

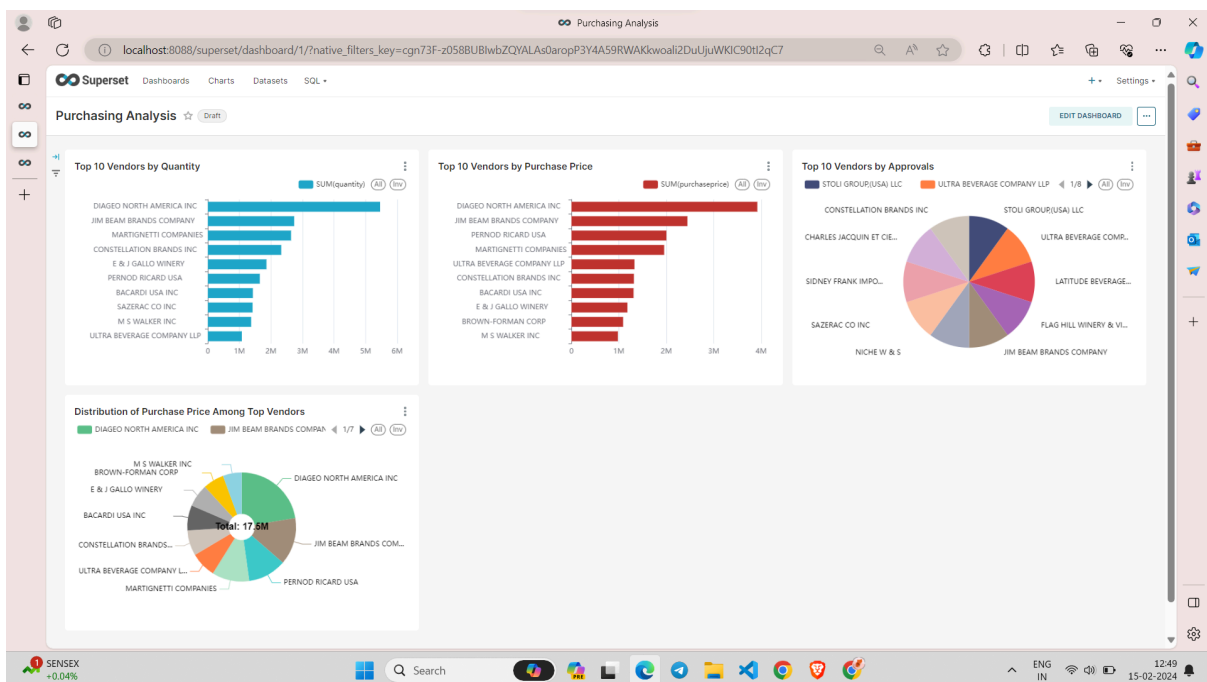
Directory

sup

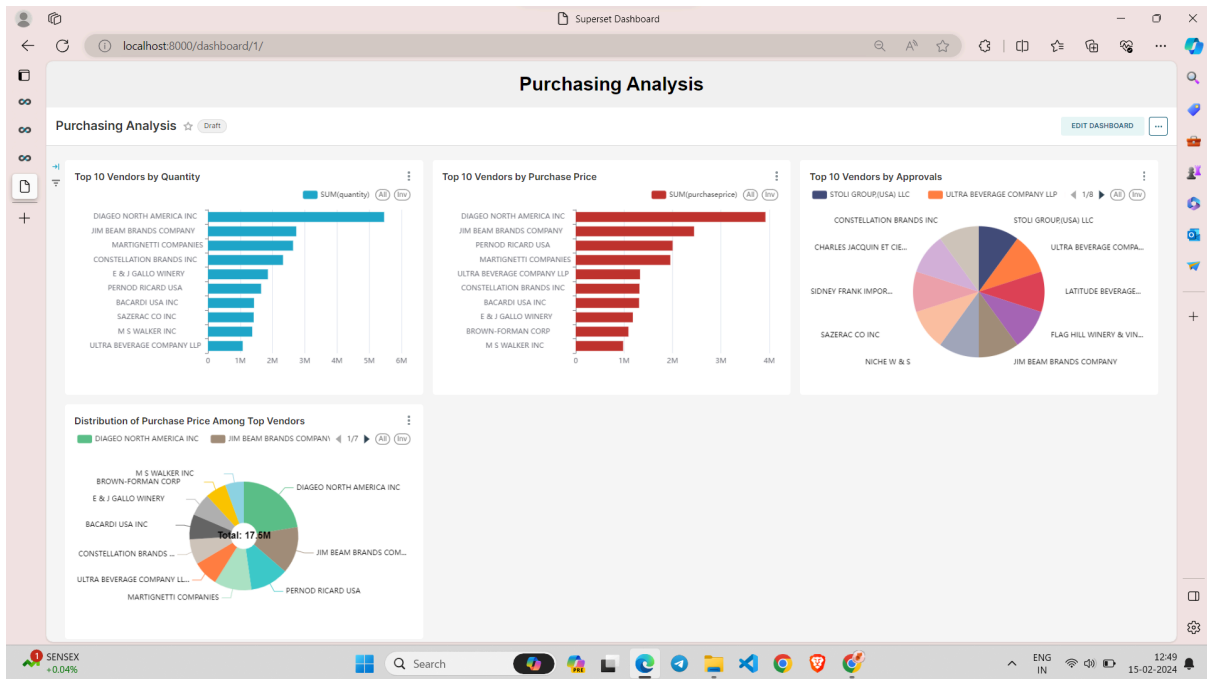
- ├── Invoices.csv
- ├── Product.csv
- ├── Purchase.csv
- ├── superset
 - ├── dockerfile
 - ├── superset_config.py
 - ├── superset-init.sh
- ├── docker-compose.yml
- ├── manage.py
- ├── requirements
 - ├── base.txt
- ├── djangoapp
 - ├── DjangoDockerfile
- ├── templates
 - ├── dashboard.html
- ├── views.py
- └── urls.py

OUTPUT

Superset



Django app



CODE WAY

```
archi@AROL:~/sup$ docker-compose build
[+] Building 4.8s (21/21) FINISHED
=> [superset internal] load build definition from dockerfile
=> => transferring dockerfile: 316B
=> [superset internal] load metadata for docker.io/apache/superset:latest
=> [superset auth] apache/superset:pull token for registry-1.docker.io
=> [superset internal] load .dockerignore
=> => transferring context: 2B
=> [superset 1/5] FROM docker.io/apache/superset:latest@sha256:dc9271154beca4dd4118cd30f6089b23e29896013dab30d5eca3e68544c80bf2
=> [superset internal] load build context
=> => transferring context: 76B
=> CACHED [superset 2/5] RUN pip install psycpg2-binary
=> CACHED [superset 3/5] COPY ./superset-init.sh /superset-init.sh
=> CACHED [superset 4/5] RUN chmod +x /superset-init.sh
=> CACHED [superset 5/5] COPY superset_config.py /app/
=> [superset] exporting to image
=> => exporting layers
=> => writing image sha256:788a2748c296900578bb0e575ee9754376b11d62cc63634cd9f7dd58056421e9
=> => naming to docker.io/library/sup-superset
=> [django internal] load build definition from DjangoDockerfile
=> => transferring dockerfile: 372B
=> [django internal] load metadata for docker.io/library/python:3.9
=> [django auth] library/python:pull token for registry-1.docker.io
=> [django internal] load .dockerignore
=> => transferring context: 2B
=> [django 1/4] FROM docker.io/library/python:3.9@sha256:eea7d8748b2448d25bba68dad87fe205dc361c854718d35865f2f7f46a81acc6
=> [django internal] load build context
=> => transferring context: 1.27kB
=> CACHED [django 2/4] RUN pip install django
