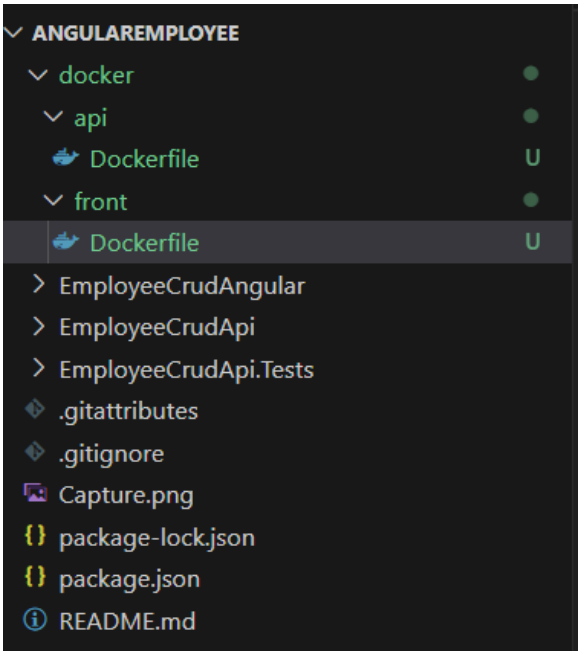
Trabajo Práctico 8 - Implementación de Contenedores en Azure y Automatización con Azure CLI

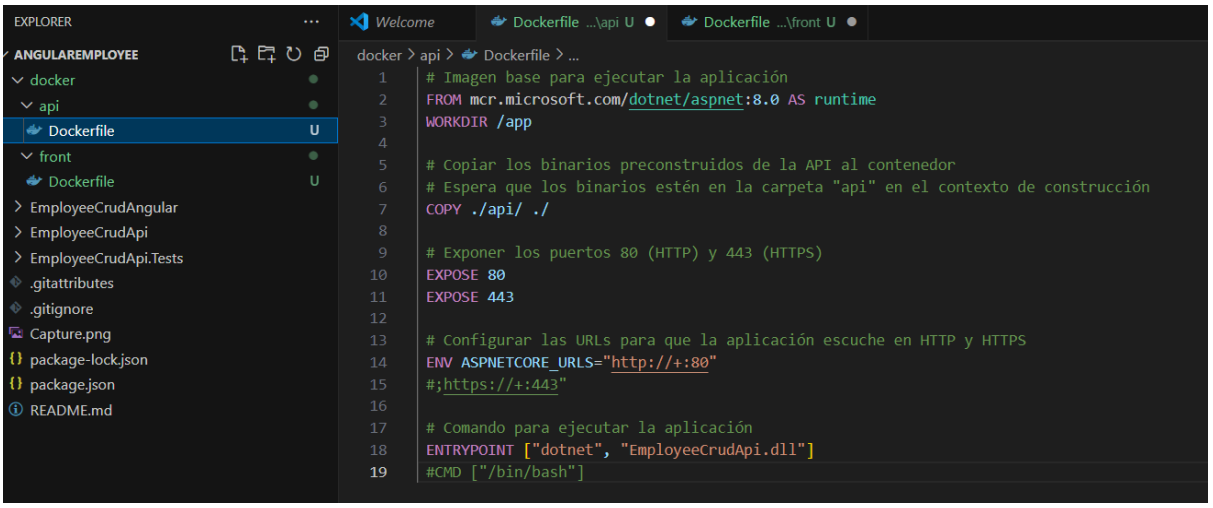
- Azure CLI instalado:

```
C:\Users\Manu>az

Welcome to Azure CLI!
---------------------
Use `az -h` to see available commands or go to https://aka.ms/cli.

Telemetry
---------
The Azure CLI collects usage data in order to improve your experience.
The data is anonymous and does not include commandline argument values.
The data is collected by Microsoft.

You can change your telemetry settings with `az configure`.


     /\
    /  \    _____   _ _ ___ _
   / /\ \  |_  / | | | \'__/ _\
  / ____ \  / /| |_| | | |  __/
 /_/    \_\/___|\__,_|_|  \___|


Welcome to the cool new Azure CLI!

Use `az --version` to display the current version.
Here are the base commands:
```

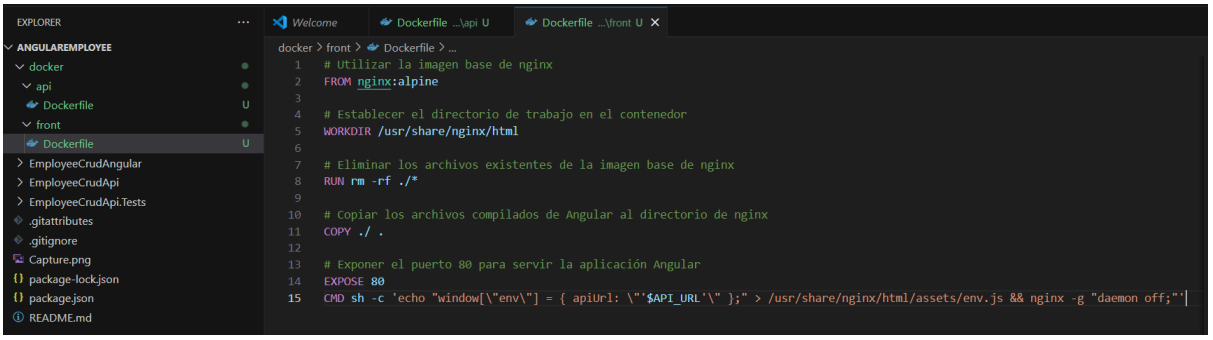
8.1 Se crea un directorio "docker" dentro del directorio raíz del proyecto que contiene 2 carpetas (api y front), y dentro de cada una de ellas se crean los correspondientes Dockerfile como se muestra a continuación.



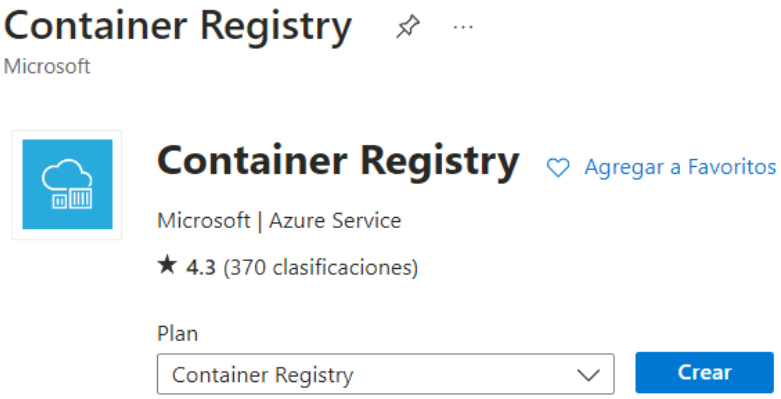
Dockerfile Back:



Dockerfile Front:



- Crear un recurso ACR en Azure Portal: *(Siguiendo el instructivo)*



Detalles del proyecto

Suscripción * — Azure subscription 1

Grupo de recursos * — TP5IngSoft3UCC2024

Crear nuevo

Detalles de instancia

Nombre del Registro * — MRMIngSoft3UCCACR

.azurecr.io

Ubicación * — Brazil South

Uso de zonas de disponibilidad

Las zonas de disponibilidad se activan en registros premium y en regiones que admiten zonas de disponibilidad. Más información

