**Provision IaC and Deploy Web Application to AKS**

The document describes a web application deployment to Azure Kubernetes (AKS) on Azure.

**Tools and applications:**

* Azure Cloud
* Azure DevOps
* ASP .NET Core Web Application
* Visual Studio Code
* Azure CLI
* Kubectl
* Terraform

**Provision IaC using Terraform**

**The terraform will provision the services below:**

* Azure Continuer Registry (ACR)
* Azure Kubernetes Cluster (AKS)
* Azure SQL Server
* Empty database

**Create Service Principal with Contributor role in Azure AD:**

az ad sp create-for-rbac --role="Contributor" --scopes="subscriptions/133bc1d5-767e-4628-a9d1-0914ff59665c" --name="TerraformServicePrincipal”"

**Create backend storage in Azure for terraform.state:**



The powershell script creates the storage and provides **access key** like this: xaoKqIfd8j8y2txJCfat95rRMMKG15vKkZsX5PTuoKUn801d7mXo/qpkgPRPpPBlSLAPpHxAUTwcRFJK9NbmFg==

**Set the access key in the environment variable:**

setx ARM\_ACCESS\_KEY xaoKqIfd8j8y2txJCfat95rRMMKG15vKkZsX5PTuoKUn801d7mXo/qpkgPRPpPBlSLAPpHxAUTwcRFJK9NbmFg==

**Create basic main.tf for ACR, AKS and SQL Server with empty Database:**



**terraform init**

**terraform plan**

**terraform apply**

**Check the AKS is up and running:**

az aks get-credentials --resource-group rgyaronz --name aksyaronz --overwrite-existing

kubectl get nodes

**Deploy the Web Application to AKS**

**Clone the web application source code from:**

<https://github.com/yaronzlot/EmployeeManagement>

git clone https://github.com/yaronzlot/EmployeeManagement.git

**Update the Azure SQL Server connection string in the web app code and update the empty database in Azure (using EF migration):**

dotnet ef database update

dotnet build

dotnet run

http://localhost:5000 – verify the web application works against the update database in Azure SQL Server

**Create Dockerfile:**

****

****

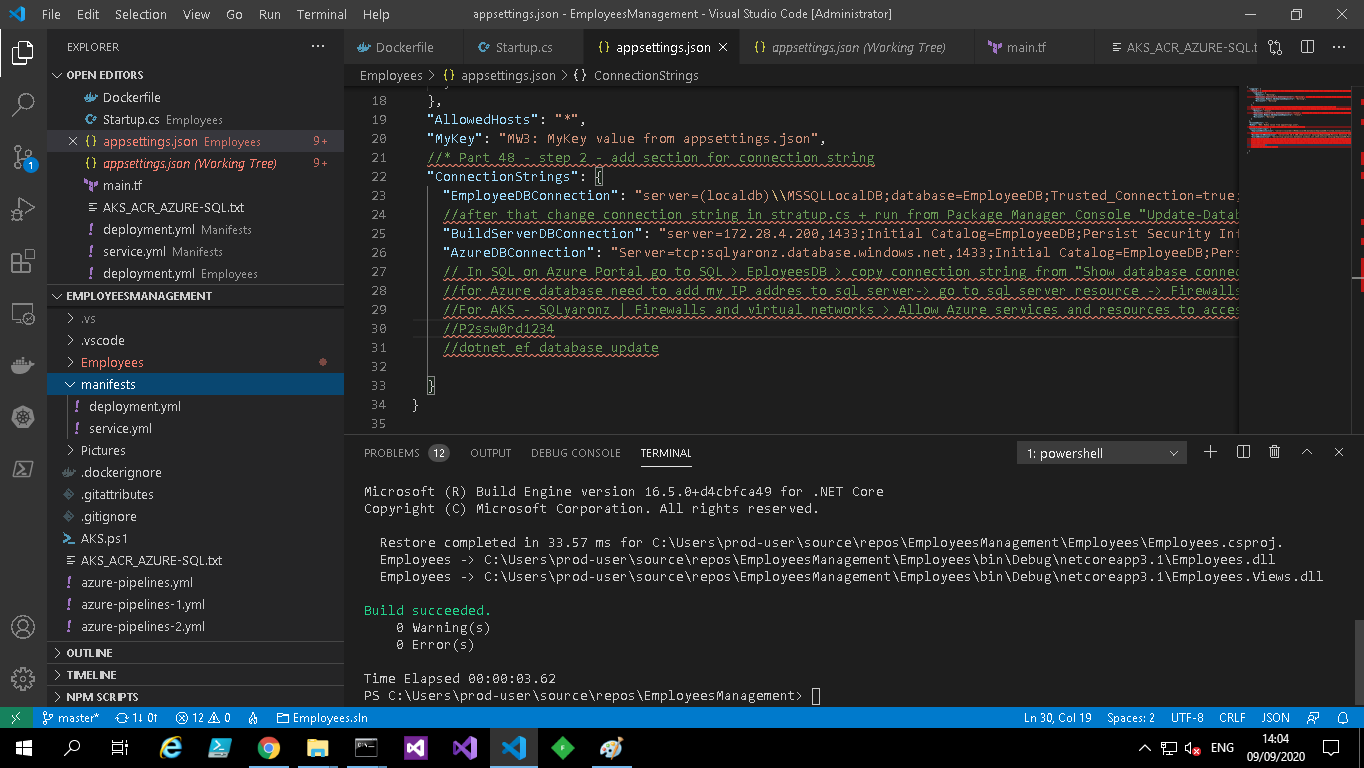
**Create** **deployment.yml and service.yml in manifest folder:**

