EaSy Delivery Setup Instructions

We will provide you with pre-registered accounts to speed up the demonstration process. Please use EaSy Delivery with these accounts.

Important Accounts for the demonstration:

|  |  |  |  |
| --- | --- | --- | --- |
| Email | Password | Role | Platform |
| [faithkoh1997@gmail.com](mailto:faithkoh1997@gmail.com) | FaithKoh1997! | User | Google |
| [slypoon@gmail.com](mailto:slypoon@gmail.com) | Ilovesmu!20 | Vendor | Facebook |
| [bentennisonrulez@gmail.com](mailto:bentennisonrulez@gmail.com) | bentennison | Driver | Facebook |

\*Note: Do note that during the demonstration, you can only purchase from HJFC. This is because there are no emails tagged with the other vendors. The Notification Microservice will not send messages to unregistered Vendors.

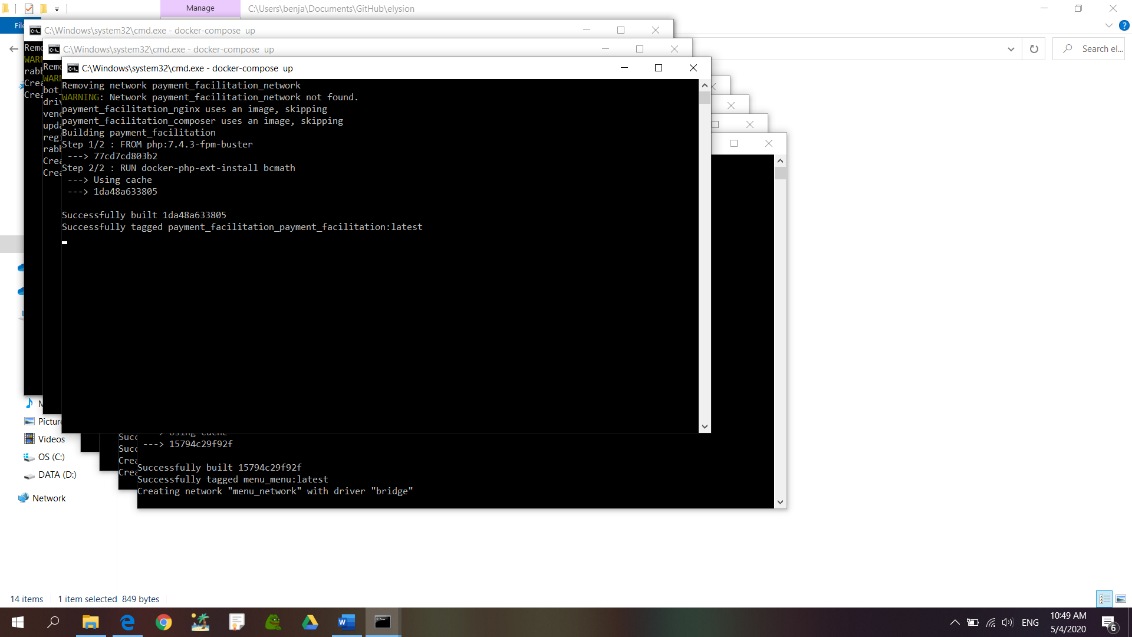
If you have Docker Desktop for Windows, Docker Compose comes with the installation. If you do not have compose, please install compose following this set of instructions: <https://docs.docker.com/compose/install/>.

Step 1 (Starting all the services):

1. In the submission folder, locate the Windows Command Script *"G3T4 EaSy Delivery Start Script.cmd"*. The script will start all the required processes. If you received a message stating that the script might put your PC at risk, do not be alarmed and simply click on 'more info' and select 'Run anyway'.
2. Double click the script to start Docker Compose. It should spawn 8 command prompt windows. Below is the rundown of each command prompt window.