Plan de precios * — Básico

Crear Registro de contenedor

Ejecutando la validación final

Datos básicos | Redes | Cifrado | Etiquetas | **Revisar y crear**

Detalles del registro

Datos básicos

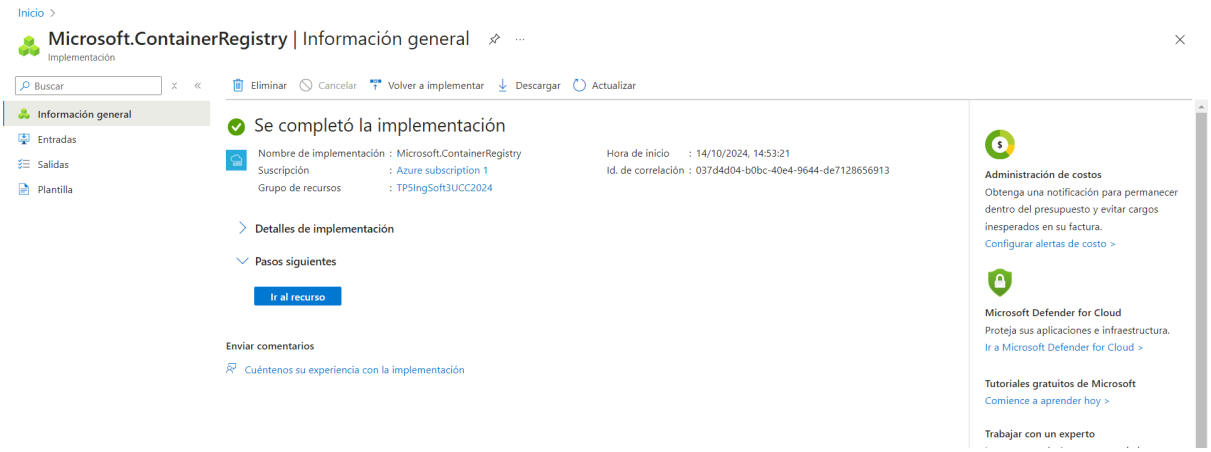
Nombre del Registro	MRMIngSoft3UCCACR
Suscripción	Azure subscription 1
Grupo de recursos	TP5IngSoft3UCC2024
Ubicación	Brazil South
Zonas de disponibilidad	Deshabilitado
Plan de precios	Basic

Redes

Acceso de red pública	Sí

Cifrado

Clave administrada por el cliente	Deshabilitado



Copiar la url de nuestro recurso ACR: ***mrmingsoft3uccacr.azurecr.io***

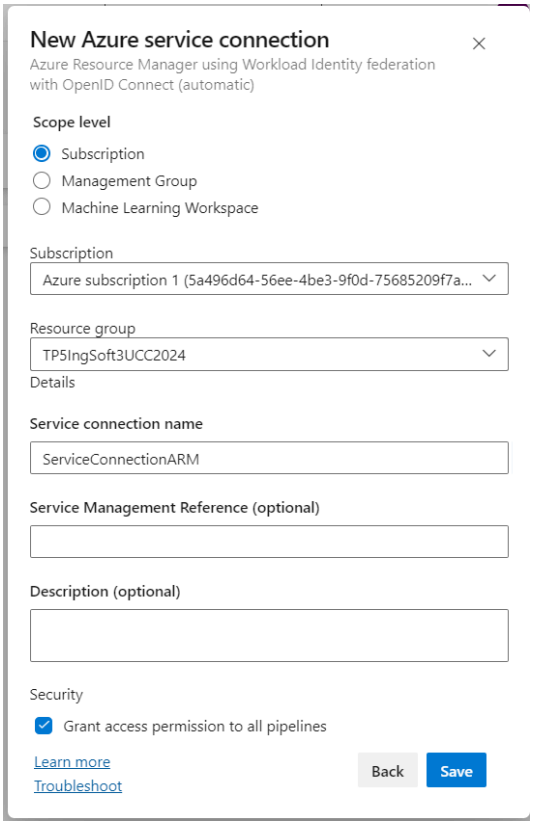
- Modificar nuestro pipeline en la etapa de Build y Test
    - Luego de la tarea de publicación de los artefactos de **Back** agregar la tarea de publicación de nuestro dockerfile de back para que esté disponible en etapas posteriores:

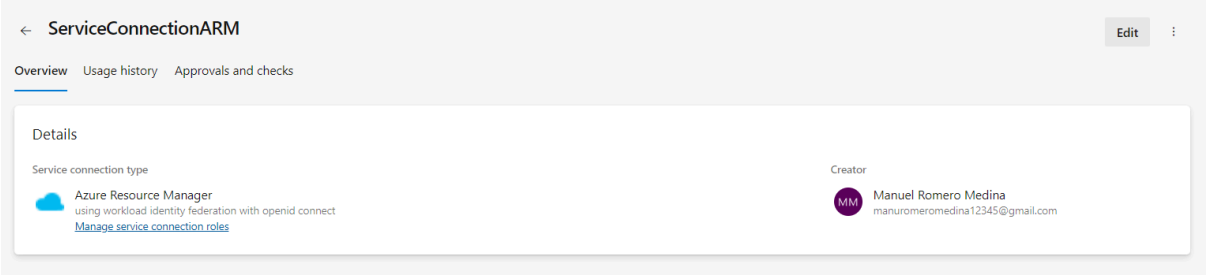
```yaml
- task: PublishBuildArtifacts@1
  displayName: 'Publicar artefactos de compilación'
  inputs:
    PathtoPublish: '$(Build.ArtifactStagingDirectory)'
    ArtifactName: 'api-drop'
    publishLocation: 'Container'

- task: PublishPipelineArtifact@1
  displayName: 'Publicar Dockerfile de Back'
  inputs:
    targetPath: '$(Build.SourcesDirectory)/docker/api/dockerfile'
    artifact: 'dockerfile-back'

- job: BuildAngular
  displayName: "Build and Test Angular"
  pool:
    vmImage: 'ubuntu-latest'
  steps:
  - task: NodeTool@0
    displayName: 'Instalar Node.js'
    inputs:
      versionSpec: '22.x'
```

    - Luego de la tarea de publicación de los artefactos de **Front** agregar la tarea de publicación de nuestro dockerfile de front para que esté disponible en etapas posteriores:

```yaml
- task: PublishBuildArtifacts@1
  displayName: 'Publicar artefactos Angular'
  inputs:
    PathtoPublish: '$(frontPath)/dist'
    ArtifactName: 'front-drop'


- task: PublishPipelineArtifact@1
  displayName: 'Publicar Dockerfile de Back'
  inputs:
    targetPath: '$(Build.SourcesDirectory)/docker/front/dockerfile'
    artifact: 'dockerfile-front'
```

En caso de no contar en nuestro proyecto con una ServiceConnection a Azure Portal para el manejo de recursos, agregar una service connection a Azure Resource Manager como se indica en instructivo 5.2





Agregar a nuestro pipeline variables

```yaml
trigger:
- main

pool:
  vmImage: 'windows-latest'

variables:
  solution: '**/*.sln'
  buildPlatform: 'Any CPU'
  buildConfiguration: 'Release'
  frontPath: './EmployeeCrudAngular'
  backPath: './EmployeeCrudApi'
  ConnectedServiceName: 'ServiceConnectionARM' #Por ejemplo 'ServiceConnectionARM'
  acrLoginServer: 'mrmingsoft3uccacr.azurecr.io' #Por ejemplo 'ascontainerregistry.azurecr.io'
  backImageName: 'employee-crud-api' #Por ejemplo 'employee-crud-api'
```

