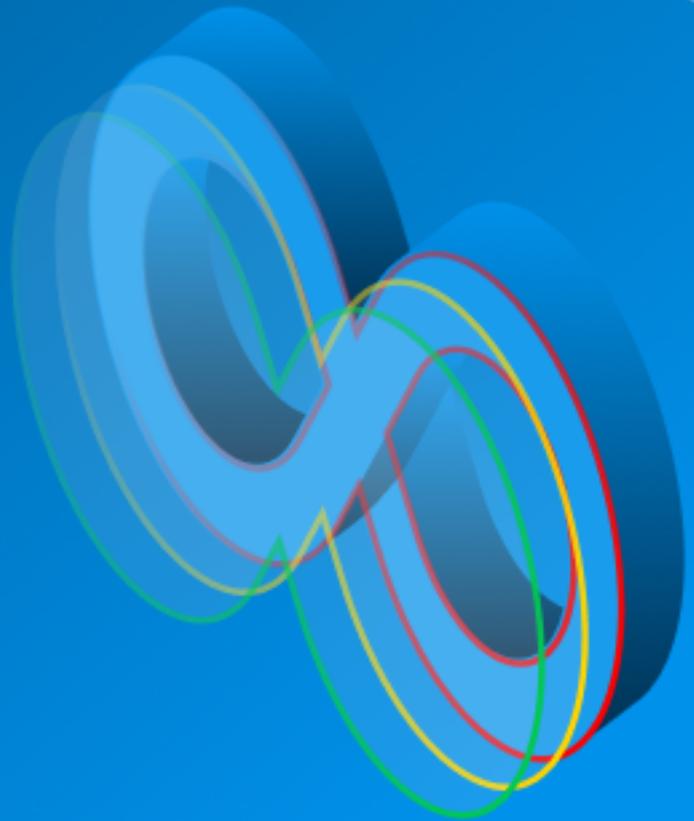
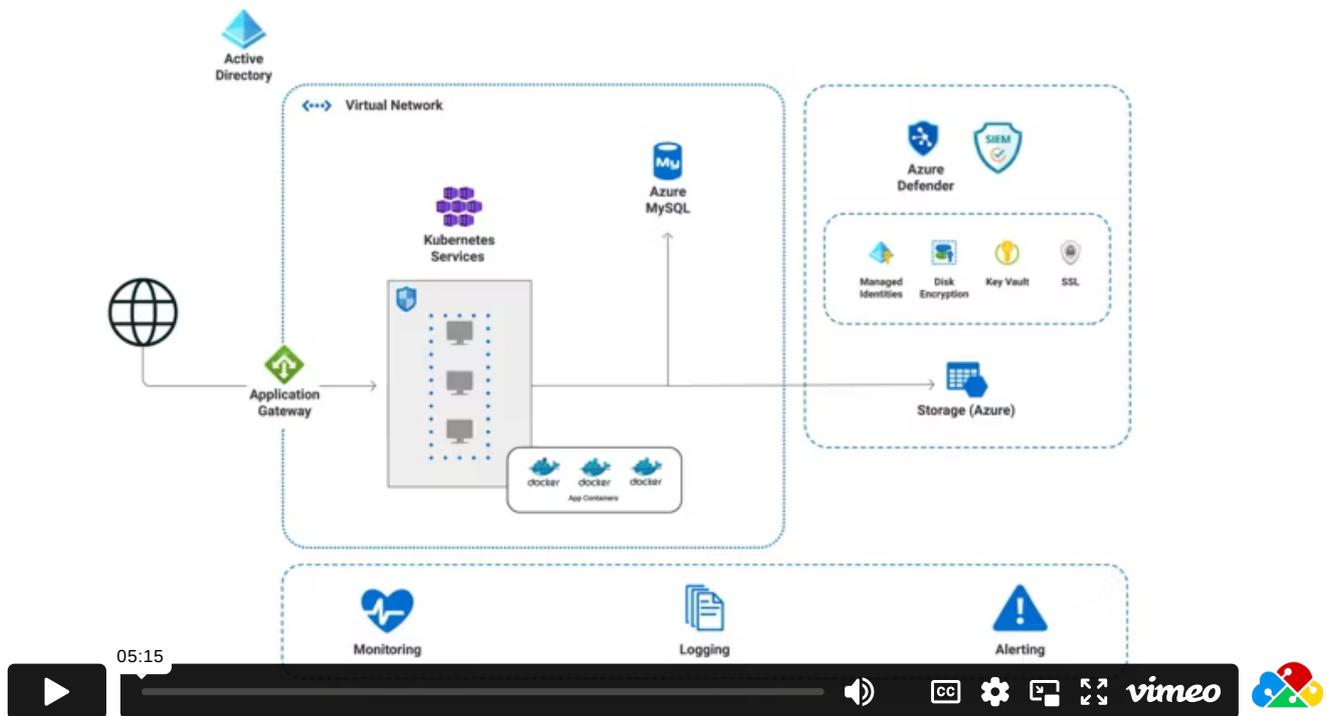


Microsoft Azure/ DuploCloud Quick Start



Quick Start

Get up and running with DuploCloud running inside a Microsoft Azure cloud environment; harness the power of generating application infrastructures.



