

Lab 2: AWS Containers

Task

Deploy two containers (API and DB) which connect using ECR and EC2 ECS.

Time to finish: 15 min ☁

Walk-through

If you would like to attempt the task, then skip the walk-through and go for the task directly. However, if you need a little bit more hand holding or you would like to look up some of the commands or code or settings, then follow the walk-through.

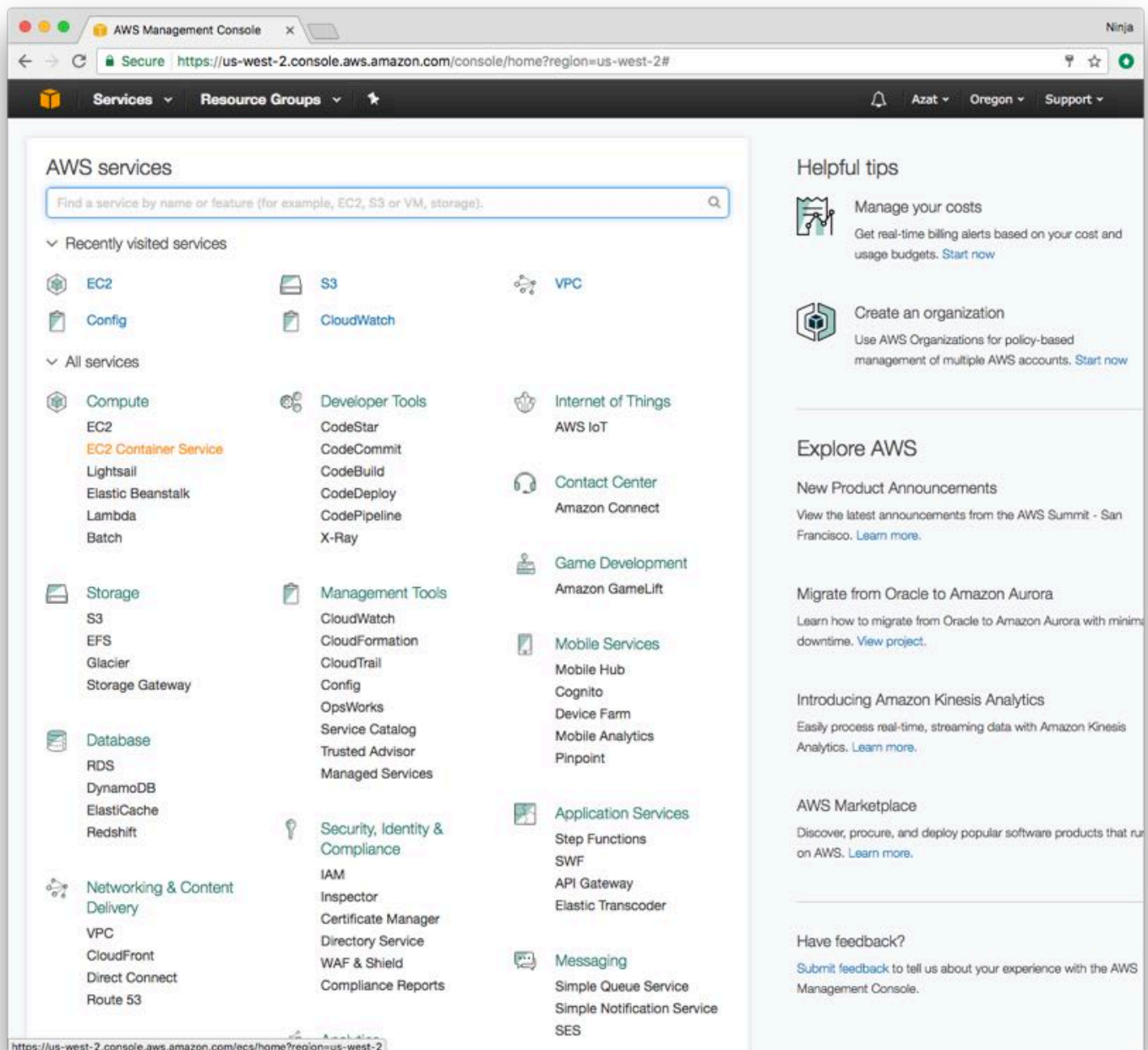
Steps to deploy a two-container project (app + database):

1. Create registry (ECR)
2. Upload the app image to ECR
3. Create task definition with 2 containers
4. Create a cluster
5. Create a service and run it

1. Create registry (ECR)

Each image needs to be uploaded to a registry before we can use it to run a container. There is registry from docker: hub.docker.com. AWS provides its own registry service called EC2 Elastic Container Registry (ECR). Let's use it.

