**Challenge #1**

A 3-tier environment is a common setup. Use a tool of your choosing/familiarity create these resources on a cloud environment (Azure/AWS/GCP). Please remember we will not be judged on the outcome but more focusing on the approach, style and reproducibility.

**Deployed the Blazer application on Azure Kubernetes Cluster**

Prerequisites: -

1. Azure Devops Portal
2. Azure Portal
3. Terraform
4. Azure Repository
5. Azure CLI
6. VS Code
7. PostgreSQL Flexible Server
8. PgAdmin tool

Created service connection between Azure Devops and Azure Portal using Service principle (AD).

**Infra CICD Pipeline:**

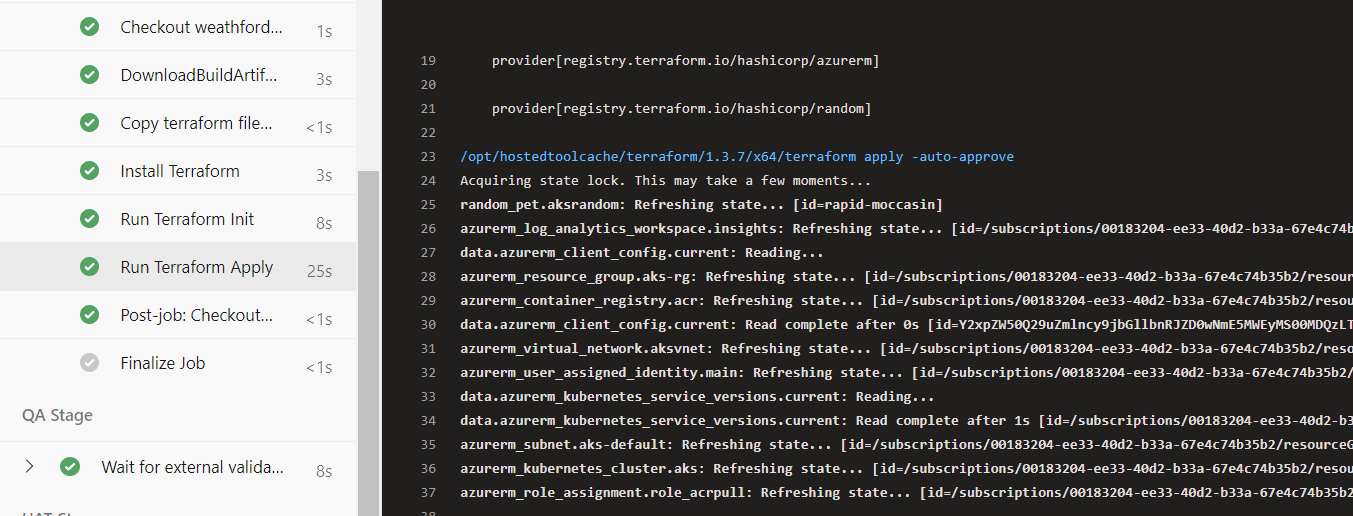
Created all Terraform modules as per our scope and pushed into one common azure repository.

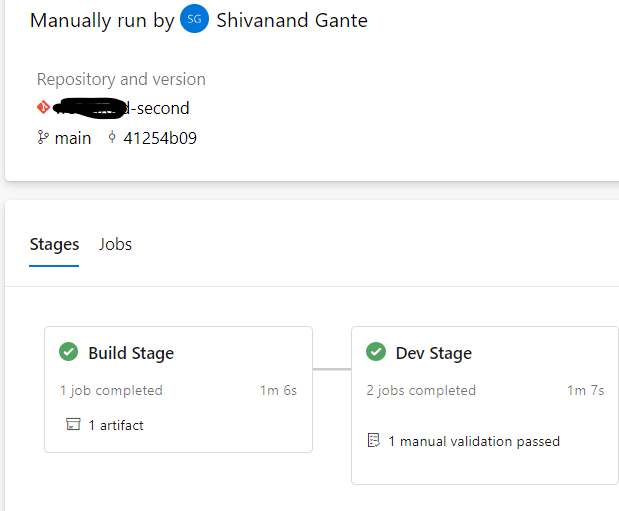
From another azure repository means root module call to the child terraform module and create a terraform pipeline(A Terraform module can call other modules to add their resources to the configuration.)

