Primera Entrega – Proyecto Integración Continua

Michael Andrés Cuevas Buitrago

Id.

Liz Tatiana López Bedoya

Id. 1811020265

Javier Rincón Huertas

Id.

Luis Oswaldo Quintero Velásquez Id. 1811024524

Politécnico Grancolombiano.

Facultad de Ingeniería, Diseño e Innovación.

Integración Continua.

**Tabla de Contenidos**

[Integración Continua 1](#_Toc87812615)

[Conceptos claves 2](#_Toc87812616)

[Repositorio Git: 2](#_Toc87812617)

[Contenedores Docker: 2](#_Toc87812618)

[Creación del repositorio: 3](#_Toc87812619)

[Creación de dos contenedores 4](#_Toc87812620)

[Lista de referencias 4](#_Toc87812621)

# Integración Continua

La evolución tecnólogica no solo ha evolucionado en nuevas herramientas y lenguajes para desarrollo de código, sino ha llegado al punto de abrir el código al mundo por medio del concepto del cloud computing, un modelo que busca la disponibilidad de servicios en modelo de red, pero sin ir mucho más allá del código abierto al mundo, el código se abre a un modelo colaborativo en el que un equipo de trabajo de un proyecto especifico puede aportar de manera iterativa producto de software de tal forma que se testear y asegura sobre el proyecto de manera continua, ahora bien, nace el concepto de la integración continua propuesto inicialmente por Martin Fowler, que consiste en hacer integraciones automáticas de un proyecto lo más a menudo posible para así poder detectar fallos cuanto antes, entendiendo así por integración la compilación y ejecución de pruebas de todo un proyecto.

La IC más que un modelo de herramientas interconectadas es una práctica de desarrollo mediante la cual la combinación de código se realiza en repositorios centrales, ejecutando y generando versiones de manera periódica, tras la cual como disparadores, se ejecutan pruebas y testeo automático. Es frente a esa capacidad integrada quelas herramientas cobran la relevancia en el modelo aplicado. El set de herramientas que se usan en el modelo de IC son de diversos usos, arquitectura, licenciamiento y especialidad, aportando al mercado un portafolio suficientemente basto para que cualquier empresa, proyecto o software pueda acoplar estas buenas prácticas.

# Conceptos claves

Para el presente proyecto, se iniciará por la implementación de un repositorio colaborativo, en el cual el equipo podrá interactuar y aportar al modelo. En paralelo se crearán dos contenedores debidamente conectados entre ellos de tal forma que se optimicen las capas del proyecto. Para ello a continuación damos contexto a los conceptos aplicados:

# Repositorio Git:

El concepto de repositorio nace con el objetivo de optimizar el desarrollo del producto, permitiendo aportes desde diversos desarrolladores, sin embargo, el modelo abierto modificando el proyecto principal lo expondría a riesgos de calidad, reprocesos o duplicidad, es por ello que el repositorio maneja dos conceptos claves: la bifurcación y la fusión.

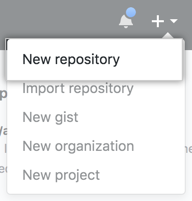
La bifurcación es la capacidad de que cada desarrollador pueda clonar en local el repositorio, realizar el desarrollo de código de forma segura sin afectar el proyecto en el proceso de prueba. Posterior al aseguramiento del cambio en local, cada desarrollador implementará la fusión, lo cual consiste en integrar al repositorio central su aporte de código creando así una nueva versión oficial a continuar iterando. Este modelo permite la trazabilidad de cambios y por tanto la reversión de los mismos en caso de fallo.

Frente a este modelo GIT es un sistema de control de versiones distribuida, creada por Linus Torvalds en el 2005.

