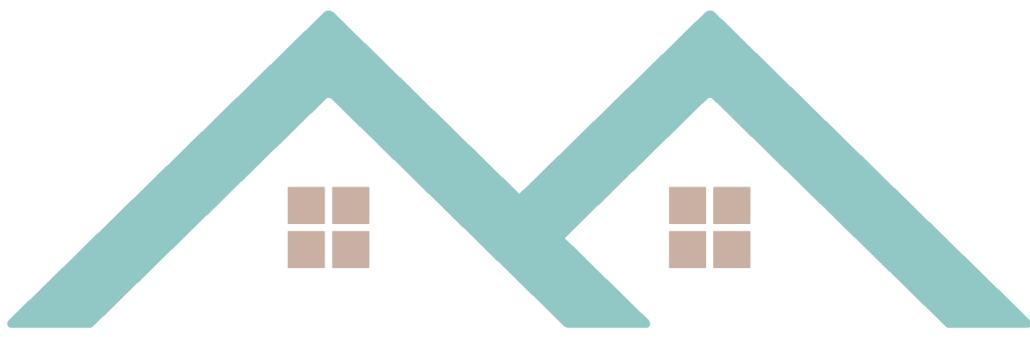


CERTIFIED TECH DEVELOPER

Proyecto Final Integrador

Equipo 7

Infraestructura



COMO EN CASA

La infraestructura del trabajo integrador que se nos propone, es una parte fundamental de nuestro proyecto. La carga de éste en la nube global de AWS supone la posibilidad de acceder a él de manera ya no local, sino a través de cualquier dispositivo electrónico ingresando la IP de nuestra instancia.

Amazon Web Service es una nube confiable, segura y amplia, que cuenta con muchísimos servidores a nivel mundial y que como tal se constituye en uno de los mayores proveedores de almacenamiento, recursos y bases de datos, entre otras cosas.

Así, como issue, se requirió elaborar en primer lugar, la infraestructura necesaria para poder alojar allí nuestro proyecto final.

SPRINT II:

En esta etapa del desarrollo de nuestro proyecto debimos crear la infraestructura necesaria para alojar el sitio. Como consecuencia, realizamos la apertura de la cuenta en AWS, creación del usuario root con su correspondiente MFA y un usuario IAM como estaba recomendado.

Con posterioridad continuamos con los requerimientos del primer issue en cuestión, tal como se prueba con las capturas de pantallas que acompañamos a continuación:

1) Crear una instancia de EC2 en AWS para el servicio web:

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with 'New EC2 Experience' and various EC2-related options like 'Panel de EC2', 'EC2 Global View', 'Eventos', 'Etiquetas', 'Límites', and 'Instancias'. Under 'Instancias', there are links for 'Tipos de instancia', 'Plantillas de lanzamiento', 'Solicitudes de spot', 'Savings Plans', 'Instancias reservadas', 'Hosts dedicados', and 'Reservas de capacidad'. The main area displays a table titled 'Instancias (1) Información' with one row for 'grupo7'. The table columns include 'Name', 'ID de la instancia', 'Estado de la i...', 'Tipo de inst...', 'Comprobación ...', 'Estado de la ...', and 'Zona de dispon...'. Below the table, a modal window titled 'Seleccione una instancia anterior' is open. At the bottom of the page, there are links for 'Comentarios', 'Español', and standard AWS footer links: '© 2008 - 2021, Amazon Web Services, Inc. o sus empresas afiliadas. Todos los derechos reservados.', 'Política de privacidad', 'Términos de uso', and 'Preferencias de cookies'.

NOTAS:

Como se visualiza, la instancia creada pertenece a la capa t2.micro -capa gratuita o free tier-, mientras no se encuentra en uso está detenida, y para poder utilizarla en el proyecto se habilitaron sus reglas de entrada de la siguiente manera:

The screenshot shows the AWS Security Groups page for the 'grupo7' instance. On the left, there's a sidebar with 'Tipos de instancia', 'Plantillas de lanzamiento', 'Solicitudes de spot', 'Savings Plans', 'Instancias reservadas', 'Hosts dedicados', and 'Reservas de capacidad'. The main area shows a table for the 'Instancia: i-011a187b8b4ce676d (grupo7)'. The table has columns for 'ID de la regla del grupo ...', 'Intervalo de p...', 'Protocolo', 'Origen', and 'Grupos de seguridad'. There are five entries in the table. Below the table, there's a section titled 'Reglas de salida'. At the bottom of the page, there are links for 'Comentarios', 'Español', and standard AWS footer links: '© 2008 - 2021, Amazon Web Services, Inc. o sus empresas afiliadas. Todos los derechos reservados.', 'Política de privacidad', 'Términos de uso', and 'Preferencias de cookies'.

Resta aclarar, que tal como fue solicitado, tiene una vpc "default" con IP pública, con una key privada y un security group que permite el acceso desde el mundo al puerto de su aplicación y acceso al puerto 22 desde sus respectivas IP. Por último, atento a lo peticionado, se instaló el servidor web Apache.



2) Crear una instancia de RDS en AWS para alojar la base de datos

NOTAS:

Se creó una base de datos o RDS "Relational Database Service" que permite obtener acceso a las funciones de conocidas bases de datos, en este caso MySQL. Esta se encuentra comprendida dentro del free-tier, tiene una vpc "default" con IP pública, un security group permitiendo el acceso al puerto 3306 desde el security group de la instancia EC2, y desde sus respectivas IP. A su vez, se la conectó al Workbench de MySQL y al backend de nuestro proyecto.

a) Imagen de la consola de AWS

AWS Servicios ▾

RDS > Databases > grupo7

Buscar servicios, características, blogs, documentos y mucho más [Alt+S]

BrendaNolan ▾ São Paulo ▾ Soporte

Amazon RDS

Dashboard

Databases

Performance Insights

Snapshots

Automated backups

Reserved instances

Subnet groups

Parameter groups

Option groups

Events

Event subscriptions

Recommendations (3)