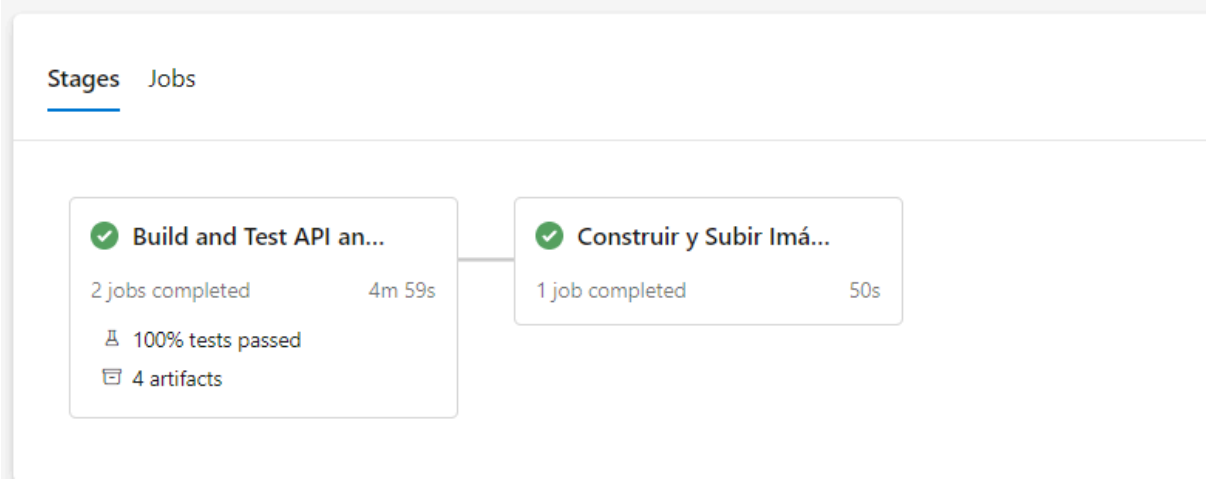
Agregar a nuestro pipeline una nueva etapa que dependa de nuestra etapa de Build y Test

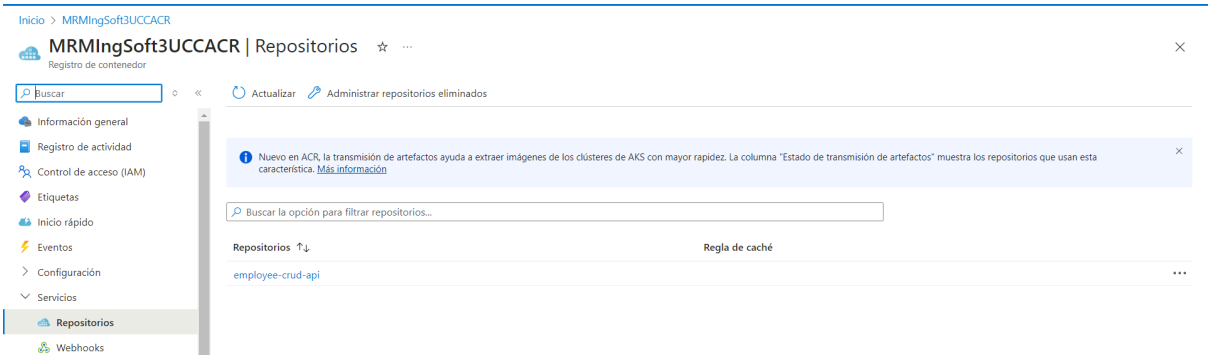
- Agregar tareas para generar imagen Docker de Back:

main / TP07-Angular / azure-pipelines.yml *

```yaml
# #-------------------------------------------------------
# ### STAGE BUILD DOCKER IMAGES Y PUSH A AZURE CONTAINER REGISTRY
# #-------------------------------------------------------

- stage: DockerBuildAndPush
  displayName: 'Construir y Subir Imágenes Docker a ACR'
  dependsOn: BuildAndTest #NOMBRE DE NUESTRA ETAPA DE BUILD Y TEST
  jobs:
    - job: docker_build_and_push
      displayName: 'Construir y Subir Imágenes Docker a ACR'
      pool:
        vmImage: 'ubuntu-latest'

      steps:
        - checkout: self

        #-------------------------------------------------------
        # BUILD DOCKER BACK IMAGE Y PUSH A AZURE CONTAINER REGISTRY
        #-------------------------------------------------------

        Settings
        - task: DownloadPipelineArtifact@2
          displayName: 'Descargar Artefactos de Back'
          inputs:
            buildType: 'current'
            artifactName: 'drop-back'
            targetPath: '$(Pipeline.Workspace)/drop-back'

        Settings
        - task: DownloadPipelineArtifact@2
          displayName: 'Descargar Dockerfile de Back'
          inputs:
            buildType: 'current'
            artifactName: 'dockerfile-back'
            targetPath: '$(Pipeline.Workspace)/dockerfile-back'
```

Ejecutar el pipeline y en Azure Portal acceder a la opción Repositorios de nuestro recurso Azure Container Registry. Verificar que exista una imagen con el nombre especificado en la variable backImageName asignada en nuestro pipeline.

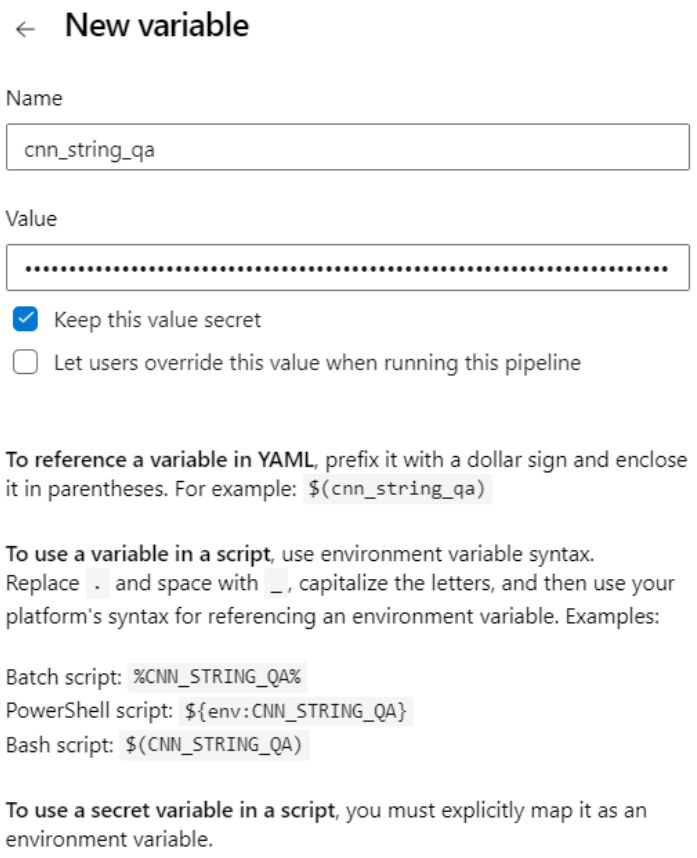




Agregar a nuestro pipeline una nueva etapa que dependa de nuestra etapa de Construcción de Imagenes Docker y subida a ACR

- Agregar variables a nuestro pipeline:

ResourceGroupName: 'NOMBRE_GRUPO_RECURSOS' #Por ejemplo 'TPS_INGSOFT3_UCC'
backContainerInstanceNameQA: 'NOMBRE_CONTAINER_BACK_QA' #Por ejemplo 'as-crud-api-qa'
backImageTag: 'latest'
container-cpu-api-qa: 1 #CPUS de nuestro container de QA
container-memory-api-qa: 1.5 #RAM de nuestro container de QA



