



## Proyecto 4: computación en la nube

Carlos Sebastian Martínez Vidal

Escuela de Ingeniería, Ciencia y Tecnología, Universidad del Rosario

28 de noviembre de 2021

### 1. Definición y arquitectura

Creé una pagina web, desarrollada en python Dash, para exponer **los 7 mejores equipos de las Bundesliga** en la temporada 20-21. La Bundesliga es una de las ligas más entretenidas del mundo y es por esto, que analicé las estadísticas sobre estos equipos; en cuanto a cuáles son: las posiciones que más hacen goles, los equipos que hacen más goles, y las edades en donde se hacen más goles y asistencias. Con esta información podemos hacernos una idea de cuáles son las pautas que tienen que cumplir los equipos de la Bundesliga si en algún año quieren quedar en el top 7.

Para incluir los servicios AWS Serverless, en mi página usé los servicios de **SNS y DynamoDB**. Se le adicionó una caja de comentarios al final de la página, para poder recibir propuestas sobre cómo mejorar la página. Al escribir un comentario y subirlo, este se me enviará por correo para recibir de primera mano el comentario al instante. Así mismo, estos comentarios se guardarán en una base de datos para tenerlos en cuenta en una futura actualización de la página.

### 2. Servicios usados

Instancia:

- SM\_proyecto4
- i-0ef88f6fd95fb747d
- 3.91.158.167
- ec2-3-91-158-167.compute-1.amazonaws.com

Description	Status Checks	Monitoring	Tags
Instance ID	i-0ef88f6fd95fb747d		Public DNS (IPv4)
Instance state	running		ec2-3-91-158-167.compute-1.amazonaws.com
Instance type	t2.micro		IPv4 Public IP
Finding	You may not have permission to access AWS Compute Optimizer.		3.91.158.167
Private DNS	ip-172-31-30-249.ec2.internal		IPv6 IPs
Private IPs	172.31.30.249		-
Secondary private IPs			Elastic IPs
VPC ID	<a href="#">vpc-694fbe14</a>		
		Availability zone	us-east-1a
		Security groups	<a href="#">SM_proyecto</a> , <a href="#">view inbound rules</a> , <a href="#">view rules</a>
		Scheduled events	No scheduled events
		AMI ID	<a href="#">ubuntu/images/hvm-ssd/ubuntu-bionic-</a>

Figura 1: Instancia

## Balanceador de carga:

- ALB-SM
- [arn:aws:elasticloadbalancing:us-east-1:150839082595:loadbalancer/app/ALB-SM/28d635adebf84937](#)
- [ALB-SM-2040897522.us-east-1.elb.amazonaws.com](#)

Load balancer: [ALB-SM](#)




Description	Listeners	Monitoring	Integrated services	Tags
<b>Basic Configuration</b>				
Name	ALB-SM			
ARN	<a href="#">arn:aws:elasticloadbalancing:us-east-1:150839082595:loadbalancer/app/ALB-SM/28d635adebf84937</a> 			
DNS name	<a href="#">ALB-SM-2040897522.us-east-1.elb.amazonaws.com</a>  (A Record)			
State	Active			
Type	application			

Figura 2: Balanceador de carga


## Route 53:

- [sebastian.cf](#)
- [Z074041820XX7S5A6XIKZ](#)



Universidad del Rosario

Escuela de Ingeniería, Ciencia y Tecnología



MACC

Matemáticas Aplicadas y

sebastian.cf

Info

Delete zone

Test record

Configure query logging

▼ Hosted zone details

Edit hosted zone

<div>Hosted zone ID</div> <div>Z074041820XX7S5A6XIKZ</div>	<div>Type</div> <div>Public hosted zone</div>	<div>Name servers</div> <div>ns-1277.awsdns-31.org</div> <div>ns-653.awsdns-17.net</div> <div>ns-291.awsdns-36.com</div> <div>ns-1666.awsdns-16.co.uk</div>
<div>Description</div> <div>-</div>	<div>Record count</div> <div>3</div>	
<div>Query log</div> <div>-</div>		

