

Trabajo Práctico N°1

Nahuel Defossé

2018

Taller de Nuevas Tecnologías

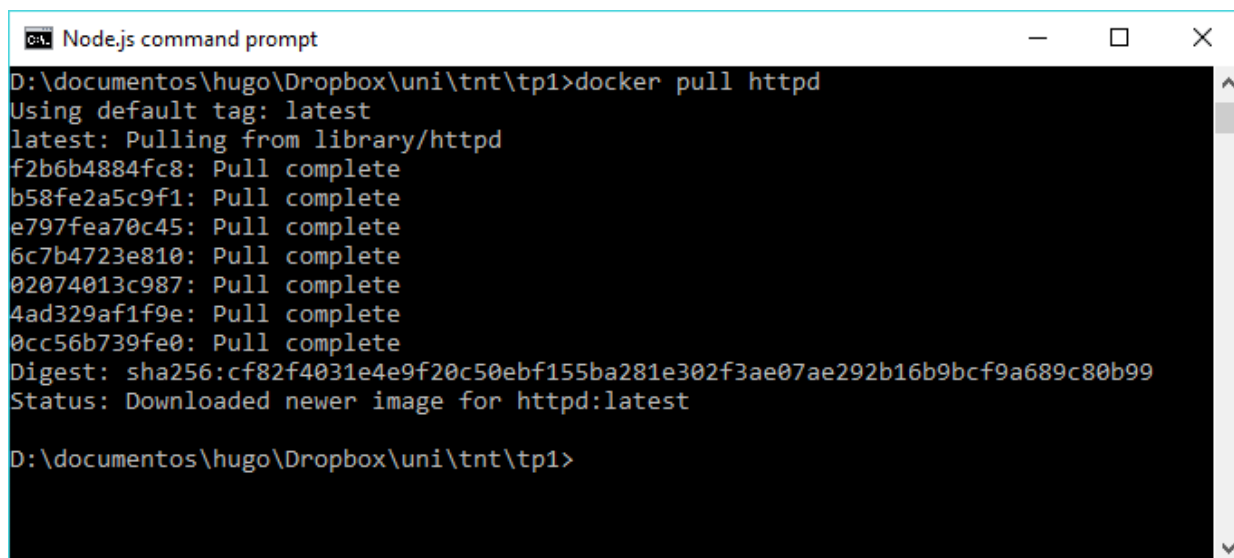
Notas

Los trabajos pueden realizarse en grupos de hasta dos personas. La fecha de entrega será pautada, en dos semanas a partir de la fecha en la cual esté disponible.

El formato de entrega puede ser PDF o HTML. Se deberá incluir evidencia del trabajo realizado. Los trabajos no aprobados tendrán un período de recuperación de una semana posterior a la fecha de recepción de las devoluciones.

Construcción y gestión de contenedores de Software

1. Descargue la imagen de Apache (httpd) desde Docker Hub y muestre la salida del comando `docker images`.



```
Node.js command prompt
D:\documentos\hugo\Dropbox\uni\tnt\tp1>docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
f2b6b4884fc8: Pull complete
b58fe2a5c9f1: Pull complete
e797fea70c45: Pull complete
6c7b4723e810: Pull complete
02074013c987: Pull complete
4ad329af1f9e: Pull complete
0cc56b739fe0: Pull complete
Digest: sha256:cf82f4031e4e9f20c50ebf155ba281e302f3ae07ae292b16b9bcf9a689c80b99
Status: Downloaded newer image for httpd:latest

D:\documentos\hugo\Dropbox\uni\tnt\tp1>
```

```

Node.js command prompt
D:\documentos\hugo\Dropbox\uni\tnt\tp1>docker images
REPOSITORY          TAG             IMAGE ID          CREATED           SIZE
httpd                latest          805130e51ae9      2 days ago       178MB
D:\documentos\hugo\Dropbox\uni\tnt\tp1>

```

2. Ejecute la imagen con política de reinicio always , exponiendo el puerto 80 al host anfitrión. Evidencie la ejecución mediante el comando uptime seguido de docker ps , reinicie el servidor o computadora dónde ejecuta el servicio docker y vuelva a ejecutar dichos comandos. ¿Qué puede inferir a partir de las columnas CREATED y STATUS?

```

Node.js command prompt
C:\Users\donat>docker rm apache
apache
C:\Users\donat>docker run -d --name apache -p 127.0.0.1:80:80 --restart always httpd
41ae62097150a700a2755cecefb7eab17d1082333f4d3dba6ab473cf9ff078c
C:\Users\donat>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS               NAMES
41ae62097150   httpd     "httpd-foreground"       6 seconds ago Up 5 seconds    127.0.0.1:80->80/tcp apache
C:\Users\donat>