Log in to your AWS web console at aws.amazon.com. Navigate to us-west-2 (or some other region, but we are using us-west-2 in this lab) and click on EC2 Container Service under Compute:



Then click on Repositories from a left menu and on a blue button *Create repository*. Then new repository wizard will look like this:

Amazon EC2 Container Service X Ninja

Secure | <https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/repositories/create/new>

Services Resource Groups

Get started with EC2 Container Registry

Step 1: Configure repository

Step 2: Build, tag, and push Docker image

Configure repository

This wizard will guide you through the steps of creating a repository in EC2 Container Registry. [Learn more](#)

Repository name*

Namespaces are optional, and they can be included in the repository name with a slash (for example, namespace/repo)

Repository URI 161599702702.dkr.ecr.us-west-2.amazonaws.com/

Permissions

As the owner, you have access to this repository by default. After completing this wizard, you can grant others permission to access this repository in the console.

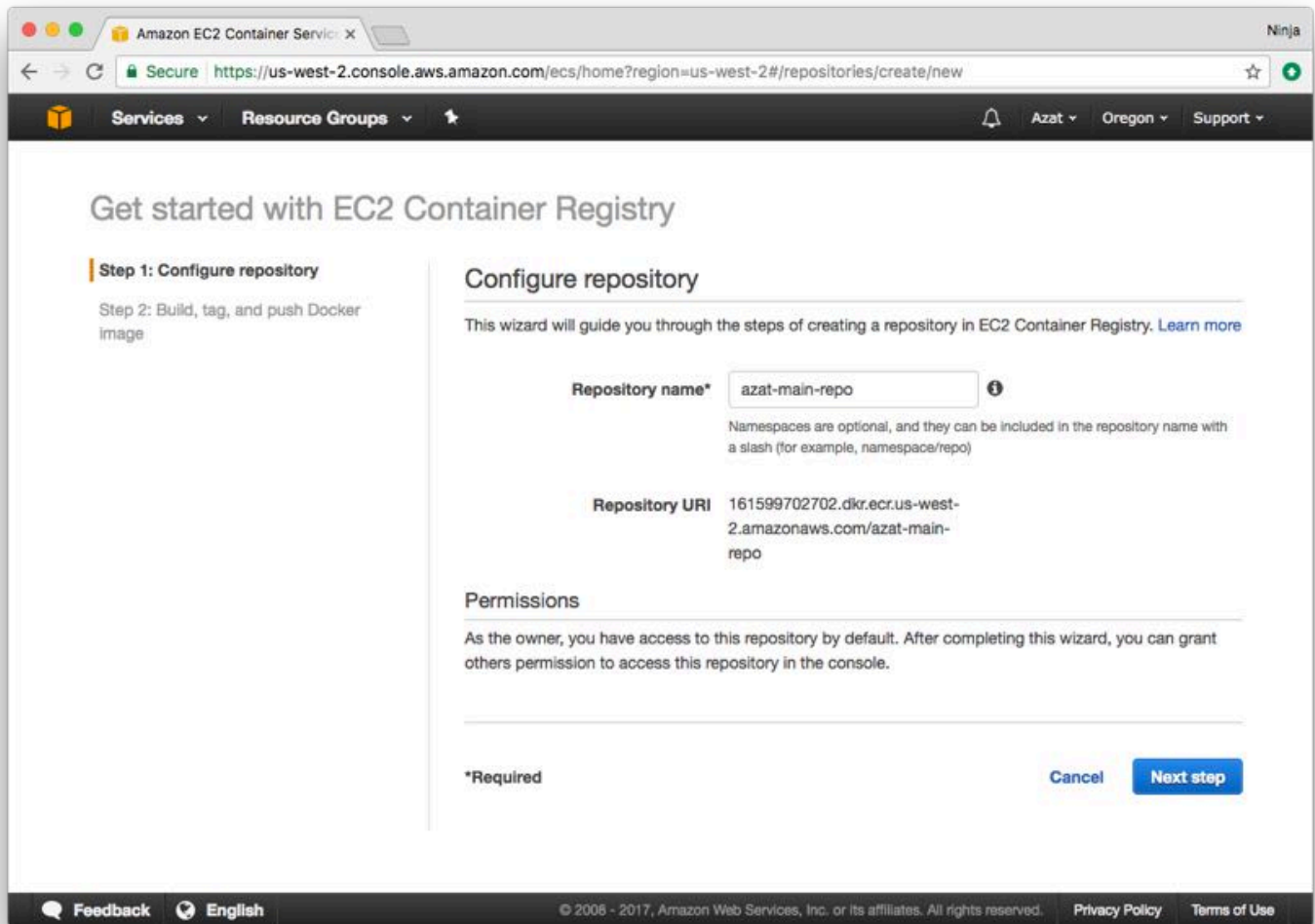
*Required

Cancel **Next step**

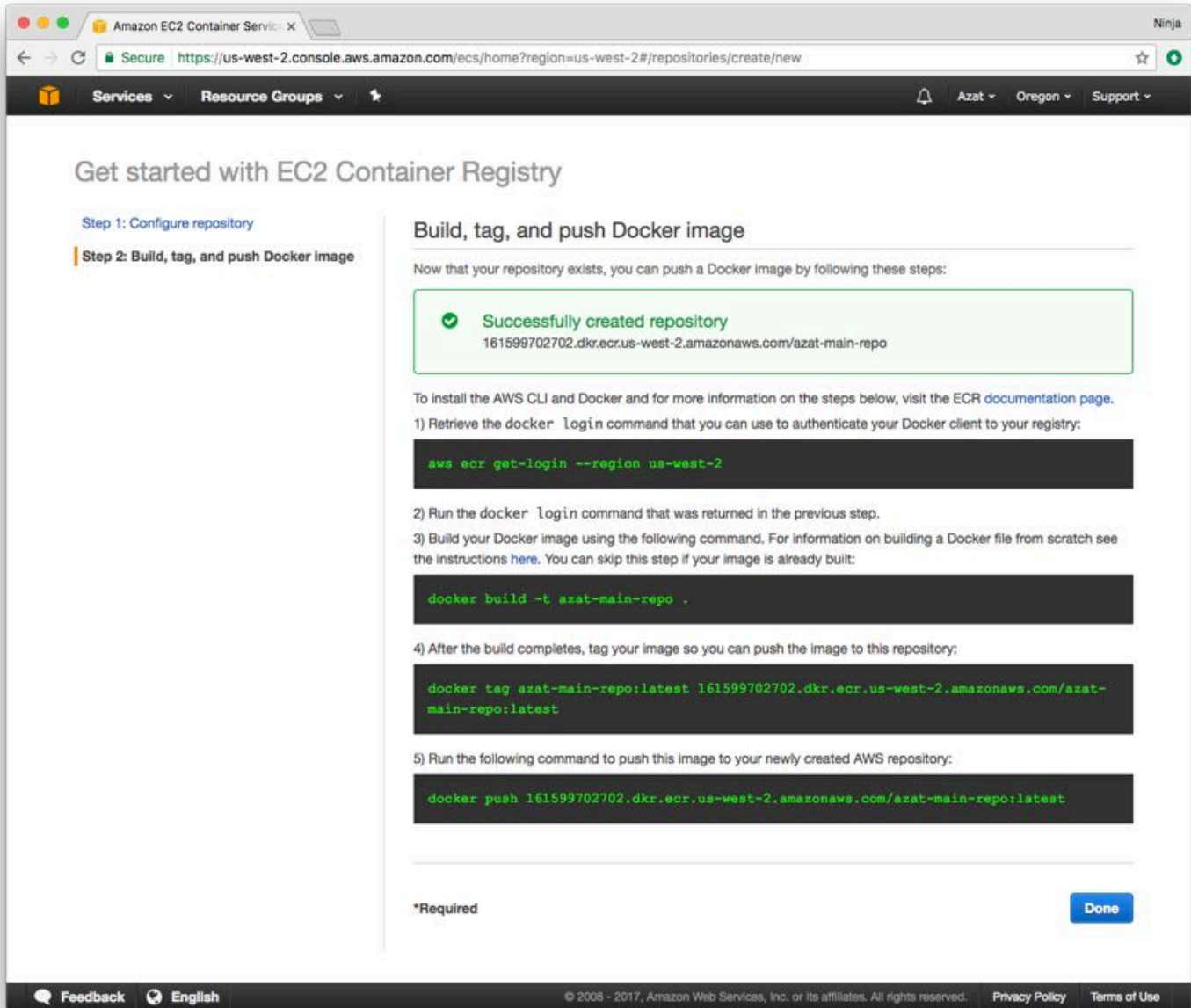
Feedback English

© 2006 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Enter the name of your repository for container images. I picked azat-main-repo because my name is Azat:



Click next and on Step 2, you will see bunch of commands.



Successfully created repository, e.g., my URL is

