UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

ANÁLISIS DE LA PLATAFORMA COMO SERVICIO OPENSHIFT ORIGIN PARA LA CREACIÓN DE NUBES PRIVADAS

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS COMPUTACIONALES

AUTOR:

IVÁN DARÍO ROJAS ROJAS

DIRECTOR:

Ing. PABLO ANDRÉS LANDETA LÓPEZ, MSc.

IBARRA, 2019

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DEL CONTACTO			
CÉDULA DE IDENTIDAD:	040170903-5		
APELLIDOS Y NOMBRES:	Rojas Rojas Iván Darío		
DIRECCIÓN:	Avenida 17 de Julio		
E-MAIL:	idrojasr@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0985329714
DATOS DE LA OBRA			
TÍTULO:	ANÁLISIS DE LA PLATAFORMA COMO SERVICIO OPENSHIFT ORIGIN PARA LA CREACIÓN DE NUBES PRIVADAS		
AUTOR (ES):	Iván Darío Rojas Rojas		
FECHA: DD/MM/AAAA	26 de febrero de 2019		
PROGRAMA:	[X] PREGRADO [ ] POSGRADO		
TÍTULO POR EL QUE OPTA:	Ingeniero en Sistemas Computacionales		
ASESOR/DIRECTOR:	MSc. Pablo Andrés Landeta López		

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de esta y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

EL AUTOR:

.....

Rojas Rojas Iván Darío

CI: 040170903-5

Ibarra, a los 25 días del mes de febrero de 2019.

UNIVERSIDAD TÉCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

Ibarra, 26 de febrero de 2019

CERTIFICACIÓN DIRECTOR DE TESIS

Por medio del presente yo MSc. Pablo Andrés Landeta López, certifico que el Sr. Iván Darío Rojas Rojas, portador de la cédula de identidad Nro. 040170903-5. Ha trabajado en el desarrollo del proyecto de tesis: **"ANÁLISIS DE LA PLATAFORMA COMO SERVICIO OPENSHIFT ORIGIN PARA LA CREACIÓN DE NUBES PRIVADAS"**, previo a la obtención del título de Ingeniero en Sistemas Computacionales, lo cual ha realizado en su totalidad con responsabilidad.

Es todo cuanto puedo certificar en honor a la verdad.

.....

MSc. Pablo Andrés Landeta López

DIRECTOR DE TESIS

Dedicatoria

El presente proyecto está dedicado mi madre Rosario Rojas quien en todo momento me ha sabido guiar por el camino correcto, brindándome en el transcurso, su apoyo incondicional moral y económico, además de enseñarme sobre todo esto los valores que ella posee, entre ellos el que más admiro, el respeto y amor hacia la vida.

A mi hermana Yolanda Rojas quien es mi más grande ejemplo por seguir, quien me ha demostrado que con esfuerzo y perseverancia se puede conseguir todo lo que se propone, siempre de la mano con honestidad y respeto hacia el resto, gracias por siempre estar cuando lo he necesitado.

Debes esperar cosas de ti mismo antes de
que las puedas hacer. - Michael Jordan
Iván Darío Rojas Rojas

Agradecimiento

Agradezco infinitamente a mi familia la culminación de este trabajo, ya que es en mayor parte gracias a ellos, en razón de ser quienes me han motivado a avanzar hasta el punto en el que me encuentro académicamente.

Gracias a todas aquellas personas que han estado en mi vida dándome el empujón moral necesario para poder continuar en los momentos difíciles, gracias por su tiempo paciencia y palabras de aliento.

Gracias también a mis maestros por su experiencia y conocimientos impartidos día a día en las aulas de clase.

