Get Started with Enterprise Tier

Follow up below instructions to get started with Enterprise Tier in Azure Spring Cloud. Table of content is listed as below.

Pre-prequisite

- You'll need an Azure subscription before you begin. If you don't have one, create an account. For more information, see Marketplace for Enterprise tiers.
- Install the Azure CLI version 2.0.67 or higher
- Install the preview version of Azure Spring Cloud extension for Enterprise tier with command:

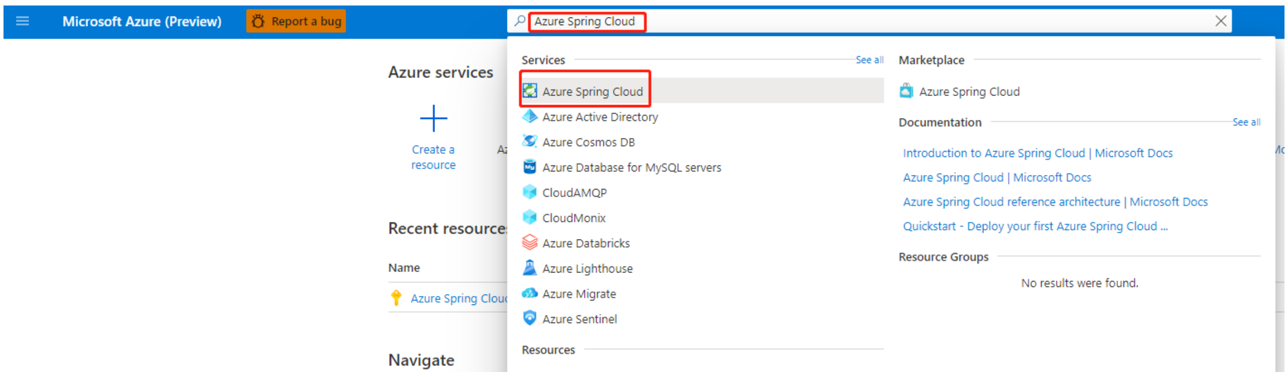
```
az extension remove -n spring-cloud
az extension add -s
https://ascprivatecli.blob.core.windows.net/enterprise/spring_cloud-2.7.0a1-
py3-none-any.whl -y
```

Provision Service Instance

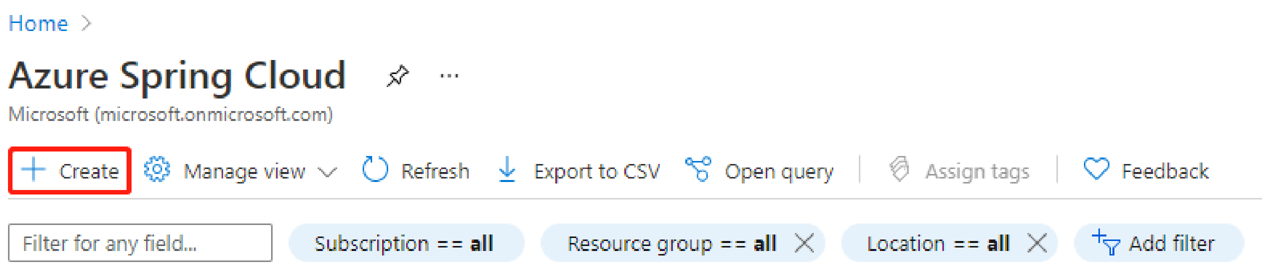
1. Open the Azure portal.

2. From the top search box, search for Azure Spring Cloud.

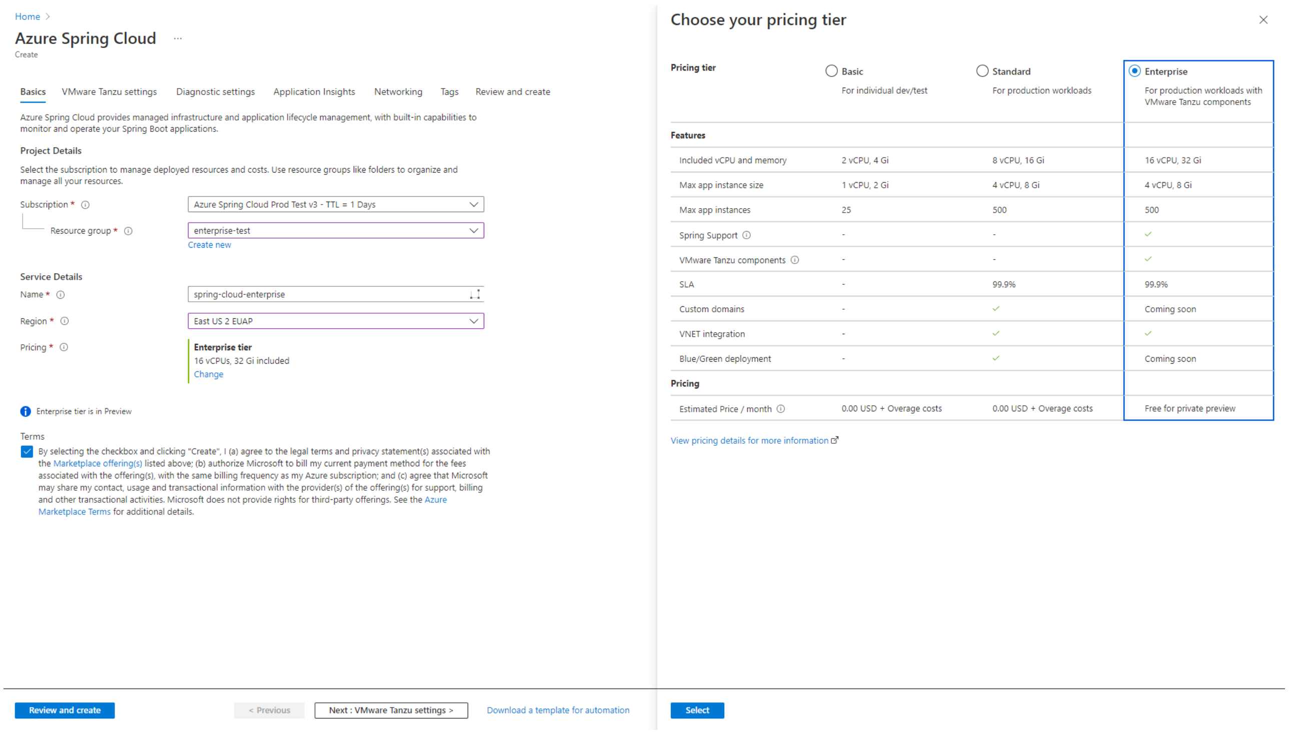
3. Select Azure Spring Cloud from the results.



4. On the Azure Spring Cloud page, click + Create.



5. Go to the Azure Spring Cloud **Create** page. In **Pricing** option, click **Change** and choose **Enterprise** tier.



Azure Spring Cloud provides managed infrastructure and application lifecycle management, with built-in capabilities to monitor and operate your Spring Boot applications.

Project Details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Azure Spring Cloud Prod Test v3 - TTL = 1 Days
Resource group *	enterprise-test Create new

Service Details

Name *	spring-cloud-enterprise
Region *	East US 2 EUAP
Pricing *	Enterprise tier 16 vCPUs, 32 Gi included Change

Enterprise tier is in Preview

Terms

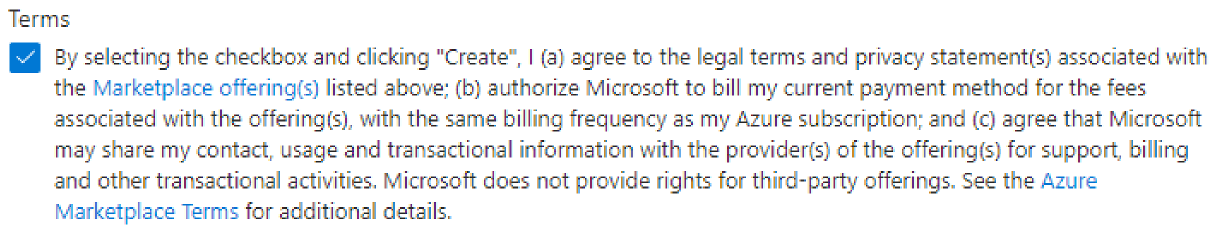
By selecting the checkbox and clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the Azure Marketplace Terms for additional details.

Choose your pricing tier

Pricing tier	Basic For individual dev/test	Standard For production workloads	[✓ Enterprise] For production workloads with VMware Tanzu components
Features			
Included vCPU and memory	2 vCPU, 4 Gi	8 vCPU, 16 Gi	16 vCPU, 32 Gi
Max app instance size	1 vCPU, 2 Gi	4 vCPU, 8 Gi	4 vCPU, 8 Gi
Max app instances	25	500	500
Spring Support	-	-	✓
VMware Tanzu components	-	-	✓
SLA	-	99.9%	99.9%
Custom domains	-	✓	Coming soon
VNET integration	-	✓	✓
Blue/Green deployment	-	✓	Coming soon
Pricing			
Estimated Price / month	0.00 USD + Overage costs	0.00 USD + Overage costs	Free for private preview

View pricing details for more information

Check **Terms** checkbox to agree all legal terms and privacy statements of Enterprise tier offering in Azure Marketplace.

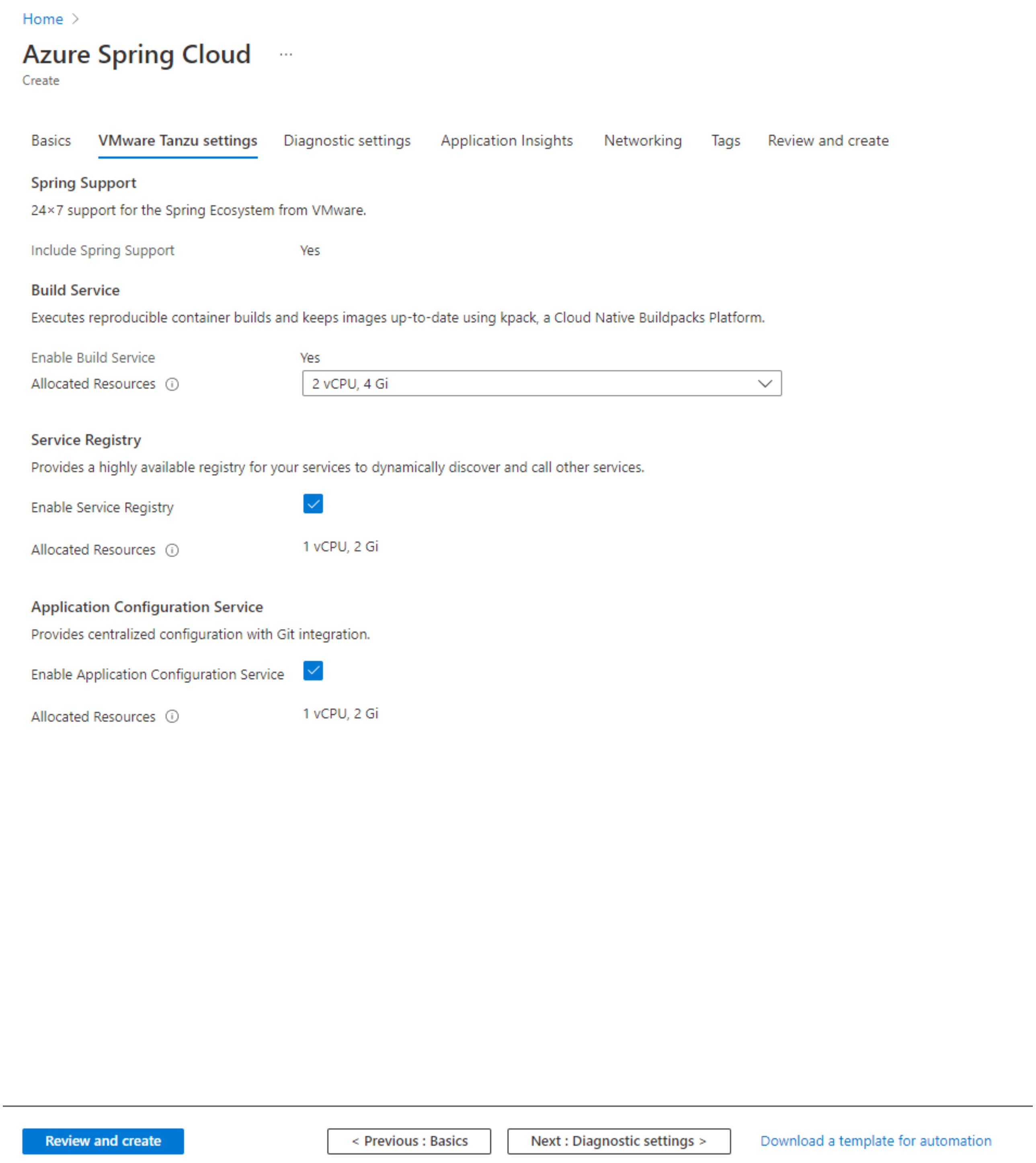


Terms

By selecting the checkbox and clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the Azure Marketplace Terms for additional details.

6. Click **Next: VMware Tanzu settings** button at the bottom right of the page to configure VMware Tanzu components.

> **NOTE**
>
> By default, Service Registry and Application Configuration Service is enabled. Suggest to keep the default value. Service Registry and Application Configuration Service cannot be enabled after service instance is provisioned in the current phase.


Home >

Azure Spring Cloud ...

Create

Basics | **VMware Tanzu settings** | Diagnostic settings | Application Insights | Networking | Tags | Review and create

Spring Support

24×7 support for the Spring Ecosystem from VMware.

Include Spring Support	Yes

Build Service

Executes reproducible container builds and keeps images up-to-date using kpack, a Cloud Native Buildpacks Platform.

Enable Build Service	Yes
Allocated Resources	2 vCPU, 4 Gi

Service Registry

Provides a highly available registry for your services to dynamically discover and call other services.

Enable Service Registry	✓
Allocated Resources	1 vCPU, 2 Gi

Application Configuration Service

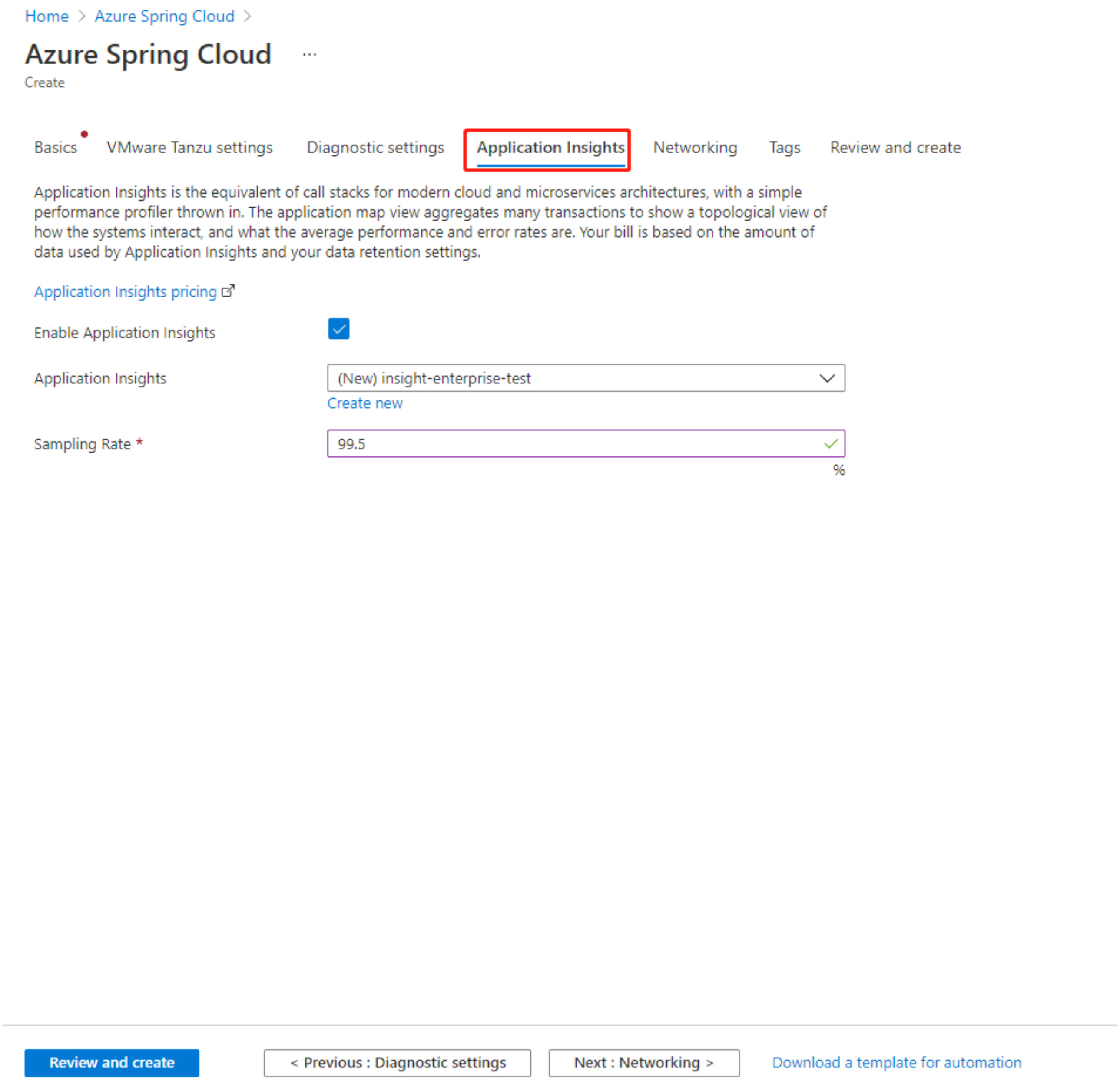
Provides centralized configuration with Git integration.

Enable Application Configuration Service	✓
Allocated Resources	1 vCPU, 2 Gi

Review and create
< Previous : Basics
Next : Diagnostic settings >
Download a template for automation

7. On the Application Insights tab, check **Enable Application Insights** checkbox. Application Insights can also be enabled after service instance is provisioned.
   - Choose an existing Application Insights instance or create a new Application Insights instance.
   - Give a **Sampling Rate** with range 0-100 or use default value 10.

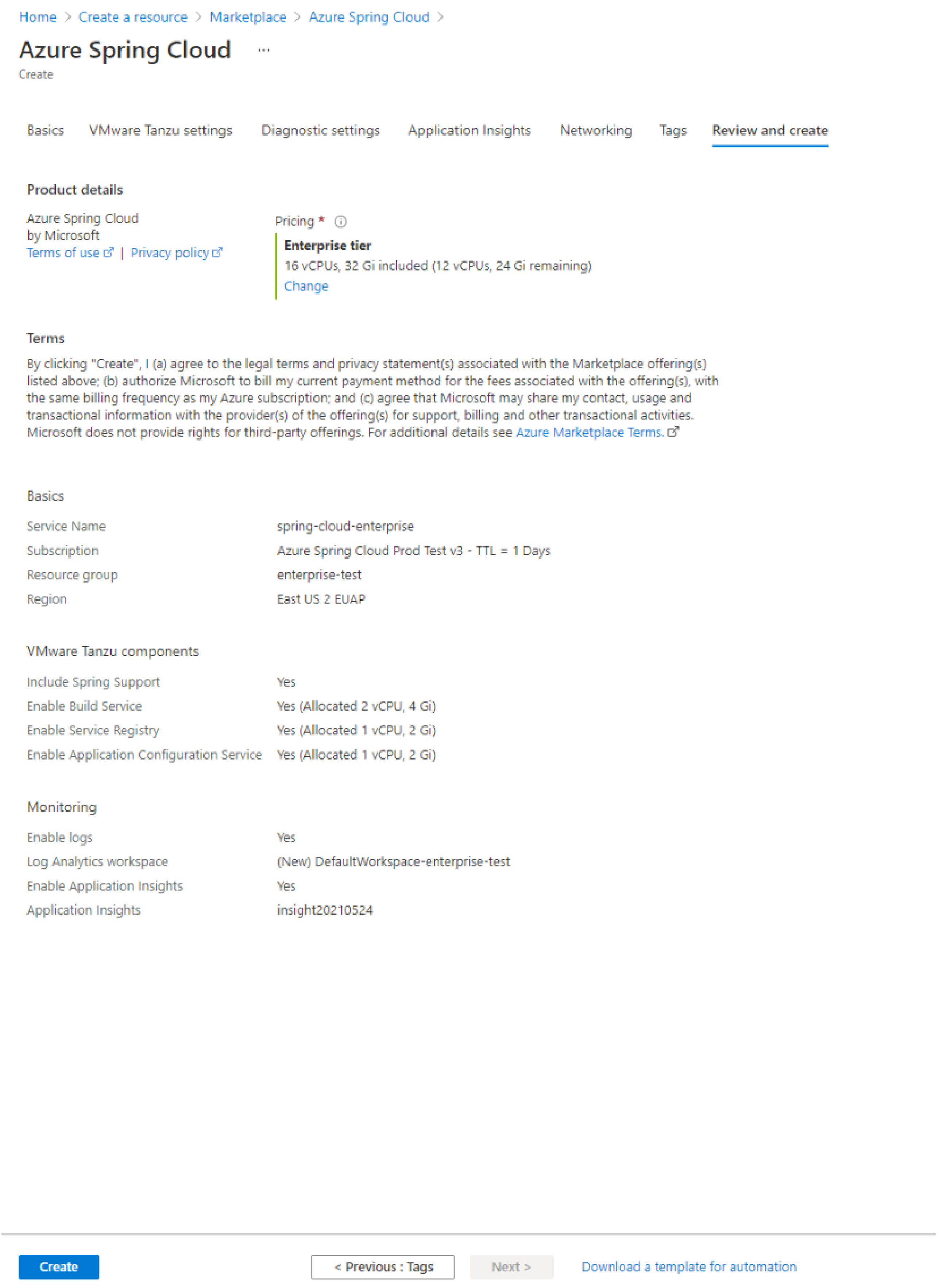
> **Note**
>
> - Please take note of the instance name of Application Insights, which is not shown in portal after provisioning.
> - You will pay for the usage from the instance of Application Insights integrated with Azure Spring Cloud. Check more details about Application Insights pricing information at Manage usage and costs for Application Insights.



Application Insights is the equivalent of call stacks for modern cloud and microservices architectures, with a simple performance profiler thrown in. The application map view aggregates many transactions to show a topological view of how the systems interact, and what the average performance and error rates are. Your bill is based on the amount of data used by Application Insights and your data retention settings.

Enable Application Insights	✓
Application Insights	(New) insight-enterprise-test
Sampling Rate *	99.5 %

8. Click **Review and create** button at the bottom left of the page. After validation is completed successfully, click **Create** button to start provisioning service instance.



Home > Create a resource > Marketplace > Azure Spring Cloud >

Azure Spring Cloud ···

Create

Product details

Azure Spring Cloud
by Microsoft
Terms of use | Privacy policy

Pricing *

Enterprise tier
16 vCPUs, 32 Gi included (12 vCPUs, 24 Gi remaining)
Change

Terms

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. For additional details see Azure Marketplace Terms.

