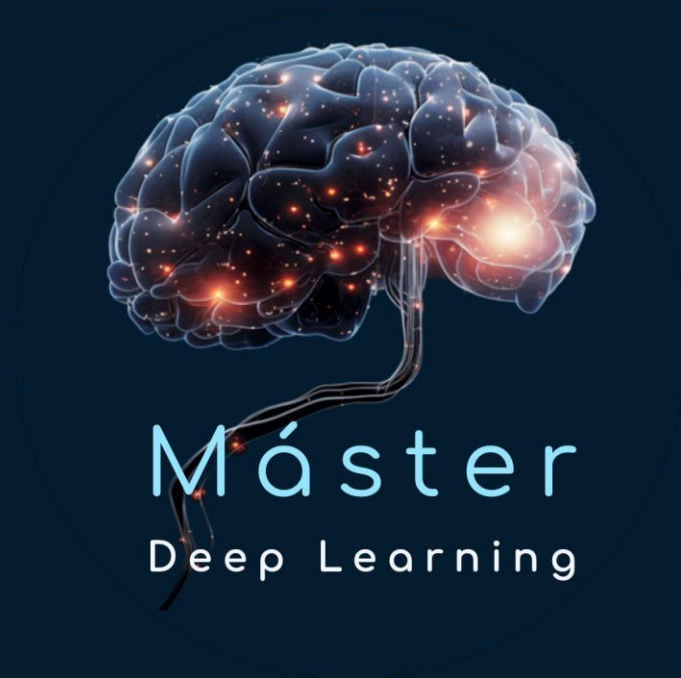


MLOps

Tema 7.

**Integración y Entrega
Continua (CI/CD)**

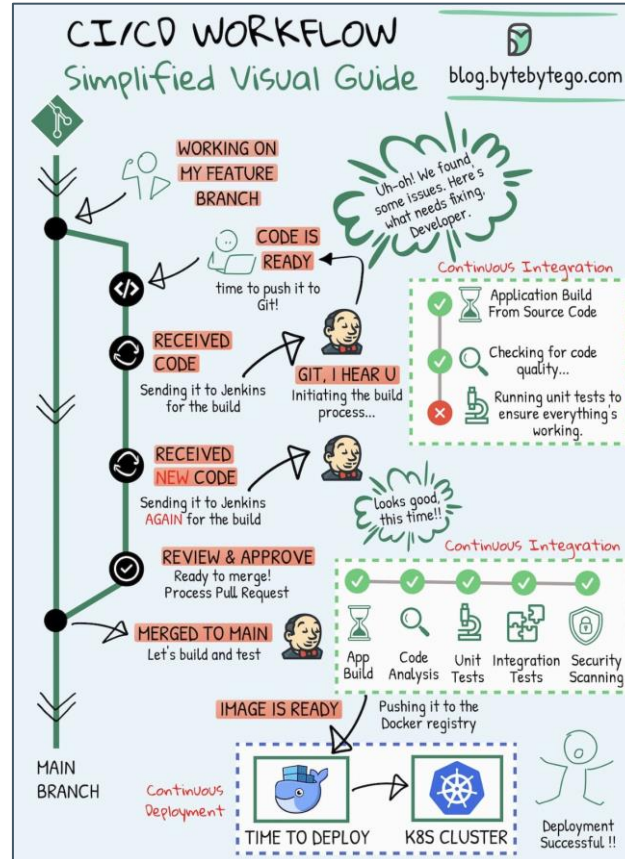


¿Qué es CI / CD?

- Es una práctica fundamental en MLOps que permite acelerar y controlar el ciclo de vida del software de manera automatizada.

