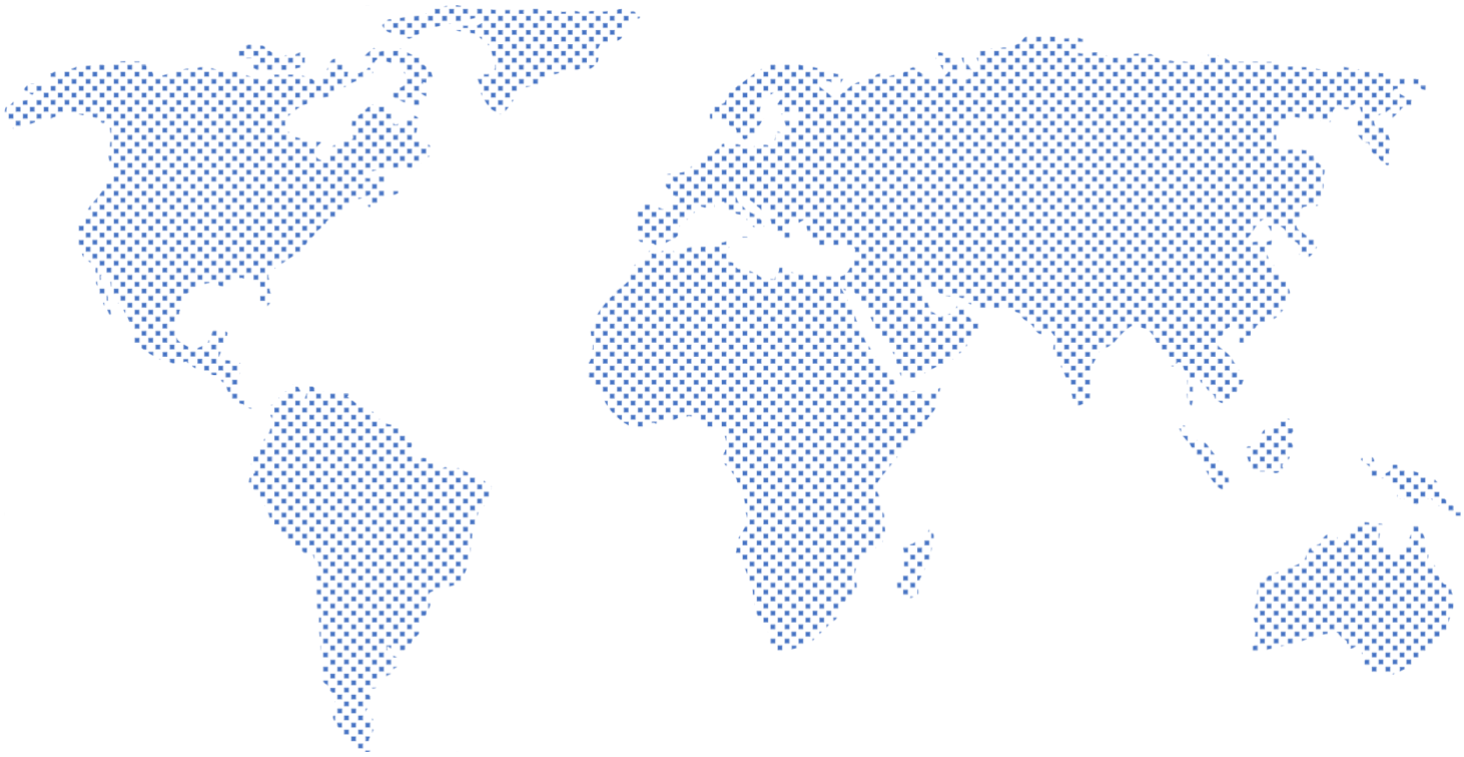
Pega Docker
Postgres/Tomcat Container



SRINIVASA KANDRU

Sreesoft Solutions

Contents

About this document

Currently Pega is available for learning in the form of personal edition (PE) or personal virtual machine (PVM). PE works only on windows & PVS does work on various platforms but eat up many resources. Also, in general tech world, for ease of development & deployment to cloud environments, Docker is used widely. There is no official docker version of Pega available. There seems a docker version just running tomcat but not any database which is key part of Pega installation. This document is to detail the steps needed to run Pega in docker containers.