=> CACHED [django 3/4] WORKDIR /code
=> CACHED [django 4/4] COPY . /code
=> [django] exporting to image
=> => exporting layers
=> => writing image sha256:1532c33e7c5aeaa127a00e501c64f842d74f4b22af3e6676e7f61c951f6e421
=> => naming to docker.io/library/sup-django
```

```

archi@AROL:~/sup$ docker-compose up -d
[+] Running 15/15
  ✓ postgres 1/4 layers [#####] 0B/0B Pulled 50.5s
  ✓ c57ae500d61 Already exists 0.0s
  ✓ 81b575116500 Pull complete 0.9s
  ✓ e12fff61d996 Pull complete 2.8s
  ✓ 50a849db7317 Pull complete 1.5s
  ✓ 432dd17f42df Pull complete 5.5s
  ✓ a1f5bcbba6b6 Pull complete 3.2s
  ✓ 6e501216828b Pull complete 3.7s
  ✓ ea24c7671c3d Pull complete 4.1s
  ✓ b7a5cd7c9b9a Pull complete 41.0s
  ✓ db7d78d9f46e Pull complete 5.3s
  ✓ 8c786fbf8634 Pull complete 6.7s
  ✓ 2831031f2a0e Pull complete 7.6s
  ✓ 75c5b068b243 Pull complete 7.8s
  ✓ 9590d9e20e85 Pull complete 8.8s
[+] Running 3/4
  ✓ Network sup_default Created 6.4s
  ✓ Container postgres_container Started 4.8s
  ✓ Container superset_container Started 2.3s
  ✓ Container django_container Started 2.8s
archi@AROL:~/sup$ docker cp InvoicePurchases.csv postgres_container:/tmp/InvoicePurchases.csv
Successfully copied 593kB to postgres_container:/tmp/InvoicePurchases.csv
archi@AROL:~/sup$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' postgres_container
192.168.208.2
archi@AROL:~/sup$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' superset_container
192.168.208.3
archi@AROL:~/sup$ docker exec -it superset_container ping 192.168.96.2
OCI runtime exec failed: exec failed: unable to start container process: exec: "ping": executable file not found in $PATH: unknown
archi@AROL:~/sup$ docker exec -it superset_container ping 192.168.96.3
OCI runtime exec failed: exec failed: unable to start container process: exec: "ping": executable file not found in $PATH: unknown
archi@AROL:~/sup$ docker exec -it postgres_container bash
root@76400626365d:/# psql -U postgres
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