Basics

Service Name	spring-cloud-enterprise
Subscription	Azure Spring Cloud Prod Test v3 - TTL = 1 Days
Resource group	enterprise-test
Region	East US 2 EUAP

VMware Tanzu components

Include Spring Support	Yes
Enable Build Service	Yes (Allocated 2 vCPU, 4 Gi)
Enable Service Registry	Yes (Allocated 1 vCPU, 2 Gi)
Enable Application Configuration Service	Yes (Allocated 1 vCPU, 2 Gi)

Monitoring

Enable logs	Yes
Log Analytics workspace	(New) DefaultWorkspace-enterprise-test
Enable Application Insights	Yes
Application Insights	insight20210524

9. It takes about 5 minutes to finish the resource provisioning.

Create and Configure Apps

Create apps on Azure Spring Cloud

1. If you didn't run the following commands in the previous quickstarts, set the CLI defaults.

```
az configure --defaults group=<resource group name> spring-cloud=<service
name>
```

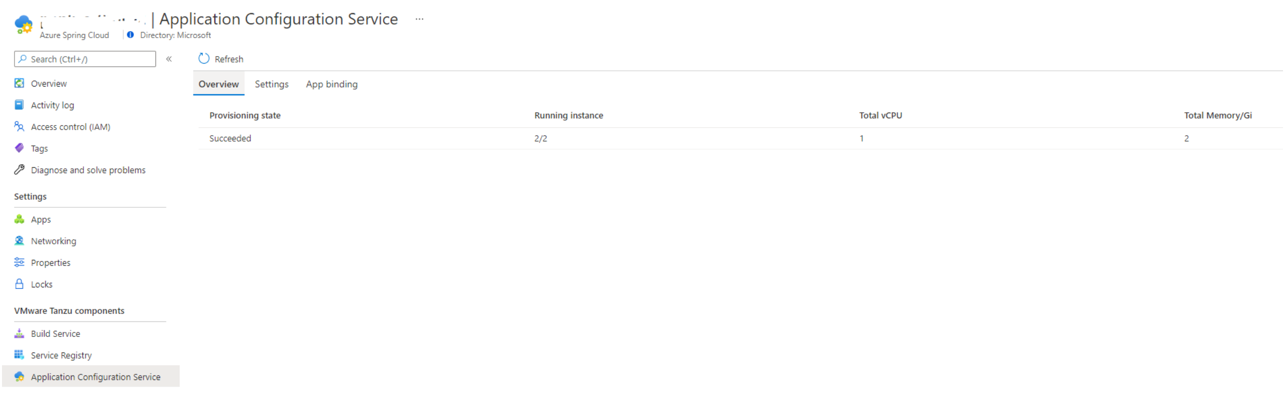
2. Create the 2 core microservices for PetClinic: API gateway and customers-service.

```
az spring-cloud app create --name api-gateway --instance-count 1 --memory
2Gi --assign-endpoint
az spring-cloud app create --name customers-service --instance-count 1 --
memory 2Gi
```

Use Application Configuration Service

> **Precondition**
>
> Application Configuration Service should be enabled when service instance is provisioned. It cannot be enabled after provisioning.

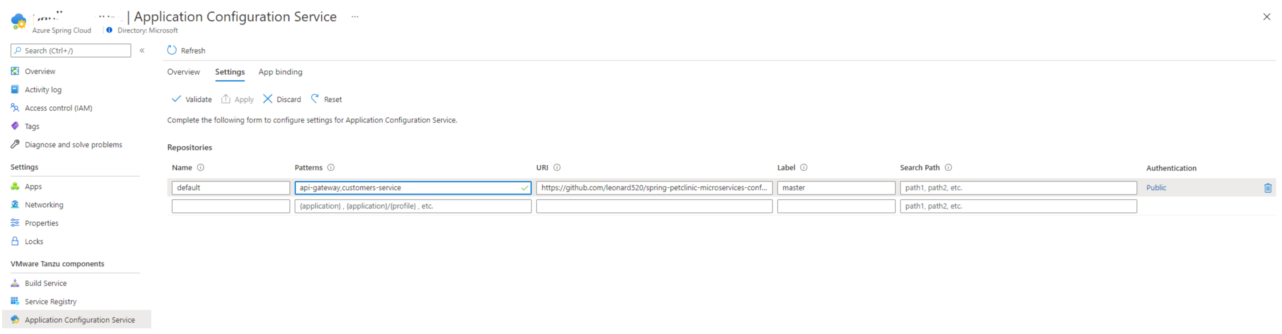
1. Open **Application Configuration Service** blade to see the **Overview** tab, which shows the running state and resources allocated to Application Configuration Service.



Provisioning state	Running instance	Total vCPU	Total Memory/Gi
Succeeded	2/2	1	2

2. Open **Settings** blade, and add a new entry with below information in **Repositories** section.
   - Name: `default`
   - Patterns: `api-gateway,customers-service`
   - URI: `https://github.com/leonard520/spring-petclinic-microservices-config`
   - Label: `master`

3. Click **Validate** button to validate access to the target URI. After validation is completed successfully, click **Apply** button to update configuration settings.

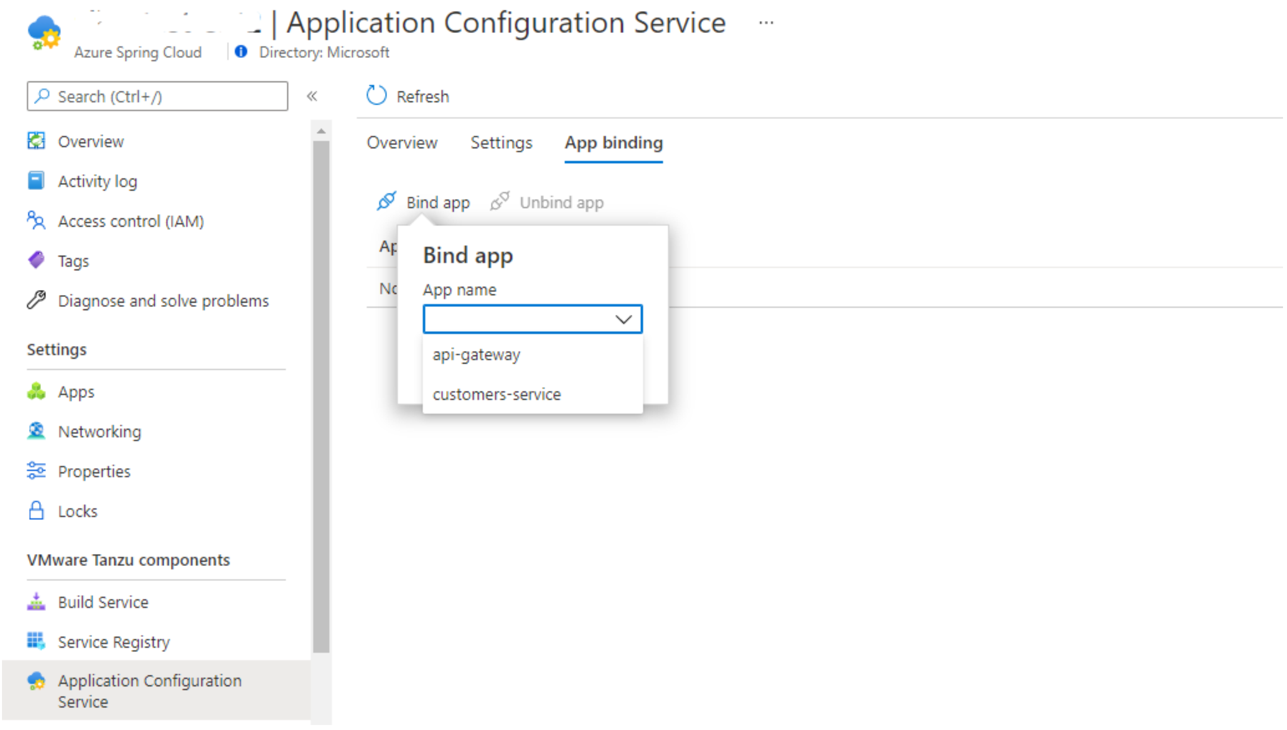

Complete the following form to configure settings for Application Configuration Service.

Repositories

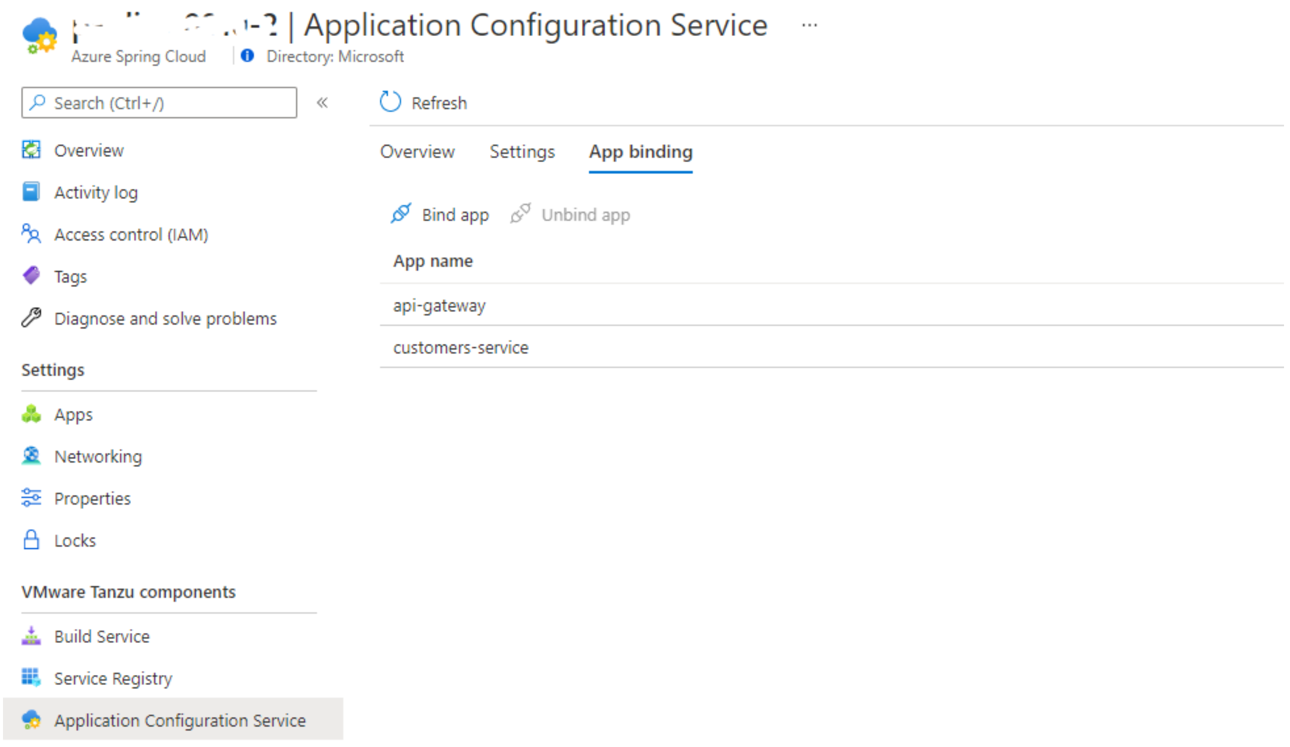
Name	Patterns	URI	Label	Search Path	Authentication
default	api-gateway,customers-service	https://github.com/leonard520/spring-petclinic-microservices-conf...	master	path1, path2, etc.	Public
	{application} , {application}/{profile} , etc.			path1, path2, etc.	

4. Open **App binding** tab to bind/unbind app to the Application Configuration Service.

   a. Click **Bind app** button and choose one app in the dropdown. Click **Apply** button to bind.



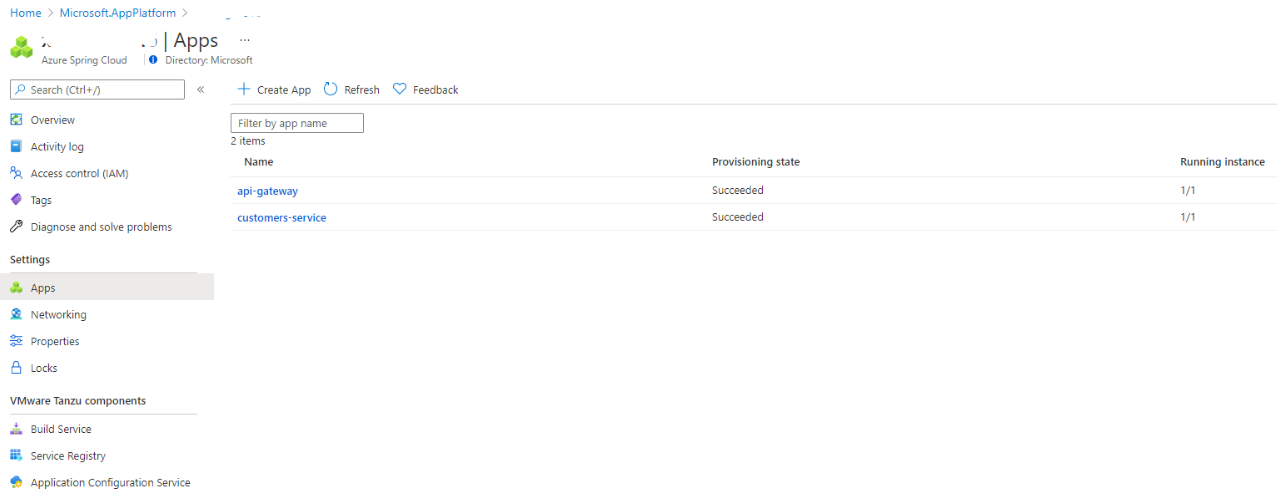
b. The list shows the bound apps with Application Configuration Service:



App name
api-gateway
customers-service

5. Go to **Apps** blade and choose the pattern(s) to be used by the apps.

a. Open **Apps** blade to list all the apps.

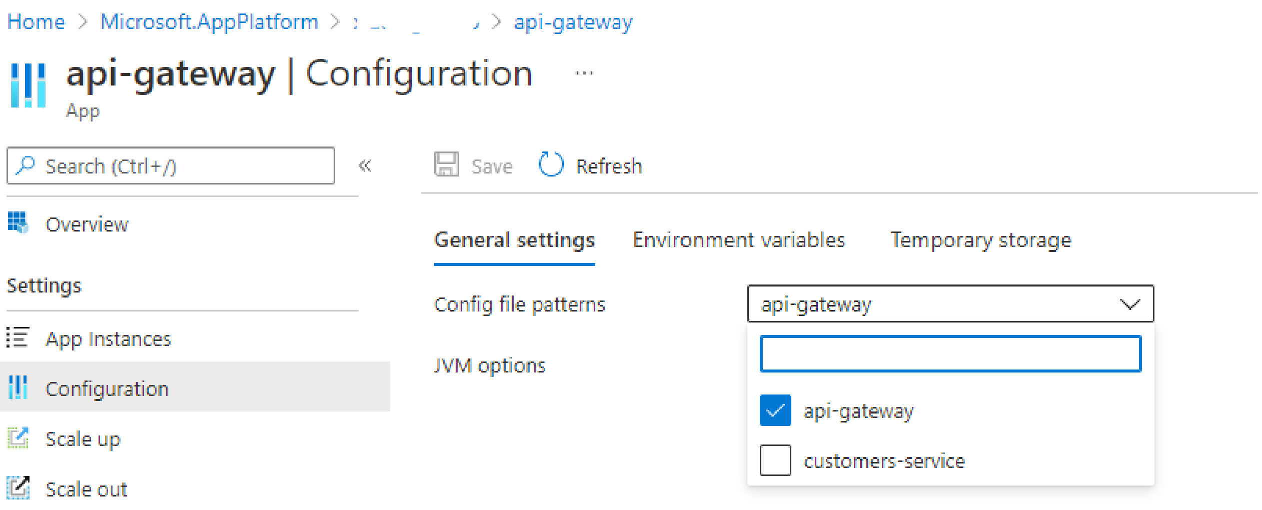


2 items

Name	Provisioning state	Running instance
api-gateway	Succeeded	1/1
customers-service	Succeeded	1/1

b. Click **api-gateway**.

c. Open **Configuration** menu and **General settings** tab. Under **Config file patterns** property, choose one or more patterns from the dropdown list. Click **Save** button to save the changes.



Config file patterns: api-gateway

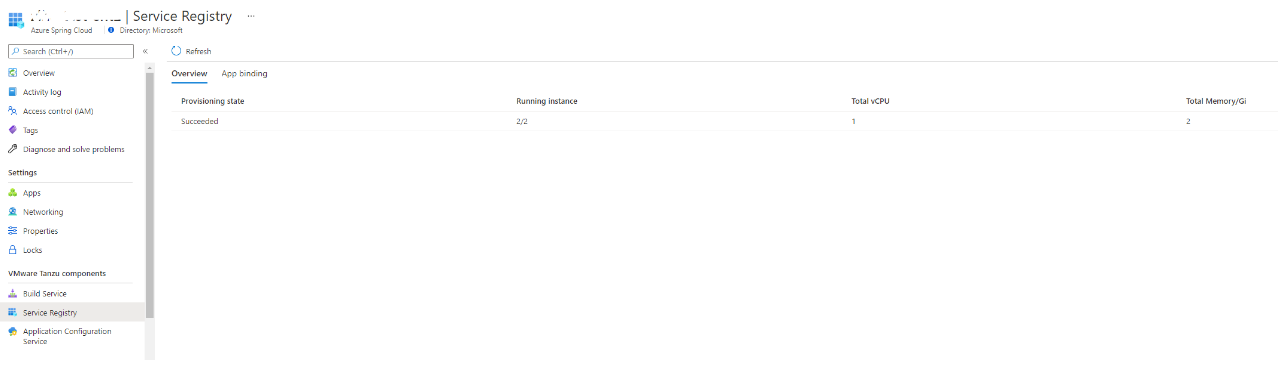
- [x] api-gateway
- [ ] customers-service

d. Apply step b & c to **customers-service** and choose the pattern `customers-service`.

Use Service Registry

> **Precondition**
>
> Service Registry should be enabled when service instance is provisioned. It cannot be enabled after provisioning.

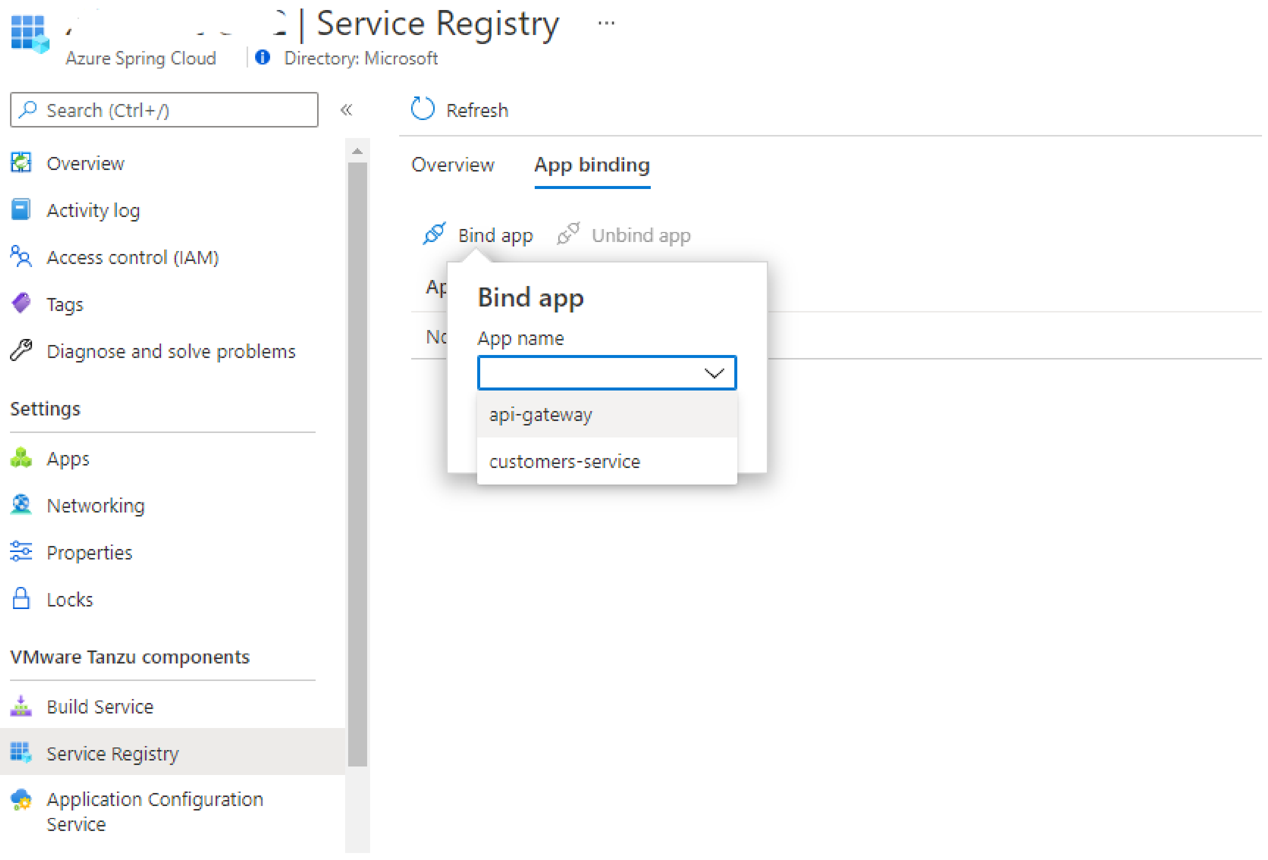
1. Open **Service Registry** blade to see the **Overview** tab, which shows the running state and resources allocated to Service Registry.