Using **multistage** pipeline, Created Azure **RG**, Vnet, Subnet, Private DNS, Key Vault, **ACR**, **AKS** and **PostgreSQL** Flexible Server etc through Terraform.

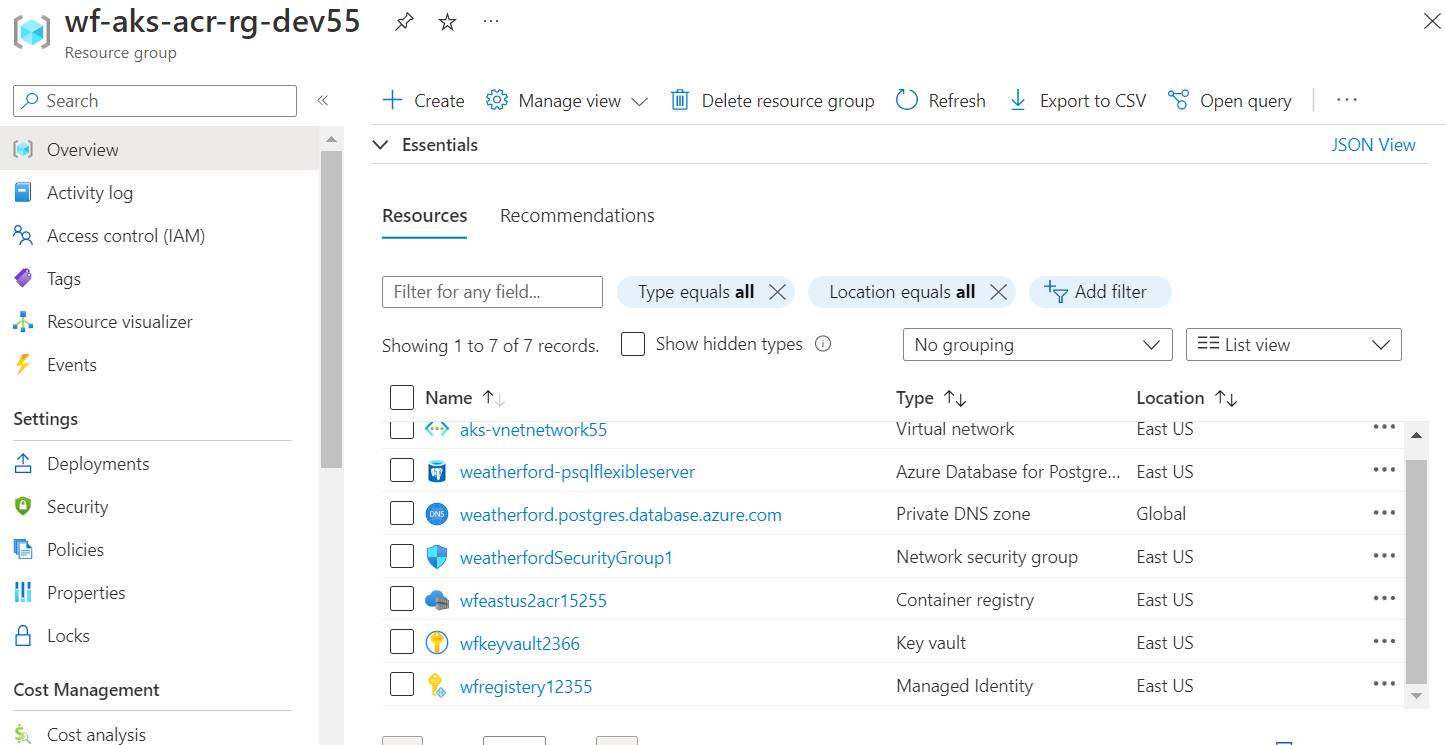
I have Created Terraform Azure Devops CICD pipeline using multistage yml file.

**Infr CICD pipeline using Multistage yml file**





**Infra got created: -**



After Provisioning the Infra, I have Deployed one Blazer Application on AKS Cluster through CI/CD Pipeline.

1. I have created Service connection between Azure Devops and Docker registry for pushing/storing the latest image.
2. I have created Service connection between Azure Devops and AKS cluster for deploying the image on aks container.

All service connection created successfully and now need to create a CICD pipeline.