This docker image can be configured in any of the following OS's

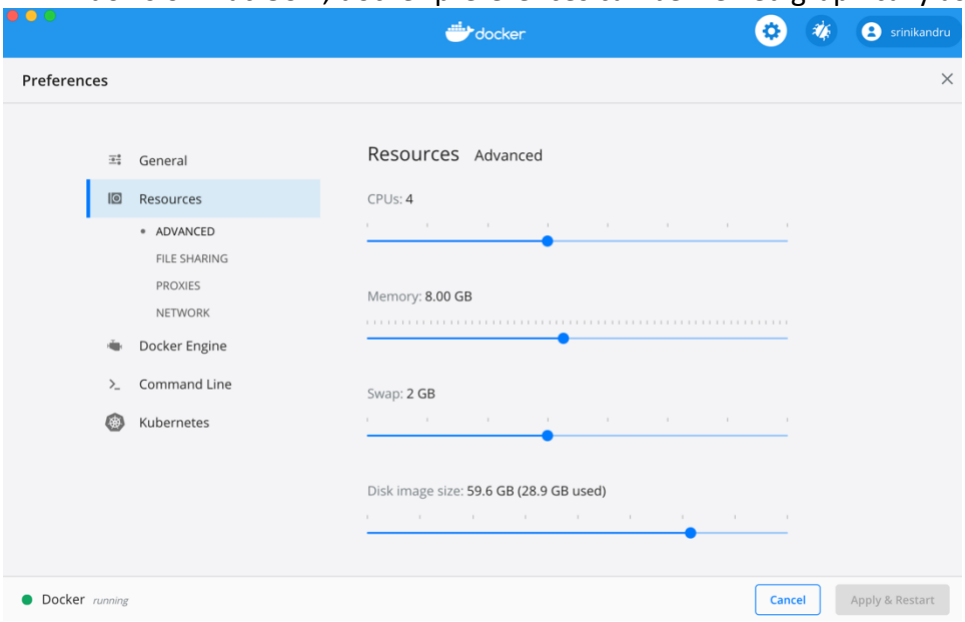
- Windows 10
- Mac OS X
- Linux

Check Prerequisites

Before installing Pega Docker images, ensure all the prerequisites are met and available in the system.

1. Ensure Docker software installed and running in the target system. Docker can be downloaded from here. https://www.docker.com/products/docker-desktop
2. Once installed, keep docker resource settings optimal for Pega. It is recommended to have 8GB of memory allocated. It may vary depends on target system capacity.

   In Windows or Mac OS X, docker preferences can be viewed graphically as below.



   In Linux distros, Docker share host OS capacities usually. It can be configured following respective Linux distro CLI commands. To see how the resources allocated to Docker, run the below command and see.

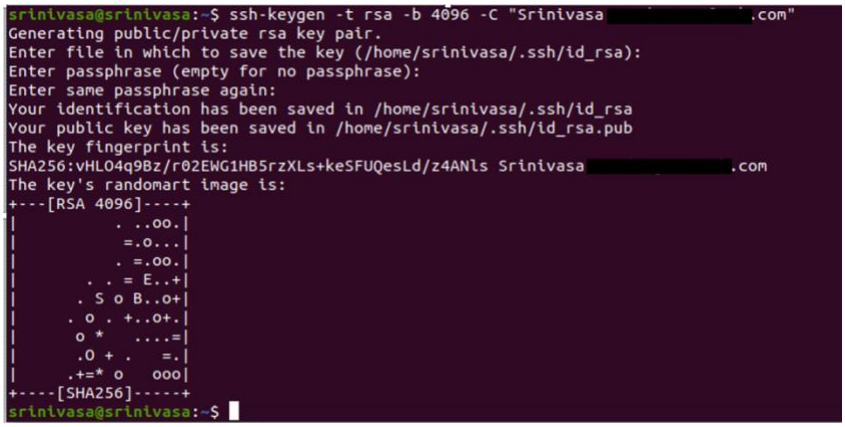
   ```
   docker info
   ```

3. Have a GitHub account and familiarise how to check-out repositories.
4. Install git command line utility as described here.
   https://www.atlassian.com/git/tutorials/install-git

   It may be required to authenticate local system with GitHub to allow check-out a repository. Please follow the below articles to resolve any issues if they arise.

   https://docs.github.com/en/free-pro-team@latest/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account

   https://docs.github.com/en/free-pro-team@latest/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

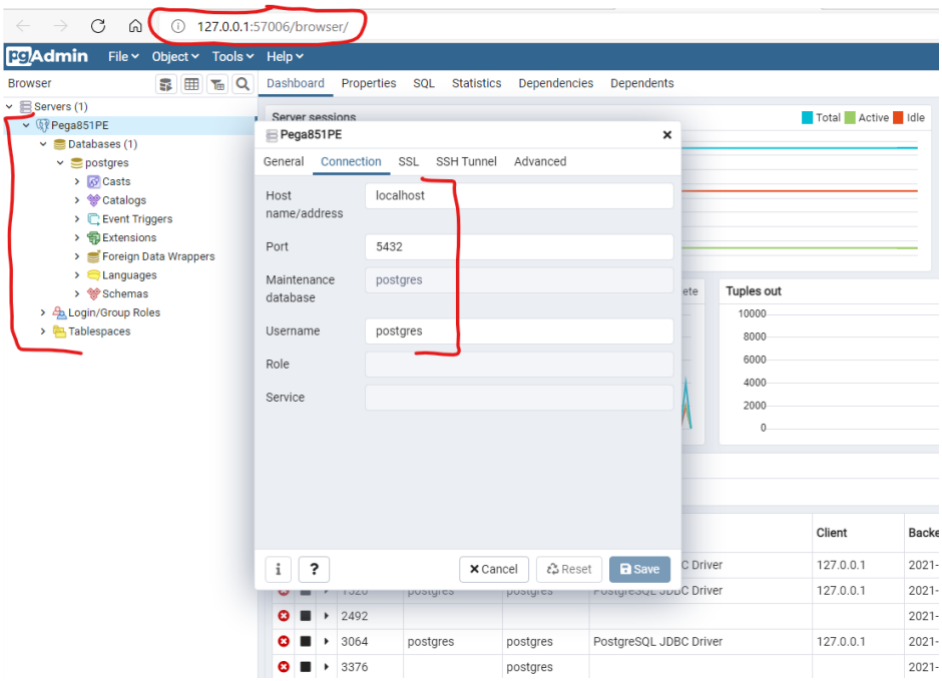


5. Install pgAdmin 4 client.
6. Keep/Extract Pega Database backup from Personal Edition. To extract, follow the related section in this document.