```
161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo
```

Next, follow instructions to upload an image (must build it before uploading/pushing).

To install the AWS CLI and Docker, and for more information on the steps below, [visit the ECR documentation page](#).

Command 1: Retrieve the docker login command that you can use to authenticate your Docker client to your registry:

```
aws ecr get-login --region us-west-2
```

Command 2: Run the docker login command that was returned in the previous step. For example,

```
docker login -u AWS -p eyJwYXlsb2FkIjoiQ1pUVnBTSHp
FNE5OSU1IdDhxeEZ3MlNrVTJGMUdBRlAxL1k4MDhRbE5lZ3JUW
...
W5VK01Ja0xQVnFSN3JpaHCJ0eXB1IjoiREFUQV9LRVkiQ==
-e none https://161599702702.dkr.ecr.us-west-2.amazonaws.com
```

Results:

```
Login Succeeded
```

Command 3:: Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:

```
cd code/banking-api
docker build -t azat-main-repo .
```

You might have done this already in the lab 1 (labs/1-dockerized-node.md). Skip to step 4. If not, then build the app image. The build command should end with a similar looking output:

```
...
Step 13/13 : CMD npm start
> Running in ee5f0fb12a2f
> 91e9122e9bed
Removing intermediate container ee5f0fb12a2f
Successfully built 91e9122e9bed
```

Command 4: After the build completes, tag your image so you can push the image to this repository:

```
docker tag azat-main-repo:latest 161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo:latest
```