```

```

postgres=# \l

```

Name	Owner	Encoding	Locale Provider	Collate	Ctype	ICU Locale	ICU Rules	Access privileges
mydb	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
postgres	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
template0	postgres	UTF8	libc	en_US.utf8	en_US.utf8			=c/postgres +
template1	postgres	UTF8	libc	en_US.utf8	en_US.utf8			=c/postgres +
								postgres=Ctc/postgres

```

(4 rows)

postgres=# \c mydb
You are now connected to database "mydb" as user "postgres".
mydb=# CREATE TABLE invoices (
    VendorNumber VARCHAR(50),
    VendorName VARCHAR(255),
    InvoiceDate DATE,
    PONumber VARCHAR(50),
    PODate DATE,
    PayDate DATE,
    Quantity INTEGER,
    Dollars NUMERIC,
    Freight NUMERIC,
    Approval VARCHAR(50)
);
CREATE TABLE
mydb=# COPY invoices FROM '/tmp/InvoicePurchases.csv' DELIMITER ',' CSV HEADER;
COPY 5543
mydb=# \q
root@76400626365d:/# exit
exit

```

docker-compose exec superset pip install google-cloud-bigquery

docker cp Product.csv postgres_container:/tmp/Product.csv

docker cp Invoice.csv postgres_container:/tmp/Invoice.csv

docker cp Purchase.csv postgres_container:/tmp/Purchase.csv

docker exec -it postgres_container bash

psql -U postgres

\c mydb

CREATE TABLE products (

Brand VARCHAR(255),

Description VARCHAR(255),

```
Price NUMERIC,  
Size VARCHAR(50),  
Volume VARCHAR(50),  
Classification VARCHAR(255),  
PurchasePrice NUMERIC,  
VendorNumber VARCHAR(50),  
VendorName VARCHAR(255)  
);  
\copy products FROM '/tmp/Product.csv' DELIMITER ',' CSV HEADER;
```

```
CREATE TABLE invoices (  
VendorNumber VARCHAR(50),  
VendorName VARCHAR(255),  
InvoiceDate DATE,  
PONumber VARCHAR(50),  
PODate DATE,  
PayDate DATE,  
Quantity INTEGER,  
Dollars NUMERIC,  
Freight NUMERIC,  
Approval VARCHAR(50)  
);  
\copy invoices FROM '/tmp/Invoice.csv' DELIMITER ',' CSV HEADER;
```

```
CREATE TABLE purchase (  
InventoryId VARCHAR(255),  
Store VARCHAR(255),  
Brand VARCHAR(255),  
Description VARCHAR(255),  
Size VARCHAR(50),  
VendorNumber VARCHAR(50),  
VendorName VARCHAR(255),  
PONumber VARCHAR(50),  
PODate DATE,  
ReceivingDate DATE,  
InvoiceDate DATE,  
PayDate DATE,  
PurchasePrice DECIMAL(10, 2),  
Quantity INT,  
Dollars DECIMAL(10, 2),  
Classification VARCHAR(50)
```

