In this scenario you will learn how to deploy containers to Azure using the Azure Container Instances (ACI).

The ACI is a fast way to deploy containers with a per-second billing model, making it ideal for short, ad-hoc, workloads.

Login to azure : az login -u $username -p $password

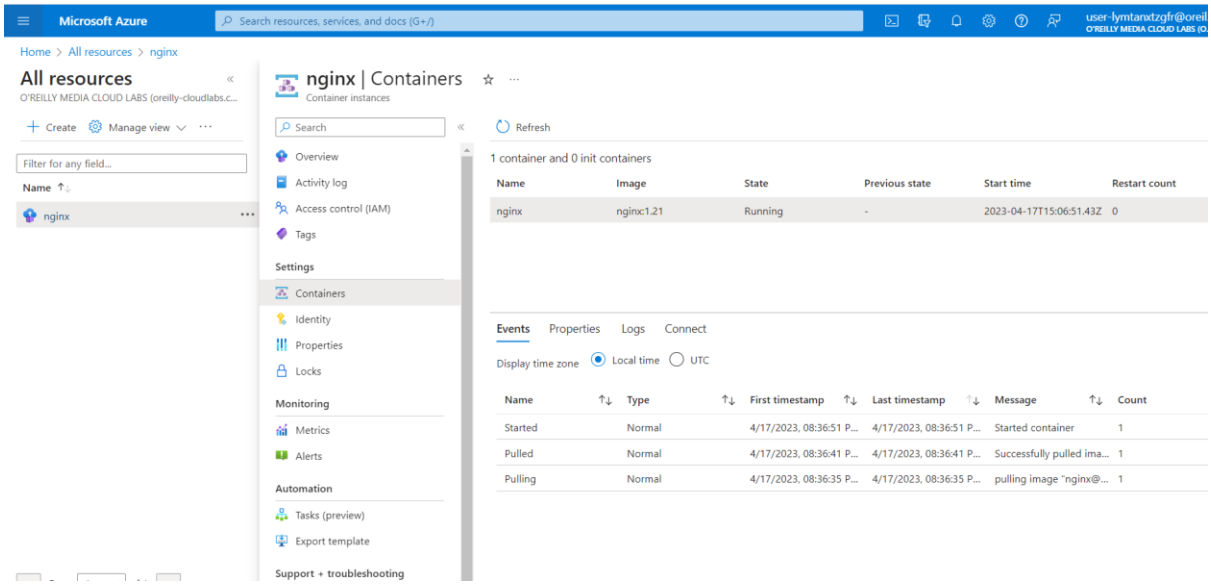
Step 2 - Deploy Container
The following command deploys an NGINX container to your allocated Azure Resource Group.

The deployment also specifies that a public IP is required. Once the container starts, the public IP address is automatically routed to the running container.

```
az container create \
 -g $resource \
 --name nginx \
 --image nginx:1.21 \
 --ip-address public
```

```
error: the --name/-n argument is required unless specified with
$ az container create -g $resource \
> --name nginx \
> --image nginx:1.21 \
> --ip-address public
{
  "containers": [
    {
      "command": null,
      "environmentVariables": [],
      "image": "nginx:1.21",
      "instanceView": {
        "currentState": {
          "detailStatus": "",
          "exitCode": null,
          "finishTime": null,
          "startTime": "2023-04-17T15:06:51.430000+00:00",
          "state": "Running"
        },
        "events": [
          {
            "count": 1,
            "firstTimestamp": "2023-04-17T15:06:35+00:00",
            "lastTimestamp": "2023-04-17T15:06:35+00:00",
            "message": "pulling image \"nginx@sha256:25dedae0ac
            "name": "Pulling",
            "type": "Normal"
          },
```

Container is created in azure .



Step 3 - View Details
Using the CLI, it's possible to view all of the running container instances:

az container list

```
$ az container list
[
  {
    "containers": [
      {
        "command": null,
        "environmentVariables": [],
        "image": "nginx:1.21",
        "instanceView": null,
        "livenessProbe": null,
        "name": "nginx",
        "ports": [
          {
            "port": 80,
            "protocol": "TCP"
          }
        ],
        "readinessProbe": null,
        "resources": {
          "limits": null,
          "requests": {
            "cpu": 1.0,
            "gpu": null,
            "memoryInGb": 1.5
          }
```

