REVISTA ODIGOS

Período junio - septiembre 2022
Vol. 3, Núm. 2
rodigos@uisrael.edu.ec
https://doi.org/10.35290/ro.v3n2.2022.585 • e-ISSN: 2697-3405

Balanceo y despliegue de carga en aplicaciones web mediante kubernetes

Load balancing and deployment in web applications using Kubernetes

Fecha de recepción: 2022-03-28 • Fecha de aceptación: 2022-05-18 • Fecha de publicación: 2022-06-10

Andrés Ricardo Ramos Rodríguez[1]
Investigador independiente, Ecuador
arramos007@gmail.com
https://orcid.org/0000-0002-0142-816X

Pablo Marcel Recalde Varela[2]
Universidad Tecnológica Israel, Ecuador
precalde@uisrael.edu.ec
https://orcid.org/0000-0001-7256-2836

RESUMEN

La presente investigación consiste en desarrollar un despliegue DevSecOps que resuelve la necesidad de automatizar de una manera óptima y segura, mediante la utilización de las buenas prácticas DevSecOps. El proceso consiste en que el código implementado de un proyecto con un Framework Laravel se versiona en un repositorio Git y se genere a partir de la rama principal una imagen de contenedor, la misma que se desplegará de manera automática en un clúster de Kubernetes con el uso de flujo de acciones. Así pues, para generar procesos de despliegue de aplicaciones web de una manera segura y rápida, reduciendo procesos de errores o conflictos de versiones, permitiendo procesos de reverso de cambios. Para el uso de las funcionalidades de las diferentes plataformas se utilizan repositorios privados, de esta manera el proyecto publicado se

implementará de manera segura. Para el trabajo automatizado se usa tecnología Git, Kubernetes, Actions, Manifiestos, PHP, Composer y Laravel.

PALABRAS CLAVE: Php, Deployment, Docker, Kubernetes, GitHub, Dockerfile, Back-end, Front-end

ABSTRACT

This research consists of developing a DevSecOps deployment that solves the need to automate in an optimal and secure way by using DevSecOps best practices. The process consists in that the code implemented in a project with a Laravel Framework is versioned in a Git repository and a container image is generated from the main branch, which is automatically deployed in a Kubernetes cluster with the use of action flow. To generate web application deployment processes in a secure and fast way, reducing error processes or version conflicts, allowing change rollback processes. For the use of the functionalities of the different platforms, private repositories are used, in this way the published project will be implemented in a secure way. For the automated work we use Git, Kubernetes, Actions, Manifestos, PHP, Composer and Laravel technology.

KEYWORDS: Php, Deployment, Docker, Kubernetes, GitHub, Dockerfile, Back-end, Front-end

Introducción

Un correcto flujo de implementación de aplicaciones web es una parte importante dentro de cualquier metodología de desarrollo; esto permite reducir tiempos y errores en los despliegues de aplicaciones web en ambientes de producción. El proceso se fundamenta por medio del manejo del código fuente versionado en una rama principal, por el cual se podrá generar un proceso de despliegue en clúster de Kubernetes.

Según Beyer et al. (2016), la ejecución de servicios confiables requiere procesos de lanzamiento confiables. Los ingenieros de confiabilidad del sitio (SRE) deben saber que los archivos binarios y las configuraciones que utilizan se construyen de manera reproducible y automatizada para que los lanzamientos sean repetibles y no sean “copos de nieve únicos”. Los cambios en cualquier aspecto del proceso de lanzamiento deben ser intencionales, en lugar de accidentales. Los SRE se preocupan por este proceso desde el código fuente hasta la implementación.

Para McNutt (2013) es recomendable utilizar un sistema de construcción automatizado. Poder construir rápidamente y bajo demanda; el proceso de construcción debe estar completamente automatizado y hacer cosas como ejecutar pruebas, empaquetar e incluso implementar. Un sistema de compilación debe admitir un continuo y periódico (por ejemplo, todas las noches) proceso de construcción. Una compilación continua generalmente se desencadena por código y envíos, las compilaciones frecuentes pueden reducir los costos mediante las primeras identificaciones (y corrección) de errores.