```

```

Node.js command prompt
wait      Block until one or more containers stop, then print their exit codes
Run 'docker COMMAND --help' for more information on a command.
C:\Users\donat>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS               NAMES
41ae62097150   httpd     "httpd-foreground"       5 minutes ago Up 2 minutes    127.0.0.1:80->80/tcp apache
C:\Users\donat>

```

CREATED es el tiempo transcurrido desde la creación del contenedor, activo y/o inactivo. STATUS es el tiempo transcurrido de la ejecución actual.

3. Inspeccione la configuración mediante docker exec -ti <contenedor> sh y ubique la configuración de la raíz de documentos. Suplante este contenido mediante un volumen. Evidencie los cambios. ¿Qué diferencia existe si un volumen está creado con una ruta absoluta vs. cuando se crea con una cadena? ¿Para qué existe el comando docker volume? Relaciónelo con la pregunta anterior. ¿Qué ciclo de vida tiene cada tipo de volumen?

```

Node.js command prompt - docker exec -ti apache sh

C:\Users\donat>docker run -d --name apache -p 127.0.0.1:80:80 --restart always httpd
6c43a964446d957bbc7f906a2070c935d4bab1c01beda6937cf500dab8abba73

C:\Users\donat>docker exec -ti apache sh
# find -name htdocs
./htdocs
# find / -name htdocs
/usr/local/apache2/htdocs
#