Server=tcp:mrmsql.database.windows.net,1433;Initial Catalog=mrmsql;Persist Security Info=False;User ID=sqladmin;Password=Manuromero1;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;

```yaml
  frontImageName: 'employee-crud-front'
  ResourceGroupName: 'TP5IngSoft3UCC2024' #Por ejemplo 'TPS_INGSOFT3_UCC'
  backContainerInstanceNameQA: 'mrm-crud-api-qa' #Por ejemplo 'as-crud-api-qa'
  backImageTag: 'latest'
  container-cpu-api-qa: 1 #CPUS de nuestro container de QA
  container-memory-api-qa: 1.5 #RAM de nuestro container de QA
```

Tenemos que modificar program.cs para que lea la variable de entorno y use esa cadena de conexión

```csharp
// Obtener la cadena de conexión de una variable de entorno o de la configuración
var connectionString = Environment.GetEnvironmentVariable("cnn_string_qa")
                       ?? builder.Configuration.GetConnectionString("DefaultConnection");

if (string.IsNullOrEmpty(connectionString))
{
    throw new InvalidOperationException("La cadena de conexión no está configurada.");
}

builder.Services.AddDbContext<ApplicationDbContext>(options =>
    options.UseSqlServer(connectionString));
```

- Agregar tareas para crear un recurso Azure Container Instances que levante un contenedor con nuestra imagen de back

```yaml
#-------------------------------------------------------
### STAGE DEPLOY TO ACI QA
#-------------------------------------------------------

- stage: DeployToACIQA
  displayName: 'Desplegar en Azure Container Instances (ACI) QA'
  dependsOn: DockerBuildAndPush
  jobs:
    - job: deploy_to_aci_qa
      displayName: 'Desplegar en Azure Container Instances (ACI) QA'
      pool:
        vmImage: 'ubuntu-latest'

      steps:
        #-------------------------------------------------------
        # DEPLOY DOCKER BACK IMAGE A AZURE CONTAINER INSTANCES QA
        #-------------------------------------------------------

        Settings
        - task: AzureCLI@2
          displayName: 'Desplegar Imagen Docker de Back en ACI QA'
          inputs:
            azureSubscription: '$(ConnectedServiceName)'
            scriptType: bash
            scriptLocation: inlineScript
            inlineScript: |
              echo "Resource Group: $(ResourceGroupName)"
              echo "Container Instance Name: $(backContainerInstanceNameQA)"
              echo "ACR Login Server: $(acrLoginServer)"
              echo "Image Name: $(backImageName)"
              echo "Image Tag: $(backImageTag)"
              echo "Connection String: $(cnn-string-qa)"

              az container delete --resource-group $(ResourceGroupName) --name $(backContainerInstanceNameQA) --yes

              az container create --resource-group $(ResourceGroupName) \
                --name $(backContainerInstanceNameQA) \
```

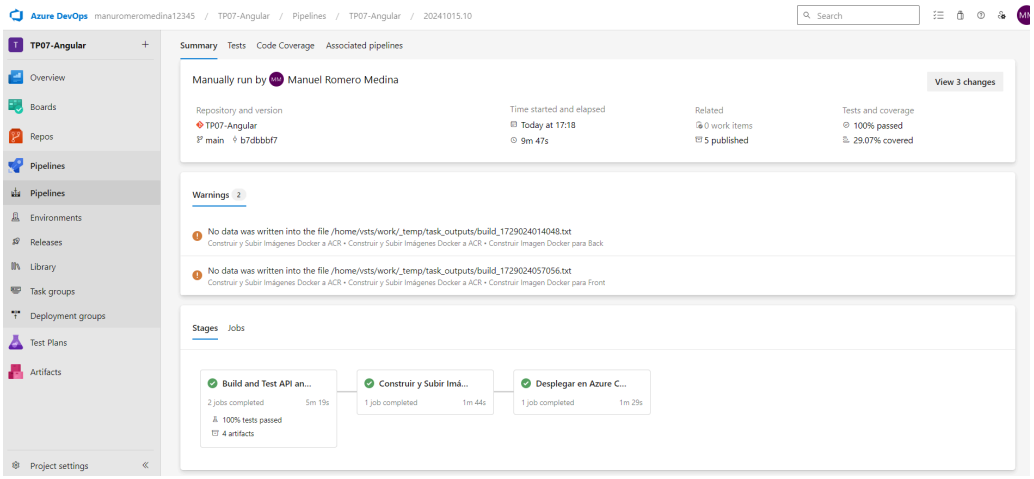
Para que funcione, hay que habilitar el acceso administrativo de ACR desde azure CLI

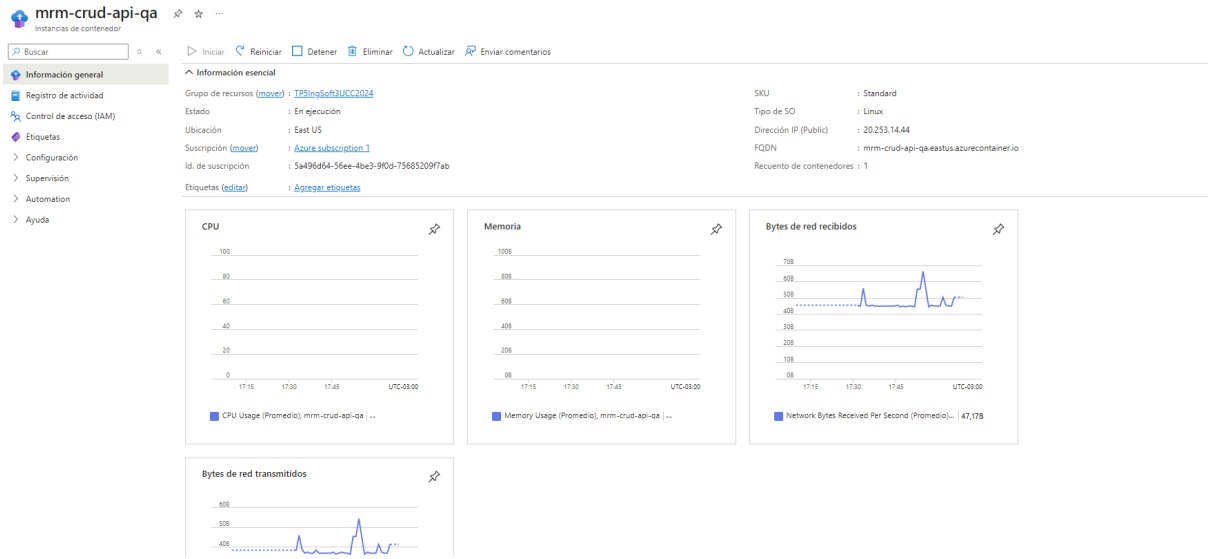
```
PS C:\Users\Manu> az acr update -n MRMIngSoft3UCCACR --admin-enabled true
>>
{
  "adminUserEnabled": true,
  "anonymousPullEnabled": false,
  "creationDate": "2024-10-14T17:53:22.687314+00:00",
  "dataEndpointEnabled": false,
  "dataEndpointHostNames": [],
  "encryption": {
    "keyVaultProperties": null,
    "status": "disabled"
  },
  "id": "/subscriptions/5a496d64-56ee-4be3-9f0d-75685209f7ab/resourceGroups/TP5IngSoft3UCC2024/providers/Microsoft.ContainerRegistry/registries/MRMIngSoft3UCCACR",
  "identity": null,
  "location": "brazilsouth",
  "loginServer": "mrmingsoft3uccacr.azurecr.io",
  "metadataSearch": "Disabled",
  "name": "MRMIngSoft3UCCACR",
  "networkRuleBypassOptions": "AzureServices",
  "networkRuleSet": null,
```