This Quick Start tutorial shows you how to set up an end-to-end cloud deployment. You will create Azure infrastructure and tenants and by the end of this tutorial, you can view a deployed sample web application.

Estimated time to complete tutorial: 90-100 minutes.

1. [Create Azure Infrastructure](#)
2. [Create a Tenant](#)
3. [Create an Azure Agent Pool](#)
4. [Create and deploy a sample app service](#)
5. [Create a Load Balancer to access the web application deployed](#)
6. [Test the sample application](#)

Step 1: Create Infrastructure and Plan

Creating the DuploCloud Infrastructure and a Plan

Each DuploCloud Infrastructure is a Virtual Private Cloud (VPC) instance that resides in a region containing Kubernetes clusters, AKS clusters, GKE clusters, EKS clusters, or a combination of all of these. An Infrastructure can reside On-Premises (On-Prem) or in a Public Cloud.

After you supply a few basic inputs DuploCloud creates an Infrastructure for you, within Azure and within DuploCloud, with a few clicks. Behind the scenes, DuploCloud does a lot with what little you supply — generating subnets, NAT gateway, routes, and a cluster in the region.

With the Infrastructure as your foundation, you can customize an extensible, versatile Platform Engineering development environment by adding Tenants, Plans, Hosts, Services, and more.

Estimated time to complete Step 1: 40 minutes. Much of this time is consumed by DuploCloud's creation of the Infrastructure and enabling your AKS cluster with Kubernetes.

Prerequisites

Before starting this tutorial:

- Learn more about DuploCloud [Infrastructures](#), [Plans](#), and [Tenants](#).
- Reference the [Access Control](#) documentation to create User IDs with the **Administrator** role. In order to perform the tasks in this tutorial, you must have Administrator privileges.

Creating a DuploCloud Infrastructure

1. In the DuploCloud Portal, navigate to **Administrator -> Infrastructure**.
2. Click **Add**. The **Add Infrastructure** page displays.

The screenshot shows the 'Add Infrastructure' form with the following values:

- Name: nonprod
- Subscription: Azure: 143ffc59-9394-4ec6-8f5a-c408a238be62
- VNET CIDR: 10.23.0.0/16
- Cloud: Azure
- Region: West US
- Subnet Name: default
- Security Group Rules: [{"Name": "ssh", "Priority": "101", "SrcAddressPrefix": "", "DstAddressPrefix": "", "SourcePortRange": "", "DestinationPortRange": "22", "Protocol": "tcp", "Direction": "Inbound"}]
- Subnet CIDR: 10.23.0.0/20
- Type: None
- Log Analytics Workspace: No

Add Infrastructure page for creating a DuploCloud Infrastructure

3. From the table below, enter the values that correspond to the fields on the **Add Infrastructure** page. Accept all other default values for fields not specified.
4. Click **Create** to create the Infrastructure. It may take up to half an hour to create the Infrastructure. While the Infrastructure is being created, a **Pending** status displays in the Infrastructure page **Status** column, often with additional information about what part of the Infrastructure DuploCloud is currently creating. When creation completes, a status of **Complete** displays.

Add Infrastructure page field	Value
Name	nonprod
Subscription	YOUR_AZURE_SUBSCRIPTION_NAME
VNET CIDR	10.23.0.0/16
Subnet CIDR	10.23.0.0/20
Cloud	Azure
Region	YOUR_GEOGRAPHIC_REGION

The screenshot shows the 'Infrastructure' page with a table listing the created infrastructure:

NAME	CLOUD	REGION	VNET	STATUS
NONPROD	Azure	West US	10.23.0.0/16	Complete

Infrastructure creation with a status of **Complete**

Verifying that a Plan exists for your infrastructure

Every DuploCloud Infrastructure generates a Plan. Plans are sets of templates that are used to configure the Tenants, or workspaces, in your Infrastructure. You will set up Tenants in the next tutorial step.

Before proceeding, confirm that a Plan exists that corresponds to your newly created Infrastructure.

1. In the DuploCloud Portal, navigate to **Administrator** -> **Plans**. The **Plans** page displays.
2. Verify that a Plan exists with the name **NONPROD**, the name that you gave to the Infrastructure you created.

Enabling the AKS Kubernetes Cluster

Once your Infrastructure and Plan have been created, the final step before creating a Tenant is to enable Azure Kubernetes Service (AKS) to connect with Azure cloud management.

1. In the DuploCloud Portal, navigate to **Administrator** -> **Infrastructure**.
2. Select the **NONPROD** Infrastructure that you created, in the **NAME** column of the Infrastructure page.
3. Click the **Kubernetes** tab. The following message displays: **Kubernetes cluster is not yet enabled. Click Here to enable the Kubernetes Cluster.**
4. Click on the **Click Here** hyperlink. The **Configure AKS Cluster** pane displays.
5. Accept the default values and click **Create** to enable the AKS service for your Infrastructure.

DuploCloud begins creating and configuring an AKS cluster using Kubernetes. You receive an alert message when the Infrastructure has been updated.