```

```

Node.js command prompt

D:\documentos\hugo\git\TNT-PM\Práctica>md tp1
D:\documentos\hugo\git\TNT-PM\Práctica>cd tp1
D:\documentos\hugo\git\TNT-PM\Práctica\tp1>dir
El volumen de la unidad D es clarooscuro
El número de serie del volumen es: 0ED7-AEC2

Directorio de D:\documentos\hugo\git\TNT-PM\Práctica\tp1
25/03/2018  15:02    <DIR>        .
25/03/2018  15:02    <DIR>        ..
                0 archivos            0 bytes
                2 dirs  219.420.942.336 bytes libres

D:\documentos\hugo\git\TNT-PM\Práctica\tp1>md htdocs && echo tp1 guacho > htdocs/index.html
D:\documentos\hugo\git\TNT-PM\Práctica\tp1>dir
El volumen de la unidad D es clarooscuro
El número de serie del volumen es: 0ED7-AEC2

Directorio de D:\documentos\hugo\git\TNT-PM\Práctica\tp1
25/03/2018  15:03    <DIR>        .
25/03/2018  15:03    <DIR>        ..
25/03/2018  15:03    <DIR>        htdocs
                0 archivos            0 bytes
                3 dirs  219.420.942.336 bytes libres

D:\documentos\hugo\git\TNT-PM\Práctica\tp1>cd htdocs
D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs>dir
El volumen de la unidad D es clarooscuro
El número de serie del volumen es: 0ED7-AEC2

Directorio de D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs
25/03/2018  15:03    <DIR>        .
25/03/2018  15:03    <DIR>        ..
25/03/2018  15:03    <DIR>        13 index.html
                1 archivos            13 bytes
                2 dirs  219.420.942.336 bytes libres

D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs>cat index.html
tp1 guacho

D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs>docker stop apache
apache

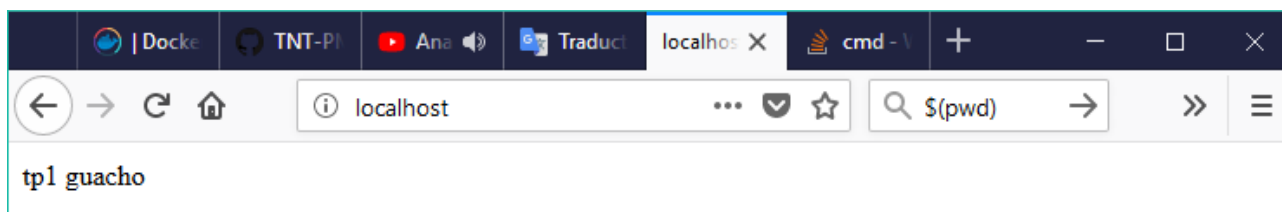
D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs>docker rm apache
apache

D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs>dir
El volumen de la unidad D es clarooscuro
El número de serie del volumen es: 0ED7-AEC2

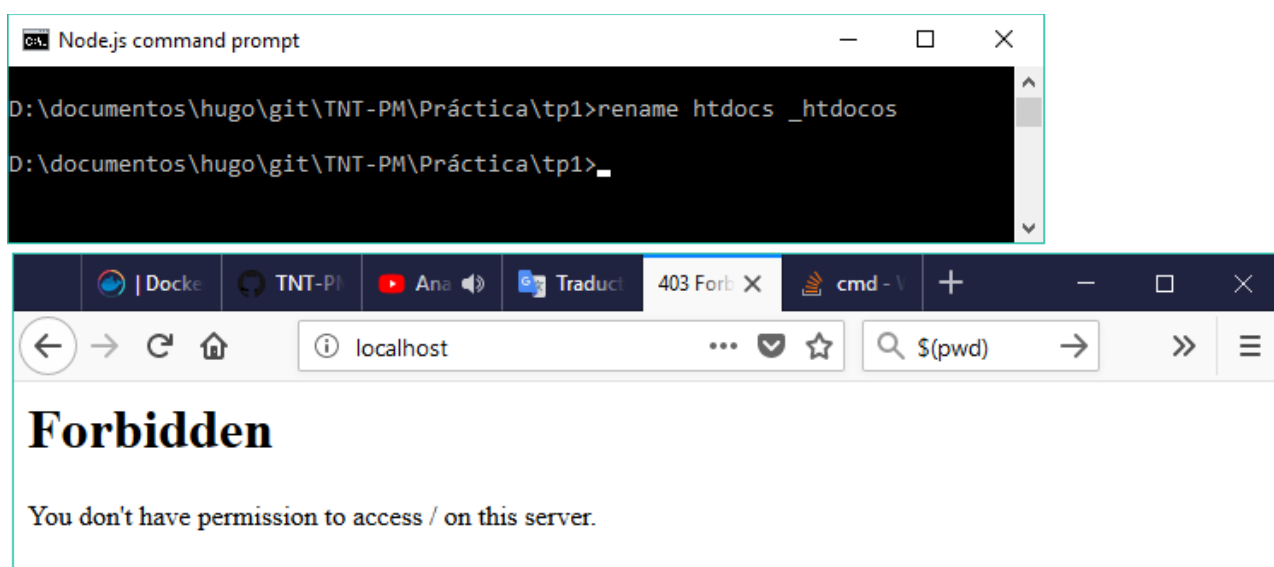
Directorio de D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs
25/03/2018  15:03    <DIR>        .
25/03/2018  15:03    <DIR>        ..
25/03/2018  15:03    <DIR>        13 index.html
                1 archivos            13 bytes
                2 dirs  219.420.942.336 bytes libres

D:\documentos\hugo\git\TNT-PM\Práctica\tp1\htdocs>cd..

```



Cuando la dirección es absoluta, si se cambia la dirección del volumen no podrá ser localizado por el contenedor.



El comando “docker volume” se usa para gestión de volúmenes. Con esta herramienta es posible crear volúmenes nombrados -o de cadena-.

El ciclo de vida de un volumen absoluto queda sujeto a la vida del contenedor. En cambio uno creado con una cadena, persiste mas allá de sus clientes.

4. Cree un contenedor a partir de una imagen (puede reutilizar el de los puntos previos) y ejecute ps dentro de este, registrando PIDs y nombres de procesos. Utilice docker exec para lanzar un nuevo shell. ¿Qué PID toma? ¿Qué sucede si desde este shell envía una señal 9 (SIGKILL) al PID 1?

El espacio de asignación de PID's es el mismo, y por lo tanto los id no se repiten entre shell's.

“kill SIGKILL 1” reinicia el servidor.

5. Genere un Dockerfile basado en la imagen descargada en el punto 1 que contenga un archivo y constrúyala mediante `docker build -t <nombre>`.

El archivo puede descargarse desde <https://github.com/poximan/TNT-PM/tree/master/Práctica/tp1>

6. Ejecute `docker history <nombre>` sobre la imagen generada en el punto anterior. ¿Qué puede inferir a partir de la salida?