(No output)

Command 5: Run the following command to push this image to your newly created AWS repository:


```
docker push 161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo:latest
```

Push output example:

```
The push refers to a repository [161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo]
9e5134c1ad7a: Pushed
e949bf24b1c4: Pushed
2b5c968a7072: Pushed
858e5e857851: Pushed
10e038bbd0ad: Pushed
ad2f0f4f7c5a: Pushed
ec6eb0ab894f: Pushed
e0380bb6c0bb: Pushed
9f8566ee5135: Pushed
latest: digest: sha256:6d1cd529ced84a6cff1eb5f6cffaed375717022b998e70b0d33c86db26a04c74 size: 2201
```

Remember digest (last hash)  Compare digest with one in the repository when you look up your image in the web console in EC2 -> ECS -> Repositories -> azat-main-repo:

The screenshot shows the Amazon ECS console interface. The left sidebar contains the navigation menu with 'Amazon ECS', 'Clusters', 'Task Definitions', and 'Repositories' (highlighted). The main content area is titled '< All repositories : azat-main-repo'. It displays the following details:

- Repository ARN:** arn:aws:ecr:us-west-2:161599702702:repository/azat-main-repo
- Repository URI:** 161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo
- View Push Commands:** A button to view push commands.

Below the details, there are two tabs: 'Images' (selected) and 'Permissions'. The 'Images' tab shows a message: 'Amazon ECR [limits](#) the number of images to 1,000 per repository. [Request a limit increase.](#) Image sizes may appear compressed. [Learn more](#)'. There is a 'Delete' button and a refresh icon. The last update time is 'May 5, 2017 3:34:30 PM (0m ago)'.

A filter bar shows 'Filter in this page', 'Tag Status: All', and 'Page size 100'. Below this is a table of image tags:

<input type="checkbox"/>	Image tags	Digest	Size (MiB)	Pushed at
<input type="checkbox"/>	latest	view all sha256:6d1cd529ced84a6cff1eb5f6cfae...	27.66	2017-05-05 15:34:09 -0700

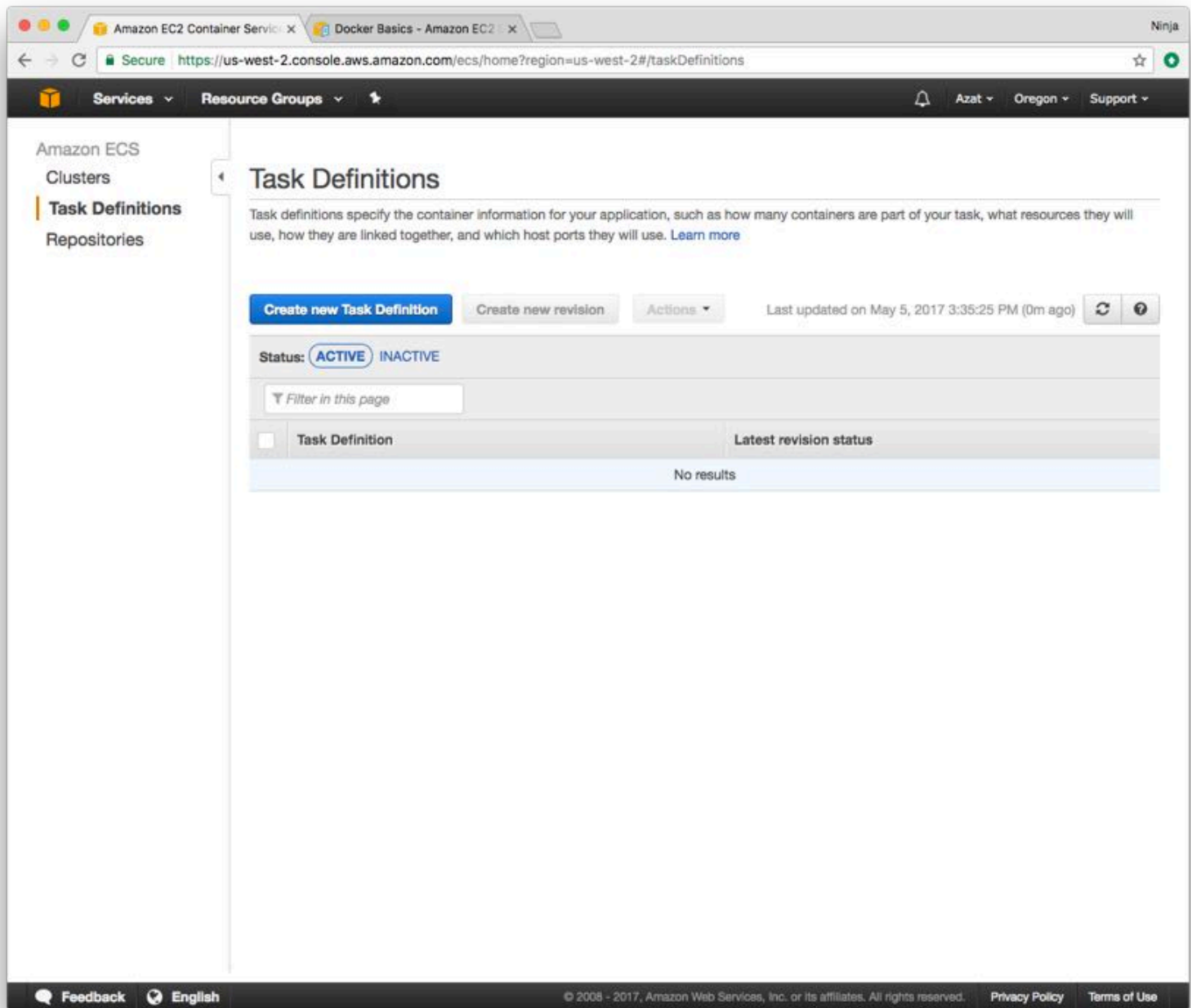
The footer contains 'Feedback', 'English', '© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.', 'Privacy Policy', and 'Terms of Use'.