- ✔ It may take some time for enablement to complete. Use the **Kubernetes** card in the Infrastructure screen to monitor the status, which should display as **Enabled** when completed. You can also monitor progress by using the **Kubernetes** tab, as DuploCloud generates your **Cluster Name**, **Default VM Size**, **Server Endpoint**, and **Token**.

Check your work

You previously verified that your Infrastructure and Plan were created. Now verify that AKS is Enabled before proceeding to [Create a Tenant](#).

When AKS has been **Enabled**, details are listed in the **Kubernetes** tab. The Infrastructure page also displays the **Enabled** status on the **Kubernetes** card.

KEY	VALUE
Cluster Name	nonprod
Default VM Size	Standard_B2s
Server Endpoint	https://nonprod-efc3131c.hcp.uswest.azmk8s.io
Token	56d884a39180ee0b2797c9a22a60a4cccdc5f5c0df6771ddb49

Kubernetes tab in the **Infrastructure** page with details about your configured AKS cluster

Infrastructure [Admin](#) > [Infrastructure](#) > nonprod

N
NONPROD
Actions

Status ⓘ
Complete ⓘ

Complete
Cloud: Azure
Region: East US
VNET: 10.23.0.0/16

[Subnets](#) [Security Group Rules](#) [Settings](#) [Kubernetes](#) [Log Analytics WS](#) [Vault Config](#) [Metadata](#)

Total 1 Show 25 + Add

NAME	ID	ADDRESSPREFIX	SUBNETTYPE	ACTIONS
duploinfra-default		10.23.0.0/20		⋮

1 total

Subnets
1 ⓘ

Kubernetes
Enabled ⓘ

NONPROD Infrastructure page with Kubernetes Enabled card

Step 2: Create a Tenant

Creating a DuploCloud Tenant that segregates your workloads

Now that the [Infrastructure and Plan](#) exist and [AKS has been enabled](#), create one or more Tenants that use the configuration DuploCloud created.

[Tenants](#) in DuploCloud are similar to projects or workspaces and have a subordinate relationship to the Infrastructure. Think of the Infrastructure as a virtual "house" (cloud), with Tenants conceptually "residing" in the Infrastructure performing specific workloads that you define. As Infrastructure is an abstraction of a Virtual Private Cloud, Tenants abstract the segregation created by a [Kubernetes Namespace](#), although Kubernetes Namespaces are only one component that Tenants can contain.

In Azure, Microsoft cloud features such as Azure resource groups, Azure managed identity, Azure application security groups (ASG), and KMS keys are exposed in Tenants, which reference these feature configurations.

Estimated time to complete Step 2: 10 minutes.

Tenant use cases

DuploCloud customers often create at least two Tenants for their production and non-production cloud environments (Infrastructures).

For example:

- Production Infrastructure
 - Pre-production Tenant - for preparing or reviewing production code
 - Production Tenant - for deploying tested code
- Non-production Infrastructure
 - Development Tenant - for writing and reviewing code
 - Quality Assurance Tenant - for automated testing

In larger organizations, some customers create Tenants based on application environments, such as creating one Tenant for Data Science applications and another Tenant for web applications, and so on.

Tenants are sometimes created to isolate a single customer workload, allowing more granular performance monitoring, scaling flexibility, or tighter security. This is referred to as a *single-Tenant* setup.

Prerequisites

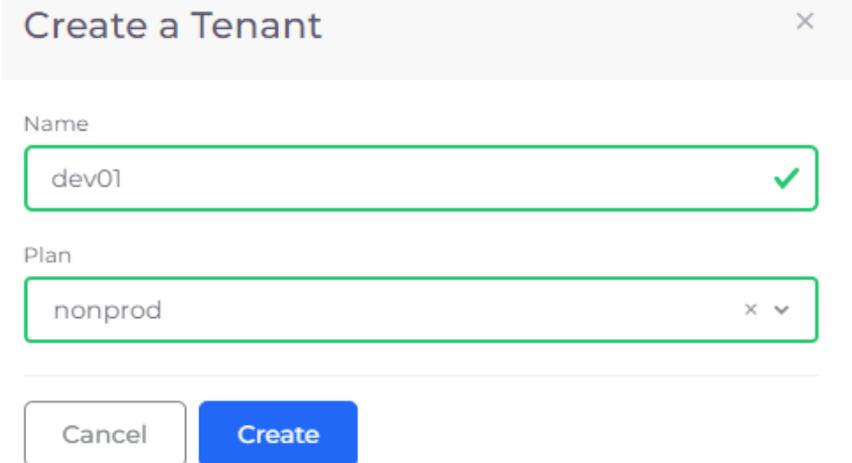
Before creating a Tenant, verify that you accomplished the tasks in [Step 1](#) of this tutorial. Using the DuploCloud Portal, confirm that:

- An [Infrastructure and Plan exist](#), both with the name **NONPROD**.
- The **NONPROD** infrastructure has [Azure Kubernetes Service \(AKS\) Enabled](#).