Además, debemos registrar el proveedor de recursos Microsoft.ContainerInstance, que es necesario para crear instancias de contenedores en Azure.

```
PS C:\Users\Manu> az provider register --namespace Microsoft.ContainerInstance
>>
Registering is still on-going. You can monitor using 'az provider show -n Microsoft.ContainerInstance'
```

8.10 - Ejecutar el pipeline y en Azure Portal acceder al recurso de Azure Container Instances creado. Copiar la url del contenedor y navegarlo desde browser. Verificar que traiga datos.





mrm-crud-api-qa.eastus.azurecontainer.io

Create or update Container Group	Correcto	hace 9 minu...	Tue Oct 15 ...	Azure subscription 1	manuromeromedina1234...
Create or update Container Group	Iniciado	hace 10 min...	Tue Oct 15 ...	Azure subscription 1	manuromeromedina1234...
Create or update Container Group	Accepted	hace 10 min...	Tue Oct 15 ...	Azure subscription 1	manuromeromedina1234...
Get Container Groups	Running	hace 10 min...	Tue Oct 15 ...	Azure subscription 1	manuromeromedina1234...
Get Container Groups	Running	hace 9 minu...	Tue Oct 15 ...	Azure subscription 1	manuromeromedina1234...

Pretty-print


```
[
  {
    "id": 1,
    "name": "Juan Perez",
    "createdDate": "2024-09-05T00:00:00"
  },
  {
    "id": 2,
    "name": "Carla Ruiz",
    "createdDate": "2024-09-05T22:22:47.440572"
  },
  {
    "id": 3,
    "name": "Carlos Gomez",
    "createdDate": "2024-09-06T09:16:50.070943"
  },
  {
    "id": 4,
    "name": "Joaquin Zarate",
    "createdDate": "2024-09-06T09:19:37.898716"
  },
  {
    "id": 5,
    "name": "Luis Rodriguez",
    "createdDate": "2024-09-06T09:23:31.324434"
  }
]
```

DESAFÍOS

1. Agregar tareas para generar imagen Docker de Front

Agregar tareas para generar imagen Docker de Front (DESAFIO)

- **A la etapa creada en 4.1.6 Agregar tareas para generar imagen Docker de Front**

```
backImageName: 'employee-crud-api' #Por
frontImageName: 'employee-crud-front'
```

```
#-----------------------------------------------------------
# BUILD DOCKER FRONT IMAGE Y PUSH A AZURE CONTAINER REGISTRY
#-----------------------------------------------------------

Settings
- task: DownloadPipelineArtifact@2
  displayName: 'Descargar Artefactos de Front'
  inputs:
    buildType: 'current'
    artifactName: 'front-drop'
    targetPath: '$(Pipeline.Workspace)/drop-front'

Settings
- task: DownloadPipelineArtifact@2
  displayName: 'Descargar Dockerfile de Front'
  inputs:
    buildType: 'current'
    artifactName: 'dockerfile-front'
    targetPath: '$(Pipeline.Workspace)/dockerfile-front'

Settings
- task: Docker@2
  displayName: 'Construir Imagen Docker para Front'
  inputs:
    command: build
    repository: $(acrLoginServer)/employee-crud-front # Cambia el nombre de la imagen según sea necesario
    dockerfile: $(Pipeline.Workspace)/dockerfile-front/Dockerfile
    buildContext: $(Pipeline.Workspace)/drop-front
    tags: 'latest'

Settings
- task: Docker@2
  displayName: 'Subir Imagen Docker de Front a ACR'
  inputs:
    command: push
    repository: $(acrLoginServer)/employee-crud-front # Cambia el nombre de la imagen según sea necesario
    tags: 'latest'
```

Jobs in run #20241014.24

- Build and Test API 2m 32s
- Build and Test Ang... 2m 9s

Construir y Subir Imágenes Docker a A...

- Construir y Subir I... 1m 31s
  - Initialize job 8s
  - Checkout TP07-Ang... 4s
  - Descargar Artefactos... 6s
  - Descargar Dockerfile... 4s
  - Iniciar Sesión en Az... 15s
  - Construir Imagen Do... 6s
  - Subir Imagen Dock... 13s
  - Descargar Artefactos... 5s
  - Descargar Dockerfile... 5s
  - Construir Imagen Do... 5s
  - Subir Imagen Dock... 15s
  - Post-job: Checkout... <1s
  - Finalize Job <1s

Construir Imagen Docker para Front

```
#2 DONE 0.0s

#3 [internal] load .dockerignore
#3 transferring context: 2B done
#3 DONE 0.0s

#4 [internal] load build context
#4 transferring context: 1.65MB 0.0s done
#4 DONE 0.0s

#5 [1/4] FROM docker.io/library/nginx:alpine
#5 DONE 0.1s

#6 [2/4] WORKDIR /usr/share/nginx/html
#6 DONE 0.0s

#7 [3/4] RUN rm -rf ./*
#7 DONE 0.2s

#8 [4/4] COPY ./ .
#8 DONE 0.0s

#9 exporting to image
#9 exporting layers
#9 exporting layers 1.0s done
#9 writing image sha256:108267c243b578344250feba2f969e0f2f28b585c7b4392b291c87feb6aea52d done
#9 naming to mrmingsoft3uccacr.azurecr.io/employee-crud-front:latest done
#9 DONE 1.0s
##[warning]No data was written into the file /home/vsts/work/_temp/task_outputs/build_1728944005056.txt
Finishing: Construir Imagen Docker para Front
```

Inicio > MRMIngSoft3UCCACR

MRMIngSoft3UCCACR | Repositorios

Registro de contenedor

Buscar

- Información general
- Registro de actividad
- Control de acceso (IAM)
- Etiquetas
- Inicio rápido
- Eventos
- Configuración
- Servicios
  - Repositorios
  - Webhooks
  - Replicaciones

Actualizar | Administrar repositorios eliminados

Nuevo en ACR, la transmisión de artefactos ayuda a extraer imágenes de los clústeres de AKS con mayor rapidez. La columna "Estado de transmisión de artefacto característica. Más información

Buscar la opción para filtrar repositorios...

Repositorios	Regla de caché
employee-crud-api	
employee-crud-front	

2. **Agregar tareas para generar en Azure Container Instances un contenedor de imagen Docker de Front. (Complemento con 4.1.11)**

- Creamos la variable

← Update variable

Name

API_URL

Value

mrm-crud-api-qa.eastus.azurecontainer.io/api/employee/getall

☐ Keep this value secret

☐ Let users override this value when running this pipeline

To reference a variable in YAML, prefix it with a dollar sign and enclose it in parentheses. For example: `$(API_URL)`

To use a variable in a script, use environment variable syntax. Replace `.` and space with `_`, capitalize the letters, and then use your platform's syntax for referencing an environment variable. Examples:

Batch script: `%API_URL%`
PowerShell script: `${env:API_URL}`
Bash script: `$(API_URL)`

mrm-crud-api-qa.eastus.azurecontainer.io/api/employee/getall

- Instalamos lo necesario

```
C:\Users\Manu\Desktop\INGENIERÍA EN SISTEMAS UCC\CUARTO AÑO\ING SOFT 3\ingsoft3\TP8\AngularEmployee\EmployeeCrudAngular>npm install @angular/cli --save-dev
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much m
ore comprehensive and powerful.
npm warn deprecated read-package-json@7.0.1: This package is no longer supported. Please use @npmcli/package-json instead.
npm warn deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported

added 1050 packages, and audited 1051 packages in 52s

132 packages are looking for funding
  run `npm fund` for details

4 low severity vulnerabilities

To address all issues, run:
  npm audit fix
```

