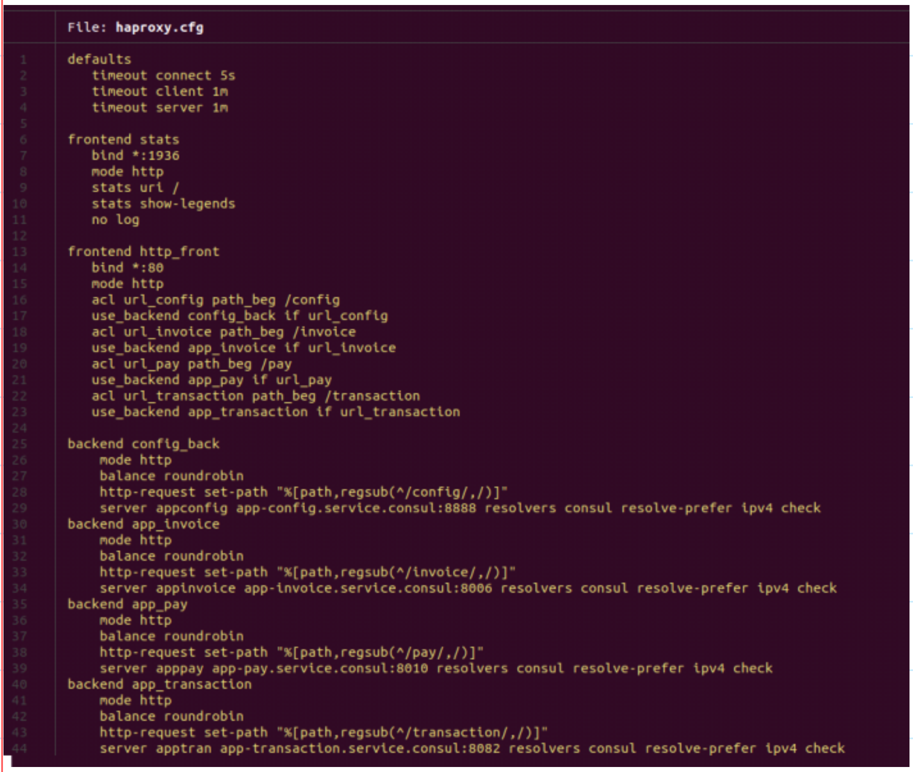
Workshop - 5

miércoles, febrero 28, 2024 3:21 PM

Name: Daniel Ramirez Gomez

This workshop is a continuation of the previous one in which each microservice was registered with a name in the DNS to change the way these services are queried. After that, in this workshop, we will proceed to use the DNS names so that the load balancer can communicate with these services. Finally, the API gateway will be created, and all services will be registered in the API so that they can communicate from the API gateway to the load balancer, and ultimately, with each of the requested services.

Load Balancer Configuration
We will use haproxy, which serves both as a reverse proxy and as a load balancer. In the haproxy folder, you will find the configuration file specifying basic settings such as the URL where the stats of the microservices will be displayed, the communication of haproxy with the microservices using their registered names in the DNS, and other basic configurations.



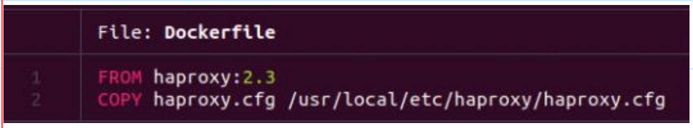
```
File: haproxy.cfg

defaults
   timeout connect 5s
   timeout client 1m
   timeout server 1m

frontend stats
   bind *:1936
   mode http
   stats uri /
   stats show-legends
   no log

frontend http_front
   bind *:80
   mode http
   acl url_config path_beg /config
   use_backend config_back if url_config
   acl url_invoice path_beg /invoice
   use_backend app_invoice if url_invoice
   acl url_pay path_beg /pay
   use_backend app_pay if url_pay
   acl url_transaction path_beg /transaction
   use_backend app_transaction if url_transaction

backend config_back
    mode http
    balance roundrobin
    http-request set-path "%[path,regsub(^/config/,/)]"
    server appconfig app-config.service.consul:8888 resolvers consul resolve-prefer ipv4 check
backend app_invoice
    mode http
    balance roundrobin
    http-request set-path "%[path,regsub(^/invoice/,/)]"
    server appinvoice app-invoice.service.consul:8006 resolvers consul resolve-prefer ipv4 check
backend app_pay
    mode http
    balance roundrobin
    http-request set-path "%[path,regsub(^/pay/,/)]"
    server apppay app-pay.service.consul:8010 resolvers consul resolve-prefer ipv4 check
backend app_transaction
    mode http
    balance roundrobin
    http-request set-path "%[path,regsub(^/transaction/,/)]"
    server apptran app-transaction.service.consul:8082 resolvers consul resolve-prefer ipv4 check
```