Iván Darío Rojas Rojas

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Resumen

El uso de internet es algo cotidiano para la mayor parte de las personas en la actualidad, lo que ha promovido el desarrollo de tecnologías y conceptos que faciliten el acceso a datos desde cualquier lugar en el mundo a través de cualquier dispositivo, el concepto más trascendental desarrollado es la nube, que hace referencia a todo lo que un sistema informático puede ofrecer pero como un servicio que se pueden acceder en una red tan extensa como es el internet, entre los tipos de nube que existen están las plataformas como servicio, de las cuales OpenShift Origin es una de estas, desarrollada bajo la ideología de software libre que más impacto ha tenido a nivel empresarial, la investigación realizada en este documento "ANÁLISIS DE LA PLATAFORMA COMO SERVICIO OPENSHIFT ORIGIN PARA LA CREACIÓN DE NUBES PRIVADAS" busca conocer todos los aspectos relevantes para una futura implementación de esta plataforma así como también el analizar su comportamiento.

Para el análisis de la plataforma OpenShift Origin se usa herramientas como, una aplicación de prueba que simula un proceso de gestión de becas desarrollado con la metodología SCRUM en el lenguaje de programación PHP conjuntamente un servidor de base de datos MySQL y un servidor CentOS 7.

Después de obtener los conceptos base sobre la plataforma OpenShift, se realiza el despliegue de la aplicación de prueba en el servidor CentOS 7 y en la plataforma como servicio OpenShift Origin, con lo cual se procede con una evaluación de rendimiento de los dos ambientes recopilando datos como el promedio en el tiempo de respuesta, la desviación estándar y el porcentaje de error de las peticiones enviadas en diferentes valores de carga, se realiza finalmente un análisis estadístico de los resultados obtenidos en los dos ambientes.

PALABRAS CLAVE: OpenShift Origin, CentOS 7, SCRUM, PaaS, SaaS, IaaS, Kubernetes, Docker, Contenedores

Abstract

The use of the internet has promoted the development of technologies and concepts that facilitate access to data from anywhere in the world through any device, the cloud is one of the most transcendental concept developed which refers to everything offered by a computer system accessed through internet, among the types of cloud that exist are the platforms as a service for example OpenShift Origin, developed under the ideology of free software that has had more impact at the enterprise level. This research with the topic "ANALYSIS OF THE PLATFORM AS OPENSHIFT ORIGIN SERVICE FOR THE CREATION OF PRIVATE CLOUDS" aims to know all the aspects for a future of this platform.

For the analysis of the OpenShift Origin platform, tools are used, such as a test application that simulates a scholarship management process developed with the SCRUM methodology in the PHP programming language, together with a MySQL database server and a CentOS 7 server.

After obtaining the basics concepts on the OpenShift platform, the application of the test application is performed on the CentOS 7 server and on the platform as OpenShift Origin service, which is followed by a performance evaluation of the two collecting data environments. As the average response time, the standard deviation and the percentage of error of the requests sent in different load values, a statistical analysis of the results obtained in the two environments is carried out.

KEY WORDS: OpenShift Origin, CentOS 7, SCRUM, PaaS, SaaS, IaaS, Kubernetes, Docker, Containers

INTRODUCCIÓN

Antecedentes

Con la llegada de la revolución tecnológica, las personas y entidades organizacionales han deseado obtener el mayor beneficio posible de ella, para lo cual era necesario el adquirir todos los recursos de hardware y software que fuesen necesarios para cumplir el objetivo perseguido por la persona o empresa, lo cual resultaba extremadamente costoso, inclusive se podía adquirir más de los recursos que eran necesarios para el correcto desempeño de la empresa, con la finalidad de solucionar este problema se crean nuevas opciones de ayuda. “Entre 2008 y 2009, surgió el nuevo paradigma tecnológico de la Computación en la Nube, con todas sus tecnologías asociadas, que al poco tiempo, despegó con su llegada al público” (Luis J.,2012,p 91)., Esta tecnología tiene el objetivo de brindar todo lo que el usuario necesite como un servicio sea hardware, software o una plataforma a un costo mucho menor del que conlleva la implementación dentro de la empresa y con la gran ventaja de obtener solo los recursos necesarios además de poder desistir de los que no sean necesarios mucho más fácilmente.