- Modificamos el código del FrontEnd

TS environment.ts M × | TS environment.prod.ts ●

EmployeeCrudAngular > src > environments > TS environment.ts > environment > apiUrl

```
// This file can be replaced during build by using the `fileReplacements` array.
// `ng build --prod` replaces `environment.ts` with `environment.prod.ts`.
// The list of file replacements can be found in `angular.json`.

export const environment = {
  production: false,
  apiUrl: 'mrm-crud-api-qa.eastus.azurecontainer.io/api/employee'  // URL de la API para desarrollo
};


/*
 * For easier debugging in development mode, you can import the following file
 * to ignore zone related error stack frames such as `zone.run`, `zoneDelegate.invokeTask`.
 *
 * This import should be commented out in production mode because it will have a negative impact
 * on performance if an error is thrown.
 */
// import 'zone.js/dist/zone-error';  // Included with Angular CLI.
```

TS environment.ts M | TS environment.prod.ts M ×

EmployeeCrudAngular > src > environments > TS environment.prod.ts > ...

```
export const environment = {
  production: true,
  apiUrl: 'mrm-crud-api-qa.eastus.azurecontainer.io/api/employee'  // URL de la API para prod
};
```

test.ts

Edit

Contents History Compare Blame

```
// Este archivo es requerido por karma.conf.js y carga recursivamente todos los archivos .spec y del marco.
(window as any).__env = (window as any).__env || {};
(window as any).__env['apiUrl'] = window['env'] && window['env']['apiUrl'] || 'mrm-crud-api-qa.eastus.azurecontainer.io/api/employee'; // Valor pr

import 'zone.js/dist/zone-testing';
import { getTestBed } from '@angular/core/testing';
import { BrowserDynamicTestingModule } from '@angular/platform-browser-dynamic/testing';

// Primero, inicializa el entorno de pruebas de Angular.
getTestBed().initTestingModule({
  BrowserDynamicTestingModule,
  platformBrowserDynamicTesting(),
});

// Luego, encontraremos todas las pruebas.
const context = require.context('./', true, /\.spec\.ts$/);
context.keys().forEach(context);

```

index.html

Contents Preview Highlight changes

Committed 9c5b50d4: Updated test.ts

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>EmployeeCrudAngular</title>
    <base href="/">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" type="image/x-icon" href="favicon.ico">
  </head>
  <body>
    <script src="assets/env.js"></script>
    <script>
      console.log('Valor de API_URLs:', window['env'] && window['env'].apiUrl);
    </script>
    <app-root></app-root>
  </body>
</html>

```

TS environment.ts M | TS environment.prod.ts M | TS global.d.ts U

EmployeeCrudAngular > src > environments > TS global.d.ts > Window

```
interface Window {
    env : {
        apiUrl: string
    }
}
```

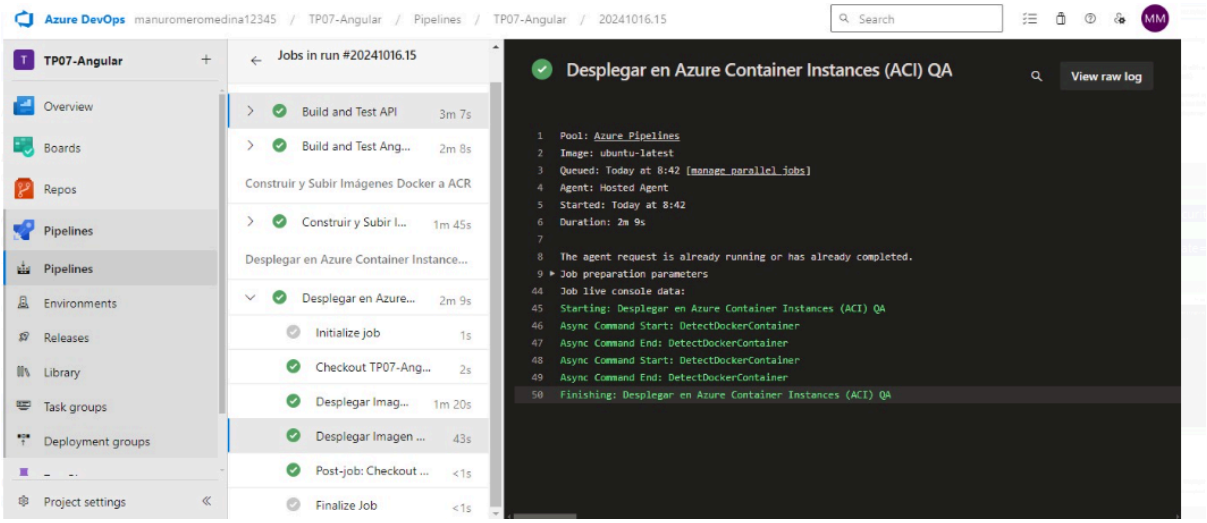
Agregamos las siguientes variables de entorno

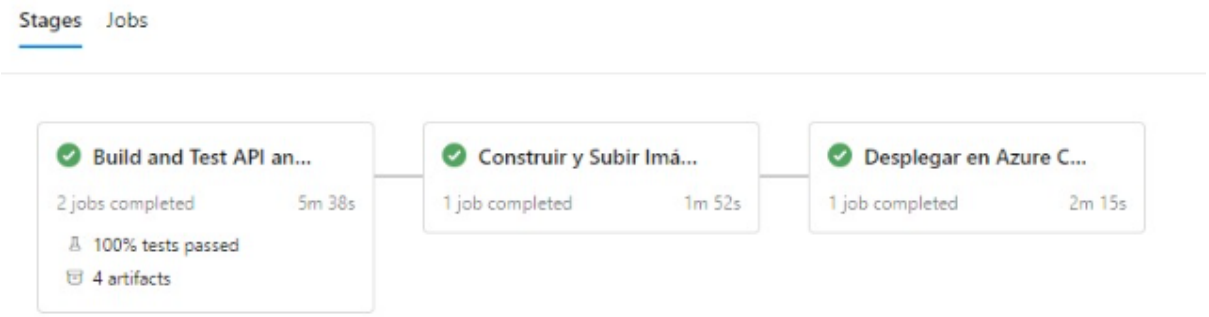
```
  frontContainerInstanceNameQA: 'mrm-crud-front-qa'  # Por ejemplo, cambiar según tu entorno
  container-cpu-front-qa: 1
  container-memory-front-qa: 1.5
  frontImageTag: 'latest'
```

Agregamos las tareas para que levante el contenedor de front

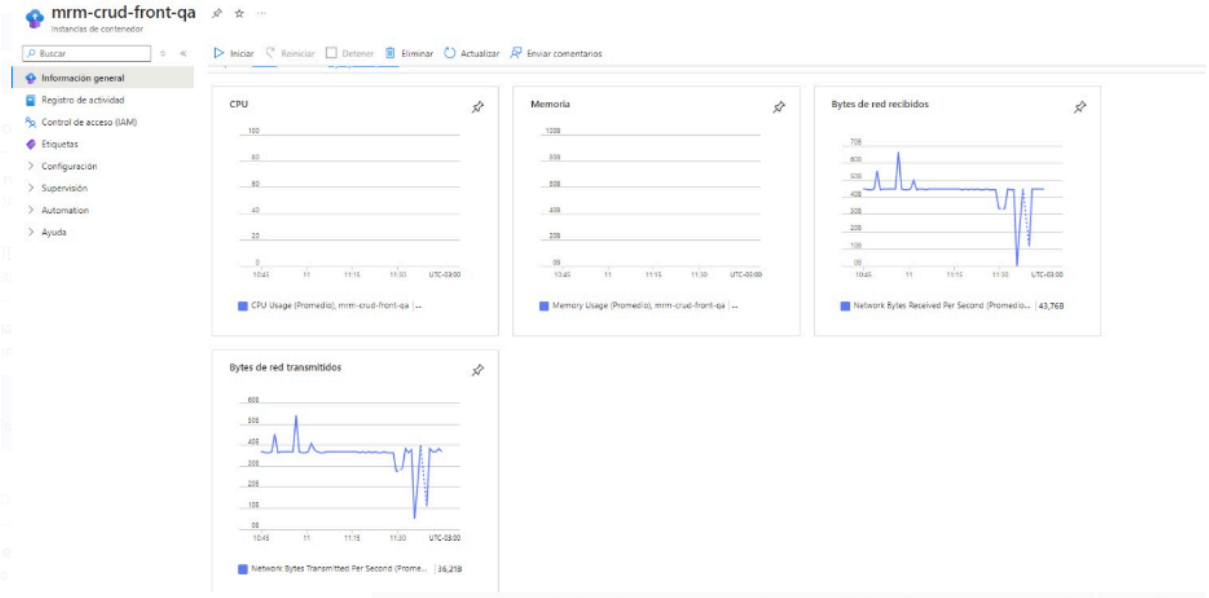
```
        #-----------------------------------------------------
        # DEPLOY DOCKER FRONT IMAGE A AZURE CONTAINER INSTANCES QA
        #-----------------------------------------------------