You can appreciate that in the backend of each microservice, the HTTP protocol is being used. The round-robin strategy is employed for the load balancer, the HTTP request path is modified before sending it to the backend server, and a backend server associated with the respective microservice is defined.

Now, this file should be copied into the container so that HAProxy can use these configurations. The command should be executed.

```
docker build -t ventana1901/loadbalancer:v1 .
```

Docker file



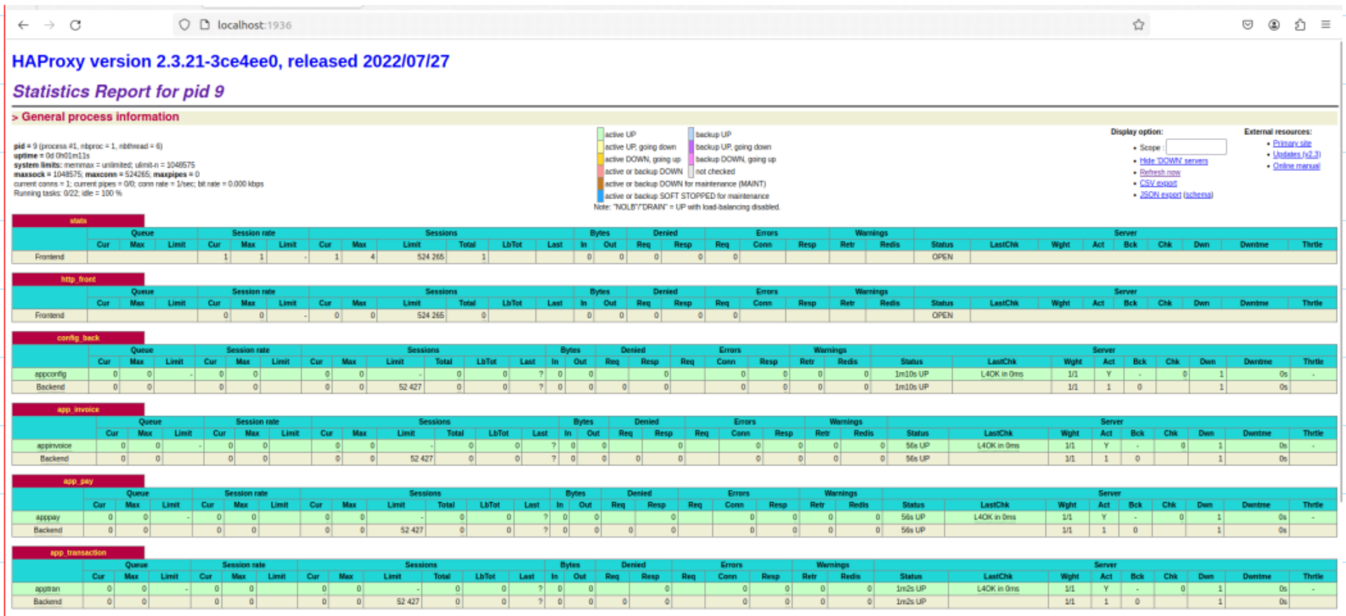
```
File: Dockerfile

FROM haproxy:2.3
COPY haproxy.cfg /usr/local/etc/haproxy/haproxy.cfg
```

Now, the container is executed with the command:

```
docker run -d -p 9000:80 -p 1936:1936 --network microservicenetwork --name loadbalancer ventana1901/loadbalancer:v1
```

Port 9000:80 is used, indicating that port 80 inside the container is mapped to port 9000 on the host. Port 1936:1936 is also used, indicating that port 1936 inside the container is mapped to port 1936 on the host, and it will be the IP to view the statistics provided by HAProxy.

