

Replicar el pipeline CI

1. Crear el Archivo .github/workflows/main.yml

2. Definir el Nombre del Flujo de Trabajo y los Eventos Desencadenantes

```
name: NestJS Test

on:
  push:
    branches:
      - master
  pull_request:
    branches:
      - master
```

Aca se define el nombre del flujo 'NestJS Test' y los dos eventos donde se corren los Github Actions definidos en este archivo, esto es al hacer push al 'master' y cuando se hace un pull request al 'master'.

3. Configurar el Trabajo de CI

```
jobs:
  test:
    runs-on: ubuntu-latest
```

Se define un trabajo llamado "test" que se ejecutará en una instancia de Ubuntu.

4. Configurar los Pasos del Trabajo de CI

```
steps:
  - name: Checkout Repository
    uses: actions/checkout@v3

  - name: Set up Node.js
    uses: actions/setup-node@v3
    with:
      node-version: 14

  - name: Install Dependencies
    run: npm install

  - name: Run Unit Tests
    run: npm test

  - name: Check Code Formatting with Prettier
    run: npm run prettier -- --check

  - name: TruffleHog Enterprise scan
    uses: trufflesecurity/trufflehog-enterprise-github-action@main
    with:
      args: --fail-verified ${github.event.repository.default_branch} HEAD
```

Checkout Repository: Clona el repositorio de GitHub en el entorno de CI.

Set up Node.js: Configura la versión de Node.js que se utilizará.

Install Dependencies: Instala las dependencias del proyecto con npm install.

Run Unit Tests: Ejecuta los tests de unidad con npm test.

Check Code Formatting with Prettier: Verifica el formato del código con Prettier.

Esto es lo revisado con Prettier:

```
{  
  "singleQuote": true,  
  "printWidth": 120,  
  "trailingComma": "none",  
  "tabWidth": 2,  
  "arrowParens": "avoid",  
  "jsxBracketSameLine": true  
}
```

TruffleHog Enterprise scan: Utiliza la acción de TruffleHog Enterprise para buscar secretos y problemas de seguridad en el código.