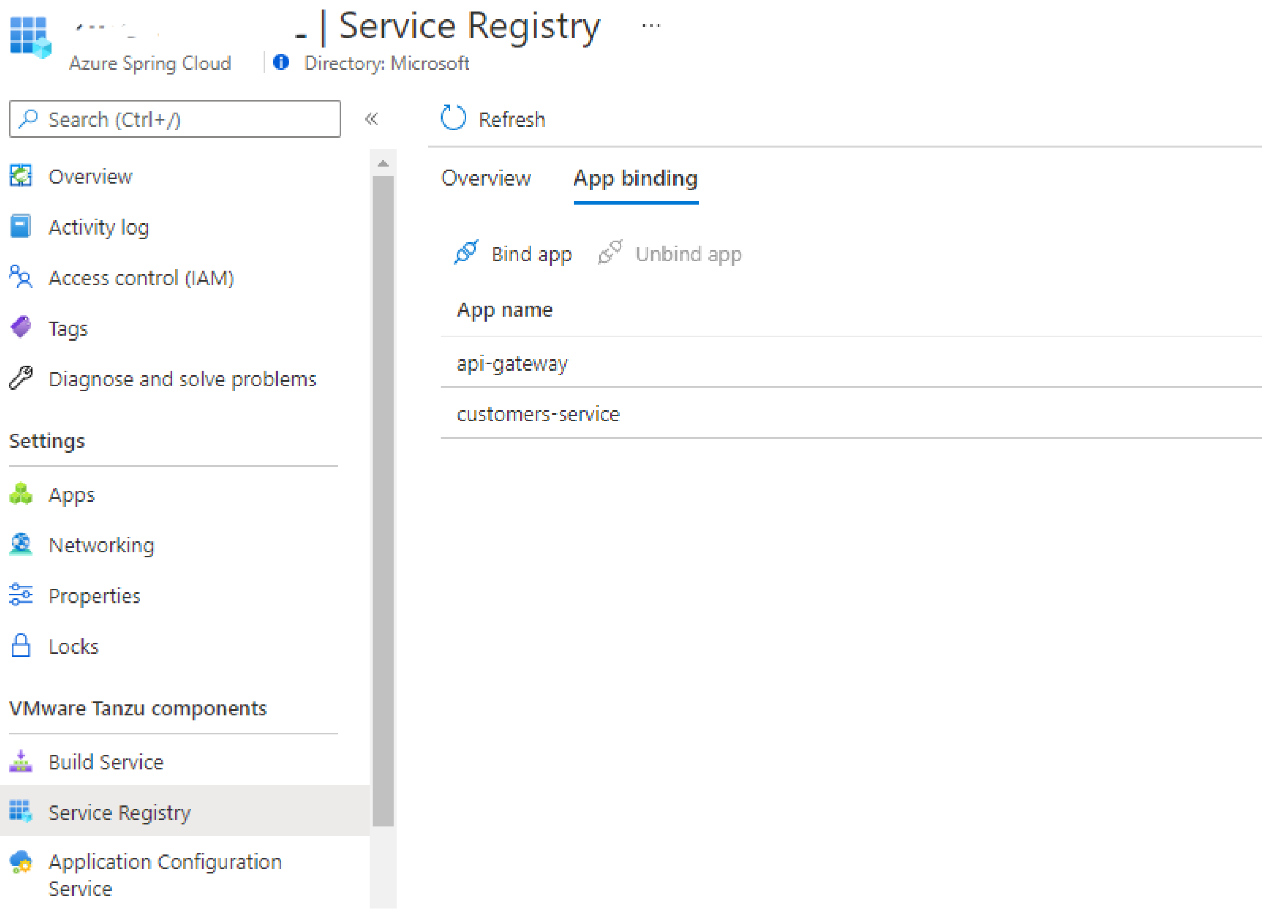
Provisioning state	Running instance	Total vCPU	Total Memory/Gi
Succeeded	2/2	1	2

2. Open **App binding** tab to bind/unbind app to the Service Registry.

a. Click **Bind app** button and choose one app in the dropdown. Click **Apply** button to bind.



b. The list shows the bound apps with Service Registry:



App name
api-gateway
customers-service

Deploy Apps

Build the microservices apps locally

1. Clone the sample app repository to your Azure Cloud account. Change the directory, and build the project.

   ```
   git clone -b xiading/enterprise https://github.com/leonard520/spring-
   petclinic-microservices.git
   cd spring-petclinic-microservices
   mvn clean package -DskipTests
   ```

   Compiling the project takes 5 - 10 minutes. Once compilation is completed, you will have individual JAR files for each service in their respective folders.

   ⚠ Please DON'T include `spring-cloud-starter-config` in the app's pom.xml
   ~~<dependency>~~
   ~~<groupId>org.springframework.cloud</groupId>~~
   ~~<artifactId>spring-cloud-starter-config</artifactId>~~
   ~~</dependency>~~

2. Deploy the JAR files built in the previous step.

   ```
   az spring-cloud app deploy --name api-gateway --artifact-path spring-
   petclinic-api-gateway/target/spring-petclinic-api-gateway-2.3.6.jar
   az spring-cloud app deploy --name customers-service --artifact-path spring-
   petclinic-customers-service/target/spring-petclinic-customers-service-
   2.3.6.jar
   ```

3. Query app status after deployments with the following command.

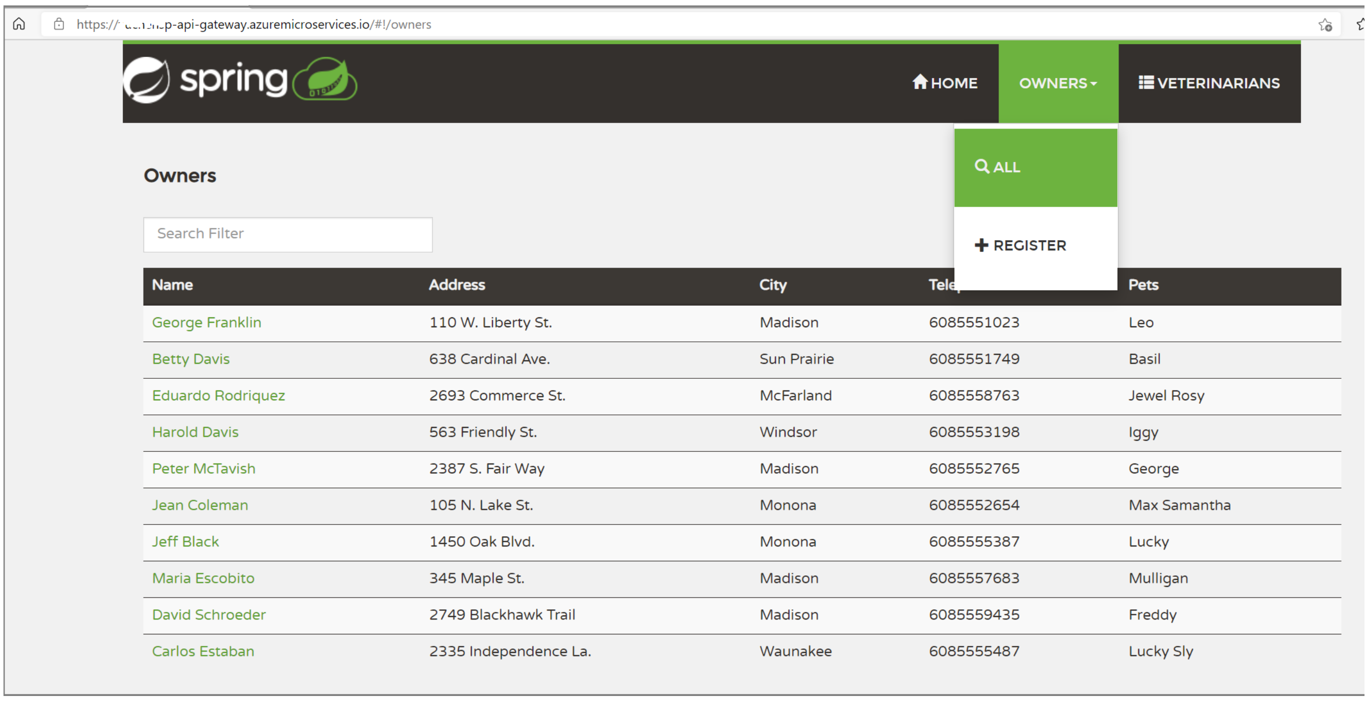
   ```
   az spring-cloud app list -o table
   ```

   ```
   Name                  Location    ResourceGroup       Public Url
   Production Deployment    Provisioning State    CPU    Memory    Running
   Instance    Registered Instance    Persistent Storage    Bind Service
   Registry    Bind Application Configuration Service
   --------------------  ----------  ---------------  -------------------------
   --------------------------------  ----------------------  -----------------
   ---  -----  --------  ------------------  ---------------------  -----------
   ---------  -----------------------  ----------------------------------------
   api-gateway           eastus      <resource group>   https://<service name>-
   api-gateway.asc-test.net                    default
   Succeeded             1      2Gi       1/1                  1/1
   ```

```
-                     True                         True
customers-service     eastus      <resource group>
default                  Succeeded             1      2Gi       1/1
1/1                    -                    True                         True
```

Verify the microservices

Access the app gateway and customers service from browser with the **Public Url** shown above, in the format of `https://<service name>-api-gateway.azuremicroservices.io`.



Owners

Name	Address	City	Tele	Pets
George Franklin	110 W. Liberty St.	Madison	6085551023	Leo
Betty Davis	638 Cardinal Ave.	Sun Prairie	6085551749	Basil
Eduardo Rodriquez	2693 Commerce St.	McFarland	6085558763	Jewel Rosy
Harold Davis	563 Friendly St.	Windsor	6085553198	Iggy
Peter McTavish	2387 S. Fair Way	Madison	6085552765	George
Jean Coleman	105 N. Lake St.	Monona	6085552654	Max Samantha
Jeff Black	1450 Oak Blvd.	Monona	6085555387	Lucky
Maria Escobito	345 Maple St.	Madison	6085557683	Mulligan
David Schroeder	2749 Blackhawk Trail	Madison	6085559435	Freddy
Carlos Estaban	2335 Independence La.	Waunakee	6085555487	Lucky Sly

Real-time app log streaming

Use the following command to get real-time logs from the app.

```
az spring-cloud app logs -n <app-name> -s <service-instance-name> -g <resource-group> --lines 100 -f
```

This will return logs:

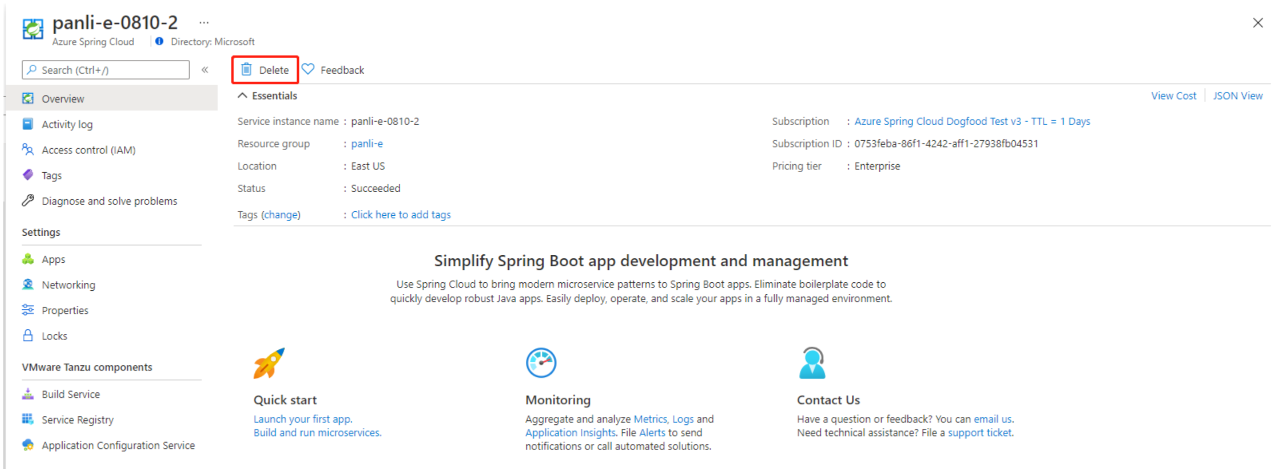
```
2021-07-15 01:54:40.481  INFO [auth-service,,,] 1 --- [main]
o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2021-07-15 01:54:40.482  INFO [auth-service,,,] 1 --- [main]
org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache
Tomcat/9.0.22]
2021-07-15 01:54:40.760  INFO [auth-service,,,] 1 --- [main] o.a.c.c.C.[Tomcat].
[localhost].[/uaa]  : Initializing Spring embedded WebApplicationContext
2021-07-15 01:54:40.760  INFO [auth-service,,,] 1 --- [main]
o.s.web.context.ContextLoader  : Root WebApplicationContext: initialization
completed in 7203 ms
```

Monitor Apps with Application Insights

Read more in Monitor Apps with Application Insights

Clean up resources

1. Open the Azure portal. Delete the service instance as below screenshot.



panli-e-0810-2
Azure Spring Cloud | Directory: Microsoft

Service instance name : panli-e-0810-2
Resource group : panli-e
Location : East US
Status : Succeeded
Tags (change) : Click here to add tags
Subscription : Azure Spring Cloud Dogfood Test v3 - TTL = 1 Days
Subscription ID : 0753feba-86f1-4242-aff1-27938fb04531
Pricing tier : Enterprise

Simplify Spring Boot app development and management

Use Spring Cloud to bring modern microservice patterns to Spring Boot apps. Eliminate boilerplate code to quickly develop robust Java apps. Easily deploy, operate, and scale your apps in a fully managed environment.

Quick start
Launch your first app.
Build and run microservices.

Monitoring
Aggregate and analyze Metrics, Logs and Application Insights. File Alerts to send notifications or call automated solutions.

Contact Us
Have a question or feedback? You can email us.
Need technical assistance? File a support ticket.

2. Run below command to remove the preview version of CLI extension.

```
az extension remove -n spring-cloud
```

Azure Marketplace

Overview

Azure Spring Cloud Enterprise Tier is optimized for the needs of enterprise Spring developers through advanced configurability, flexibility, portability, and enterprise-ready VMware Spring Runtime 24x7 support. Developers also benefit from proprietary Tanzu components such as Tanzu Build Service, Tanzu Application Configuration Service and Tanzu Service Registry; and access to Spring experts. Customers obtain and pay for a license to Tanzu components through an Azure Marketplace offer. Azure Spring Cloud manages the license acquisition so that you won't have to do it separately. To purchase in the Azure Marketplace, you must meet the following prerequisties:

- Your Azure subscription has a valid payment instrument. Azure credits or free MSDN subscriptions aren't supported.
- Your organization allows Azure Marketplace purchases.
- Your private Azure Marketplace must contain the Azure Spring Cloud Enterprise Tier w/VMware Tanzu offer.

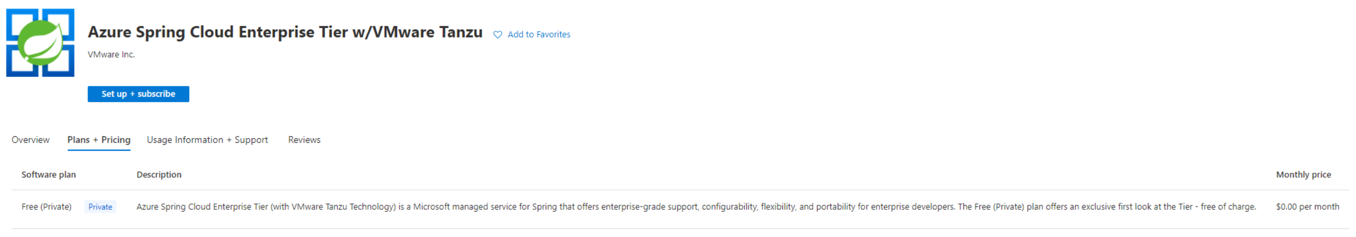
This document guides you how to include Azure Spring Cloud Enterprise Tier w/VMware Tanzu offer to your private Azure Marketplace and how to redirect to Azure Spring Cloud Enterprise tier creation page from Azure

Marketplace.

View Azure Spring Cloud Enterprise Tier w/VMware Tanzu offer from your private Azure Marketplace

Visit Azure Spring Cloud Enterprise Tier w/VMware Tanzu, you can see the offer and read detailed description of the offer.

Click "Plans + Pricing", then you will see the supported plans in your market.



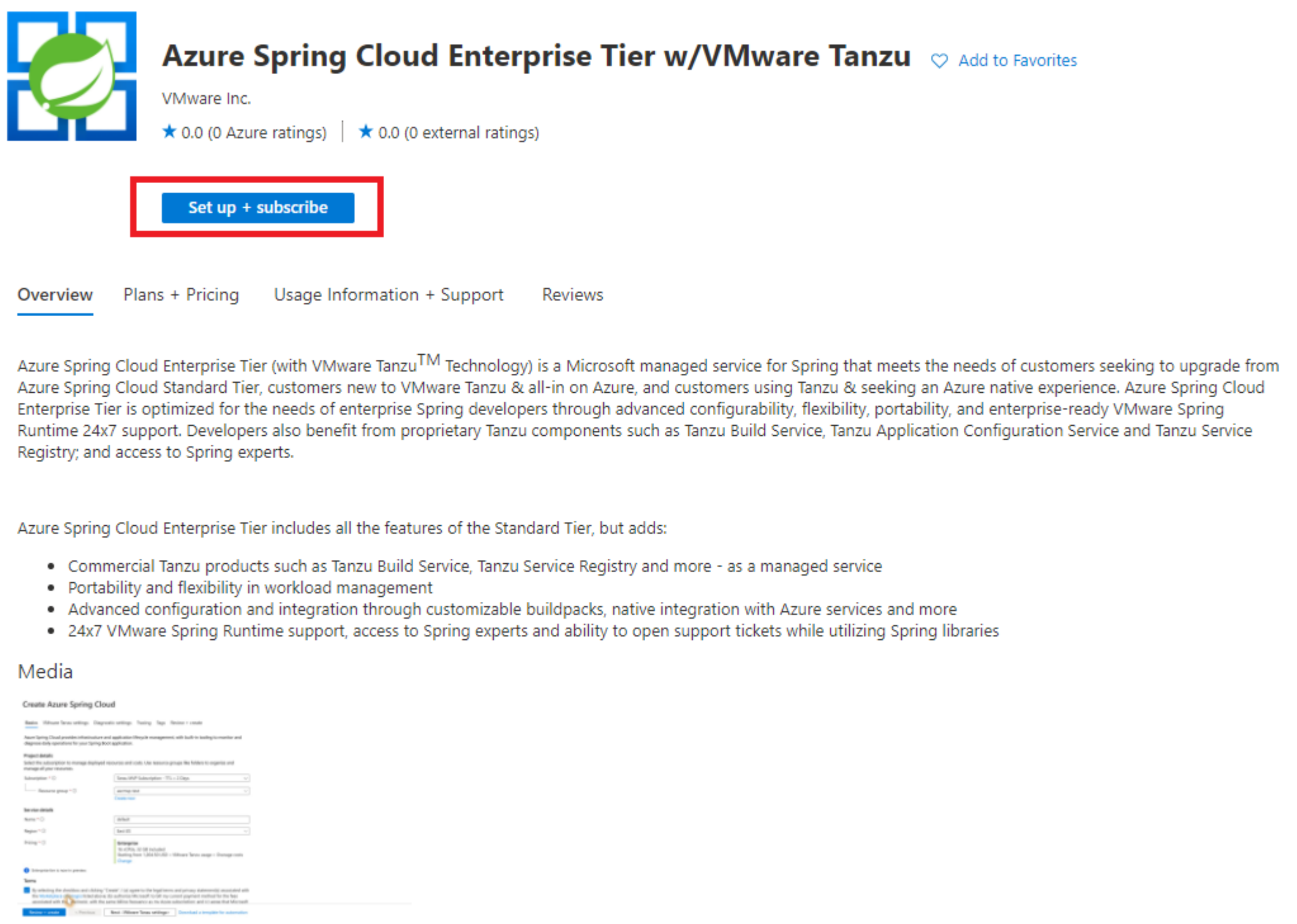
Azure Spring Cloud Enterprise Tier w/VMware Tanzu

VMware Inc.

Software plan		Description	Monthly price
Free (Private)	Private	Azure Spring Cloud Enterprise Tier (with VMware Tanzu Technology) is a Microsoft managed service for Spring that offers enterprise-grade support, configurability, flexibility, and portability for enterprise developers. The Free (Private) plan offers an exclusive first look at the Tier - free of charge.	$0.00 per month

> If you see "No plans are available for market '<Location>'", that means none of your Azure subscription can purchase the SaaS offer. See "No plans are available for market '<Location>'" for more details.

Click "Set up + subscribe", it will redirect you to the Enterprise tier creation page.



Azure Spring Cloud Enterprise Tier w/VMware Tanzu

VMware Inc.

★ 0.0 (0 Azure ratings) | ★ 0.0 (0 external ratings)

Azure Spring Cloud Enterprise Tier (with VMware Tanzu^{TM} Technology) is a Microsoft managed service for Spring that meets the needs of customers seeking to upgrade from Azure Spring Cloud Standard Tier, customers new to VMware Tanzu & all-in on Azure, and customers using Tanzu & seeking an Azure native experience. Azure Spring Cloud Enterprise Tier is optimized for the needs of enterprise Spring developers through advanced configurability, flexibility, portability, and enterprise-ready VMware Spring Runtime 24x7 support. Developers also benefit from proprietary Tanzu components such as Tanzu Build Service, Tanzu Application Configuration Service and Tanzu Service Registry; and access to Spring experts.

Azure Spring Cloud Enterprise Tier includes all the features of the Standard Tier, but adds:

- Commercial Tanzu products such as Tanzu Build Service, Tanzu Service Registry and more - as a managed service
- Portability and flexibility in workload management
- Advanced configuration and integration through customizable buildpacks, native integration with Azure services and more
- 24x7 VMware Spring Runtime support, access to Spring experts and ability to open support tickets while utilizing Spring libraries

Media

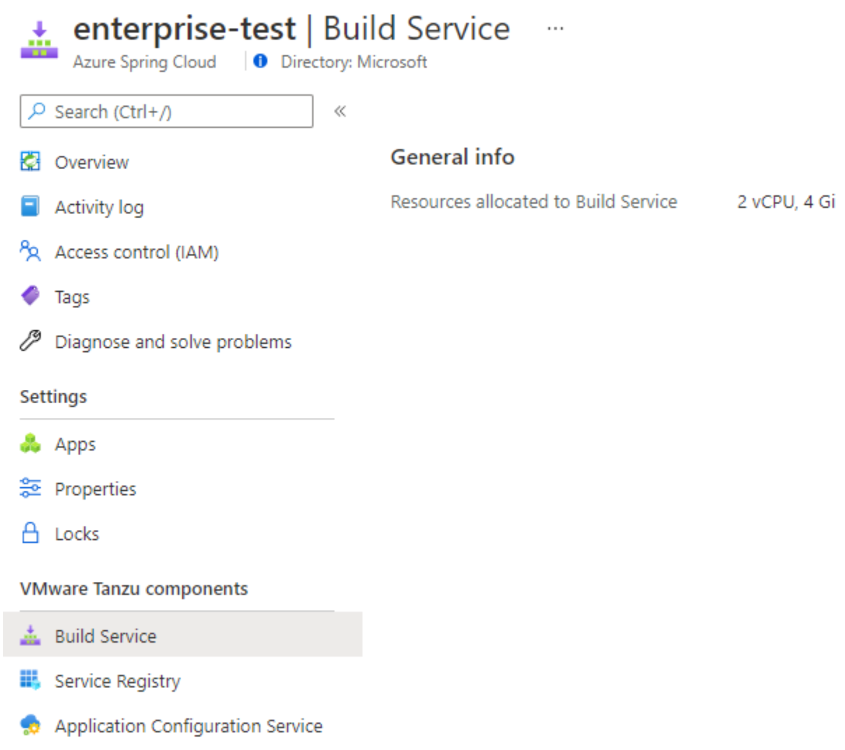
Build Service

Overview

In Azure Spring Cloud, the existing Standard tier already supports compiling user source code into OCI images through kpack - a Kubernetes (K8s) implementation of Cloud Native Buildpacks (CNB) provided by VMware. In Enterprise tier, more functionalities and configurations are exposed. Read more details in the following sections.

Build Agent Pool

Build Service in Enterprise tier will play as the entry point to containerize user application from both source codes and artifacts. There is a dedicated build agent pool that reserves compute resources for a given number of concurrent build tasks. So that there will not be resource contention with your running apps. For now, 2 vCPU and 4 Gi memory are allocated to the build agent pool. In the coming releases after Private Preview, you will be able to configure the amount of the resources allocated to build agent pool to accommodate number of concurrent build tasks you want.