If we access the microservice management page, we can observe the following.


localhost:1936

HAProxy version 2.3.21-3ce4ee0, released 2022/07/27

Statistics Report for pid 9

> General process information

pid = 9 (process #1, nbproc = 1, nbthread = 6)
uptime = 0d 0h01m11s
system limits: memmax = unlimited; ulimit-n = 1048575
maxsock = 1048575; **maxconn** = 524265; **maxpipes** = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec; bit rate = 0.000 kbps
Running tasks: 0/22; idle = 100 %

active UP	backup UP
active UP, going down	backup UP, going down
active DOWN, going up	backup DOWN, going up
active or backup DOWN	not checked
active or backup DOWN for maintenance (MAINT)	
active or backup SOFT STOPPED for maintenance	

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
- Scope :
- Hide 'DOWN' servers
- Refresh now
- CSV export
- JSON export (schema)

External resources:
- Primary site
- Updates (v2.3)
- Online manual

stats

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	1	-	1	4	524 265	1			0	0	0	0	0					OPEN								

http_front

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	0	-	0	0	524 265	0			0	0	0	0	0					OPEN								

config_back

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
appconfig	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	1m10s UP	L4OK in 0ms	1/1	Y	-	0	1	0s	-
Backend	0	0		0	0		0	0	52 427	0	0	?	0	0	0	0		0	0	0	0	1m10s UP		1/1	1	0		1	0s	

app_invoice

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
appinvoice	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	56s UP	L4OK in 0ms	1/1	Y	-	0	1	0s	-
Backend	0	0		0	0		0	0	52 427	0	0	?	0	0	0	0		0	0	0	0	56s UP		1/1	1	0		1	0s	

app_pay

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
apppay	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	56s UP	L4OK in 0ms	1/1	Y	-	0	1	0s	-
Backend	0	0		0	0		0	0	52 427	0	0	?	0	0	0	0		0	0	0	0	56s UP		1/1	1	0		1	0s	

app_transaction

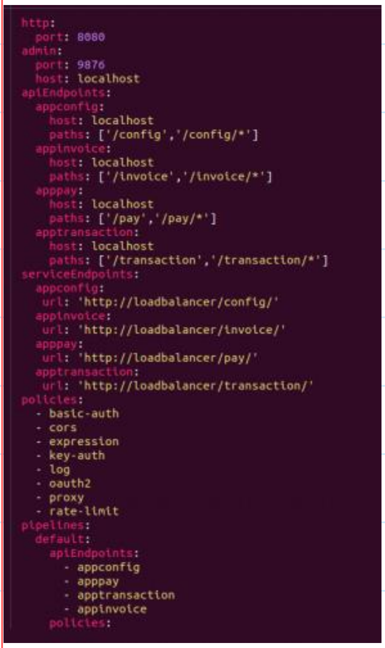
	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
apptran	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	1m2s UP	L4OK in 0ms	1/1	Y	-	0	1	0s	-
Backend	0	0		0	0		0	0	52 427	0	0	?	0	0	0	0		0	0	0	0	1m2s UP		1/1	1	0		1	0s	

Now, it remains only to configure the API gateway. For this purpose, a database in a container will be used, which will be Redis.

First, run the container with the command:

```
docker run --network microservicenetwork-d --name express-gateway-data-store -p 6379:6379
```

Now, proceed to configure the gateway.config file to set up the IP addresses that the API gateway can access for these services.



```
http:
  port: 8080
admin:
  port: 9876
  host: localhost
apiEndpoints:
  appconfig:
    host: localhost
    paths: ['/config','/config/*']
  appinvoice:
    host: localhost
    paths: ['/invoice','/invoice/*']
  apppay:
    host: localhost
    paths: ['/pay','/pay/*']
  apptransaction:
    host: localhost
    paths: ['/transaction','/transaction/*']
serviceEndpoints:
  appconfig:
    url: 'http://loadbalancer/config/'
  appinvoice:
    url: 'http://loadbalancer/invoice/'
  apppay:
    url: 'http://loadbalancer/pay/'
  apptransaction:
    url: 'http://loadbalancer/transaction/'
policies:
  - basic-auth
  - cors
  - expression
  - key-auth
  - log
  - oauth2
  - proxy
  - rate-limit
pipelines:
  default:
    apiEndpoints:
      - appconfig
      - apppay
      - apptransaction
      - appinvoice
    policies:
```

This file has the basic configurations for the API gateway, and what is noteworthy here is:

-Everything will be communicated through port 8080.
-"localhost" will be used as the host.
-API endpoints are also defined, specifying how requests to those endpoints should be handled. They are named based on the microservices to be used, in this case, config, invoice, etc.