En la actualidad, para un despliegue ágil en el proceso de *Release Engineering*, los avances tecnológicos proponen nuevas herramientas que apoyan a estos procesos de una manera estructurada y con automatizaciones que complementan la información de todo el proceso. Asimismo, en estos tiempos, dentro de las diferentes metodologías de desarrollo la etapa de despliegue es muy importante para el proceso de implementación de una aplicación web. Se debe cumplir con las mejores prácticas de la ingeniería de lanzamiento utilizando las mejores herramientas para estos procesos (Song et al., 2019).

Las aplicaciones web orientadas al usuario se reconstruyen con frecuencia con un enfoque de lanzamientos permanentes, permitiendo automatizar hasta minimizar al máximo la participación de los equipos de despliegue. En los procesos de compilación la expectativa es tener los mismos resultados, indistintamente de la infraestructura tecnológica en la cual se realiza el despliegue sin crear una dependencia de proveedor de servicios.

Se debe garantizar que la versión final que se aloja en el repositorio utilice las dependencias acordes a esta. Esto garantiza la compatibilidad al momento del lanzamiento. También, con la implementación de un flujo de despliegue para los aplicativos webs en el cual los ambientes, tanto el de pruebas como el de producción, manejen la misma arquitectura, lo que permitirá a los equipos implementar automáticamente y con la certeza de que el proceso se cumple sin presentar errores, así como el generar puntos de retorno de versiones de los aplicativos y reducir los tiempos.

Con el uso de plataformas de control de versiones de código e imágenes Docker se podrá realizar un flujo de trabajo para el despliegue de aplicaciones web Laravel (Laravel, 2020) en un ambiente de producción real.

Metodología

Para el desarrollo de este trabajo se utilizan dos metodologías: la de investigación y la de desarrollo de ágil, a continuación, se explican:

2.1 Metodología de investigación

La metodología utilizada para recopilar información en esta investigación es la observación directa, esto significa que "el investigador está ahí, en el lugar donde se desarrolla la acción y está preparado para registrar lo que está ocurriendo" (Mendoza, 1994).

2.2 Marco de trabajo Scrum

Administrar un equipo de trabajo para el correcto desempeño y lograr alcanzar los objetivos de puesta en marcha de un sistema es necesario una metodología de trabajo; en este caso Scrum es la mejor opción. Miguel Ángel de Dios (2020) afirmó lo siguiente: "A la hora de poner en marcha un proyecto, toda empresa debe asegurar que el equipo implicado conoce sus tareas y plazos de tiempo de entrega. Scrum es un marco de trabajo que nos ayuda a conseguirlo y que, además, permite agilizar la entrega de valor al cliente en iteraciones cortas de tiempo".

2.3 Artefactos usados

Los principales artefactos usados para la gestión del trabajo en el aspecto de desarrollo son (Roche, s.f):

- **Product Backlog:** es la principal fuente de información sobre el producto en Scrum, una lista, en cualquier formato, que contiene todos los requerimientos que necesitamos implementar en el producto. Esta lista es el resultado del trabajo del *Product Owner* con el cliente, los distintos *stakeholders*, sponsors, comités, etc, y refleja el estado real del trabajo pendiente de implementar en el producto, así como el ya realizado.
- **Sprint Backlog: s**e trata de una lista de elementos en los que trabajar durante la etapa de Sprint. Estos elementos normalmente se componen de tareas técnicas más pequeñas que permiten conseguir un incremento de *software* terminado. Todo el trabajo que el Development Team haya seleccionado para hacer durante el siguiente Sprint pasa al Sprint Backlog.

Resultados

DevOps son profesionales de TI con experiencia en el desarrollo de *software* con un alto nivel de estándares de buenas prácticas de desarrollo, secuencias de comandos y gestión de la operación general de desarrollo e implementación de productos.

Los desarrolladores transforman las formas tradicionales de desarrollo clásico de *software,* equipos de operaciones y pruebas en un entorno holístico para el desarrollo de productos de calidad superior en organizaciones de cualquier tamaño operacional. Combinan su experiencia práctica y su gran conocimiento especializado en el desarrollo de *software* con su experiencia en análisis empresarial básico para ofrecer soluciones empresariales innovadoras (Morales, 2020).

