**Guía Laboratorio 10**

**CI/CD - Github Actions**

# **Objetivo**

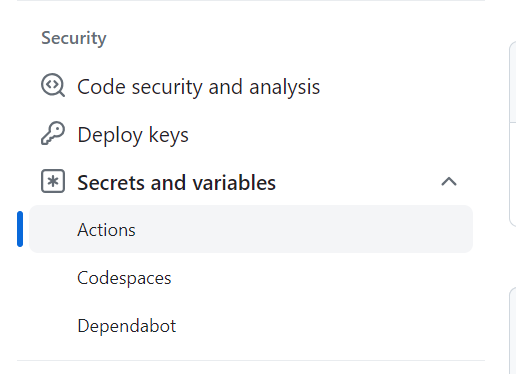
Comprender los principios operativos que permiten que los cambios de software se entreguen a los usuarios de manera oportuna, repetible y segura al introducir la automatización en los procesos de desarrollo de software.

# **Duración**

Ciento veinte (120) minutos

React build & S3 deploy

1. Ir al repositorio de Github. En la sección de **Settings > Secrets and variables > Actions**.



1. Seleccionar **New Repository Secret**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico, Teams

Descripción generada automáticamente

1. Ingresar uno por uno **AWS\_ACCESS\_KEY\_ID , AWS\_SECRET\_ACCESS\_KEY , AWS\_SESSION\_TOKEN**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Al final deben existir los 3 **secrets**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

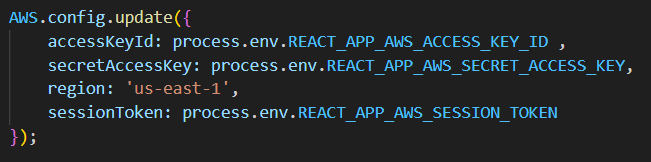
1. En la carpeta de nuestro proyecto, vamos a crear el directorio **.github/workflows** , dentro crear el archivo **build-deploy.yml** con el siguiente contenido

|  |
| --- |
| name: Build & deploy  on:    push:      branches:        - master  env:            CI: false            REACT\_APP\_AWS\_ACCESS\_KEY\_ID: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}            REACT\_APP\_AWS\_SECRET\_ACCESS\_KEY: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}            REACT\_APP\_AWS\_SESSION\_TOKEN: ${{ secrets.AWS\_SESSION\_TOKEN }}  jobs:    deploy:      runs-on: ubuntu-latest      steps:        - uses: actions/checkout@v2        - name: Use Node.js          uses: actions/setup-node@v1          with:            node-version: 12.x        - name: Configure AWS credentials          uses: aws-actions/configure-aws-credentials@v1          with:            aws-access-key-id: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}            aws-secret-access-key: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}            aws-session-token: ${{ secrets.AWS\_SESSION\_TOKEN }}            aws-region: us-east-1        - name: Install packages          run: npm install        - name: Build          run: npm run build        - name: Deploy          run: aws s3 sync ./build s3://react-test-deploy-cd |

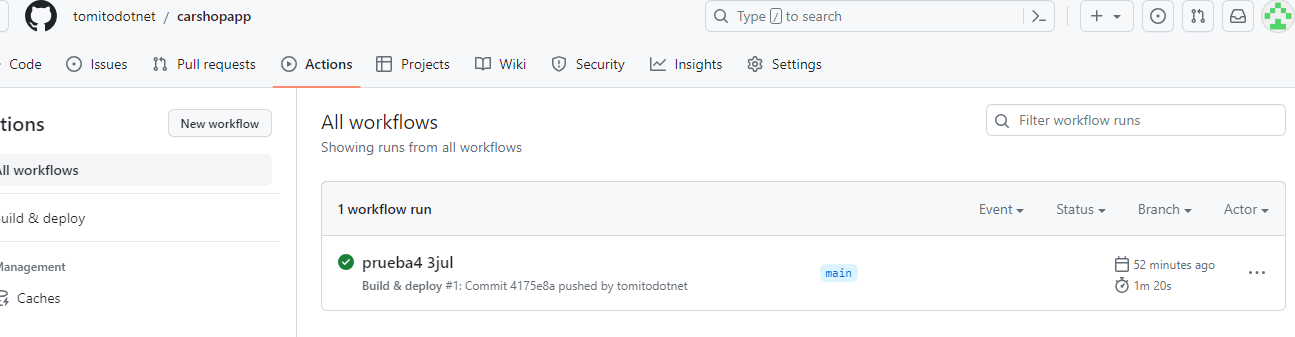
Consideraciones:

* Si usa una versión diferente de node, cambiar.
* Si ud. puso un nombre distinto a las variables en el secret, cambiar la variable entre {{}}
* Si ud. cambió el directorio de salida (/build)
* Cambiar por su nombre de bucket

1. Realizados los cambios hacer el commit.
2. En el repositorio, ir a la pestaña Actions y escoger la opción Set up a workflow yourself, verificar que se haya implementado el workflow.
3. Si Ud. Desea usar alguna variable de entorno puede hacer el cambio respectivo en el Código para acceder a través de **process.env.REACT\_APP\_NOMBRE** como el siguiente ejemplo.

****

1. Adjuntar captura de su pestaña de Github Actions cuando haya realizado algunos pipelines correctamente.



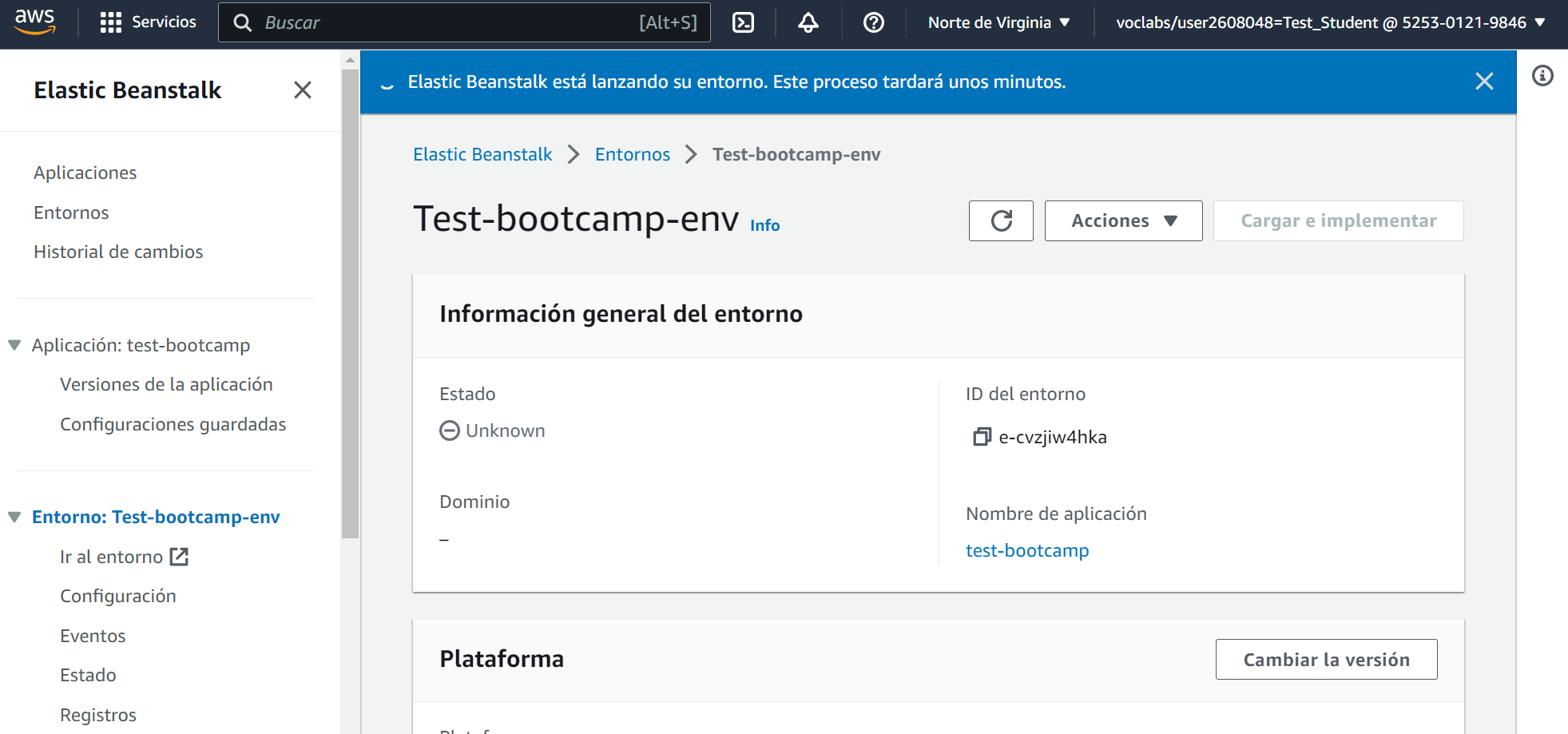
Express build & ElasticBeanstalk deploy

1. Buscar el servicio Elastic Beanstalk y escoger Crear aplicación.

****

1. Configurar los siguientes parámetros. Al final debe esperar cerca de 5 minutos hasta que el despliegue se haya realizado correctamente.

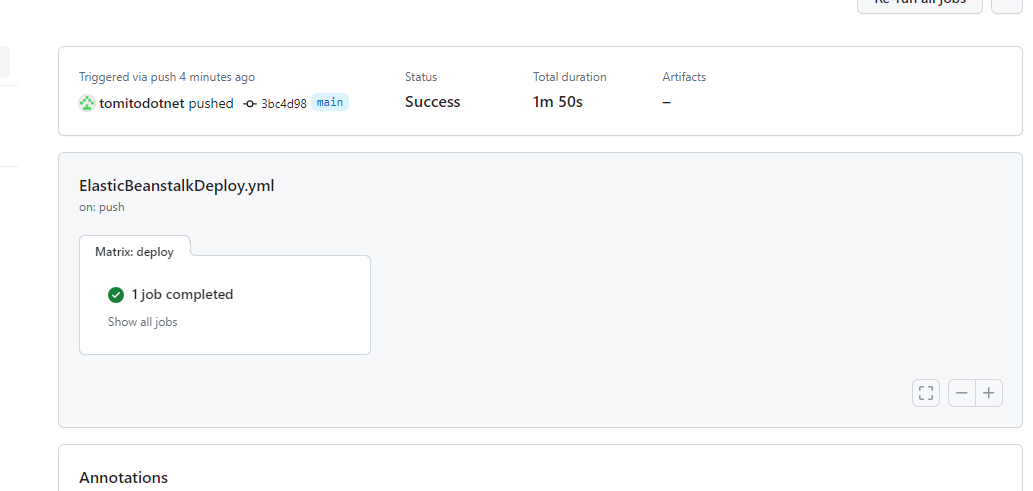
* Nivel de entorno: Entorno de servidor web
* Nombre de la aplicación: a su gusto
* Plataforma: node.js
* Ramificacion de la plataforma: node.js 14
* El resto de campos de esta sección por defecto
* Seleccionar los roles LabRole y Vockey
* Seleccionar la VPC por defecto
* Direccion IP Publica: Activado
* Escoger al menos 1 subred de instancia
* Proceder hasta el final de la configuración