“En marzo de 2010 publicó un informe Top Threats a la Computación en la nube V1.0, sobre las siete mayores amenazas de la infraestructura de la Computación en la nube, con el propósito de asistir a las organizaciones en la toma de decisiones y en la adopción de estrategias que incluyan esta tecnología.” (INTECO, 2011, p12). Lo que comprobaba que muchos ambientes de nubes de cómputo tenían debilidades ya sea por fallas, vulnerabilidades o problemas de infraestructura, lo que ponía en duda que tan bueno era optar por poner la información importante y privada de la persona u organización en un lugar físicamente desconocido a pesar de su bajo costo.

“Richard Stallman[1] advirtió que la computación en nube es simplemente una trampa destinada a obligar a más gente a adquirir sistemas propietarios, bloqueados, que costarán cada vez más conforme pase el tiempo" (Periódico Guardian, 2008), lo que llevó a que se opte por la construcción o implementación de nubes privadas dentro de la empresa para brindar los servicios que sean necesitados por los distintos empleados o departamentos de la organización, esto tiene la ventaja de poder ser vinculado a una nube contratada en donde no se almacene la información importante que será resguardada en una nube privada.

[1] Richard Stallman: Padre fundador de la Free Software Foundation y del Proyecto GNU

"En el año 2006, se lanzó Google Docs. y trajo la computación en la nube a la vanguardia en la conciencia del público. Un año después, se dio una colaboración entre Google, IBM y universidades estadounidenses. En 2008, se lanzaron Eucalyptus[2] y OpenNebula[3]." (Zain M, 2013). Junto con ellas en los años consiguientes se dieron a conocer muchas herramientas, que brindan estas posibilidades para lo cual es necesario analizarlas y estudiarlas para conocer su potencial, correcto uso y servicios que brindan.

"El 4 de mayo del 2011, Red Hat anuncia su propia plataforma como servicio de código abierto OpenShift, la PaaS fue desarrollada y soportada por Red Hat y está comunidad" (Alexander L.,2014, p3). Esta herramienta soporta la mayoría de los lenguajes de programación, framework y bases de datos comúnmente usadas.

Situación Actual

"Durante mucho tiempo el concepto de Web 2.0 ha estado rodando los términos más comunes de la WWW (World Wide Web). En la actualidad el concepto más nuevo es el de la computación en la nube que es la tendencia de disponer archivos y aplicaciones directamente desde la Web." (Reyna, 2013).

Existen muchas opciones en el mercado de computación en la nube, sean o no de código abierto, como es de esperar cada uno con su propia manera de llevar sus servicios, presentación, características y estructura que ofrecen en todo lo que respecta a productos de computación en la nube, lo que quiere decir que brindan un ambiente basado en una nube de cómputo, donde es posible desarrollar, probar, ejecutar y desplegar las aplicaciones que sean necesarias para una persona o una organización, lo que da el beneficio de usarla sin la necesidad de conocer la infraestructura que existe detrás o preocuparse de la compleja tarea de comprar, instalar y poner en marcha una infraestructura.

"En los últimos 10 años la tendencia de guardar o almacenar en algún lugar la información es la constante de las empresas, por eso cada vez la distancia se acorta entre el usuario y la red de redes. Cada usuario que usa un ordenador tendrá que usar algún tipo de aplicación de ofimática y utilidades que probablemente no tenga instalado en su computador, por lo cual esta teoría viene a revolucionar el mundo de la información." (Reyna, 2013).

[2] Eucalyptus es una infraestructura (plataforma) de código abierto para la implementación de computación en nube privada en clúster de ordenadores.
[3] OpenNebula es una plataforma para computación en la nube orientado a centros de datos distribuidos y heterogéneos, proporcionando la infraestructura como servicio (IaaS).

Entre tantas opciones en el mercado es difícil la elección de una de ellas basándose en razones estadísticas o de experiencia para poder estar seguros de la elección tomada, para que esto sea posible hay que analizar las nuevas propuestas del mercado realizando estudios de estas para poseer información suficiente teniendo el conocimiento de cuáles son las herramientas, características y ventajas como rendimiento y escalabilidad que conlleven a elegir una u otra alternativa.