2. Create new Task Definition

Tasks are like run commands in docker CLI (`docker run`) but for multiple containers. Typical tasks define:

- Container images to use
- Volumes if any
- Networks
- Environment variables
- Port mappings

Go to the Task Definitions in EC2 ECS and as you might guess, press on the button which says *Create new Task Definition*:



Main Task settings for the example

Use the following settings for the task to make sure your project is running (because some other values might make the project nonfunctional):

- Two containers: `banking-api` (private AWS ECR) and `mongodb` (docker hub)
- Connect to `mongodb` via network alias
- Map 80 (host) to 3000 (container) for `banking-api`
- Set env vars for `NODE_ENV` and `DB_URI`

Let's define the first container — app.

First container—App

Enter the name: banking-api-container.

Define the image URL (your URL will be different), e.g.,

```
161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo:latest
```

Define host 80 and container 3000 ports in port mappings. Name, image and ports are shown below:

Amazon EC2 Container Service

Secure <https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/taskDefinitions/create>

Services Resource Groups

Amazon ECS

Clusters

Task Definitions

Repositories

Create a Task Definition

A task definition specifies the containers to use.

Constraint

Constraints allow you to specify constraints for your task definition that match the constraints of the instances that match the task definition.

Type

+ Add constraint

Container Definition

+ Add container

Container Name

Volumes

Name

+ Add volume

Configure via JSON

Add container

Standard

Container name* banking-api-container

Image* 161599702702.dkr.ecr.us-west-2.amazonaws.com/azat-main-repo:latest

Custom image format: [registry-uri]/[namespace]/[image]:[tag]

Memory Limits (MB)*

Hard limit 128

+ Add Soft limit

Define hard and/or soft memory limits in MiB for your container. Hard and soft limits correspond to the 'memory' and 'memoryReservation' parameters, respectively, in task definitions. ECS recommends 300-500 MB as a starting point for web applications.

Port mappings

Host port	Container port	Protocol
80	3000	tcp

+ Add port mapping

Advanced container configuration

ENVIRONMENT

CPU units

Essential ☒

Entry point comma delimited: sh, -c

* Required

Cancel Add

Scroll down in the same modal view and add Env Variables:

```
DB_URI=mongodb://mongod-banking-api-prod-container:27017/db-prod
NODE_ENV=production
```

Add to Links the name of the MongoDB container (not defined yet) to give access to the database container to the app container such as one is the name of the container in the task definition and the other is the host name in the DB_URI:

```
mongod-banking-api-prod-container:mongod-banking-api-prod-container
```

See the screengrab below:

Amazon EC2 Container Service: x

Services Resource Groups

Create a Task Definition

A task definition specifies the containers to use.

Task Definitions

Repositories

Constraint

Constraints allow you to specify instances that match the task definition.

Type

Add constraint

Container Definition

Add container

Container Name

Volumes

Name

Add volume

Configure via JSON

Add container

Command

comma delimited: echo,hello world

Working directory

/usr/app

Env Variables

Key	Value
DB_URI	mongodb://mongox
NODE_ENV	production
Add key	Add value

NETWORK SETTINGS

Disable networking

Links

mongod-banking-api-prod-container:mongod-banking-api-prod-container

Hostname

<hostname>.<domainname>

DNS servers

line separated: 8.8.8.8

DNS search domains

line separated: search.example.com

Extra hosts

* Required

Cancel Add

Second container—Database

Analogous to the previous container, define name and URL with these values:

- Name: mongod-banking-api-prod-container
- Image URL: registry.hub.docker.com/library/mongo:latest

The screenshot shows the Amazon ECS console interface. On the left, the navigation pane includes 'Services', 'Resource Groups', 'Amazon ECS', 'Clusters', 'Task Definitions', and 'Repositories'. The main area displays the 'Create a Task Definition' page. A modal dialog titled 'Add container' is open, showing the configuration for a new container.