1. CI Pipeline - Build the new Balzer Image through docker file and pushed into ACR Repository along with BuildID (This process happened in the Continuous Integration (CI)) Pipeline.
2. Created one AKS Deployment.yml file to deploy the PODS, Services on AKS cluster.
3. CD Pipeline (Release) - I have created pipeline variables and Task groups. In the CD pipeline Added few tasks like Create a secrete for pull the image from ACR and deploy on AKS.
4. Successfully deployed the image on AKS cluster with the help of aks deploy action task.

Verify on Azure portal.

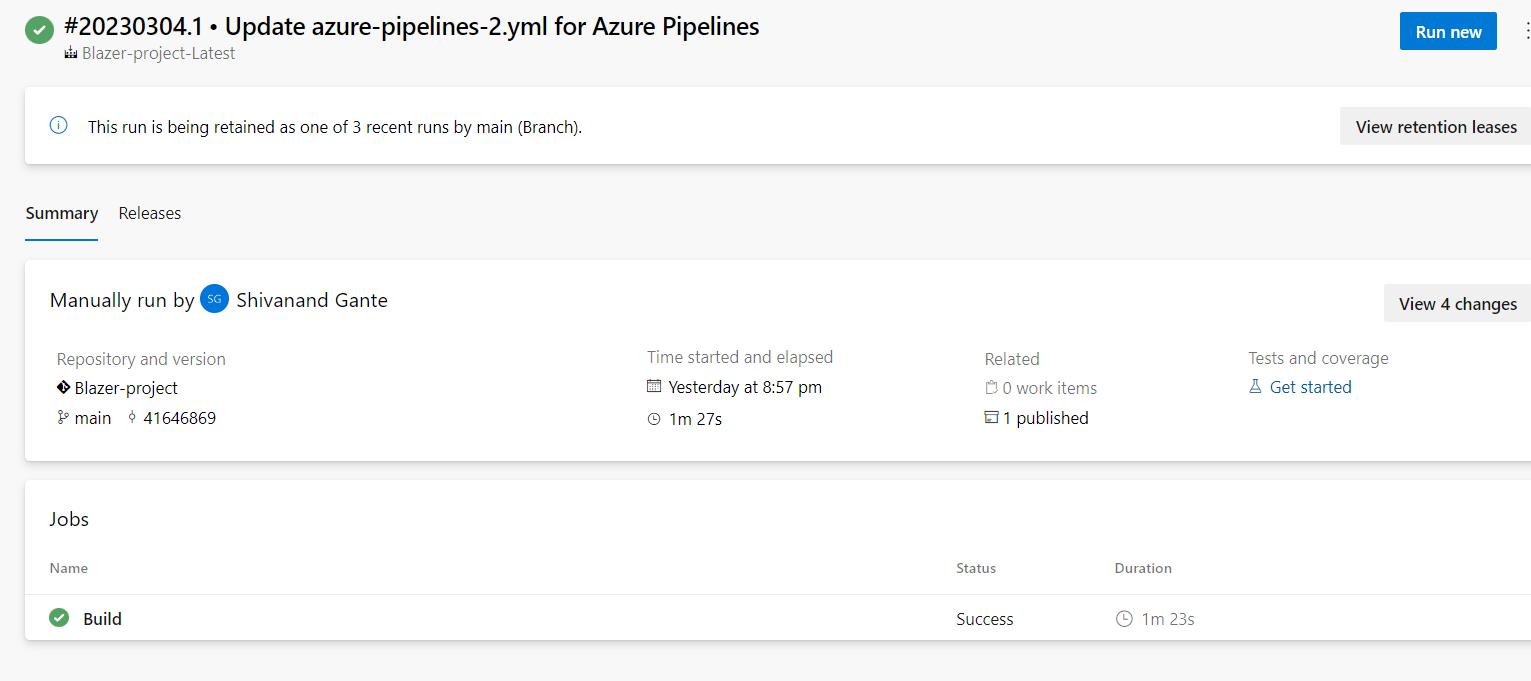
1. Go to Azure portal – AKS – connect with help of below commands.