Prospectiva

El estudio de la plataforma como servicio OpenShift[4] permitirá tener un conocimiento más amplio de todo lo que esta herramienta posee entre su estructura, arquitectura, herramientas, características y componentes, para poder implementar una nube privada aprovechando todas las ventajas que puede ofrecer sobre las estructuras normalmente usadas, lo que permitirá tener la información más organizada, un despliegue, gestión, control y monitoreo de las aplicaciones mucho más eficiente.

Se tendrá la información suficiente para poder tener un conocimiento teórico para seleccionar una herramienta a ser usada si se presenta la necesidad de crear o poseer una nube privada ya sea para una persona, organización o para cualquier estudio posterior que se pretenda realizar como análisis de herramientas o comparativas de la plataforma estudiada con cualquier otra respecto a su arquitectura o a su funcionamiento.

Planteamiento del Problema

Actualmente existe una gran cantidad de usuarios que usan la tecnología de nubes de cómputo, especialmente empresas que brindan sus servicios a través del internet. No se posee un estudio que analice las diferentes herramientas que ofrece este servicio sea como plataforma, software o infraestructura. En el grupo de las plataformas como servicio existen muchas herramientas enormemente usadas, entre las cuales se destaca OpenShift, que a pesar de ser una de las más destacadas en este ámbito, se desconoce su estructura, arquitectura, herramientas, características y componentes lo que limita el provecho y alcance que se puede tener de esta herramienta.

[4] OpenShift es un producto de computación en la nube de plataforma como servicio de Red Hat. Los desarrolladores pueden usar Git para desplegar sus aplicaciones Web en los diferentes lenguajes de la plataforma.

El desconocimiento de esta información también dificulta en gran parte el estudio de la herramienta OpenShift; se propone un estudio de la correcta instalación y los entornos que se deben usar para su correcto funcionamiento, así como el despliegue de aplicaciones en la plataforma como servicio OpenShift.

Objetivos

Objetivo General

Estudiar la plataforma como servicio OpenShift Origin como alternativa para la implementación de una nube de cómputo privada.

Objetivos Específicos

- Identificar las bases de conocimiento para la implementación de la plataforma como servicio OpenShift Origin su estructura, arquitectura, herramientas, características y componentes además del despliegue, monitoreo y control de aplicaciones.

- Diseñar una aplicación web con el lenguaje PHP y el servidor de base de datos PostgreSQL para desplegarla en la plataforma como servicio OpenShift Origin.

- Evaluar el rendimiento y tiempos de respuesta que tiene las aplicaciones desplegadas en la plataforma como servicio OpenShift en una nube privada, en comparación con aplicaciones desplegadas en servidores tradicionales.

- Validar los resultados del estudio de la plataforma como servicio OpenShift Origin para mejorar la adecuada implementación, control, gestión, despliegue y monitorización de aplicaciones en una nube de cómputo privada.

Alcance

El alcance de este proyecto se define desde la instalación de la plataforma como servicio OpenShift Origin en un servidor local con un sistema operativo Linux como una nube de cómputo privada como se muestra en la Figura 1.