1. Extract Pega Database Backup from PE (Personal Edition)

To run Pega in Docker containers, we are relying on postgres database backup extracted from a running Pega instance. In this case, it is going to be Personal Edition of Pega. (Following steps can be applied to any other pega instance running on postgres)

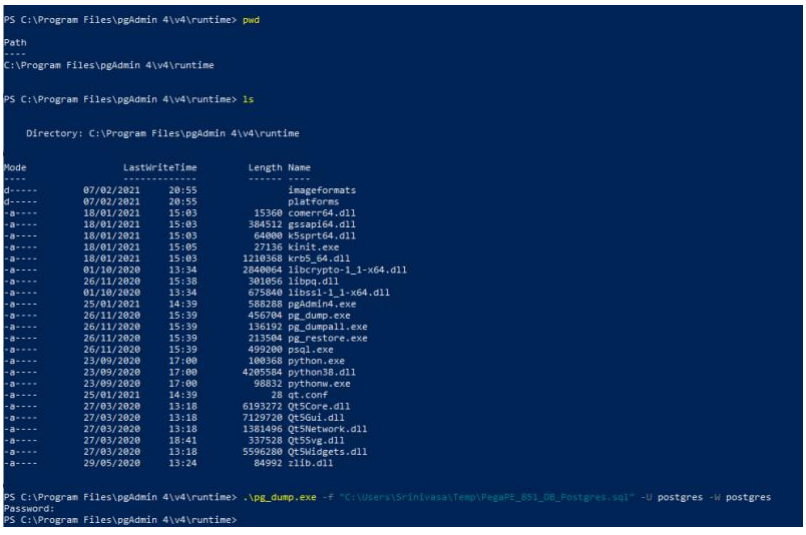
1. To extract database backup Postgres DB, we are going to use **pg_dump** tool. (this tool comes with pgAdmin 4 standard installation)
2. Install Pega personal edition before going to next step.
3. Check/Configure DB access with pre-installed pgadmin-4 client as shown in below screenshot. (password is usually either ‘postgres’ or ‘password’)



4. Below example is taken from Windows, for other OS, go through the postgres recommendations.
5. Run below command (or similar as per your environment) to extract postgres DB backup to a chosen folder (For ex : "C:\Users\Srinivasa\Temp")

In Windows :

C:\Program Files\pgAdmin 4\v4\runtime> .\pg_dump.exe -f "C:\Users\Srinivasa\Temp\PegaPE_851_DB_Postgres.sql" -U postgres -W postgres



In Linux (with remote postgres) :

/usr/bin/pg_dump --host "192.168.2.111" --port "5432" --username "postgres" --password --dbname "postgres" --file "/home/srinivasa/Downloads/PegaPE_851_ DB_Postgres.sql"

6. **Keep the backed-up copy handy for future steps**.

To restore the backup copy into fresh postgres image, below/similar commands are used. (These commands are useful in future steps too)

pg_restore if binary backup taken

/usr/bin/pg_restore --host "192.168.2.71" --port "5432" --username "postgres" --no-password --dbname "postgres" --verbose "/share/samba/PegaPE_851_DB_Postgres.sql"

psql if sql/text backup taken

/usr/bin/psql --host "192.168.2.111" --port "5432" --username "postgres" --password --dbname "postgres" --file "/home/srinivasa/Downloads/PegaPE_851_DB_Postgres.sql"

2. Build Docker Image(s)

To install Pega, it is required to run a Database container and an App server container. Key details are as below.

1. A postgres container will be configured first and a backup DB file will be dumped into it to call it Pega.
2. A tomcat container will be configured next and linked to postgres container to read from Database.
3. This tomcat container will act as app/web server.
4. All necessary files and dependencies are made available at respective GitHub repositories.
   https://github.com/sreesoft/pegapostgres
   https://github.com/sreesoft/pegatomcat

Follow step-by-step guide to clone these repositories and build your own local copy.

1. First, choose a base folder to keep all files cloned from GitHub. (Example folder is visible in the below screenshot. 'Test')
2. Use Terminal/Command Line/Power shell (based on operating system) to go to this folder.
3. Once in this folder, use git clone command to clone 'pegapostgres' repo as shown in below screenshot.

git clone git@github.com:sreesoft/pegapostgres.git

```
Srinivasas-MacBook-Pro:Test Srinivasa$ pwd
/Users/Srinivasa/OneDrive/MyWork/Docker/Test
Srinivasas-MacBook-Pro:Test Srinivasa$ git clone git@github.com:sreesoft/pegapostgres.git
Cloning into 'pegapostgres'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 11 (delta 0), reused 11 (delta 0), pack-reused 0
Receiving objects: 100% (11/11), 1.67 MiB | 2.93 MiB/s, done.
Srinivasas-MacBook-Pro:Test Srinivasa$ ll
total 0
drwxr-xr-x  10 Srinivasa  staff  320 27 Aug 16:27 pegapostgres
Srinivasas-MacBook-Pro:Test Srinivasa$
```