Add container

Standard

Container name* mongod-banking-api-prod-container

Image* registry.hub.docker.com/library/mongo:latest

Custom image format: {registry-uri}/{namespace}/{image}:{tag}

Memory Limits (MB)* Soft limit 128

+ Add Hard limit

Define hard and/or soft memory limits in MiB for your container. Hard and soft limits correspond to the 'memory' and 'memoryReservation' parameters, respectively, in task definitions. ECS recommends 300-500 MB as a starting point for web applications.

Port mappings

Host port	Container port	Protocol
		tcp

+ Add port mapping

Advanced container configuration

ENVIRONMENT

CPU units

Essential ☒

Entry point comma delimited: sh,-c

* Required

Cancel Add

Scroll down to the hostname in Network settings and enter Hostname as `mongod-banking-api-prod-container` as shown below:

The screenshot shows the 'Add container' dialog box in the Amazon ECS console. The dialog is titled 'Add container' and has a close button (X) in the top right corner. It contains two main sections: 'NETWORK SETTINGS' and 'STORAGE AND LOGGING'.

NETWORK SETTINGS

- Disable networking:** A checkbox that is currently unchecked.
- Links:** A text input field containing the value `comma delimited; othercontainer:alias, db:db`.
- Hostname:** A text input field containing the value `mongod-banking-api-prod-container`.
- DNS servers:** A text input field with a placeholder text 'line separated: 8.8.8.8'.
- DNS search domains:** A text input field with a placeholder text 'line separated: search.example.com'.
- Extra hosts:** A section with two input fields: 'Hostname' and 'IP address'. Below these fields is a button labeled 'Add extra host'.

STORAGE AND LOGGING

- Read only root file system:** A checkbox that is currently unchecked.

At the bottom of the dialog, there is a '* Required' label and two buttons: 'Cancel' and 'Add'.

After you added two container to the task, create the task and you'll see a screen similar to the one shown below:

Amazon EC2 Container Service

Services Resource Groups

Amazon ECS
Clusters
Task Definitions
Repositories

Created Task Definition successfully

Task Definitions > banking-api-task > 1

Task Definition: banking-api-task:1

View detailed information for your task definition. To modify the task definition, you need to create a new revision and then make the required changes to the task definition

Create new revision Actions

Builder JSON

Task Definition Name banking-api-task

Task Role None
Optional IAM role that tasks can use to make API requests to authorized AWS services. Create an Amazon EC2 Container Service Task Role in the IAM Console

Network Mode Bridge

Task Placement

Constraint No constraints

Container Definitions

Container Name	Image	CPU Units	Hard/Soft memory limits (MB)	Essential
banking-api-cont...	161599702702.d...	0	--/128	true
mongod-banking...	registry.hub.dock...	0	--/128	true

Volumes

Alternatively, you could specify volumes for database or/and the app at the stage of the task creation.

3. Create Cluster

Cluster is the place where AWS runs containers. They use configurations similar to EC2 instances. Define the following:

- Cluster name: `banking-api-cluster`
- EC2 instance type: `m4.large` (for more info on EC2 type, see [AWS Intro course on Node University](#))
- Number of instances: 1
- EBS storage: 22
- Key pair: None
- VPC: New

Amazon EC2 Container Service

Services ▾ Resource Groups ▾

Amazon ECS

- Clusters
- Task Definitions
- Repositories

Create Cluster

When you run tasks using Amazon ECS, you place them on a cluster, which is a logical grouping of EC2 instances. This wizard will guide you through the process to [create a cluster](#). You will name your cluster, and then configure the container instances that your tasks can be placed on, the security group for your container instances to use, and the IAM role to associate with your container instances so that they can make calls to the AWS APIs on your behalf.

Cluster name*

☐ Create an empty cluster

EC2 instance type*

Number of instances*

EC2 Ami Id*

EBS storage (GiB)*

Key pair

You will not be able to SSH into your EC2 instances without a key pair. You can create a new key pair in the [EC2 console](#).

Networking

Configure the VPC for your container instances to use. A VPC is an isolated portion of the AWS cloud populated by AWS objects, such as Amazon EC2 instances. You can choose an existing VPC, or create a new one with this wizard.

VPC

CIDR block

Launch the cluster. It might take a few minutes.

The screenshot shows the Amazon ECS console in the 'us-west-2' region. The left sidebar contains 'Amazon ECS' with sub-links for 'Clusters', 'Task Definitions', and 'Repositories'. The main content area is titled 'Launch status' and includes a message: 'Your container instances are launching, and it may take a few minutes until they are in the running state and ready to access. Usage hours on your new container instances start immediately and continue to accrue until you stop or terminate them.' Below this message are two buttons: 'Back' and 'View Cluster'. A progress indicator shows 'ECS status - 2 of 3 complete banking-api-cluster'. The status items are: 'ECS cluster' (successfully created), 'ECS Instance IAM Policy' (successfully attached), and 'CloudFormation Stack' (creating resources). At the bottom, a 'Cluster Resources' table lists configuration details.