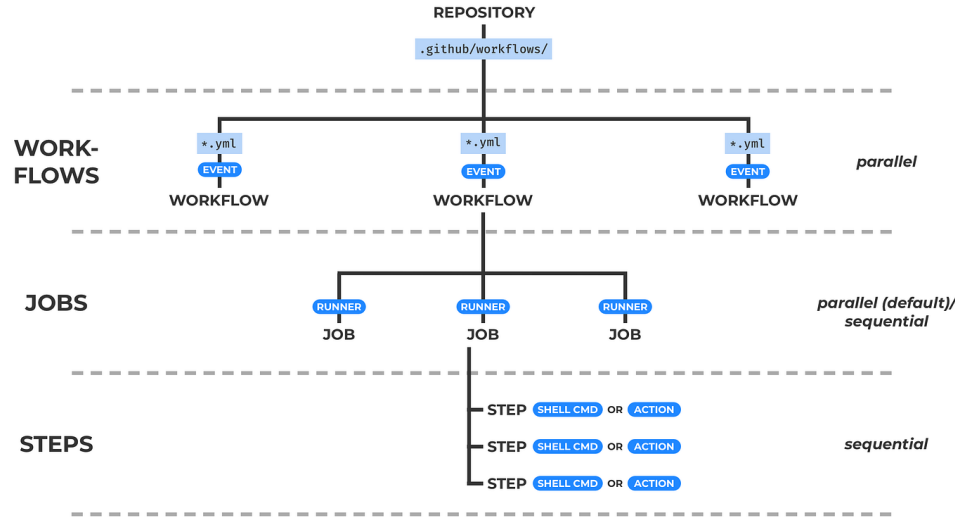


CI/CD Workflow



¿Qué es GitHub Actions?

- ▶ Sistema de automatización de GitHub
- ▶ Permite definir workflows para tareas como tests y despliegues.
- ▶ Los workflows se colocan en: `.github/workflows/*.yaml`
- ▶ Se ejecutan en eventos `push`, `pull_request`, `tag`,...



Configuración de GitHub Actions



- Funciona con cualquier repositorio GitHub
- Directorio `.github/workflows` en la raíz del proyecto
- Añadir archivos YAML con los workflows deseados
- Confirmar que la rama tenga habilitado GitHub Actions
 - Settings > Actions

Integración Continua (CI)

¿Qué es Integración Continua?



- ▶ Automatiza la ejecución de tests y builds cada vez que se hace un cambio en el código.
- ▶ Detecta errores lo antes posible.
- ▶ Verifica que todo siga funcionando tras cada commit.

*En nuestro proyecto, queremos ejecutar **test** en cada cambio que haya en el repositorio. Además, subiremos a Azure Registry una nueva imagen si **web** o **api** han cambiado su código.*

Workflow CI > Programación de pasos



```
name: CI-CD-Test, Build & Deploy to Azure
```

```
on:  
  push:  
    branches: ["master"]
```

```
jobs:  
  detect-changes:  
    runs-on: ubuntu-latest  
    outputs:  
      api: ${ steps.filter.outputs.api }  
      web: ${ steps.filter.outputs.web }  
    steps:  
      - uses: actions/checkout@v3  
  
      - name: Detect changes  
        id: filter  
        uses: dorny/paths-filter@v3  
        with:  
          filters: |  
            api:  
              - 'project/src/api/**'  
            web:  
              - 'project/src/frontend/**'
```

```
<<<<<<<<< CONTINÚA >>>>>>>>
```

Este workflow se aplicará con **push** a la rama master

Cada tarea se ejecuta en una VM de GitHub

Primera tarea > Detectar cambios

Outputs > Retorna dos booleanos (“api” y “web”). Se pueden utilizar después

Steps > Pasos secuenciales a ejecutar

1. Se copia el repositorio a la VM (checkout@v3)
2. Se detectan cambios en el push.

En “api” y “web” se guardan si hay cambios en el subdirectorio api/ y web/ respectivamente.