You might prefer the output in a table:
az container list -o table

```
$ az container list -o table
Name    ResourceGroup      Status     Image       IP:ports           Network    CPU/Memory       OsType    Location
------  -----------------  ---------  ----------  -----------------  ---------  ---------------  --------  ----------
nginx   user-lymtanxtzgfr  Succeeded  nginx:1.21  52.151.221.210:80  Public     1.0 core/1.5 gb  Linux     eastus
```

The CLI gives you access to various runtime aspects of the container, such as the logs, allowing you to understand the execution of the workload:

az container logs \
 -g $resource \
 --name nginx

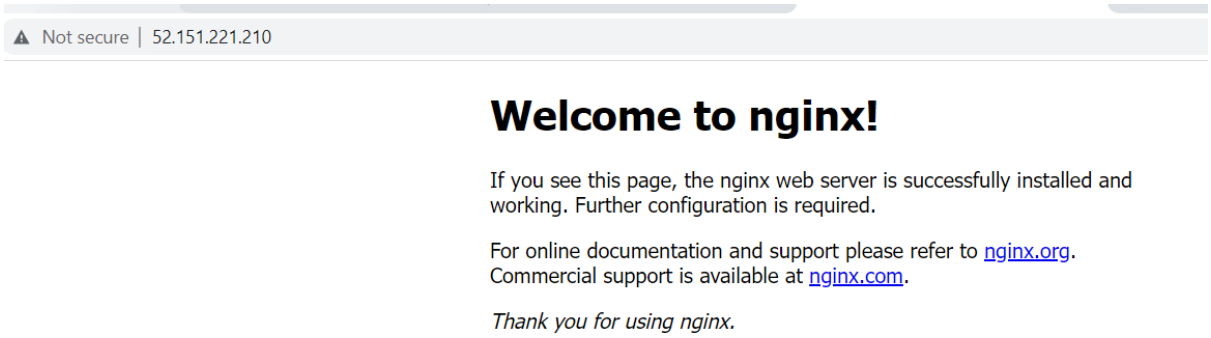
```
nginx   user-lymtanxtzgfr  Succeeded  nginx:1.21  52.151.221.210:80  Public     1.0 core/1.5 gb  Li
$ az container logs \
> -g $resource \
> --name nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/04/17 15:06:51 [notice] 19#19: using the "epoll" event method
2023/04/17 15:06:51 [notice] 19#19: nginx/1.21.6
2023/04/17 15:06:51 [notice] 19#19: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/04/17 15:06:51 [notice] 19#19: OS: Linux 5.10.102.2-microsoft-standard
2023/04/17 15:06:51 [notice] 19#19: getrlimit(RLIMIT_NOFILE): 1024:1048576
2023/04/17 15:06:51 [notice] 19#19: start worker processes
2023/04/17 15:06:51 [notice] 19#19: start worker process 48
2023/04/17 15:09:00 [error] 48#48: *3 open() "/usr/share/nginx/html/aaa9" failed (2: No such file o
 HTTP/1.1", host: "52.151.221.210"
10.92.0.11 - - [17/Apr/2023:15:09:00 +0000] "GET /aaa9 HTTP/1.1" 404 153 "-" "Mozilla/5.0 zgrab/0.x
10.92.0.10 - - [17/Apr/2023:15:09:08 +0000] "GET /aab8 HTTP/1.1" 404 153 "-" "Mozilla/5.0 zgrab/0.x
2023/04/17 15:09:08 [error] 48#48: *6 open() "/usr/share/nginx/html/aab8" failed (2: No such file o
 HTTP/1.1", host: "52.151.221.210"
10.92.0.10 - - [17/Apr/2023:15:09:22 +0000] "GET / HTTP/1.1" 200 615 "-" "Mozilla/5.0 zgrab/0.x" "-
```

Use curl to access the IP address and see the results of the container.

```
$ curl 52.151.221.210:80
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is succe
working. Further configuration is required.</p>

<p>For online documentation and support please refer t
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
$
```



Step 5 - Delete Container
Once you've finished with the container, you can delete it using the CLI:

```
az container delete \
  --resource-group $resource \
  --name nginx
```

```
</html>
$ az container delete \
> -g $resource \
> --name nginx
Are you sure you want to perform this operation? (y/n): y
{
  "containers": [
    {
      "command": null,
      "environmentVariables": [],
      "image": "nginx:1.21",
      "instanceView": {
        "currentState": {
          "detailStatus": "",
          "exitCode": null,
          "finishTime": null,
          "startTime": "2023-04-17T15:06:51.430000+00:00",
          "state": "Running"
        },
        "events": [
```

Viewing the container list will show that the container has been stopped and removed:
az container list -o table

```
}
$ az container list -o table

$
```