General info

Resources allocated to Build Service 2 vCPU, 4 Gi

Tanzu Buildpacks

On top of the open-source Paketo Buildpacks provided in Standard tier, a list of proprietary Tanzu Buildpacks are available by default in Enterprise tier. Tanzu Buildpacks make it easier to integrate with other software like New Relic etc.. They are configured as optional and will only run with proper configuration. More details are explained in Buildpacks Bindings section.
Below is the complete list of all proprietary Tanzu Buildpacks.

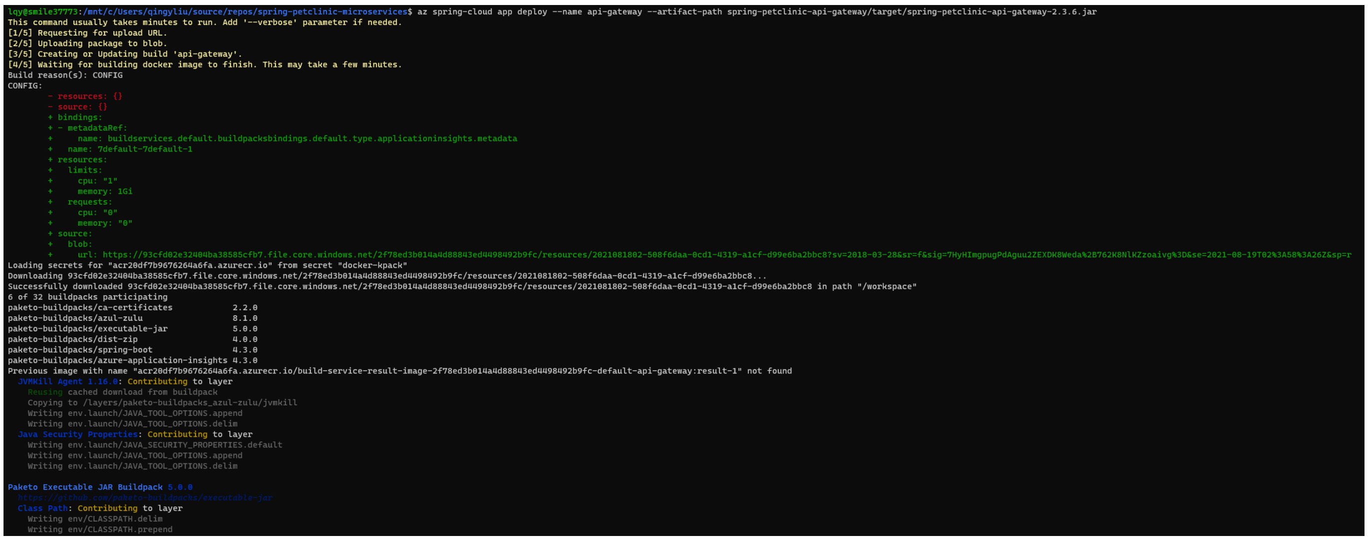
- tanzu-buildpacks/apache-skywalking
- tanzu-buildpacks/appdynamics
- tanzu-buildpacks/aspectj
- tanzu-buildpacks/checkmarx
- tanzu-buildpacks/contrast-security
- tanzu-buildpacks/dynatrace

- tanzu-buildpacks/elastic-apm
- tanzu-buildpacks/icu
- tanzu-buildpacks/jacoco
- tanzu-buildpacks/jprofiler
- tanzu-buildpacks/jrebel
- tanzu-buildpacks/new-relic
- tanzu-buildpacks/node-engine
- tanzu-buildpacks/overops
- tanzu-buildpacks/snyk
- tanzu-buildpacks/synopsys
- tanzu-buildpacks/yourkit

You can find details about each of them in https://docs.pivotal.io/tanzu-buildpacks/partner-integrations/partner-integration-buildpacks.html

Real-time Build Logs

A build task will be triggered when an app is deployed from Azure CLI command. Build logs are streamed out in real-time as part of the CLI command output. Each build task consists of the following phases: `prepare`, `detect`, `restore`, `analyze`, `build`, `export` and `completion`.
Here is a screenshot of sample build logs.



```
lqy@smile37773:/mnt/c/Users/qingyliu/source/repos/spring-petclinic-microservices$ az spring-cloud app deploy --name api-gateway --artifact-path spring-petclinic-api-gateway/target/spring-petclinic-api-gateway-2.3.6.jar
This command usually takes minutes to run. Add '--verbose' parameter if needed.
[1/5] Requesting for upload URL.
[2/5] Uploading package to blob.
[3/5] Creating or Updating build 'api-gateway'.
[4/5] Waiting for building docker image to finish. This may take a few minutes.
Build reason(s): CONFIG
CONFIG:
        - resources: {}
        - source: {}
        + bindings:
        + - metadataRef:
        +     name: buildservices.default.buildpacksbindings.default.type.applicationinsights.metadata
        +   name: 7default-7default-1
        + resources:
        +   limits:
        +     cpu: "1"
        +     memory: 1Gi
        +   requests:
        +     cpu: "0"
        +     memory: "0"
        + source:
        +   blob:
        +     url: https://93cfd02e32404ba38585cfb7.file.core.windows.net/2f78ed3b014a4d88843ed4498492b9fc/resources/2021081802-508f6daa-0cd1-4319-a1cf-d99e6ba2bbc8?sv=2018-03-28&sr=f&sig=7HyHImgpugPdAguu2ZEXDK8Weda%2B762K8NlKZzoaivg%3D&se=2021-08-19T02%3A58%3A26Z&sp=r
Loading secrets for "acr20df7b9676264a6fa.azurecr.io" from secret "docker-kpack"
Downloading 93cfd02e32404ba38585cfb7.file.core.windows.net/2f78ed3b014a4d88843ed4498492b9fc/resources/2021081802-508f6daa-0cd1-4319-a1cf-d99e6ba2bbc8...
Successfully downloaded 93cfd02e32404ba38585cfb7.file.core.windows.net/2f78ed3b014a4d88843ed4498492b9fc/resources/2021081802-508f6daa-0cd1-4319-a1cf-d99e6ba2bbc8 in path "/workspace"
6 of 32 buildpacks participating
paketo-buildpacks/ca-certificates           2.2.0
paketo-buildpacks/azul-zulu                 8.1.0
paketo-buildpacks/executable-jar            5.0.0
paketo-buildpacks/dist-zip                  4.0.0
paketo-buildpacks/spring-boot               4.3.0
paketo-buildpacks/azure-application-insights 4.3.0
Previous image with name "acr20df7b9676264a6fa.azurecr.io/build-service-result-image-2f78ed3b014a4d88843ed4498492b9fc-default-api-gateway:result-1" not found
  JVMKill Agent 1.16.0: Contributing to layer
    Reusing cached download from buildpack
    Copying to /layers/paketo-buildpacks_azul-zulu/jvmkill
    Writing env.launch/JAVA_TOOL_OPTIONS.append
    Writing env.launch/JAVA_TOOL_OPTIONS.delim
  Java Security Properties: Contributing to layer
    Writing env.launch/JAVA_SECURITY_PROPERTIES.default
    Writing env.launch/JAVA_TOOL_OPTIONS.append
    Writing env.launch/JAVA_TOOL_OPTIONS.delim

Paketo Executable JAR Buildpack 5.0.0
  https://github.com/paketo-buildpacks/executable-jar
  Class Path: Contributing to layer
    Writing env/CLASSPATH.delim
    Writing env/CLASSPATH.prepend
```

Buildpacks Bindings

Kpack Images can be configured with Service Bindings as described in the Cloud Native Buildpacks Bindings specification. We leverage this Binding to integrate with Tanzu Partner Buildpacks. For example, we use Binding to integrate Azure Application Insights by paketo-buildpacks/azure-application-insights.

For integration with Azure Application Insights, see Monitor Apps with Application Insights.

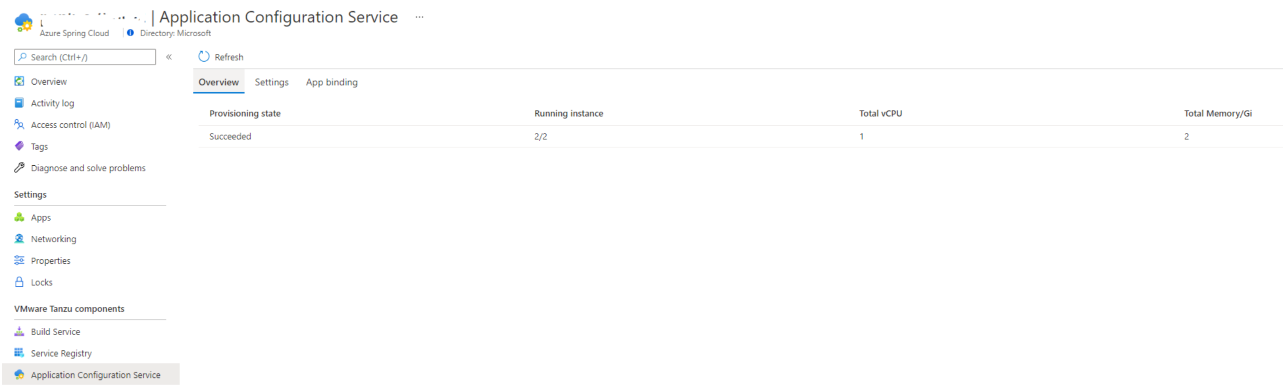
Application Configuration Service

Overview

Application Configuration Service is one of the proprietary VMware Tanzu components. It enables the management of Kubernetes-native ConfigMap resources that are populated from properties defined in one or more Git repositories. With Application Configuration Service, you have a central place to manage external properties for applications across all environments.

Prerequisites

- An already provisioned Azure Spring Cloud Enterprise tier service instance with Application Configuration Service enabled. See get-started

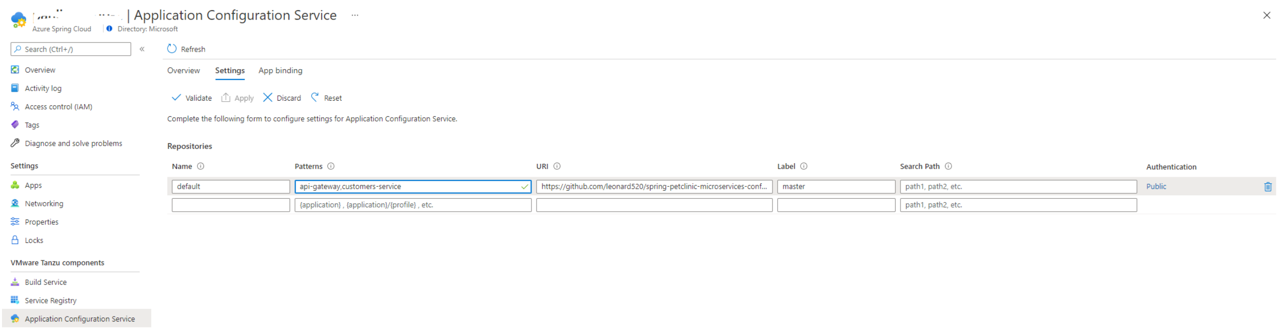


Provisioning state	Running instance	Total vCPU	Total Memory/Gi
Succeeded	2/2	1	2

Manage Application Configuration Service settings

Application Configuration Service supports Azure DevOps, GitHub, GitLab, and Bitbucket for storing you config file.

1. Open **Settings** blade, and add a new entry with below information in **Repositories** section.



Complete the following form to configure settings for Application Configuration Service.

Repositories

Name	Patterns	URI	Label	Search Path	Authentication
default	api-gateway,customers-service	https://github.com/leonard520/spring-petclinic-microservices-conf...	master	path1, path2, etc.	Public
	{application} , {application}/{profile} , etc.			path1, path2, etc.	

Terminology

Property	Required	Explaination
Name	Yes	Unique name to label each git repository.
Patterns	Yes	Use patterns to search in Git repositories. For each pattern, use format like {application} or {application}/{profile} instead of {application}-{profile}.yml, and separate them by comma. See Patterns for more detail explaination.
URI	Yes	Git URI (e.g. https://github.com/Azure-Samples/piggymetrics-config, git@github.com:Azure-Samples/piggymetrics-config)
Label	Yes	branch name to search in the Git repository.

Property	Required	Explaination
Search path	No	Optional search paths separated by comma to search subdirectories of the Git repository.

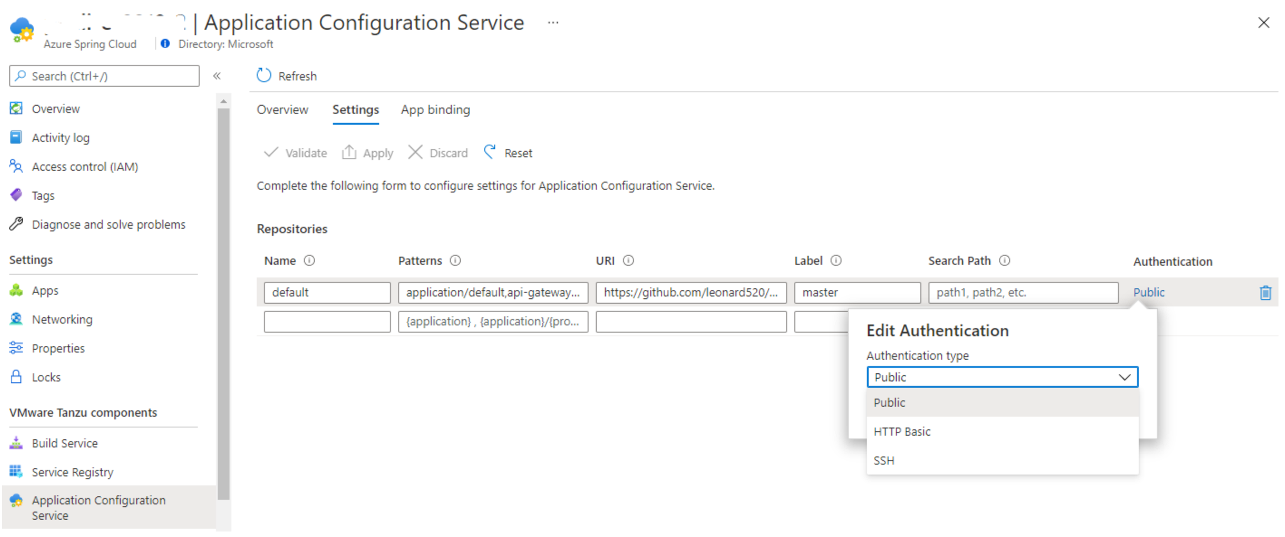
Pattern

Configuration will be pulled from git backends using what it defined in a pattern which is combined of {application}/{profile}.

- {application} - The name of an application for which the configuration is being retrieved. If “application”, then this is considered the default application and includes configuration shared across multiple applications. Any other value specifies a specific application and will include properties for both the specified application as well as shared properties for the default application.
- {profile} - Optional. The name of a profile for which properties may be retrieved. If “default” or empty, then this includes properties that are shared across any all profiles. If any non-default value, then include properties for the specified profile as well as properties for the default profile.

Authentication

Application Configuration Service supports three repository types. Please check the detail below.


[illegible] | Application Configuration Service

Azure Spring Cloud | Directory: Microsoft

Complete the following form to configure settings for Application Configuration Service.

Repositories

Name	Patterns	URI	Label	Search Path	Authentication
default	application/default,api-gateway...	https://github.com/leonard520/...	master	path1, path2, etc.	Public
	{application} , {application}/{pro...				

Edit Authentication

Authentication type: Public

- Public repository

You don't need extra Authentication configuration when using a public repository but just select **Public** in the **Authentication** form.

- Private repository with basic authentication

All configurable properties used to set up private Git repository with basic authentication are listed below.

Property	Required	Explaination
username	Yes	The username used to access the repository.

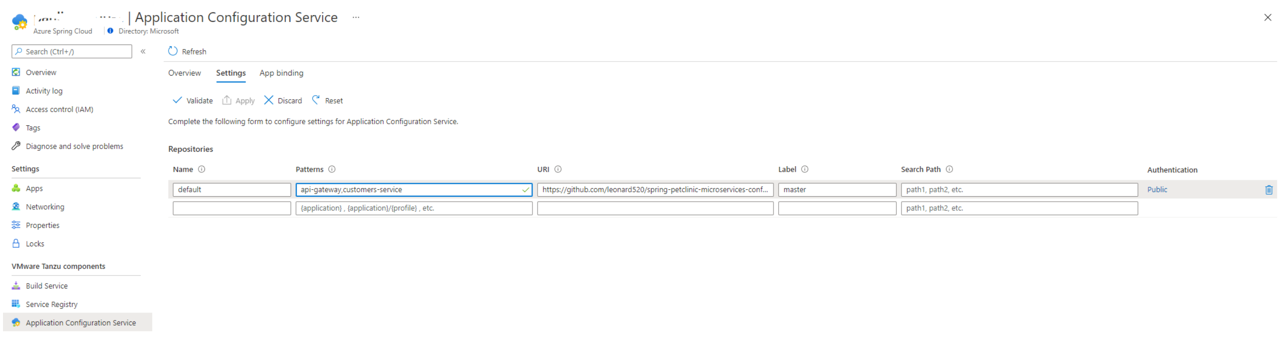
Property	Required	Explaination
password	Yes	The password used to access the repository.

- Private repository with SSH authentication

All configurable properties used to set up private Git repository with SSH are listed in the following table:

Property	Required	Explaination
Private key	Yes	The private key that identifies the Git user. Passphrase-encrypted private keys are not supported.
Host key	No	The host key of the Git server. If you have connected to the server via git on the command line, this is in your .ssh/known_hosts. Do not include the algorithm prefix; this is specified in `Host key algorithm`.
Host key algorithm	No	The algorithm of hostKey: one of "ssh-dss", "ssh-rsa", "ecdsa-sha2-nistp256", "ecdsa-sha2-nistp384", and "ecdsa-sha2-nistp521". (Required if supplying `Host key`).
Strict host key checking	No	Whether or not the backend should be ignored it encounters an error when using the provided `Host key`. (Optional.) Valid values are true and false. Default is true.

2. Click **Validate** button to validate access to the target URI. After validation is completed successfully, click **Apply** button to update configuration settings.



Complete the following form to configure settings for Application Configuration Service.

Repositories

Name	Patterns	URI	Label	Search Path	Authentication
default	api-gateway,customers-service	https://github.com/leonard520/spring-petclinic-microservices-conf...	master	path1, path2, etc.	Public
	{application} , {application}/{profile} , etc.			path1, path2, etc.	

Use Application Configuration Service with Apps

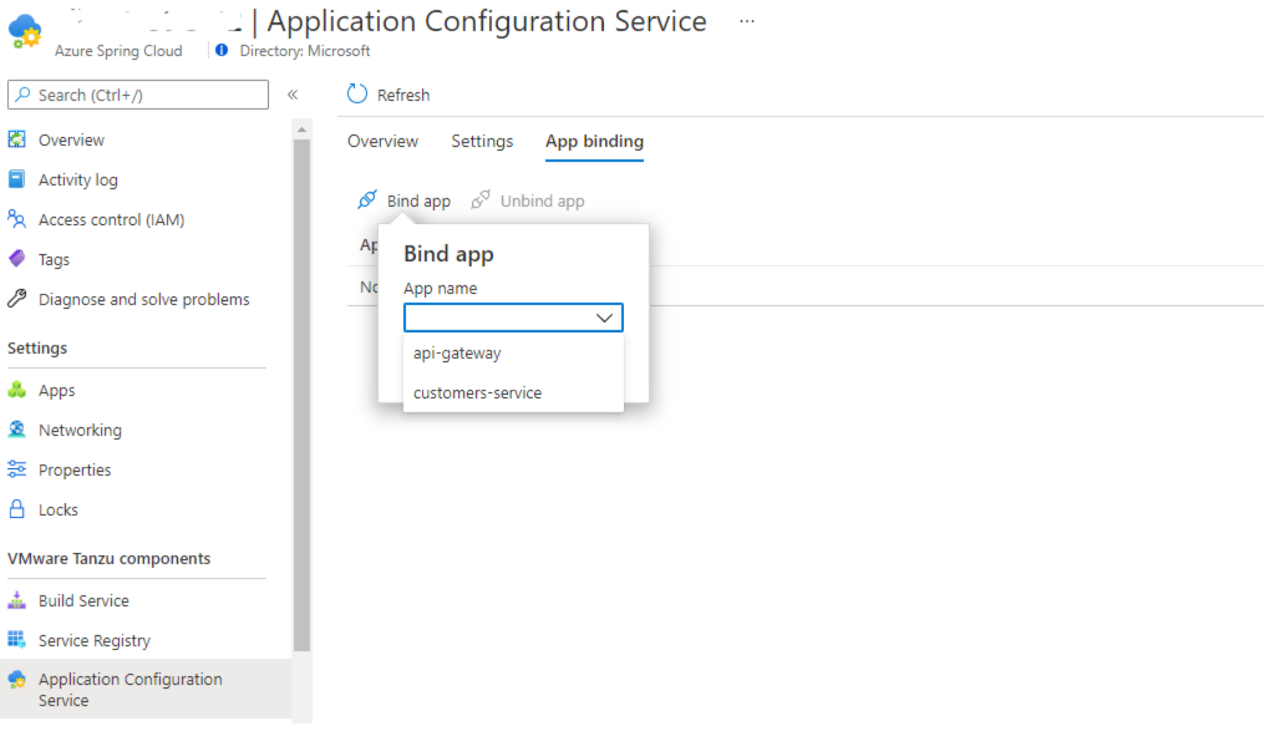
Restriction

There are some restrictions when you use Application Configuration Service with a Git back end.

⚠ Please DON'T include `spring-cloud-starter-config` in the app's pom.xml
~~<dependency>~~
~~<groupId>org.springframework.cloud</groupId>~~
~~<artifactId>spring-cloud-starter-config</artifactId>~~
~~</dependency>~~

You have to bind the app to Application Configuration Service to claim that the app will use the centralized configurations. After that, you will need to configure which pattern to be used by the app.

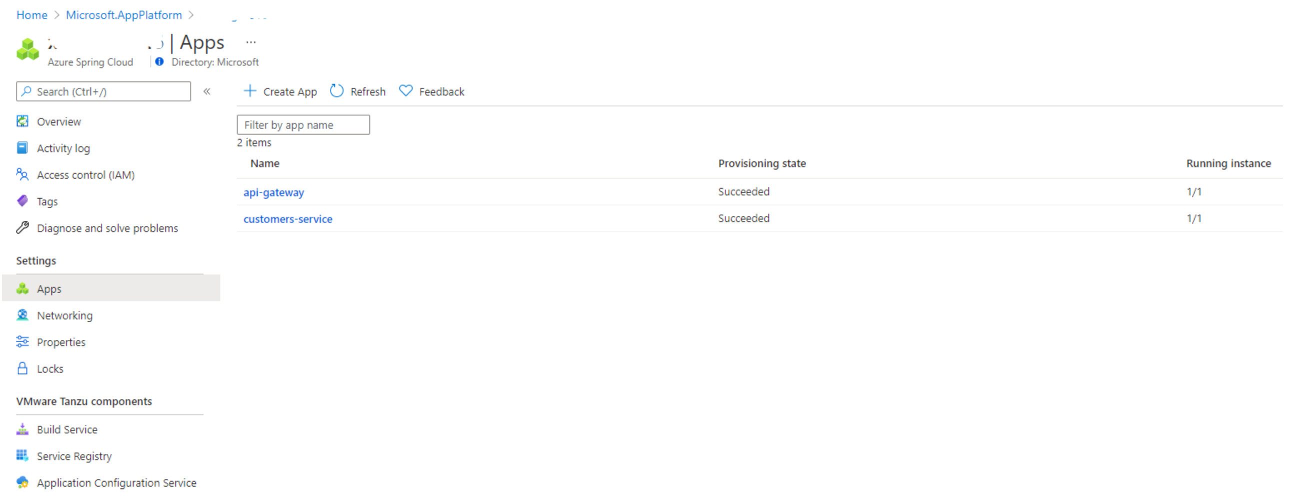
1. Open **App binding** tab to bind/unbind app to the Application Configuration Service.
2. Click **Bind app** button and choose one app in the dropdown. Click **Apply** button to bind.