3. Debug. Se imprimen las variables

Workflow CI > Programación de pasos



```
test:
  runs-on: ubuntu-latest
  needs: detect-changes
  steps:
    - uses: actions/checkout@v3
    - name: Install docker-compose
      run: |
        curl -SL https://github.com/docker/compose/releases/download/v2.24.2/
$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose
        chmod +x /usr/local/bin/docker-compose
        docker-compose version
    - name: Run tests with inline env
      run: |
        WANDB_API_KEY="${{ secrets.WANDB_API_KEY }}" \
        WANDB_ARTIFACT_PATH="${{ vars.WANDB_ARTIFACT_PATH }}" \
        docker-compose -f ./project/src/docker-compose.yml up --abort-on-cont
```

```
build-api:
  runs-on: ubuntu-latest
  needs: [test, detect-changes]
  if: ${{ needs.detect-changes.outputs.api == 'true' }}
  steps:
    - uses: actions/checkout@v3
    - uses: azure/login@v1
      with:
        creds: ${{ secrets.AZURE_CREDENTIALS }}
    - name: Docker login
      run: az acr login --name ${{ vars.AZURE_ACR_NAME }}
    - name: Build & push API image
      run: |
        docker build -t ${{ vars.AZURE_ACR_NAME }}.azurecr.io/house-price-mlops:
./project/src/api
        docker push ${{ vars.AZURE_ACR_NAME }}.azurecr.io/house-price-mlops:c
```

Segunda tarea > Testear proyecto

Needs > Después de detectar cambios

Steps > Pasos secuenciales a ejecutar

1. Se copia el repositorio a la VM (checkout@v3)
2. Instalar docker-compose
3. Ejecutar docker-compose up (contiene test)

¡IMPORTANTE LAS VARIABLES DE ENTORNO!

Tercera tarea > Construir y subir imagen (API)

Needs > Después de detectar cambios y testear

If > Si se han detectado cambios en API

Steps > Pasos secuenciales a ejecutar









1. Se copia el repositorio a la VM (checkout@v3)
2. Log con Azure con credenciales
3. Log en Azure Container Registry
4. Construir imagen y push a Azure CR.

En cada push...



Se ejecutará el workflow de CI.yml

Some checks haven't completed yet
2 successful, 1 skipped, and 1 in progress checks

 	CI - Test + Build + Push to ACR / build-web (push) <i>In progress - This check has started...</i>	Details
 	CI - Test + Build + Push to ACR / detect-changes (push) <i>Successful in 4s</i>	Details
 	CI - Test + Build + Push to ACR / test (push) <i>Successful in 1m</i>	Details
 	CI - Test + Build + Push to ACR / ...	

javierpg | Repositories
Container registry

Search Refresh

Overview
Activity log
Access control (IAM)
Tags
Quick start
Resource visualizer
Events
Settings
Services
Repositories

Repositories ↑↓

- house-price-mlops ...
- test-house-price-mlops ...
- web-house-price-mlops ...

house-price-mlops ...
Repository

Refresh Start artifact streaming Manage deleted artifacts Delete repository

Essentials

Repository	Tag count
house-price-mlops	2
Last updated date	Manifest count
4/15/2025, 4:37 PM GMT+2	4

Search to filter tags ...

Tags ↑↓	Digest ↑↓	Last modified
latest	sha256:02763359354027e7ddbca98e124cefa858fb7...	4/7/2025, 5:28 PM GMT+2
ci	sha256:0719530370ff7fe2b6c939e089a30de8ced46...	4/15/2025, 4:37 PM GMT+2

Se subirán las nuevas imágenes automáticamente a nuestro Azure CR

Algunas acciones predefinidas



actions/checkout@v3

- ▶ Clona el repositorio dentro del runner
- ▶ Sin esto, el código no está disponible
- ▶ Siempre debe ir primero si necesitamos acceso al código

azure/login@v1

- ▶ Inicia sesión en Azure usando tus credenciales
- ▶ Requerido antes de hacer acciones en Azure

Referencia oficial: **GitHub Actions Marketplace**

<https://github.com/marketplace?type=actions>

Variables y Secrets en GitHub



Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Secrets Variables






Environment secrets

This environment has no secrets.