);

\copy purchase FROM '/tmp/Purchase.csv' DELIMITER ',' CSV HEADER;

Check for network

`docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'`


containername

`docker inspect -f '{{range .IPAM.Config}}{{.Subnet}}{{end}}' networkname`

```
archi@AROL:~/sup$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                NA
MES
9ec2dca91401   sup-django    "python3 manage.py r..." 2 hours ago   Up 2 hours   0.0.0.0:8000->8000/tcp  dj
ango_container
1d31d8affce6   sup-superset  "/superset-init.sh"      17 hours ago  Up 3 hours (healthy)  0.0.0.0:8088->8088/tcp  su
perset_container
1fc66ec6ee0b   postgres     "docker-entrypoint.s..." 17 hours ago  Up 3 hours   0.0.0.0:5432->5432/tcp  po
stgres_container
archi@AROL:~/sup$ docker network ls
NETWORK ID      NAME      DRIVER  SCOPE
d92bac320c82    bridge   bridge  local
c54f366dc856    host     host    local
c70302b58502    none     null     local
0985b9abdd9     sup_default bridge  local
archi@AROL:~/sup$ docker inspect -f '{{range .IPAM.Config}}{{.Subnet}}{{end}}' sup_default
192.168.0.0/20
archi@AROL:~/sup$ docker inspect -f '{{range .IPAM.Config}}{{.Subnet}}{{end}}' sup_default
192.168.0.0/20
archi@AROL:~/sup$ docker inspect -f '{{range .IPAM.Config}}{{.Subnet}}{{end}}' bridge
172.17.0.0/16
archi@AROL:~/sup$ docker inspect -f '{{range .IPAM.Config}}{{.Subnet}}{{end}}' host
archi@AROL:~/sup$ docker inspect -f '{{range .IPAM.Config}}{{.Subnet}}{{end}}' none
archi@AROL:~/sup$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' superset_container
192.168.0.3
archi@AROL:~/sup$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' django_container
192.168.0.4
archi@AROL:~/sup$ |
```

Test Superset's availability: Verify that Superset is accessible from other containers within the Docker network by attempting to access it using tools like `curl` or `requests`.

bash

 Copy code

```
docker exec -it django_container curl http://superset_container:8088
```

Replace `django_container` with the name of your Django container and `superset_container` with the name of your Superset container. This command will attempt to fetch the Superset homepage from the Django container. If successful, it indicates that Superset is accessible and responding to requests from other containers.



These commands will display the current firewall rules configured on your system, including any rules that may be blocking or allowing traffic on specific ports. Look for any rules related to port `8088` (or whichever port Superset is configured to listen on) and ensure that there are no rules blocking traffic to that port.

If you're using a firewall within the Docker network, you may need to inspect the firewall configuration within the Docker network itself. By default, Docker uses Linux iptables rules for network traffic management.

You can inspect Docker's iptables rules using the following command:

```
bash
```

[Copy code](#)

```
sudo iptables -L -n -t nat
```

This command will display the current NAT (Network Address Translation) rules configured by Docker, including any rules related to port forwarding or traffic redirection.



```
docker inspect -f '{{ .Config.Hostname }}' django_container
```

```
archi@AROL:~/sup$ docker exec -it django_container curl http://superset_container:8088
<!doctype html>
<html lang=en>
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to the target URL: <a href="/superset/welcome/">superset/welcome</a>. If not, click the link.
archi@AROL:~/sup$ sudo iptables -L
[sudo] password for archi:
Chain INPUT (policy ACCEPT)
target    prot opt source      destination

Chain FORWARD (policy ACCEPT)
target    prot opt source      destination

Chain OUTPUT (policy ACCEPT)
target    prot opt source      destination
```


You

how to hostname of my djnagocontianer

ChatGPT

To find the hostname of your Django container, you can use the `docker inspect` command. Here's how you can do it:

```
bash
```

 Copy code

```
docker inspect -f '{{ .Config.Hostname }}' django_container
```

Replace `django_container` with the name of your Django container. This command will output the hostname of the container.