> Note
>
> When you change the bind/unbind status, you have to restart or redeploy the app to take effective.

3. Go to **Apps** blade and choose the pattern(s) to be used by the apps.

a. Open **Apps** blade to list all the apps.

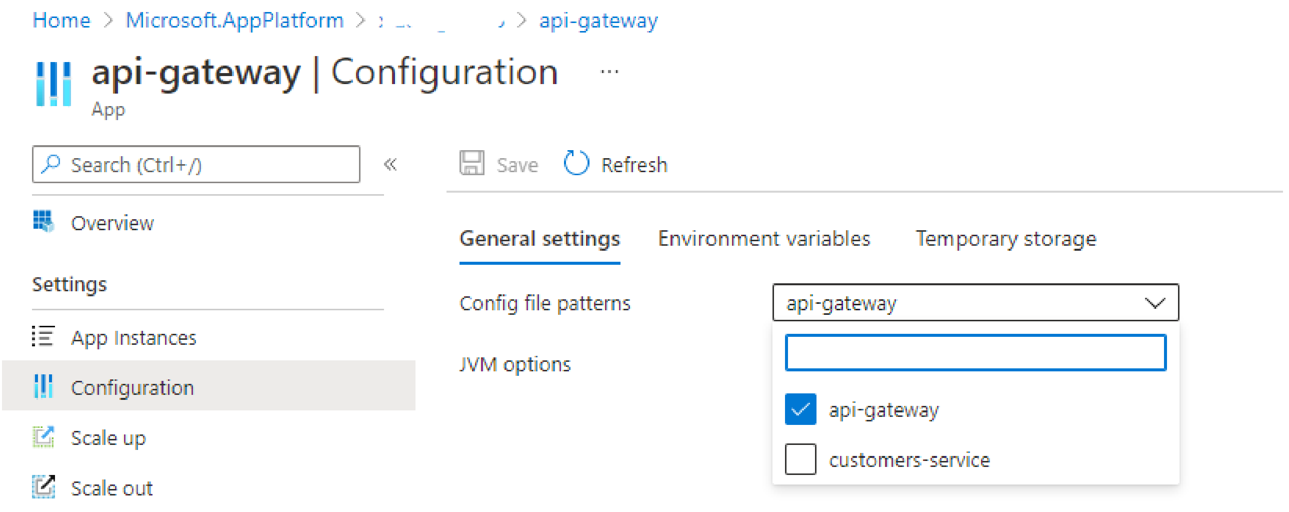


2 items

Name	Provisioning state	Running instance
api-gateway	Succeeded	1/1
customers-service	Succeeded	1/1

b. Click target app to configure patterns.

c. Open **Configuration** menu and **General settings** tab. Under **Config file patterns** property, choose one or more patterns from the dropdown list. Click **Save** button to save the changes.



Config file patterns: api-gateway

[✓] api-gateway
[ ] customers-service

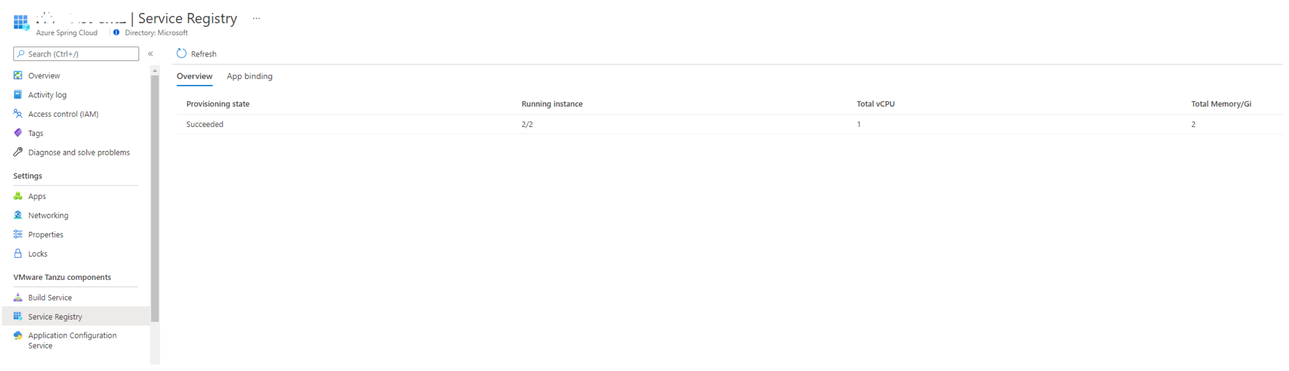
Service Registry

Overview

Service Registry is one of the proprietary VMware Tanzu components. It provides your apps with an implementation of the Service Discovery pattern, one of the key tenets of a microservice-based architecture. Trying to hand-configure each client of a service or adopt some form of access convention can be difficult and prove to be brittle in production. Instead, your apps can use the Service Registry to dynamically discover and call registered services.

Prerequisites

- An already provisioned Azure Spring Cloud Enterprise tier service instance with Service Registry enabled. See get-started



[redacted] | Service Registry ···
Azure Spring Cloud | Directory: Microsoft

Provisioning state	Running instance	Total vCPU	Total Memory/Gi
Succeeded	2/2	1	2

Use Service Registry with Apps

Before your application can manage service registration and discovery using Service Registry, several dependencies must be included in the application's pom.xml file.

```
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
        </dependency>
```

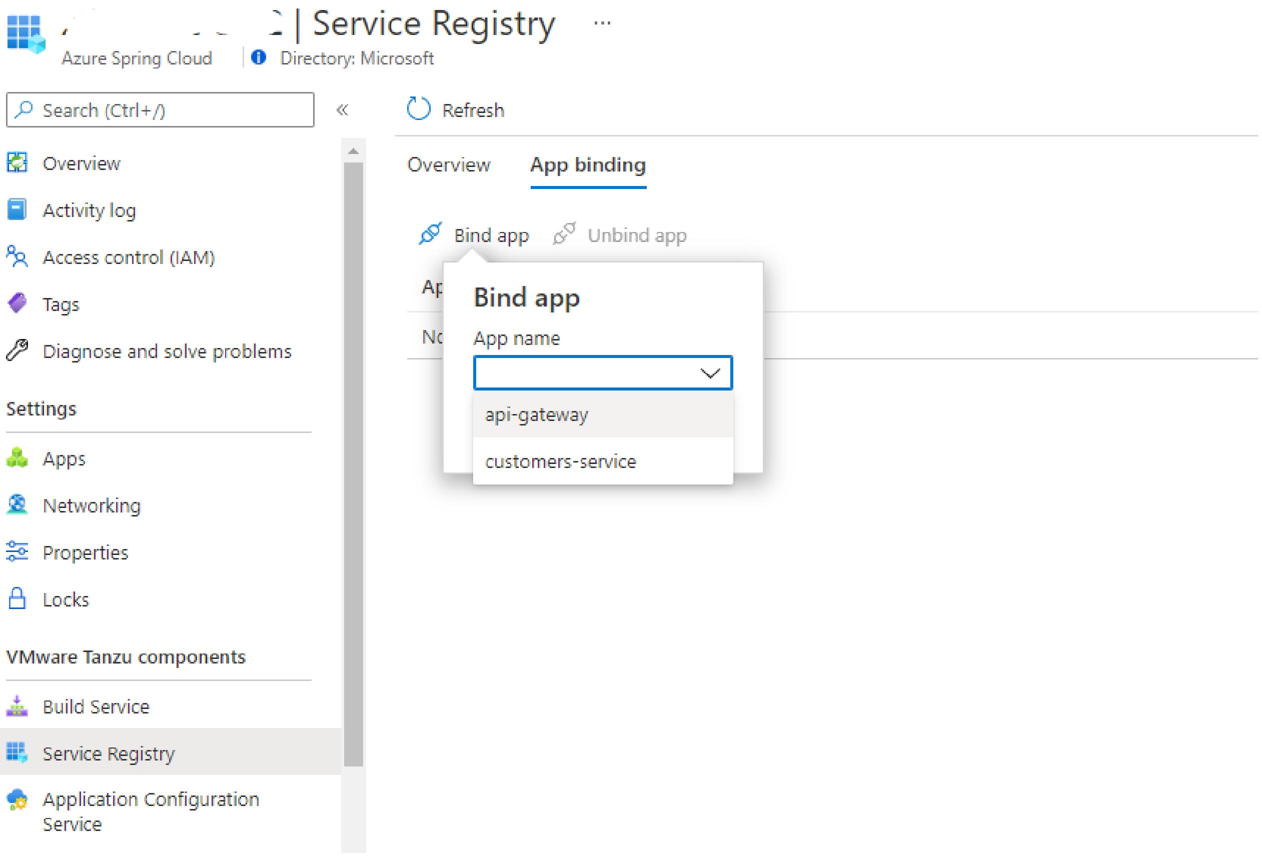
Finally, add an annotation to the top level class of your application

```
    @SpringBootApplication
    @EnableEurekaClient
    public class DemoApplication {

     public static void main(String[] args) {
            SpringApplication.run(DemoApplication.class, args);
        }
    }
```

1. Open **App binding** tab to bind/unbind app to the Service Registry.

2. Click **Bind app** button and choose one app in the dropdown. Click **Apply** button to bind.

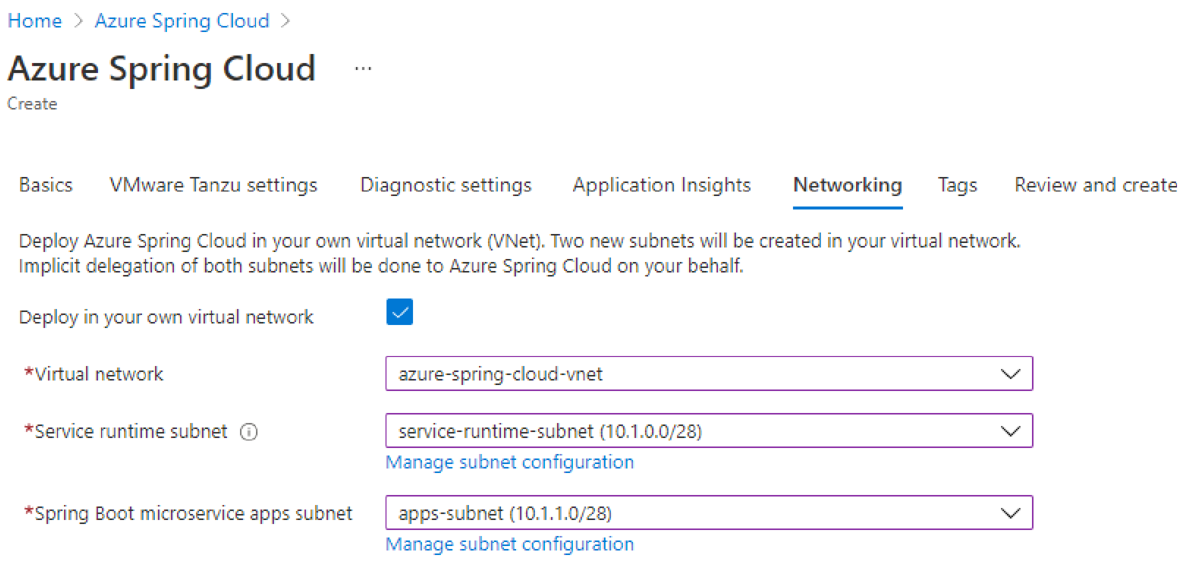


Deploy Azure Spring Cloud in a virtual network

This tutorial explains how to deploy an Azure Spring Cloud instance in your virtual network. Please refer to Deploy Azure Spring Cloud in a virtual network for details.

Note: Please go to this Azure portal to deploy an Azure Spring Cloud Enterprise instance.

The **Networking** tab of Azure Spring Cloud **Create** page is as below:


Home > Azure Spring Cloud >

Azure Spring Cloud ...

Create

Basics | VMware Tanzu settings | Diagnostic settings | Application Insights | Networking | Tags | Review and create

Deploy Azure Spring Cloud in your own virtual network (VNet). Two new subnets will be created in your virtual network. Implicit delegation of both subnets will be done to Azure Spring Cloud on your behalf.

Deploy in your own virtual network	✓
*Virtual network	azure-spring-cloud-vnet
*Service runtime subnet	service-runtime-subnet (10.1.0.0/28)
	Manage subnet configuration
*Spring Boot microservice apps subnet	apps-subnet (10.1.1.0/28)
	Manage subnet configuration

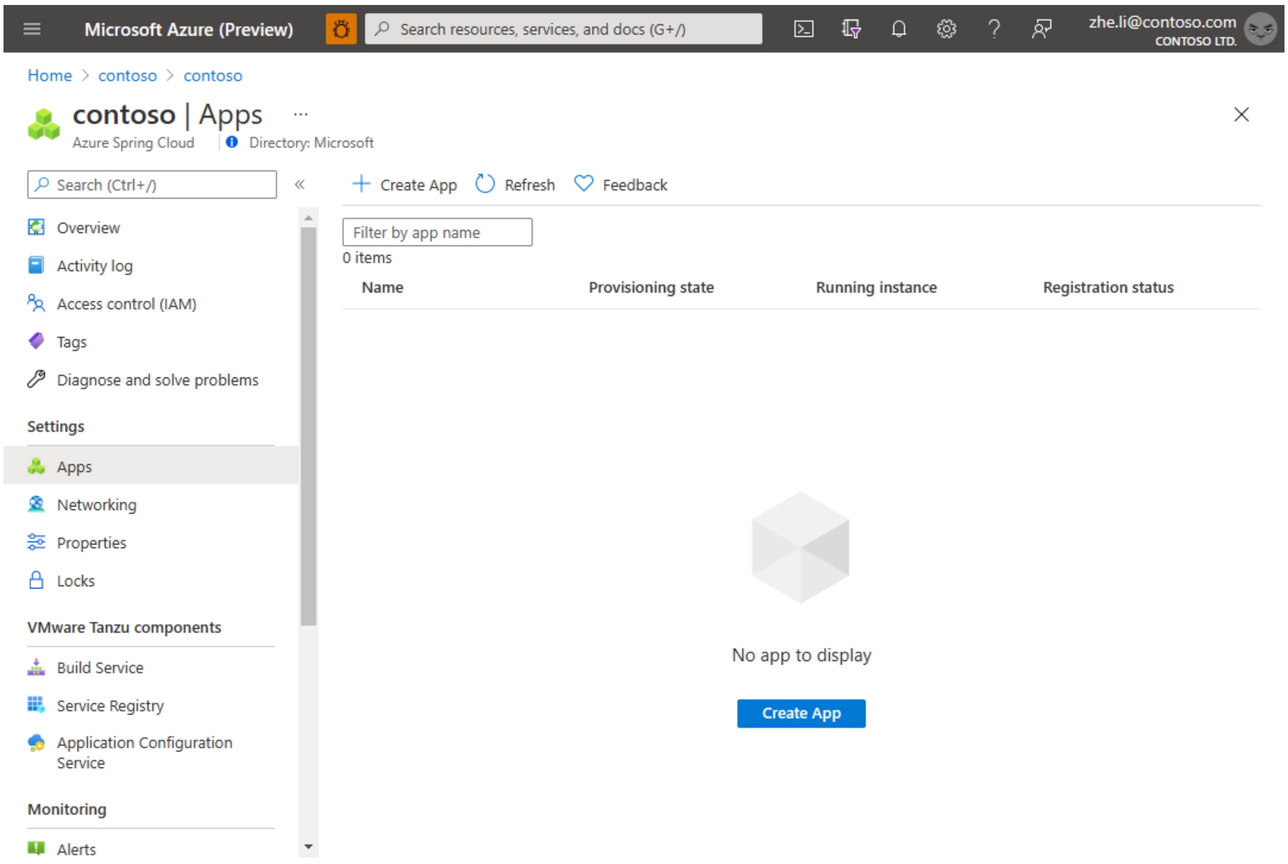
App and Deployment management

Portal

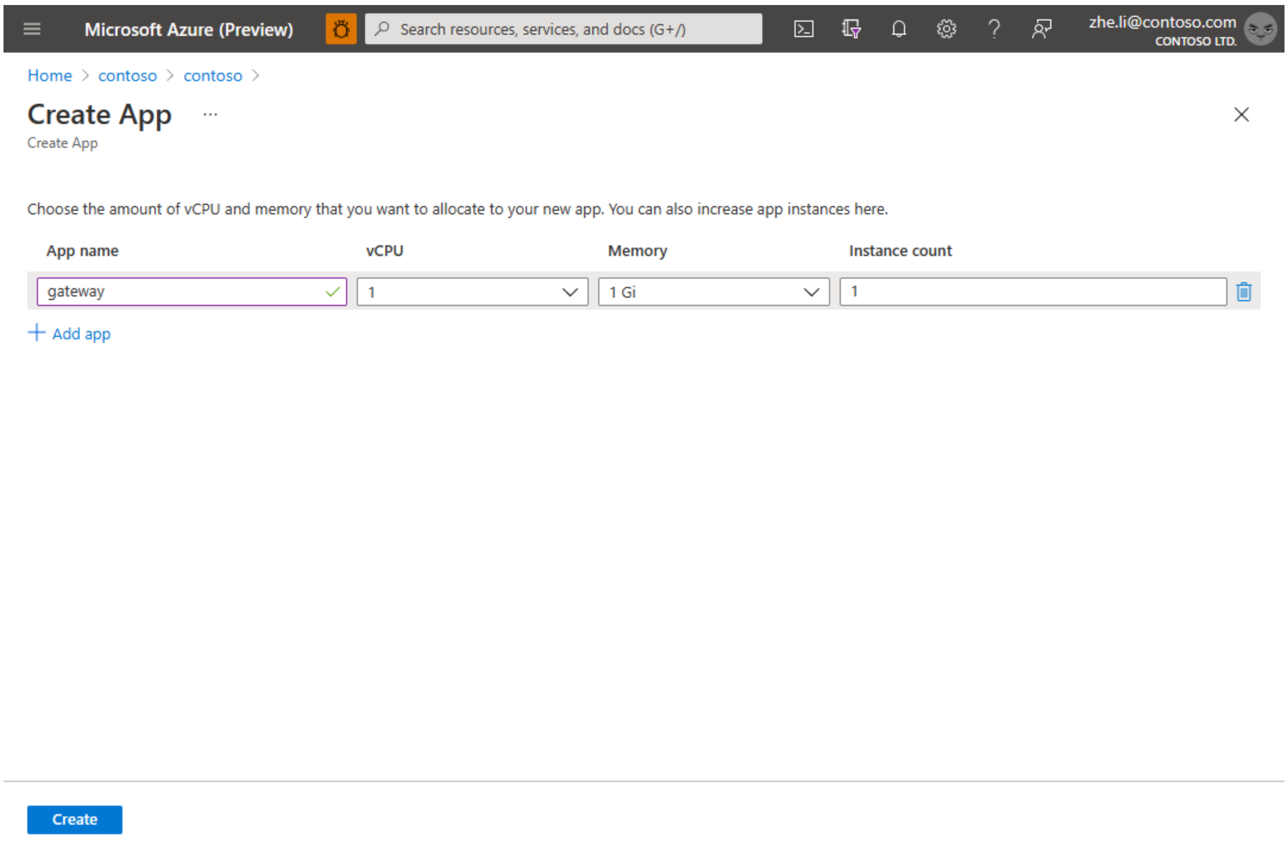
Create Apps

1. Open the Azure portal.

2. Go to the Azure Spring Cloud **Apps** page. Click **Create App** button to create apps.



3. Add a new entry in **Create App** blade with app name, vCPU, memory and instance count. You can add more entries by clicking **+ Add app** link to create more apps at one time. Then click **Create** button.


Microsoft Azure (Preview)
Search resources, services, and docs (G+/)
zhe.li@contoso.com
CONTOSO LTD.
Home > contoso > contoso >
Create App
Create App

Choose the amount of vCPU and memory that you want to allocate to your new app. You can also increase app instances here.

App name	vCPU	Memory	Instance count
gateway	1	1 Gi	1

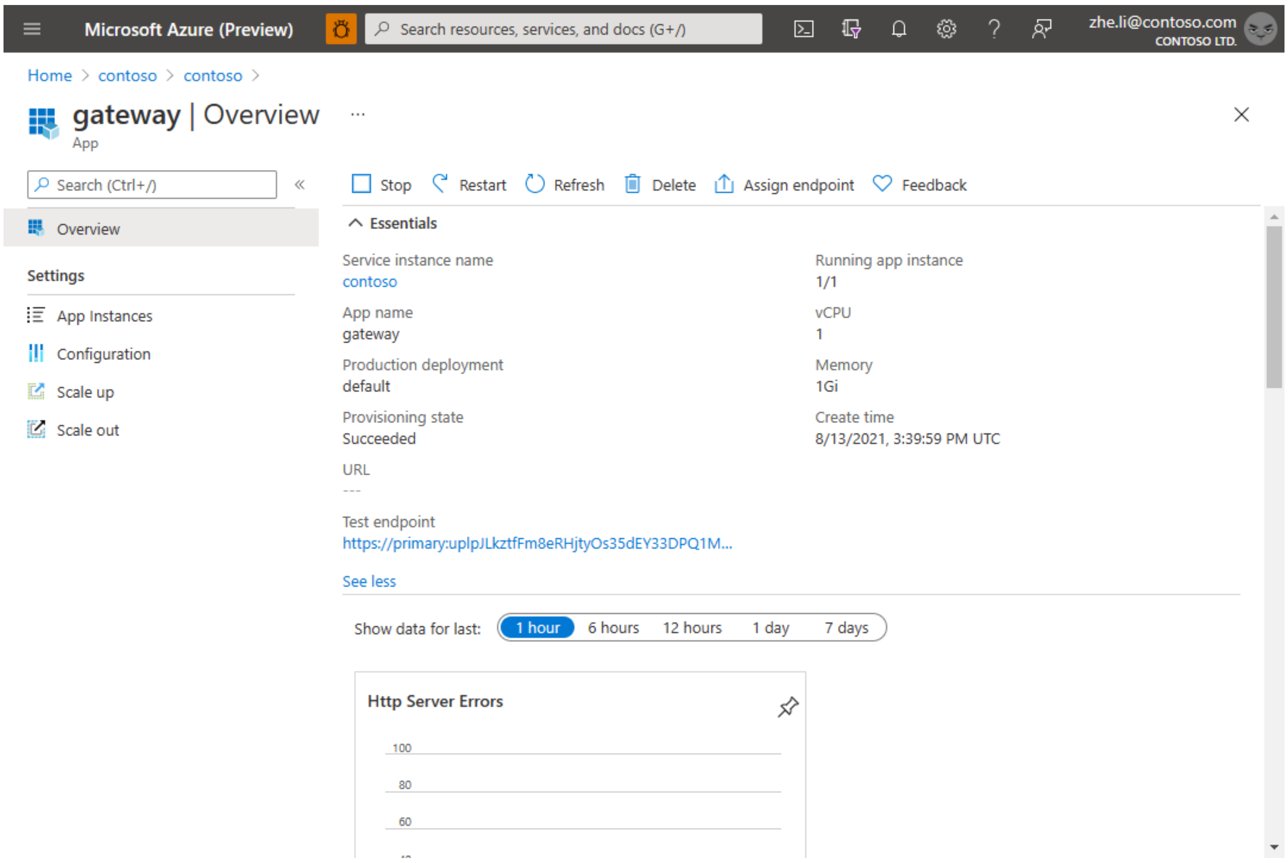
App overview and instance list

1. Once the app is created, open the app overview page by clicking the app name in **Apps** blade.

In the app overview page, you can find a set of information of the app, including resource basic information, provisioning state, public endpoint and test endpoint.