Manage environment secrets

Repository secrets

New repository secret

Name 	Last updated	
 AZURE_CREDENTIALS	yesterday	 
 WANDB_API_KEY	5 hours ago	 

Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Secrets Variables








Environment variables

This environment has no variables.

Manage environment variables

Repository variables

New repository variable

Name 	Value	Last updated	
 AZURE_ACR_NAME	javierpg	yesterday	 
 WANDB_ARTIFACT_PATH	javierpg-universidad-de-murcia/house-price-r...	5 hours ago	 

Variables y Secrets en GitHub



JAMÁS HAY QUE SUBIR CREDENCIALES AL REPOSITORIO DE GITHUB

- ▶ Creados por el admin en `Settings > Secrets and variables > Actions`
- ▶ Variables sensibles que no deben aparecer en el código
 - ▶ New repository secret → Add secret
- ▶ Variables para parametrizar información
 - ▶ New repository variable → Add variable
- ▶ Por ejemplo, en nuestro proyecto
 - ▶ Secretos
 - ◆ `WANDB_API_KEY`. Se consulta con `${{ secrets.WAND_API_KEY }}`
 - ◆ `AZURE_CREDENTIALS`. Se consulta con `${{ secrets.AZURE_CREDENTIALS }}`
 - ▶ Variables
 - ◆ `WANDB_ARTIFACT_PATH`. Se consulta con `${{ vars.WANDB_ARTIFACT_PATH }}`
 - ◆ `AZURE_ACR_NAME`. Se consulta con `${{ vars.AZURE_ACR_NAME }}`

AZURE_CREDENTIALS



- ▶ Es un **JSON** con permisos para desplegar en Azure.
- ▶ Este JSON lo generarás una sola vez y lo cargarás como secreto en GitHub.

```
$ az ad sp create-for-rbac --name "github-actions-deploy" --role contributor --scopes /subscriptions/<subscription-id>/resourceGroups/<resource-group> --sdk-auth
```

▶ <subscription-id> =

```
$ az account show --query id --output tsv
```

▶ <resource-group> =

```
$ az group list --query "[].name" --output tsv
```

```
javier [ ~ ]$ az account show --query id --output tsv
c03e99ba-d123-4e5e-beb8-416237ac097f
javier [ ~ ]$ az group list --query "[].name" --output tsv
DefaultResourceGroup-CCAN
DefaultResourceGroup-EUS
upm-mlops
DefaultResourceGroup-ESC
javier [ ~ ]$ az ad sp create-for-rbac --name "github-actions-deploy" --role contributor --scopes /subscriptions/c03e99ba-d123-4e5e-beb8-416237ac097f/resourceGroups/upm-mlops --sdk-auth
Option '--sdk-auth' has been deprecated and will be removed in a future release.
Creating 'contributor' role assignment under scope '/subscriptions/c03e99ba-d123-4e5e-beb8-416237ac097f/resourceGroups/upm-mlops'
The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check
ps://aka.ms/azadsp-cli
{
  "clientId": "834f873d-5042-4fc8-be0f-89f19a066181",
  "clientSecret": "EG[REDACTED].UcOx",
  "subscriptionId": "c03e99ba-d123-4e5e-beb8-416237ac097f",
  "tenantId": "6afea85d-c323-4270-b69d-a4fb3927c254",
  "activeDirectoryEndpointUrl": "https://login.microsoftonline.com/",
  "resourceManagerEndpointUrl": "https://management.azure.com/",
  "activeDirectoryGraphResourceId": "https://graph.windows.net/",
  "sqlManagementEndpointUrl": "https://management.core.windows.net:8443/",
  "galleryEndpointUrl": "https://gallery.azure.com/",
  "managementEndpointUrl": "https://management.core.windows.net/"
}
javier [ ~ ]$
```

