Build, Run & Ship Containerized Applications with Docker Datacenter

Launch & Learn Hands-on Lab Manual

This Launch & Learn hands-on lab will give you a brief introduction to building, shipping and deploying containerized applications on Azure with Docker Datacenter. This fully integrated solution stack is based on ARM templates that customers can deploy on Azure with the simple click of a button.

Introduction

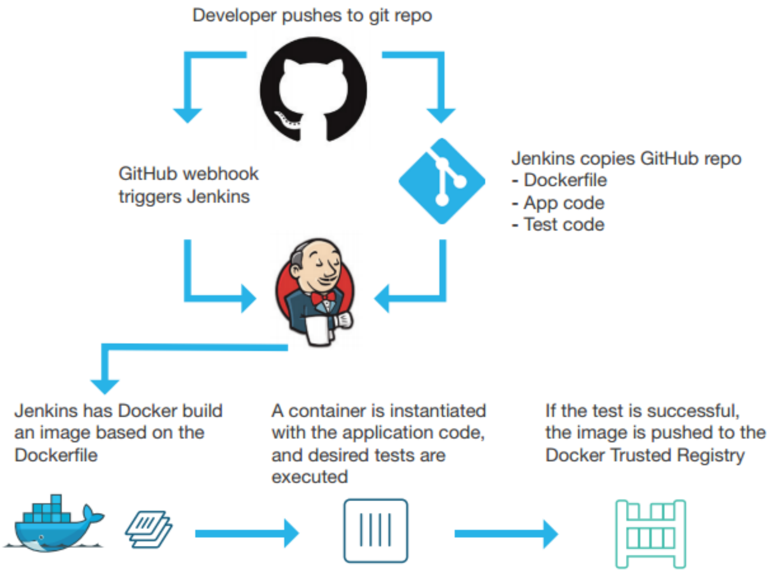
The **Build, Run & Ship Containerized Applications with Docker Datacenter** Azure Partner Quickstart template launches Docker Datacenter, an integrated platform that enables agile application development and management.

What You Will Learn in this Session

In this Launch & Learn session, you will learn how to leverage the Jenkins Continuous Integration/Continuous Deployment (CI/CD) platform to build and push a container-based application into the Docker Trusted Registry (DTR). Once the images are pushed, you will then leverage the Docker Universal Control Plane (UCP) to compose the application and run it on Docker Datacenter. You will also be able to scale up the application and see how Docker Datacenter seamlessly scales your application.

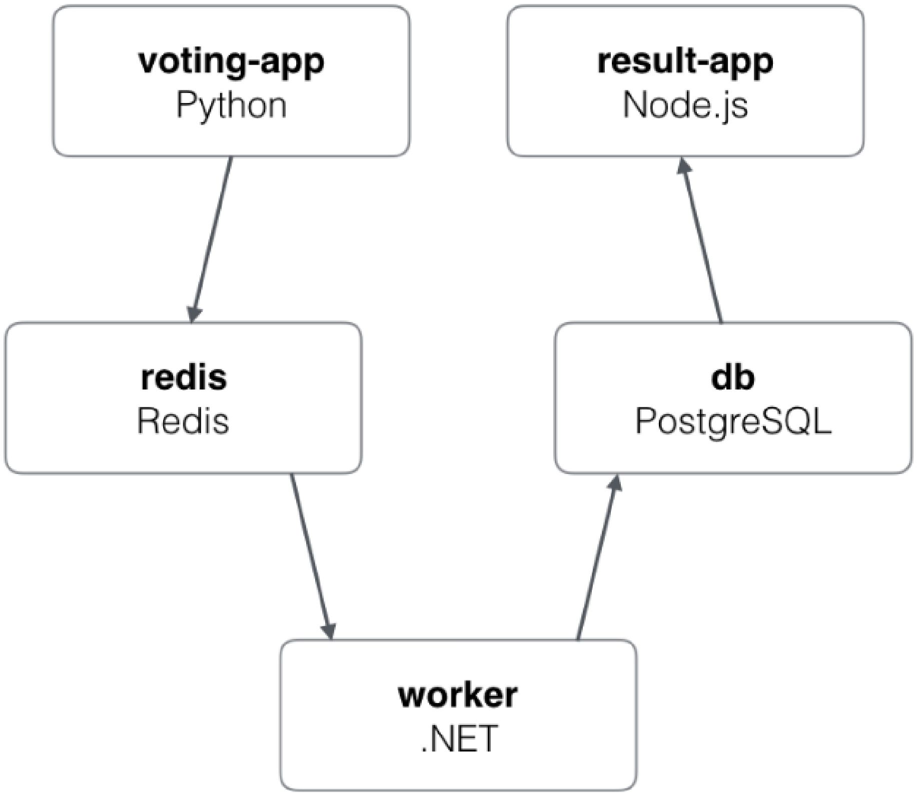
CI & CD Workflow

The diagram below shows a typical developer/operations (DevOps) workflow using Docker Datacenter.


Developer pushes to git repo
GitHub webhook triggers Jenkins
Jenkins copies GitHub repo
- Dockerfile
- App code
- Test code
Jenkins has Docker build an image based on the Dockerfile
A container is instantiated with the application code, and desired tests are executed
If the test is successful, the image is pushed to the Docker Trusted Registry

Sample Application

- To better understand the Continuous Integration and Continuous Deployment (CI/CD) process, we will use the Sample Voting Application that has been pulled from GitHub.
- The Components of the Sample Application are shown below:



- **Voting App:** A Python web-app, which lets you vote between two options.
- **Caching App:** A Redis queue, which collects new votes.
- **Worker App:** A .NET worker, which consumes votes and stores them in a Postgres database.
- **Results App:** A Node.js web-app, which shows the results of the voting in real-time.

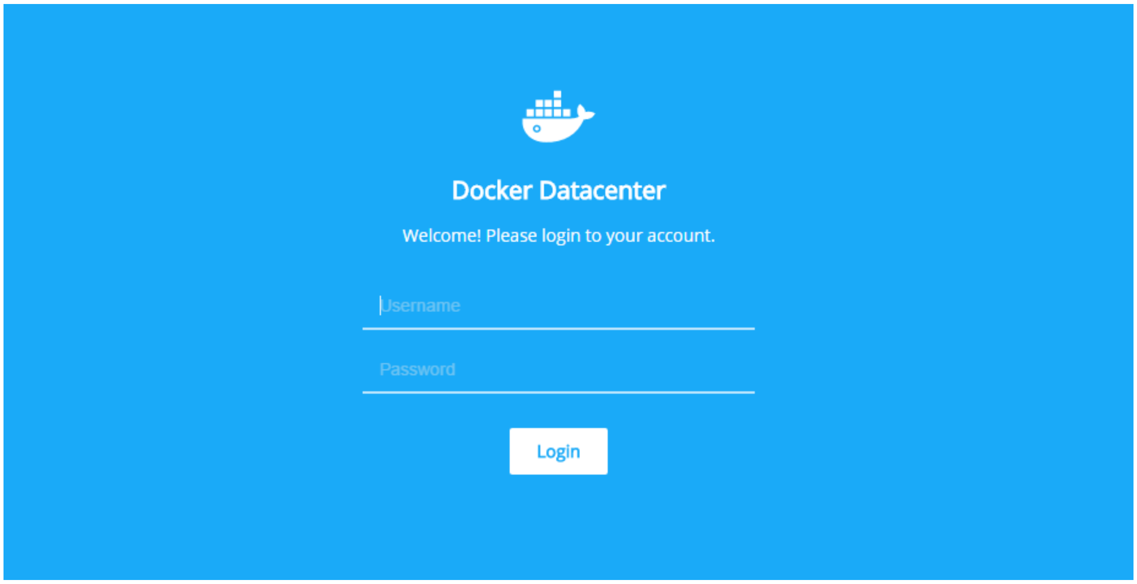
Lab Steps

1. The instructor, assuming the role of a developer, will show how to fork the Sample Voting Application from the Git repository. The instructor will then use the CI feature of Cloudbees Jenkins to build and push (publish) the Docker images of the Voting Application to the DTR. The Developer will also push the Docker Compose YAML file into the DTR.

   As the IT Operations person, you will now use Docker Trusted Registry to view the images and share compliance, and then use the YAML file to compose the application and deploy it to Docker Datacenter using Universal Control Plane.

2. Login to Docker Trusted Registry and verify that the Docker images published by the developer are available.

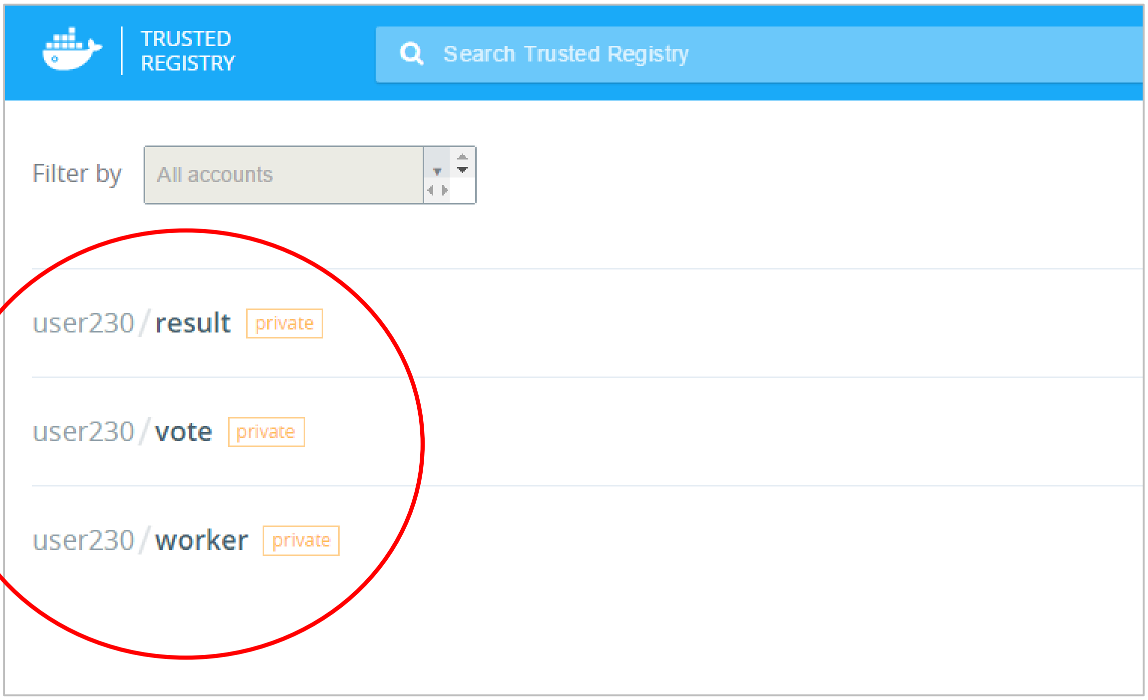
   - Open a web browser and go to the following URL:
   - http://bit.ly/2dBxbWW
   - In this file, you will see a list of sections for Docker Trusted Registry, Docker Universal Control Plane, etc.
   - Go to the **DTR section**, copy the **DTR URL**, and open it in a web browser window/tab. The DTR login page will look like the page below:



Docker Datacenter

Welcome! Please login to your account.