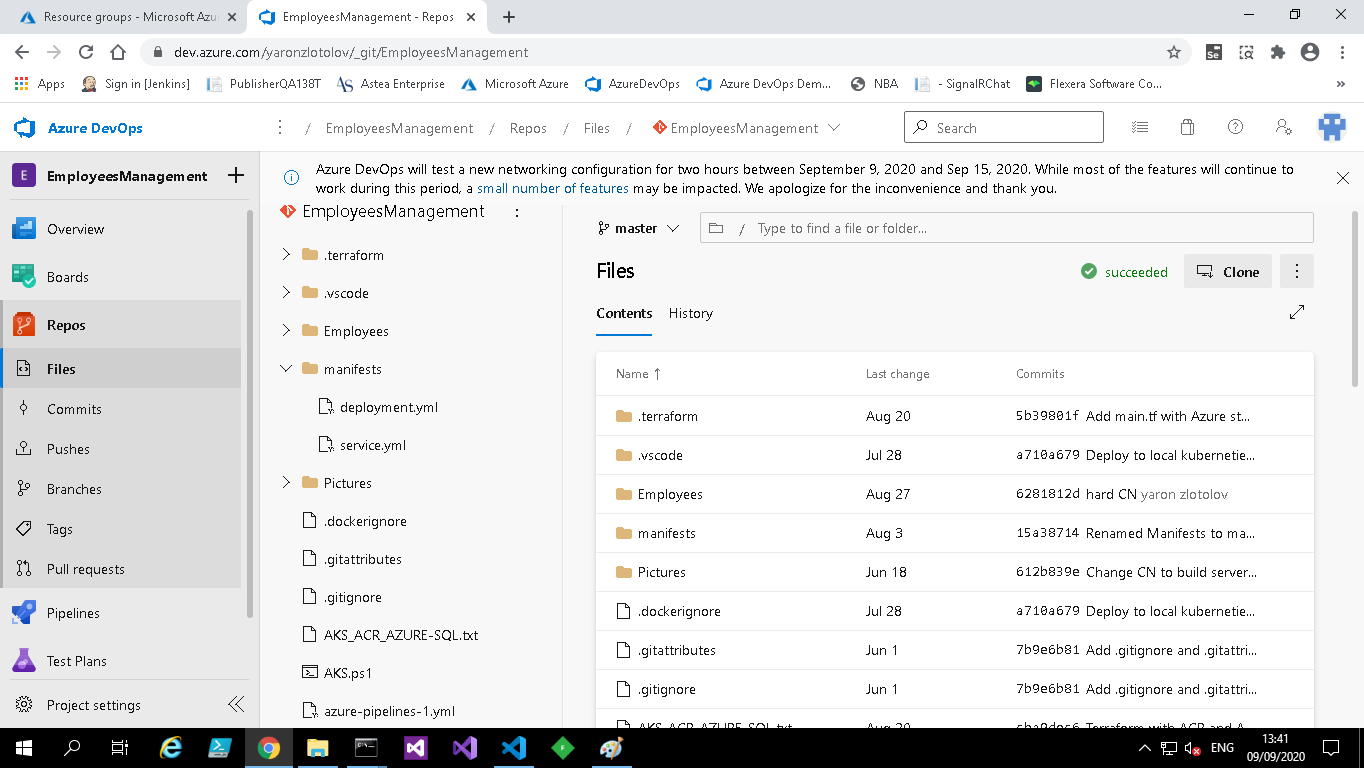






**Create project in Azure DevOps and push the code into the Azure GIT repo:**





**Create the pipeline (below is the pipeline yml):**

****

**NOTE**: Before this step it is required to