Azure CLI

Actions secrets / New secret

Name *

AZURE_CREDENTIALS

Secret *

```
{
  "clientId": "834f873d-5042-4fc8-be0f-89f19a066181",
  "clientSecret": "EG[REDACTED].UcOx",
  "subscriptionId": "c03e99ba-d123-4e5e-beb8-416237ac097f",
  "tenantId": "6afea85d-c323-4270-b69d-a4fb3927c254",
  "activeDirectoryEndpointUrl": "https://login.microsoftonline.com/",
  "resourceManagerEndpointUrl": "https://management.azure.com/",
  "activeDirectoryGraphResourceId": "https://graph.windows.net/",
  "sqlManagementEndpointUrl": "https://management.core.windows.net:8443/",
  "galleryEndpointUrl": "https://gallery.azure.com/",
  "managementEndpointUrl": "https://management.core.windows.net/"
}
```

Add secret

GitHub

Despliegue Continuo (CD)

¿Qué es Despliegue Continua?



- Automatiza el paso de subir a producción
- Evita actualizar a mano el servicio online

En nuestro proyecto, queremos que automáticamente se actualice la página web de predicción de casas si se cambia correctamente el código.

Crear app multicontenedor



- ▶ Abrimos la terminal en Azure
- ▶ Subir docker-compose.prod.yml

```
$ az webapp create \  
--resource-group <your-rg> \  
--plan <your-plan> \  
--name <webapp-name> \  
--multicontainer-config-type compose \  
--multicontainer-config-file docker-compose.prod.yml
```

```
javier [ ~ ]$ az webapp create --resource-group upm-elops --plan house-prices-plan --name house-prices-cd --multicontainer-config-type compose --multicontainer-config-file docker-compose.prod.yml  
{  
  "availabilityState": "Normal",  
  "clientAffinityEnabled": true,  
  "clientCertEnabled": false,  
  "clientCertificateExclusionPaths": null,  
  "clientCertMode": "Required",  
  "cloningInfo": null,  
  "containerSize": 0,  
  "customDomainVerificationId": "31C95DD286257208823C56A895C687907CECE8E766F94C251A27855F23F059E",  
  "dailyMemoryQuota": 0,  
  "daprConfig": null,  
  "defaultHostName": "house-prices-cd.azurewebsites.net",  
  "enabled": true,  
  "enabledHostNames": [  
    "house-prices-cd.azurewebsites.net",  
    "house-prices-cd.scm.azurewebsites.net"  
  ],  
  "endOfEndEncryptionEnabled": false,  
  "extendedLocation": null,  
  "ftpPublishingUrl": "ftp://www-prod-esc-007.ftp.azurewebsites.windows.net/site/wwwroot",  
  "hostNameSslStates": [  
    {  
      "certificateResourceId": null,  
      "hostType": "Standard",  
      "ipBasedSsl": null,  
      "ipBasedSslState": "notConfigured",  
      "name": "house-prices-cd.azurewebsites.net",  
      "sslState": "Disabled",  
      "thumbprint": null,  
      "update": null,  
      "updateIpBasedSsl": null,  
      "virtualIpV6": null,  
      "virtualIp": null  
    }  
  ],  
  ...  
}
```

Workflow CD > Pasos



```
# ----- DEPLOY -----  
deploy:  
  runs-on: ubuntu-latest  
  needs: [build-api, build-web]  
  if: ${{ always() && (needs.build-api.result == 'success' ||  
needs.build-web.result == 'success') }}  
  steps:  
    - uses: actions/checkout@v3  
    - uses: azure/login@v1  
      with:  
        creds: ${{ secrets.AZURE_CREDENTIALS }}  
    - name: Prepare docker-compose.prod.yml  
      run: |  
        cd project/src  
        cp docker-compose.prod.yml docker-compose.yml  
        zip deploy.zip docker-compose.yml  
    - name: Deploy to Azure Web App  
      uses: azure/webapps-deploy@v2  
      with:  
        app-name: ${{ vars.AZURE_WEBAPP_NAME }}  
        package: ./project/src/deploy.zip
```