Algunas responsabilidades principales de los desarrolladores de DevOps:

- Planificación y desarrollo de proyectos
- Despliegue del proyecto
- Gestión del rendimiento carga y estrés
- Mantenimiento y solución de problemas
- Prueba e implementación de código
- Gestión de código fuente y programación en lenguaje Script

El papel en las organizaciones empresariales de los desarrolladores DevOps es muy cambiante y exigente. Los desarrolladores tienen que trabajar en una amplia gama de tareas para cumplir con sus responsabilidades principales, incluidas las secuencias de comandos, la codificación y la reingeniería de *software*. En este sentido el aprovechar de arquitecturas de *software* que incluyan herramientas y buenas prácticas de despliegue permite apoyar de una manera organizada y dinámica el desarrollo de estas tareas.

Con la gran cantidad de herramientas disponibles es importante el análisis y estudios para determinar las que mejor se adapten a los proyectos que se desarrollan, esto permite el no entrar en limitaciones para el manejo de estas.

En la siguiente *Figura 1* se muestra el diagrama de despliegue de una aplicación.

Figura 1

Diagrama de Despliegue de una Aaplicación

Nota. Valdes (2021)

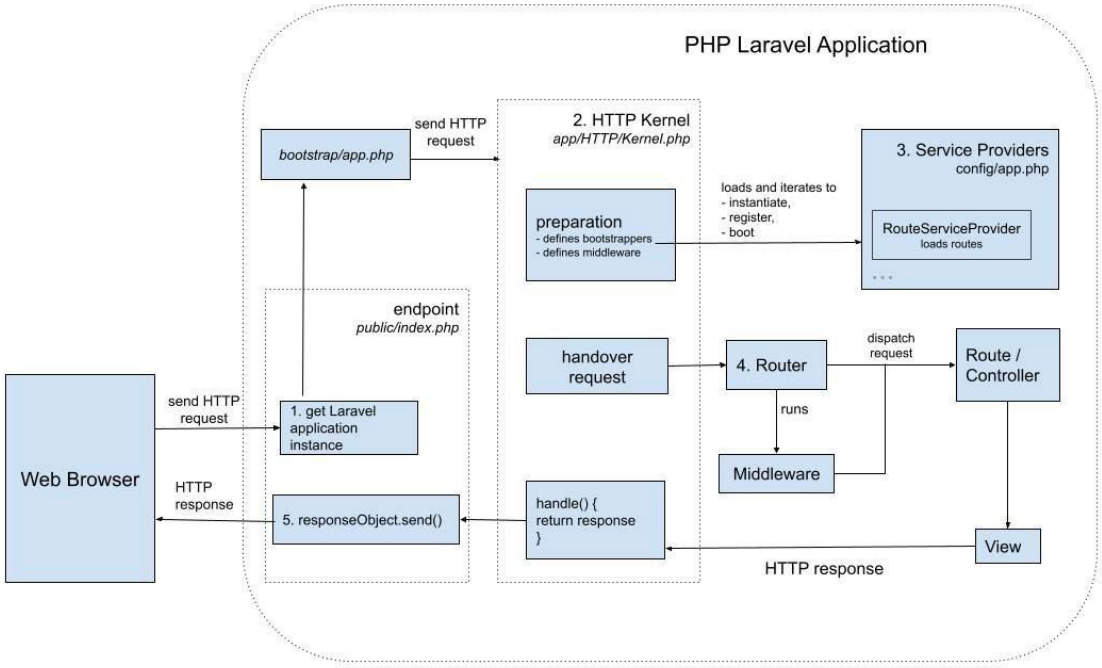
Laravel está comprometido en brindar una experiencia de desarrollo increíble, al mismo tiempo que brinda características poderosas, como la inyección de dependencia completa, la capa de abstracción de la base de datos de expresión, las colas, las tareas programadas, la integración, entre otras. Laravel (ver *Figura 2*) es un marco que puede crecer con el proyecto, es un marco "progresivo".

Este crecerá conforme las funcionalidades o la dificultad del proyecto lo haga. La extensa biblioteca de documentación y tutoriales permite reducir la curva de aprendizaje significativamente en los equipos de desarrolladores.

Asimismo, brinda herramientas poderosas para inyección de dependencia, pruebas unitarias, colas, eventos en tiempo real. Está optimizado para crear aplicaciones web profesionales y está listo para manejar cargas de trabajo corporativas. Este marco de desarrollo ha permitido una facilidad al momento de integración con herramientas de despliegue y un desarrollo acelerado para la implementación de estos procesos.