Creating a Tenant

Create a Tenant for your Infrastructure and Plan:

1. In the DuploCloud Portal, navigate to **Administrator --> Tenants**.
2. Click **Add**. The **Create a Tenant** pane displays.
3. Enter **dev01** in the **Name** field.
4. Select the **Plan** that you created in the previous step (**NONPROD**).
5. Click **Create**.



Create a Tenant ×

Name

dev01 ✓

Plan

nonprod × ▾

Cancel Create

Create a Tenant pane

Check your work

Navigate to **Administrator -> Tenants** and verify that the **DEV01** Tenant displays in the list.

Tenant: DEV01 Switch to Old UI Bob Administrator

Tenants | Admin > Tenants

Total 6 Show 25 Search + Add

NAME	ID	PLAN	ACTIONS
PVC01	218535d1-48d6-42f3-9ac2-2a249fc6cd79	pvctest	🔗 ⋮
DUPLO	50d1897a-33ad-46f3-853a-2398c5940aed	pvctest	🔗 ⋮
TK8V-01	e7f39845-c69d-452f-9887-93af03c7c92c	k8version	🔗 ⋮
QAAUTO-07	cafa97ba-246f-4b60-8686-67c9ac708ffd	sanqa1001	🔗 ⋮
BG-TST-AZURE	2f104765-8584-4882-817d-b8e50b6370fb	sanqa1001	🔗 ⋮
DEV01	c6bd1313-68db-4251-853d-31b01eb54e8a	nonprod	🔗 ⋮

6 total

Tenant page with Tenant dev01 using Plan NONPROD

Navigate to **Administrator -> Infrastructure** and select **DEV01** from the **Tenant** list box at the top left in the DuploCloud Portal. The **NONPROD** Infrastructure appears in the list of Infrastructures, indicating that the **DEV01** Tenant is associated with Infrastructure **NONPROD**.

Tenant: DEV01 Switch to Old UI Bob Administrator

Infrastructure | Admin > Infrastructure

Total 4 Show 25 Search + Add

NAME	CLOUD	REGION	VNET	STATUS	ACTIONS
PVCTEST	Azure	South India	10.111.0.0/16	Complete	⋮
SANQA1001	Azure	West US 3	10.120.0.0/16	Complete	⋮
K8VERSION	Azure	West US 2	10.121.0.0/16	Complete	⋮
NONPROD	Azure	East US	10.23.0.0/16	Complete	⋮

4 total

Tenant list box with DEV01 selected; NONPROD Infrastructure with Status Complete

Step 3: Create Agent Pools

Creating Azure Agent Pools as shared resources across Tenants

So far you have created an [Infrastructure](#), a [Plan](#), and a [Tenant](#). Now you need to create Agent Pools to serve computing and storage resources to your Tenants, using agents that monitor resource allocation.

Instead of managing each agent individually, agents are grouped into [agent pools](#) for maximum efficiency. You share Azure Agent Pools across workloads defined by the Tenants that you set up. Azure Agent Pools are scoped to a Host (Virtual Machine or VM) or a group of Hosts by [Azure Pipeline Agents](#). In this tutorial, you won't be creating specific Hosts, but you will create an Azure Agent Pool to which a group of VMs has already been defined by DuploCloud.

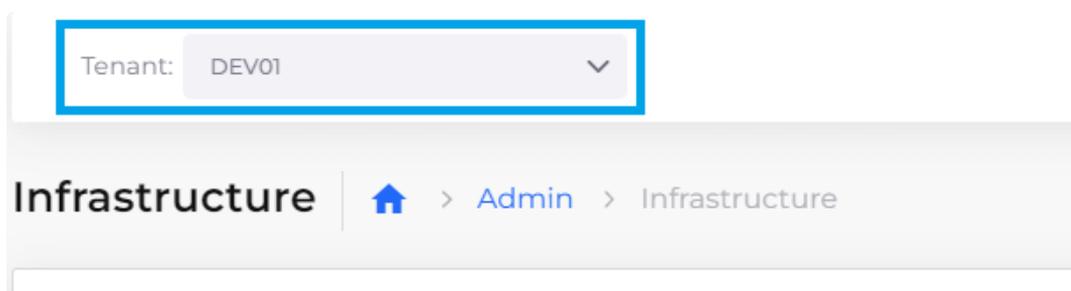
DuploCloud ensures that your application development platform conforms to Azure best practices. While you provide only high-level specifications, DuploCloud does the rest, configuring encryption, linking to managed identity, and logging you into a virtual Linux workstation to access Kubernetes constructs like [Pods](#), [Namespaces](#), and [ConfigMaps](#).

Estimated time to complete Step 3: 10 minutes.

Prerequisites

Before creating Azure Agent Pools, verify that you accomplished the tasks in [Step 2](#) of this tutorial. In DuploCloud Portal, in the **Administrator** navigation group, confirm that you have:

- An [Infrastructure](#) named **NONPROD**.
- A [Plan](#) named **NONPROD**.
- A [Tenant](#) named **DEV01**
- Selected Tenant **DEV01** in the **Tenant** list box, at the top of the DuploCloud Portal.