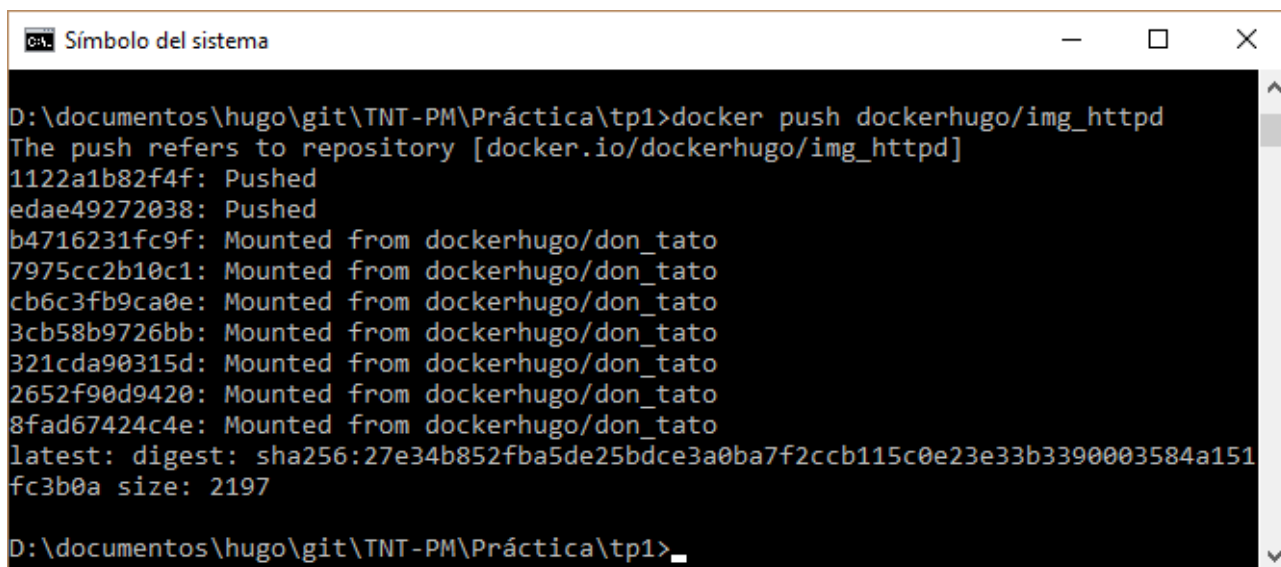
Que la nueva imagen es en realidad la imagen original mas una serie de operaciones de edición. Se parece a un sistema de control de versiones (como una implementación git o npm).

```

C:\Users\donat>docker history dockerhugo/don_tato
IMAGE          COMMENT          CREATED          CREATED BY          SIZE
3b5166054a49   6 minutes ago   /bin/sh -c #(nop) CMD ["/home/don_tato/mi_p...  0B
c7f501ef0667   6 minutes ago   /bin/sh -c #(nop) ADD dir:0780c911eed2d4665e...  178kB
adcbff953540    6 minutes ago   /bin/sh -c apt-get install -qq -y build-esse...  156MB
38d8ede23b63   12 minutes ago   /bin/sh -c apt-get update -qq                    21.1MB
785ae2563e94   18 minutes ago   /bin/sh -c #(nop) MAINTAINER <poxi_man@yaho...  0B
805130e51ae9    3 days ago      /bin/sh -c #(nop) CMD ["httpd-foreground"]      0B
<missing>       3 days ago      /bin/sh -c #(nop) EXPOSE 80/tcp                  0B
<missing>       3 days ago      /bin/sh -c #(nop) COPY file:761e313354b918b6...  133B
<missing>       3 days ago      /bin/sh -c set -eux; buildDeps=" bzip2 ...      9.93MB
<missing>       3 days ago      /bin/sh -c #(nop) ENV APACHE_DIST_URLS=http...  0B
<missing>       3 days ago      /bin/sh -c #(nop) ENV HTTPD_PATCHES=            0B
<missing>       3 days ago      /bin/sh -c #(nop) ENV HTTPD_SHA256=de025118...  0B
<missing>       3 days ago      /bin/sh -c #(nop) ENV HTTPD_VERSION=2.4.33      0B
<missing>       11 days ago     /bin/sh -c apt-get update && apt-get instal...  44.2MB
<missing>       11 days ago     /bin/sh -c { echo 'deb http://deb.debian.o...  161B
<missing>       11 days ago     /bin/sh -c #(nop) ENV OPENSSL_VERSION=1.0.2...  0B
<missing>       11 days ago     /bin/sh -c #(nop) ENV NGHTTP2_VERSION=1.18...  0B
<missing>       11 days ago     /bin/sh -c #(nop) WORKDIR /usr/local/apache2    0B
<missing>       11 days ago     /bin/sh -c mkdir -p "$HTTPD_PREFIX" && chow...  0B
<missing>       11 days ago     /bin/sh -c #(nop) ENV PATH=/usr/local/apach...  0B
<missing>       11 days ago     /bin/sh -c #(nop) ENV HTTPD_PREFIX=/usr/loc...  0B
<missing>       12 days ago     /bin/sh -c echo 'deb http://deb.debian.org/d...  55B
<missing>       12 days ago     /bin/sh -c #(nop) CMD ["bash"]                  0B
<missing>       12 days ago     /bin/sh -c #(nop) ADD file:bc844c4763367b5f0...  123MB
  