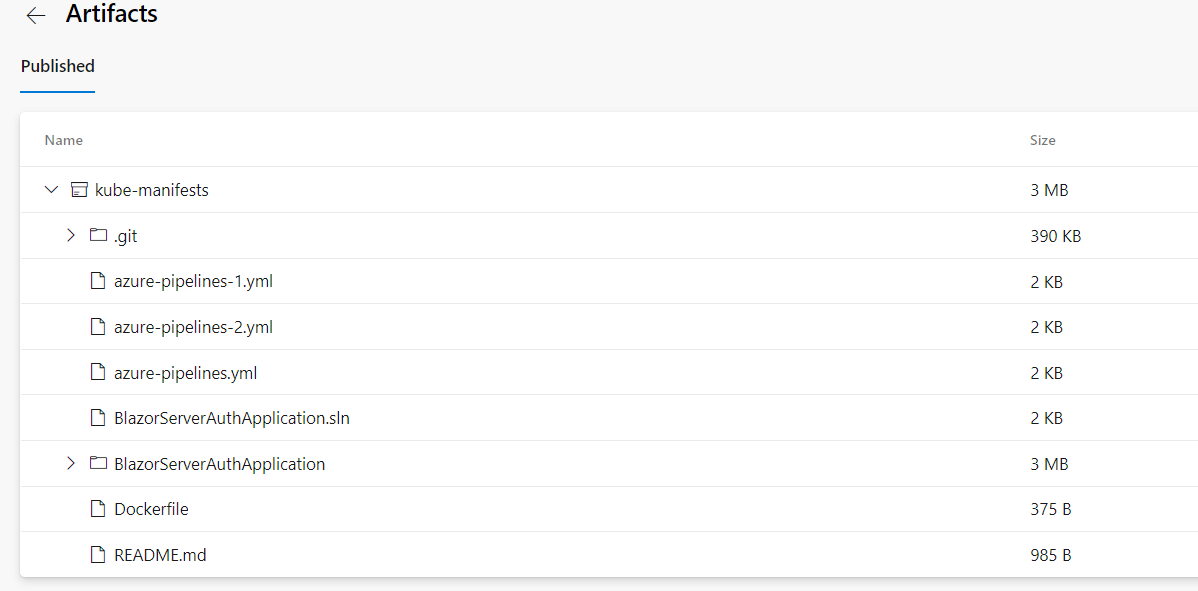
* az account set --subscription xxxx-xxxx-xxx-xxxx-xxxx-xxx
* az aks get-credentials --resource-group xx-aks-acr-rg-dev12 --name xxxxxxxxdaks1

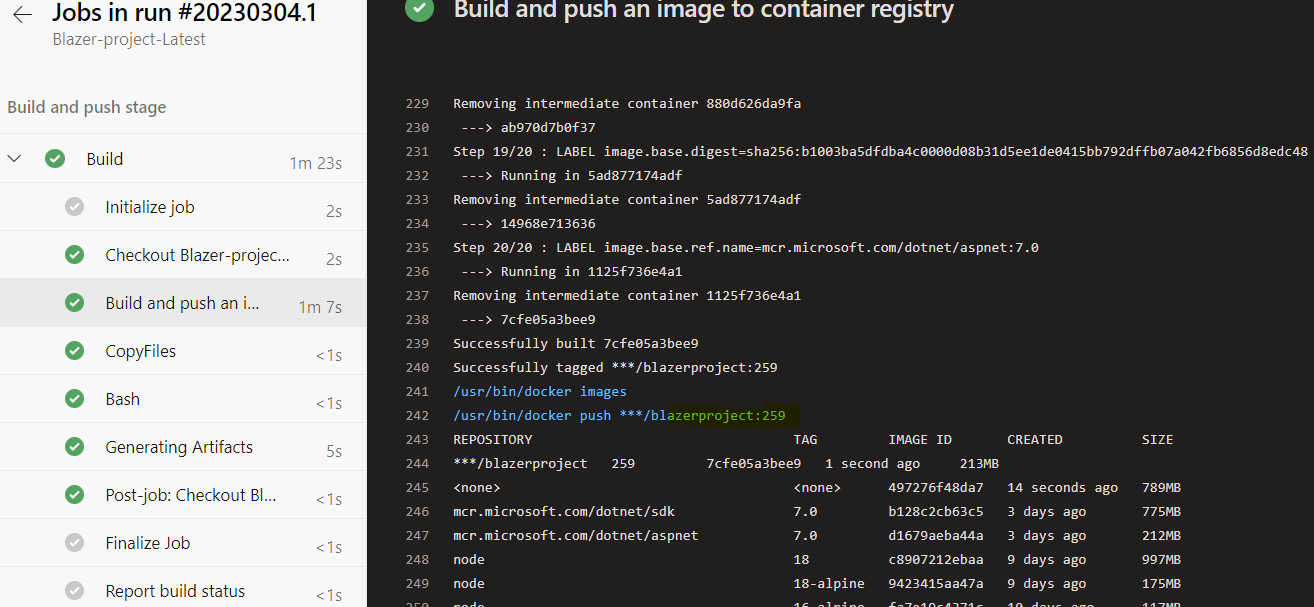
Verifying the all pods and services created or not with specific namespace.