Certificate update

RDS > Databases > grupo7

grupo7

Modificar Acciones ▾

Resumen

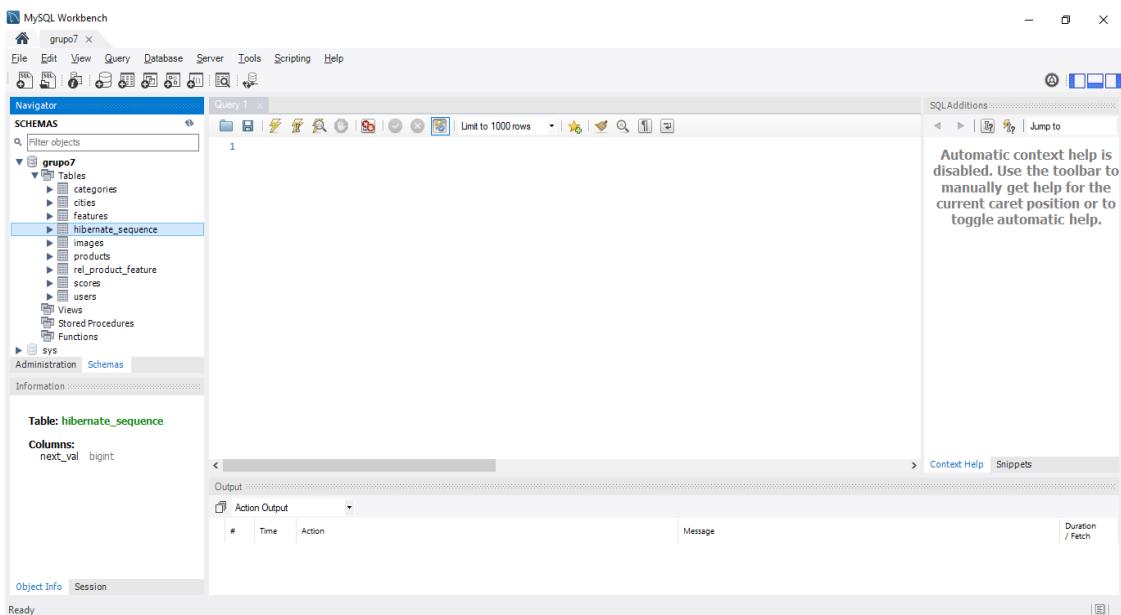
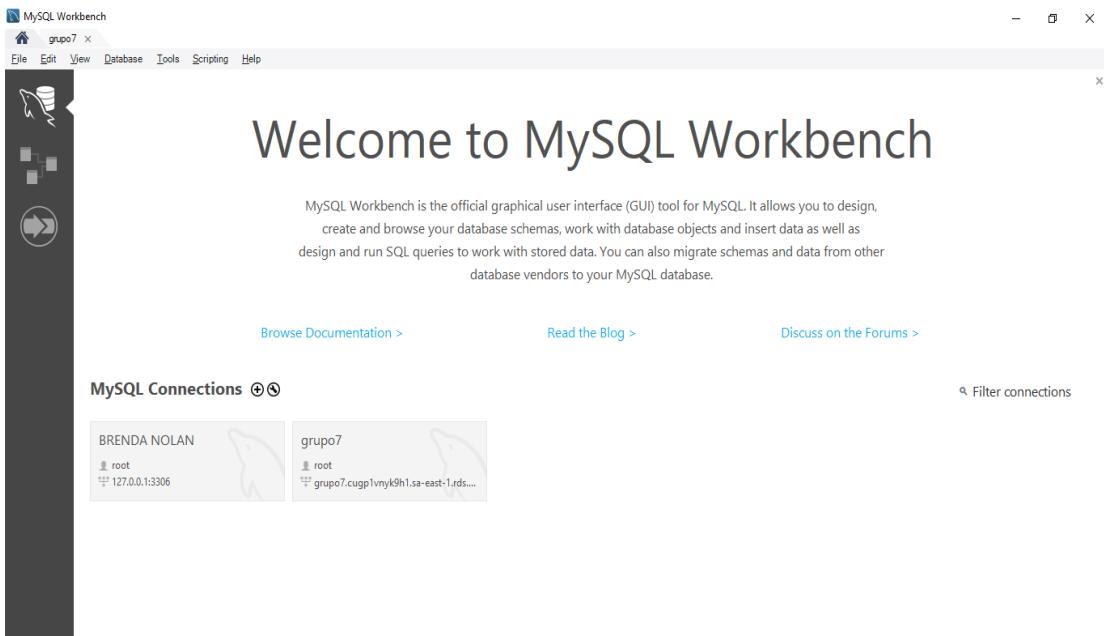
Identificador de base de datos grupo7	CPU -	Estado DETENIDO	Clase db.t2.micro
Role Instancia	Actividad actual	Motor MySQL Community	Región y AZ sa-east-1c

Conectividad y seguridad | Monitoreo | Registros y eventos | Configuración | Mantenimiento y copias de seguridad | Etiquetas

Conectividad y seguridad

Punto de enlace y puerto	Redes	Seguridad
Punto de enlace grupo7.cugp1nyk9h1.sa-east-1.rds.amazonaws.com	Zona de disponibilidad sa-east-1c	Grupos de seguridad de la VPC default (sg-0a40dcecb2dc77cc) Activo
Puerto 3306	VPC vpc-040cb91058a325523	Accesible públicamente Sí
	Grupo de subredes default-vpc-040cb91058a325523	Entidad de certificación

- b) Imágenes de MySQL Workbench (en la primera la conexión hecha y en la segunda las tablas creadas provenientes del código alojado en la instancia local)



3) Crear un Bucket S3 para almacenar las imágenes de los productos

NOTAS:

Para finalizar se requería la creación de un bucket, es decir, de un contenedor de objetos. Esto, con la finalidad de poder almacenar todas las imágenes de nuestro proyecto, que por el momento serían aquellas pertenecientes a las categorías y productos de nuestra página.

a) Imagen del bucket creado

The screenshot shows the AWS S3 service page. On the left, there's a sidebar with 'Amazon S3' selected. Under 'Buckets', it lists several options like 'Puntos de acceso', 'Lambda', etc., and shows a table for 'Buckets (1)'. The table has one row for 'grupo7' located in 'América del Sur (São Paulo)' with 'sa-east-1' access. The table includes columns for Nombre, Región de AWS, Acceso, and Fecha de creación.

Nombre	Región de AWS	Acceso	Fecha de creación
grupo7	América del Sur (São Paulo) sa-east-1	Bucket y objetos que no son públicos	3 Nov 2021 2:48:18 PM -03

b) Captura del bucket con las correspondientes carpetas creadas, que contienen algunas imágenes en su interior.

This screenshot shows the 'grupo7' bucket details page. The 'Objetos' tab is active, displaying two objects: 'Categorías/' and 'Productos/'. Both are listed as 'Carpeta' type objects. The table includes columns for Nombre, Tipo, Última modificación, Tamaño, and Clase de almacenamiento.

Nombre	Tipo	Última modificación	Tamaño	Clase de almacenamiento
Categorías/	Carpeta	-	-	-
Productos/	Carpeta	-	-	-

- c) Captura del registro creado para la tabla “categories”, donde se insertan una imagen por cada category “campo, ciudad, playa y montaña”

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar displays the Navigator with the 'group7' schema selected, showing tables like 'cities', 'features', and 'categories'. The 'categories' table has columns: id_category (bigint PK), description (varchar(255)), title (varchar(255)), and url_image (varchar(255)). The main area contains a query editor titled 'Query 1' with the following SQL code:

```
1 • INSERT INTO categories VALUES ('1', 'Si sos de los que busca siempre un respiro y preferís pasar tus vacaciones en ');
2 );
3 • INSERT INTO categories VALUES ('2', 'Momentos divertidos, alegres, de aventura, de emociones, lo vivís aquí, en la ');
4 );
5 • INSERT INTO categories VALUES ('3', 'En armonía con el espíritu de la naturaleza salvaje que nos rodea, vení a dis ');
6 );
7 • INSERT INTO categories VALUES ('4', 'Si lo tuyo es el turismo urbano, visitar museos y vivir las mejores experienc ');
8 );
9
10
11
```

The status bar at the bottom indicates 'Query Completed'.