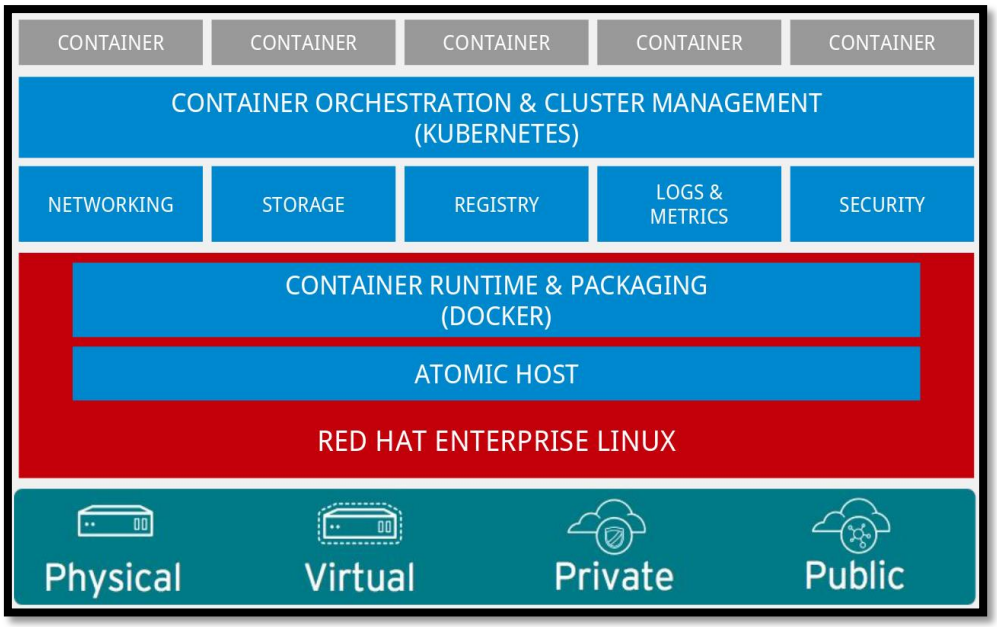

Figura 1: Arquitectura de OpenShift Origin

Fuente: página oficial de OpenShift: www.openshift.org

También se realizará el estudio de la estructura, arquitectura, herramientas, características y componentes que brinda y posee la plataforma como servicio OpenShift Origin.

En la nube creada se desplegará una aplicación Web de gestión de becas para estudiantes desarrollada bajo la metodología de desarrollo SCRUM en el lenguaje de programación PHP en conjunto con el servidor de base de datos PostgreSQL, esta aplicación se desarrollará bajo la arquitectura representada en la Figura 2.

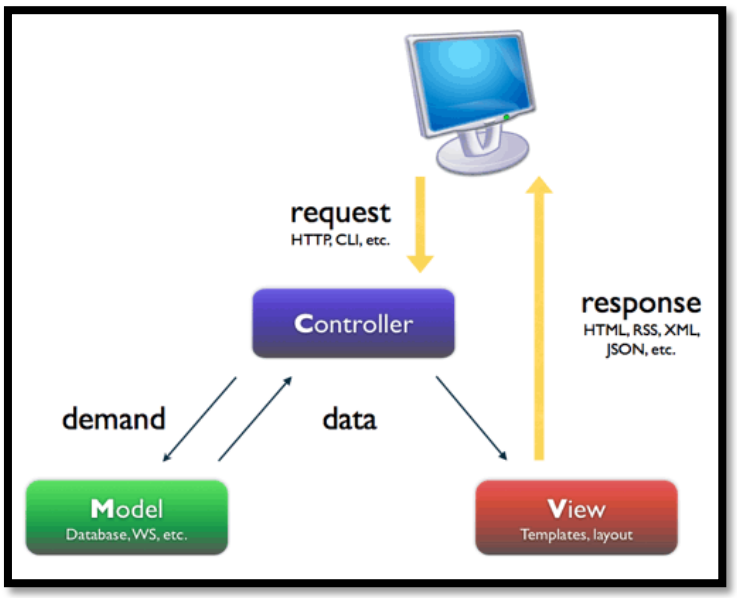

Figura 2: Arquitectura de la Aplicación de Prueba

Fuente: libro tutorial de Jobbet 2012

La aplicación desarrollada contendrá las opciones de gestión de becas de los estudiantes de la Facultad de Ingeniería en ciencias aplicadas de la Universidad Técnica del Norte del período académico 2015 – 2016 y registro de los estudiantes que obtuvieron cada una de ellas.