* UI
* RabbitMQ Broker
* User Microservice
* Menu Microservice
* Notification Microservice
* Order Processing Microservice
* Recommendation Microservice
* Payment Facilitation Microservice

It should look something like this:



1. Allow sometime for the images to be pulled from Docker Hub and for the services to start. (This should take around 5 – 6 minutes)
2. We can identify each command prompt window by the name of the container running on the side show in the Standard Output. For example:



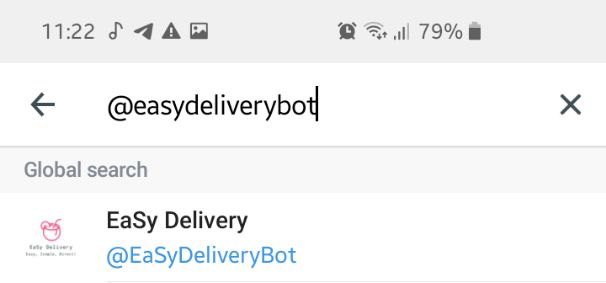
crm\_postgres\_database | shows this is the crm service

1. We know each service has started when we see the following messages in each command prompt window:

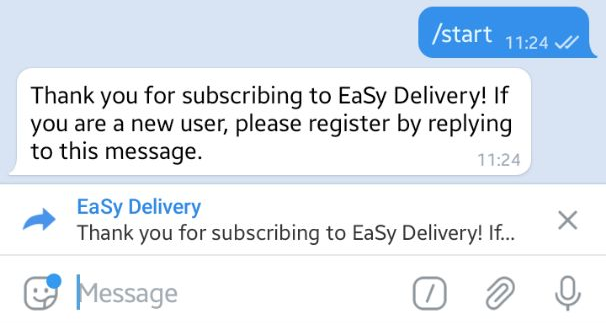
|  |  |
| --- | --- |
| CMD Window | Message |
| UI | *ui\_container | [05-Apr-2020 06:07:48] NOTICE: fpm is running, pid 1*  *ui\_container | [05-Apr-2020 06:07:48] NOTICE: ready to handle connections* |
| RabbitMQ  Broker | *rabbitmq\_broker | 2020-04-05 03:04:58.650 [info] <0.9.0> Server startup complete; 3 plugins started.*  *rabbitmq\_broker | \* rabbitmq\_management*  *rabbitmq\_broker | \* rabbitmq\_management\_agent*  *rabbitmq\_broker | \* rabbitmq\_web\_dispatch*  *rabbitmq\_broker | completed with 3 plugins.  \*Note: if you see authentication messages below, this means other services have successfully connected to the broker.* |
| User | *user\_postgres\_database | 2020-04-05 03:05:02.200 UTC [1] LOG: database system is ready to accept connections* |
| Menu | *menu\_postgres\_database | 2020-04-05 03:05:02.440 UTC [1] LOG: database system is ready to accept connections* |
| Notification | *updater | Connection Successful*  *updater | Updater has successfully started with no errors.*  \*Note: some containers may take longer to start than others, if there are no further messages seen in the CMD window it should signify all processes have started. |
| Order  Processing | *order\_microservice | 2020-04-09 14:53:35.508 INFO 1 --- [ntContainer#0-2] o.s.a.r.c.CachingConnectionFactory : Created new connection: rabbitConnectionFactory#4bb33f74:7/SimpleConnection@17c4097a [delegate=amqp://rabbit@192.168.65.2:5673/, localPort= 55142]* |
| Recommendation | *recommendation\_microservice | [2020-04-05 03:04:22 +0000] [7] [INFO] Booting worker with pid: 7* |
| Payment  Facilitation | *payment\_facilitation\_microservice | [05-Apr-2020 03:05:12] NOTICE: ready to handle connections*  \*Note: If you see payment\_facilitation\_composer exiting with code 0, this is expected, composer exists to install the library dependencies and will exit once it’s done. |

Step 2 (Telegram Bot Registration):

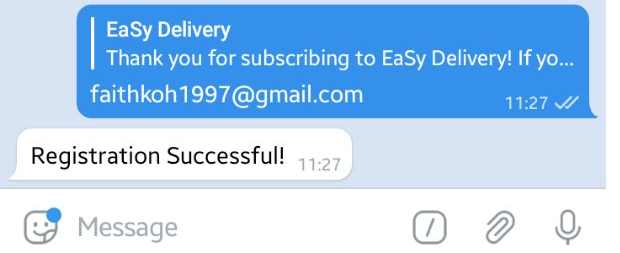
1. You will need 3 separate telegram accounts for this, each telegram account is for each user shown in Page 1.
2. Find the Bot @easydeliverybot.