- Enter your User name and the password for the Docker Trusted Registry at the login page. You can find the User name & password in the same DTR section mentioned before – replace "**user2xx**" with your assigned User name. (Example: "user230")

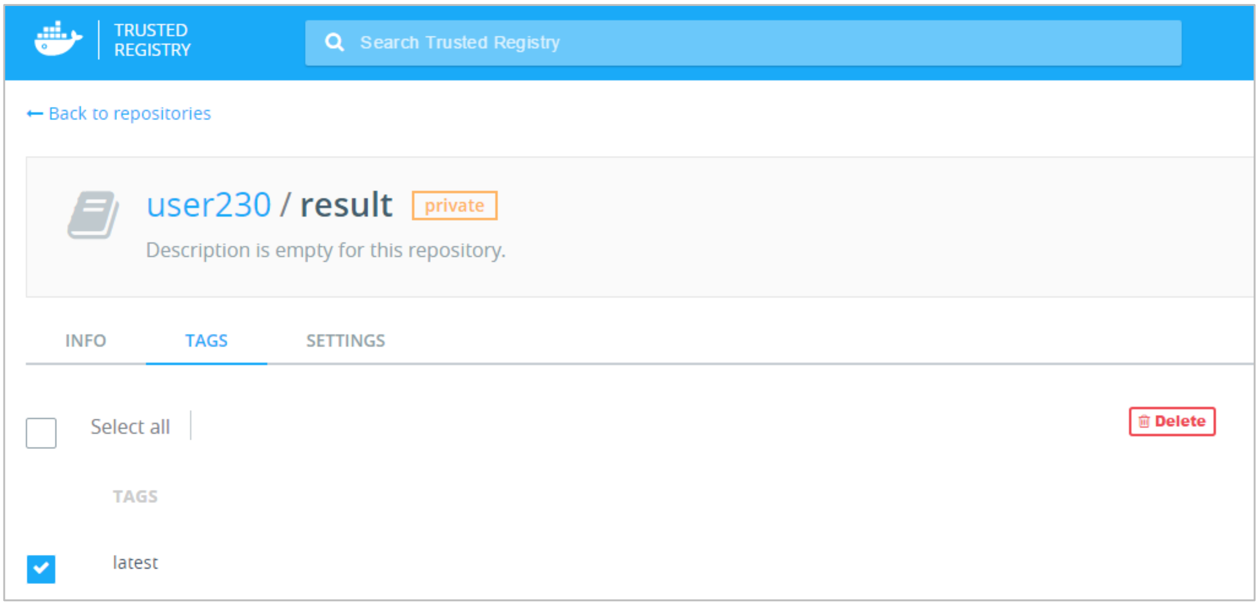


user230 / result private

user230 / vote private

user230 / worker private

- Once you have logged in, the screen should look like the above image.
- The circled sections are the images from the **Sample Voting Application**.
- Select one of the images, and it will show more details about that image, such as its tags, permissions, and settings – as shown below:



user230 / result private

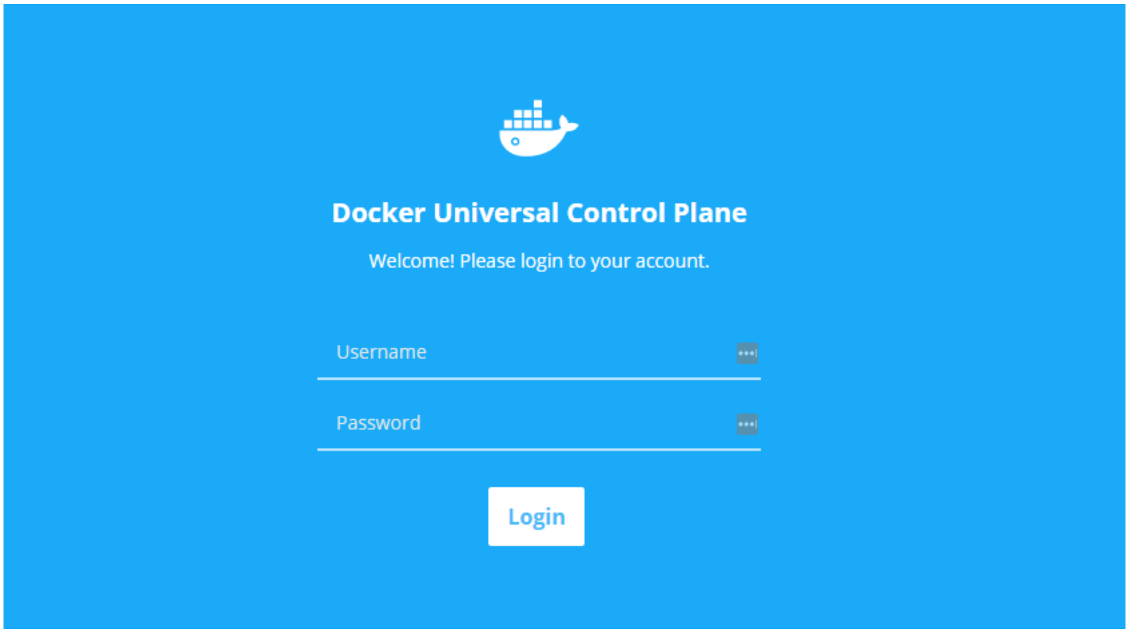
Description is empty for this repository.

☑ latest

Now that you've seen these images in DTR, we will see how these images will be deployed as a containerized application in Docker Datacenter in the next section.

3. Login to the **Docker Datacenter Universal Control Plane** and use a Docker Compose file to deploy the application.

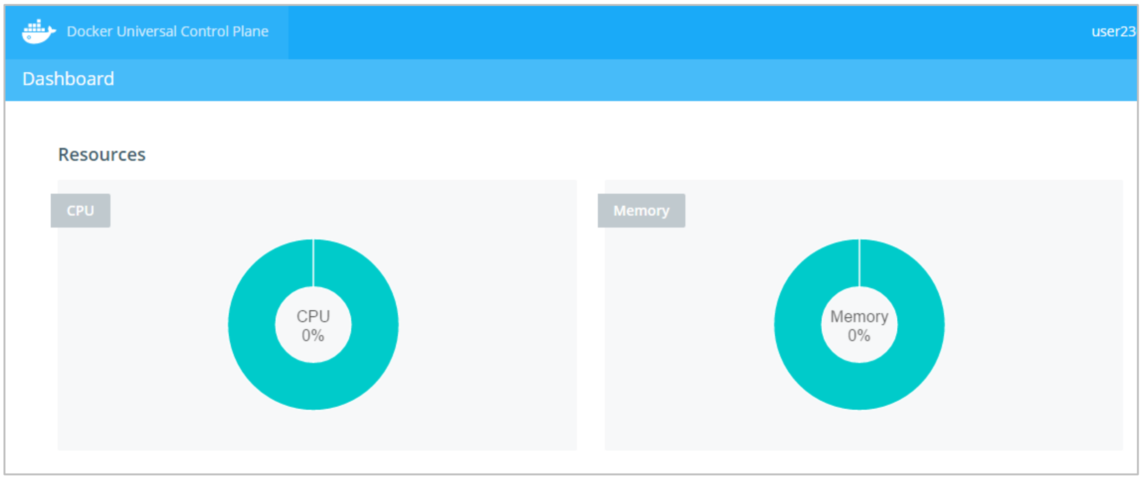
- Return to the open GitHub file, and navigate to the **UCP section** and copy the UCP URL.
- Open the UCP URL in a new browser tab/window. The UCP login page will look like the image below:



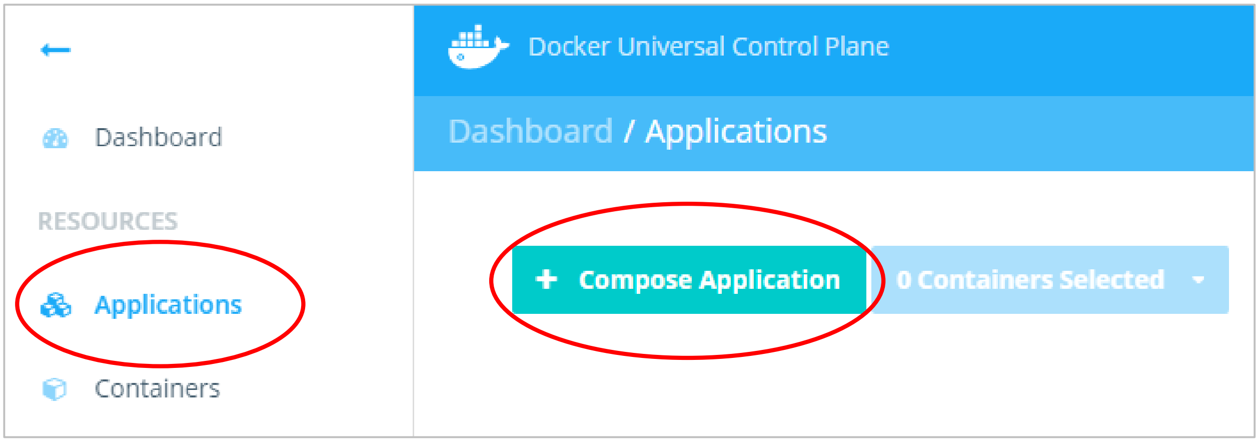
Docker Universal Control Plane

Welcome! Please login to your account.