- d) Captura del registro creado para la tabla “cities”, donde se insertan nombres de ciudades, junto con su id y país

The screenshot shows the MySQL Workbench interface with the following details:

- File**, **Edit**, **View**, **Query**, **Database**, **Server**, **Tools**, **Scripting**, **Help** menu.
- Navigator** pane on the left showing **SCHEMAS** (selected), **Tables** (selected), and **Columns**.
- Query 1** tab with the title **Insert Ciudades**.
- SQL Editor** content:

```
1 • INSERT INTO cities VALUES ('1', 'Argentina', 'Buenos Aires');
2 • INSERT INTO cities VALUES ('2', 'Argentina', 'Rosario');
3 • INSERT INTO cities VALUES ('3', 'Argentina', 'Córdoba');
4 • INSERT INTO cities VALUES ('4', 'Argentina', 'San Juan');
5 • INSERT INTO cities VALUES ('5', 'Argentina', 'Mendoza');
6 • INSERT INTO cities VALUES ('6', 'Argentina', 'Bariloche');
7 • INSERT INTO cities VALUES ('7', 'Argentina', 'Villa La Angostura');
8 • INSERT INTO cities VALUES ('8', 'Argentina', 'Neuquén');
9 • INSERT INTO cities VALUES ('9', 'Argentina', 'San Miguel de Tucumán');
10 • INSERT INTO cities VALUES ('10', 'Argentina', 'La Rioja');
11 • INSERT INTO cities VALUES ('11', 'Argentina', 'Río Gallegos');
```
- Result Grid** pane showing the first five rows of the **cities** table:

	id_city	country	name
1	Argentina	Buenos Aires	
2	Argentina	Rosario	
3	Argentina	Córdoba	
4	Argentina	San Juan	
5	Argentina	Mendoza	

- Action Output** pane showing the log of recent actions:

#	Time	Action	Message	Duration / Fetch
32	16:34:22	INSERT INTO cities VALUES ('27', 'Colombia', 'Ibagué')	1 row(s) affected	0.062 sec
33	16:34:22	INSERT INTO cities VALUES ('28', 'Colombia', 'Villavicencio')	1 row(s) affected	0.063 sec
34	16:34:22	select * from cities LIMIT 0, 1000	28 row(s) returned	0.063 sec / 0.000 sec

- Object Info** and **Session** tabs at the bottom.

SPRINT III:

Para éste sprint se requería el Deploy de la aplicación (tanto del front end y el back end) de manera obligatoria al menos manualmente en el servidor web EC2 en AWS. Para efectuar esto se utilizó la instancia de EC2 ya creada en el sprint anterior.

Fue necesario como paso previo la instalación desde la consola de EC2 de ciertos paquetes utilizados desde el back end de nuestro proyecto y del servidor nginx.

- a) En primer lugar se instaló Node.js con el comando “sudo apt install nodejs” y se comprobó que la instalación se haya realizado de forma correcta haciendo una consulta a node sobre su número de versión con el comando “nodejs -v”.
- b) Luego, se instaló el npm con el comando “sudo apt install npm”.

```
node-clients node-clusters node-color node-color-converter node-color-name node-columnify node-combined-stream node-concat-map node-concat-stream
node-config-chain node-configstore node-console-control-strings node-copy-concurrently node-core-util-is node-cross-spawn node-crypto-random-string node-cyclist
node-dashdash node-debug node-decompress node-decompress-response node-default-extname node-define-properties node-decode-encoding node-delegates
node-detect-indent node-detect-newline node-dot-prop node-duplexify3 node-ecc-jspb node-editor node-encoding node-end-of-stream node-err-code
node-errno node-es6-promise node-escape-string-regexp node-exec node-extend node-extrify node-fast-equal node-find-up node-flush-large-stream
node-forever-agent node-form-data node-from2 node-fs-vacuum node-fs-write-stream node-fs-atoms node-fs-readdirpath node-function-bind node-gauge node-genfun
node-get-called file node-get-stream node-getpass node-glob node-goodegraceful fs node-gyp node-har-schema node-hex-validation node-has-flag
node-has-symbol-support node-has-to-string-tag node-has-unicomp node-hosted-git info node-http-signature node-iconv-lite node-ifr node-import-lazy
node-inumurhash node-inflight node-inherits node-invert-kv node-ip-rege node-is-node-is-object node-is-path-inside
node-isplainobj node-is-stream node-is-streamarray node-isstream node-isxex node-isxstream node-isurl node-isbn node-json-parse-better-errors
node-json-schema node-json-schema-traverse node-json-stable-stringify node-json-stringify safe node-jsonify node-jsonparse node-jsonstream node-jsprim
node-latest-version node-lazy-property node-lcid node-libnpx node-locate-path node-lockfile node-lodash node-lodash-packages node-lowercase-keys node-lru-cache
node-make-dir node-mem node-mime node-mime-types node-mimic-fn node-mimic-response node-minimatch node-minimist node-mississippi node-mkdirp node-move-concurrently
node-ms node-mute-stream node-node-normalize-package-data node-npm-bundled node-npm-packagarg node-npm-run-path node-npmlog node-number-isnan
node-oauth-sign node-object-assign node-once node-openner node-os-locale node-openssl node-osenv node-p-cancelable node-p-finlly node-p-is-pronise node-p-limit
node-p-locate node-p-lineout node-package-json node-paralleter transform node-path-is-absolute node-path-is-inside node-performance-now node-pify
node-prepend-http node-process-nextick-args node-promise-inflight node-promise-retry node-promzard node-proto-list node-prn node-pseudomap node-psl node-pump
node-pumpify node-punycode node-qm node-re node-read node-read-package json node-readable-stream node-registry-auth-token node-registry-url node-request
node-require-directory node-require-main-filename node-resolve node-resolve-from node-retry node-rimraf node-run-queue node-safe-buffer node-semver node-semver-diff
node-set-blocking node-sha-bash-command node-shebang-regex node-signal-exit node-slash node-slide node-sorted-object node-spdx-correct node-spdx-exceptions
node-spdx-expression-parse node-spdx-license-ids node-sshpk node-ssri node-stream-each node-stream-iterate node-stream-shift node-strict-uri-encode
node-string-decoder node-string-width node-stripansi node-strip-eof node-stripjson-comments node-supports-color node-term-size node-text-table
node-through node-through2 node-timed-out node-touchy-cookie node-tunnel-agent node-tweetnact node-typearray node-typearray-to-buffer node-uid-number
node-unique-filename node-unique-string node-urnipe node-uris node-urts-parse-tax node-urts-to-options node-util-deprecate node-uuid
node-validate-npm-package-license node-validate-npm-package-name node-veyor node-wcwidth.js node-which node-which-module node-wide-align node-widest-line
node-wrap-ansi node-wrapry node-write-file-atomic node-xdg-basedir node-xtend node-y18n node-yallist node-yargs node-yargs-parser npm perl-openssl-defaults
python-pkg-resources python2 python2-minimal python2.7 python2.7-minimal x11-xserver-utils xdg-utils
The following packages will be upgraded:
  libssl1.1
1 upgraded, 378 newly installed, 0 to remove and 23 not upgraded.
Need to get 56.6 MB of archives.
After this operation, 251 Mb of additional disk space will be used.
Do you want to continue? [Y/n] ■
```