1. Now clone other repository ‘pegatomcat’ as shown below.

   git clone git@github.com:sreesoft/pegatomcat.git

```
Srinivasas-MacBook-Pro:Test Srinivasa$ ll
total 0
drwxr-xr-x@ 11 Srinivasa  staff  352 27 Aug 17:12 pegapostgres
Srinivasas-MacBook-Pro:Test Srinivasa$ git clone git@github.com:sreesoft/pegatomcat.git
Cloning into 'pegatomcat'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0
Receiving objects: 100% (9/9), 1.20 MiB | 1.65 MiB/s, done.
Srinivasas-MacBook-Pro:Test Srinivasa$ ll
total 0
drwxr-xr-x@ 11 Srinivasa  staff  352 27 Aug 17:12 pegapostgres
drwxr-xr-x  10 Srinivasa  staff  320 27 Aug 17:40 pegatomcat
Srinivasas-MacBook-Pro:Test Srinivasa$
```

2. There should be 2 folders created after these clone commands executed successfully.
3. Go into ‘pegapostgres’ folder and inspect the contents.

```
Srinivasas-MacBook-Pro:Test Srinivasa$ cd pegapostgres/
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$ ll
total 6824
-rwxr-xr-x@ 1 Srinivasa  staff  2774201 27 Aug 16:27 Docker_Pega_Postgres_Image_Build.docx
-rwxr-xr-x@ 1 Srinivasa  staff      944 27 Aug 16:27 Dockerfile
-rw-r--r--@ 1 Srinivasa  staff       15 27 Aug 16:27 README.md
-rwxr-xr-x@ 1 Srinivasa  staff     4574 27 Aug 16:27 pg_hba.conf
-rwxr-xr-x@ 1 Srinivasa  staff   340436 27 Aug 16:27 pljava.jar
-rwxr-xr-x@ 1 Srinivasa  staff   329672 27 Aug 16:27 pljava.so
-rwxr-xr-x@ 1 Srinivasa  staff    24184 27 Aug 16:27 postgresql.conf
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$
```

4. Check if you have any docker images existing. Please verify if same named image exists. If exists, be mindful this build will overwrite it. To check existing docker images, use this command.

   docker images

```
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$
```

5. Now, build this image as shown below. It may take few minutes based on internet bandwidth.

   docker build -t pegapostgres .

```
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$ ll
total 6824
-rwxr-xr-x@ 1 Srinivasa  staff  2774201 27 Aug 16:27 Docker_Pega_Postgres_Image_Build.docx
-rwxr-xr-x@ 1 Srinivasa  staff      944 27 Aug 16:27 Dockerfile
-rw-r--r--@ 1 Srinivasa  staff       15 27 Aug 16:27 README.md
-rwxr-xr-x@ 1 Srinivasa  staff     4574 27 Aug 16:27 pg_hba.conf
-rwxr-xr-x@ 1 Srinivasa  staff   340436 27 Aug 16:27 pljava.jar
-rwxr-xr-x@ 1 Srinivasa  staff   329672 27 Aug 16:27 pljava.so
-rwxr-xr-x@ 1 Srinivasa  staff    24184 27 Aug 16:27 postgresql.conf
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$ docker build -t pegapostgres .
```

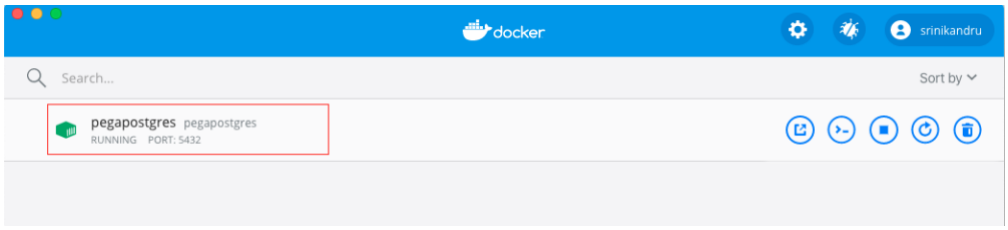
```
update-alternatives: warning: skip creation of /usr/share/man/it/man1/editor.1.gz because associated file /usr/share/man/it/ma
update-alternatives: warning: skip creation of /usr/share/man/pl/man1/editor.1.gz because associated file /usr/share/man/pl/ma
update-alternatives: warning: skip creation of /usr/share/man/ru/man1/editor.1.gz because associated file /usr/share/man/ru/ma
update-alternatives: warning: skip creation of /usr/share/man/ja/man1/editor.1.gz because associated file /usr/share/man/ja/ma
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/vim
Removing intermediate container 1b946a7d78e5
 ---> 86135bd903cf
Step 6/11 : RUN ln -s /usr/lib/jvm/java-8-openjdk-amd64/jre/lib/amd64/server/libjvm.so /lib/libjvm.so
 ---> Running in 509bad5ea790
Removing intermediate container 509bad5ea790
 ---> 4d8b0b129be6
Step 7/11 : COPY ./pljava.so /usr/lib/postgresql/11/lib
 ---> 7ff905518243
Step 8/11 : COPY ./pljava.jar /usr/lib/postgresql/11/lib
 ---> fb2aa51539e0
Step 9/11 : RUN mkdir -p /var/lib/postgresql-static/data
 ---> Running in d002fc903041
Removing intermediate container d002fc903041
 ---> 24ff2119c648
Step 10/11 : RUN chown -R postgres:postgres /var/lib/postgresql-static
 ---> Running in 5b1ccf51d56a
Removing intermediate container 5b1ccf51d56a
 ---> c353372c978c
Step 11/11 : ENV PGDATA /var/lib/postgresql-static/data
 ---> Running in b73a8a4c667b
Removing intermediate container b73a8a4c667b
 ---> f936642c5b7e
Successfully built f936642c5b7e
Successfully tagged pegapostgres:latest
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$
```