Tenant list box with Tenant **DEV01** selected

Create an Azure Agent Pool

1. In the DuploCloud Portal, navigate to **DevOps** -> **Hosts**.
2. Click the **Azure Agent Pool** tab.
3. Click **Add**. The **Azure Agent Pool** page displays.
4. From the table below, enter the values corresponding to the fields and options on the **Azure Agent Pool** page. Accept the defaults for fields that are not listed.
5. Click **Add**. After a few minutes, the Azure Agent Pool is created.

Azure Agent Pool page fields and options	Value or action
Id	1
Instance Type	(4 CPU 16GB)

Add Azure Agent Pool

Id

Instance Type

Min Capacity

Max Capacity

Desired Capacity

Allocation Tag

Enable Autoscaling

Cancel
Add

Add Azure Agent Pool page

Check your work

On the **Azure Agent Pool** page, verify that the created agent pool (with a **Name** generated by DuploCloud) has a **Status** of **Succeeded**.

Tenant: DEV01 Switch to Old UI Bob Administrator

Azure Agent Pool | [Home](#) > [DevOps](#) > [Hosts](#) > Azure Agent Pool

Host
VM Scale Set
Azure Agent Pool
BYOH

Total 1 Show 25 + Add

NAME	CAPACITY	DESIRED	AUTO SCALING	MIN	MAX	STATUS	ACTIONS
DLDEV01	Standard_D4s_v3	1	true	1	1	Succeeded	⊙ ⋮

1 total

Successfully creted Azure Agent Pool

Step 4: Create a Service

Create a DuploCloud Service for application deployment

With all of the core components of your Duplocloud platform configured, enabled, and running, you're ready to deploy applications with Azure, using AKG and Kubernetes.

In order to deploy applications, you must first create a DuploCloud Service to connect to the Docker containers and images where your application code exists. Once you create a service from the DuploCloud Portal, you can also perform tasks that you might perform when working with a [Kubernetes service](#). For example, you can view container logs, container state, and container shell, as well as get access to `kubectl`, which allows you to work directly with Kubernetes constructs such as Pods.

In this step, we create a service to connect a [Docker](#) container image with code that displays text on a web page. The Docker container and image name is `nginx:latest`. `nginx` is the image name and `:latest` indicates that the latest version of that image will be used.

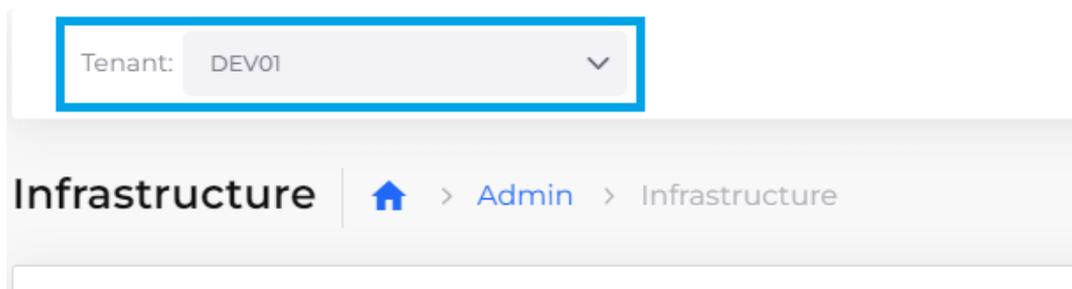
Estimated time to complete Step 4: 15 minutes.

 See the Docker documentation for an [overview of containers and images](#).

Prerequisites

Before creating your DuploCloud Service, ensure that:

- All previous steps in this tutorial to create an [Infrastructure and Plan, Tenant, Host, and Azure Agent Pool](#) are complete.
- The [AKS Kubernetes cluster](#) is enabled.
- Tenant **DEV01** is selected in the **Tenant** list box, at the top of the DuploCloud Portal.



Tenant list box with Tenant **DEV01** selected

Creating a Service

1. In the DuploCloud Portal, navigate to **DevOps** -> **Containers** -> **AKS/Native**.
2. Click **Add**. The **Add Service** page displays.

Add Service

1 Basic Options Minimal Inputs to start service > 2 Advanced Options More options to configure service

Service Name ✓

Cloud x v

Platform x v

Docker Image ✓

Allocation Tag ✓

Replication Strategy x v

Environment Variables

Replicas ✓

← Previous Next →

Add Service page to add nginx-service

3. In the **Service Name** field, enter **nginx-service**.
4. Specify the Docker image that you use to run the application. In the Docker Image field, enter **nginx:latest**.
5. Click **Next**, accepting all other defaults. The **Advanced Options** page displays.
6. Scroll down if needed and click **Create**.

Check your work

After a few minutes, the Service initializes and starts up. Shortly afterward, you can see the service and the containers running.

Tenant: DEV01 Switch to Old UI Bob Administrator

Services [Home](#) > [DevOps](#) > [Containers](#) > [AKS/Native](#) > Services

Services Containers K8S Secrets K8S Config Maps K8S Ingress K8S Storage

Total 1 Show 25 Search KubeCtl Shell KubeCtl Token Kube Config + Add

NAME	IMAGE	DNS	REPLICAS	RUNNING	ACTIONS
nginx-service	nginx:latest	nginx-service-dev01.azure-qa.duplocloud.net	1	1/1	

1 total

nginx-service page with service **RUNNING (1/1)**

Step 5: Create a Load Balancer

Create a load balancer to access your application

Now that your service is running, you have a mechanism to expose the containers and images in which your application resides.