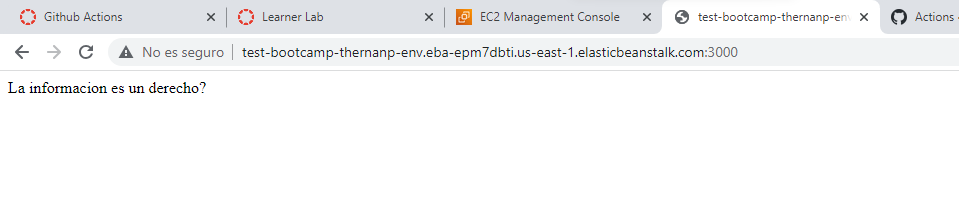


1. En un proyecto de Express, siguiendo los pasos de la actividad anterior, use el siguiente workflow para el nuevo proyecto. Debe reemplazar los nombres de su despliegue de Elastic Beanstalk y asegurarse que existen los Secrets.

|  |
| --- |
| name: ElasticBeanstalkDeploy  on:    push:      branches:        - main  env:      CI: false  jobs:    deploy:      runs-on: ubuntu-latest      strategy:        matrix:          node-version: [14.x]      steps:      - uses: actions/checkout@v2      - name: Use Node.js ${{ matrix.node-version }}        uses: actions/setup-node@v2        with:          node-version: ${{ matrix.node-version }}        - name: Installing NPM        run: npm install        - name: Generate deployment package        run: zip -r deploy.zip \* -x "\*\*node\_modules\*\*"        - name: Get timestamp        uses: gerred/actions/current-time@master        id: current-time      - name: Run string replace        uses: frabert/replace-string-action@master        id: format-time        with:          pattern: '[:\.]+'          string: "${{ steps.current-time.outputs.time }}"          replace-with: '-'          flags: 'g'        - name: Beanstalk Deploy my app        uses: einaregilsson/beanstalk-deploy@v18        with:          aws\_access\_key: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}          aws\_secret\_key: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}          aws\_session\_token: ${{ secrets.AWS\_SESSION\_TOKEN }}          application\_name: test-bootcamp          environment\_name: Test-bootcamp-env          region: us-east-1          version\_label: "app-version-${{ steps.format-time.outputs.replaced }}" #Using timestamp for the versions          deployment\_package: deploy.zip      - name: Deployed successfully!        run: echo App deployed to ELB |

1. Cargue una imagen de su app en funcionamiento.





React build & EC2 deploy

1. Descargar su archivo .PEM para la conexión ssh
2. Abrir el archivo con un block de notas y copiar el contenido.
3. Crear los siguientes 3 secrets:
   1. SSH\_PRIVATE\_KEY (copiar el valor obtenido en el paso anterior)
   2. REMOTE\_HOST ( poner la dirección ipv4 publica)
   3. REMOTE\_USER (usualmente ‘ubuntu’)
4. Usar el siguiente action

name: Build  & EC2 Deploy

on:

  push:

    branches:

      - master

env:

          CI: false

          REACT\_APP\_AWS\_ACCESS\_KEY\_ID: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}

          REACT\_APP\_AWS\_SECRET\_ACCESS\_KEY: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}

          REACT\_APP\_AWS\_SESSION\_TOKEN: ${{ secrets.AWS\_SESSION\_TOKEN }}

jobs:

  deploy:

    runs-on: ubuntu-latest

    steps:

      - uses: actions/checkout@v2

      - name: Use Node.js

        uses: actions/setup-node@v1

        with:

          node-version: 12.x

      - name: Configure AWS credentials

        uses: aws-actions/configure-aws-credentials@v1

        with:

          aws-access-key-id: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}

          aws-secret-access-key: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}

          aws-session-token: ${{ secrets.AWS\_SESSION\_TOKEN }}

          aws-region: us-east-1

      - name: Install packages

        run: npm install

      - name: Build

        run: npm run build

      - name: Deploy to Server 1

        uses: easingthemes/ssh-deploy@main

        env:

          SSH\_PRIVATE\_KEY: ${{ secrets.SSH\_PRIVATE\_KEY }}

          REMOTE\_HOST: ${{ secrets.REMOTE\_HOST }}

          REMOTE\_USER: ${{ secrets.REMOTE\_USER }}

          TARGET: "/home/ubuntu/web"

          SOURCE: "./build"