# Azure Infrastructure

# TLDR;

The bicep files build the complete Azure infrastructure needed for the dev environment. Note the role assignments and the potential bicep bug (?) when attempting to deploy a key vault key (a secret seems to be no problem). Use VS Code with the bicep extension, you don’t need the cli.

## Azure Container Registry

ACR

## Azure Kubernetes Service

A straight-forward aks deployment of AKS with perhaps one point of note – For a dev environment I have deployed the HTTP application routing add-on. The add-on is designed to let you quickly create an ingress controller and access applications. When enabled, it configures an Ingress controller in an AKS cluster. As applications are deployed, the solution also creates publicly accessible DNS names for application endpoints. It creates a DNS Zone in the subscription

This add-on is *not designed for use in a production environment* and is not recommended for production use. For production-ready ingress deployments that include multiple replicas and TLS support.

## Azure Key Vault

## Cosmos Db – Sql Api

A Cosmos Db account, a database and 4 containers are deployed

Partition keys - all *“/id”* have been setup – this key needs confirmed with the microservice owner.

Throughput is set to the minimum (400Rus) and is available as a parameter in the deployment

## Cosmos Db - Mongo Api

A Cosmos Db account, a database and 1 container are deployed

The indices still need to be specified in the bicep file with the Register microservice owner.

Throughput is set to the minimum (400Rus) and is available as a parameter in the deployment

## Azure Service Bus

For siccar pilot-prod we should follow Microsoft’s ACR best practices document

## Pipeline

An Azure DevOps Pipeline is used to run the bicep file and deploy the infrastructure

Ingress choice

• Ocelot does not support gRPC

traefik