Figura 2

Arquitectura de Laravel

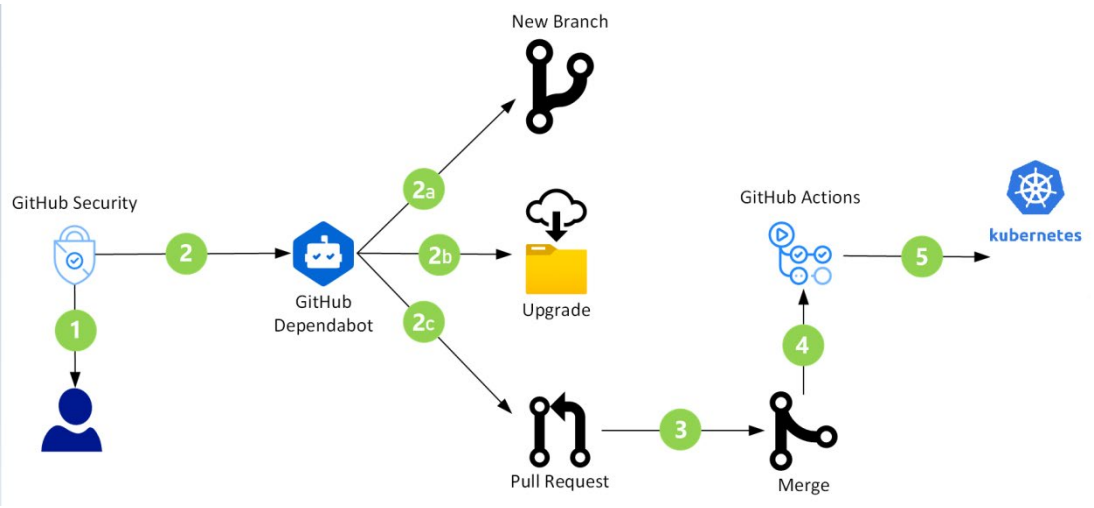


Dentro de las buenas prácticas de desarrollo, ya sea individualmente o en grupos colaborativos, es importante el versionado del código fuente; dentro de los beneficios está el mejoramiento en la eficiencia a través del control de cómo y cuándo se realizan los cambios.

Esta propuesta permitirá la ejecución de todo el proceso del ciclo de despliegue. En la siguiente *Figura 3* se muestra la arquitectura de GitHub.

Figura 3

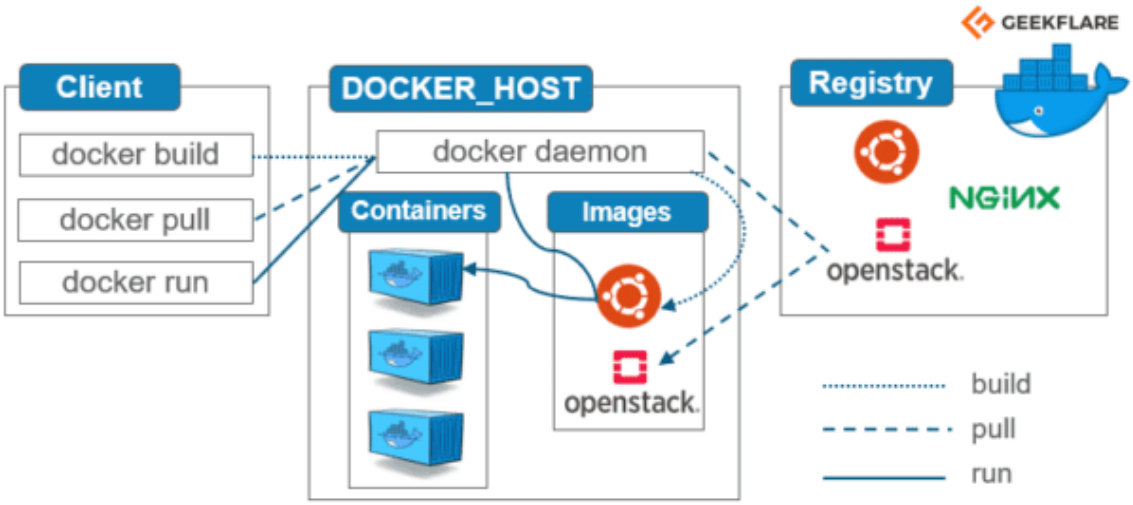
Arquitectura de GitHub



Docker (ver Figura 4)es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos (Vmware, 2021).