6. Once successfully built the image, run it using the below command. Note the database password is used as ‘postgres’.

```
docker run --name pegapostgres -e POSTGRES_PASSWORD=postgres -p 5432:5432 -d pegapostgres
```

```
postgres      11.8      31d22a1554df   8 months ago    282MB
Srinivasas-MBP:pegapostgres Srinivasa$ docker run --name pegapostgres -e POSTGRES_PASSWORD=postgres -p 5432:5432 -d pegapostgres
7ba9f64186defc69694555542a7ba5b4ec95a2d95162b6dfe3f2bd16e2bbd4e2
Srinivasas-MBP:pegapostgres Srinivasa$ ll
```

7. On success, docker dashboard will show running container as below.



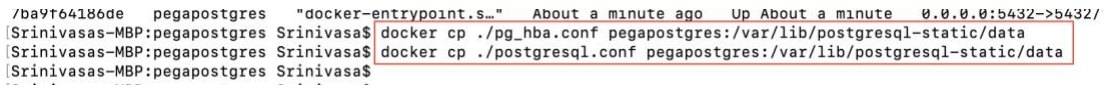
3. Configure Pega Postgres image

Postgres image maintains its configuration files & data at a default volume (think as volume mount) which may not be persisted upon committing docker containers.

It is required to update few configurations to make it work for Pega. All required configuration files are prepared and available in this repository. Follow below steps to amend your local image.

- Copy config files into container as below.

```
docker cp ./pg_hba.conf pegapostgres:/var/lib/postgresql-static/data
docker cp ./postgresql.conf pegapostgres:/var/lib/postgresql-static/data
```



- This docker container does not have any Pega database installed yet. We are going to utilise the backed-up copy made earlier in this document.

Please note, PegaPE_851_DB_Postgres is the filename of DB backup extracted earlier. It is only working name for it, different name can be used.

There are 2 ways to export the DB Backup into postgres docker container. First one is to use ‘psql’ utility provided as part of PGAdmin4 tool. Second one is to upload this backup file into Docker container and install. Both options are documented below. Use according to the convenience.

3.1 Export Backup file via host ‘psql’ utility.

Find ‘psql’ utility in host machine. If its Mac, it usually available in the shown path in below image. Once found, run the below command as shown below. File paths may vary.

```
./psql --host "localhost" --port "5432" --username "postgres" --password --dbname "postgres" --file
"/Users/Srinivasa/Downloads/PegaPE_851_DB_Postgres.sql"
```

Password to use here is ‘postgres’

Once password is accepted, it will start loading all tables and objects into Postgres container. It may take a while depends on target system capacity. Once completed, it will show similar as below.

```
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
Srinivasas-MBP:SharedSupport Srinivasa$
```

If no errors are presented, that's it. DB is successfully loaded. Then skip section 3.2 and go to 3.3.

3.2 Upload backup file into container and load

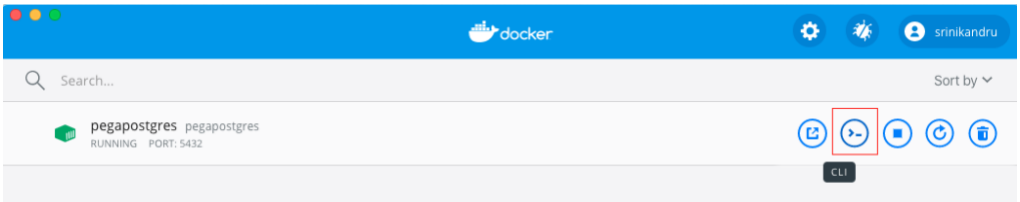
When 'psql' utility not available in host machine or not interested to run from host, these steps to be followed.

First load the DB backup file into containers temp location as shown below.

```
docker cp ./PegaPE_851_DB_Postgres.sql pegapostgres:/tmp
```

```
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$ docker cp ./PegaPE_840_DB_Postgres.sql pegapostgres:/tmp
Srinivasas-MacBook-Pro:pegapostgres Srinivasa$
```

Now, go to docker dashboard and open CLI (Command Line Interface) utility to run below commands.




```
sudo bash
sudo -su postgres
psql -d postgres -f /tmp/PegaPE_851_DB_Postgres.sql
```

```
Srinivasa — docker exec -it fb8b30be3d30622fe5a4f6d20c396901e2525c25099c54976c8528132aaa1747 /bin/sh — 120×31
Last login: Thu Aug 27 15:34:47 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Srinivasas-MacBook-Pro:~ Srinivasa$ docker exec -it fb8b30be3d30622fe5a4f6d20c396901e2525c25099c54976c8528132aaa1747 /bi
n/sh; exit
# sudo bash
root@fb8b30be3d30:/# sudo -su postgres
postgres@fb8b30be3d30:/$ psql -d postgres -f /tmp/PegaPE_840_DB_Postgres.sql
```