But because your containers are running inside a private network, you also need a load balancer to listen on the correct ports in order to access the application.

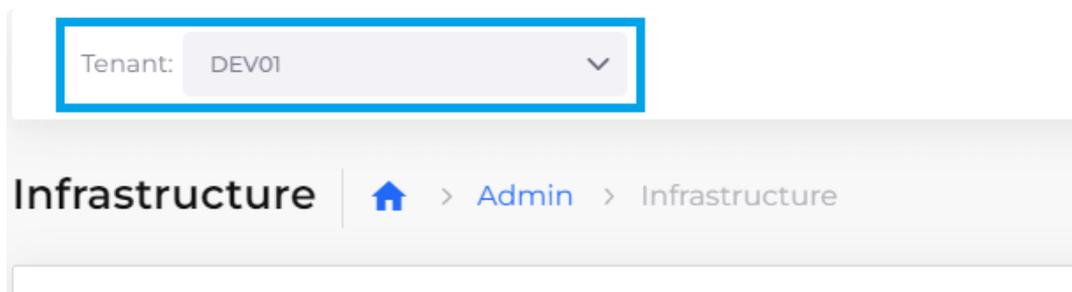
In this step, we add a Load Balancer Listener to complete this network configuration.

Estimated time to complete Step 5: 20 minutes.

Prerequisites

Before creating your DuploCloud load balancer, ensure that:

- All previous steps in this tutorial to create an [Infrastructure and Plan](#), [Tenant](#), [Azure Agent Pool](#), and [Service](#) are complete.
- [AKS Kubernetes cluster](#) is enabled.
- **DEV01** is selected in the **Tenant** list box, at the top of the DuploCloud Portal.



Tenant list box with Tenant DEV01 selected

Adding and configuring a load balancer

Add a load balancer for your running service that listens on port **80**:

1. In the DuploCloud Portal, navigate to **DevOps** -> **Containers** -> **AKS / Native**.
2. In the **Services** tab, select the **nginx-service** that you started when [creating a service in the previous step](#).
3. Click the **Load Balancers** tab.
4. Click the **Configure Load Balancer** link. The **Add Load Balancer Listener** pane displays.
5. Select **K8S Node Port** from the **Select Type** list box.
6. Enter **80** in the **Container port** field.
7. Enter **30008** in the **External port** field.
8. Type **/** (forward-slash) in the **Health Check** field to indicate that the cluster we want Kubernetes to perform Health Checks on is located at the `root` level.
9. Select **TCP** from the **Backend Protocol** list box.

Add Load Balancer Listener ✕

Select Type ⓘ

K8S Node Port ✕ ▼

Container port ⓘ

80 ✓

External port ⓘ

30008 ✓

Health Check ⓘ

/ ✓

Backend Protocol ⓘ

TCP ✕ ▼

Advanced Kubernetes settings

Additional health check configs

Add Load Balancer Listener pane using K8S Node Port

10. Click **Add**. The Load Balancer is created and started. After a few minutes, the **LB Status** card displays a status of **Ready**, indicating that the Load Balancer is ready for use.

The screenshot displays the 'nginx-service Load Balancers' configuration page. At the top, the tenant is 'DEV01' and the user is 'Bob Administrator'. The breadcrumb trail is 'Services > DevOps > Containers > AKS/Native > Services > nginx-service'. The main header shows 'nginx-service Load Balancers' with an 'Actions' dropdown, 'Image: nginx:latest', 'Load Balancers configured for this service', 'KubeCtl' dropdown, 'Replicas: 1', and 'Status: Running'. A 'Copy' button is also present. Below the header, there are tabs for 'Containers', 'Load Balancers', and 'Configuration'. The 'Load Balancers' tab is active, showing 'LB Configuration' (Visibility: N/A - No Cloud LB, Type: K8S Node Port, DNS Name:), 'Certificate' (N/A), and 'LB Listeners'. The 'LB Listeners' section contains a table with one listener:

HOSTNAME	PROTOCOL	EXT PORT	HOST PORT	CONTAINER PORT	IS NATIVE	HEALTH	#
	tcp	30008	30007	80	No	/	✎ ⏸ ⋮

On the right sidebar, there are several status indicators: 'Running 1/1', 'LB Status Ready' (highlighted with a red box), 'DNS n/a', 'LB Visibility External', 'Operating System Linux', and 'Container Platform AKS Linux'.

nginx-service Load Balancers tab with LB Status Ready

Enable the Ingress Controller

When we created the Load Balancer Listener, we used the **K8S Node Port** type. Even though the Node Port is ready, before you use it, you must enable the [Kubernetes Ingress Controller](#) to open the application gateway.

1. In the DuploCloud Portal, navigate to **Administrator -> Infrastructure**.
2. Select your Infrastructure from the **Name** column.
3. Click the **Settings** tab.
4. Click **Add**. The **Infra-Set Custom Data** pane displays.
5. In the **Setting Name** field, select **Enable App Gateway Ingress Controller**.
6. Click **Enable**.
7. Click **Set**. In the **Settings** tab, the **Enable App Gateway Ingress Controller** setting now contains a value of **true**.