```

7. Empuje la imagen del punto anterior a <https://hub.docker.com/> mediante `docker push` etiqueta.

https://hub.docker.com/r/dockerhugo/img_httpd/



```

D:\documentos\hugo\git\TNT-PM\Práctica\tp1>docker push dockerhugo/img_httpd
The push refers to repository [docker.io/dockerhugo/img_httpd]
1122a1b82f4f: Pushed
edae49272038: Pushed
b4716231fc9f: Mounted from dockerhugo/don_tato
7975cc2b10c1: Mounted from dockerhugo/don_tato
cb6c3fb9ca0e: Mounted from dockerhugo/don_tato
3cb58b9726bb: Mounted from dockerhugo/don_tato
321cda90315d: Mounted from dockerhugo/don_tato
2652f90d9420: Mounted from dockerhugo/don_tato
8fad67424c4e: Mounted from dockerhugo/don_tato
latest: digest: sha256:27e34b852fba5de25bdce3a0ba7f2ccb115c0e23e33b3390003584a151fc3b0a size: 2197
D:\documentos\hugo\git\TNT-PM\Práctica\tp1>_

```

8. Aprovechone una máquina virtual o real (con acceso ssh configurado con clave pública) mediante el comando:

```

docker-machine create \

--driver generic \

--generic-ip-address=<ip o nombre> \

--generic-ssh-key ~/.ssh/id_rsa \

<nombre>

```

1. Ejecute docker-machine ls
 2. Evalúe en el shell el ambiente para dicha máquina y ejecute docker ps.
9. Cree un proyecto nodejs y cree su correspondiente Dockerfile . ¿Cómo organiza los comandos dentro del Dockerfile ? ¿Qué interacción hay entre COPY y RUN en cuanto a caché? ¿Por qué cree que es común utilizar COPY <fuente> <destino> en conjunto con un volumen en <destino>?

Los comando se organizan según el patrón propuesto en <https://docs.docker.com/engine/reference/builder/#environment-replacement>. Es decir, primero definir la *imagen:version* que se tomará de base. Luego crear la estructura de carpetas en destino (si fuera necesario), copiar los fuentes y ejecutar en destino el comando deseado.

La interacción se basa en que RUN (siempre que sea ejecutado luego de COPY) verá el sistema de archivos tal como quedó luego de las modificaciones introducidas por COPY.

10. Agregue una imagen de mongodb mediante la creación de un archivo docker-compose.yml

```

C:\Simbolo del sistema - docker-compose up --build
D:\documentos\hugo\git\TNT-PM\Práctica\tp1\pto10>docker-compose up --build
Building web
Step 1/7 : FROM node:alpine
----> 5d68146e371d
Step 2/7 : MAINTAINER <poxi_man@yahoo.com>
----> Using cache
----> 4390c0e41d88
Step 3/7 : COPY ./pto10.js /home/src/
----> c5255f0177ff
Step 4/7 : COPY ./package.json /home/src/
----> 79246f4bf2d7
Step 5/7 : WORKDIR /home/src
Removing intermediate container dccbc6bacd10
----> 5fe148830836
Step 6/7 : RUN npm install
----> Running in 2f9a8de449ad
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN pto10@1.0.0 No repository field.

added 56 packages in 9.167s
Removing intermediate container 2f9a8de449ad
----> 7995a555ca55
Step 7/7 : CMD ["node", "pto10.js"]
----> Running in 4360cbb4f04d
Removing intermediate container 4360cbb4f04d
----> 01e53da16228
Successfully built 01e53da16228
Successfully tagged pto10_web:latest
Starting pto10_mongo_1 ... done
Recreating pto10_web_1 ... done
Attaching to pto10_mongo_1, pto10_web_1
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] MongoDB starting : pid=1 port=27017 dbpath=/data/db 64-bit host=d2a47e
c8d0bd
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] db version v3.6.3
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] git version: 9586e57d54ef70f9ca4b43c26892cd55257e1a5
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.1t 3 May 2016
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] allocator: tcmalloc
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] modules: none
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] build environment:
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] distmod: debian81