i-011a187b8b4ce676d (grupo7)
Public IPs: 18.228.196.51 Private IPs: 172.31.12.71

- c) A su vez, se bajó el jdk versión 17 que es la utilizada desde el back end del proyecto.

```
Setting up node-configstore (5.0.1-1) ...
Setting up node-boxen (4.2.0-2) ...
Setting up g++ (4:9.3.0-ubuntu2) ...
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mode
Setting up build-essential (12.ubuntu1.1) ...
Setting up node-pnmlog (4.1.2-2) ...
Setting up node-yargs (15.3.0-1) ...
Setting up node-cacache (11.3.3-2) ...
Setting up node-read-package-json (2.1.1-1) ...
Setting up node-gyp (6.1.0-3) ...
Setting up node-libnpx (10.2.1-2) ...
Setting up npm (6.14.4-0ubuntu2) ...
Setting up libwww-perl (6.43-1) ...
Setting up liblwp-protocol-https-perl (6.07-2ubuntu2) ...
Setting up libxml-parser-perl (2.46-1) ...
Setting up libxml-treegr-perl (1.3.50-2) ...
Setting up libnet-dbus-perl (1.2.0-1)
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for nisme-support (3.6ubuntu1) ...
Processing triggers for libc-bin (2.31-ubuntu9.2) ...
ubuntu@ip-172-31-12-71:~$ wget https://download.java.net/java/GA/jdk17/0d48333a00540d886896bac774ff48b/35/GPL/openjdk-17_linux-x64_bin.tar.gz
--2021-11-24 22:17:37 -> https://download.java.net/java/GA/jdk17/0d48333a00540d886896bac774ff48b/35/GPL/openjdk-17_linux-x64_bin.tar.gz
Resolving download.java.net (download.java.net)... 104.112.148.65
Connecting to download.java.net (download.java.net)|104.112.148.65|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 186661523 (178M) [application/x-gzip]
Saving to: openjdk-17_linux-x64_bin.tar.gz

openjdk-17_linux-x64_bin.tar.gz          100%[=====] 178.01M  112MB/s   in 1.6s
2021-11-24 22:17:40 (112 MB/s) - 'openjdk-17_linux-x64_bin.tar.gz' saved [186661523/186661523]
ubuntu@ip-172-31-12-71:~$ ■
```

i-011a187b8b4ce676d (grupo7)
Public IPs: 18.228.196.51 Private IPs: 172.31.12.71

d) Se instaló maven:

```
[jdk-17/lib/modules
[jdk-17/lib/psfnt.properties.ja
[jdk-17/lib/psfntj2d.properties
[jdk-17/lib/security/blocked.certs
[jdk-17/lib/security/cacerts
[jdk-17/lib/security/default.policy
[jdk-17/lib/security/public_suffix_list.dat
[jdk-17/lib/server/classes.jsa
[jdk-17/lib/server/classes.nocoops.jsa
[jdk-17/lib/server/libjsig.so
[jdk-17/lib/server/libjvm.so
[jdk-17/lib/src.zip
[jdk-17/lib/tzdb.dat
[jdk-17/release
ubuntu@ip-172-31-12-71:~$ sudo mv jdk-17 /opt/
ubuntu@ip-172-31-12-71:~$ export JAVA_HOME=/opt/jdk-17
ubuntu@ip-172-31-12-71:~$ export PATH=$PATH:$JAVA_HOME/bin
ubuntu@ip-172-31-12-71:~$ source ./bashrc
ubuntu@ip-172-31-12-71:~$ echo $JAVA_HOME
/opt/jdk-17
ubuntu@ip-172-31-12-71:~$ /opt/jdk-17
.bash: /opt/jdk-17: Is a directory
ubuntu@ip-172-31-12-71:~$ java -version
openjdk 11.0.11 2021-04-20
OpenJDK Runtime Environment (build 11.0.11+9-Ubuntu-2.20.04)
OpenJDK 64-Bit Server VM (build 11.0.11+9-Ubuntu-2.20.04, mixed mode)
ubuntu@ip-172-31-12-71:~$ mvn -version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 17, vendor: Oracle Corporation, runtime: /opt/jdk-17
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "5.11.0-1021-aws", arch: "amd64", family: "unix"
ubuntu@ip-172-31-12-71:~$ ]
```

i-011a187b8b4ce676d (grupo7)

Public IPs: 18.228.196.51 Private IPs: 172.31.12.71

e) Por último se instaló nginx con el comando “sudo apt install nginx”.

```
Unpacking libnginx-mod-http-xslt-filter (1.18.0~Ubuntu1.2) ...
Selecting previously unselected package libnginx-mod-mail.
Preparing to unpack .../07-libnginx-mod-mail_1.18.0~Ubuntu1.2_amd64.deb ...
Unpacking libnginx-mod-mail (1.18.0~Ubuntu1.2) ...
Selecting previously unselected package libnginx-mod-stream.
Preparing to unpack .../08-libnginx-mod-stream_1.18.0~Ubuntu1.2_amd64.deb ...
Unpacking libnginx-mod-stream (1.18.0~Ubuntu1.2) ...
Selecting previously unselected package nginx-core.
Preparing to unpack .../09-nginx-core_1.18.0~Ubuntu1.2_amd64.deb ...
Unpacking nginx-core (1.18.0~Ubuntu1.2) ...
Selecting previously unselected package nginx.
Preparing to unpack .../10-nginx_1.18.0~Ubuntu1.2_all.deb ...
Unpacking nginx (1.18.0~Ubuntu1.2) ...
Setting up nginx-common (1.18.0~Ubuntu1.2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /lib/systemd/system/nginx.service.
Setting up libnginx0:amd64 (2.1.3~1~Ubuntu2) ...
Setting up libnginx-mod-http-xslt-filter (1.18.0~Ubuntu1.2) ...
Setting up libwebp0:amd64 (0.8.1~2~Ubuntu20.04.1) ...
Setting up libtiff5:amd64 (4.1.0~git191117~Ubuntu0.20.04.2) ...
Setting up libnginx-mod-mail (1.18.0~Ubuntu1.2) ...
Setting up libnginx-mod-stream (1.18.0~Ubuntu1.2) ...
Setting up libgd3:amd64 (2.2.5~5~Ubuntu2.1) ...
Setting up libnginx-mod-http-image-filter (1.18.0~Ubuntu1.2) ...
Setting up nginx-core (1.18.0~Ubuntu1.2) ...
Not attempting to start NGINX, port 80 is already in use.
Setting up nginx (1.18.0~Ubuntu1.2) ...
Processing triggers for ufw (0.36-6) ...
Processing triggers for systemd (245.4-4ubuntu3.13) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
ubuntu@ip-172-31-12-71:~$ ]
```

i-011a187b8b4ce676d (grupo7)

Public IPs: 18.228.196.51 Private IPs: 172.31.12.71

Finalizadas las instalaciones de paquetes se procedió a comenzar con el deploy manual.

1) Deploy manual del backend

En primer lugar se hizo el deploy del backend, para esto se confeccionó el archivo .jar específicamente desde maven. Desde el lifecycle de éste se hizo un install y se creó entonces allí en la carpeta target el archivo mencionado. Así, con este archivo y el archivo pem de las claves obtenido en el sprint anterior, se confeccionó una carpeta de archivos a la cual denominamos “deploy”. Hecho ésto, desde gitbash se copió el archivo via ssh a la EC2.