The screenshot shows the DuploCloud Infrastructure console for a tenant named 'DEV01'. The main header displays 'NONPROD' with a status of 'Complete'. The environment details include 'Cloud: Azure', 'Region: East US', and 'VNET: 10.23.0.0/16'. The 'Settings' tab is active, showing a table with one entry: 'Enable App Gateway Ingress Controller' with a value of 'true'. Other tabs include Subnets, Security Group Rules, Kubernetes, Log Analytics WS, Vault Config, and Metadata. On the right, there are status indicators for 'Subnets: 2' and 'Kubernetes: Enabled'.

NAME	VALUE	ACTIONS
Enable App Gateway Ingress Controller	true	:

NONPROD Infrastructure page with Enable App Gateway Ingress Controller set to true

Add Kubernetes Ingress

Now that your gateway is established and opened, you add a [Kubernetes Ingress](#) to expose the backend HTTP routes from outside the cluster to your service.

The Ingress object communicates with the Kubernetes NodePort that your Load Balancer Listener uses. Ingress objects are flexible constructs in Kubernetes, and their use here is an example of how DuploCloud leverages the power of Kubernetes constructs while abstracting away their native complexity. To manually create these components (and maintain them) in Kubernetes, takes a significant amount of developer time.

1. In the DuploCloud Portal, navigate to **DevOps** -> **Containers** -> **AKS / Native**.
2. Click the **K8S Ingress** tab.
3. Click **Add**. The **Add Kubernetes Ingress** page displays.
4. In the **Ingress Name** field, type **viewwebsite**.
5. In the **Ingress Controller** list box, select **azure-application-gateway**.
6. In the **Visibility** list box, select **Public**.
7. Click **Add Rule**. The **Add Ingress Rule** pane displays.

The screenshot shows the 'Add Kubernetes Ingress' configuration page. At the top, there is a breadcrumb trail: Ingress > DevOps > Containers > AKS / Native > Ingress > Add. Below this, the 'Add Kubernetes Ingress' form is displayed. It includes several input fields: 'Ingress Name' with the value 'viewwebsite', 'Ingress Controller' with a dropdown menu showing 'azure-application-gateway', 'Visibility' with a dropdown menu showing 'Public', 'Certificate ARN', and 'Port Override'. Below these fields is a table for 'Ingress Rules' with columns: PATH, PATH TYPE, HOST, SERVICE, and ACTIONS. The table currently shows 'No data to display'. A red box highlights the 'Add Rule' button. At the bottom of the form, there are 'Annotations' and 'Labels' sections, each with a text area containing the number '1'. Finally, there are 'Cancel' and 'Add' buttons at the bottom left.

Add Rule option on the Add Kubernetes Ingress page

8. In the **Path** field, type **/** (forward-slash).
9. In the **Service Name** field, select **nginx-service:80**.
10. Click **Add Rule** to add the rule and to close the **Add Ingress Rule** pane. You should be back to viewing the **Add Kubernetes Ingress** page.

The screenshot shows the 'Add Ingress Rule' configuration pane. It has a title bar with 'Add Ingress Rule' and a close button (X). Below the title bar, there are several input fields: 'Path' with the value '/', 'Path Type' with a dropdown menu showing 'Prefix', 'Host' with the value 'Host', 'Service Name' with a dropdown menu showing 'nginx-service:80', and 'Container port'. The 'Path' and 'Service Name' fields are highlighted with green boxes and have green checkmarks next to them.

CancelAdd Rule

Add Ingress Rule pane

11. On the **Add Kubernetes Ingress** page, click **Add** to add Ingress. On the **K8S Ingress** tab, the **VIEWWEBSITE** Ingress that you defined, with an **Ingress Class** of **azure-application-gateway**, displays.

NAME	INGRESSCLASS	ACTIONS
VIEWWEBSITE	azure-application-gateway	

K8S Ingress tab displaying VIEWWEBSITE Ingress

Check your work

Before you proceed to the final step and run your application, ensure that you:

- Configured a Load Balancer Listener that uses **K8S Node Port**.
- Enabled the **App Gateway Ingress Controller**.
- **Defined an Ingress and a rule** for your DuploCloud Service to listen on port 80.

Step 6: Test your application

Test the application to ensure you get the results you expect

You can test the sample web page application directly from the **VIEWWEBSITE** Ingress [that you created in the previous step](#).

Estimated time to complete Step 6 and finish tutorial: 10 minutes.