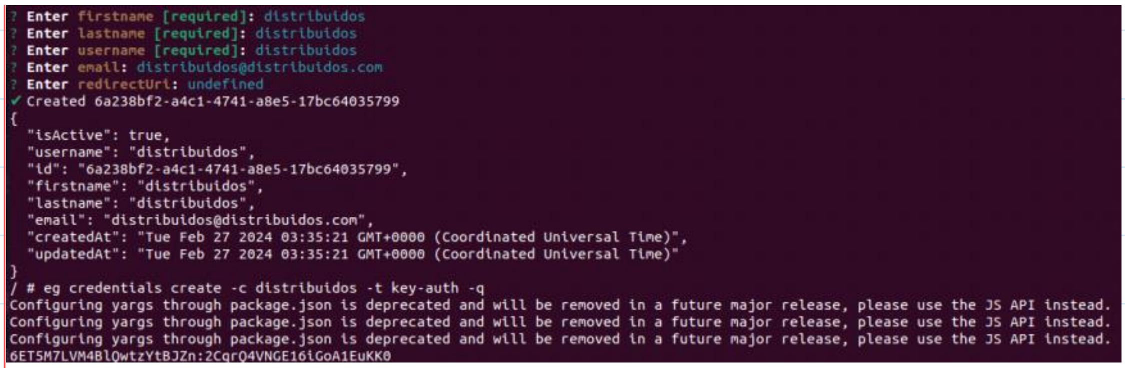
After that, the container containing the API gateway itself can be executed. It is run with the following command:

```
docker run -d --name express-gateway --network microservicenetwork -v $(pwd):/var/lib/eg -p 8080:8080 -p 9876:9876 express-gateway
```

It is desired to copy said configuration file into the container.
Now, a user must be created so that they can access the API gateway service. To do this, we access the container we just ran, the express-gateway.
Access is obtained with the following command:

```
docker exec -it express-gateway sh
```

In the container, the credentials are configured to our liking.



```
? Enter firstname [required]: distribuidos
? Enter lastname [required]: distribuidos
? Enter username [required]: distribuidos
? Enter email: distribuidos@distribuidos.com
? Enter redirectUri: undefined
✓ Created 6a238bf2-a4c1-4741-a8e5-17bc64035799
{
  "isActive": true,
  "username": "distribuidos",
  "id": "6a238bf2-a4c1-4741-a8e5-17bc64035799",
  "firstname": "distribuidos",
  "lastname": "distribuidos",
  "email": "distribuidos@distribuidos.com",
  "createdAt": "Tue Feb 27 2024 03:35:21 GMT+0000 (Coordinated Universal Time)",
  "updatedAt": "Tue Feb 27 2024 03:35:21 GMT+0000 (Coordinated Universal Time)"
}
/ # eg credentials create -c distribuidos -t key-auth -q
Configuring yargs through package.json is deprecated and will be removed in a future major release, please use the JS API instead.
Configuring yargs through package.json is deprecated and will be removed in a future major release, please use the JS API instead.
Configuring yargs through package.json is deprecated and will be removed in a future major release, please use the JS API instead.
6ET5M7LVM4BlQwtzYtBJZn:2CqrQ4VNGE16iGoA1EuKK0
```

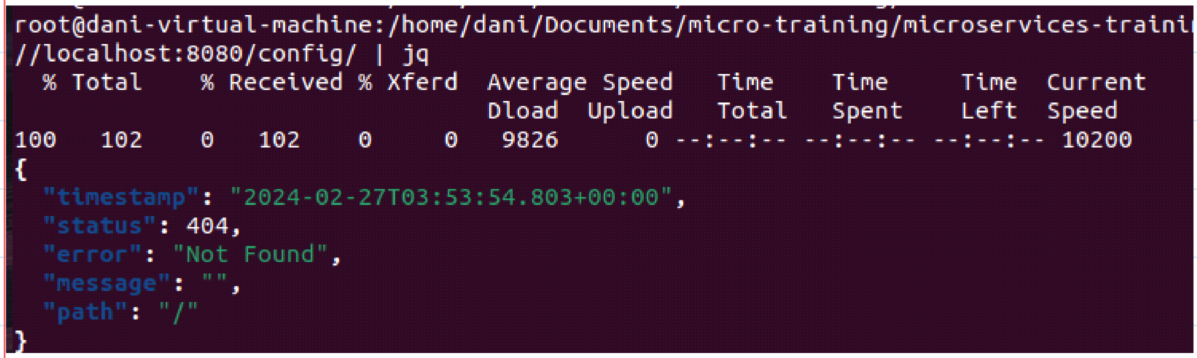
This will ultimately provide us with a token with which we can access the API.