remove locks and IP restrictions from AKS done by Terraform:

az lock list

az lock delete –ids $lockid

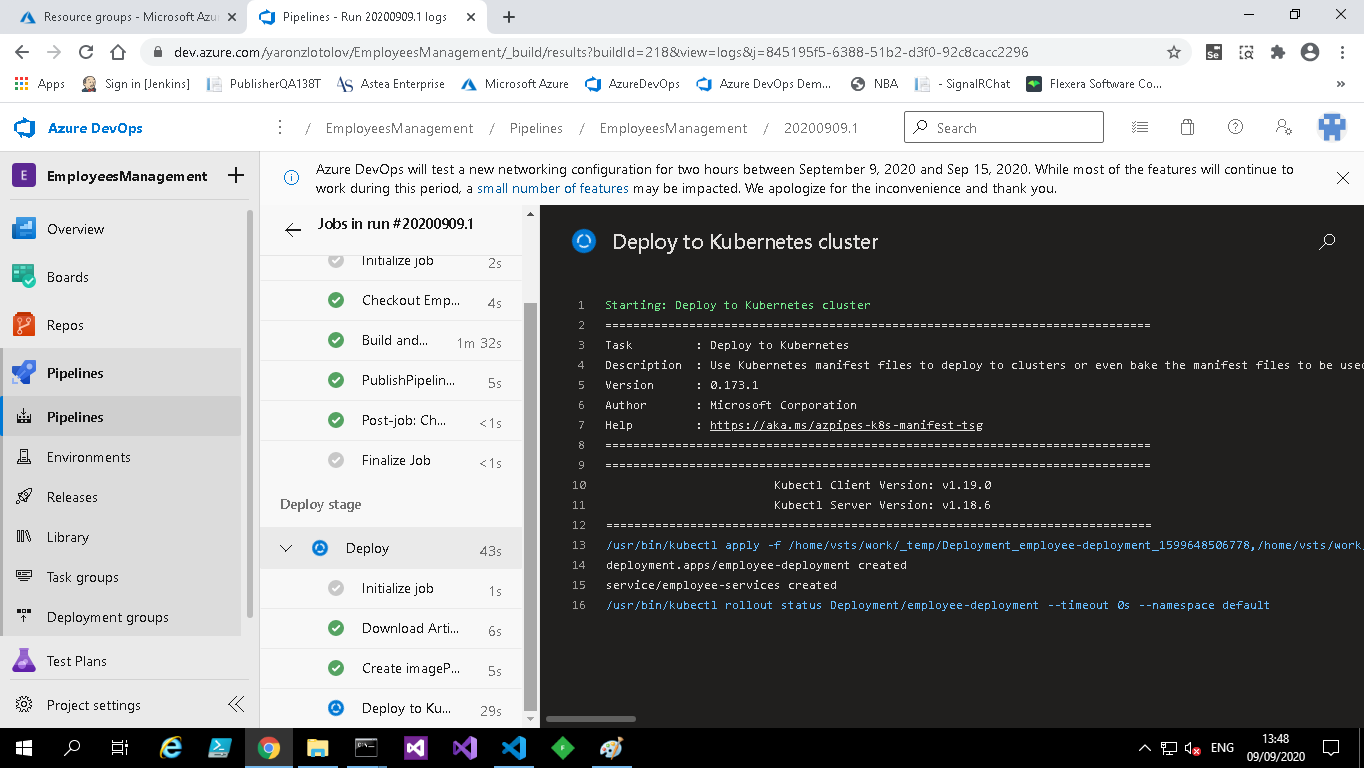
az aks update --resource-group rgyaronz --name aksyaronz --api-server-authorized-ip-ranges=