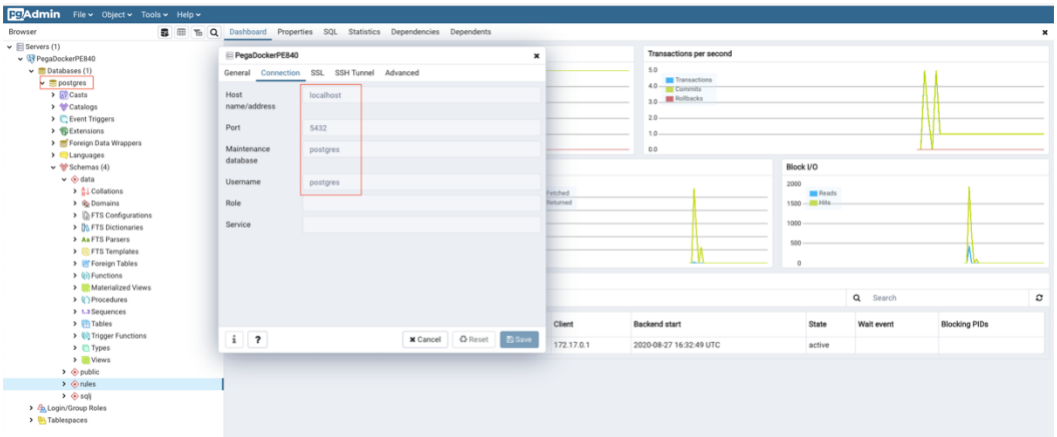
```
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
CREATE INDEX
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
postgres@fb8b30be3d30:/$
```

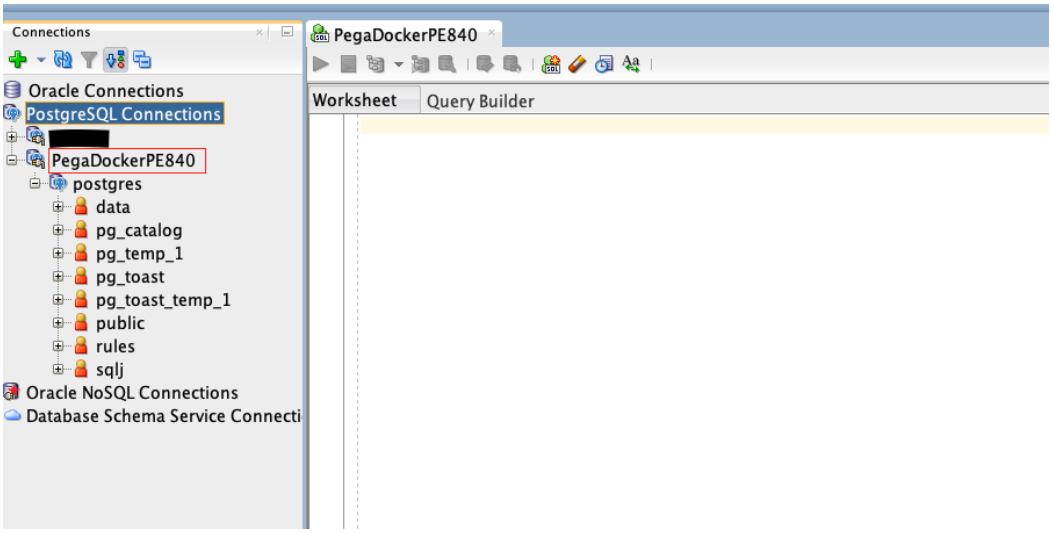
3.3 Test Postgres Connection

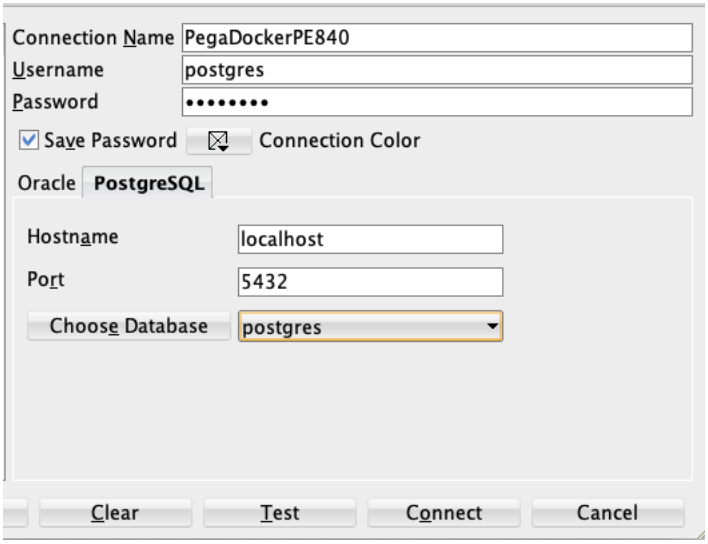
When backup file loaded successfully, It is time to test Postgres connection via PGAdmin4 or SQL Developer

Use connection details as shown in screenshot.

Use password as ‘postgres’.