Justificación

Con la realización de este trabajo de investigación se beneficiará en un alto porcentaje a los usuarios en razón del actual incremento que existe de aplicaciones desplegadas en nubes de cómputo, como empresas que brinden sus servicios por internet que utilicen este trabajo de grado como fuente de conocimiento, en especial a los estudiantes de la carrera de Ingeniería en sistemas computacionales, ya que brindara una información confiable, para su posterior implementación, una forma de resolver sus necesidades o como medio de búsqueda de innovación por medio de la herramienta estudiada.

Analizar una nueva propuesta de plataforma como servicio de código libre, que permita la implementación de nubes privadas y el despliegue de aplicaciones en ellas para las organizaciones que lo necesiten, va a ayudar centralizar y monitorear de una manera más eficaz y eficiente que otras tecnologías usadas para este mismo fin.

CAPÍTULO 1
Revisión Bibliográfica

1.1. Computación en la nube

1.1.1. Qué es la computación en la nube

La computación en la nube es un servicio que posee muchas definiciones, ya que es el resultado de la colaboración de varias disciplinas de las ciencias de la computación, es necesario entonces conocer algunas de esas definiciones para poder comprender en qué se basa la computación en la nube, por esa razón a continuación, se plantearán algunos conceptos que se le han sido atribuido a la computación en la nube (Cloud Computing) por diversos autores a lo largo del tiempo desde su aparición.

"Computación en la nube definida como, un modelo de entrega y acceso donde se puede escalar dinámicamente los recursos y se ofrecen como un servicio a través de Internet. La computación en la nube proporciona un cambio de paradigma de las empresas y las tecnologías de la información, donde la potencia informática, el almacenamiento de datos y los servicios, se subcontratan a terceros y se ponen a disposición como productos para empresas y clientes". (Ruparelia, 2016).

"La computación en la nube representa un avance en el campo de la tecnología informática donde la gestión de los servicios informáticos se subcontrata en mayor medida y en múltiples dimensiones. Para ser específicos, las instalaciones informáticas están disponibles como servicios ofrecidos por vendedores con alta reputación." (Bhowmik, 2017).

"La computación en la nube proporciona los medios para que los usuarios utilicen fácilmente las instalaciones informáticas cuando sea y donde sea requerido. No tienen que preocuparse por la configuración de la infraestructura, adquirir un equipo nuevo o invertir en la adquisición de software con licencia. Más bien pueden acceder a cualquier volumen, grande o pequeño, de instalaciones informáticas a cambio de algún pago nominal." (Bhowmik, 2017).

"El término nube que se incluye en la computación en la nube se refiere a los medios a través de los cuales se accede a los servicios contratados: poder de cómputo para la infraestructura informática, aplicaciones, procesos de negocios, y colaboración personal,

pueden ser entregados al usuario final, como servicio donde y cuando lo necesite. Una nube es un grupo de servidores de red interconectados o computadores que pueden ser privados o públicos.” (Chopra, 2017).

Uno de los organismos más reconocidos como es el Instituto Nacional de Estándares y Tecnología, que es una agencia del Departamento de Comercio de los EE. UU., conocido como NIST por sus siglas en inglés, da la siguiente definición de computación en la nube:

“La computación en la nube es una manera de permitir a una red ser difundida prácticamente en cualquier lugar, bajo demanda, y conveniente, además que brinde acceso a un grupo compartido de recursos informáticos configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que se pueden visionar y publicar rápidamente con un mínimo de esfuerzo de gestión o interacción con el proveedor del servicio.” (Huang & Wu, 2017).

Como se puede observar, en todos los conceptos que se han planteado anteriormente de diferentes libros y autores, todos tienen algo en común y es que la computación en la nube es la contratación de servicios por medio de internet como una plataforma, una infraestructura o simplemente un sistema o software, sin preocuparse de la parte administrativa que implica, esto realizado sin un mayor esfuerzo e interacción con la empresa o persona que brinda el servicio.

