**Lab 6**

Ziyang Lin zlin32@jhu.edu

**Setting-up**

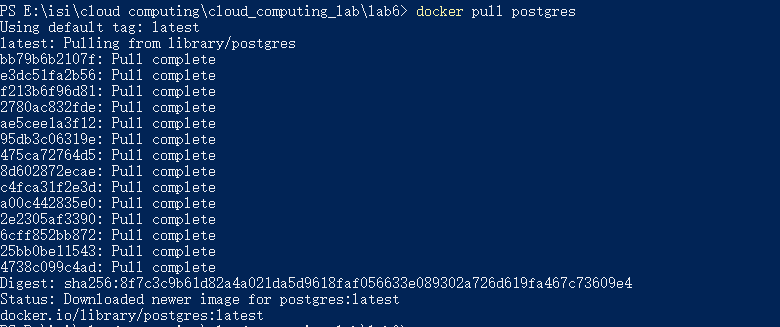
1. Create Docker network

Command: docker network create pgnet



1. Create Postgres Container
   1. Pull the Postgres image from repositories.

Command: docker pull postgres



* 1. Run container using Postgres image. Connect it to the network pgnet

Command: docker run –network pgnet –name pg1 \

-e POSTGRES\_PASSWORD=secret -d postgres



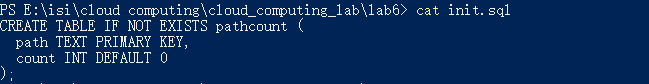
1. Init Postgres Container
   1. Find the IP address of postgres database in container.

Command: docker container inspect pg1 \

-f ‘{{.NetworkSettings.Networks.pgnet.IPAddress}}’



3.2 Use init.sql to build table in database.

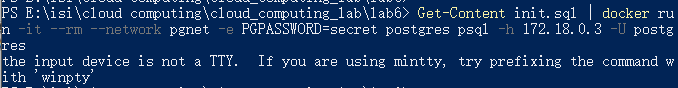


3.3 Use init.sql to initialize database.

Command: Get-Content init.sql | docker run -it –rm –network pgnet \

-e PGPASSWORD=secret postgres psql -h 172.18.0.3 -U postgres

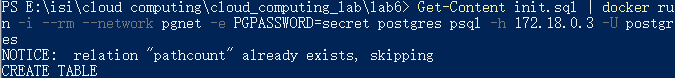
But I got an ‘Input device is not a TTY’ error.



Command: Get-Content init.sql | docker run -i –rm –network pgnet \

-e PGPASSWORD=secret postgres psql -h 172.18.0.3 -U postgres

Using -i can work.



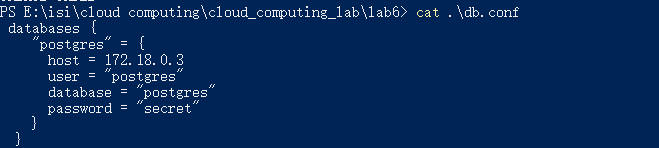
**Create sevice Container**

1. Package requirements

I wrote my web application by using Python and Flask. And I need to install Flask for creating application, psycopg2 for working with postgresql database, and pyhocon to handle environment variables.

1. Environment Variables

I defined all configuration variables in environment variables. And setting the variables in db.conf file, which will be loaded in application.

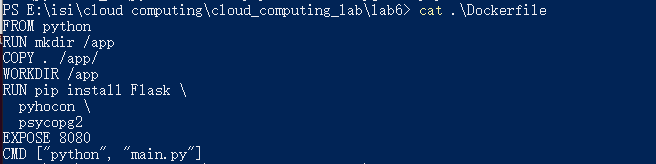


1. Code Files

I wrote application file in main.py and HTML file in index.html

1. Then I built my container using Dockerfile.

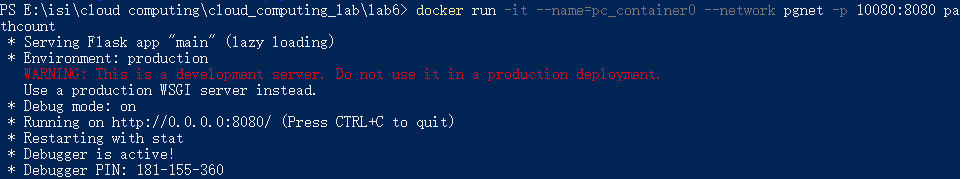
Command: docker build -t pathcount .



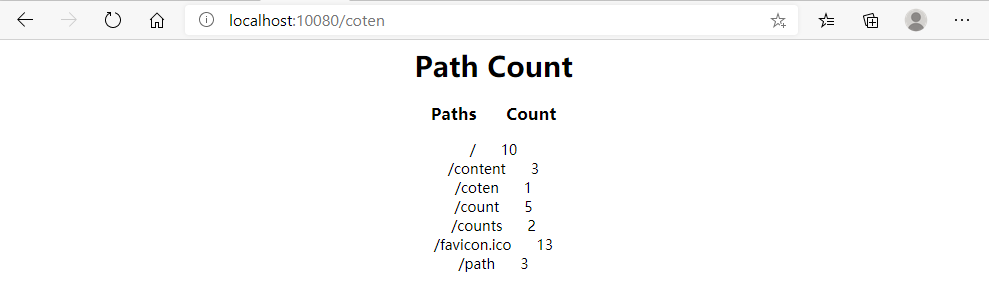


1. Then ran service container in port 10080.

Command: docker run -it --name=pc\_container0 –network pgnet -p 10080:8080 pathcount



1. Finally, go to localhost:10080 on browser and the page show like this.



**Questions**

* **Why did we create a special network instead of exposing the host network?**

It can isolate the database from host network, which can ensure data security.

* **Why didn’t we use exposed ports everywhere (that they exist)?**

It use docker bridge network to isolate the database in postgres container. If we use exposed ports, the database may be exposed in outside attacks.

* **What could happen if you didn’t use SQL parameters, but relied on string formatting for setting the path in your queries?**

The application will be easily attacked by SQL injection attacks.

* **Why is that particularly important in this setup? What makes those parameters potentially dangerous?**

The parameters didn’t be filtered and checked when running the container. So there are potential threats that parameter can be used in injection attacks.

* **The bridge network we define only works on a single host. What would you have to do to make these containers talk to each other if they were running on different host machines?**

If the containers are running on different host machines, I would use the host network like I did in this lab.

* **What parts of this did you wish were simpler? Which parts seemed unnecessarily difficult?**

The most difficult things I met in this lab is that I firstly tried to set up the container in VirtualBox, however I didn’t left enough disk space for this lab. So I have to delete file but still no enough space. I hope if possible we can put the space requirements in the lab document, then I will not waste so much time. And the

-it stuff in instrument couldn’t use in lab. I think that is unnecessarily difficult.