Comandos:

i) scp -i "grupo7.pem" ./grupo7-0.0.1-SNAPSHOT.jar ubuntu@52.67.178.177:~

```
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrado
r/deploy
$ scp -i "grupo7.pem" ./grupo7-0.0.1-SNAPSHOT.jar.original ubuntu@52.67.178.177:~
The authenticity of host '52.67.178.177 (52.67.178.177)' can't be established.
ECDSA key fingerprint is SHA256:ytmevKvdw5zzcafKgqlRBqHlkvdhTIhqXL4SycTFls.
Are you sure you want to continue connecting (yes/no/[Fingerprint])? yes
Warning: Permanently added '52.67.178.177' (ECDSA) to the list of known hosts.
grupo7-0.0.1-SNAPSHOT.jar.original                                         100%   89KB 401.5KB/s  00:00

Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ scp -i "grupo7.pem" ./grupo7-0.0.1-SNAPSHOT.jar ubuntu@52.67.178.177:~
grupo7-0.0.1-SNAPSHOT.jar                                              100%   52MB  2.2MB/s  00:23

Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ |
```

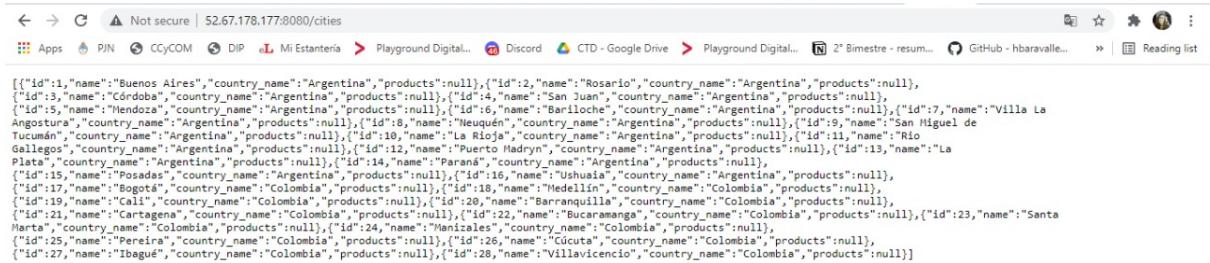
Una vez que realizamos esto, desde la consola EC2, ejecutamos el comando:

ii) nohup java -jar grupo7-0.0.1-SNAPSHOT.jar & (es para ejecutar el back de nuestro proyecto de manera permanente.

Nos sucedía que ejecutándolo con el comando java -jar grupo7-0.0.1-SNAPSHOT.jar & una vez que cerrábamos la instancia desaparecía la posibilidad de hacer llamados al back.

De ésta manera, con la instancia de EC2 funcionando, y, abriendo una pestaña con la IP Pública:8080/llamado que se desea efectuar, accedes a tu información del backend.

En nuestro caso sería <http://52.67.178.177:8080/cities> por ejemplo:



2) Deploy manual del frontend

Ahora en la parte del deploy manual del frontend, comenzamos abriendo visual studio code. Desde allí, con el comando:

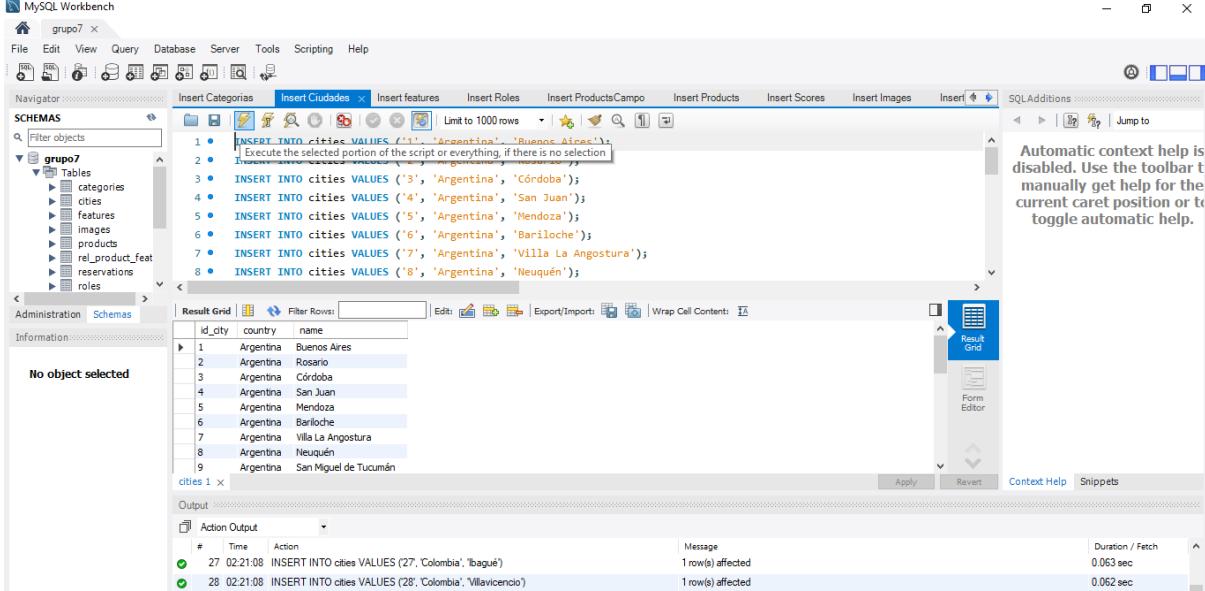
- i) “npm run build” se crea la carpeta correspondiente en la carpeta de nuestro Front del proyecto, que se agrega a la carpeta creada con anterioridad, en nuestro caso a la carpeta “deploy”. Ojo, de manera previa, tengo que eliminar la carpeta build anteriormente creada para el deploy porque de lo contrario, la instancia de ec2 presenta errores. Así, eliminada la carpeta build desde la ec2 con el comando rm -r build y una vez creada nuevamente la carpeta build en la carpeta deploy, desde gitbash ingresas el comando “scp -r -i "grupo7.pem" ./build ubuntu@52.67.178.177:~
- ii) Luego desde la consola EC2, con cd build me paro en la posición correcta y con el comando “sudo cp -r * /var/www/html/” subis a la instancia de EC2 la parte de front.
- iii) Luego se puede acceder desde otra pestaña con la IP pública a visualizar la pantalla.

```
MINGW64:/d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ scp -i "grupo7.pem" ./grupo7-0.0.1-SNAPSHOT.jar.original ubuntu@52.67.178.177:~
The authenticity of host '52.67.178.177 (52.67.178.177)' can't be established.
ECDSA key fingerprint is SHA256:ytmewVdW5ZzcafkqgLRBgHlkvdhTlhQX4YSycTFIs.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '52.67.178.177' (ECDSA) to the list of known hosts.
grupo7-0.0.1-SNAPSHOT.jar.original                                         100%   89KB  401.5KB/
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ scp -i "grupo7.pem" ./grupo7-0.0.1-SNAPSHOT.jar ubuntu@52.67.178.177:~
grupo7-0.0.1-SNAPSHOT.jar                                              100%   52MB   2.2MB/
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ ls
build/ grupo7-0.0.1-SNAPSHOT.jar grupo7.pem
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ scp -r -i "grupo7.pem" ./build ubuntu@52.67.178.177:~
asset-manifest.json                                                 100% 1066   13.6KB/
favicon.ico                                                       100% 3870   56.2KB/
index.html                                                        100% 2287   35.4KB/
manifest.json                                                    100% 517    8.2KB/
robots.txt                                                       100% 70    1.0KB/
2.2b9b9819.chunk.css                                             100% 25KB  338.4KB/
2.2b9b9819.chunk.css.map                                         100% 35KB  459.5KB/
main.b2obbdse.chunk.css                                           100% 31KB  301.3KB/
main.b2obbdse.chunk.css.map                                     100% 95KB  661.8KB/
2.b00c2e2c.chunk.js                                              100% 444KB  1.7MB/
2.b00c2e2c.chunk.js/LICENSE.txt                                100% 1602   21.9KB/
2.b00c2e2c.chunk.js.map                                         100% 2809KB  1.9MB/
main.cbc4c42bc.chunk.js                                         100% 25KB  345.7KB/
main.cbc4c42bc.chunk.js.map                                    100% 52KB  545.5KB/
runtime-main.17e08ee8.js                                         100% 1561   20.2KB/
runtime-main.17e08ee8.js.map                                  100% 8265  129.6KB/
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/deploy
$ |
```