Última tarea > Desplegar servicio

Needs > Después de construir imágenes

If > Evaluar siempre y si alguno se construyó

Steps > Pasos secuenciales a ejecutar

1. Se copia el repositorio a la VM (checkout@v3)
2. Login en Azure con credenciales
3. Crear ZIP con código del proyecto
4. Deploy en Azure Web App

Algunas acciones predefinidas



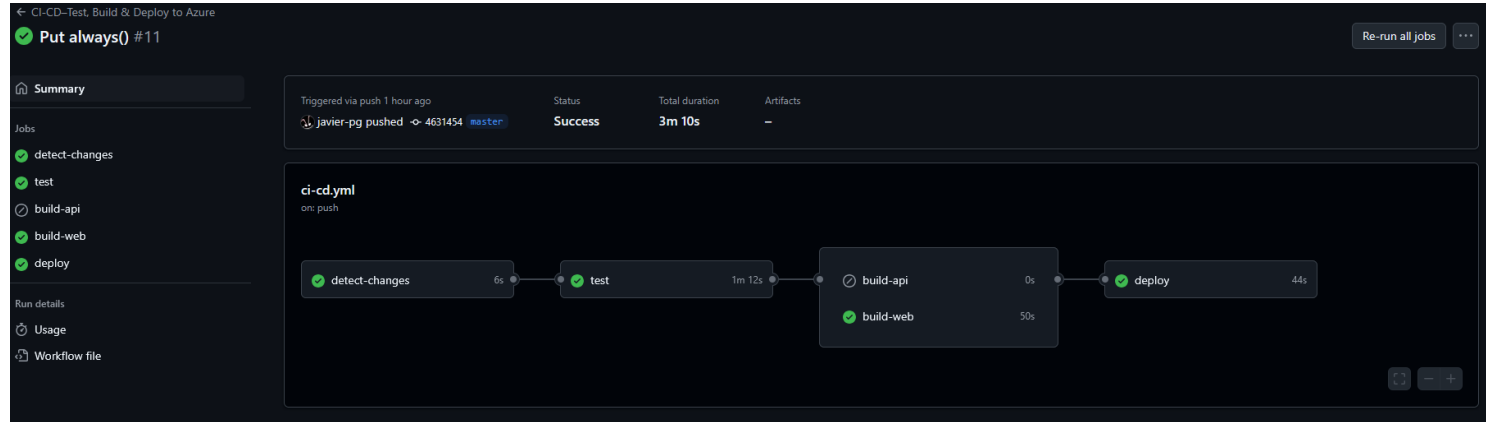
azure/webapps-deploy@v2

- ▶ Acción oficial para subir Azure App Service (Web App)
- ▶ Usa el contexto de sesión abierta por **azure/login@v1**
- ▶ Se establece el nombre de la app destino (*app-name*)
- ▶ Se envía el fichero comprimido actualizado (*package*)

En cada push...



Se ejecutará el workflow de CI-CD.yml



1. Se detectan cambios en web o api.
2. Se testea el proyecto y posibles errores.
3. Se construyen nuevas imágenes, si aplica, subiéndolos a Azure Container Registry
4. Se reinicia la aplicación con el nuevo despliegue actualizado.

