

## Estrategia de Despliegue para la Calculadora en Java

**Objetivo.** Definir una estrategia de despliegue adecuada para el proyecto de la calculadora en Java, considerando el entorno, los riesgos, la automatización y la experiencia del usuario. Elaborar un documento técnico que justifique la elección.

### Parte 1: Contexto del proyecto

**Proyecto base:** Calculadora en Java con pruebas automatizadas y pipeline CI configurado en GitHub Actions.

**Supuesto:** El proyecto está listo para ser desplegado en un entorno de producción (por ejemplo, una aplicación web o API que expone las operaciones de la calculadora).

### Parte 2: Opciones de estrategia de despliegue

Presenta a tus estudiantes las siguientes estrategias comunes:

| Estrategia                   | Descripción   | Ventajas                   | Riesgos                        |
|------------------------------|---|----------------------------|--------------------------------|
| <b>Despliegue directo</b>    | Se reemplaza la versión anterior por la nueva                                     | Simple y rápido            | Riesgo alto si hay errores     |
| <b>Blue-Green Deployment</b> | Se mantiene una versión activa (blue) y se despliega la nueva en paralelo (green) | Permite rollback inmediato | Requiere doble infraestructura |
| <b>Canary Deployment</b>     | Se lanza la nueva versión a un pequeño grupo de usuarios                          | Detecta errores tempranos  | Requiere monitoreo fino        |
| <b>Rolling Deployment</b>    | Se actualiza gradualmente por grupos de servidores                                | Minimiza interrupciones    | Puede complicar el rollback    |

### Parte 3: Documento técnico

Elabora un documento con esta estructura:

## 1. Introducción

El proyecto Calculadora en Java consiste en una aplicación que realiza operaciones aritméticas básicas (suma, resta, multiplicación y división).

Cuenta con pruebas automatizadas desarrolladas con JUnit y un pipeline CI configurado en GitHub Actions, el cual valida la correcta compilación y ejecución de los tests en cada commit.

### Objetivo del despliegue:

Definir e implementar una estrategia de despliegue automatizada que permita publicar nuevas versiones del proyecto con el menor riesgo posible, asegurando estabilidad, trazabilidad y una experiencia continua para el usuario final.

## 2. Entorno de despliegue

- **¿Dónde se desplegará? (Ej. servidor local, nube, contenedor Docker, etc., explica)**

El sistema se desplegará en un contenedor Docker, ejecutado sobre una instancia en la nube (GitHub Actions).

Esto permite una configuración reproducible, control de versiones y facilidad para realizar actualizaciones.

- **¿Qué herramientas se usarán? (Ej. GitHub Actions, Docker, Heroku, etc., y justifica)**

GitHub Actions: automatiza compilación, pruebas y despliegue.

## 3. Estrategia seleccionada

### Estrategia elegida: Blue-Green Deployment

#### Descripción técnica:

Se mantendrán dos versiones del entorno:

- **Blue (actual)**: versión estable en producción.
- **Green (nueva)**: nueva versión desplegada en paralelo.

El pipeline CI/CD realizará los pasos:

1. Construir la imagen Docker del proyecto.
2. Desplegar la versión *green* en un entorno alterno.
3. Ejecutar pruebas automáticas.
4. Si todo es exitoso, redirigir el tráfico al entorno *green*.

#### **4. Justificación**

Blue-Green Deployment reduce riesgos durante actualizaciones, ya que permite probar la nueva versión sin interrumpir la versión actual.

En caso de fallo, se puede revertir el tráfico al entorno anterior inmediatamente.

#### **Ventajas para este proyecto:**

- Cero tiempo de inactividad.
- Ideal para proyectos pequeños en Java con pipelines CI automatizados.

#### **Minimización de riesgos:**

- El entorno *green* se prueba completamente antes de redirigir usuarios.
- Logs y notificaciones (Slack) alertan en caso de fallo.

#### **5. Automatización**

##### **Integración con el pipeline CI (GitHub Actions):**

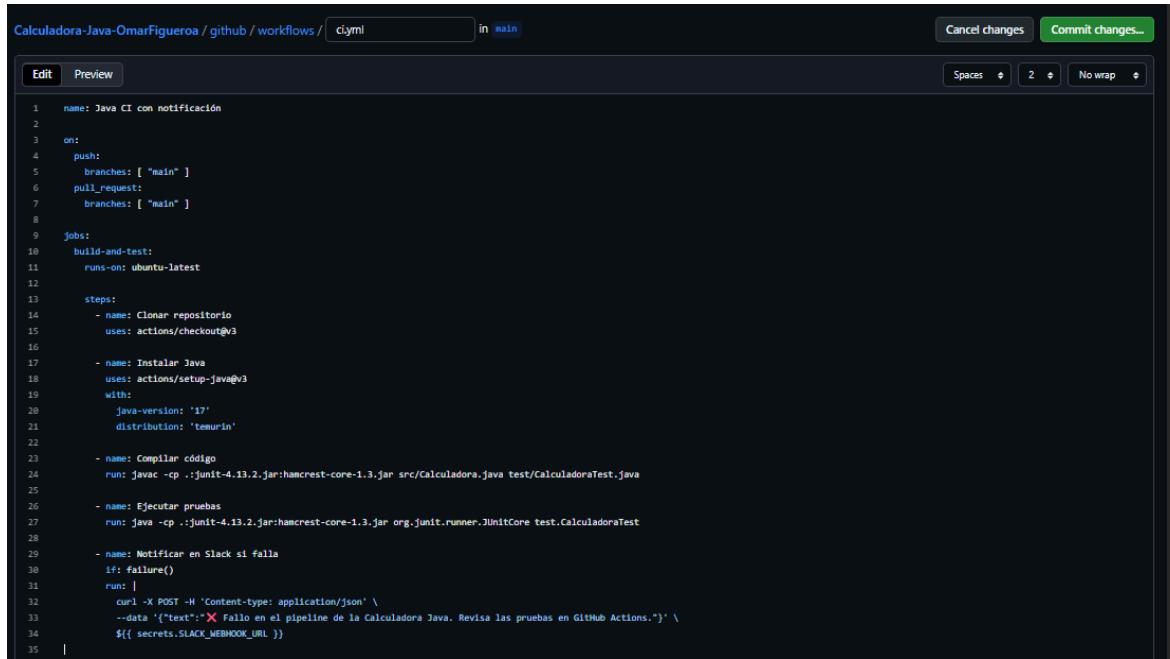
El pipeline actual de GitHub Actions se ampliará con un job de despliegue automático.