Ahora, como éste deploy fue hecho manualmente, para volver a actualizar back end y front end de nuestro proyecto deben repetirse los comandos ingresados y subir los cambios cada

vez que se efectúen. Recordemos que la base de datos creada en AWS consume el back end de nuestro proyecto, por lo que no es necesario actualizarla conforme el deploy.

3) Creación de tablas de la RDS en AWS.



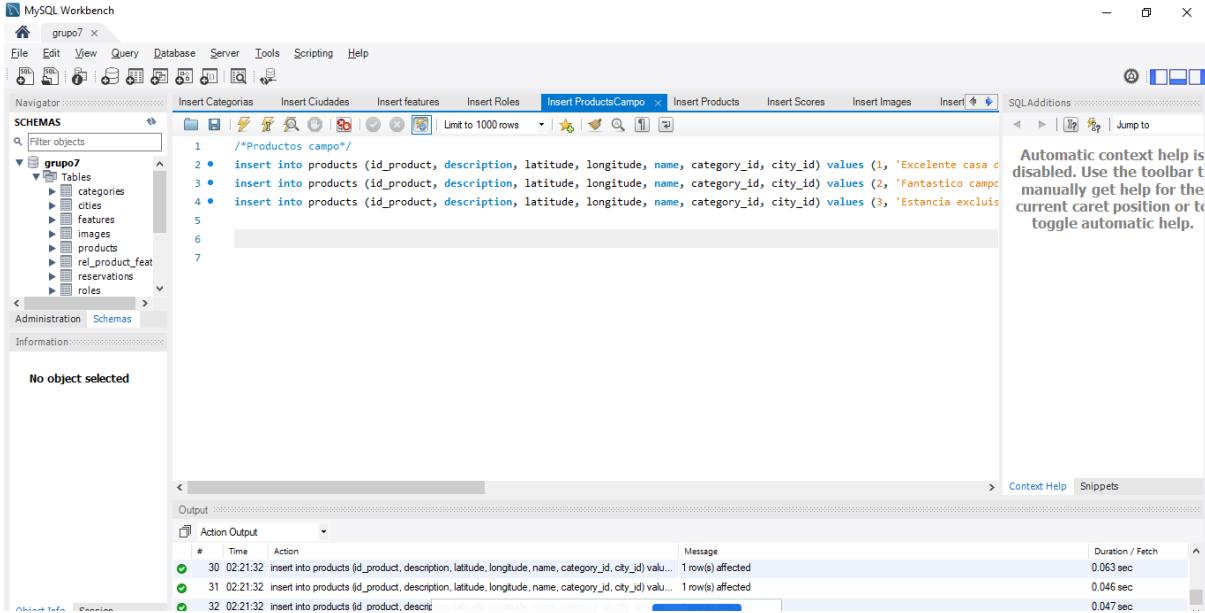
The screenshot shows the MySQL Workbench interface with the 'grupo7' schema selected. In the central pane, a SQL script titled 'Insert Ciudades' is being run. The script contains several `INSERT INTO cities VALUES` statements. The results grid shows the inserted data:

	id_city	country	name
1	Argentina	Buenos Aires	
2	Argentina	Rosario	
3	Argentina	Córdoba	
4	Argentina	San Juan	
5	Argentina	Mendoza	
6	Argentina	Bariloche	
7	Argentina	Villa La Angostura	
8	Argentina	Neuquén	
9	Argentina	San Miguel de Tucumán	

The output pane shows the execution log with two successful insertions:

#	Time	Action	Message	Duration / Fetch
27	02-21-08	INSERT INTO cities VALUES (27, 'Colombia', 'Ibagué')	1 row(s) affected	0.063 sec
28	02-21-08	INSERT INTO cities VALUES (28, 'Colombia', 'Villavicencio')	1 row(s) affected	0.062 sec

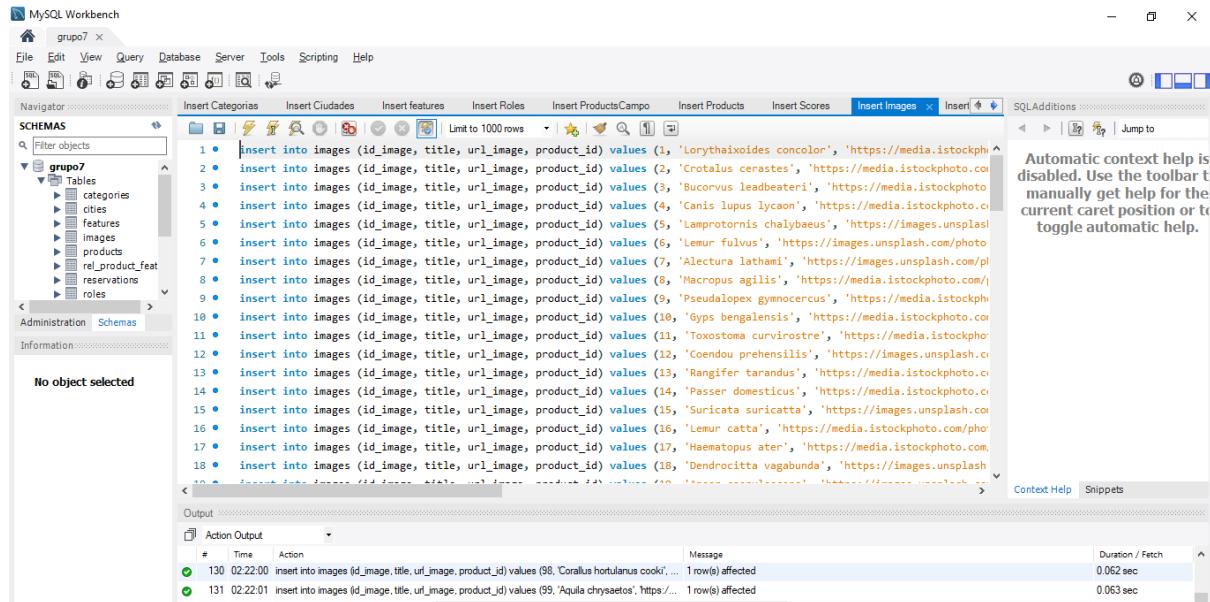
- a) En la imagen se ven los registros de las ciudades que se cargaron, puntuizando que las tablas se crean automáticamente cuando el proyecto del backend se corre, puesto que está hecha la conexión de la base de datos RDS a nuestro proyecto desde el application.properties



