method": "$context.httpMethod",
3    "path": "$context.path",
4    "headers": {
5      "Authorization": "$input.params('Authorization')"
6    },
7    "body": $input.body
8  }
```

Serverless

Ejercicio 1: Sistema de Seguridad

Paso 7: Probar api productos: Crear Producto

POST ▼ https://1g7q4hzrtb.execute-api.us-west-2.amazonaws.com/prod/productos/crear

Params Authorization Headers (8) Body ● Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☒ JSON ▼

```
1 {
2   "tenant_id": "PLAZA_VEA",
3   "producto_id": "DET001",
4   "nombre": "Detergente Sapolio 2.5 Kg"
5 }
```

POST ▼ https://1g7q4hzrtb.execute-api.us-west-2.amazonaws.com/prod/productos/crear

Params Authorization Headers (8) ● Body Scripts Settings

Headers 👁 7 hidden

	Key	Value
<input checked="" type="checkbox"/>	Authorization	123

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```
1 {
2   "statusCode": 403,
3   "status": "Forbidden - Acceso No Autorizado"
4 }
```

Serverless

Ejercicio 1: Sistema de Seguridad

Paso 7: Probar api productos: Crear Producto

POST ▼ <https://1g7q4hzrtb.execute-api.us-west-2.amazonaws.com/prod/productos/crear>

Params Authorization Headers (8) Body ● Scripts Settings

Headers 7 hidden

	Key	Value
<input checked="" type="checkbox"/>	Authorization	ae975067-bdd4-4e39-89fe-272c8d499ed8

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```
1 {  
2   "statusCode": 200,
```

token (Cadena) ▾	expires
ae975067-bdd4-4e39-89fe-272c8d499ed8	2024-06-24 01:53:04

t_tokens_acceso

t_productos

tenant_id (Cadena) ▾	producto_id (Cadena) ▾	nombre
PLAZA_VEA	DET001	Detergente Sapolio 2.5 Kg

Contenido

Sistema de Seguridad

1. Objetivo del taller 1
2. Arquitectura de Referencia
3. Ejercicio 1: Sistema de Seguridad
4. **Ejercicio 2: Ejercicio Propuesto**
5. Cierre

Serverless

Ejercicio 2: Ejercicio propuesto

- Completar el api productos (**protegida**) con los lambdas siguientes y enviando un token para poder ejecutarlos:
 - ListarProductos (POST) - Enviar tenant_id
 - BuscarProducto (POST) - Enviar tenant_id y producto_id
 - ModificarProducto (PUT) - Enviar tenant_id y producto_id y datos a modificar
 - EliminarProducto (DELETE) - Enviar tenant_id y producto_id

Contenido

Sistema de Seguridad

1. Objetivo del taller 1
2. Arquitectura de Referencia
3. Ejercicio 1: Sistema de Seguridad
4. Ejercicio 2: Ejercicio Propuesto
5. **Cierre**

Cierre:

Sistema de Seguridad - Qué aprendimos?

- Proteger el acceso a las apis

Gracias

Elaborado por docente: Geraldo Colchado