Figura 3: Dominio

## SNS:

- SM\_Proyecto4\_comments
- arn:aws:sns:us-east-1:150839082595:SM\_proyecto4\_comments

SM\_proyecto4\_comments


Details

<div>Name</div> <div>SM_proyecto4_comments</div>	<div>Display name</div> <div>-</div>
<div>ARN</div> <div>arn:aws:sns:us-east-1:150839082595:SM_proyecto4_comments</div>	<div>Topic owner</div> <div>150839082595</div>
<div>Type</div> <div>Standard</div>	

Figura 4: sns


## DynamoDB:

- SM\_Proyecto4\_comments
- arn:aws:dynamodb:us-east-1:150839082595:table/SM\_Proyecto4\_comments



Universidad del Rosario

Escuela de Ingeniería, Ciencia y Tecnología



MACC

Matemáticas Aplicadas y

SM\_Proyecto4\_comments

Actions

View items

Overview

Indexes

Monitor

Global tables

Backups

Exports and streams

Additional settings

General information

Partition key comment (String)	Sort key -	Capacity mode Provisioned	Table status <div>Active</div> <div>No active alarms</div>
-----------------------------------	---------------	------------------------------	---

Additional info

Figura 5: DynamoDB

### 3. Documentación e instrucciones de despliegue

Ahora, con el fin de que también otras personas aprendan a hacer esta pagina con AWS, se mostrará una documentación sobre que servicios utilizar, como configurarlos y cual es el paso a paso.

Asumimos que se tiene una aplicación(app.py) en Dash y se tiene la intención de subirla a un servidor web con los servicios de amazon AWS.

#### 3.1. EC2

El primer paso es crear una instancia EC2 - Ubuntu Server 18.04, t2.micro. En el grupo de seguridad hay que darle salida por el puerto 80 para que se pueda conectar con el servidor web. Al lanzar la instancia y acceder por el terminal a ella, se sigue el siguiente script para lanzar la aplicación en nuestro servidor para Dash Gunicorn.

```

sudo apt update
sudo apt install python-pip

#Descargar todas las librerias con sudo para no tener problemas con los permisos.

sudo pip3 install plotly
sudo pip3 install pandas
sudo pip3 install dash
sudo pip3 install gunicorn
#Libreria para integrar aplicación de python con los servicios de AWS por consola.
sudo pip3 install boto3

sudo apt-get install gunicorn
#Lanza servidor con el archivo app.py por el puerto 80.
sudo screen gunicorn --workers 1 --log-level=debug --timeout 60 --bind :80 app:server
#Para abandonar el terminal CTRL+A y CTRL+D. El servidor seguira corriendo.

```

Figura 6: Lanzar servidor web en gunicorn

Ya teniendo nuestra app lanzada, se puede acceder usando el IPv4 público de la instancia creada.

---

## 3.2. Dominio

Ahora, se busca que nuestra aplicación web esté alojada en un dominio de nuestro gusto. Para esto primero se tiene que conseguir un dominio en **freenom.com**. Mi dominio es **sebastian.cf**.

### 3.2.1. Target Group

El primer paso entonces consiste en crear un Target Group que alojé la instancia creada anteriormente. El target type debe ser instance, el protocolo HTTP y el puerto 80. Al ser ya creada, en acciones podemos registrar la instancia.

### 3.2.2. Load Balancer

El segundo paso consiste en crear un balanceador de carga de aplicación. En el Scheme debe ser internet-facing, el IP address type debe ser ipv4, y el protocolo y puerto debe ser HTTP y 80. Ahora, para elegir las zonas de disponibilidad en las que trabajará su balanceador de carga, tiene que tener en cuenta la zona en la que fue lanzada su instancia EC2. Seleccioné 2 zonas de disponibilidad y en el grupo de seguridad, use preferiblemente el mismo usado en la instancia. Ahora para configurar el enrutamiento buscamos el Target Group creado anteriormente y usamos el mismo protocolo y puerto.