Finally, to access the APIs, the following command is used with the included token.

```
curl -H "Authorization: apikey 6ET5M7LVM4B1QwtzYtBJZn:2CqrQ4VNGE16iGoA1EUKK0" http: //localhost:8080/transaction/ | jq
```

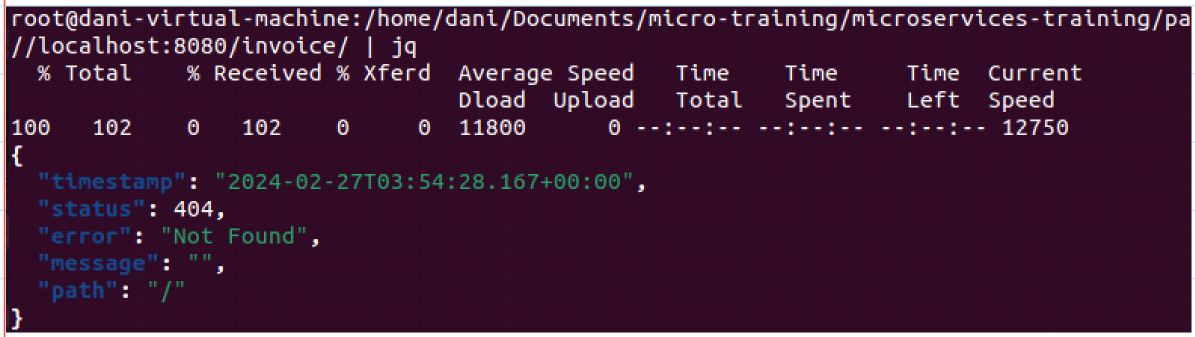
The operation can be evidenced with each of the microservices.

Microservicio config



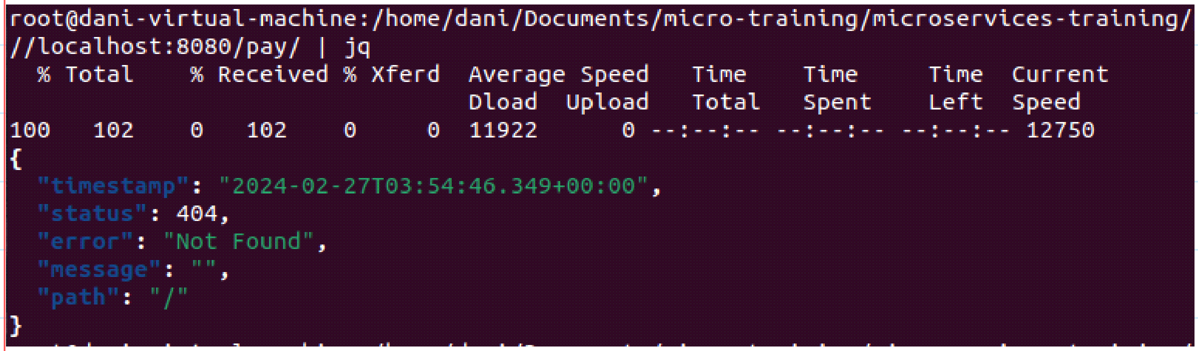
```
root@dani-virtual-machine:/home/dani/Documents/micro-training/microservices-traini
//localhost:8080/config/ | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   102    0   102    0     0   9826      0 --:--:-- --:--:-- --:--:-- 10200
{
  "timestamp": "2024-02-27T03:53:54.803+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "",
  "path": "/"
}
```

