

# GitHub Actions

---

## ¿Qué es GitHub Actions y dónde van los workflows?

GitHub Actions es la plataforma CI/CD integrada en GitHub: automatiza tareas (construir, probar, revisar estilo, desplegar) cuando ocurre un evento (push, PR, botón manual, cron). Los workflows son archivos YAML dentro de `.github/workflows/` en tu repositorio.

[https://docs.github.com/es/actions/get-started/understand-github-actions?utm\\_source=chatgpt.com](https://docs.github.com/es/actions/get-started/understand-github-actions?utm_source=chatgpt.com)

## Crea los scripts de ejecución en tu package.json

En tu repo debe existir `package.json` con scripts como estos, si no existe, créalo:

```
{
  "scripts": {
    "lint": "eslint .",
    "format": "prettier . --check"
  }
}
```

## Crea tu primer workflow con ESLint y Prettier desde la UI

1. Abre tu repositorio en GitHub y entra a la pestaña Actions.
2. Si es tu primera vez, verás sugerencias y un botón "New workflow". También puedes elegir "set up a workflow yourself" para empezar desde cero.
3. Elige la opción "set up a workflow yourself".
4. En el editor que aparece, cambia el nombre del archivo en la parte superior a: `lint-format.yml`
5. Copia el siguiente código:

```
name: Lint & Format (ESLint + Prettier)

on:
  pull_request:
  push:
    branches: [main]

jobs:
  quality:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout repo
        uses: actions/checkout@v5

      - name: Setup Node
```

```
uses: actions/setup-node@v4
with:
  node-version: "20.x"
  cache: "npm"

- name: Install deps
  run: npm ci

- name: ESLint
  run: npm run lint

- name: Prettier (check)
  run: npm run format
```

## Ejecutarlo manualmente desde la UI (opcional)

1. Entra a Actions → elige el workflow "CI Ruff" en la barra lateral → botón "Run workflow" → selecciona la rama y confirma.
2. Abre la ejecución para ver los logs de cada paso (lint y formato).

*El botón Run workflow aparece si el workflow está en la rama por defecto y tiene workflow\_dispatch.*

## Otros ejemplos útiles

Escaneo de seguridad con CodeQL (análisis de código estático)

[https://docs.github.com/es/code-security/code-scanning/introduction-to-code-scanning/about-code-scanning-with-codeql?utm\\_source=chatgpt.com](https://docs.github.com/es/code-security/code-scanning/introduction-to-code-scanning/about-code-scanning-with-codeql?utm_source=chatgpt.com)

Publicar paquetes en PyPI al crear un tag

[https://github.com/pypa/gh-action-pypi-publish?utm\\_source=chatgpt.com](https://github.com/pypa/gh-action-pypi-publish?utm_source=chatgpt.com)

Construir & publicar imágenes Docker (Docker Hub o GHCR)

[https://docs.docker.com/build/ci/github-actions/?utm\\_source=chatgpt.com](https://docs.docker.com/build/ci/github-actions/?utm_source=chatgpt.com)

Pruebas end-to-end (Playwright/Cypress)

[https://github.com/microsoft/playwright-github-action?utm\\_source=chatgpt.com](https://github.com/microsoft/playwright-github-action?utm_source=chatgpt.com)

Tareas programadas (cron): recordatorios, scrapers, backups

Ejecuta scripts aunque no haya commits.

```
name: nightly-job
on:
  schedule:
    - cron: "0 3 * * *" # 03:00 UTC diario
```

```
jobs:
  run:
    runs-on: ubuntu-latest
    steps:
      - run: echo "Tarea nocturna $(date -u)"
```

Evento schedule y tutorial oficial. [https://docs.github.com/actions/learn-github-actions/events-that-trigger-workflows?utm\\_source=chatgpt.com](https://docs.github.com/actions/learn-github-actions/events-that-trigger-workflows?utm_source=chatgpt.com)