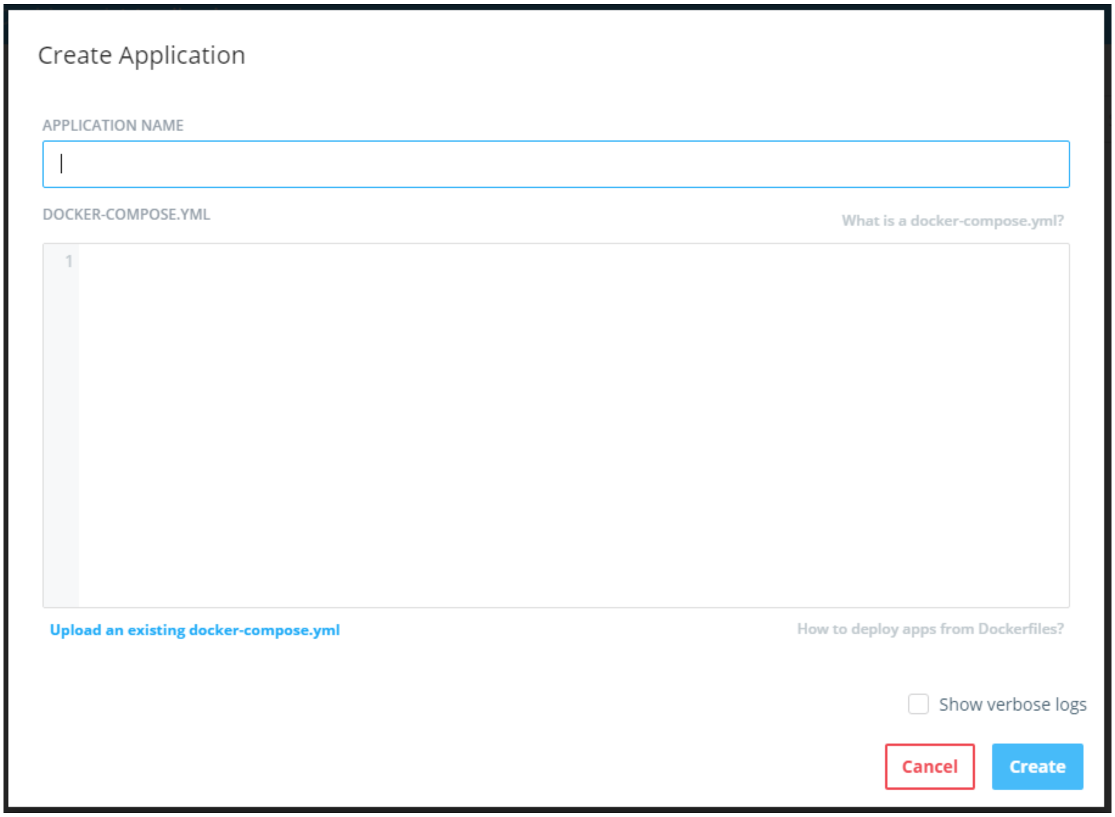
- Once you have logged in to the UCP, the screen should look like the image below:



- Now that you have logged in to the UCP, you will now deploy the **Sample Voting Application**.
- Select '**Applications'** from the left side navigation menu in the UCP window. Then select the '**Compose Application**' button.



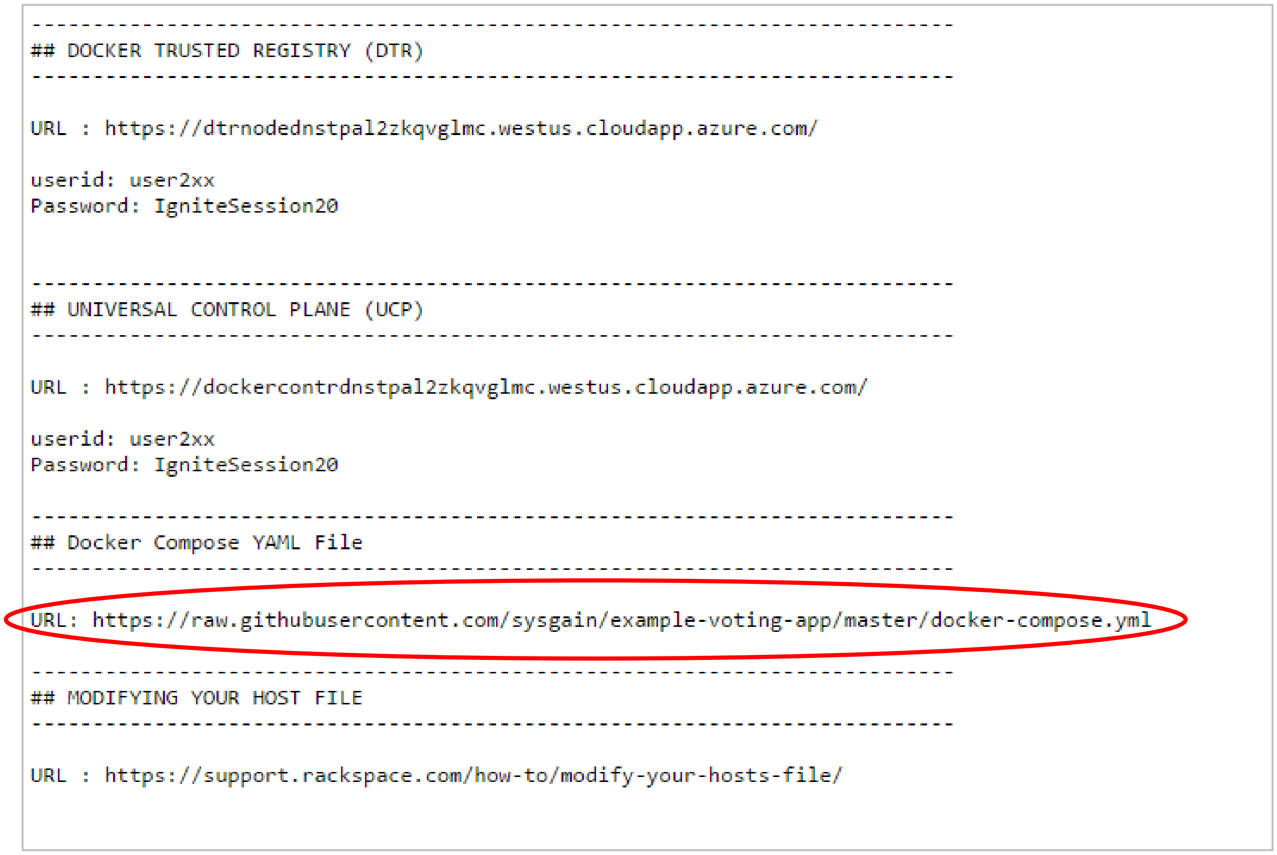
- The **Create Application** dialog box should appear, as shown below:


Create Application
APPLICATION NAME
DOCKER-COMPOSE.YML
What is a docker-compose.yml?
1
Upload an existing docker-compose.yml
How to deploy apps from Dockerfiles?
□ Show verbose logs
Cancel
Create

Keep this browser tab open, as we will return to it in the next section.

4. Return to the GitHub URL and go to the **Docker Compose YAML File** section. Copy the contents of the **docker-compose.yml** file to compose and deploy the application.

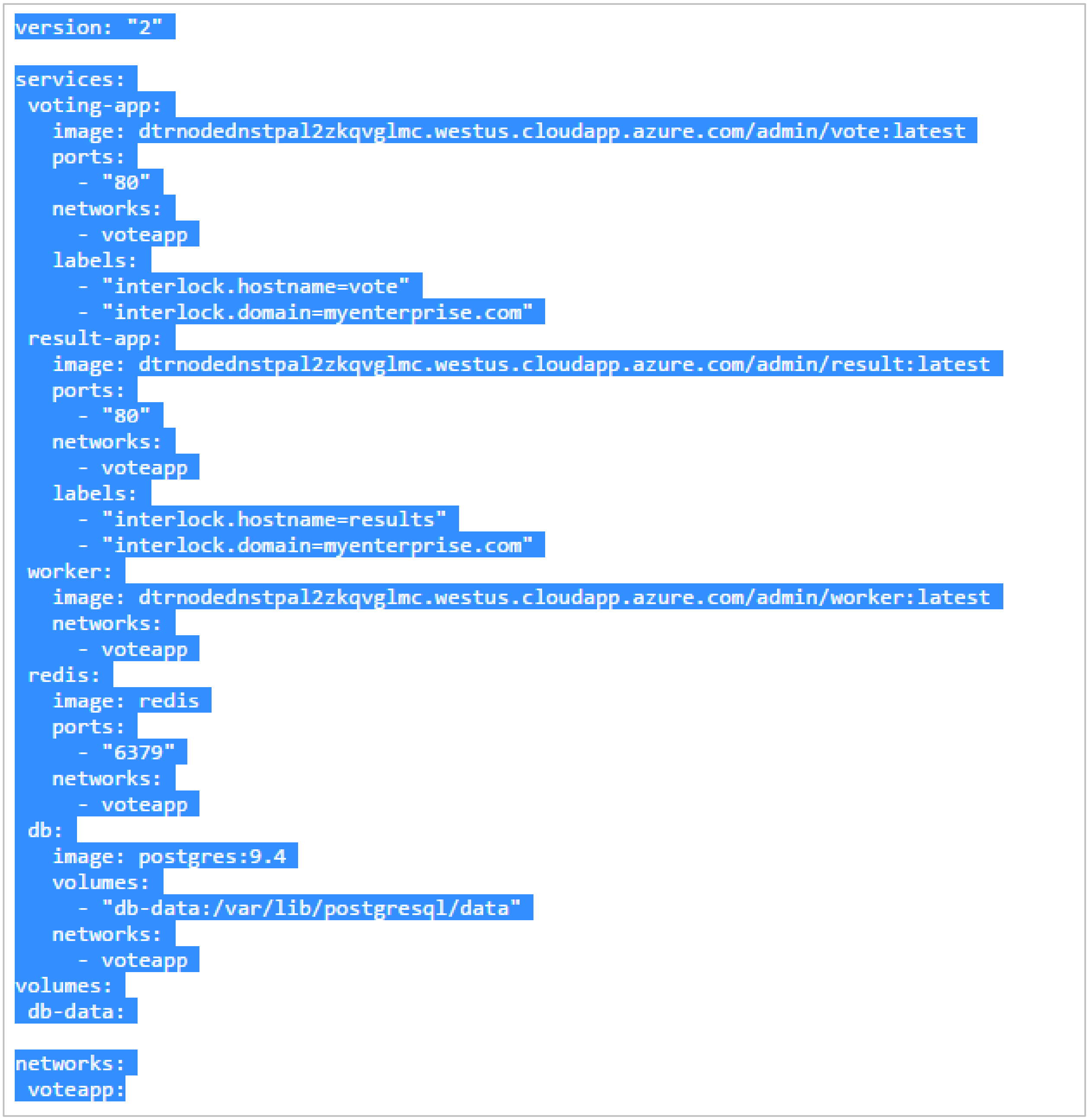
   - Return to the GitHub file you opened in Step 2, navigate to the **Docker Compose YAML File** section and copy the URL, as shown below:



```
---------------------------------------------------------------------------
## DOCKER TRUSTED REGISTRY (DTR)
---------------------------------------------------------------------------

URL : https://dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/

userid: user2xx
Password: IgniteSession20


---------------------------------------------------------------------------
## UNIVERSAL CONTROL PLANE (UCP)
---------------------------------------------------------------------------

URL : https://dockercontrdnstpal2zkqvglmc.westus.cloudapp.azure.com/

userid: user2xx
Password: IgniteSession20

---------------------------------------------------------------------------
## Docker Compose YAML File
---------------------------------------------------------------------------

URL: https://raw.githubusercontent.com/sysgain/example-voting-app/master/docker-compose.yml

---------------------------------------------------------------------------
## MODIFYING YOUR HOST FILE
---------------------------------------------------------------------------

URL : https://support.rackspace.com/how-to/modify-your-hosts-file/
```