Figura 4

Arquitectura de Docker



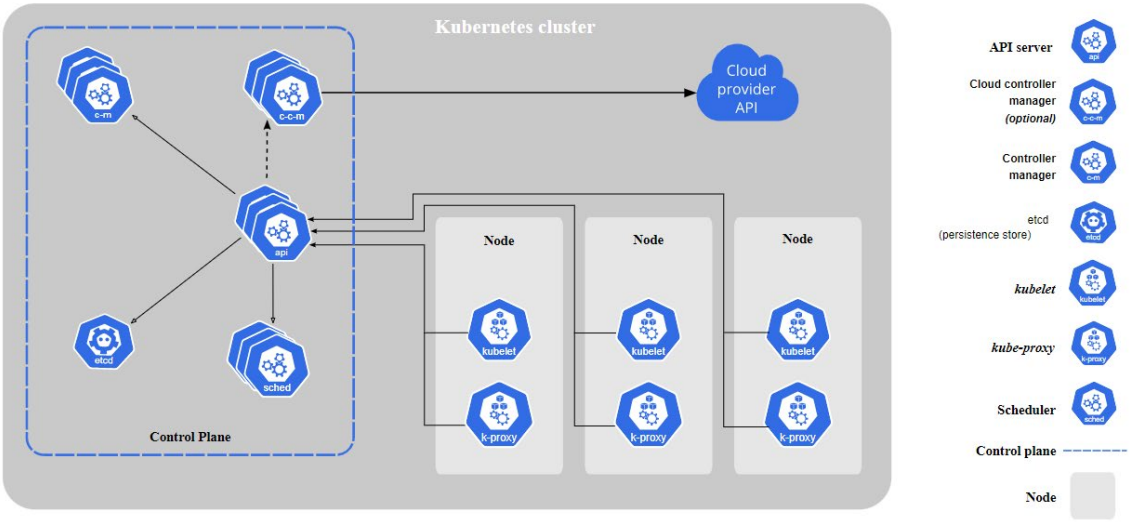
Por otra parte, Kubernetes tienen la capacidad de conectarse al API de un proveedor de servicios Cloud y ejecutar funcionalidades, como puede ser crear un Load Balancer, crear o destruir instancias, solicitar aprovisionamiento de servicios adicionales e interconectar a instancias (Burns et al., 2020).

Esto permite una escalabilidad de los servicios presentados en el Cloud de cualquier proveedor. Actualmente, se puede levantar un clúster de Kubernetes en casi cualquier infraestructura, tanto en ambientes de prueba, como en ambientes de producción (Morejón, 2022).

En la siguiente *Figura 5* se observa el proceso de despliegue.

Figura 5

Proceso de Despliegue a Través de un Manifiesto



Al escalar aplicaciones web horizontalmente, los primeros desafíos que enfrentará son el almacenamiento de archivos y la persistencia de datos. Esto se debe principalmente a la dificultad de mantener la coherencia de los datos cambiantes en múltiples nodos de aplicaciones. Se deben implementar estrategias apropiadas para garantizar que los datos generados en un nodo estén inmediatamente disponibles para otros nodos en el clúster.