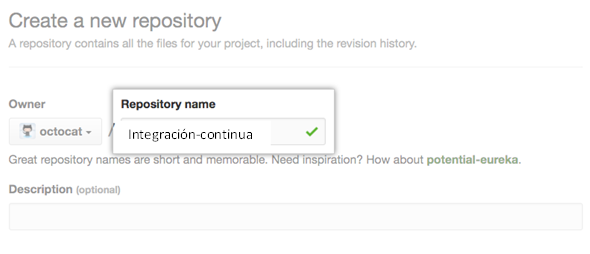
**Creación paso a paso Repositorio GitHub**

Puedes almacenar distintos proyectos en los repositorios de GitHub, incluso proyectos de código abierto. Con proyectos de código abierto, puedes compartir el código para hacer que el software funcione mejor y sea más confiable. Puedes utilizar los repositorios para colaborar con otros y rastrear tu trabajo.

1. En la esquina superior derecha de cualquier página, utiliza el menú desplegable y selecciona **Repositorio Nuevo.**

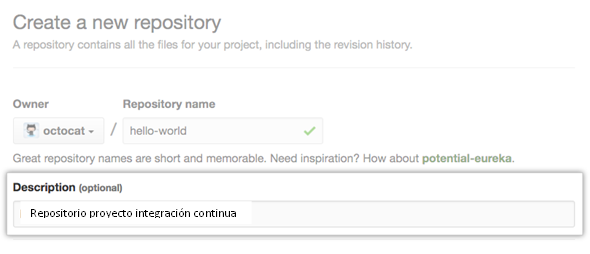
.

2. Escribe un nombre corto y fácil de recordar para tu repositorio. Por ejemplo: "integración continua"

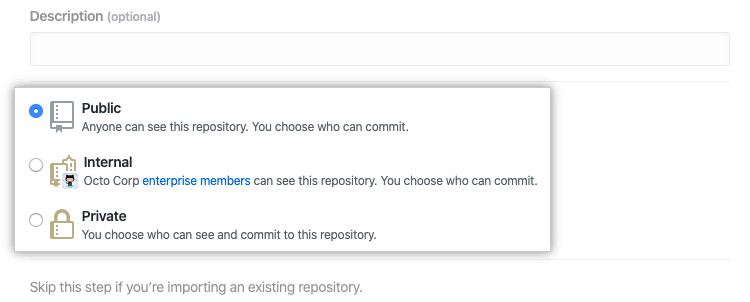


.

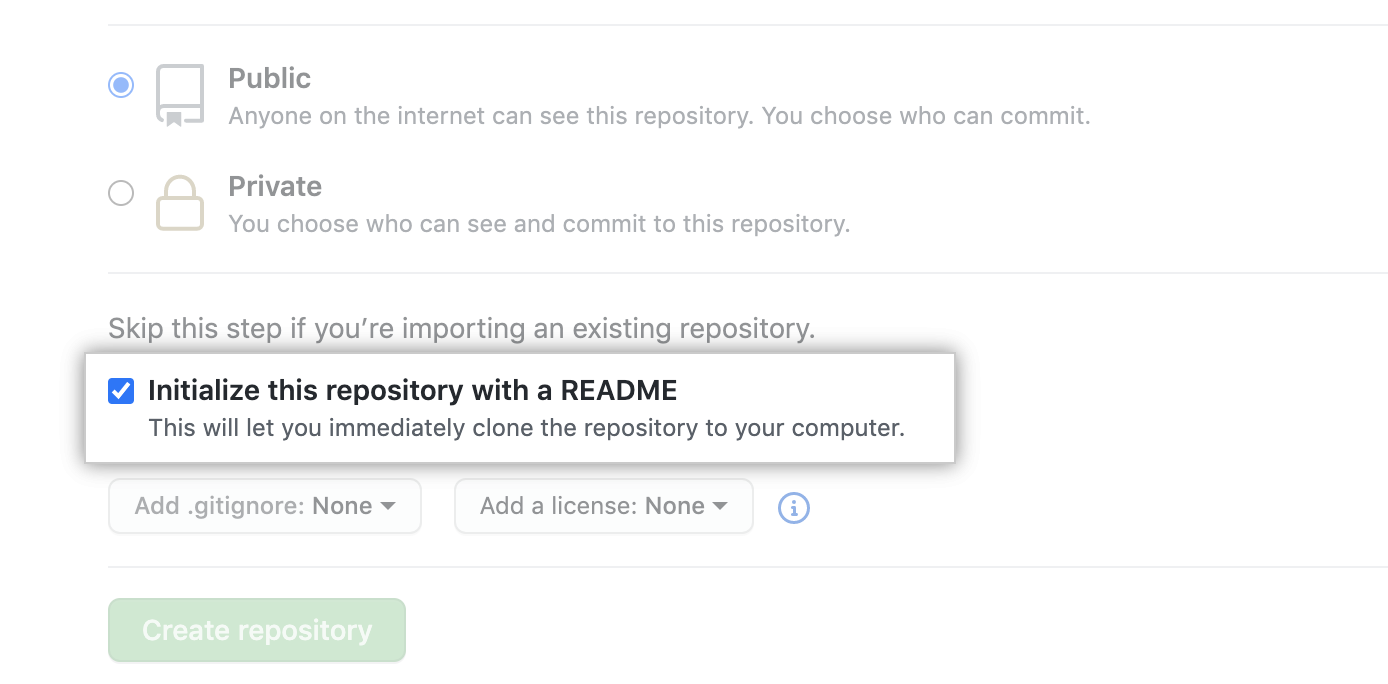
3. También puedes agregar una descripción de tu repositorio. Por ejemplo, "repositorio proyecto integración continua"