Otras funcionalidades

CI y CD en workflows diferentes



```
name: CI
on:
  push: { branches: [master] }

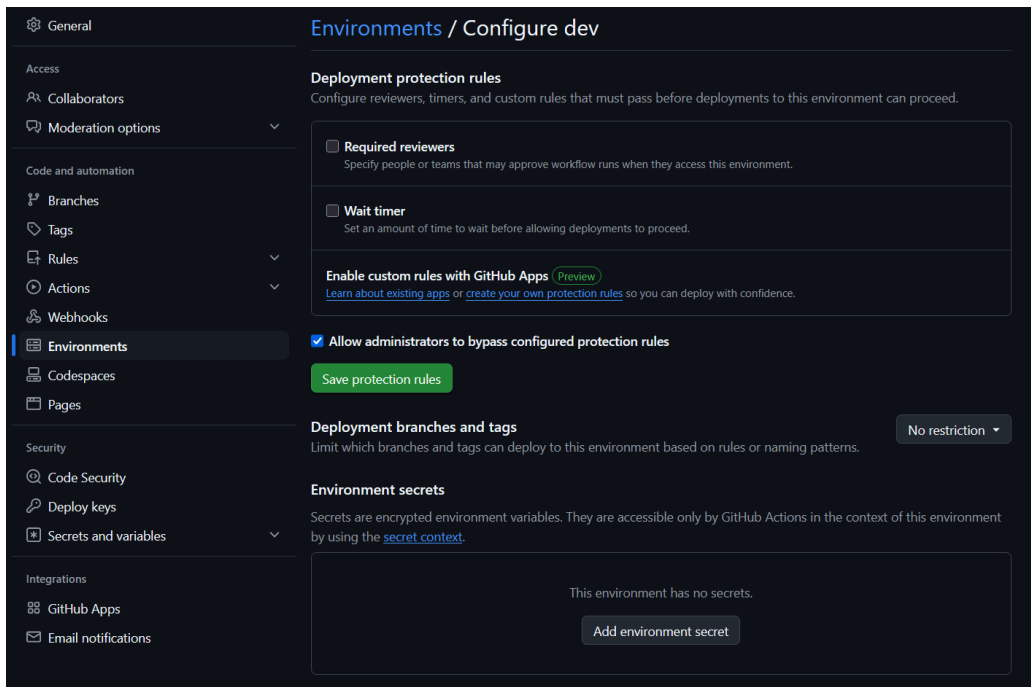
jobs:
  build-test:
  ...
```

```
name: CD
on:
  workflow_run:
    workflows: ["CI"]      # << vincula al workflow anterior
    types: [completed]

jobs:
  deploy:
  ...
```

Gestión de entornos (envs)

- ▶ Crear entorno de desarrollo y producción
 - ▶ Diferentes condiciones y variables



The screenshot shows the GitHub 'Environments / Configure dev' page. On the left is a sidebar with navigation links: General, Access, Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Environments (selected), Codespaces, Pages, Security, Code Security, Deploy keys, Secrets and variables, and Integrations, GitHub Apps, and Email notifications. The main content area is titled 'Environments / Configure dev' and contains several sections: 'Deployment protection rules' with options for 'Required reviewers', 'Wait timer', and 'Enable custom rules with GitHub Apps'; a checkbox for 'Allow administrators to bypass configured protection rules'; a 'Save protection rules' button; 'Deployment branches and tags' with a 'No restriction' dropdown; 'Environment secrets' with a description and an 'Add environment secret' button.

Gestión de entornos (envs)



- Permite condicionar tareas

```
jobs:
  deploy-dev:
    if: ${github.event.workflow_run.conclusion == 'success' }
    runs-on: ubuntu-latest
    environment: dev          # ← habilita secrets-dev
    steps:
      - uses: actions/download-artifact@v4
        with: { name: image-tag }
      - uses: azure/login@v1
        with: { creds: ${secrets.AZURE_CREDENTIALS_DEV} }
      - uses: azure/webapps-deploy@v2
        with:
          app-name: ${vars.WEBAPP_DEV}
          images:  ${vars.ACR }/app:${steps.tag.outputs.version }

  promote-prod:
    needs: deploy-dev
    environment: prod        # ← reviewers + secrets-prod
    runs-on: ubuntu-latest
    # Sólo arranca cuando el reviewer aprueba la promo
    steps:
      - uses: azure/login@v1
        with: { creds: ${secrets.AZURE_CREDENTIALS_PROD} }
      - uses: azure/container-apps-deploy@v1
        with:
          app-name: ${vars.WEBAPP_PROD}
          images:  ${needs.deploy-dev.outputs.deployed-image }
```


Referencias



- ▶ [CI/CD GitHub Actions and Azure \(2025\).](#)
- ▶ [GitHub Actions for Azure \(2025\).](#)