Prerequisites

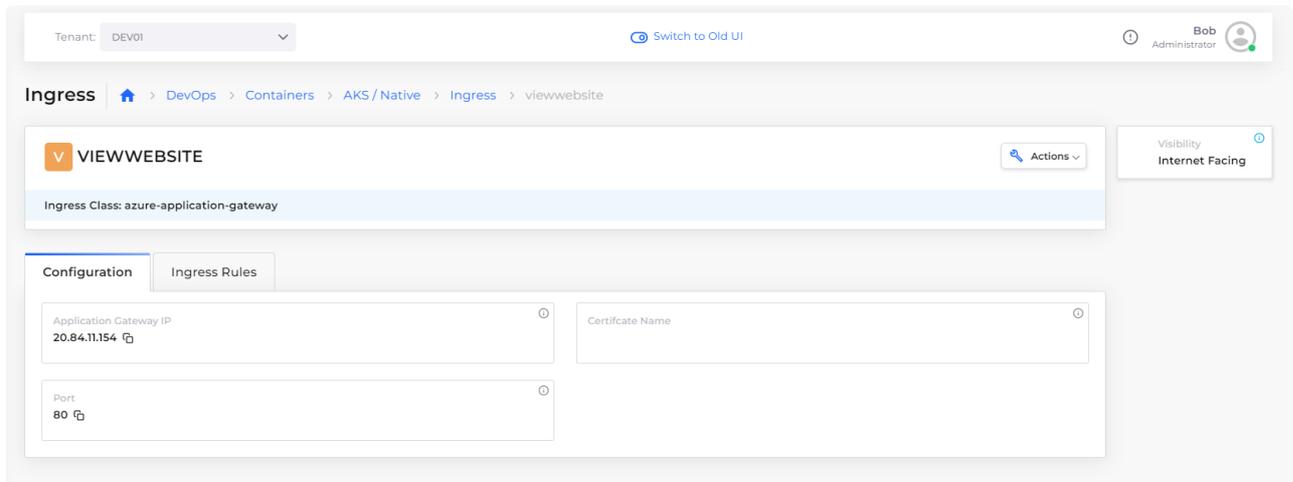
Before testing your application, ensure that:

- Previous steps in this tutorial to create an [Infrastructure and Plan](#), [Tenant](#), [Azure Agent Pool](#), [Service](#), and [Load Balancer Listener](#) are complete.
- [AKS Kubernetes cluster](#) is enabled.
- Tenant **DEV01** is selected in the **Tenant** list box, at the top of the DuploCloud Portal.
- [nginx-service](#) is **Running** and the **LB Status** is **Ready**.
- You [defined an Ingress and a rule](#) for your DuploCloud Service to listen on port 80, enabled the [App Gateway Ingress Controller](#), and configured a Load Balancer Listener that uses [K8S Node Port](#).

Testing the application

Display the web page that the application creates:

1. In the DuploCloud Portal, navigate to **DevOps** -> **Containers** -> **AKS / Native**.
2. Click the **K8S Ingress** tab.
3. Select the **VIEWWEBSITE** Ingress from the **Name** column.
4. Click the **Configuration** tab.



5. In the **Application Gateway IP** card, copy the displayed IP address to your clipboard. In this example, the IP address is **20.84.11.154**.
6. Open a web browser and paste the copied IP address in your browser's URL field.
7. Press **Enter**. Your application runs and your web page renders as shown below. Congratulations! You just launched your first web service with Azure on DuploCloud!



Reviewing what you learned

In this tutorial, your objective was to create a cloud environment to deploy an application for testing purposes, and to understand how the various components of DuploCloud work together.

The application rendered a simple web page with text, coded in JavaScript, from software application code residing in a Docker container. You can use this same procedure to deploy much more complex cloud applications.

In the previous steps, you:

- [Created a DuploCloud Infrastructure](#) named **NONPROD**, a Virtual Private Cloud instance, backed by an AKS-enabled Kubernetes cluster.
- [Created a Tenant](#) named **DEV01** in Infrastructure **NONPROD**. While generating the Infrastructure, DuploCloud created a set of templates ([Plan](#)) to configure multiple Azure and Kubernetes components needed for your environment.
- [Created an Azure Agent Pool](#) backed by pre-existing hosts (VMs), so that your application has storage resources with which to run.
- [Created a Service](#) to connect the Docker containers and associated images, in which your application code resides, to the DuploCloud Tenant environment.
- [Created a Load Balancer Listener](#) and a Kubernetes Node Port to expose your application via ports and backend network configurations. You enabled an Azure application gateway and created a Kubernetes Ingress to communicate with the node port and the AKS-enabled Kubernetes cluster in the Infrastructure.
- [Verified that your web page rendered](#) as expected by testing the IP address exposed by the Kubernetes Ingress.

Cleaning up your tutorial environment

In this tutorial, you created many artifacts for testing purposes. When you are ready, clean them up so that another person can run this tutorial from the start, using the same names for Infrastructure and Tenant.

1. To delete the **DEV01** tenant [follow these instructions](#) and then return to this page. As you learned, the Tenant segregates all work in one isolated environment, so deleting the Tenant that you created cleans up most of your artifacts.
2. Finish by deleting the **NONPROD** Infrastructure. In the DuploCloud Portal, navigate to **Administrator** -> **Infrastructure**. Click the **Action** menu icon () for the **NONPROD** row and select **Delete**.

The **NONPROD** Infrastructure is deleted and you have completed the clean-up of your test environment.

Thanks for completing this tutorial and proceed to the next section to learn more about using DuploCloud with Microsoft Azure.