Resource	Value
Instance type	m4.large
Desired number of instances	1
Key pair	
ECS AMI ID	ami-62d35c02
VPC Availability Zones	us-west-2c, us-west-2a, us-west-2b
Auto Scaling group	Pending...

You'll see the progress:

The screenshot shows the Amazon ECS console in the 'us-west-2' region. The left sidebar contains links to 'Clusters', 'Task Definitions', and 'Repositories'. The main content area is titled 'Launch status' and includes a message: 'Your container instances are launching, and it may take a few minutes until they are in the running state and ready to access. Usage hours on your new container instances start immediately and continue to accrue until you stop or terminate them.' Below this message are two buttons: 'Back' and 'View Cluster'. The 'ECS status' section shows '2 of 3 complete' for the 'banking-api-cluster'. It lists three items: 'ECS cluster' (successfully created), 'ECS Instance IAM Policy' (successfully attached), and 'CloudFormation Stack' (creating resources). The 'Cluster Resources' section lists various EC2 resources created by ECS, including instance type, VPC, subnets, route tables, and security groups.

Resource	Value
Instance type	m4.large
Desired number of instances	1
Key pair	
ECS AMI ID	ami-62d35c02
VPC	vpc-863e06e1
Subnet 1	subnet-4cda2517
Subnet 1 route table association	Pending...
Subnet 2	subnet-9c3a6efb
Subnet 2 route table association	Pending...
VPC Availability Zones	us-west-2c, us-west-2a, us-west-2b
Security group	sg-e845dc93
Internet gateway	igw-4d8242a
Route table	rtb-0c96236a
Amazon EC2 route	Pending...
Virtual private gateway attachment	EC2Co-Attac-WGM7OR3FDXV0
Auto Scaling group	Pending...

ECS creates a lot of EC2 resources for you such as Internet Gateway, VPC, security group, Auto Scaling group, etc. which is great because you don't have to create them manually.

Amazon ECS

Clusters

Task Definitions

Repositories

Launch status

Your container instances are launching, and it may take a few minutes until they are in the running state and ready to access. Usage hours on your new container instances start immediately and continue to accrue until you stop or terminate them.

Back

View Cluster

ECS status - 3 of 3 complete **banking-api-cluster**

✓

ECS cluster

ECS Cluster banking-api-cluster successfully created

✓

ECS Instance IAM Policy

IAM Policy for the role ecsInstanceRole successfully attached

✓

CloudFormation Stack

CloudFormation stack EC2ContainerService-banking-api-cluster and its resources successfully created

Cluster Resources

Instance type	m4.large
Desired number of instances	1
Key pair	
ECS AMI ID	ami-62d35c02
VPC	vpc-863e06e1
Subnet 1	subnet-4cda2517
Subnet 1 route table association	rtbassoc-e704ab9e
Subnet 2	subnet-9c3a6efb
Subnet 2 route table association	rtbassoc-a505aad6
VPC Availability Zones	us-west-2c, us-west-2a, us-west-2b
Security group	sg-e845dc93
Internet gateway	igw-4d8242a
Route table	rtb-0c96236a
Amazon EC2 route	EC2Co-Publi-1C5WGQYDQTFB
Virtual private gateway attachment	EC2Co-Attac-WGM7OR3FDXV0
Launch configuration	EC2ContainerService-banking-api-cluster-EcsInstanceLc-CNJTUMMLGSAQ
Auto Scaling group	EC2ContainerService-banking-api-cluster-EcsInstanceAsg-1H01BNG3D4ZPS