Ya con esto nuestra instancia esta en nuestro balanceador de carga y se puede acceder a el mediante el nombre del DNS en una pestaña de un buscador.

### 3.2.3. Route 53

Ahora si, para poder montar el dominio tenemos que dirigirnos a Router 53 y creamos una hosted zone. Usamos el nombre de nuestro dominio adquirido anteriormente en freenom y usamos el tipo publico y lo creamos. Al crearlo tendremos por defecto dos records, uno de tipo NS, y uno de tipo SOA. Accedemos al record de tipo NS y encontramos 4 valores. Estos valores son los que usaremos en nuestro dominio de freenom. Entonces, en freenom, en la configuración de nuestro dominio en Nameservers pegamos los 4 valores mencionados anteriormente de nuestro record. Al hacer esto, creamos la conexión entre el dominio y Route 53. No obstante, nos falta la conexión del dominio con el balanceador de carga para poder mostrar el servidor web. Nos dirigimos entonces a Route 53 y creamos un nuevo record. Este tiene que ser de tipo A, en el value tenemos que activar la opción de **Alias** y usar el **Alias to Application and Classic Load Balancer**. Seleccionamos la región general que fue usada para nuestra instancia y nuestro balanceador de



carga, y escogemos el balanceador de carga. La routing policy debe ser Simple routing. Al crear este record ya tenemos la conexión y nuestro dominio ya debe estar en funcionamiento con nuestro servidor web. Se accede escribiendo el nombre del dominio en una pestaña de un buscador.

### 3.3. AWS serverless

Ahora para la integración con los servicios AWS serverless, se usarán DynamoDB y SNS para guardar comentarios en una base de datos y recibir estos por correo. Para hacer esto con nuestra aplicación de dash necesitamos usar estos servicios mediante la terminal. Para esto, necesitamos poder autenticarnos con una IAM Role.

#### 3.3.1. IAM Role

En nuestra cuenta de AWS nos dirigimos a la parte superior derecha y buscamos las security credentials. En la primera pagina donde aparece, procedemos a ir **Access Keys for CLI**. Creamos una access key y descargamos el .csv que se genera. Estas credenciales deben estar guardadas de manera segura, ya que es un acceso directo a su cuenta. Ahora con el ID de acceso y la Clave de acceso generada, seguiremos los pasos del siguiente script .

```
sudo apt install awscli
sudo apt-get update
aws configure
#Usar ID y clave de acceso creada
#default region la usada en su instancia(us-east-1)
#default output format dejelo vacio.
aws ec2 describe-instances
#Listara las instancias creadas en su cuenta aws.
```

Figura 7: Ingreso por CLI de AWS

Con esto, ya podremos trabajar desde la terminal para usar los servicios de AWS Serverless con nuestra cuenta.

#### 3.3.2. Dynamo DB

Ahora, nos dirigimos a DynamoDB y creamos una tabla. Asigne un nombre y tenga en cuenta la **Partition Key** y **Sort Key** y sus tipos. Estas servirán como la primary key y la key de búsqueda de su tabla, respectivamente. Tenga en cuenta, el nombre de su tabla porque lo necesitaremos luego.

#### 3.3.3. SNS

Ahora, nos dirigimos a SNS y creamos un tema, del tipo standard. Procedemos a crear una subscripción para nuestro tema. Al crearlo, seleccionamos como protoco-



lo: correo electrónico y en el punto de enlace, el correo al cual quiere que le lleguen las notificaciones. Finalmente, cree la subscripción y acepte la confirmación de esta en su correo usado anteriormente. Tenga en cuenta, el ARN de su tema porque lo necesitaremos luego.