1. kubectl get pods -n dev and kubectl get svc -n dev
2. Verify the blazer application access with the help of External IP Address.

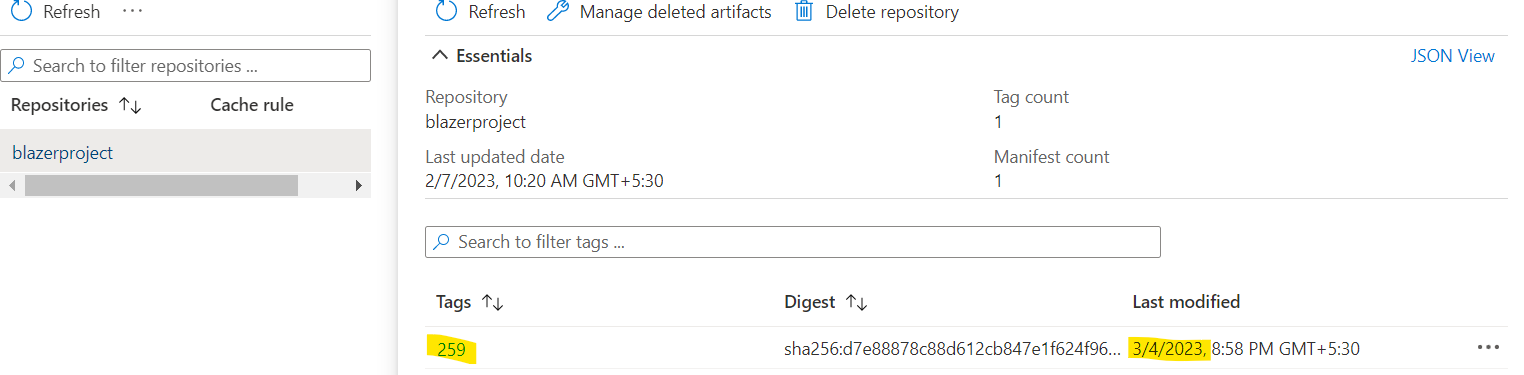
**Blazer Continuous Integration Pipeline: -**