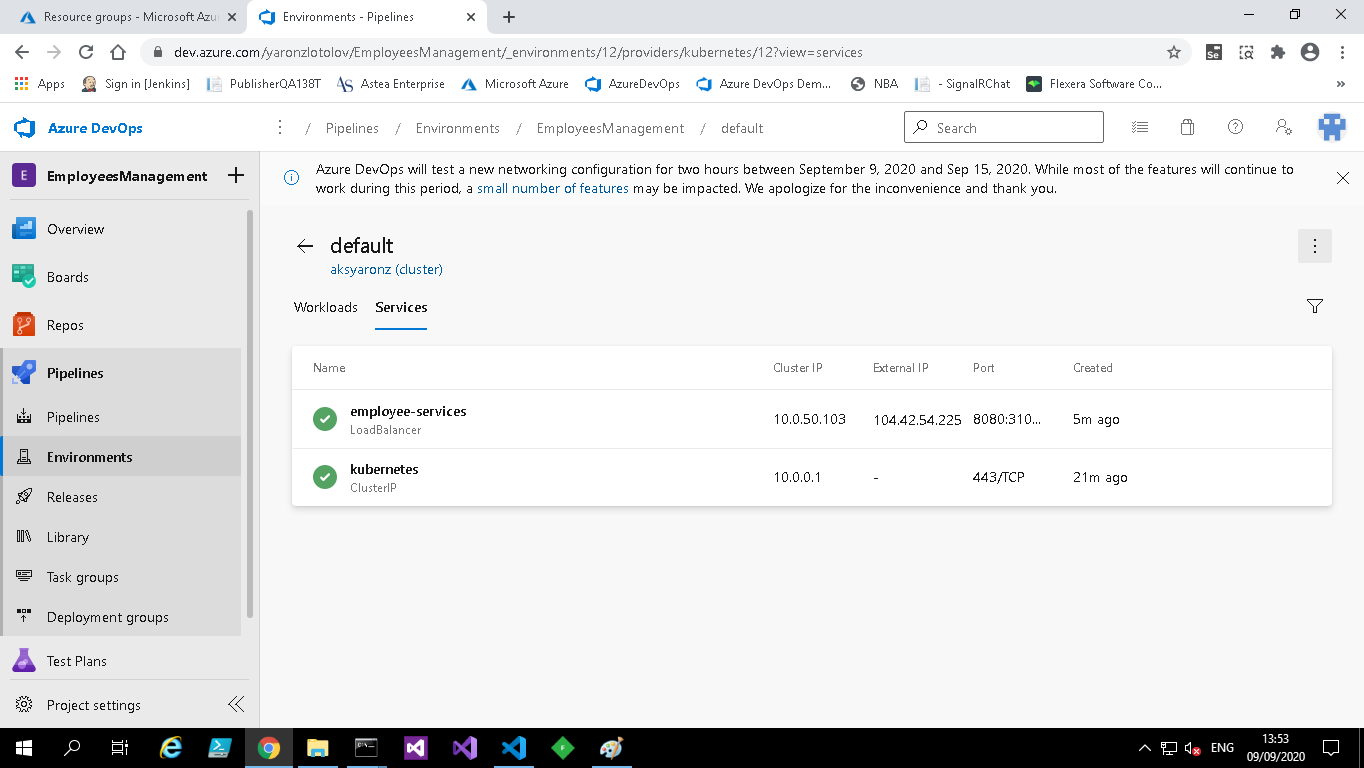
**Build**:

* Checkout the code from GIT repo
* Build the web application
* Create docker image
* Push the docker image to ACR
* Publish artifact for deployment

**Deploy**:

* Download the artifact
* Pull the image from ACR
* Deploy the docker image to AKS

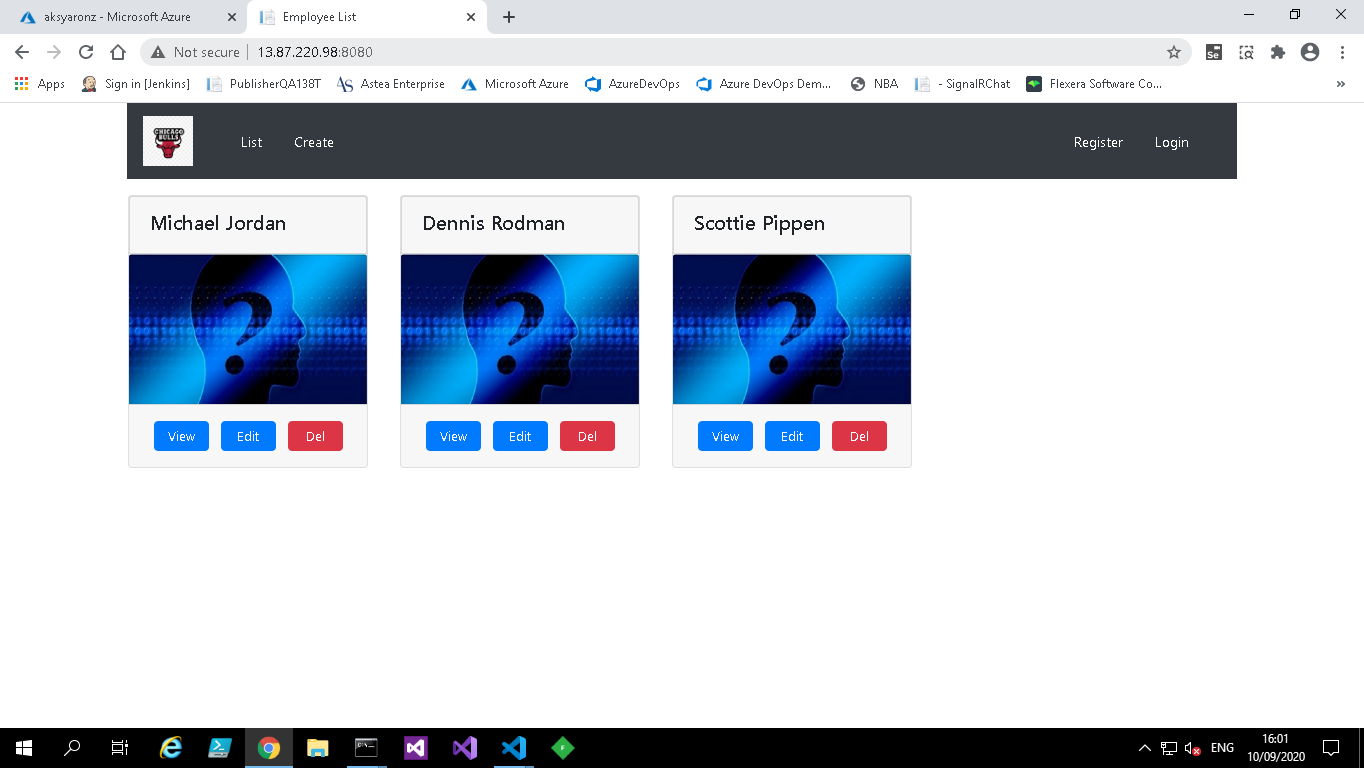




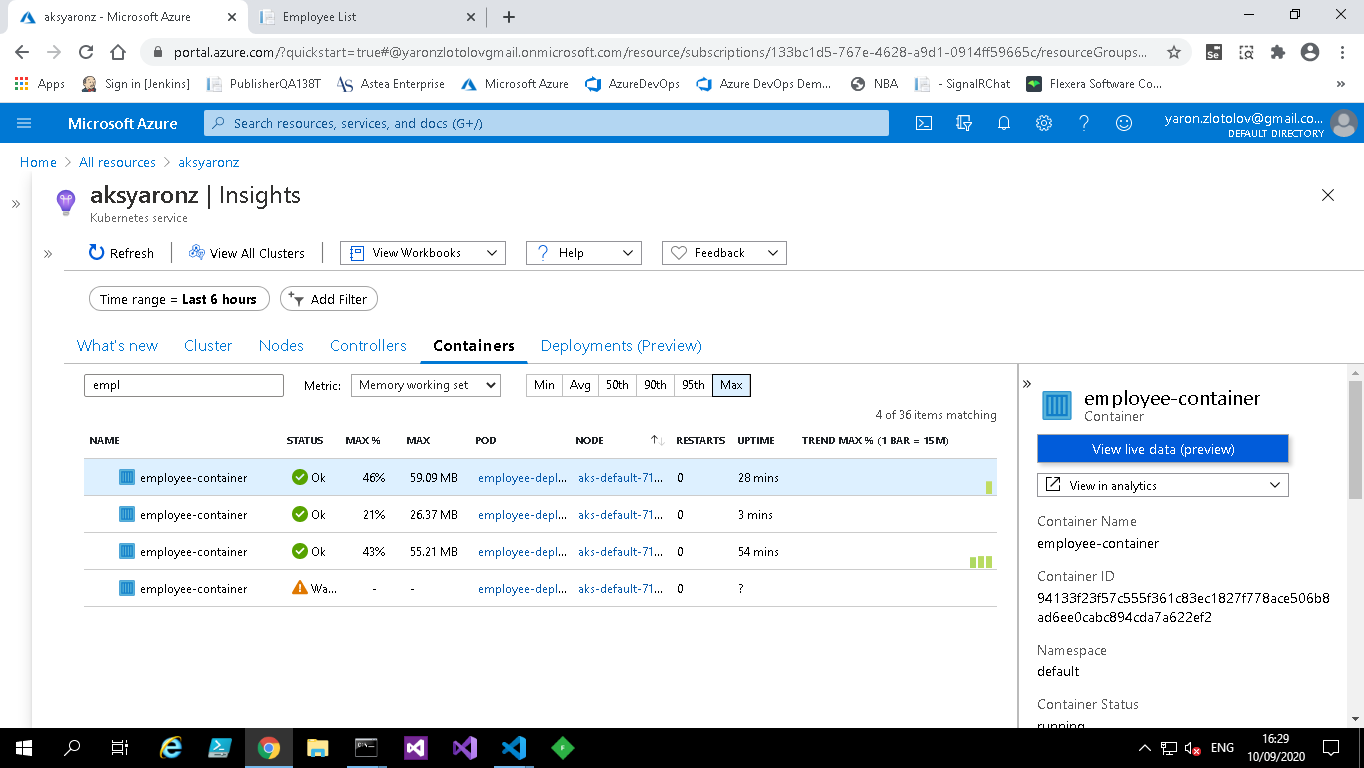
**In case of problems with Azure DevOps need to deploy manually:**