### 3.3.4. Uso por terminal

Ya con nuestros servicios creados y listos para usar, procedemos a usarlos mediante nuestra aplicación de Dash. Para esto use en su archivo app.py las instrucciones del siguiente Script.

```
import boto3

#Se busca el servicio
sns = boto3.resource('sns')
#Se inicializa el tema creado
topic = sns.Topic(arn="arn:aws:sns:us-east-1:150839082595:SM_proyecto4_comments")
#Se publica el mensaje
topic.publish(Message=value)

dynamodb = boto3.resource('dynamodb')
#Se inicializa la tabla creada
table = dynamodb.Table('SM_Proyecto4_comments')
#Se ingresa el item a la tabla
table.put_item(
    Item={
        'partition key' : 'sort key',
    })
```

Figura 8: AWS serverless

Noté que, ya con el CLI podemos buscar libremente sobre nuestra cuenta de AWS. El primer paso, consiste en buscar el servicio que se necesita. En nuestro caso **sns** y **DynamoDB**. Para sns necesitamos el ARN del tema creado y para DynamoDB el nombre de la tabla solamente. Al momento de ingresar el item a la tabla note que funciona como un diccionario, en donde el primer valor es la partition key y el segundo la sort key. Con esto, ya podremos ingresar items a nuestra base de datos y publicar mensajes hacia nuestro correo.