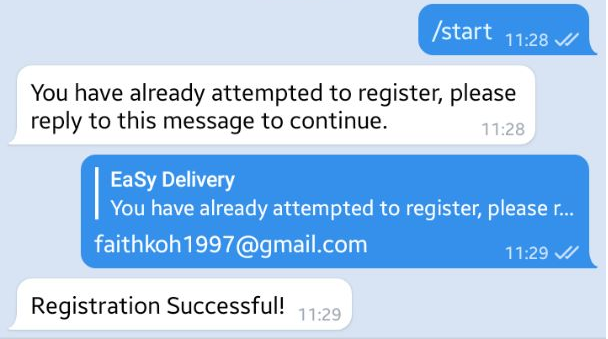
1. Start the registration process by pressing /start. You will receive this message:



1. Reply to the Message by keying in each of the emails show in page 1. For this set of instructions, we will use [faithkoh1997@gmail.com](mailto:faithkoh1997@gmail.com). After keying in a valid email, the bot will respond with “Registration Successful!”



\*Note: If you do not reply to the message, you will not get a response. Please type /start to restart the registration process:



After testing out User Scenarios, the steps below are for taking down the microservices:

Step 1 (Stopping the services):

1. In the submission folder, locate the Windows Command Script *"G3T4 EaSy Delivery Stop Script.cmd"*. The script will stop all the required processes. If you received a message stating that the script might put your PC at risk, do not be alarmed and simply click on 'more info' and select 'Run anyway'.
2. Double click the script to stop all processes using docker-compose down. It should spawn 8 command prompt windows. Each window will exit upon a complete execution.

Alternatively, you might want to manually terminate each process:

1. On each command prompt window, press Cntrl + C. You should see a message:

Gracefully stopping…

If it does not show, press Cntrl + C again.

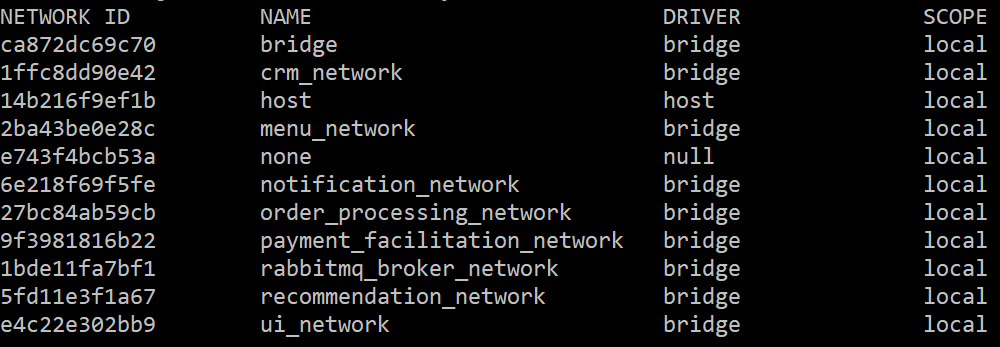
To know the service has stopped, you should see a green … done message.

1. On each command prompt widow, enter the command:

docker-compose down

This will remove all stopped containers pertaining to the service.

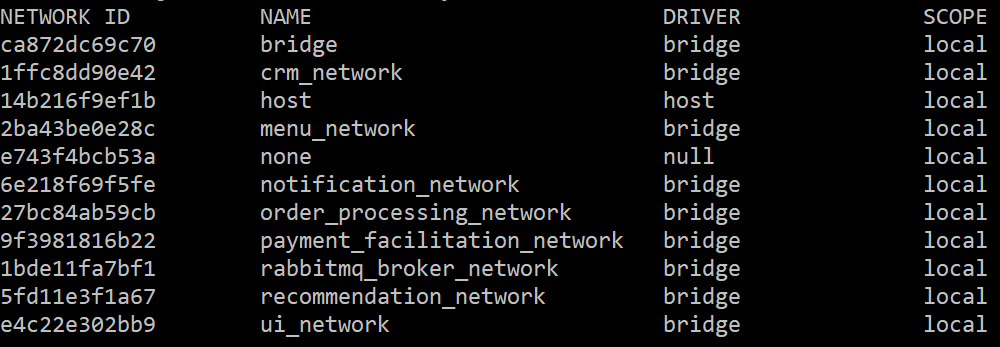
1. Find the images we used for the demonstration by entering the command:

docker network ls

1. Remove each image by entering the command:

docker rmi <image id>

1. Find the networks we used for the demonstration by entering the command:

docker network ls

1. Remove each network (except none, bridge and host) by entering the command:

docker network rm <image id>