1. EERORS: tried running superset and django container in sup and removing postgres image from docker compose file and creating postgres externally which caused diff network problem , tried connecting with creating custom network for docker but not worked,,,, here is the way i did initially

```
archi@AROL:~/sup$ docker run --name postgres_container -e POSTGRES_USER=postgres -e POSTGRES_PASSWORD=superset -e POSTGRES_DB=mydb -p 5432:5432 -d postgres
Unable to find image 'postgres:latest' locally
latest: Pulling from library/postgres
c57ee5000d61: Already exists
81b575116500: Pull complete
e12fff61d996: Pull complete
50a849db7317: Pull complete
432dd17f42df: Pull complete
a1f5bcbba6b6: Pull complete
6e501216828b: Pull complete
ea24c7671c3d: Pull complete
b7a5cd7c9b9a: Pull complete
db7d78d9f46e: Pull complete
8c786fbf8634: Pull complete
2031031f2a0e: Pull complete
75c5b068b243: Pull complete
9590d9e20e85: Pull complete
Digest: sha256:09f23e02d76670d3b346a3c00aa33a27cf57aab8341eedfcdaed41459d14f5c4
Status: Downloaded newer image for postgres:latest
bd6e1c24156d28a852d4c09a291f4507cd71d1057df6c77c0a479c496c05ed99
```

```
archi@AROL:~/sup$ docker cp InvoicePurchases.csv postgres_container:/tmp/InvoicePurchases.csv
Successfully copied 593kB to postgres_container:/tmp/InvoicePurchases.csv
```

```
archi@AROL:~/sup$ docker exec -it postgres_container bash
root@bd6e1c24156d:/# \l
bash: \l: command not found
root@bd6e1c24156d:/# psql -U mydb
psql: error: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: FATAL:  role "mydb" does not exist
root@bd6e1c24156d:/# psql -U postgres
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=# \l

```

Name	Owner	Encoding	Locale Provider	Collate	Ctype	ICU Locale	ICU Rules	Access privileges
mydb	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
postgres	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
template0	postgres	UTF8	libc	en_US.utf8	en_US.utf8			=c/postgres +
template1	postgres	UTF8	libc	en_US.utf8	en_US.utf8			postgres=Ctc/postgres +
								=c/postgres +
								postgres=Ctc/postgres

(4 rows)

```

postgres=# \l

```

Name	Owner	Encoding	Locale Provider	Collate	Ctype	ICU Locale	ICU Rules	Access privileges
mydb	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
postgres	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
template0	postgres	UTF8	libc	en_US.utf8	en_US.utf8			=c/postgres +
template1	postgres	UTF8	libc	en_US.utf8	en_US.utf8			postgres=Ct/postgres +
								=c/postgres +
								postgres=Ct/postgres

```

(4 rows)

postgres=# \c mydb
You are now connected to database "mydb" as user "postgres".
mydb=# CREATE TABLE invoices (
  VendorNumber VARCHAR(50),
  VendorName VARCHAR(255),
  InvoiceDate DATE,
  PONumber VARCHAR(50),
  PODate DATE,
  PayDate DATE,
  Quantity INTEGER,
  Dollars NUMERIC,
  Freight NUMERIC,
  Approval VARCHAR(50)
);
ERROR:  syntax error at or near "]"
LINE 1: ]l
        ^

mydb=# \c mydb
You are now connected to database "mydb" as user "postgres".
mydb=# CREATE TABLE invoices (
  VendorNumber VARCHAR(50),
  VendorName VARCHAR(255),
  InvoiceDate DATE,
  PONumber VARCHAR(50),
  PODate DATE,
  PayDate DATE,
  Quantity INTEGER,
  Dollars NUMERIC,
  Freight NUMERIC,
  Approval VARCHAR(50)
);
CREATE TABLE
mydb=# COPY invoices FROM '/tmp/InvoicePurchases.csv' DELIMITER ',' CSV HEADER;
COPY 5543
mydb=#