You can also make some operations on the app in the overview page, such as stop, start and restart the app, assign public endpoint to the app.

Some general metrics are listed at bottom of the page, you can learn the app health state from the charts.



Service instance name
contoso

App name
gateway

Production deployment
default

Provisioning state
Succeeded

URL

Test endpoint
https://primary:uplpJLkztfFm8eRHjtyOs35dEY33DPQ1M...

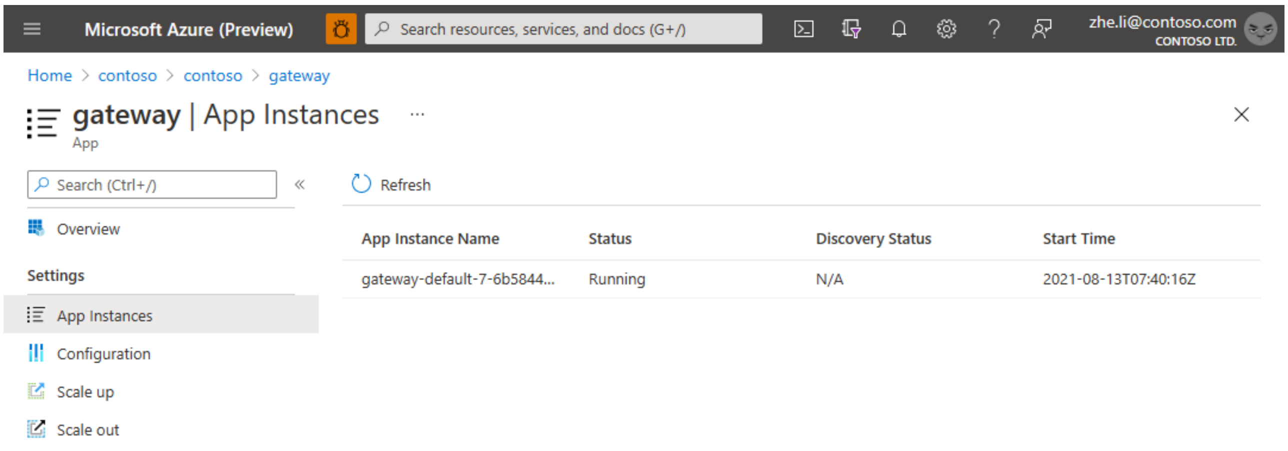
Running app instance
1/1

vCPU
1

Memory
1Gi

Create time
8/13/2021, 3:39:59 PM UTC

2. To see all instance of the app, go to **App Instance** page.

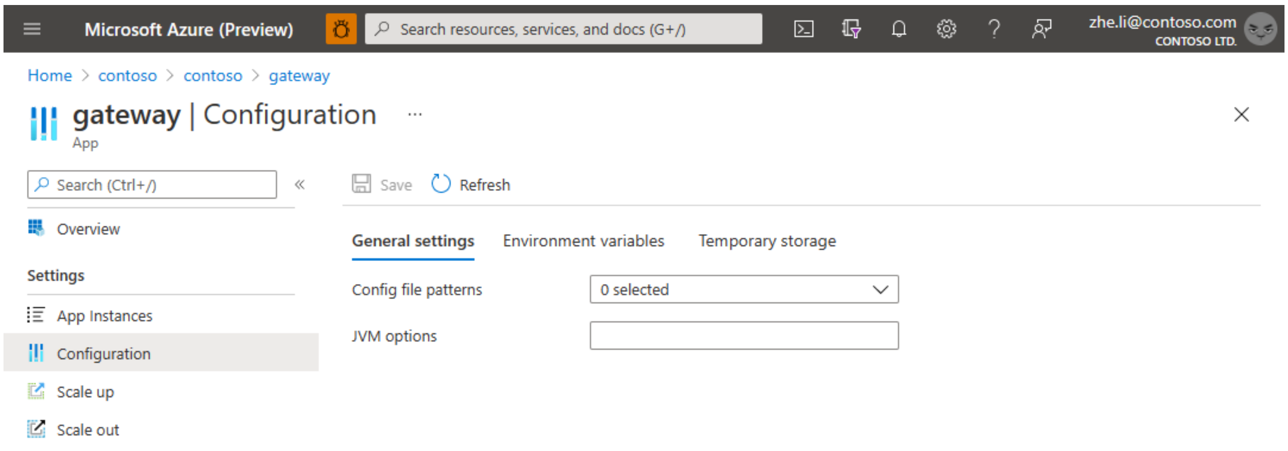


App Instance Name	Status	Discovery Status	Start Time
gateway-default-7-6b5844...	Running	N/A	2021-08-13T07:40:16Z

App configuration

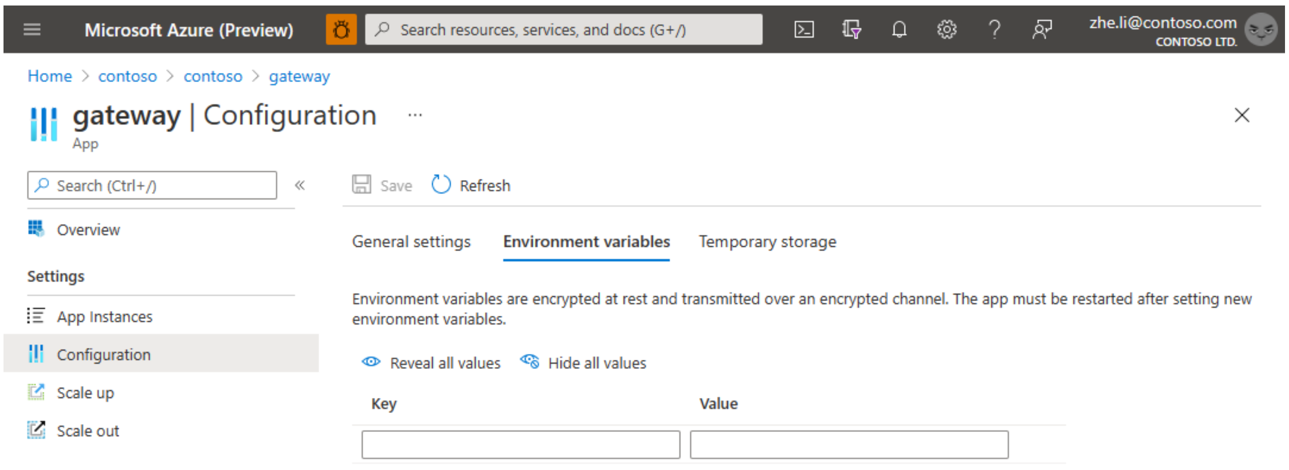
Configuration blade enables you to update config file patterns, JVM options (if your app is built with Java), environment variables. You can also view temporary storage information in this page.

1. General settings



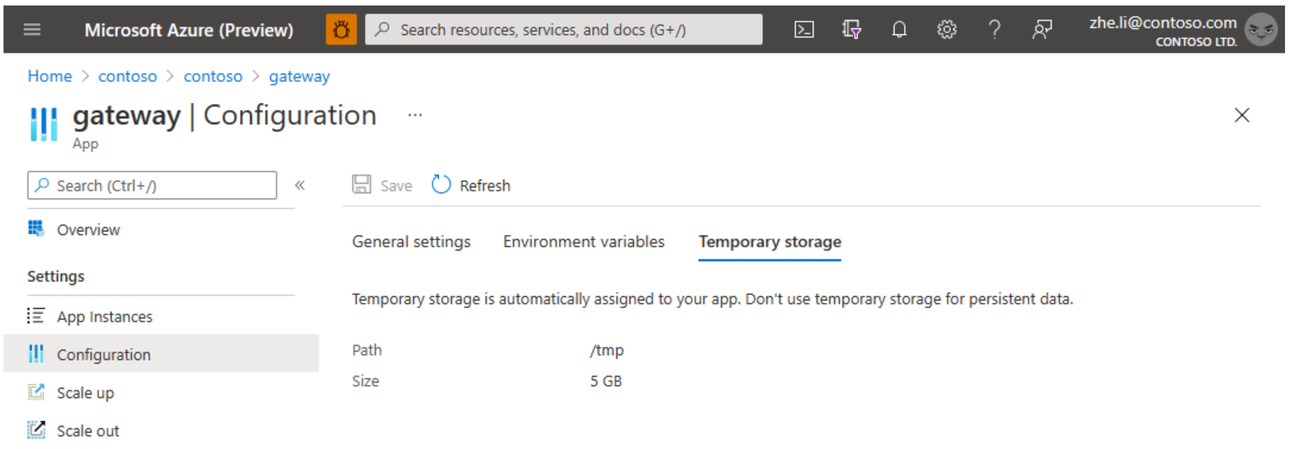
Config file patterns: 0 selected
JVM options:

2. Environment variables



Environment variables are encrypted at rest and transmitted over an encrypted channel. The app must be restarted after setting new environment variables.

3. Temporary storage

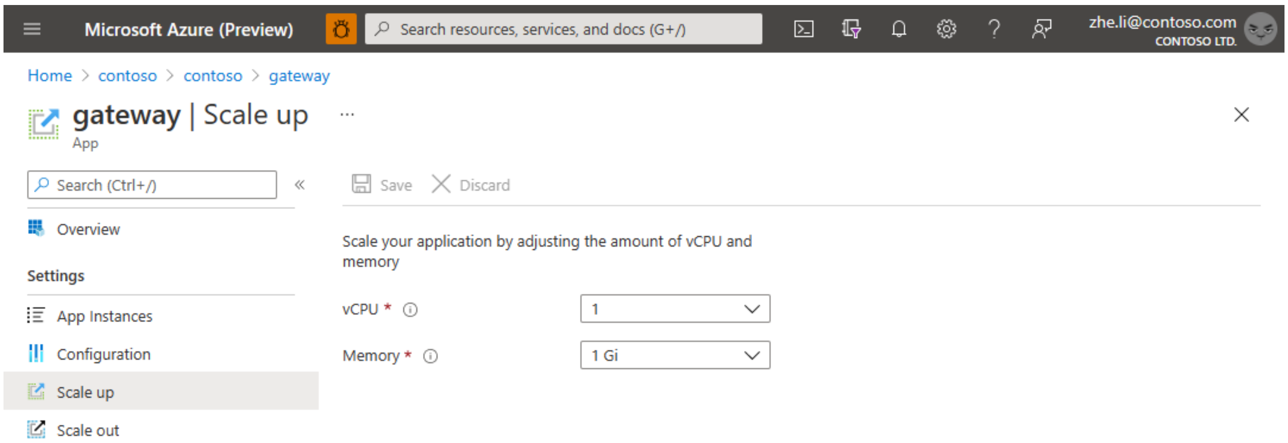


Temporary storage is automatically assigned to your app. Don't use temporary storage for persistent data.

Path	/tmp
Size	5 GB

Scale up and scale out

1. To scale up/down your app by updating vCPU and memory, open **Scale up** blade.

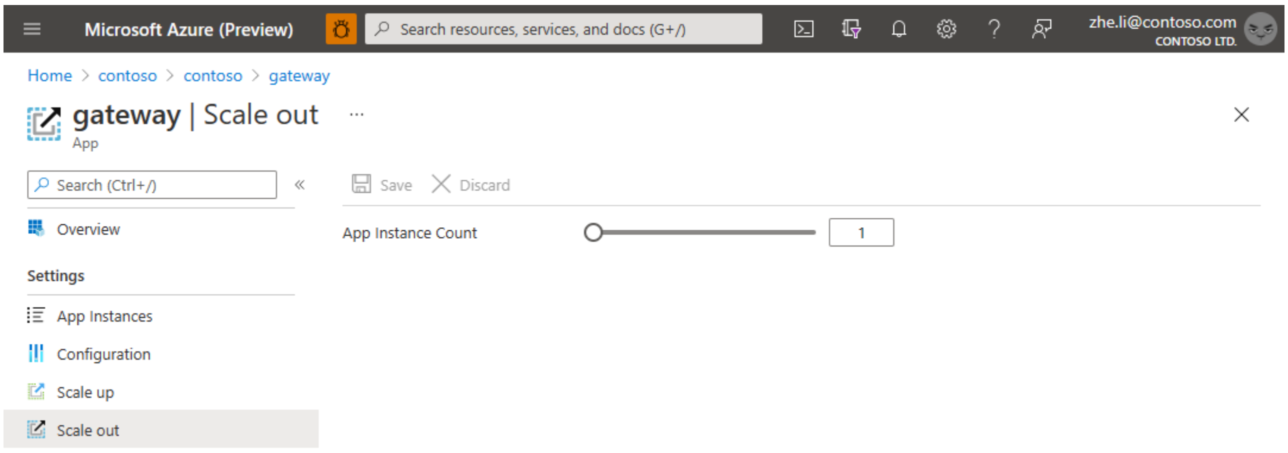


Scale your application by adjusting the amount of vCPU and memory

vCPU * : 1

Memory * : 1 Gi

2. Open **Scale out** blade to change instance count of the app.



App Instance Count: 1

Azure CLI

Monitor Apps with Application Insights

This article explains how to monitor apps and microservices by using the Application Insights Java agent in Azure Spring Cloud.

With this feature you can:

- Search tracing data with different filters.
- View dependency map of microservices.
- Check request performance.
- Monitor real-time live metrics.
- Check request failures.
- Check app metrics.
- Check app log.

Application Insights provide many observable perspectives, including:

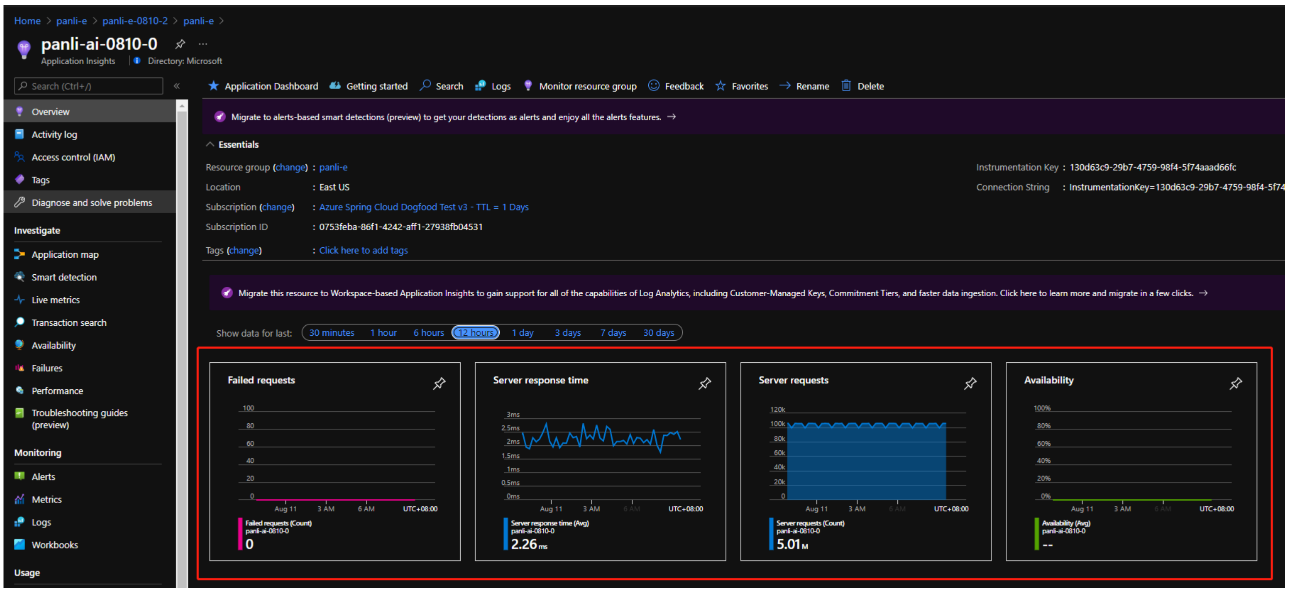
- Application map
- Performance
- Failures
- Metrics
- Live Metrics
- Log
- Availability

Using the Application Insights feature

When the **Application Insights** feature is enabled, you can:

Go to the **Overview** page of Application Insights.

- You can see overview of all apps.



Migrate to alerts-based smart detections (preview) to get your detections as alerts and enjoy all the alerts features. →

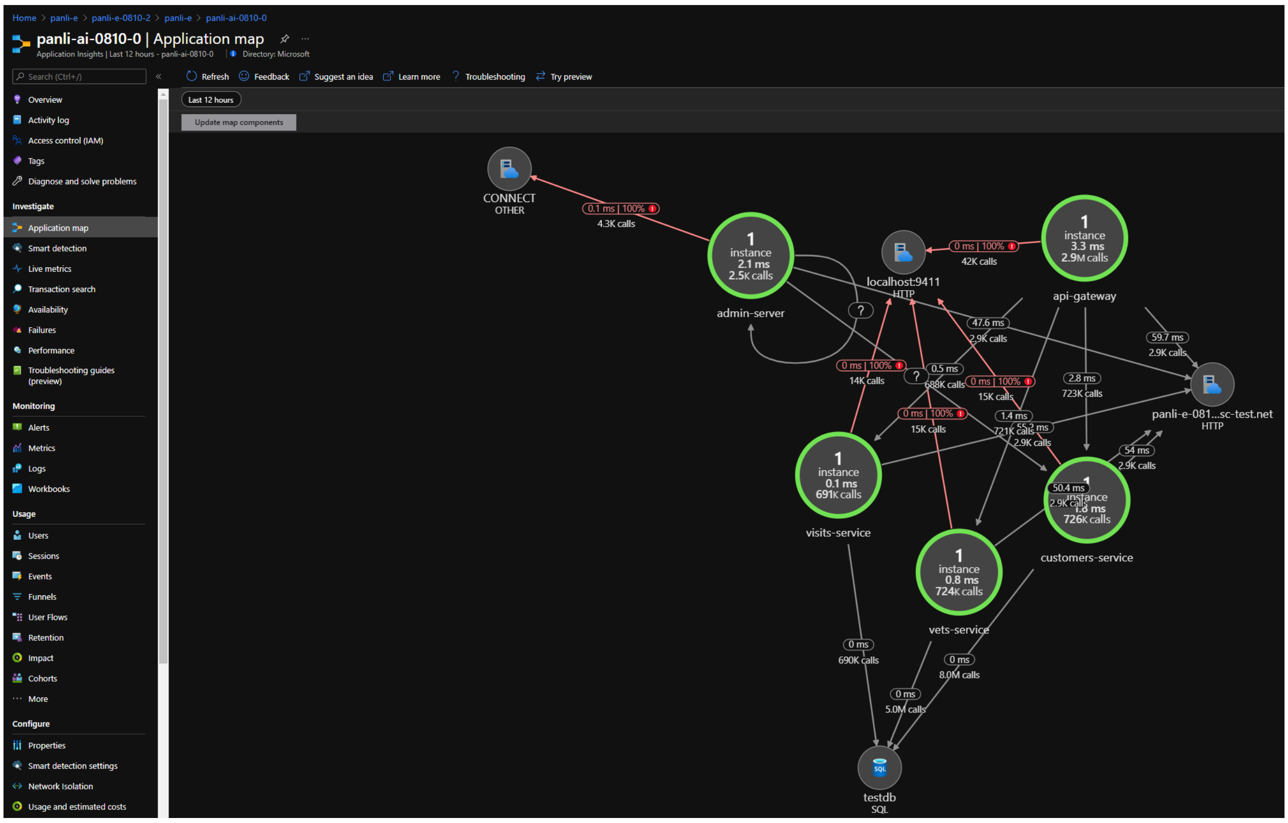
Resource group (change) : panli-e
Location : East US
Subscription (change) : Azure Spring Cloud Dogfood Test v3 - TTL = 1 Days
Subscription ID : 0753feba-86f1-4242-aff1-27938fb04531
Tags (change) : Click here to add tags
Instrumentation Key : 130d63c9-29b7-4759-98f4-5f74aaad66fc
Connection String : InstrumentationKey=130d63c9-29b7-4759-98f4-5f74

Migrate this resource to Workspace-based Application Insights to gain support for all of the capabilities of Log Analytics, including Customer-Managed Keys, Commitment Tiers, and faster data ingestion. Click here to learn more and migrate in a few clicks. →

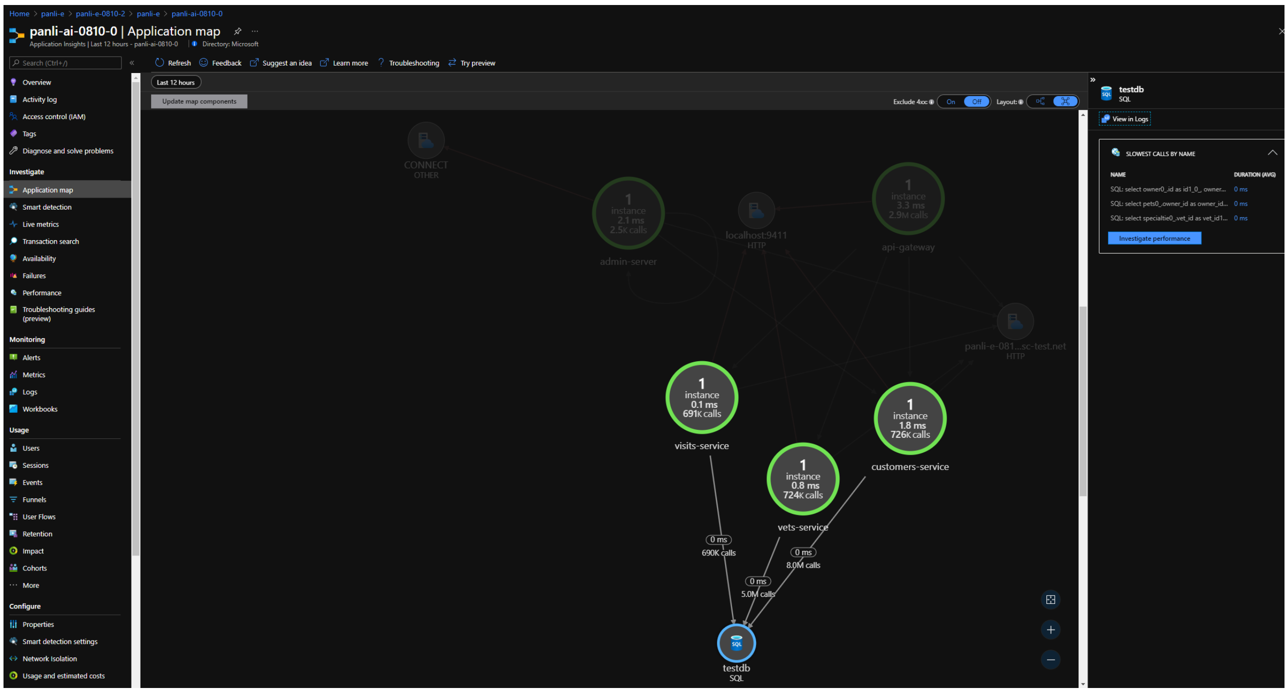
Show data for last: 30 minutes | 1 hour | 6 hours | [✓ 12 hours] | 1 day | 3 days | 7 days | 30 days

Failed requests (Count) panli-ai-0810-0: 0
Server response time (Avg) panli-ai-0810-0: 2.26 ms
Server requests (Count) panli-ai-0810-0: 5.01 M
Availability (Avg) panli-ai-0810-0: --

- Click the **Application Map** to see the status of calls between apps.