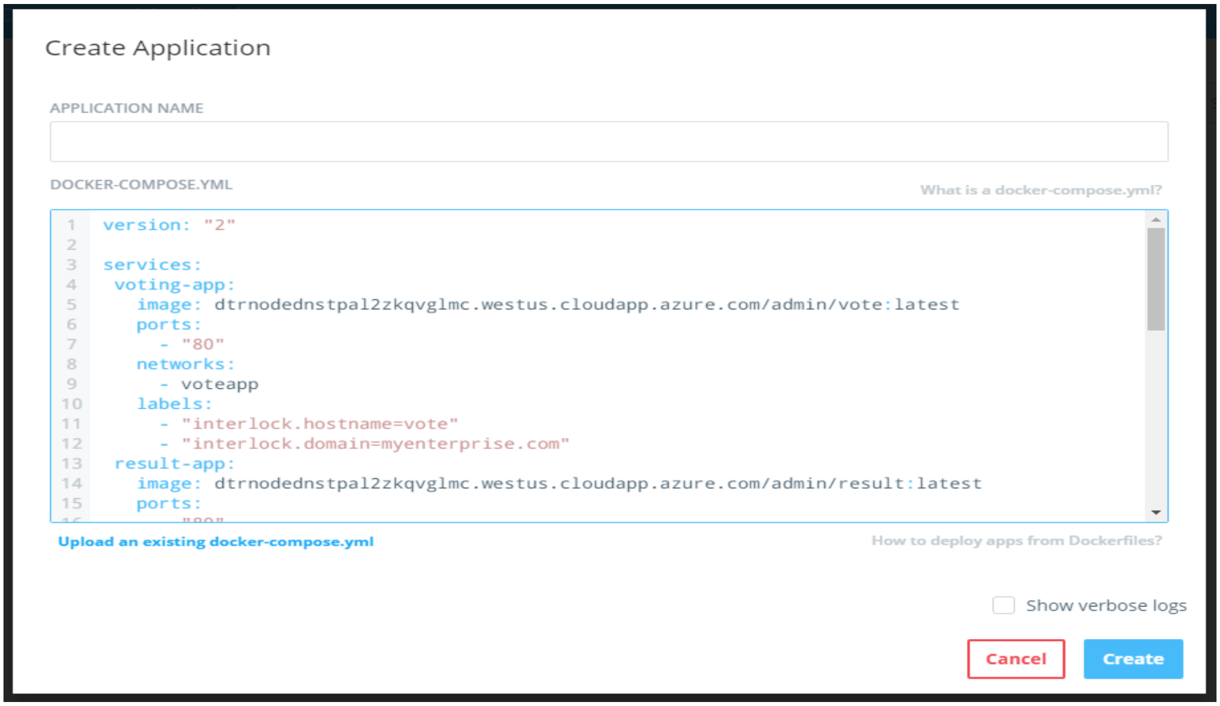
   - Open the URL in a new browser window/tab. This should show the docker-compose.yml file contents.

- Copy the entire contents of the **docker-compose.yml** file, as shown below:



```
version: "2"

services:
 voting-app:
   image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/admin/vote:latest
   ports:
     - "80"
   networks:
     - voteapp
   labels:
     - "interlock.hostname=vote"
     - "interlock.domain=myenterprise.com"
 result-app:
   image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/admin/result:latest
   ports:
     - "80"
   networks:
     - voteapp
   labels:
     - "interlock.hostname=results"
     - "interlock.domain=myenterprise.com"
 worker:
   image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/admin/worker:latest
   networks:
     - voteapp
 redis:
   image: redis
   ports:
     - "6379"
   networks:
     - voteapp
 db:
   image: postgres:9.4
   volumes:
     - "db-data:/var/lib/postgresql/data"
   networks:
     - voteapp
volumes:
 db-data:

networks:
 voteapp:
```

- Then, return to the UCP window/tab in your web browser and paste the copied docker-compose.yml file code into the window as shown below (next page):



Create Application

APPLICATION NAME

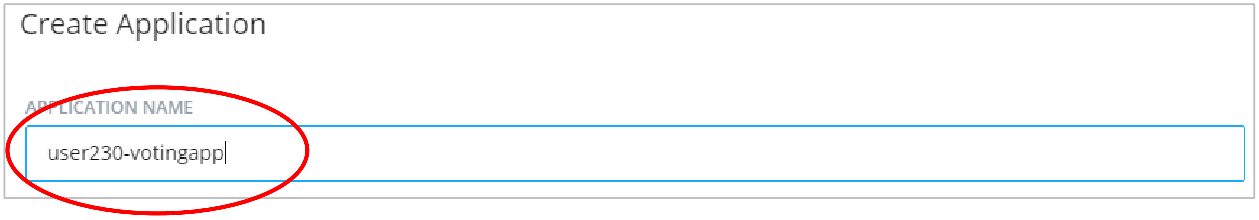
DOCKER-COMPOSE.YML

What is a docker-compose.yml?

```
1  version: "2"
2
3  services:
4   voting-app:
5     image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/admin/vote:latest
6     ports:
7       - "80"
8     networks:
9       - voteapp
10    labels:
11      - "interlock.hostname=vote"
12      - "interlock.domain=myenterprise.com"
13  result-app:
14    image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/admin/result:latest
15    ports:
```

Upload an existing docker-compose.yml
How to deploy apps from Dockerfiles?
□ Show verbose logs
Cancel
Create

- For the **Application Name** field, enter a unique name with the following format, as shown below:
  - [your_username-app_name]