The screenshot shows the MySQL Workbench interface with the 'grupo7' schema selected. In the central pane, a SQL script titled 'Insert Products' is being run. The script contains several `insert into products` statements. The output pane shows the execution log with three successful insertions:

#	Time	Action	Message	Duration / Fetch
30	02-21-32	insert into products (id_product, description, latitude, longitude, name, category_id, city_id) values (1, 'Excelente casa c...	1 row(s) affected	0.063 sec
31	02-21-32	insert into products (id_product, description, latitude, longitude, name, category_id, city_id) values (2, 'Fantastico campo...	1 row(s) affected	0.046 sec
32	02-21-32	insert into products (id_product, description, latitude, longitude, name, category_id, city_id) values (3, 'Estancia exclusi...	1 row(s) affected	0.047 sec

- b) En la anterior los productos de campo por ej.



c) En ésta las imágenes de los productos.

Así, conforme lo requerido se puede acceder a la base de datos en AWS a través de Workbench y se crean las correspondientes tablas y relaciones.

4) Añadir imágenes en AWS Bucket para generar la URL

Nombre	Tipo	Última modificación	Tamaño	Clase de almacenamiento
Producto3/	Carpetas	-	-	-
Producto2/	Carpetas	-	-	-
Producto1/	Carpetas	-	-	-

Desde la consola de AWS se ingresó al bucket creado en el SPRINT anterior, se agregaron imágenes, se concedió acceso público a las mismas y su url se usó para hacer el registro en la base de datos.

5) OPCIONAL: deploy automatizado

En éste sprint como opcional se daba la posibilidad de realizar el deploy automatizado ofrecido por Gitlab, pero se descubrió que era necesario el uso de tarjetas de crédito y después de una reunión con el TL se aconsejó no seguir por ese camino, por lo que la pipeline fracasó y sigue figurando de esa forma en nuestro GitLab.

Otra forma que se intentó concretar para hacer el deploy automatizado fue, por el lado de front, la utilización de AWS amplify, que se concretó exitosamente.

The screenshot shows the AWS Amplify console interface. On the left, a sidebar lists 'App settings' with options like General, Admin UI management, Domain management, Build settings, Previews, Notifications, Environment variables, Access control, Monitoring, Rewrites and redirects, Custom headers, Documentation, and Support. The main content area shows the app homepage for 'camada-1/grupo-7'. It includes a 'Learn how to get the most out of Amplify Console' card (1 of 5 steps complete). Below it are tabs for 'Frontend environments' and 'Backend environments'. A section titled 'main' shows 'Continuous deploys set up (Edit)'. It features a preview window showing a browser with the URL <https://main.amplifyapp.com>. To the right is a CI/CD pipeline diagram with four stages: Provision, Build, Deploy, and Verify, all marked with green checkmarks. Below the pipeline are details: Last deployment (11/18/2021, 6:50:00 PM), Last commit (Autobuild | GitLab - main), and Previews (Disabled).

This screenshot shows the 'Edit target backend' dialog box overlaid on the AWS Amplify main dashboard. The dialog box has fields for 'App name' (set to 'camada-1/grupo-7 (this app)'), 'Environment' (set to 'Choose an existing environment or create a new one'), and a checked checkbox for 'Enable full-stack continuous deployments (CI/CD)'. Below the dialog are standard dashboard elements: a preview of the app homepage at <https://main.amplifyapp.com>, deployment and commit details, and a 'Verify' button.

El link de la página: <https://main.d2qu09af11fxgh.amplifyapp.com/>

En cambio, para hacer el deploy de back, se utilizó el archivo dockerfile creado en la carpeta raíz del back, luego desde la terminal se creó un contenedor con el comando:

`docker build --tag "brendanolan/grupo7"`.

Con este comando construimos la imagen del contenedor, le asignamos un tag para saber cómo está referenciado en nuestro repositorio de dockerhub, seguido de nombre de usuario y el nombre del contenedor. el punto dice que voy a encontrar un archivo en la raíz desde donde lo estoy llamando.

```

MINGW64:/d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/ProyectoIntegrador/grupo-7/BackEnd/grupo7
$ sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 18.87MB / 186.80MB 5.9s
$ sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 28.31MB / 186.80MB 7.5s
$ sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 37.75MB / 186.80MB 9.2s
$ ...

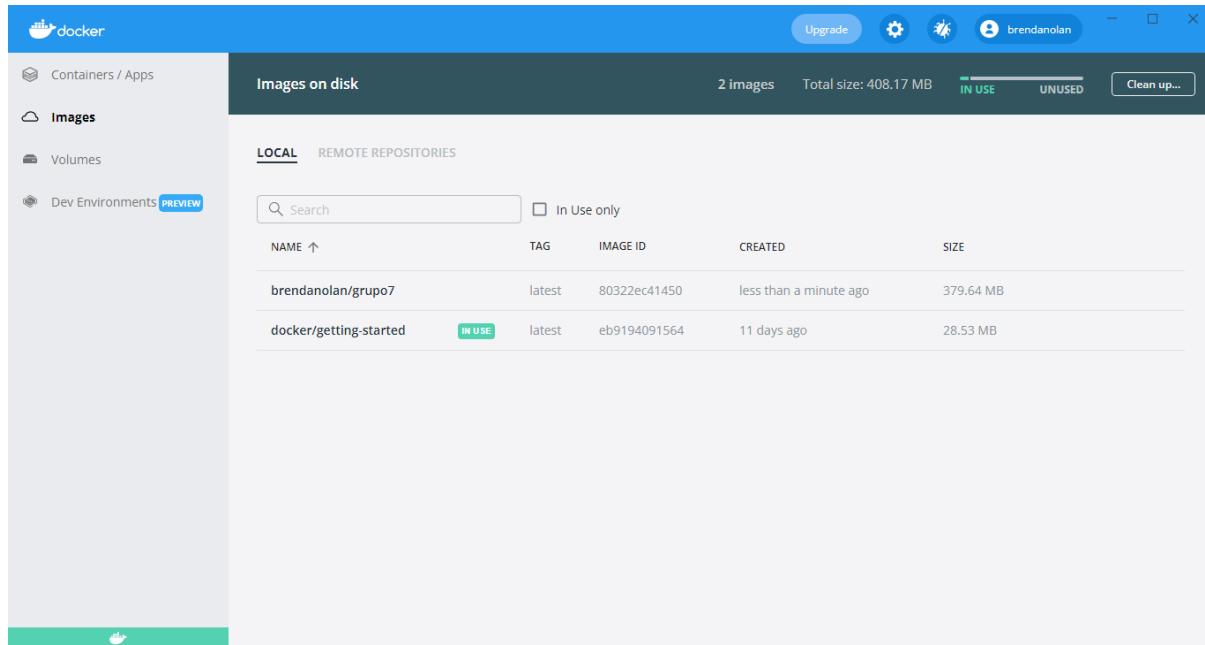
#6 [internal] Load build context
#6 sha256:3a742e3e213ba5c86c23b990b05334b3fcc83ee2026cda79d73333f98a9d87f2
#6 transferring context: 54.06MB 9.5s done
#6 DONE 9.5s

#5 [2/2] FROM docker.io/library/openjdk:17-alpine@sha256:4b6abae565492dbe9e7a894137c966a7485154238902f2f25e9dbd9784383d81
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 28.23MB / 186.80MB 10.6s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 58.72MB / 186.80MB 12.1s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 68.16MB / 186.80MB 13.4s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 87.53MB / 186.80MB 17.7s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 87.53MB / 186.80MB 16.0s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 96.47MB / 186.80MB 18.1s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 105.91MB / 186.80MB 20.3s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 115.34MB / 186.80MB 22.8s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 124.78MB / 186.80MB 24.8s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 134.22MB / 186.80MB 26.6s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 143.65MB / 186.80MB 28.5s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 160.53MB / 186.80MB 32.6s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 171.97MB / 186.80MB 37.9s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 181.40MB / 186.80MB 39.9s
#5 sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 186.80MB / 186.80MB 41.0s done
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 5.1s
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 10.2s
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 15.3s
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 20.3s
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 27.3s
#5 extracting sha256:dd8d15783b80cab15f5bf9726bcadas57651c624b64ca2bb4f42f94998d4662 32.2s done
#5 DONE 3.4s

#8 exporting to image
#8 sha256:e8c131e07b0b7ff33893b694f7759a10d42e180f2b4dc349fb57dc6b71dcab00
#8 exporting layers
#8 exporting layers 1.1s done
#8 writing image sha256:80322ec41450f9e38159d54f4fc8ac5ac4878920e3ae1e02d24f98daaa7ca9b 0.0s done
#8 naming to docker.io/brendanolan/grupo7
#8 naming to docker.io/brendanolan/grupo7 0.0s done
#8 DONE 1.2s