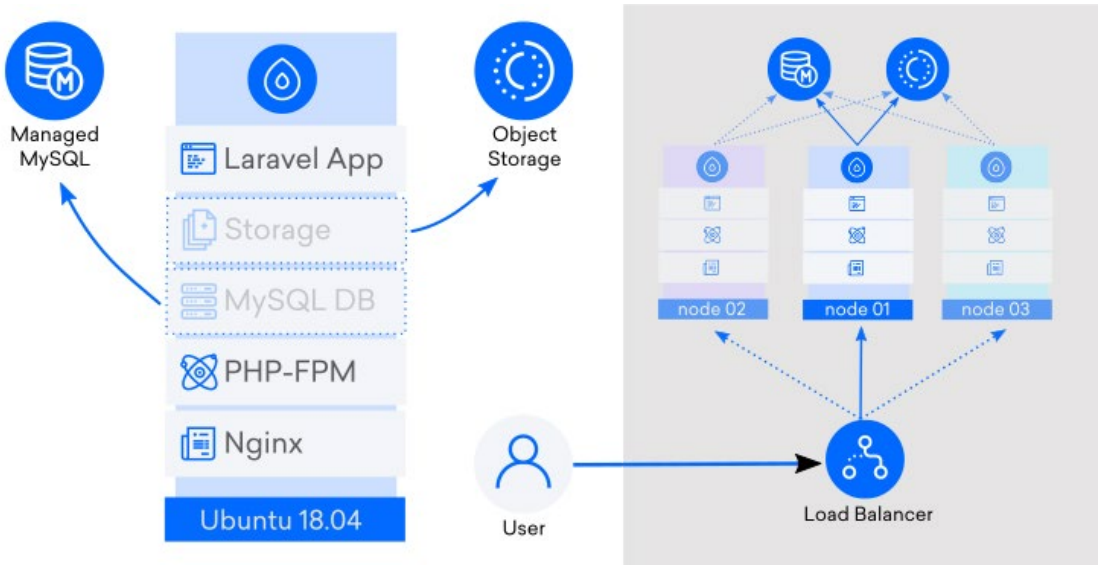
Una forma práctica de lidiar con la consistencia es usar una base de datos centralizada y un sistema de almacenamiento de objetos.

El primero subcontratará la persistencia de los datos a una base de datos administrada, y el segundo proporcionará almacenamiento remoto donde puede guardar archivos estáticos y contenido variable, tales como imágenes. Luego, cada nodo puede conectarse a estos servicios en el nivel de la aplicación.

La siguiente *Figura 6* demuestra cómo se puede usar una configuración de este tipo para la escalabilidad horizontal en el contexto de las aplicaciones PHP (Ahmad,2017):

Figura 6

Arquitectura de Laravel en un Clúster de Kubernetes



Dentro del proceso de ciclo de vida del desarrollo de software se plantea la ejecución de todo un proceso desde el desarrollo hasta la implementación en el ambiente de producción.

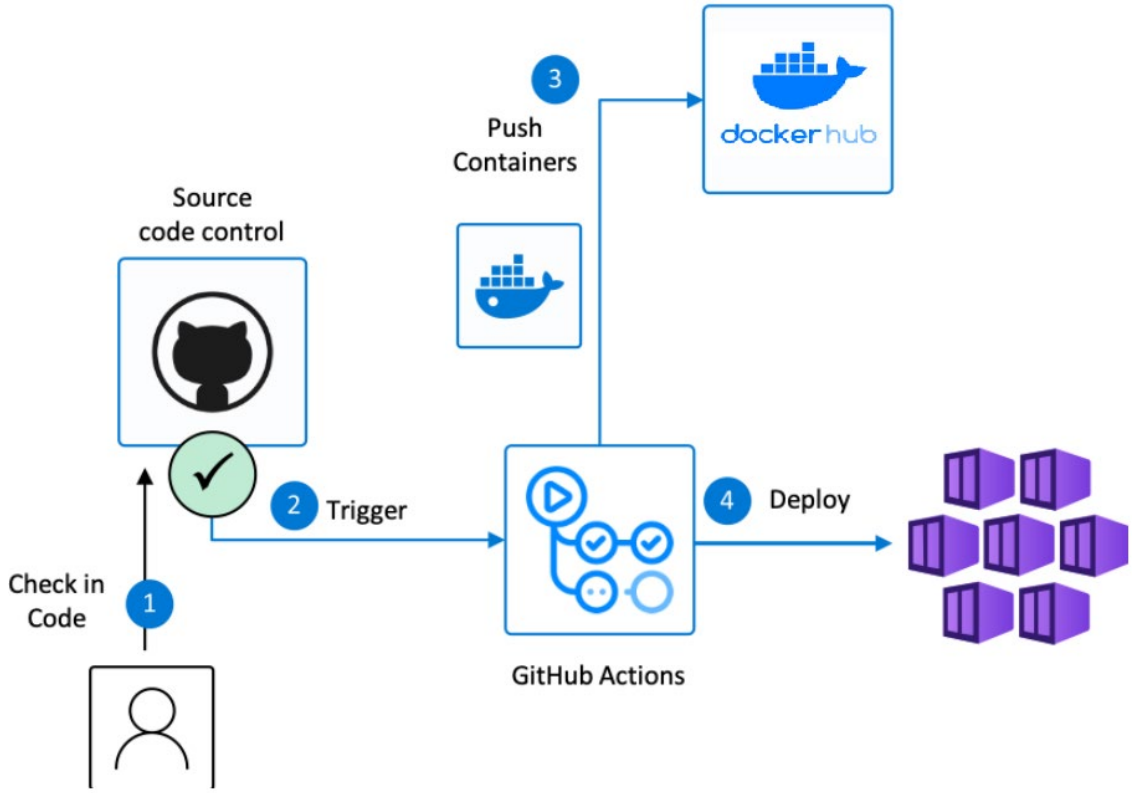
GitHub Actions permite la flexibilidad para crear un flujo de trabajo de ciclo de vida de desarrollo de *software* automatizado.