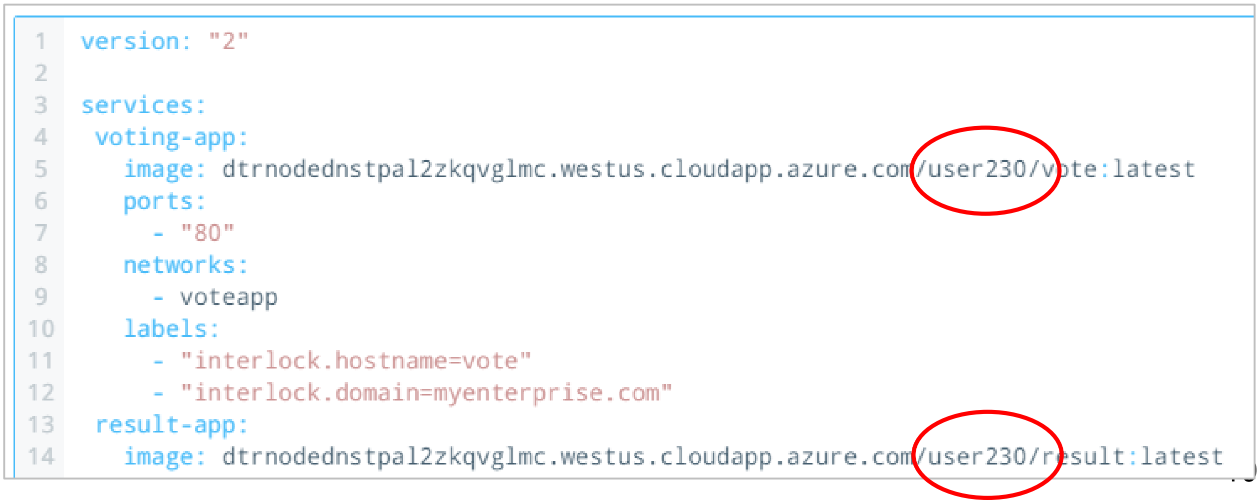


Create Application

APPLICATION NAME

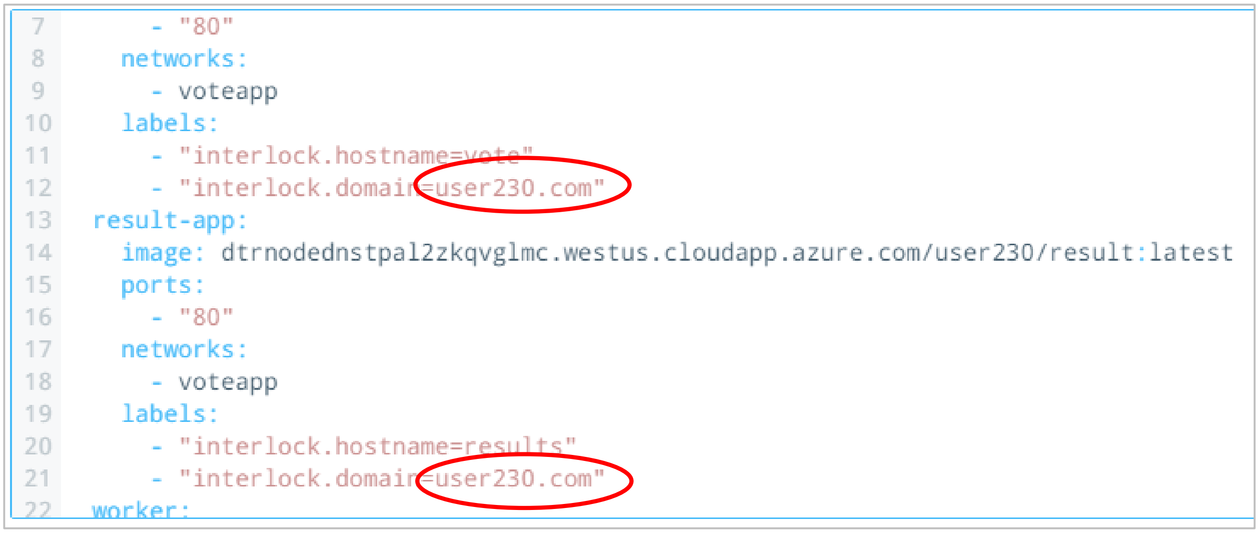
user230-votingapp

- Go to lines **5, 14 and 23** in the docker-compose.yml window and replace the user "**admin**" with your own username, as shown below:



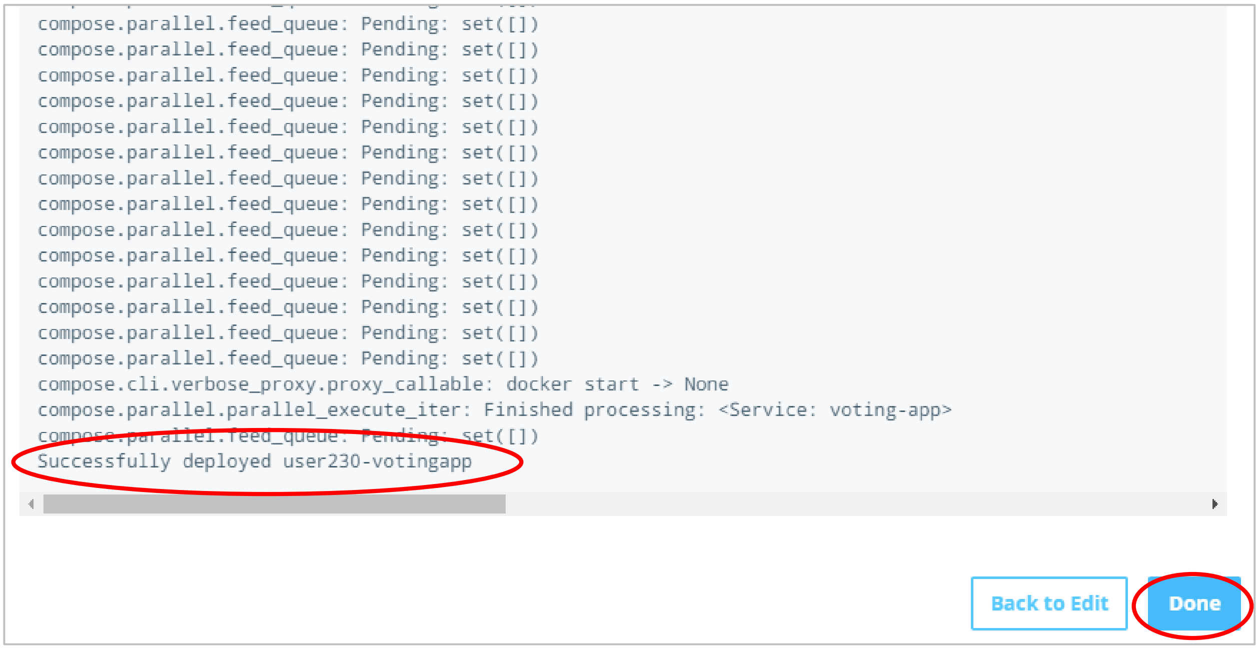
```
1  version: "2"
2
3  services:
4   voting-app:
5     image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/vote:latest
6     ports:
7       - "80"
8     networks:
9       - voteapp
10    labels:
11      - "interlock.hostname=vote"
12      - "interlock.domain=myenterprise.com"
13  result-app:
14    image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/result:latest
```

- Go to lines **11, 12, 20 and 21** in the docker-compose.yml window and rename the interlock.domain values as **[username].com**, as shown below:



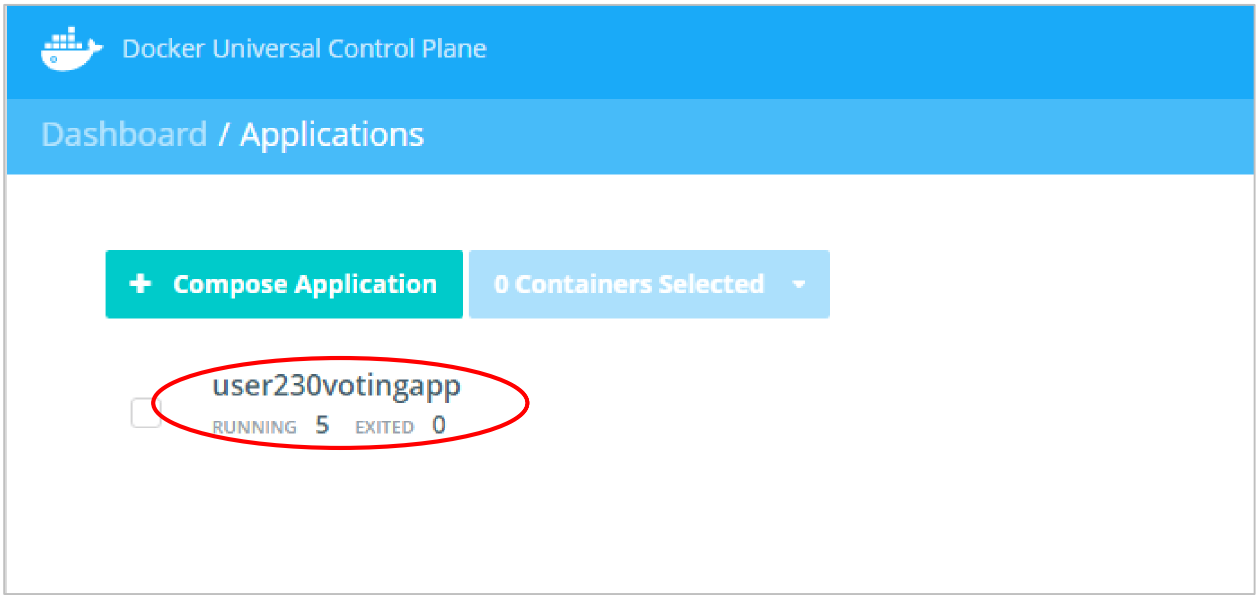
```
7        - "80"
8      networks:
9        - voteapp
10     labels:
11       - "interlock.hostname=vote"
12       - "interlock.domain=user230.com"
13   result-app:
14     image: dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/result:latest
15     ports:
16       - "80"
17     networks:
18       - voteapp
19     labels:
20       - "interlock.hostname=results"
21       - "interlock.domain=user230.com"
22   worker:
```

- Enable the **Show Verbose Logs** checkbox near the bottom of the dialog box, and click **Create** to deploy the application.
- Once the deployment succeeds (it may take a few minutes) you will now see that your application deployed successfully, as shown below:



```
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.parallel.feed_queue: Pending: set([])
compose.cli.verbose_proxy.proxy_callable: docker start -> None
compose.parallel.parallel_execute_iter: Finished processing: <Service: voting-app>
compose.parallel.feed_queue: Pending: set([])
Successfully deployed user230-votingapp
```

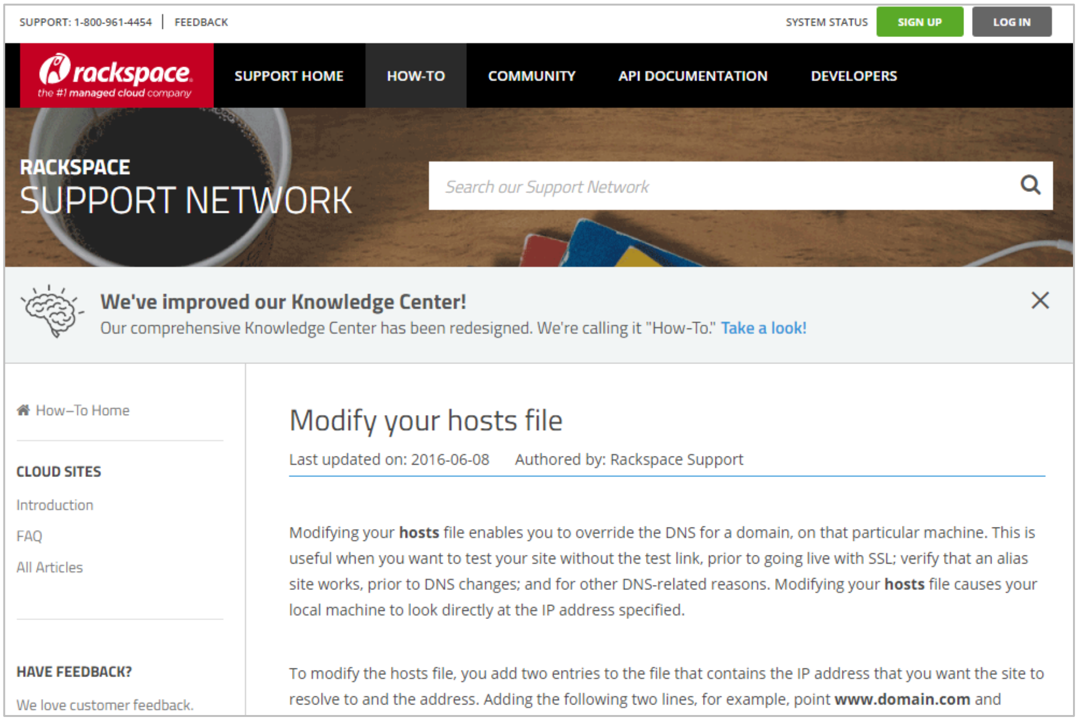
- Click the **Done** button to see your application deployed in the Universal Control Plane, as shown below:



user230votingapp
RUNNING 5 EXITED 0

5. Set up a local host to actually see the results in a browser using the interlock and hosting domains we changed in the docker-compose.yml file earlier. This sample application will have one URL to cast votes and a separate URL to see the results.