```

Así, vemos que está hecho el contenedor:



Así, se realizó un push del contenedor a nuestro repositorio creado en la cuenta de docker hub bajo el nombre grupo7:

docker push brendanolan/grupo7

```
MINGW64:/d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/ProyectoIntegrador/grupo-7/BackEnd/grupo7
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 105.91MB / 186.80MB 20.3s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 115.34MB / 186.80MB 22.8s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 124.78MB / 186.80MB 24.8s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 134.22MB / 186.80MB 26.6s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 143.66MB / 186.80MB 28.5s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 162.53MB / 186.80MB 35.1s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 171.97MB / 186.80MB 37.9s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 181.40MB / 186.80MB 39.9s
$ sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 186.80MB / 186.80MB 41.0s done
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 5.1s
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 10.7s
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 15.3s
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 20.3s
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 27.3s
$ extracting sha256:dd7d15783b0cab15f5bf9726bcadas5265c1624b64ca2bb4f642f94998d4662 32.2s done
$ DONE 3.4s

#8 exporting to image
#8 sha256:e5c613e07b0b7ff33893b694f759ax10d42e180f2b4dc349fb57dc6b71dcab00
#8 exporting layers
#8 exporting layers 1.s
#8 writing image sha256:80322ec41450f9e38159d54f4cf8cac5ac4878920e3ae1e02d24f98daaa7ca9b 0.0s done
#8 naming to docker.io/brendanolan/grupo7
#8 naming to docker.io/brendanolan/grupo7 0.0s done
#8 DONE 1.2s

Use `docker scan` to run Snyk tests against images to find vulnerabilities and learn how to fix them
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/ProyectoIntegrador/grupo-7/BackEnd/grupo7 (main)
$ docker push brendanolan/grupo7
Using default tag: latest
The push refers to a repository [docker.io/brendanolan/grupo7]
c37d319df22: Preparing
34f7384834b2: Preparing
5836cecc05bfbd: Preparing
72e630a4dff5: Preparing
5836cecc05bfbd: Mounted from library/openjdk
34f7384834b2: Mounted from library/openjdk
72e630a4dff5: Mounted from library/openjdk
c37d319df227: Pushed
latest: digest: sha256:112cd6dfaf3447696677417533e1005dfbec7eef15f4b205447b002de4e5989b size: 1163
Brenda@Brenda-PC MINGW64 /d/Users/Brenda/Desktop/Digital_House/ProyectoIntegrador/ProyectoIntegrador/grupo-7/BackEnd/grupo7 (main)
$
```

al no asignar tag queda la última versión, “latest”

que se ve reflejado acá:

The screenshot shows the Docker Hub repository page for 'brendanolan/grupo7'. At the top, there's a message about Advanced Image Management. Below it, the repository name 'brendanolan / grupo7' is displayed, along with a note that it 'This repository does not have a description'. To the right, there's a 'Docker commands' section with a 'Push' button. Under 'Tags and Scans', it shows 'latest' as the current tag, pushed a minute ago. In the 'Automated Builds' section, there's a note about manually pushing images to Hub and connecting to GitHub or Bitbucket for automated builds.

Por último, se creó un archivo docker run que va a ser utilizado en el entorno de elasticbeanstalk que estamos creando

Ya se admite AWS Graviton
AWS Graviton, un procesador basado en Arm64, puede ofrecer un rendimiento en relación con el precio hasta un 40 % mejor en comparación con el procesador x86 de características similares. Para actualizar el tipo de instancia a Arm64, elija esa opción en el ajuste "Capacidad" en "Configuración adicional".

Elastic Beanstalk > Entornos > Grupo7-env-1

Creando Grupo7-env-1
Este proceso tardará unos minutos. ...

12:37am Using elasticbeanstalk-sa-east-1-604110091184 as Amazon S3 storage bucket for environment data.
12:37am createEnvironment is starting.

¿Qué sucede al tener que actualizar el archivo .jar?

Como primera medida para que no se produzca un error no debe haber registros en la base de datos y desde ahí volver a ejecutar el clean y el package desde maven. Deberá crearse nuevamente el archivo .jar. y volver a hacer el build desde la raíz del back del proyecto. Se chequea en docker images si se actualizó el contenedor y se vuelve a hacer push al repositorio.

Así el entorno creado desde EB quedó exitosamente levantado.

sa-east-1.console.aws.amazon.com/elasticbeanstalk/home?region=sa-east-1#/environment/dashboard?applicationName=grupo7&environmentId=e-wm2h8h4xpp

Grupo7-env-1.eba-wqpmqybx.sa-east-1.elasticbeanstalk.com (e-wm2h8h4xpp)
Nombre de la aplicación: grupo7

Elastic Beanstalk

Actualizar Acciones

Estado: Aceptar

Versión en ejecución: grupo7-source-1

Cargar e implementar

Plataforma: Docker running on 64bit Amazon Linux 2/3.4.9

Eventos recientes:

Hora	Tipo	Detalles
24-11-2021 21:59:32 UTC-0300	INFO	Successfully rebuilt environment: Grupo7-env-1
24-11-2021 21:59:32 UTC-0300	INFO	Application available at Grupo7-env-1.eba-wqpmqybx.sa-east-1.elasticbeanstalk.com.

El deploy automatizado aún no funciona correctamente, esperamos tenerlo listo próximamente.



Copyright © 2021 Como En Casa - Todos los derechos reservados