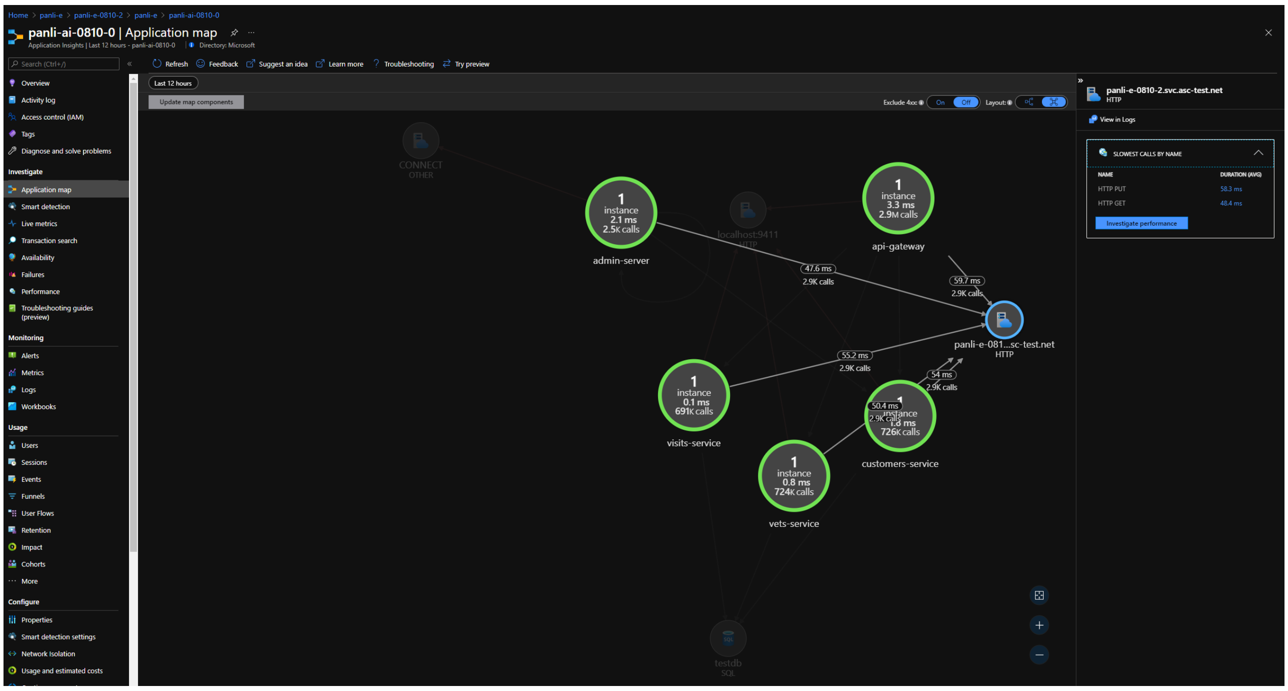
- Click the testdb to see all the apps connecting to database.



SLOWEST CALLS BY NAME

NAME	DURATION (AVG)
SQL: select owner0_.id as id1_0_, owner...	0 ms
SQL: select pets0_.owner_id as owner_id...	0 ms
SQL: select specialtie0_.vet_id as vet_id1...	0 ms

- Click an endpoint to see all apps requesting to the endpoint.

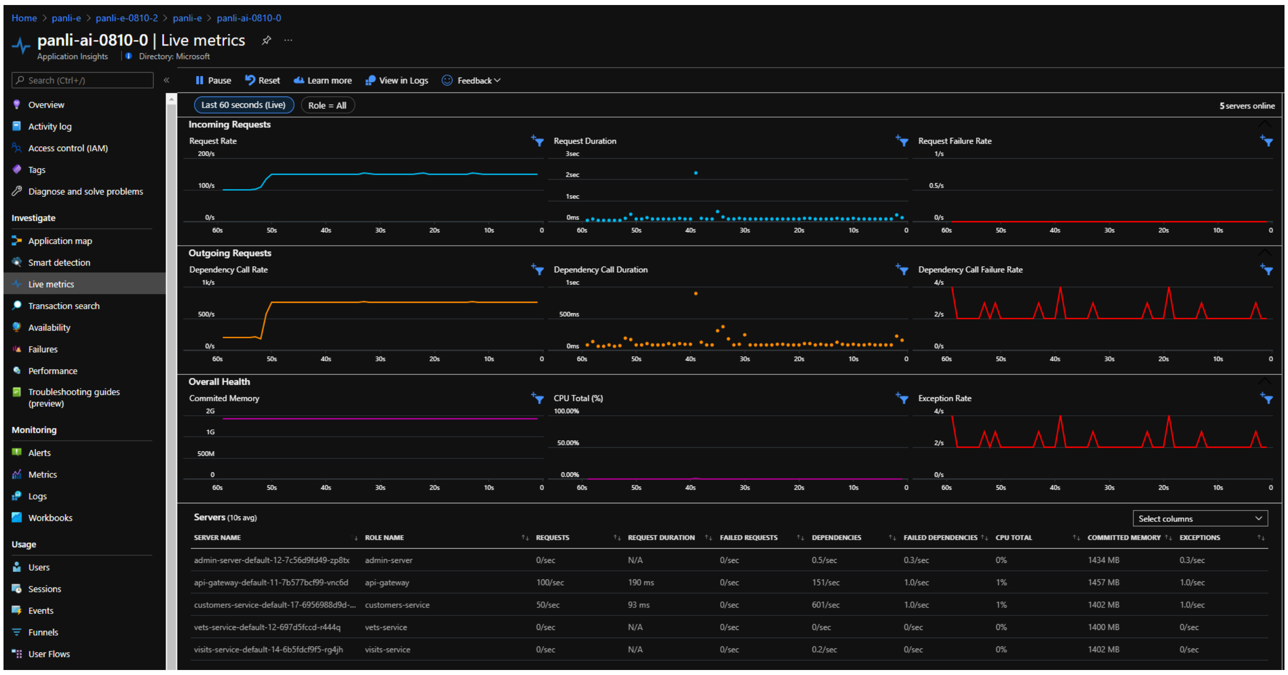


panli-e-0810-2.svc.asc-test.net
HTTP

SLOWEST CALLS BY NAME

NAME	DURATION (AVG)
HTTP PUT	58.3 ms
HTTP GET	48.4 ms

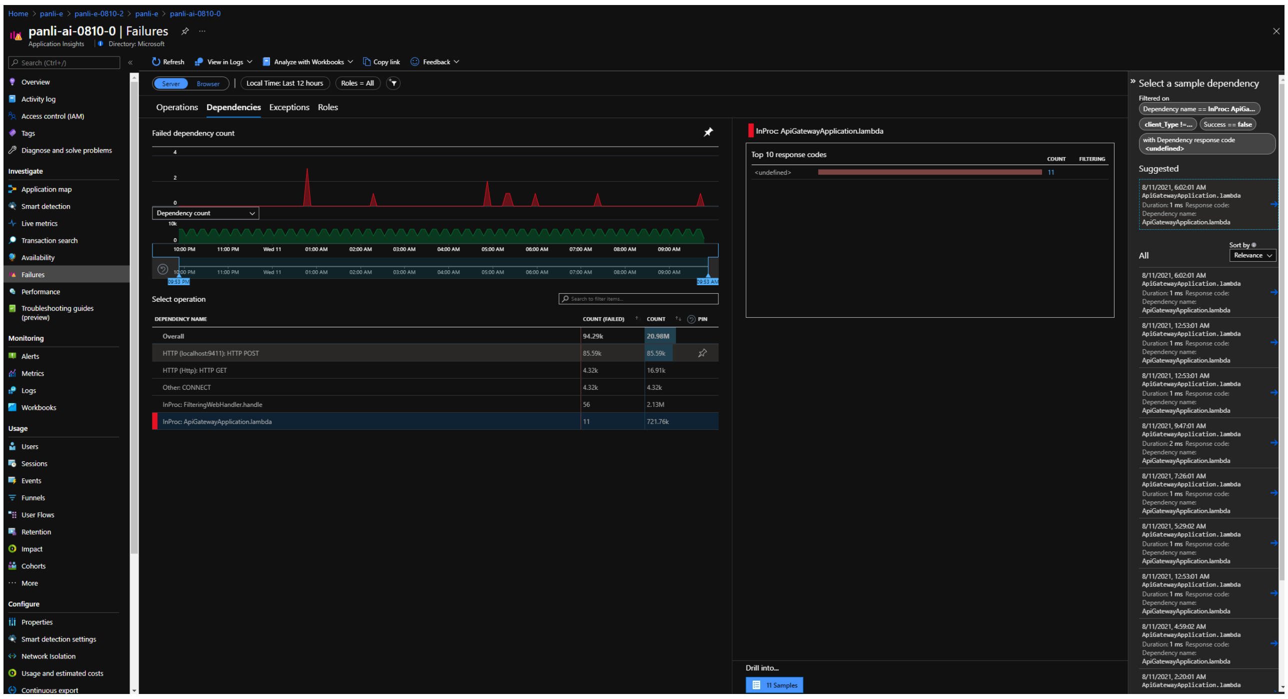
- In the left navigation pane, click **Live Metrics** to see the real time metrics of all apps' in last 60 seconds..



Servers (10s avg)

SERVER NAME	ROLE NAME	REQUESTS	REQUEST DURATION	FAILED REQUESTS	DEPENDENCIES	FAILED DEPENDENCIES	CPU TOTAL	COMMITTED MEMORY	EXCEPTIONS
admin-server-default-12-7c56d9fd49-zp8tx	admin-server	0/sec	N/A	0/sec	0.5/sec	0.3/sec	0%	1434 MB	0.3/sec
api-gateway-default-11-7b577bcf99-vnc6d	api-gateway	100/sec	190 ms	0/sec	151/sec	1.0/sec	1%	1457 MB	1.0/sec
customers-service-default-17-6956988d9d-...	customers-service	50/sec	93 ms	0/sec	601/sec	1.0/sec	1%	1402 MB	1.0/sec
vets-service-default-12-697d5fccd-r444q	vets-service	0/sec	N/A	0/sec	0/sec	0/sec	0%	1400 MB	0/sec
visits-service-default-14-6b5fdcf9f5-rg4jh	visits-service	0/sec	N/A	0/sec	0.2/sec	0/sec	0%	1402 MB	0/sec

- In the left navigation pane, click **Failures** to see if something unexpected of dependencies from your apps.


panli-ai-0810-0 | Failures

Failed dependency count

Select operation

DEPENDENCY NAME	COUNT (FAILED)	COUNT
Overall	94.29k	20.98M
HTTP (localhost:9411): HTTP POST	85.59k	85.59k
HTTP (Http): HTTP GET	4.32k	16.91k
Other: CONNECT	4.32k	4.32k
InProc: FilteringWebHandler.handle	56	2.13M
InProc: ApiGatewayApplication.lambda	11	721.76k

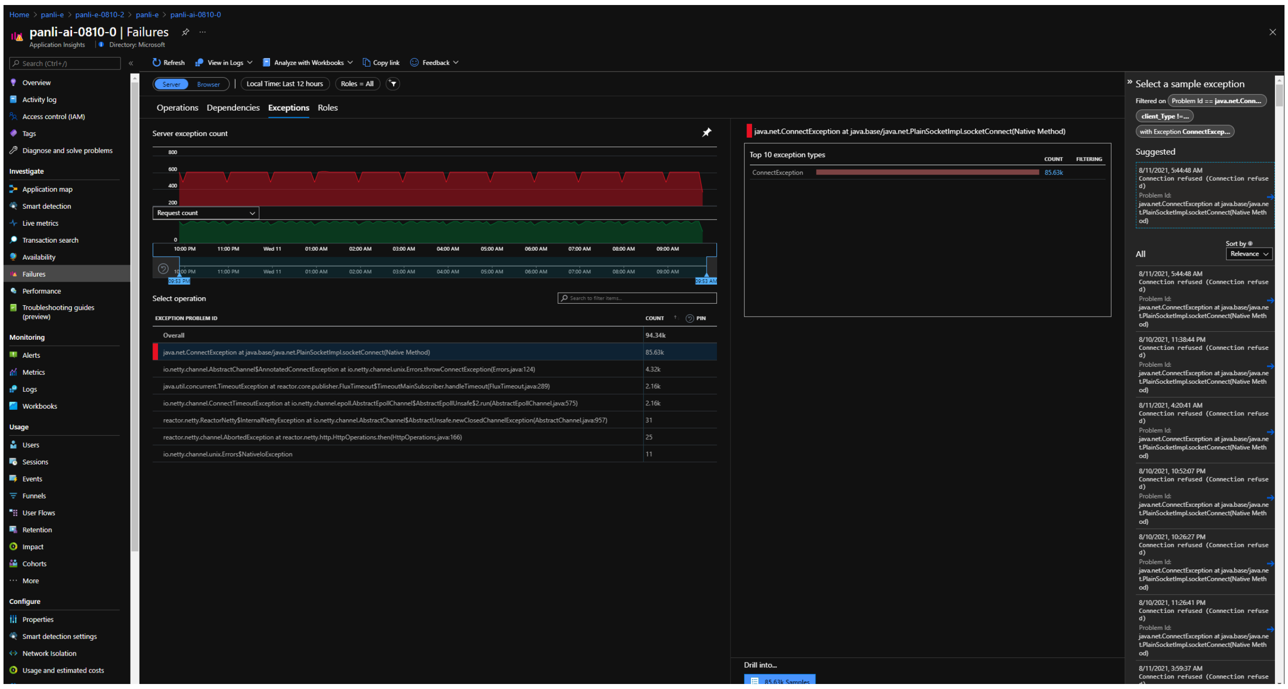
InProc: ApiGatewayApplication.lambda

Top 10 response codes

	COUNT
<undefined>	11

Select a sample dependency

- In the left navigation pane, click **Failures** to see if something unexpected of exceptions from your apps.


panli-ai-0810-0 | Failures

Server exception count

Select operation

EXCEPTION PROBLEM ID	COUNT
Overall	94.34k
java.net.ConnectException at java.base/java.net.PlainSocketImpl.socketConnect(Native Method)	85.63k
io.netty.channel.AbstractChannel$AnnotatedConnectException at io.netty.channel.unix.Errors.throwConnectException(Errors.java:124)	4.32k
java.util.concurrent.TimeoutException at reactor.core.publisher.FluxTimeout$TimeoutMainSubscriber.handleTimeout(FluxTimeout.java:289)	2.16k
io.netty.channel.ConnectTimeoutException at io.netty.channel.epoll.AbstractEpollChannel$AbstractEpollUnsafe$2.run(AbstractEpollChannel.java:575)	2.16k
reactor.netty.ReactorNetty$InternalNettyException at io.netty.channel.AbstractChannel$AbstractUnsafe.newClosedChannelException(AbstractChannel.java:957)	31
reactor.netty.channel.AbortedException at reactor.netty.http.HttpOperations.then(HttpOperations.java:166)	25
io.netty.channel.unix.Errors$NativeIoException	11

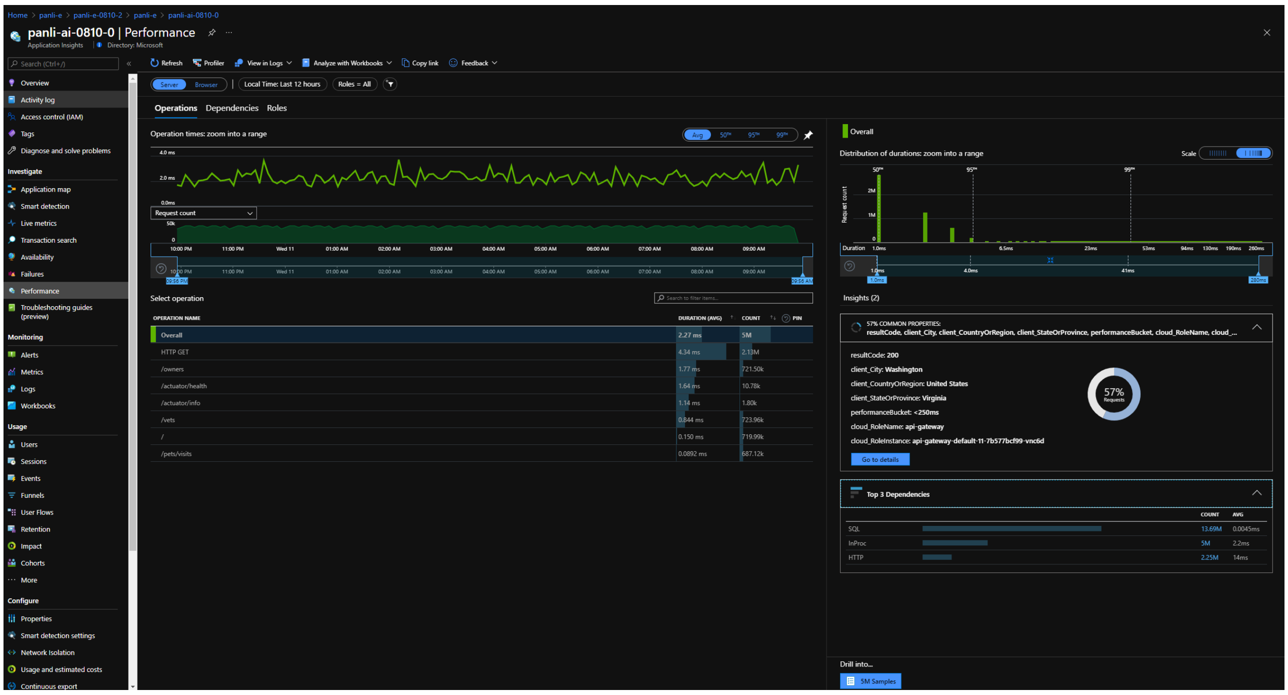
java.net.ConnectException at java.base/java.net.PlainSocketImpl.socketConnect(Native Method)

Top 10 exception types

	COUNT
ConnectException	85.63k

Select a sample exception

- In the left navigation pane, click **Performance** to see the performance data of all apps' operations.



Select operation

OPERATION NAME	DURATION (AVG)	COUNT
Overall	2.27 ms	5M
HTTP GET	4.34 ms	2.13M
/owners	1.77 ms	721.50k
/actuator/health	1.64 ms	10.78k
/actuator/info	1.14 ms	1.80k
/vets	0.844 ms	723.96k
/	0.150 ms	719.99k
/pets/visits	0.0892 ms	687.12k

Insights (2)

57% COMMON PROPERTIES:
resultCode, client_City, client_CountryOrRegion, client_StateOrProvince, performanceBucket, cloud_RoleName, cloud_...

resultCode: 200
client_City: Washington
client_CountryOrRegion: United States
client_StateOrProvince: Virginia
performanceBucket: <250ms
cloud_RoleName: api-gateway
cloud_RoleInstance: api-gateway-default-11-7b577bcf99-vnc6d

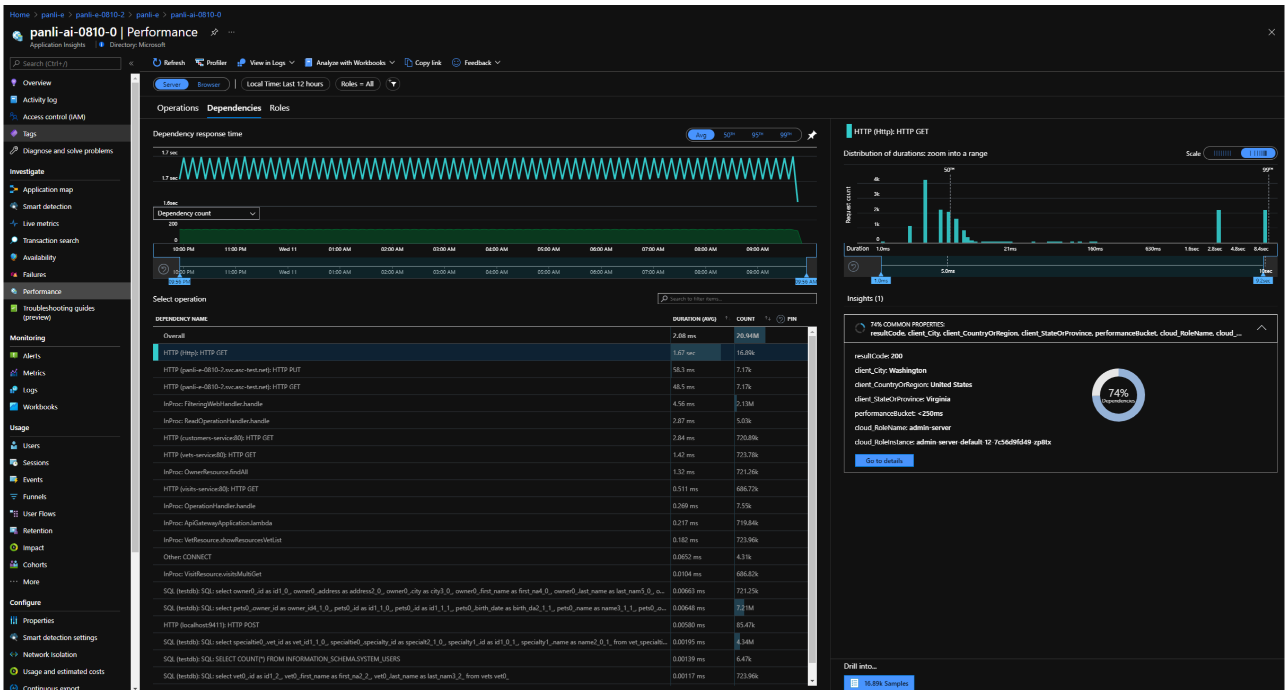
57% Requests

Go to details

Top 3 Dependencies

	COUNT	AVG
SQL	13.69M	0.0045ms
InProc	5M	2.2ms
HTTP	2.25M	14ms

- In the left navigation pane, click **Performance** to see the performance data of all apps' dpendencies.