Microservicio invoice



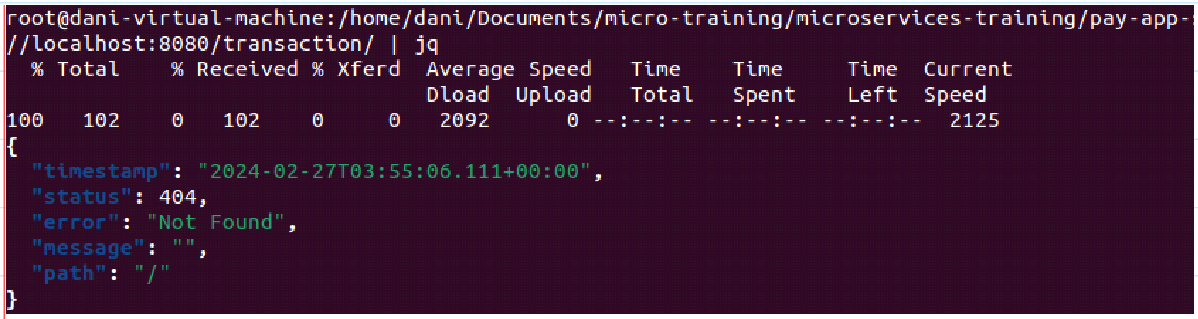
```
root@dani-virtual-machine:/home/dani/Documents/micro-training/microservices-training/pa
//localhost:8080/invoice/ | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   102    0   102    0     0  11800      0 --:--:-- --:--:-- --:--:-- 12750
{
  "timestamp": "2024-02-27T03:54:28.167+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "",
  "path": "/"
}
```

Microservicio pay



```
root@dani-virtual-machine:/home/dani/Documents/micro-training/microservices-training/
//localhost:8080/pay/ | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   102    0   102    0     0  11922      0 --:--:-- --:--:-- --:--:-- 12750
{
  "timestamp": "2024-02-27T03:54:46.349+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "",
  "path": "/"
}
```

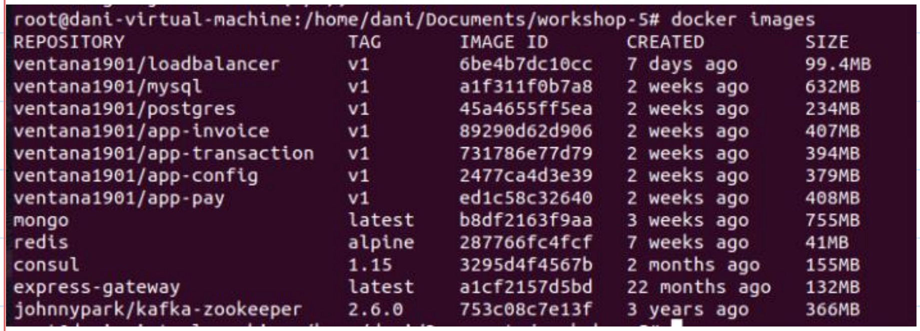
Microservicio transaction



```
root@dani-virtual-machine:/home/dani/Documents/micro-training/microservices-training/pay-app-
//localhost:8080/transaction/ | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   102    0   102    0     0   2092      0 --:--:-- --:--:-- --:--:--  2125
{
  "timestamp": "2024-02-27T03:55:06.111+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "",
  "path": "/"
}
```