En este proyecto se usará el centro de implementación para generar flujos de acción de GitHub para crear una canalización de implementación para su código fuente basado en GitHub en un clúster de Kubernetes para cualquier proveedor de servicios de Cloud.

En la *Figura 7* se puede observar la arquitectura de un despliegue de una aplicación web Laravel

Figura 7

Arquitectura de un Despliegue de una Aplicación Web Laravel



Las consideraciones tomadas para la creación del archivo Dockerfile corresponden a entender los pre-requisitos de instalación del framework utilizado en este proyecto, de igual manera, los archivos complementarios a ser utilizados y aplicaciones de ejecución y de código fuente, al ser un proyecto web.

Es importante mencionar que el servidor web de aplicación utilizado es Apache dentro de una Kernel Linux, el cual es utilizado para montar sobre este las aplicaciones web.

Se determinaron dos archivos: Dockerfile y vhost.conf; a continuación se detalla la configuración de cada uno de ellos.

El vhost.conf es un archivo propio de configuración de Apache, el cual define el manejo de host virtuales con que la aplicación es publicada; se puede definir el manejo tanto de un protocolo http como https.

En esta primera fase de integración del proyecto se presenta la posibilidad de generar una imagen para generar el despliegue de la aplicación en cualquier ambiente.

La siguiente fase es generar un archivo de despliegue del proyecto en un ambiente Kubernetes; para esto se consideran 3 aspectos:

- Seguridad de repositorios
- Despliegue de aplicación
- Balanceo de carga de aplicativo

Para el despliegue de la aplicación se define el nombre del nodo, el número de réplicas que se tendrá, que se relaciona directamente al número de nodos que configura en el proveedor de servicios Cloud. A su vez, se define la imagen que se instalará desde el repositorio de imágenes privadas; finalmente, se configuran las variables de entorno que son utilizadas por la aplicación. La variable que se envía es para garantizar la privacidad de la aplicación por la definición del *framework*.

En cuanto al manejo de los nodos se administrará a través de un balanceador de carga configurado para los nodos, que previamente estén definidos por un mismo nombre, ya que se puede tener dentro del clúster de Kubernetes otros nodos disponibles para otras aplicaciones. Se define a través del puerto por el cual el balanceador expondrá la aplicación, internamente se distribuye la carga a cada uno de los nodos.

Dentro del proceso de automatización en el proyecto, la siguiente fase es integrar todo el proyecto para que, de manera automática, cualquier cambio que se realice se vea reflejado de modo automatizado en los nodos en el clúster de Kubernetes.

Para este proceso se ha realizado una configuración dentro del sistema de gestión de versiones mediante el uso de acciones; las cuales permiten ejecutar procesos de integración que cualquier cambio realizado en la versión principal se vea reflejada en todos los nodos; este proceso minimiza errores en el despliegue de la aplicación.

En el proceso de implementación se utilizaron herramientas de uso libre que reducen costos para los equipos de desarrollo. En cuanto al manejo de suscripciones para los diferentes repositorios para su manejo de manera privada, pueden variar dependiendo del tamaño de la organización y de los equipos de desarrollo.

Para una exitosa implementación del prototipo se siguen los puntos definidos por la metodología ágil de desarrollo. Las pruebas realizadas en contenedores locales previas a la puesta en producción en el clúster de Kubernetes por el proveedor de servicios web, manteniendo la misma configuración de despliegue de la aplicación en los dos entornos.