4. Configure Pega Tomcat image

Go to ‘pegatomcat’ folder created earlier and build image as shown below.

```
docker build -t pegatomcat .
```

```
Srinivasas-MacBook-Pro:Test Srinivasa$ cd pegatomcat/
Srinivasas-MacBook-Pro:pegatomcat Srinivasa$ ll
total 2640
-rwxr-xr-x  1 Srinivasa  staff    1046 27 Aug 17:40 Dockerfile
-rw-r--r--  1 Srinivasa  staff      13 27 Aug 17:40 README.md
-rwxr-xr-x  1 Srinivasa  staff    2587 27 Aug 17:40 context.xml
-rwxr-xr-x  1 Srinivasa  staff  932808 27 Aug 17:40 postgresql-42.2.14.jar
-rwxr-xr-x  1 Srinivasa  staff  397135 27 Aug 17:40 prweb.war
-rwxr-xr-x  1 Srinivasa  staff     313 27 Aug 17:40 setenv.sh
-rwxr-xr-x  1 Srinivasa  staff    2320 27 Aug 17:40 tomcat-users.xml
Srinivasas-MacBook-Pro:pegatomcat Srinivasa$ docker build -t pegatomcat .
```

```
Removing intermediate container e4c815d1bae8
 ---> c18e1a104765
Step 11/22 : EXPOSE 8080
 ---> Running in 63a505f80f26
Removing intermediate container 63a505f80f26
 ---> 3e18b825e60b
Step 12/22 : COPY ./postgresql-42.2.14.jar /opt/tomcat/lib
 ---> 70b6c5418432
Step 13/22 : RUN mkdir /pega
 ---> Running in 2384a1d2a252
Removing intermediate container 2384a1d2a252
 ---> 88536ae70d7f
Step 14/22 : RUN mkdir /pega/logs
 ---> Running in 1ca8f21d9f48
Removing intermediate container 1ca8f21d9f48
 ---> 0798bed97220
Step 15/22 : RUN mkdir /pega/index
 ---> Running in 910d106496ab
Removing intermediate container 910d106496ab
 ---> 088949b37775
Step 16/22 : RUN mkdir /pega/temp
 ---> Running in 193d8f5e3153
Removing intermediate container 193d8f5e3153
 ---> 0d56c0c5e013
Step 17/22 : RUN mkdir /pega/cassandra_data
 ---> Running in 703718d7ac51
Removing intermediate container 703718d7ac51
 ---> 31c463a7d2e3
Step 18/22 : COPY ./context.xml /opt/tomcat/conf
 ---> 3ea3ffea53c3
Step 19/22 : COPY ./tomcat-users.xml /opt/tomcat/conf
 ---> fb9709fa38ac
Step 20/22 : COPY ./setenv.sh /opt/tomcat/bin
 ---> e2ffc695a6ef
Step 21/22 : COPY ./prweb.war /opt/tomcat/webapps
 ---> d6835841fd44
Step 22/22 : CMD /opt/tomcat/bin/catalina.sh run
 ---> Running in 44f092e2289c
Removing intermediate container 44f092e2289c
 ---> cf312b984f93
Successfully built cf312b984f93
Successfully tagged pegatomcat:latest
Srinivasas-MacBook-Pro:pegatomcat Srinivasa$
```

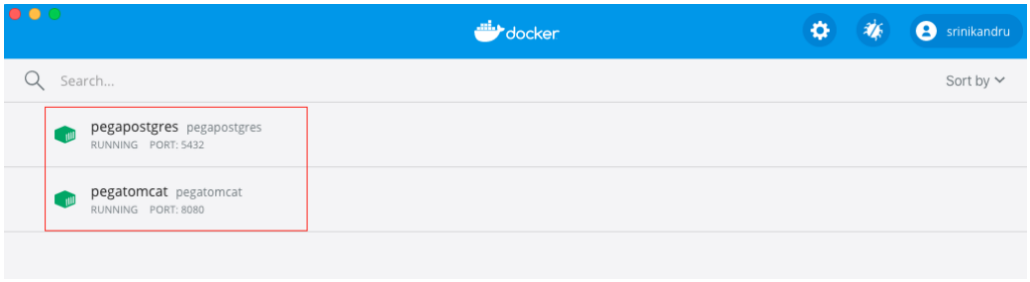
5. Running Pega Instance(s)

1. Once pegatomcat image is build, it is required to run this image by linking pegapostgres container. Use the below command to run it.

   docker run --name pegatomcat -d -p 8080:8080 --link pegapostgres:pegapostgres pegatomcat

```
Successfully tagged pegatomcat:latest
Srinivasas-MacBook-Pro:pegatomcat Srinivasa$ docker run --name pegatomcat -d -p 8080:8080 --link pegapostgres:pegapostgres pegatomcat
53da18961427e3f583220bd421d2f9e048d63a854691f33aba86cb02752f499d
Srinivasas-MacBook-Pro:pegatomcat Srinivasa$
```

2. It should have below 2 running instances as shown.

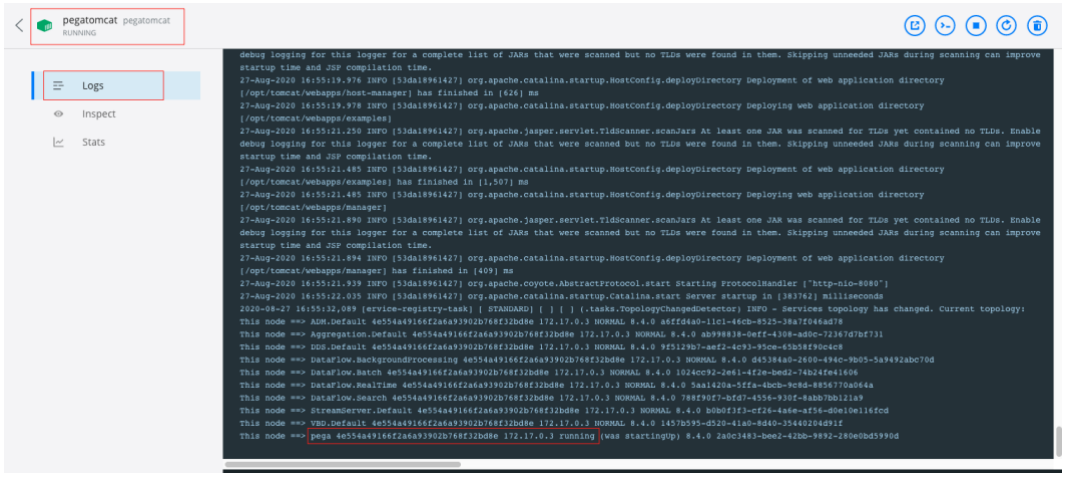


If using Linux, run the below command to see docker running instances.