        Settings
        - task: AzureCLI@2
          displayName: 'Desplegar Imagen Docker de Front en ACI QA'
          inputs:
            azureSubscription: '$(ConnectedServiceName)'
            scriptType: bash
            scriptLocation: inlineScript
            inlineScript: |
              echo "Resource Group: $(ResourceGroupName)"
              echo "Container Instance Name: $(frontContainerInstanceNameQA)"  # Debes agregar la variable para el front
              echo "ACR Login Server: $(acrLoginServer)"
              echo "Image Name: $(frontImageName)"
              echo "Image Tag: $(frontImageTag)"
              echo "Api Url: $(API_URL)"

              az container delete --resource-group $(ResourceGroupName) --name $(frontContainerInstanceNameQA) --yes

              az container create --resource-group $(ResourceGroupName) \
                  --name $(frontContainerInstanceNameQA) \
                  --image $(acrLoginServer)/$(frontImageName):$(frontImageTag) \
                  --registry-login-server $(acrLoginServer) \
                  --registry-username $(acrName) \
                  --registry-password $(az acr credential show --name $(acrName) --query "passwords[0].value" -o tsv) \
                  --dns-name-label $(frontContainerInstanceNameQA) \
                  --ports 80 \
                  --environment-variables API_URL="$(API_URL)" \
                  --restart-policy Always \
                  --cpu $(container-cpu-front-qa) \
                  --memory $(container-memory-front-qa)
```

- Corremos el pipeline





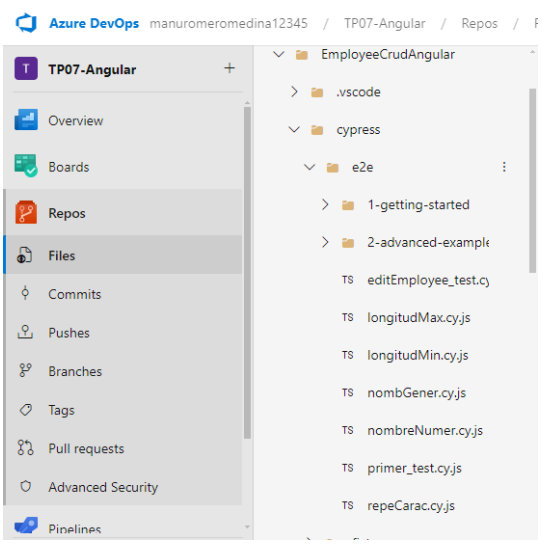
- Comprobamos



3. **Agregar tareas para correr pruebas de integración en el entorno de QA de Back y Front creado en ACI. (Complemento con 4.1.12)**
    - Modificamos el pipeline con un “Stage Cypress” (VER PORQUE NO ME EJECUTAN LAS PRUEBAS)

```yaml
# STAGE: CYPRESS TESTING
- stage: CypressTesting
  displayName: 'Pruebas E2E con Cypress'
  jobs:
    - job: IntegrationTesting
      displayName: 'Cypress'
      variables:
        baseUrl: '$(frontContainerInstanceNameQA).eastus.azurecontainer.io'
      steps:
        - script: |
            cd $(Build.SourcesDirectory)/EmployeeCrudAngular
            npm install typescript ts-node
          displayName: 'Instalar TypeScript'


        - script: |
            if not exist $(Build.SourcesDirectory)\EmployeeCrudAngular\cypress\results  mkdir "$(Build.SourcesDirectory)\EmployeeCrudAngular\cypress\results"
          displayName: 'Crear el directorio de resultados'


        - script: |
            cd $(Build.SourcesDirectory)/EmployeeCrudAngular
            npx cypress run --config-file cypress.config.ts --env baseUrl=$(baseUrl)
          displayName: 'Ejecutar Pruebas E2E con Cypress'

        Settings
        - task: PublishTestResults@2
          inputs:
            testResultsFiles: '$(Build.SourcesDirectory)/EmployeeCrudAngular/cypress/results/*.xml'
            testRunTitle: 'Pruebas E2E con Cypress (QA)'
          displayName: 'Publicar Resultados de las Pruebas de Cypress'
```

- Creamos pruebas



- Longitud máxima de caracteres:

```
Welcome | longitudMin.cy.js | repeCarac.cy.js | nombGener.cy.js | nombreNumer.cy.js | longitudMax.cy.js
EmployeeCrudAngular > cypress > e2e > longitudMax.cy.js > describe('Validación: longitud máxima de caracteres') callback > it('Debe mostrar un mensaje de
1  describe('Validación: longitud máxima de caracteres', () => {
2      it('Debe mostrar un mensaje de error cuando el nombre excede los 100 caracteres', () => {
3          const apiUrl = Cypress.env('baseUrl'); // Obtener la URL desde las variables de entorno
4
5          // Verificar si apiUrl está definido
6          if (!apiUrl) {
7              throw new Error('Error: apiUrl is not defined. Please check your environment variables.');
8          }
9
10         cy.visit(apiUrl); // Visitar la página principal
11
12         // Hacer clic en el botón que dice "Add New Employee"
13         cy.contains('Add New Employee').click();
14
15         // Continuar con la validación del formulario
16         cy.get('input[name="name"]').type('p'.repeat(101));
17         cy.get('.btn').click();
18
19         // Comprobar que se muestre el mensaje de error
20         cy.get('.toast-message').should('contain.text', 'El nombre no puede tener más de 100 caracteres.');
21     });
22 });
```

- Longitud mínima de caracteres:

```
Welcome | longitudMin.cy.js | repeCarac.cy.js | nombGener.cy.js | nombreNumer.cy.js | longitudMax.cy.js
EmployeeCrudAngular > cypress > e2e > longitudMin.cy.js > ...
1  describe('Validación: Longitud mínima del nombre', () => {
2      it('Debe mostrar un mensaje de error cuando el nombre tiene menos de 2 caracteres', () => {
3          const apiUrl = Cypress.env('baseUrl'); // Obtener la URL desde las variables de entorno
4
5          // Verificar si apiUrl está definido
6          if (!apiUrl) {
7              throw new Error('Error: apiUrl is not defined. Please check your environment variables.');
8          }
9
10         cy.visit(apiUrl); // Visitar la página principal
11
12         // Hacer clic en el botón que dice "Add New Employee"
13         cy.get('button.btn-primary').contains('Add New Employee').click();
14
15         // Continuar con la validación del formulario
16         cy.get('input[name="name"]').type('P');
17         cy.get('.btn').click();
18
19         // Comprobar que se muestre el mensaje de error
20         cy.get('.toast-message').should('contain.text', 'El nombre no puede tener menos de 2 caracteres.');
21     });
22 });
```

- Nombre con números:

```
Welcome | longitudMin.cy.js | repeCarac.cy.js | nombGener.cy.js | nombreNumer.cy.js | longitudMax.cy.js
EmployeeCrudAngular > cypress > e2e > nombreNumer.cy.js > ...
1  describe('Validación: Nombre contiene números', () => {
2      it('Debe mostrar un mensaje de error cuando el nombre contiene números', () => {
3          const baseUrl = Cypress.env('baseUrl'); // Obtener la URL desde las variables de entorno
4
5          // Verificar si baseUrl está definido
6          if (!baseUrl) {
7              throw new Error('Error: baseUrl is not defined. Please check your environment variables.');
8          }
9
10         cy.visit(baseUrl); // Visitar la página principal
11
12         // Hacer clic en el botón que dice "Add New Employee"
13         cy.get('button.btn-primary').contains('Add New Employee').click();
14
15         // Continuar con la validación del formulario
16         cy.get('input[name="name"]').type('Paulina05');
17         cy.get('.btn').click();
18
19         // Comprobar que se muestre el mensaje de error
20         cy.get('.toast-message').should('contain.text', 'El nombre no puede tener números.');
21     });
22 });
```

- Repeticiones de caracteres:

```
Welcome | longitudMin.cy.js | repeCarac.cy.js | nombGener.cy.js | nombreNumer.cy.js | longitudMax.cy.js
EmployeeCrudAngular > cypress > e2e > repeCarac.cy.js > ...
describe('Validación: Repetición de caracteres del nombre', () => {
    it('Debe mostrar un mensaje de error cuando el nombre contiene caracteres repetidos excesivamente', () => {
        const apiUrl = Cypress.env('baseUrl'); // Obtener la URL desde las variables de entorno

        // Verificar si apiUrl está definido
        if (!apiUrl) {
            throw new Error('Error: apiUrl is not defined. Please check your environment variables.');
        }

        cy.visit(apiUrl); // Visitar la página principal

        // Hacer clic en el botón que dice "Add New Employee"
        cy.get('button.btn-primary').contains('Add New Employee').click();

        // Continuar con la validación del formulario
        cy.get('input[name="name"]').type('Juuuuannnn');
        cy.get('.btn').click();

        // Comprobar que se muestre el mensaje de error
        cy.get('.toast-message').should('contain.text', 'El nombre contiene repeticiones excesivas de caracteres.');
    });
});
```

- Nombre trivial o genérico:

```
Welcome | longitudMin.cy.js | repeCarac.cy.js | nombGener.cy.js | nombreNumer.cy.js | longitudMax.cy.js
EmployeeCrudAngular > cypress > e2e > nombGener.cy.js > ...
describe('Validación: Nombre genérico o trivial', () => {
    it('Debe mostrar un mensaje de error cuando el nombre es genérico o trivial', () => {
        const apiUrl = Cypress.env('baseUrl'); // Obtener la URL desde las variables de entorno

        // Verificar si apiUrl está definido
        if (!apiUrl) {
            throw new Error('Error: apiUrl is not defined. Please check your environment variables.');
        }

        cy.visit(apiUrl); // Visitar la página principal

        // Hacer clic en el botón que dice "Add New Employee"
        cy.get('button.btn-primary').contains('Add New Employee').click();

        // Continuar con la validación del formulario
        cy.get('input[name="name"]').type('Empleado');
        cy.get('.btn').click();

        // Comprobar que se muestre el mensaje de error
        cy.get('.toast-message').should('contain.text', 'El nombre no puede ser genérico o trivial.');
    });
});
```

4. **Agregar etapa que dependa de la etapa de Deploy en ACI QA y genere contenedores en ACI para entorno de PROD.**

```
  backContainerInstanceNameProd: 'mrm-container-back-prod'
  frontContainerInstanceNameProd: 'mrm-container-front-prod'
```

BACK PROD:

```
#STAGE BACK Y FRONT PROD

- stage: DeployToACIPROD
  displayName: 'Desplegar PROD'
  dependsOn:
  - DeployToACIQA
  jobs:
  - deployment: DeployToProd
    displayName: 'Desplegar PROD'
    environment:
      name: 'Production'
    strategy:
      runOnce:
        deploy:
          steps:
            Settings
            - task: AzureCLI@2
              displayName: 'Desplegar Imagen Docker de Back en ACI (Prod)'
              inputs:
                azureSubscription: '$(ConnectedServiceName)'
                scriptType: bash
                scriptLocation: inlineScript
                inlineScript: |
                  echo "Resource Group: $(ResourceGroupName)"
                  echo "Container Instance Name: $(backContainerInstanceNameProd)"
                  echo "ACR Login Server: $(acrLoginServer)"
                  echo "Image Name: $(backImageName)"
                  echo "Image Tag: $(backImageTag)"
                  echo "Connection String: $(cnn-string-qa)"

                  az container delete --resource-group $(ResourceGroupName) --name $(backContainerInstanceNameProd) --yes

                  az container create --resource-group $(ResourceGroupName) \
                    --name $(backContainerInstanceNameProd) \
                    --image $(acrLoginServer)/$(backImageName):$(backImageTag) \
                    --registry-login-server $(acrLoginServer) \
                    --registry-username $(acrName) \
                    --registry-password $(az acr credential show --name $(acrName) --query "passwords[0].value" -o tsv) \
                    --dns-name-label $(backContainerInstanceNameProd) \
                    --ports 80 \
                    --environment-variables ConnectionStrings__DefaultConnection="$(cnn-string-qa)" \
                    --restart-policy Always \
                    --cpu $(container-cpu-api-qa) \
                    --memory $(container-memory-api-qa)
```

FRONT PROD:

```yaml
Settings
- task: AzureCLI@2
  displayName: 'Desplegar Imagen Docker de Front en ACI (PROD)'
  inputs:
    azureSubscription: '$(ConnectedServiceName)'
    scriptType: bash
    scriptLocation: inlineScript
    inlineScript: |-
      echo "Resource Group: $(ResourceGroupName)"
      echo "Container Instance Name: $(frontContainerInstanceNameProd)"
      echo "ACR Login Server: $(acrLoginServer)"
      echo "Image Name: $(frontImageName)"
      echo "Image Tag: $(frontImageTag)"
      echo "API URL: $(PROD)"

      az container delete --resource-group $(ResourceGroupName) --name $(frontContainerInstanceNameProd) --yes

      az container create --resource-group $(ResourceGroupName) \
        --name $(frontContainerInstanceNameProd) \
        --image $(acrLoginServer)/$(frontImageName):$(frontImageTag) \
        --registry-login-server $(acrLoginServer) \
        --registry-username $(acrName) \
        --registry-password $(az acr credential show --name $(acrName) --query "passwords[0].value" -o tsv) \
        --dns-name-label $(frontContainerInstanceNameProd) \
        --ports 80 \
        --environment-variables API_URL="$(PROD)" \
        --restart-policy Always \
        --cpu $(container-cpu-front-qa) \
        --memory $(container-memory-front-qa)
```