- Return to the open GitHub file and navigate to the **Modify Hosts** section and copy the URL.
- Open the Modify Hosts URL in a new browser window/tab, as shown in the image below (next page):



RACKSPACE
SUPPORT NETWORK

We've improved our Knowledge Center!
Our comprehensive Knowledge Center has been redesigned. We're calling it "How-To." Take a look!

HAVE FEEDBACK?
We love customer feedback.

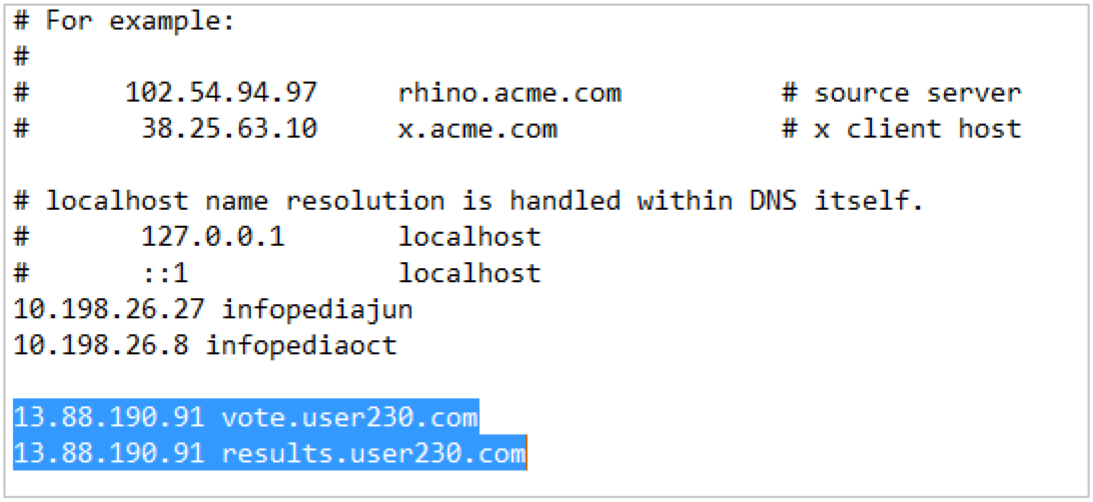
Modify your hosts file

Last updated on: 2016-06-08 Authored by: Rackspace Support

Modifying your **hosts** file enables you to override the DNS for a domain, on that particular machine. This is useful when you want to test your site without the test link, prior to going live with SSL; verify that an alias site works, prior to DNS changes; and for other DNS-related reasons. Modifying your **hosts** file causes your local machine to look directly at the IP address specified.

To modify the hosts file, you add two entries to the file that contains the IP address that you want the site to resolve to and the address. Adding the following two lines, for example, point **www.domain.com** and

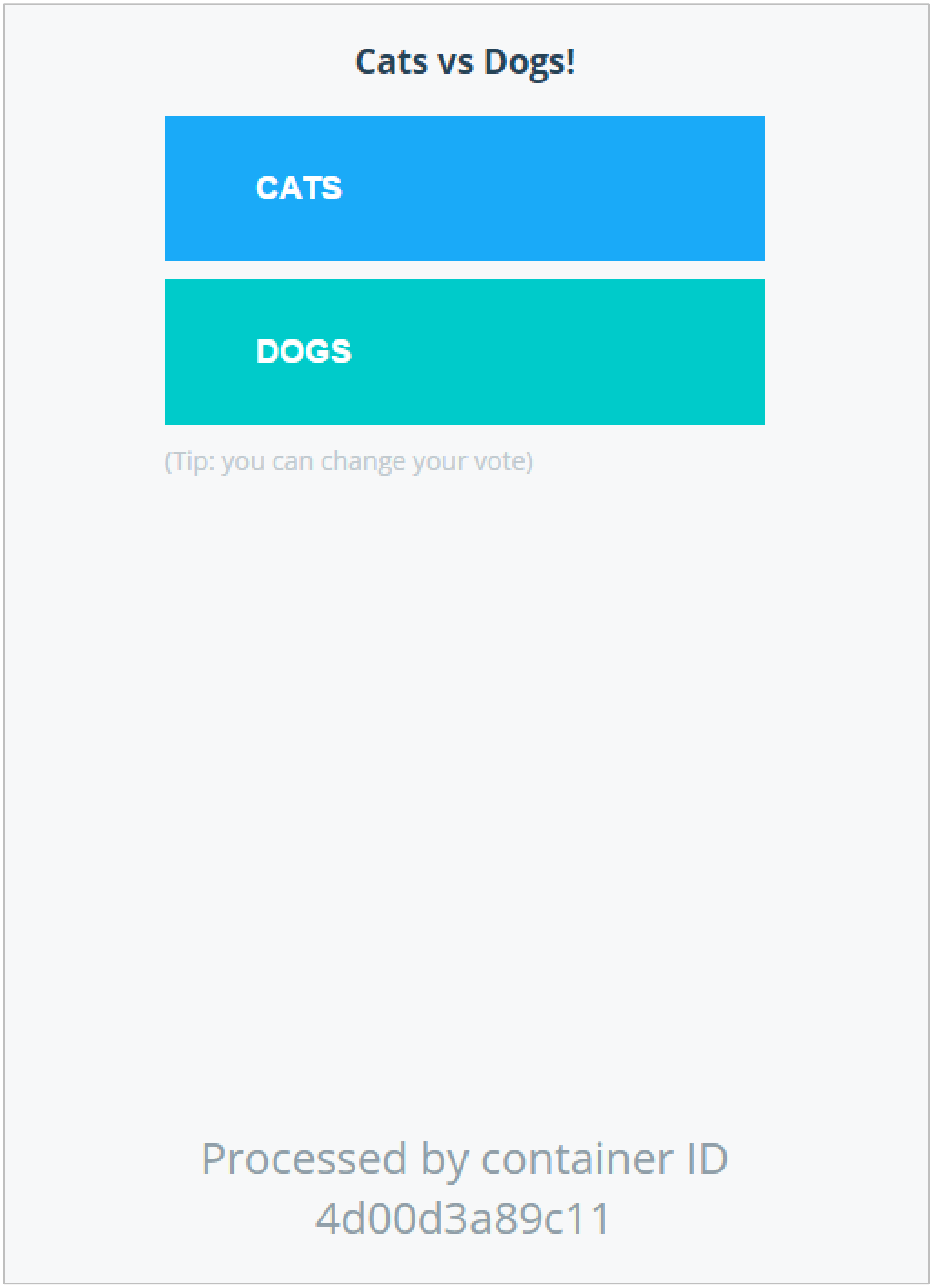
- Follow the instructions on the page as described to modify your host file corresponding to your operating system.
- Add the following IP Address and Hostname to the host file:
  - **13.88.190.91 vote.[username].com**
  - **13.88.190.91 results.[username].com**
- Once you add these to your host file, **Save** and **Exit**. Before saving, your notepad file should look like the image below:



```
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost
10.198.26.27 infopediajun
10.198.26.8 infopediaoct

13.88.190.91 vote.user230.com
13.88.190.91 results.user230.com
```

6. Open up the newly created Voting App and Results App in a web browser/tab to see the deployed application in action.

   - Open up a new browser window/tab and enter the address for the **Voting App** that you added to your host file. **For example: vote.user230.com.**
   - You should see the running **Voting Application**, as shown below (next page):