docker ps

```
skandru@SrinivasaUbuntu:~$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED      STATUS      PORTS                    NAMES
65245723b156   pegatomcat     "/bin/sh -c '/usr/lo…"   3 days ago   Up 3 days   0.0.0.0:8080->8080/tcp   pegatomcat
464d1be09ea0   pegapostgres   "docker-entrypoint.s…"   4 days ago   Up 4 days   0.0.0.0:5432->5432/tcp   pegapostgres
skandru@SrinivasaUbuntu:~$
```

3. Check its logs and observe the highlighted log message to know pega is up & running. It may take few minutes to start up this container.

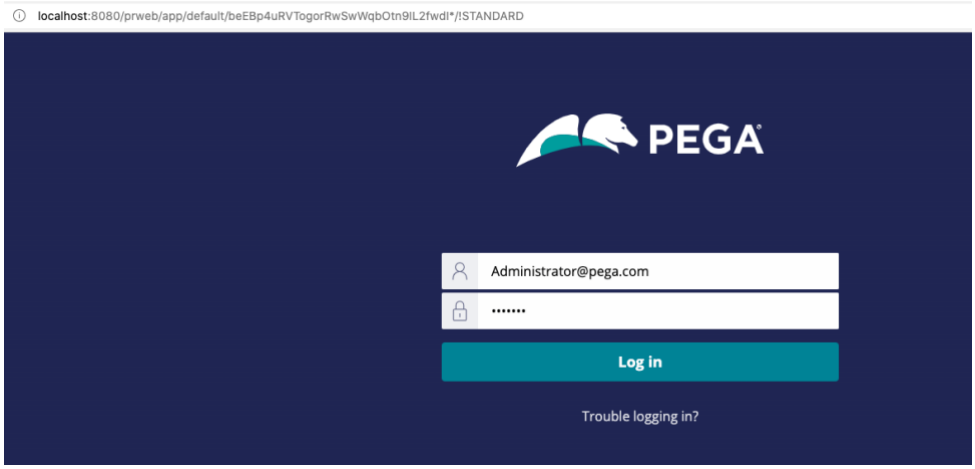


4. To open terminal of docker container from command line, use the below and check logs & other configs.

docker exec -it pegatomcat /bin/bash

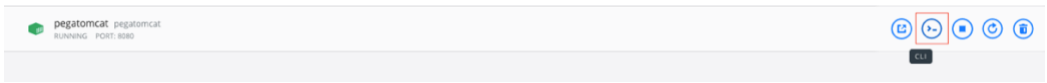
```
skandru@SrinivasaUbuntu:~$ docker exec -it pegatomcat /bin/bash
root@65245723b156:/usr/local/tomcat# ll
total 184
-rw-r--r--   1 root root  6898 Dec  3 11:48 RELEASE-NOTES
-rw-r--r--   1 root root  3257 Dec  3 11:48 README.md
-rw-r--r--   1 root root  2333 Dec  3 11:48 NOTICE
-rw-r--r--   1 root root 57092 Dec  3 11:48 LICENSE
-rw-r--r--   1 root root  5409 Dec  3 11:48 CONTRIBUTING.md
-rw-r--r--   1 root root 18982 Dec  3 11:48 BUILDING.txt
-rw-r--r--   1 root root 16507 Dec  3 11:48 RUNNING.txt
drwxr-xr-x   2 root root  4096 Dec 18 08:59 native-jni-lib
drwxr-xr-x   1 root root  4096 Dec 23 20:13 lib
drwxr-xr-x   1 root root  4096 Dec 23 20:13 bin
drwxr-xr-x   1 root root  4096 Dec 23 20:13 webapps.dist
drwxr-xr-x   1 root root  4096 Dec 23 20:14 conf
drwxr-xr-x   1 root root  4096 Dec 23 20:14 webapps
drwxrwxrwx   1 root root  4096 Dec 23 20:14 work
drwxr-xr-x   7 root root  4096 Dec 23 20:14 kafka-1.1.0.4
drwxr-xr-x  10 root root  4096 Dec 23 20:15 cassandra
drwxrwxrwx   1 root root  4096 Dec 23 20:15 temp
drwxrwxrwx   1 root root  4096 Dec 27 07:57 logs
drwxr-xr-x 387 root root 20480 Dec 27 19:04 kafka-data
root@65245723b156:/usr/local/tomcat# pwd
/usr/local/tomcat
root@65245723b156:/usr/local/tomcat#
```

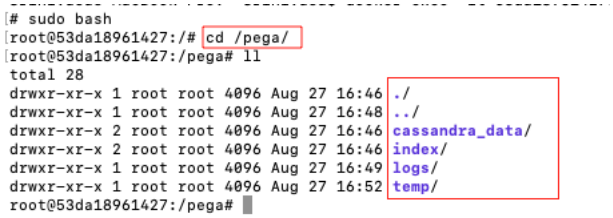
5. Access Pega using http://localhost:8080/prweb



Administrator@pega.com / install (or password@1)

6. All required folders for logs, temp… etc are available as below.





7. Happy adventure… 😊