Pasos automatizados:

1. Build: compila el proyecto Java.
2. Test: ejecuta pruebas JUnit.
3. Deploy: publica la imagen en Docker Hub o despliega en el servidor *green*.
4. Notificación: si el despliegue falla, envía alerta a Slack usando el secreto SLACK\_WEBHOOK\_URL.

## 6. Evidencias

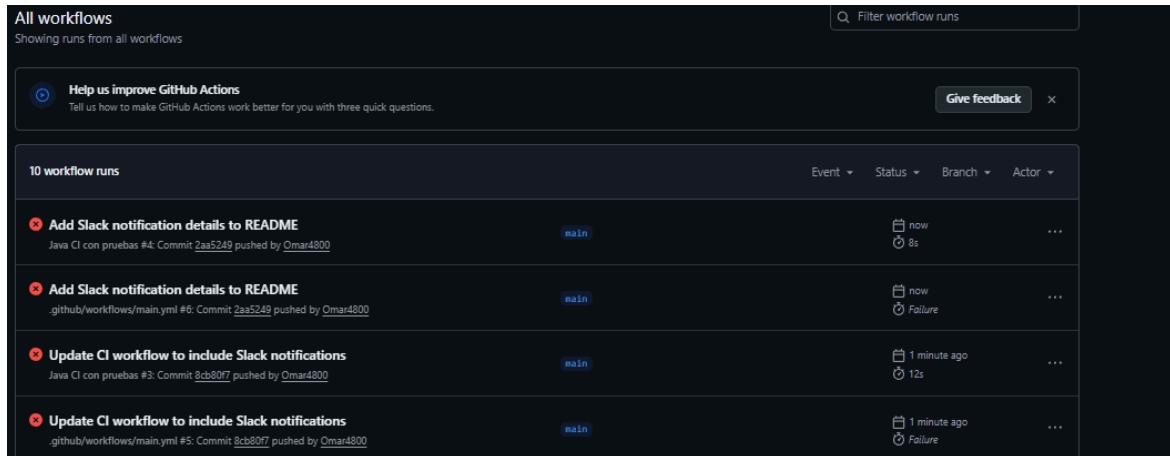
- Captura del pipeline ejecutando el despliegue



The screenshot shows the GitHub Actions workflow editor for a repository named 'Calculadora-Java-OmarFigueroa'. The workflow file is named '.github/workflows/main.yml' and is currently in the 'main' branch. The workflow definition is as follows:

```
name: Java CI con notificación
on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]
jobs:
  build-and-test:
    runs-on: ubuntu-latest
    steps:
      - name: Clonar repositorio
        uses: actions/checkout@v3
      - name: Instalar Java
        uses: actions/setup-java@v3
        with:
          java-version: '17'
          distribution: 'temurin'
      - name: Compilar código
        run: javac -cp :junit-4.13.2.jar:hamcrest-core-1.3.jar org.junit.runner.JUnitCore test.CalculadoraTest
      - name: Ejecutar pruebas
        run: java -cp :junit-4.13.2.jar:hamcrest-core-1.3.jar org.junit.runner.JUnitCore test.CalculadoraTest
      - name: Notificar en Slack si falla
        if: failure()
        run:
          curl -X POST -H 'Content-type: application/json' \
--data '{"text": "Fallos en el pipeline de la Calculadora Java. Revisa las pruebas en GitHub Actions."}' \
${{ secrets.SLACK_WEBHOOK_URL }}
```

- Logs o mensajes de éxito/fallo



The screenshot shows the 'All workflows' page on GitHub, specifically the 'Workflow runs' section. It displays four recent workflow runs for the 'Java CI con pruebas' workflow, all triggered by pushes to the 'main' branch. The runs are listed from newest to oldest:

| Run   | Event        | Status  | Branch | Actor    |
|---|--------------|---------|--------|----------|
| Add Slack notification details to README          | now          | Success | main   | Omar4800 |
| Add Slack notification details to README          | now          | Failure | main   | Omar4800 |
| Update CI workflow to include Slack notifications | 1 minute ago | Failure | main   | Omar4800 |
| Update CI workflow to include Slack notifications | 1 minute ago | Failure | main   | Omar4800 |

- Enlace al repositorio/documentación