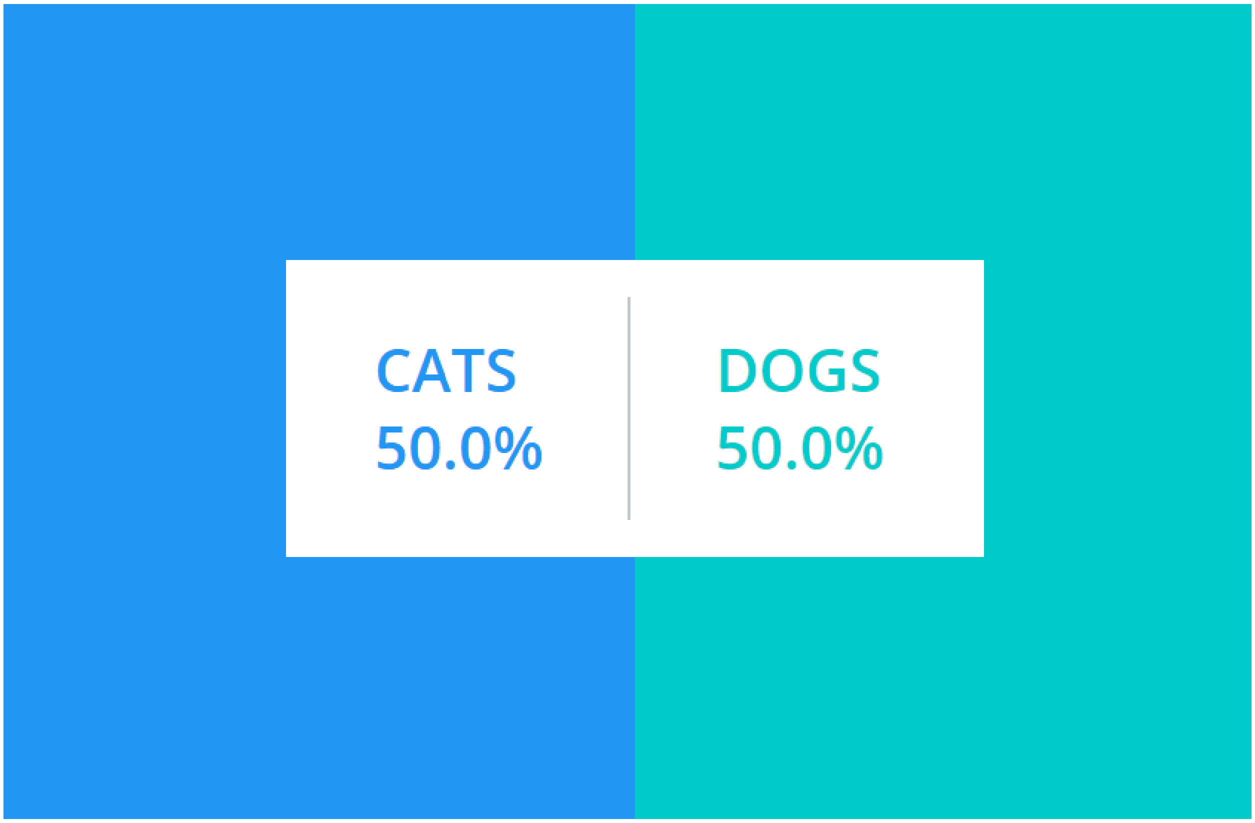
Cats vs Dogs!

(Tip: you can change your vote)

Processed by container ID 4d00d3a89c11

- Open up a new browser window/tab and enter the address for the **Results App** that you added to your host file. **For example: results.user230.com.**

- You should see the running **Results Application**, as shown below:



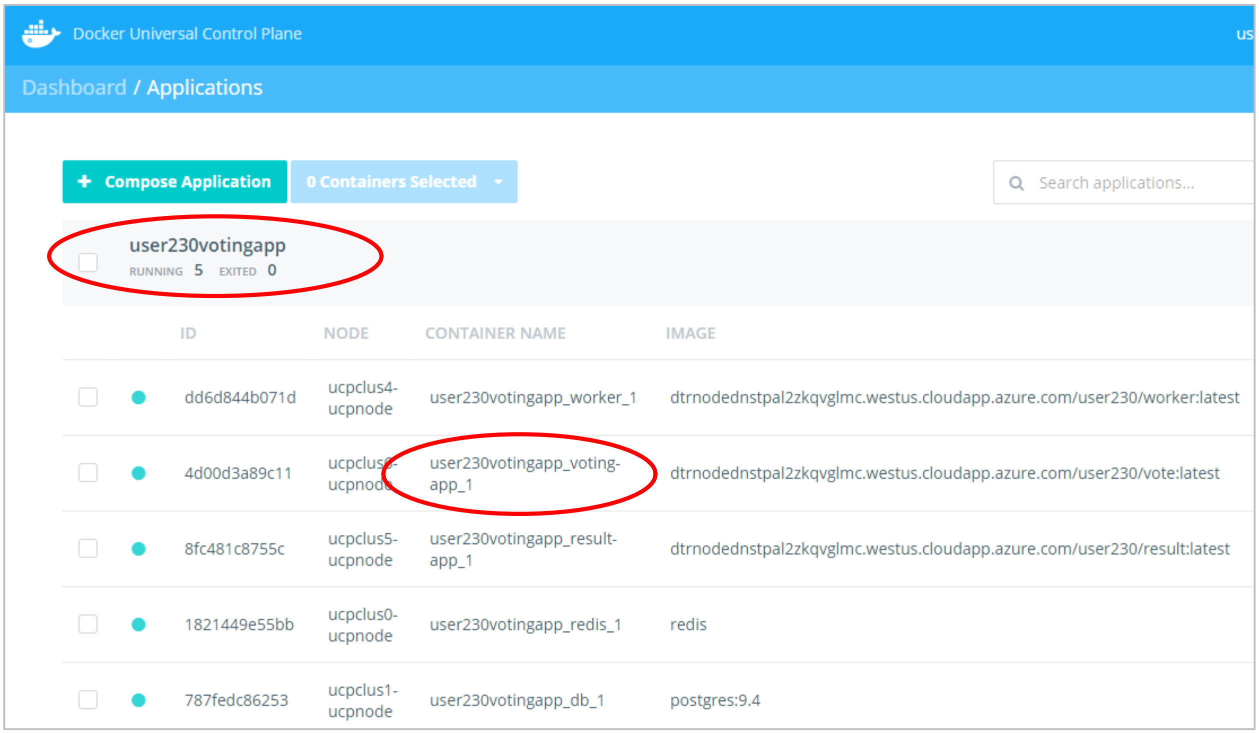
CATS 50.0%

DOGS 50.0%

- Congratulations! You now have both the Voting and Results applications running.

7. Scale up the Voting App to 2 containers, because we believe we need additional resources to handle the load.

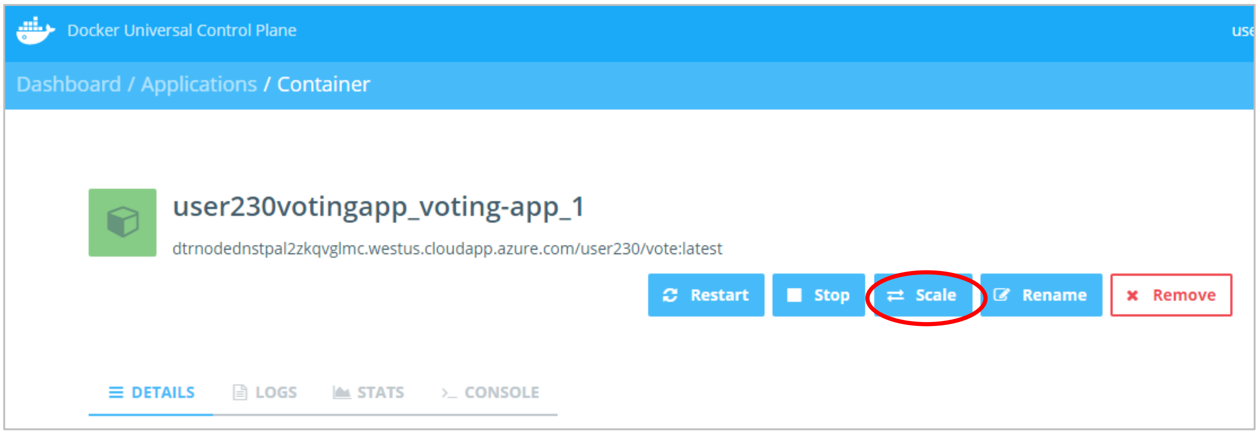
    - Return to the UCP web page and select '**Applications'** from the left side navigation menu.
    - Expand the list of running containers by clicking on the running **application** you created previously, then select the **Voting App container**, as shown below (next page):



user230votingapp
RUNNING 5 EXITED 0

		ID	NODE	CONTAINER NAME	IMAGE
□	●	dd6d844b071d	ucpclus4-ucpnode	user230votingapp_worker_1	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/worker:latest
□	●	4d00d3a89c11	ucpclus2-ucpnode	user230votingapp_voting-app_1	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/vote:latest
□	●	8fc481c8755c	ucpclus5-ucpnode	user230votingapp_result-app_1	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/result:latest
□	●	1821449e55bb	ucpclus0-ucpnode	user230votingapp_redis_1	redis
□	●	787fedc86253	ucpclus1-ucpnode	user230votingapp_db_1	postgres:9.4

- Once you select the voting app form the list of containers, you should see the page as shown below:



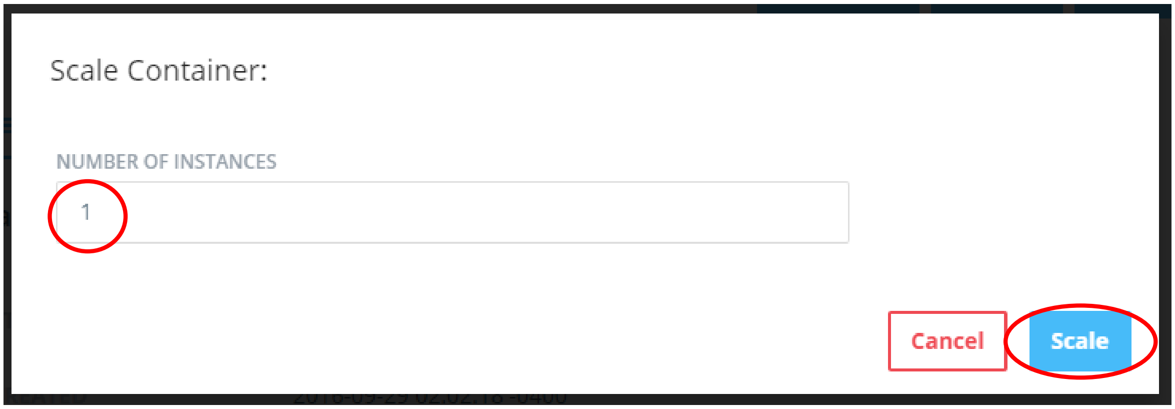
Docker Universal Control Plane
use
Dashboard / Applications / Container

user230votingapp_voting-app_1

dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/vote:latest

Restart
Stop
Scale
Rename
Remove
DETAILS
LOGS
STATS
CONSOLE

- Select the **Scale** button (as shown above), then input "**1**" into the "**Number of Instances**" field to increase the number of running containers to 2, as shown below (next page):