```

```

C:\Simbolo del sistema - docker-compose up --build
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] build environment:
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] distmod: debian81
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] distarch: x86_64
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] target_arch: x86_64
web_1 | Example app listening on port 3001!
mongo_1 | 2018-04-02T19:35:52.713+0000 I CONTROL [initandlisten] options: { net: { bindIpAll: true } }
mongo_1 | 2018-04-02T19:35:52.782+0000 I - [initandlisten] Detected data files in /data/db created by the 'wiredTiger' storage en
gine, so setting the active storage engine to 'wiredTiger'.
mongo_1 | 2018-04-02T19:35:52.783+0000 I STORAGE [initandlisten]
mongo_1 | 2018-04-02T19:35:52.783+0000 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the
WiredTiger storage engine
mongo_1 | 2018-04-02T19:35:52.783+0000 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
mongo_1 | 2018-04-02T19:35:52.784+0000 I STORAGE [initandlisten] wiredtiger_open config: create,cache size=478M,session_max=20000,evict
ion=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file
_manager=(close_idle_time=100000),statistics_log=(wait=0),verbose=(recovery_progress),
mongo_1 | 2018-04-02T19:35:54.070+0000 I STORAGE [initandlisten] WiredTiger message [1522697754:70695][1:0x7fb366aaaa00], txn-recover:
Main recovery loop: starting at 2/16256
mongo_1 | 2018-04-02T19:35:54.158+0000 I STORAGE [initandlisten] WiredTiger message [1522697754:158476][1:0x7fb366aaaa00], txn-recover:
Recovering log 2 through 3
mongo_1 | 2018-04-02T19:35:54.471+0000 I STORAGE [initandlisten] WiredTiger message [1522697754:471657][1:0x7fb366aaaa00], txn-recover:
Recovering log 3 through 3
mongo_1 | 2018-04-02T19:35:56.827+0000 I CONTROL [initandlisten]
mongo_1 | 2018-04-02T19:35:56.827+0000 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
mongo_1 | 2018-04-02T19:35:56.827+0000 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestr
icted.
mongo_1 | 2018-04-02T19:35:56.827+0000 I CONTROL [initandlisten]
mongo_1 | 2018-04-02T19:35:56.847+0000 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory '/data/d
b/diagnostic_data'
mongo_1 | 2018-04-02T19:35:56.848+0000 I NETWORK [initandlisten] waiting for connections on port 27017
mongo_1 | 2018-04-02T19:35:58.476+0000 I NETWORK [listen] connection accepted from 172.18.0.3:36766 #1 (1 connection now open)
mongo_1 | 2018-04-02T19:35:58.480+0000 I NETWORK [conn1] received client metadata from 172.18.0.3:36766 conn: { driver: { name: "nodejs
", version: "3.0.5" }, os: { type: "Linux", name: "linux", architecture: "x64", version: "4.9.0-linuxkit-aufs" }, platform: "Node.js v9.
10.0, LE, mongodb-core: 3.0.5" }
mongo_1 | 2018-04-02T19:35:58.492+0000 I STORAGE [conn1] createCollection: bd_docker.colecc_docker with generated UUID: 2fc35647-cb4c-4
bad-84e8-cde7bffd434
mongo_1 | 2018-04-02T19:35:59.415+0000 I COMMAND [conn1] command bd_docker.colecc_docker command: create { create: "colecc_docker", lsi
d: { id: UUID("a9068bee-21ee-4bfe-8e65-b15cc658b4b9") }, $db: "bd_docker" } numYields:0 reslen:37 locks:{ Global: { acquireCount: { r: 1,
w: 1 } }, Database: { acquireCount: { W: 1 } } } protocol:op_query 922ms

```

El log continua en la hoja siguiente, los archivos completos pueden consultarse en:

<https://github.com/poximan/TNT-PM/tree/master/Práctica/tp1/pto10>