.

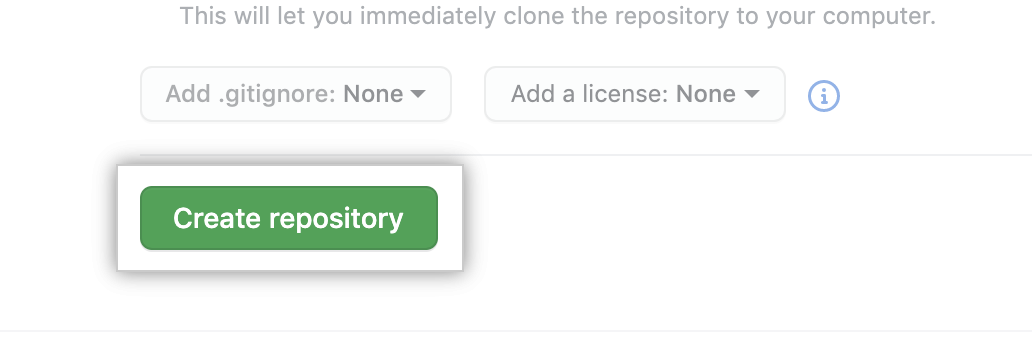
4. Elige la visibilidad del repositorio.



5. Selecciona **Inicializar este repositorio con un README.**



6. Haz clic en **Crear repositorio**



# Contenedores Docker:

Dada la necesitar de movilidad de las aplicaciones y del producto de software, este no solo está compuesto por código software, este requiere las librerías, bibliotecas, herramientas del sistema, archivos de configuración entre otros. El contenedor nace como una solución para virtualización de aplicaciones por medio de empaquetar de manera estandarizada estas soluciones, a diferencia de las metodologías de virtualización docker no integra el sistema operativo creando módulos más ligeros, incluso permitiendo ejecución de varios contenedores logrando soluciones aún más amplias. Entre los beneficios de contenedores docker adicional a la movilidad y ligereza de los mismos, está la estandarización de las operaciones, la transferencia de manera simple, y por tanto el bajo costo en su implementación.

# Creación del repositorio:

https://github.com/micuevas1/Integracion-continua

# Creación de dos contenedores

# Lista de referencias

<https://www.jetbrains.com/es-es/teamcity/ci-cd-guide/continuous-integration/>

<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/integracion-continua/>

<https://kinsta.com/es/base-de-conocimiento/que-es-github/>

<https://www.xataka.com/basics/que-github-que-que-le-ofrece-a-desarrolladores>

<https://www.netapp.com/es/devops-solutions/what-are-containers/>

<https://docs.github.com/es/get-started/quickstart/create-a-repo>