#### 4. Diagrama y wireframe

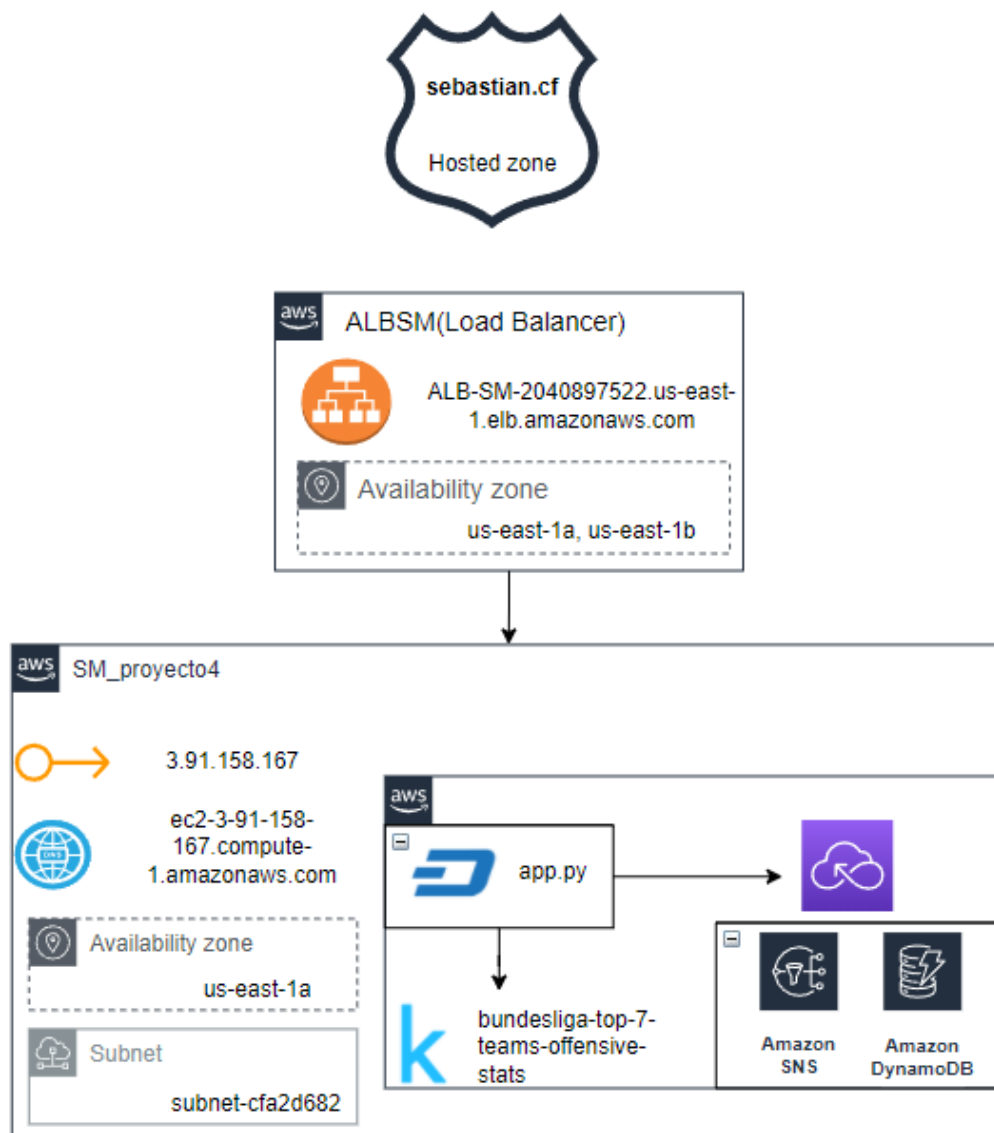


Figura 9: Arquitectura de la aplicación



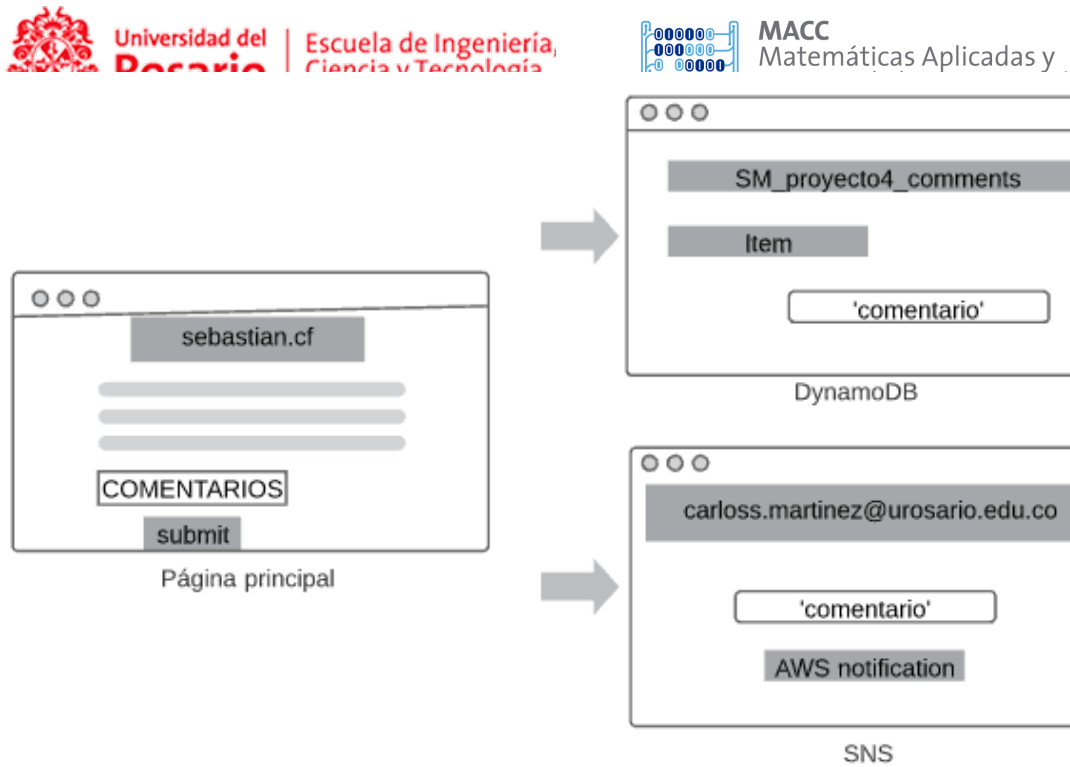


Figura 10: Wireframe de los servicios Serverless

## 5. Evidencia de funcionamiento

Primero, nos dirigimos a hacer un comentario en la pagina.

If you have any suggestions on how to upgrade this web page, please leave a comment below.

