Laboratory Practice Report

Version control (git) and continuous deployment (CI/CD)

# 02/04/2025

Departamento de Electrónica, Sistemas e Informática (DESI)

Cloud Architecture *(Arquitectura en la Nube)*

Mtro. Rodolfo Luthe Ríos

# Daniel García Hernández

# 715989

# Ciencia de datos

*Keywords: pipeline, versión control, entrega continua, DevOps.*

## Introduction

Aplicar usos prácticos de los servicios de la nube pública.

Objetivos:

* Usar un cliente de control de versiones.
* Configurar servicio de control de versiones centralizado.
* Control de versiones de un documento.
* Implementar un ambiente de entrega continua DevOps.

Actividades generales:

* Configurar un repositorio local.
* Crear un repositorio de GitHub.
* Control de versiones de un documento.
* Configurar un ambiente de entrega continua de GitHub a ElasticBeanstalk.

## Theoretical Framework

Como nos menciona Yépez [1] las ventajas ofrecidas por la aequitectura serverless, ha dado lugar a un creciente desarrollo de este tipo de aplicaciones en Cloud Computing, por lo que surge la necesidad de contar con una referencia para la implementación de CI/CD que apoyen este tipo de desarrollo de aplicaciones, y por ende constituyan un aporte al ciclo DevOps.

Dentro del entorno AWS, el desarrollador de aplicaciones sin servidor, puede hacer uso de la combinación de servicios como AWS CodeCommit, AWS CodePipeline, AWS CodeBuild, AWS CodeDeploy y AWS CloudFormation para automatizar la construcción, pruebas y despliegue de aplicaciones serverless que están expresadas en plantillas construidas con AWS SAM (Serverless Application Model).

De igual manera Yépez [1] define la integración continua (CI) y la entrega continua/despliegue continuo (CD) como prácticas de desarrollo de software para producir software en ciclos cortos, en la que la integración de cambios en el código fuente y la actualización de aplicaciones es bastante frecuente. El objetivo final de estas prácticas es reducir los costos, el tiempo y los riesgos mediante la entrega de software en piezas pequeñas.

CI/CD constituye la base para la metodología DevOps, donde desarrolladores y operaciones trabajan de manera colaborativa para asegurar el despliegue ágil y continuo de sofware de calidad, enfocados en agregar mejas continuas al proceso en cada iteración.

Por otro lado, Moreno [2] menciona que una de las partes mas relevantes cuando se esta codificando, si no la que más, es el repositorio de código y el uso que hacemos de este. Tenemos que tener en cuenta las funcionalidades que nos proporciona el sistema de gestión de versiones de código, la herramienta git es open source que a día de hoy es el sistema de control de versiones más usado por la comunidad y el standard de facto en cualquier organización. Además debemos de considerar las características intrínsecas cuando se está definiendo programáticamente la infraestructura.

## Architectural diagram

Diagram of the implemented architecture.

## Practice Development

Version control

Lo primero que se hace es instalar git en la computadora local y configurar nombre y correo electrónico.

Texto

El contenido generado por IA puede ser incorrecto.

Se realizan 3 versiones del archivo versions y se realiza el commit en cada una de ellas.

Texto

El contenido generado por IA puede ser incorrecto.

En la imagen podemos observar las versiones del repositorio y vemos que efectivamente se realizaron los 3 commits.

Después de esto, se crea un nuevo repositorio en github, y dentro de nuestro repositorio local se conecta por medio de Add Hub y se suben nuestros archivos por medio del push.

Texto

El contenido generado por IA puede ser incorrecto.

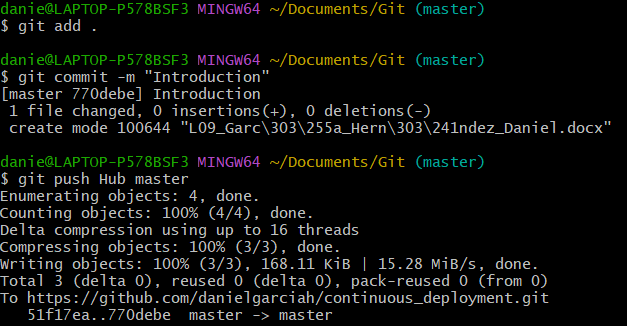
Para corroborar que efectivamente se subieron los cambios a Github abrimos en el navegador el repositorio.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Y observamos que efectivamente tenemos los 3 commits que se realizaron.

Después, para cada sección del presente documento se subirá un commit.



Y observamos en GitHub si nuestro documento se subió correctamente.