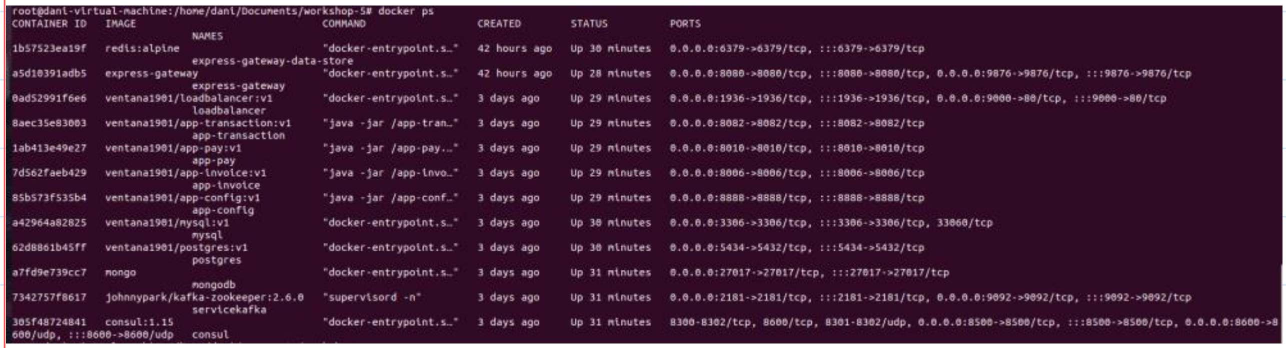
Anexos

Imágenes necesarias



```
root@dani-virtual-machine:/home/dani/Documents/workshop-5# docker images
REPOSITORY                      TAG       IMAGE ID       CREATED         SIZE
ventana1901/loadbalancer        v1        6be4b7dc10cc   7 days ago      99.4MB
ventana1901/mysql               v1        a1f311f0b7a8   2 weeks ago     632MB
ventana1901/postgres            v1        45a4655ff5ea   2 weeks ago     234MB
ventana1901/app-invoice         v1        89290d62d906   2 weeks ago     407MB
ventana1901/app-transaction     v1        731786e77d79   2 weeks ago     394MB
ventana1901/app-config          v1        2477ca4d3e39   2 weeks ago     379MB
ventana1901/app-pay             v1        ed1c58c32640   2 weeks ago     408MB
mongo                           latest    b8df2163f9aa   3 weeks ago     755MB
redis                           alpine    287766fc4fcf   7 weeks ago     41MB
consul                          1.15      3295d4f4567b   2 months ago    155MB
express-gateway                 latest    a1cf2157d5bd   22 months ago   132MB
johnnypark/kafka-zookeeper      2.6.0     753c08c7e13f   3 years ago     366MB
```

Contenedores necesarios



```
root@dani-virtual-machine:/home/dani/Documents/workshop-5# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS
                          NAMES
1b57523ea19f   redis:alpine                         "docker-entrypoint.s…"   42 hours ago     Up 30 minutes   0.0.0.0:6379->6379/tcp, :::6379->6379/tcp
                          express-gateway-data-store
a5d10391adb5   express-gateway                      "docker-entrypoint.s…"   42 hours ago     Up 28 minutes   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 0.0.0.0:9876->9876/tcp, :::9876->9876/tcp
                          express-gateway
0ad52991f6e6   ventana1901/loadbalancer:v1          "docker-entrypoint.s…"   3 days ago       Up 29 minutes   0.0.0.0:1936->1936/tcp, :::1936->1936/tcp, 0.0.0.0:9000->80/tcp, :::9000->80/tcp
                          loadbalancer
8aec35e83003   ventana1901/app-transaction:v1       "java -jar /app-tran…"   3 days ago       Up 29 minutes   0.0.0.0:8082->8082/tcp, :::8082->8082/tcp
                          app-transaction
1ab413e49e27   ventana1901/app-pay:v1               "java -jar /app-pay.…"   3 days ago       Up 29 minutes   0.0.0.0:8010->8010/tcp, :::8010->8010/tcp
                          app-pay
7d562faeb429   ventana1901/app-invoice:v1           "java -jar /app-invo…"   3 days ago       Up 29 minutes   0.0.0.0:8006->8006/tcp, :::8006->8006/tcp
                          app-invoice
85b573f535b4   ventana1901/app-config:v1            "java -jar /app-conf…"   3 days ago       Up 29 minutes   0.0.0.0:8888->8888/tcp, :::8888->8888/tcp
                          app-config
a42964a82825   ventana1901/mysql:v1                 "docker-entrypoint.s…"   3 days ago       Up 30 minutes   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp
                          mysql
62d8861b45ff   ventana1901/postgres:v1              "docker-entrypoint.s…"   3 days ago       Up 30 minutes   0.0.0.0:5434->5432/tcp, :::5434->5432/tcp
                          postgres
a7fd9e739cc7   mongo                                "docker-entrypoint.s…"   3 days ago       Up 31 minutes   0.0.0.0:27017->27017/tcp, :::27017->27017/tcp
                          mongodb
7342757f8617   johnnypark/kafka-zookeeper:2.6.0     "supervisord -n"         3 days ago       Up 31 minutes   0.0.0.0:2181->2181/tcp, :::2181->2181/tcp, 0.0.0.0:9092->9092/tcp, :::9092->9092/tcp
                          servicekafka
305f48724841   consul:1.15                          "docker-entrypoint.s…"   3 days ago       Up 31 minutes   8300-8302/tcp, 8600/tcp, 8301-8302/udp, 0.0.0.0:8500->8500/tcp, :::8500->8500/tcp, 0.0.0.0:8600->8
600/udp, :::8600->8600/udp   consul
```