```

```

postgres=# COPY invoices FROM '/tmp/InvoicePurchases.csv' DELIMITER ',' CSV HEADER;
COPY 5543
postgres=# SELECT * FROM invoices;

[1]+  Stopped                  psql -U postgres
root@bd6e1c24156d:/# psql -U postgres
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=# SELECT EXISTS (
  SELECT 1
  FROM pg_tables
  WHERE schemaname = 'public'
  AND tablename = 'invoices'
);
exists
-----
t
(1 row)

```



```
postgres=# \q
root@bd6e1c24156d:/# exit
exit
There are stopped jobs.
root@bd6e1c24156d:/# exit
exit
archi@AROL:~/sup$ docker-compose build
[+] Building 6.4s (21/21) FINISHED
=> [superset internal] load build definition from dockerfile
=> [superset internal] load metadata for docker.io/apache/superset:latest
=> [superset internal] load .dockerignore
=> [superset internal] load build context
=> [superset 1/5] FROM docker.io/apache/superset:latest@sha256:dc9271154beca4dda118cd30f6089b23e29896013dab30d5eca3e68544c80bf2
=> [superset internal] load build context
=> [superset internal] load build context
=> [superset 2/5] RUN pip install psycopg2-binary
=> [superset 3/5] COPY ./superset-init.sh /superset-init.sh
=> [superset 4/5] RUN chmod +x /superset-init.sh
=> [superset 5/5] COPY superset_config.py /app/
=> [superset] exporting to image
=> [superset] exporting layers
=> [superset] writing image sha256:1e3d5215607da0c218744be80db4ee2fed105ac0108a584e4d106a8e7d7c0b34
=> [superset] naming to docker.io/library/sup-superset
=> [superset internal] load build definition from DjangoDockerfile
=> [superset internal] load metadata for docker.io/library/python:3.9
=> [superset internal] load .dockerignore
=> [superset internal] load build context
=> [superset 1/4] FROM docker.io/library/python:3.9@sha256:ea77d8748b2448d26bba68dad87fe205dc361c854718d35865f2f7f46a81acc6
=> [superset internal] load build context
=> [superset internal] load build context
=> [superset 2/4] RUN pip install django
=> [superset 3/4] WORKDIR /code
```

```
archi@AROL:~/sup$ docker-compose up -d
[+] Running 2/3
  Network sup_default          Created
  Container superset_container Started
  Container django_container   Started
archi@AROL:~/sup$ |
```

For this superset error

```
2024-02-07 19:38:26 Loaded your LOCAL configuration at [/app/superset_config.py]
2024-02-07 19:38:36 Loaded your LOCAL configuration at [/app/superset_config.py]
2024-02-07 19:38:46 Loaded your LOCAL configuration at [/app/superset_config.py]
2024-02-07 19:38:57 Loaded your LOCAL configuration at [/app/superset_config.py]
2024-02-07 19:39:07 Loaded your LOCAL configuration at [/app/superset_config.py]
2024-02-07 19:38:26 2024-02-07 14:08:26,485:ERROR:flask_appbuilder.security.sqla.manager:DB Creation and initialization failed: (psycopg2.OperationalError) could not tran
slate host name "postgres" to address: Temporary failure in name resolution
2024-02-07 19:38:26
2024-02-07 19:38:26 (Background on this error at: https://sqlalche.me/e/14/e3q8)
2024-02-07 19:38:26 [2024-02-07 14:08:26 +0000] [563] [INFO] Worker exiting (pid: 563)
2024-02-07 19:38:26 [2024-02-07 14:08:26 +0000] [70] [ERROR] Worker (pid:563) exited with code 1
2024-02-07 19:38:26 [2024-02-07 14:08:26 +0000] [70] [ERROR] Worker (pid:563) exited with code 1.
2024-02-07 19:38:26 [2024-02-07 14:08:26 +0000] [581] [INFO] Booting worker with pid: 581
2024-02-07 19:38:28 logging was configured successfully
2024-02-07 19:38:28 2024-02-07 14:08:28,649:INFO:superset.utils.logging_configurator:logging was configured successfully
2024-02-07 19:38:28 2024-02-07 14:08:28,654:INFO:root:Configured event logger of type <class 'superset.utils.Log.DBEventLogger'>
2024-02-07 19:38:28 /usr/local/lib/python3.9/site-packages/flask_limiter/extension.py:293: UserWarning: Using the in-memory storage for tracking rate limits as no storage
was explicitly specified. This is not recommended for production use. See: https://flask-limiter.readthedocs.io#configuring-a-storage-backend for documentation about con
figuring the storage backend.
2024-02-07 19:38:28 warnings.warn(
2024-02-07 19:38:36 2024-02-07 14:08:36,687:ERROR:flask_appbuilder.security.sqla.manager:DB Creation and initialization failed: (psycopg2.OperationalError) could not tran
slate host name "postgres" to address: Temporary failure in name resolution
2024-02-07 19:38:36
2024-02-07 19:38:36 (Background on this error at: https://sqlalche.me/e/14/e3q8)
2024-02-07 19:38:36 [2024-02-07 14:08:36 +0000] [581] [INFO] Worker exiting (pid: 581)
2024-02-07 19:38:36 [2024-02-07 14:08:36 +0000] [70] [ERROR] Worker (pid:581) exited with code 1
2024-02-07 19:38:36 [2024-02-07 14:08:36 +0000] [70] [ERROR] Worker (pid:581) exited with code 1.
2024-02-07 19:38:36 [2024-02-07 14:08:36 +0000] [599] [INFO] Booting worker with pid: 599
2024-02-07 19:38:38 logging was configured successfully
```