Me gustaría que tuviera mas

SUBMIT

Figura 11: Dash comment

Al hacer este comentario, notamos que, en DynamoDB se guarda como un item de nuestra tabla.

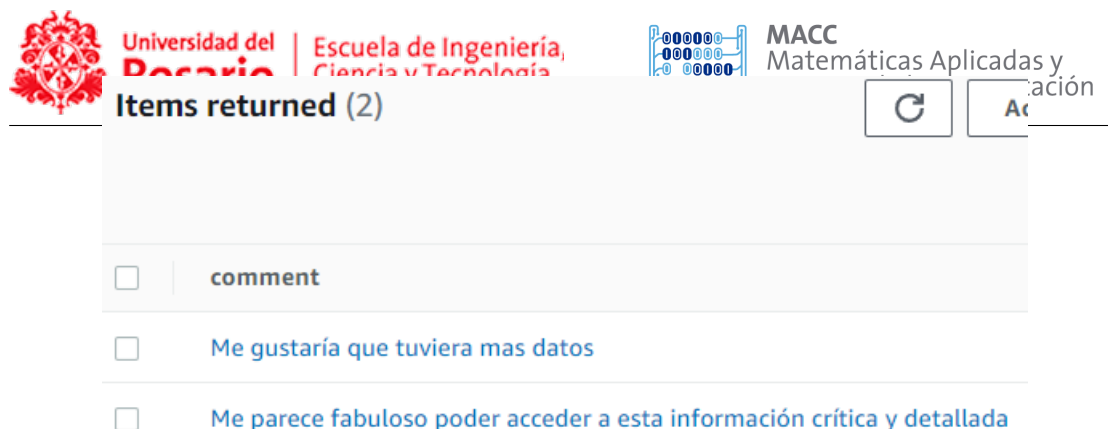


Figura 12: DynamoDB comment

Así mismo, este comentario llega por correo usando SNS.

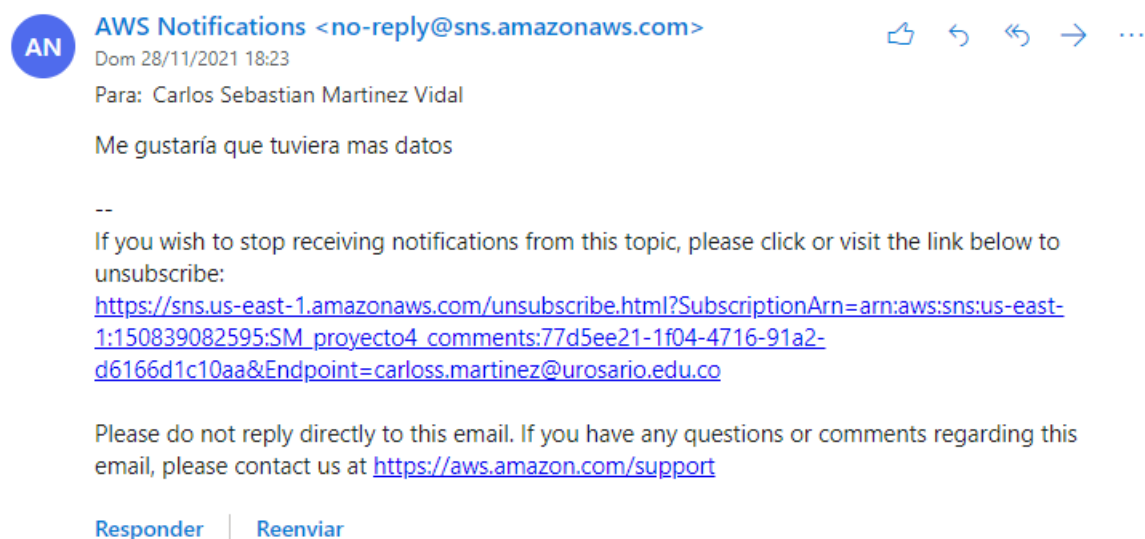


Figura 13: SNS comment