****

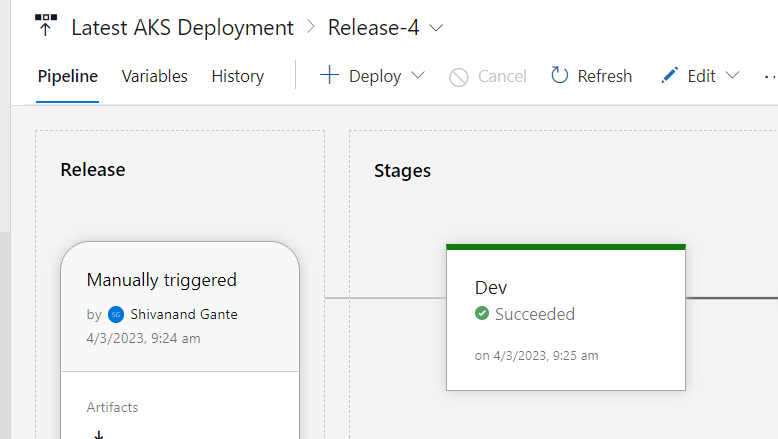
****

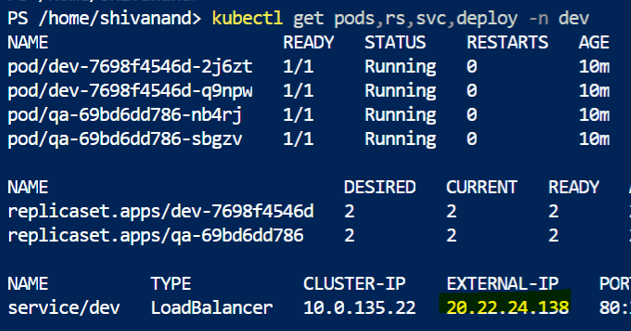
****

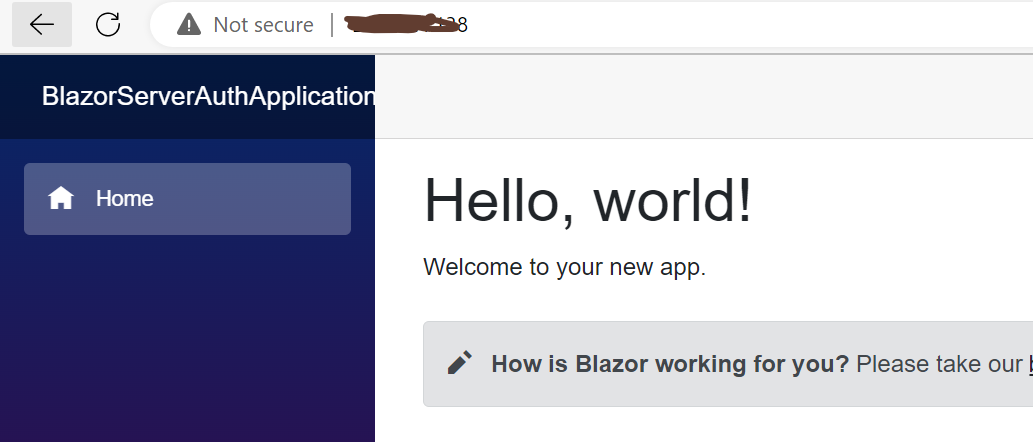
**Azure Container Registry: Image got pushed**