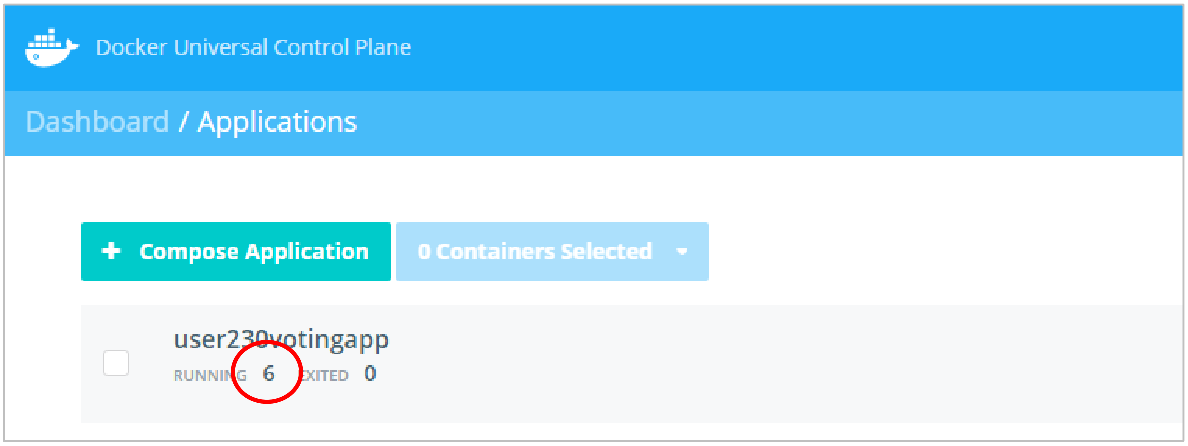
Scale Container:

NUMBER OF INSTANCES

1

- Select the **Scale** button (above) once you have entered the number to initiate the increase. Once it finishes loading, select '**Applications'** from the left side navigation menu, then expand the list of containers from your running application again.

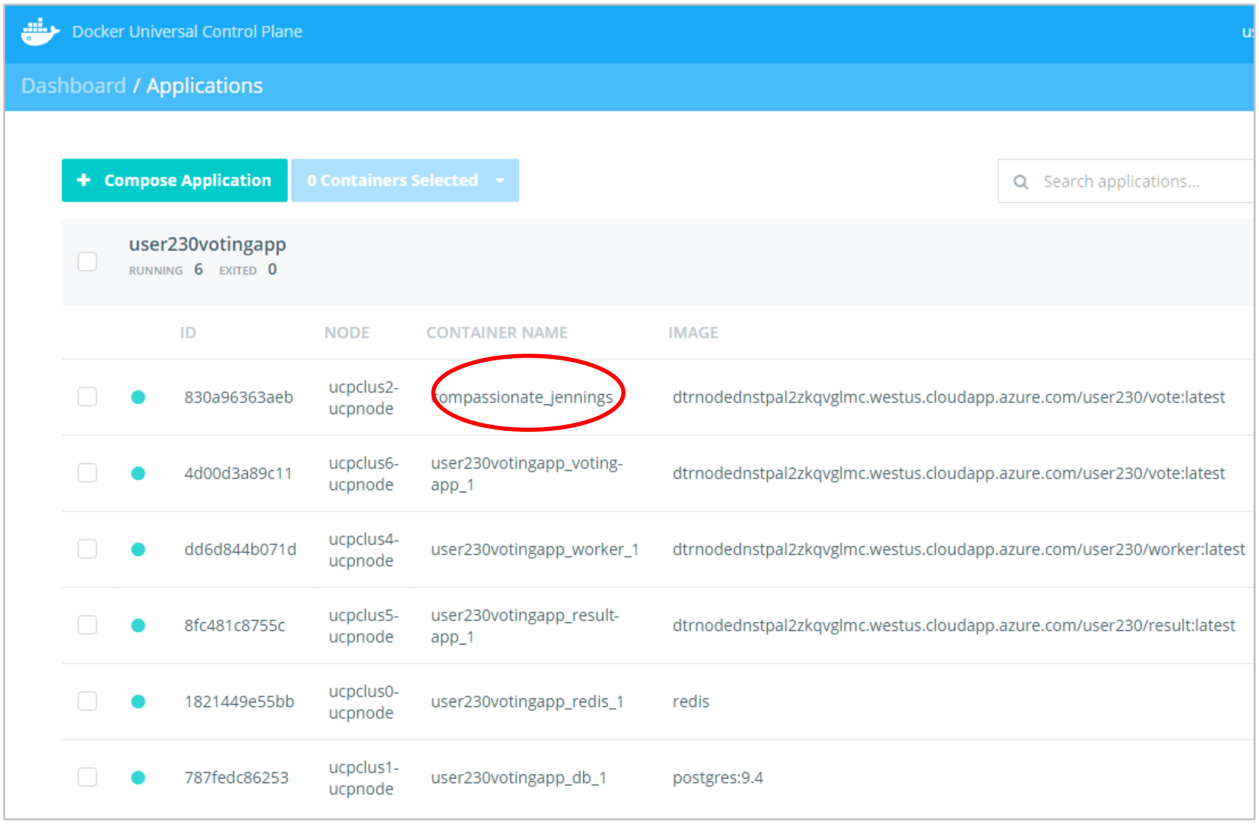
  You should see that the total containers running on your application have increased by 1, and you should also see the new container, with a new name, at the top of the expanded list, as shown below:



□ user230votingapp

RUNNING 6 EXITED 0

(Continued on next page)



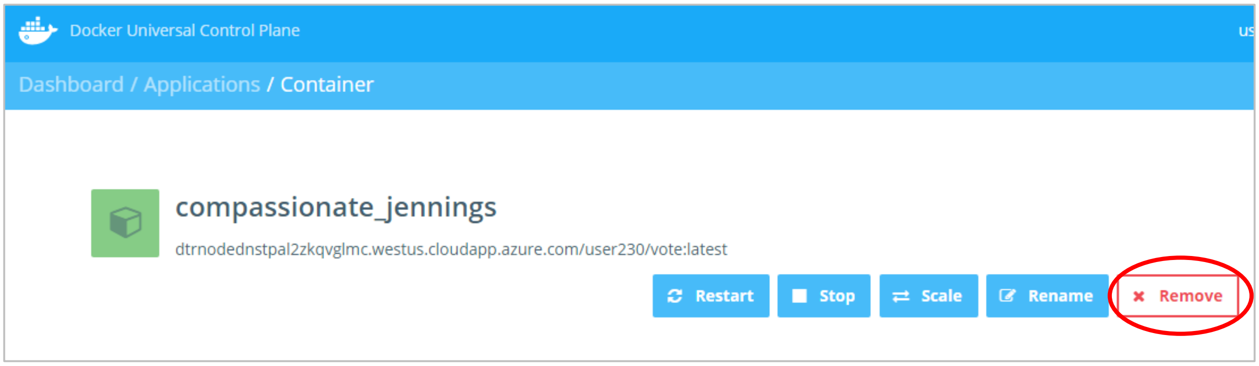
user230votingapp

RUNNING 6 EXITED 0

		ID	NODE	CONTAINER NAME	IMAGE
□	●	830a96363aeb	ucpclus2-ucpnode	compassionate_jennings	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/vote:latest
□	●	4d00d3a89c11	ucpclus6-ucpnode	user230votingapp_voting-app_1	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/vote:latest
□	●	dd6d844b071d	ucpclus4-ucpnode	user230votingapp_worker_1	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/worker:latest
□	●	8fc481c8755c	ucpclus5-ucpnode	user230votingapp_result-app_1	dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/result:latest
□	●	1821449e55bb	ucpclus0-ucpnode	user230votingapp_redis_1	redis
□	●	787fedc86253	ucpclus1-ucpnode	user230votingapp_db_1	postgres:9.4

- **Congratulations**! You've now scaled up your containers running in the application you created in UCP.
- You can also remove the containers you have scaled up to include.

  Select the new container you just created from the expanded list, then select '**Remove'** and select '**Yes'** on the next screen to delete the container, as shown below:



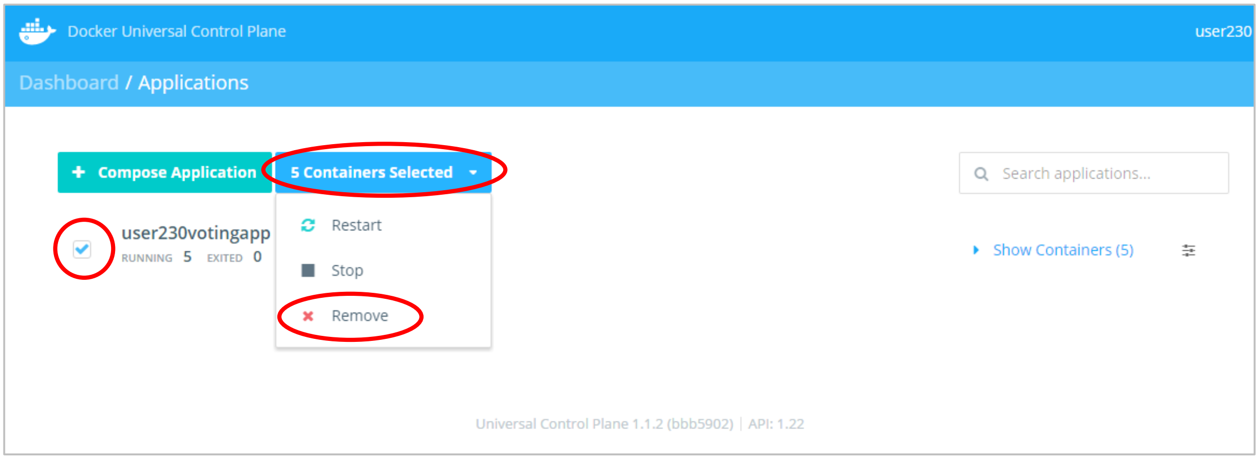
Docker Universal Control Plane
us
Dashboard / Applications / Container
Restart
Stop
Scale
Rename
Remove

compassionate_jennings

dtrnodednstpal2zkqvglmc.westus.cloudapp.azure.com/user230/vote:latest

- Now that you've scaled up and scaled down (removed) a container, let's clean up the application.
- Select '**Applications'** from the left side navigation menu, then **check the box** next to your application and select the **'# Containers Selected'** tab to drop down the menu options.

  From those options, select '**Remove'**, as shown below:



- Once done loading, you can refresh the web page to see that your application has been successfully removed.