The error message you're encountering indicates that Superset is unable to resolve the hostname "postgres" to an IP address. This typically happens when the container running Superset is unable to communicate with the PostgreSQL container using the hostname provided.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
90d3b7f8176e	sup-superset	"/superset-init.sh"	9 minutes ago	Up 9 minutes (unhealthy)	0.0.0.0:8088->8088/tcp	superset_container
bd6e1c24156d	postgres	"docker-entrypoint.s..."	2 hours ago	Up 2 hours	0.0.0.0:5432->5432/tcp	postgres_container

Use the IP address of the PostgreSQL container: Instead of using the hostname "postgres" to connect to the PostgreSQL container, you can use the IP address of the container. First, find out the IP address of the PostgreSQL container by running:

```
arch1@AROL:~/sup$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' postgres_container  
172.17.0.2
```

```
arch1@AROL:~/sup$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' superset_container  
192.168.96.2
```

Both of my container are working in diff docker network

If your containers are running in different Docker networks and you want to connect them to the same network, you have a couple of options:

• Create a New Docker Network and Connect Both Containers:

You can create a new Docker network and connect both the Superset and PostgreSQL containers to that network. Here's how:

```
bash Copy code  
  
# Create a new Docker network  
docker network create my_network  
  
# Connect the Superset container to the new network  
docker network connect my_network superset_container  
  
# Connect the PostgreSQL container to the new network  
docker network connect my_network postgres_container
```

After connecting both containers to the same network, they will be able to communicate with each other using their container names or service names within the network.

• Use Docker Compose:



```
yaml Copy code

version: '3'

services:
  superset:
    image: superset_image
    networks:
      - my_network

  postgres:
    image: postgres_image
    networks:
      - my_network

networks:
  my_network:
    driver: bridge
```

Error similar to

ConnectionError at /dashboard/1/

HTTPConnectionPool(host='localhost', port=8088): Max retries exceeded with url: /api/v1/security/login (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x7f... establish a new connection: [Errno 111] Connection refused'))

```
Request Method: GET
Request URL: http://localhost:8000/dashboard/1/
Django Version: 2.0.8
Exception Type: ConnectionError
Exception Value: HTTPConnectionPool(host='localhost', port=8088): Max retries exceeded with url: /api/v1/security/login (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x7f... failed to establish a new connection: [Errno 111] Connection refused'))
Exception Location: /usr/local/lib/python3.9/site-packages/requests/adapters.py in send, line 519
Python Executable: /usr/local/bin/python3
Python Version: 3.9.18
Python Path: ['/app',
              '/usr/local/lib/python3.9.zip',
              '/usr/local/lib/python3.9',
              '/usr/local/lib/python3.9/lib-dynload',
              '/usr/local/lib/python3.9/site-packages',
              '/app/djangoapp']
Server time: Tue, 13 Feb 2024 18:49:06 +0000
```

Traceback [Switch to console view](#)

Using container name instead of localhost as django itself takes the localhost when we don't have any port 8088 so use superset_container in place of localhost in views.py