Feedback

English

© 2016 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Privacy Policy

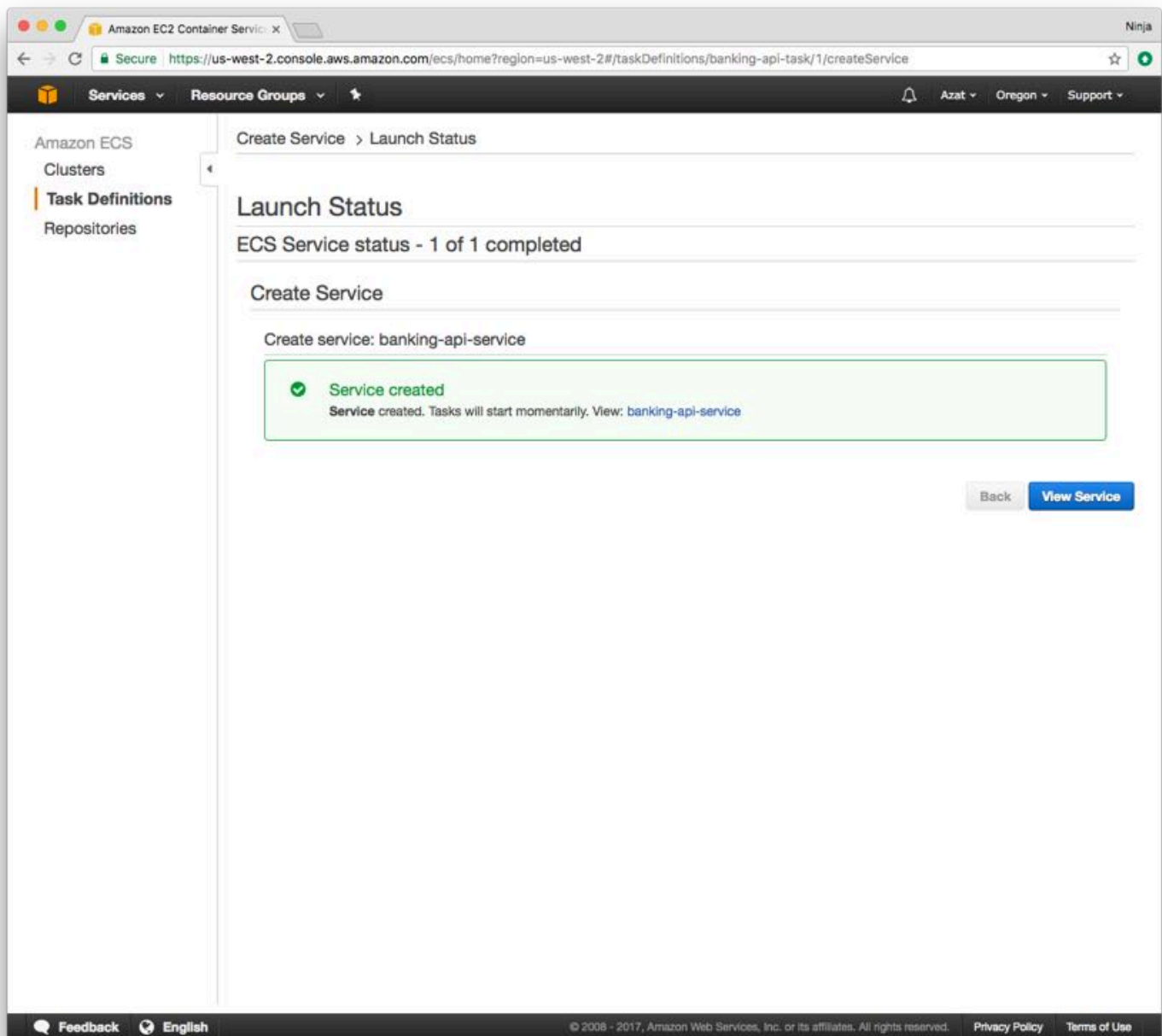
Terms of Use

24 of 28

4. Create Service and Verify

The last step is to create a service which will take the task and the cluster and make the containers in the task run in the specified cluster. It's oversimplified explanation because service will do more such as monitor health and restart containers.

Go to Create Services which is under Task Definition -> banking-api-task -> Actions -> Create Service. You will see this:



Everything is ready

Phew. Everything should be ready by now. To verify, we need to grab a public IP or public DNS. To do so, click Clusters -> banking-api-cluster (cluster name) -> ESC Instances (tab) and Container instance:

The screenshot shows the Amazon ECS console interface. The breadcrumb navigation is Clusters > banking-api-cluster > Container Instance: eaa733aa-6f1c-4d45-9a11-12b39b535fa0. The main heading is 'Container Instance : eaa733aa-6f1c-4d45-9a11-12b39b535fa0' with 'Update agent' and 'Deregister' buttons. The 'Details' section lists the following information:

- Cluster: banking-api-cluster
- EC2 Instance: i-04e92a15e2d0f3f99
- Availability Zone: us-west-2c
- Public DNS: ec2-54-187-207-5.us-west-2.compute.amazonaws.com
- Private DNS: ip-10-0-0-10.us-west-2.compute.internal
- Public IP: 54.187.207.5
- Private IP: 10.0.0.10
- Status: ACTIVE
- Running tasks count: 1
- Agent version: 1.14.1
- Docker version: 1.12.6

The 'Resources' section shows a table with the following data:

Resources	Registered	Available
CPU	2048	2048
Memory	7986	7730
Ports	6 ports	

The 'Tasks' section shows a 'Stop' button and 'Stop All' button. The task status is 'Running' (highlighted in blue) and 'Stopped'. Below this is a table with the following data:

Task	Task Definition	Group	Last status	Desired status
<input type="checkbox"/> e6321816-34ad-4409-ac...	banking-api-task:1	service:banking-api-serv...	RUNNING	RUNNING

Copy public IP or DNS .

Dynamic Test

To test the dynamic content (content generated by the app with the help of a database), open in browser with `{PUBLIC_DNS}/accounts`. Most likely the response will be `[]` because the database is empty but that's a good response. The server is working and can connect to the database from a different container.

Static Test

To test the static content such as an image which was downloaded from the Internet by Docker (ADD in Dockerfile) and baked into the image, navigate to http://{PUBLIC_DNS}/node-university-logo.png to see the images with Docker downloaded via ADD. Using ADD, you can fetch any data such as HTTPS certificates (from a private S3 for example).

Terminate Service and Cluster/Instances

Don't forget to terminate your service and instances. You can do it from the web console.