Select operation

DEPENDENCY NAME	DURATION (AVG)	COUNT
Overall	2.08 ms	20.94M
HTTP (Http): HTTP GET	1.67 sec	16.89k
HTTP (panli-e-0810-2.svc.asc-test.net): HTTP PUT	58.3 ms	7.17k
HTTP (panli-e-0810-2.svc.asc-test.net): HTTP GET	48.5 ms	7.17k
InProc: FilteringWebHandler.handle	4.56 ms	2.13M
InProc: ReadOperationHandler.handle	2.87 ms	5.03k
HTTP (customers-service:80): HTTP GET	2.84 ms	720.89k
HTTP (vets-service:80): HTTP GET	1.42 ms	723.78k
InProc: OwnerResource.findAll	1.32 ms	721.26k
HTTP (visits-service:80): HTTP GET	0.511 ms	686.72k
InProc: OperationHandler.handle	0.269 ms	7.55k
InProc: ApiGatewayApplication.lambda	0.217 ms	719.84k
InProc: VetResource.showResourcesVetList	0.182 ms	723.96k
Other: CONNECT	0.0652 ms	4.31k
InProc: VisitResource.visitsMultiGet	0.0104 ms	686.82k
SQL (testdb): SQL: select owner0_.id as id1_0_, owner0_.address as address2_0_, owner0_.city as city3_0_, owner0_.first_name as first_na4_0_, owner0_.last_name as last_nam5_0_, o...	0.00663 ms	721.25k
SQL (testdb): SQL: select pets0_.owner_id as owner_id4_1_0_, pets0_.id as id1_1_0_, pets0_.id as id1_1_1_, pets0_.birth_date as birth_da2_1_1_, pets0_.name as name3_1_1_, pets0_.o...	0.00648 ms	7.21M
HTTP (localhost:9411): HTTP POST	0.00580 ms	85.47k
SQL (testdb): SQL: select specialtie0_.vet_id as vet_id1_1_0_, specialtie0_.specialty_id as specialt2_1_0_, specialty1_.id as id1_0_1_, specialty1_.name as name2_0_1_ from vet_specialti...	0.00195 ms	4.34M
SQL (testdb): SQL: SELECT COUNT(*) FROM INFORMATION_SCHEMA.SYSTEM_USERS	0.00139 ms	6.47k
SQL (testdb): SQL: select vet0_.id as id1_2_, vet0_.first_name as first_na2_2_, vet0_.last_name as last_nam3_2_ from vets vet0_	0.00117 ms	723.96k

Insights (1)

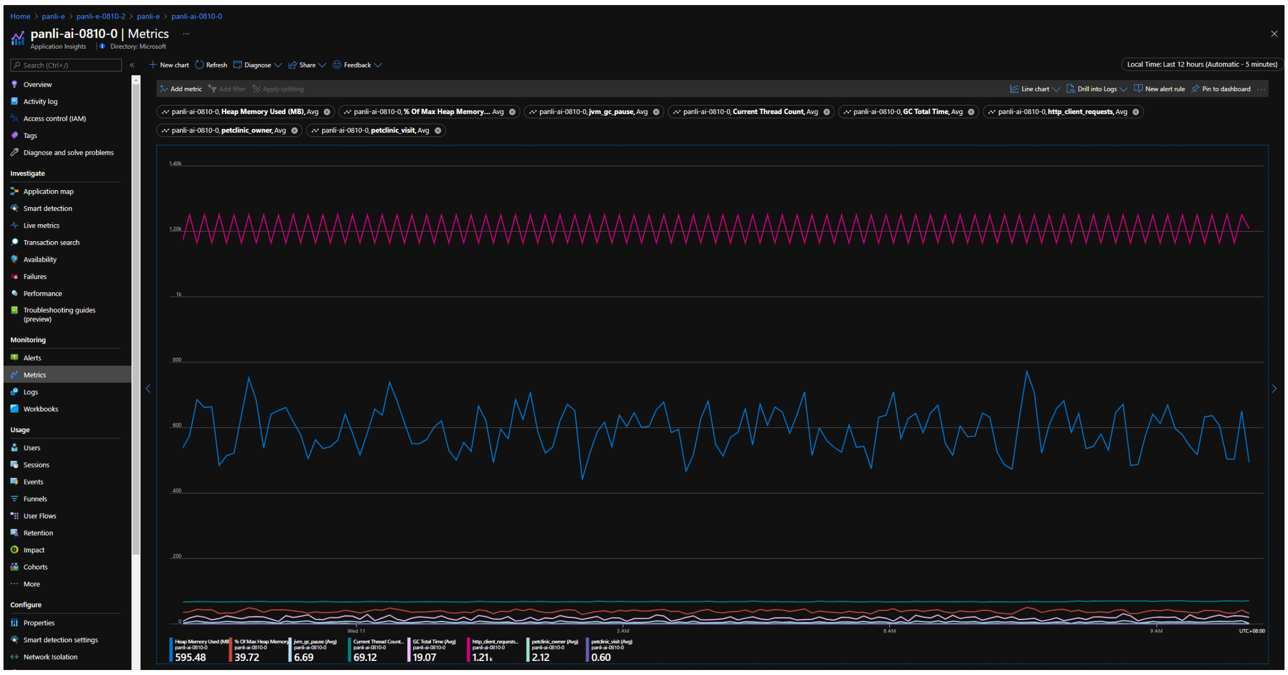
74% COMMON PROPERTIES:
resultCode, client_City, client_CountryOrRegion, client_StateOrProvince, performanceBucket, cloud_RoleName, cloud_...

resultCode: 200
client_City: Washington
client_CountryOrRegion: United States
client_StateOrProvince: Virginia
performanceBucket: <250ms
cloud_RoleName: admin-server
cloud_RoleInstance: admin-server-default-12-7c56d9fd49-zp8tx

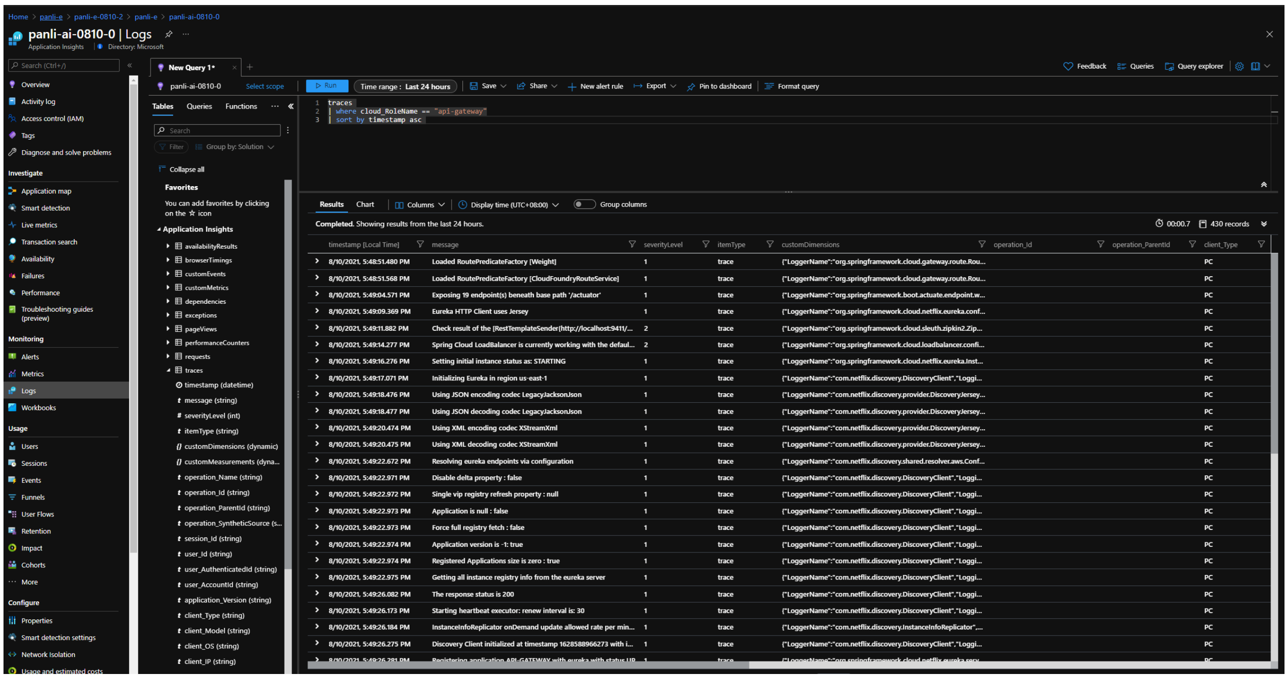
74% Dependencies

Go to details

- In the left navigation pane, click Metrics and select the namespace, you will see both Spring Boot metrics and custom metrics, if any.


panli-ai-0810-0 | Metrics
Heap Memory Used (MB) 595.48
% Of Max Heap Memory 39.72
jvm_gc_pause (Avg) 6.69
Current Thread Count 69.12
GC Total Time (Avg) 19.07
http_client_requests 1.21k
petclinic_owner (Avg) 2.12
petclinic_visit (Avg) 0.60

- In the left navigation pane, click **Log** to check logs of one(filter by cloud_RoleName) or all apps.



panli-ai-0810-0 | Logs

```
traces
| where cloud_RoleName == "api-gateway"
| sort by timestamp asc
```

Completed. Showing results from the last 24 hours. 00:00.7 430 records

timestamp [Local Time]	message	severityLevel	itemType	customDimensions	operation_Id	operation_ParentId	client_Type
8/10/2021, 5:48:51.480 PM	Loaded RoutePredicateFactory [Weight]	1	trace	{"LoggerName":"org.springframework.cloud.gateway.route.Rou...			PC
8/10/2021, 5:48:51.568 PM	Loaded RoutePredicateFactory [CloudFoundryRouteService]	1	trace	{"LoggerName":"org.springframework.cloud.gateway.route.Rou...			PC
8/10/2021, 5:49:04.571 PM	Exposing 19 endpoint(s) beneath base path '/actuator'	1	trace	{"LoggerName":"org.springframework.boot.actuate.endpoint.w...			PC
8/10/2021, 5:49:09.369 PM	Eureka HTTP Client uses Jersey	1	trace	{"LoggerName":"org.springframework.cloud.netflix.eureka.conf...			PC
8/10/2021, 5:49:11.882 PM	Check result of the [RestTemplateSender{http://localhost:9411/...	2	trace	{"LoggerName":"org.springframework.cloud.sleuth.zipkin2.Zip...			PC
8/10/2021, 5:49:14.277 PM	Spring Cloud LoadBalancer is currently working with the defaul...	2	trace	{"LoggerName":"org.springframework.cloud.loadbalancer.confi...			PC
8/10/2021, 5:49:16.276 PM	Setting initial instance status as: STARTING	1	trace	{"LoggerName":"org.springframework.cloud.netflix.eureka.Inst...			PC
8/10/2021, 5:49:17.071 PM	Initializing Eureka in region us-east-1	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:18.476 PM	Using JSON encoding codec LegacyJacksonJson	1	trace	{"LoggerName":"com.netflix.discovery.provider.DiscoveryJersey...			PC
8/10/2021, 5:49:18.477 PM	Using JSON decoding codec LegacyJacksonJson	1	trace	{"LoggerName":"com.netflix.discovery.provider.DiscoveryJersey...			PC
8/10/2021, 5:49:20.474 PM	Using XML encoding codec XStreamXml	1	trace	{"LoggerName":"com.netflix.discovery.provider.DiscoveryJersey...			PC
8/10/2021, 5:49:20.475 PM	Using XML decoding codec XStreamXml	1	trace	{"LoggerName":"com.netflix.discovery.provider.DiscoveryJersey...			PC
8/10/2021, 5:49:22.672 PM	Resolving eureka endpoints via configuration	1	trace	{"LoggerName":"com.netflix.discovery.shared.resolver.aws.Conf...			PC
8/10/2021, 5:49:22.971 PM	Disable delta property : false	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:22.972 PM	Single vip registry refresh property : null	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:22.973 PM	Application is null : false	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:22.973 PM	Force full registry fetch : false	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:22.974 PM	Application version is -1: true	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:22.974 PM	Registered Applications size is zero : true	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:22.975 PM	Getting all instance registry info from the eureka server	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:26.082 PM	The response status is 200	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:26.173 PM	Starting heartbeat executor: renew interval is: 30	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC
8/10/2021, 5:49:26.184 PM	InstanceInfoReplicator onDemand update allowed rate per min...	1	trace	{"LoggerName":"com.netflix.discovery.InstanceInfoReplicator",...			PC
8/10/2021, 5:49:26.275 PM	Discovery Client initialized at timestamp 1628588966273 with i...	1	trace	{"LoggerName":"com.netflix.discovery.DiscoveryClient","Loggi...			PC

Use Azure CLI to manage settings of Application Insights

We use Buildpacks Binding to integrate Azure Application Insights with type `ApplicationInsights`.

- Create Application Insights buildpacks binding.

```
az spring-cloud build-service buildpacks-binding create --name {your-
binding-name} \
    --type ApplicationInsights \
    --properites sampling-rate={your-sampling-rate} \
    --secrets connection-string={your-connection-string} \
    -s ${asc-resource-name} -g ${resource-group-name}
```

- Replace Application Insights buildpacks binding.

  ```
  az spring-cloud build-service buildpacks-binding set --name {your-binding-
  name} \
     --type ApplicationInsights \
     --properites sampling-rate={your-sampling-rate} \
     --secrets connection-string={your-connection-string} \
     -s ${asc-resource-name} -g ${resource-group-name}
  ```

- Get Application Insights buildpacks binding.

  ```
  az spring-cloud build-service buildpacks-binding show --name {your-binding-
  name} \
     -s ${asc-resource-name} -g ${resource-group-name}
  ```

- Delete Application Insights buildpacks binding.

  ```
  az spring-cloud build-service buildpacks-binding delete --name {your-
  binding-name} \
     -s ${asc-resource-name} -g ${resource-group-name}
  ```

Concept mapping between Azure Spring Cloud and Application Insights

Azure Spring Cloud	Application Insights
`App`	* **Application Map**/Role * **Live Metrics**/Role * **Failures**/Roles/Cloud Role * **Performance**/Roles/Could Role
`App Instance`	* **Application Map**/Role Instance * **Live Metrics**/Service Name * **Failures**/Roles/Cloud Instance * **Performance**/Roles/Could Instance

The name `App Instance` from Azure Spring Cloud will be changed or generated in the following scenarios:

- You create a new app.
- You deploy a JAR file or source code to an existing app.
- You initiate a blue/green deployment.
- You restart the app.
- You stop the deployment of an app, and then restart it.

When data is stored in Application Insights, it contains the history of Azure Spring Cloud app instances created or deployed since the Java agent was enabled. This means that, in the Application Insights portal, you can see app data created yesterday, but then deleted within a specific time range, like the last 24 hours. The following scenarios show how this works:

- You created an app around 8:00 AM today from Azure Spring Cloud with the Java agent enabled, and then you deployed a JAR file to this app around 8:10 AM today. After some testing, you change the code and deploy a new JAR file to this app at 8:30 AM today. Then, you take a break, and when you come back around 11:00 AM, you check some data from Application Insights. You will see:
  - Three instances in Application Map with time ranges in the last 24 hours, as well as Failures, Performance, and Metrics.
  - One instance in Application Map with a time range in the last hour, as well as Failures, Performance, and Metrics.
  - One instance in Live Metrics.
- You created an app around 8:00 AM today from Azure Spring Cloud with the Java agent enabled, and then you deployed a JAR file to this app around 8:10 AM today. Around 8:30 AM today, you try a blue/green deployment with another JAR file. Currently, you have two deployments for this app. After a break around 11:00 AM today, you want to check some data from Application Insights. You will see:
  - Three instances in Application Map with time ranges in the last 24 hours, as well as Failures, Performance, and Metrics.
  - Two instances in Application Map with time ranges in last hour, as well as Failures, Performance, and Metrics.
  - Two instances in Live Metrics.

Frequently Asked Questions

Fail to deploy apps

> ```
> az spring-cloud app deploy --name <app name> <app.jar>
> ```
>
> 112404: Failed to wait for deployment instances to be ready. Please check the application log (see https://aka.ms/azure-spring-cloud-doc-log), and try again later.

Please check if the below error messages in the application log.

1. Spring Boot 2.4 and higher versions

   ```
   Application failed to start due to an exception
   org.springframework.cloud.commons.ConfigDataMissingEnvironmentPostProcessor$
   ImportException: No spring.config.import set
   ```

2. Spring Boot 2.3 and earlier versions

   ```
   WARN c.c.c.ConfigServicePropertySourceLocator : Could not locate
   PropertySource: I/O error on GET request for
   ```

```
"http://localhost:8888/application/default": Connection refused: connect;
nested exception is java.net.ConnectException: Connection refused: connect
```

It is probably caused by adding spring-cloud-starter-config starter by mistake. Please remove the dependency and try again.

112034: Failed to provision resource.

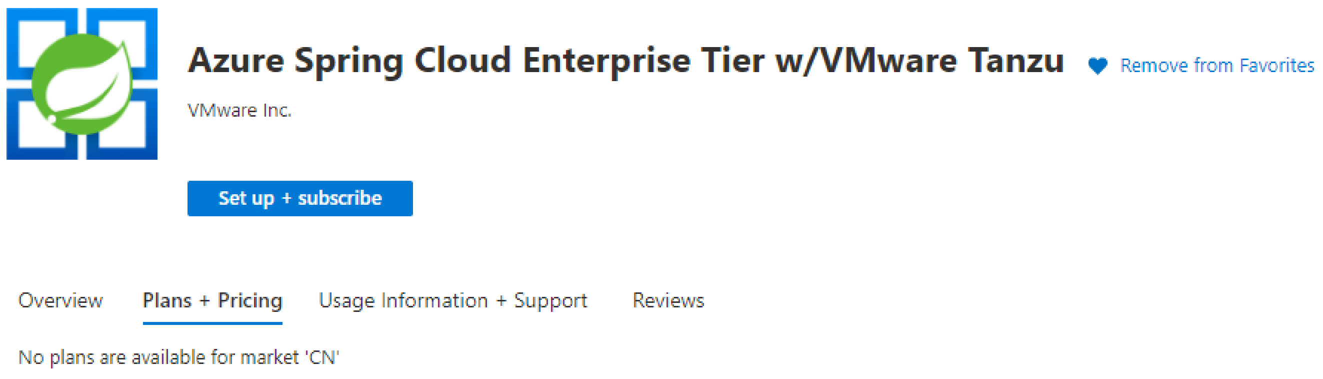
When failed to create Azure Spring Cloud Enterprise tier instance with error message "112034: Failed to provision resource.". Check whether your Azure subscription's billing account address is in the supported location. See "No plans are available for market '<Location>'" for more details.

If that doesn't help, you can contact support team with the following info.

```
AZURE_TENANT_ID=<your Azure Tenant Id which host the Azure subscription>
AZURE_SUBSCRIPTION_ID=<your Azure subscription Id which is used to create Spring
Cloud instance>
SPRING_CLOUD_NAME=<the failed instance name>
```

No plans are available for market '<Location>'

When you visit SaaS offer Azure Spring Cloud Enterprise Tier w/VMware Tanzu in Marketplace, it may say "No plans are available for market '<Location>'" like the following image.



No plans are available for market 'CN'

Azure Spring Cloud Enterprise tier needs customers to pay for a license to Tanzu components through an Azure Marketplace offer. To purchase in Azure Marketplace, the billing account's country or region for your Azure subscription should be in the SaaS offer's support geographic locations.

Azure Spring Cloud Enterprise Tier w/VMware Tanzu now supports all geographic locations that Azure Marketplace supports. See Supported geographic locations.

You can view your billing account for your subscription if you have admin access, see view billing accounts.

> Azure Spring Cloud Enterprise tier is free for private preview.

Azure CLI Command Reference

> This reference is part of the spring-cloud extension for Azure CLI and requires version 2.0.67 or higher. Install extension by running `az extension add -s https://ascprivatecli.blob.core.windows.net/enterprise/spring_cloud-2.7.0a1-py3-none-any.whl -y`

Commands to manage Azure Spring Cloud Enterprise tier.

Commands

az spring-cloud

- **az spring-cloud list** - List all Azure Spring Cloud in the given resource group, otherwise list the subscription's.
- **az spring-cloud show** - Show the details for an Azure Spring Cloud.
- **az spring-cloud delete** - Delete an Azure Spring Cloud.

az spring-cloud app

- **az spring-cloud app create** - Create a new app with a default deployment in the Azure Spring Cloud.
- **az spring-cloud app delete** - Delete an app in the Azure Spring Cloud.
- **az spring-cloud app deploy** - Deploy source code or pre-built binary to an app and update related configurations.
- **az spring-cloud app list** - List all apps in the Azure Spring Cloud.
- **az spring-cloud app logs** - Show logs of an app instance, logs will be streamed when setting '-f/--follow'.
- **az spring-cloud app restart** - Restart instances of the app, default to production deployment.
- **az spring-cloud app scale** - Manually scale an app or its deployments.
- **az spring-cloud app show** - Show the details of an app in the Azure Spring Cloud.
- **az spring-cloud app start** - Start instances of the app, default to production deployment.
- **az spring-cloud app stop** - Stop instances of the app, default to production deployment.
- **az spring-cloud app update** - Update configurations of an app.

az spring-cloud test-endpoint

- **az spring-cloud test-endpoint disable** - Disable test endpoint of the Azure Spring Cloud.
- **az spring-cloud test-endpoint enable** - Enable test endpoint of the Azure Spring Cloud.
- **az spring-cloud test-endpoint list** - List test endpoint keys of the Azure Spring Cloud.
- **az spring-cloud test-endpoint renew-key** - Regenerate a test-endpoint key for the Azure Spring Cloud.

az spring-cloud build-service buildpacks-binding

- **az spring-cloud build-service buildpacks-binding create** - Create a buildpacks binding.
- **az spring-cloud build-service buildpacks-binding set** - Set a buildpacks binding.
- **az spring-cloud build-service buildpacks-binding show** - Show a buildpacks binding, the secrets will be masked.
- **az spring-cloud build-service buildpacks-binding delete** - Delete a buildpacks binding.

az spring-cloud application-configuration-service

- **az spring-cloud application-configuration-service clear** - Reset all Application Configuration Service settings.
- **az spring-cloud application-configuration-service git repo add** - Add an item of git property to Application Configuration Service settings.
- **az spring-cloud application-configuration-service git repo update** - Update an existing item of git property to Application Configuration Service settings.
- **az spring-cloud application-configuration-service git repo list** - List all git settings of Application Configuration Service.
- **az spring-cloud application-configuration-service git repo remove** - Delete an existing item of git property to Application Configuration Service settings.
- **az spring-cloud application-configuration-service show** - Show provisioning status, runtime status and settings of Application Configuration Service.
- **az spring-cloud application-configuration-service bind** - Bind app to Application Configuration Service.
- **az spring-cloud application-configuration-service unbind** - Unbind app to Application Configuration Service.

az spring-cloud service-registry

- **az spring-cloud service-registry show** - Show provisioning status and runtime status of Service Registry.
- **az spring-cloud service-registry bind** - Bind app to Service Registry.
- **az spring-cloud service-registry unbind** - Unbind app to Service Registry.