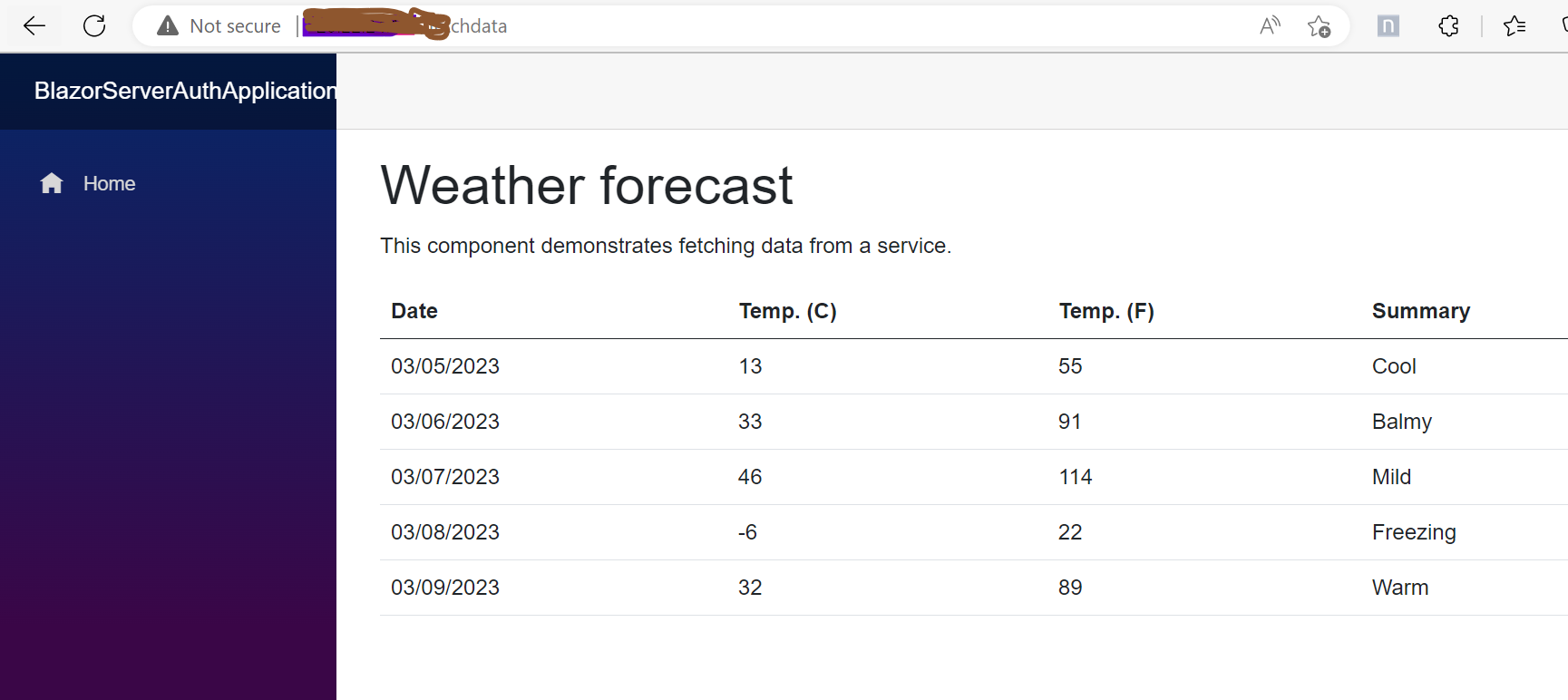


**Blazer Release Pipeline (CD):**

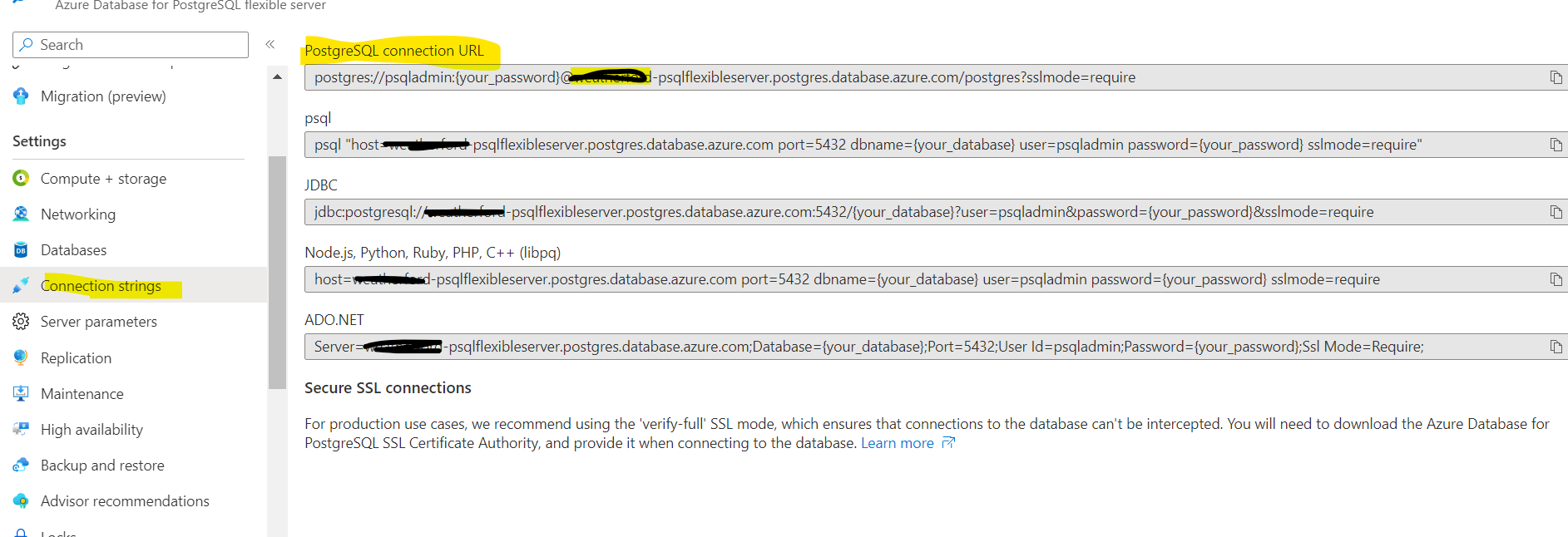








**Database: PostgreSQL Flexible Server**



Add this Postgresql URL into Balzer application - appsettings.json - ConnectionString

Once deployed our application what happened it will connect both blazer application and database.

Using PgAdmin tool, login to Postgresql server.