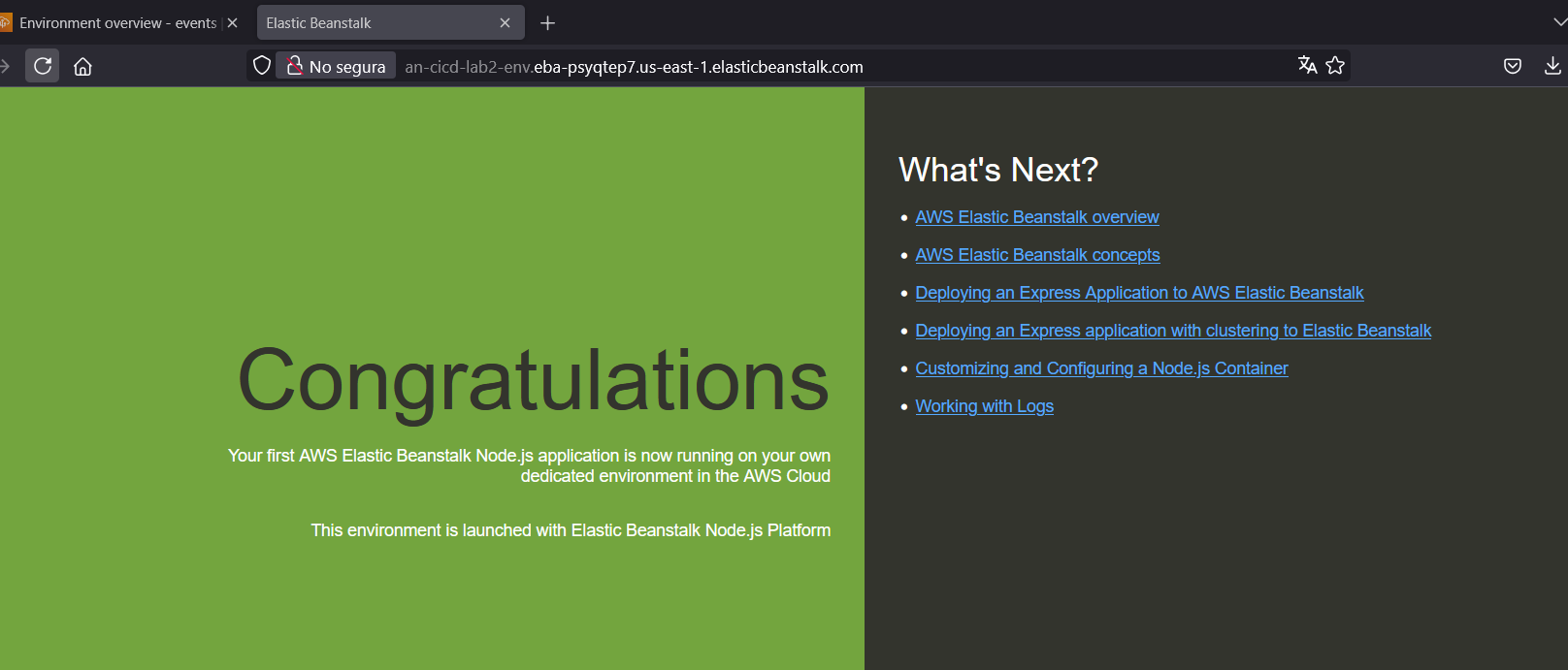
Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Continuous deployment environment

Lo primero que se hace es crear el environment y aplicación dentro de Amazon ElasticBeanstalk.

En donde se crea la aplicación dentro de Nodejs.



En la imagen podemos observar la aplicación de prueba, el cual nos confirma que efectivamente se creó el ambiente y la aplicación de manera correcta.

## Problems and Solutions

List all the problems encountered during the development of the practice and how they were resolved.

## Experiments and Results

This section should explain each of the experiments conducted and their results. The results should be properly interpreted to understand their significance and why they are considered good or bad. Diagrams, photographs, or images can be included in this section to help represent the results.

## Cost analysis

Explain the cost of the implemented solution, justifying the chosen solution based on costs. Should detail monthly and annual costs.

## Conclusions

The conclusions should be a reflective work presenting the knowledge gained from the experiments, results, and the theoretical framework presented. Invalid conclusions include: "I learned a lot!", "I really liked the practice", "everything worked correctly." (1 or 2 paragraphs)

## Bibliography

[1] Yépez, C. “Implementación de un ciclo de CI/CD de una API serverless usando AWS Lambda”, trabajo de grado, Universidad Internacional de La Rioja.

[2] Moreno, J. “CI/CD en Infraestructura como código (IaC). Caso real en Amazon Web Services (AWS).”, trabajo de grado, Universitat Oberta de Catalunya, España.