Conclusiones

El trabajo realizado representa el nivel de detalle de cada fase que se requiere para la correcta elaboración de la documentación, así como para la parte de desarrollo. Es importante sumergirse en cada capítulo y seguir los lineamientos planteados para maximizar el porcentaje de éxito.

Como parte de las conclusiones finales se determina que:

- Se logró implementar una arquitectura de clúster de Kubernetes con administración centralizada, la cual brinda servicios autogestionables para los usuarios del Cloud. Esta arquitectura permitió gestionar los recursos desde la configuración del proyecto con una comunicación eficiente.
- De igual manera, se logró reducir drásticamente los tiempos de despliegue de la aplicación en un ambiente de producción, dado que la tecnología de Kubernetes permite mediante una manera declarativa integrar todos los recursos que la aplicación necesita, esto gracias a la portabilidad y el fácil despliegue.
- La implementación y las configuraciones realizadas permitieron el despliegue de la aplicación web en cualquier infraestructura de Kubernetes, en los diferentes proveedores de soluciones Cloud. Esto permite el no depender exclusivamente de un proveedor de soluciones para las implementaciones en ambientes de producción.
- Para gestionar el clúster de Kubernetes en los proveedores de soluciones Cloud, se recomienda que las configuraciones se realicen en función de la arquitectura, y no en función de las especificaciones del proveedor.
- Es importante utilizar las buenas prácticas de desarrollo para que las implementaciones en diferentes capas sean factibles de implementar, y al momento de la puesta en producción, no se presenten inconvenientes conceptuales en la implementación.
- Resulta destacado, al momento de la implementación de soluciones de clúster de Kubernetes, definir en los requisitos no funcionales el fin del proyecto para que, al momento del aprovisionamiento, se puedan escoger los recursos en función del procesamiento, almacenamiento o la carga de uso de memoria RAM. Esto permite optimizar el rendimiento de la aplicación de la mano con los recursos contratados.

Referencias

Ahmad, H. (2017). *Building RESTful Web Services with PHP 7*. Packt Publishing.

Beyer, B., Murphy, N., Jones, C. & Petoff, J. (2016). *Site Reliability Engineering: How Google Runs Production Systems*. O'Reilly Media Inc.

Burns, B., Villalba, E., Strebel, D. y Evenson, L. (2020). *Guía práctica de Kubernetes*. Marcombo.

De Dios, M.A (09 de mayo de 2022). Scrum: qué es y cómo funciona este marco de trabajo. *WAM Global*. https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html

Laravel. (2020). Installation. https://laravel.com/docs/8.x

Mcnutt, D. (2013). Maintaining Consistency in a Massively Parallel Environment. In *Usenix Configuration Management Summit (UCMS'13)* [Conference]. https://www.usenix.org/conference/ucms13/summit-program

Mendoza, M. (1994). Técnicas de observación directa para estudiar interacciones sociales infantiles entre los Toba. *RUNA: archivo para las ciencias del hombre, 21*(1), 241-262. https://doi.org/10.34096/runa.v21i1.1400

Morejón, M. (2022). *Érase una vez Kubernetes*. Leanpub. https://leanpub.com/erase-una-vez-kubernetes/

Morales, I. (2020). *Programación Avanzada con PHP*. RC Libros.

Roche, J. (s.f). Artefactos Scrum: las 3 herramientas clave de gestión.

Elementos para la gestión de un proyecto de desarrollo de software. *Deloitte*. https://www2.deloitte.com/es/es/pages/technology/articles/artefactos-scrum.html

Song, P., Aborabh, A., & Mariappan, Y. (2019). *Day One: Building Containers Using Kubernetes and Contrail.* Juniper Networks Books.

Valdes, A. (24 de noviembre de 2021). DevOps Team: Roles and Responsibilities for 2022. *FAUN.* https://faun.pub/devops-team-roles-and-responsibilities-for-2021-17c1a475cdbd

Vmware. (29 de septiembre de 2021). *Tutorial del libro de visitas de Tanzu Kubernetes*.

https://docs.vmware.com/es/VMware-vSphere/7.0/vmware-vsphere-with-tanzu/GUID-CC395BC6-5E65-43F0-9828-5C3BAD6B8385.html