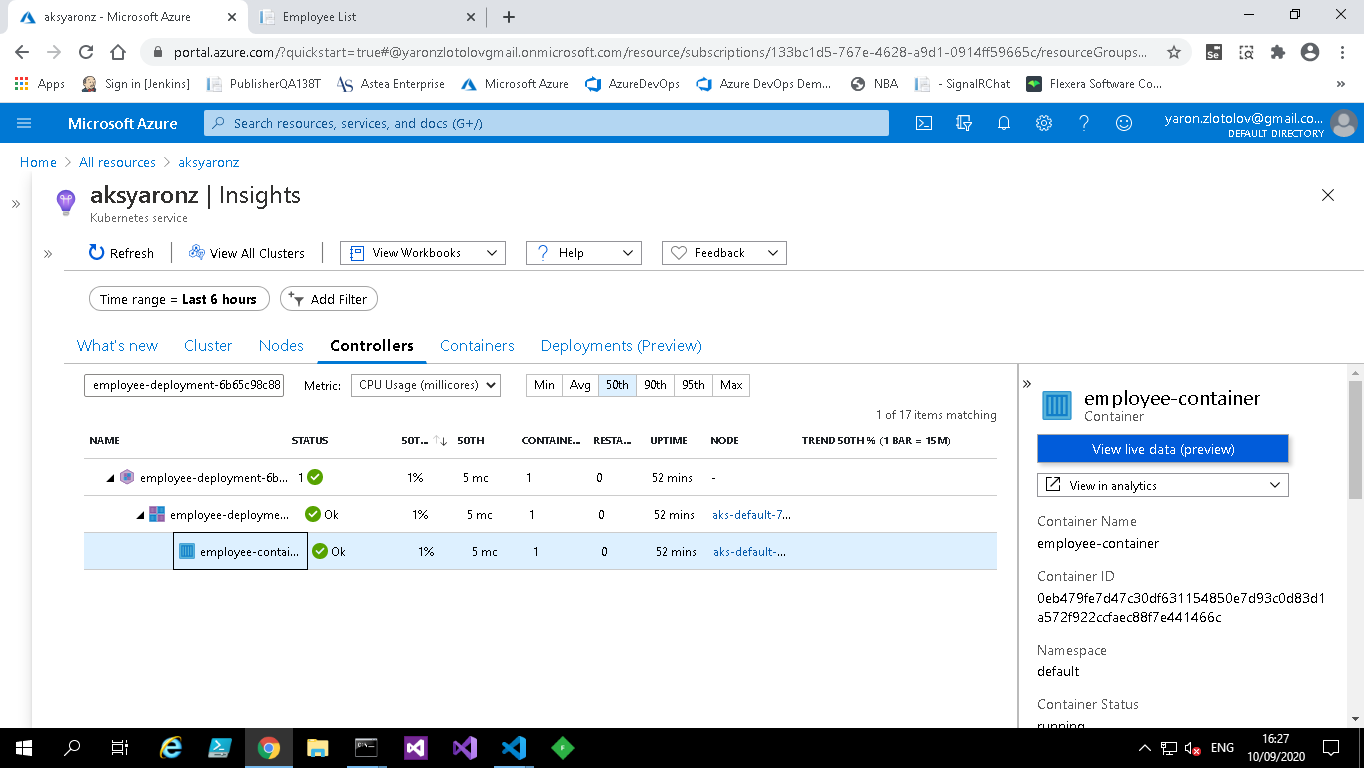


**Web application is running on Azure:**



**Monitoring AKS Memory and CPU usage:**





**Done -** Set the lock and the IP restriction again to AKS