Se podría realizar una analogía para comprender qué es la computación en la nube, sería como comprar un espacio de tierra o un edificio y poder alquilarlo por partes, dependiendo de cuál sea la necesidad de los clientes finales que vendrían a ser los inquilinos los cuales pagarían solamente por los espacios en el lugar que ocuparían, cuando se habla de computación en la nube es similar, el proveedor mantiene su servicio en internet, que será usado por los clientes pagando únicamente por la cantidad que necesite de aquel servicio.

Hasta ahora, la descripción operacional más simple de la computación en la nube sería: "paradigma de la computación donde los servicios y los datos residen en recursos compartidos, en centros de datos escalables, y esos servicios y datos son accesibles por cualquier dispositivo autenticado en Internet” (Khalid, 2010).

1.1.2. Historia y evolución de la computación en la nube

"Podemos identificar Internet como una red de redes de todo el mundo, dado que, en sentido amplio, la nube utiliza Internet como medio de abastecimiento, El término "nube" es una metáfora de Internet. Una mejor comprensión de la historia de la computación en la nube se puede entender conociendo la combinación de tecnologías que evolucionaron hacia la nube" (Alani, 2016).

A continuación, se presenta algunos hechos relevantes en la historia de la computación en la nube, recopilados de diferentes textos en los que se encontró información sobre este tema.

La empresa pionera en nubes de cómputo es Salesforce.com, que se estableció en 1999. Esta empresa proporciona las aplicaciones de nivel empresarial a los clientes a través de Internet. Después de Salesforce.com, en 2002, Amazon.com[5] comenzó sus servicios minoristas basados en la web. Amazon había modernizado sus centros de infraestructura de datos, pero solo el 10% de su capacidad se utilizó en un momento dado. En 2005, Eze Castle[6] construyó e implementó la primera plataforma alojada de computación en la nube. En 2006, Google surgió como uno de los principales proveedores de servicios en la nube informática mediante el lanzamiento de Google Docs Services.

En 2009, la Web 2.0 creó una nueva marca en la computación en la nube. Durante este período, Google, Microsoft y otras compañías proporcionaron una empresa basada en internet, con servicios de aplicaciones para los usuarios finales. Google proporciona sus aplicaciones a través de la nube de Google Apps.

El año 2011 es un año extraordinario para marcar revolución en la computación en la nube. Numerosos proveedores de servicios comenzaron a ofrecer servicios de computación en la nube. En 2012, Salesforce.com expone Government Cloud y AppExchange, servicios multiusuario diseñados para el sector público. En 2013, la CIA[7] ocupó de Amazon Web Services para construir una nube privada, esto ayudó a aumentar la confianza de los servicios en la nube con respecto a las preocupaciones de seguridad. (K, 2015, p. 4).

[5] Amazon, Inc es una compañía estadounidense de comercio electrónico y servicios de computación en la nube a todos los niveles.
[6] Eze Castle Consulting proporciona infraestructura de TI y servicios tecnológicos, incluyen archivado, copia de seguridad y recuperación de desastres.
[7] CIA. Agencia Central de Inteligencia, Es una de las mayores agencias de inteligencia del Gobierno de EEUU.

La historia de la computación en la nube se remonta a principios de 1960, Cuando John McCarthy[8] propuso el modelo de computación como servicio público. El primer uso público del término "Nube" como símbolo de Internet, apareció en el periódico MIT publicado, en 1996. En 1999, el primer paso hacia este término moderno fue tomado por Salesforce.com, que lideró el camino de la entrega de aplicaciones comerciales a través de una web. En 2002, Amazon introdujo una colección de servicios basados en la nube que comprende almacenamiento y computación a través de la red de datos de Amazon. Mientras tanto, IBM[9] adoptó este modelo en sus aplicaciones y mostró los nuevos métodos de computación (computación generalizada, automatizada y de utilidad). La primera aplicación web ampliamente accesible fue Elastic Compute Cloud (EC2) de Amazon, como un servicio web comercial que permiten arrendar a negocios pequeños e individuos, computadoras en las que pueden disfrutar de sus propias aplicaciones. En 2007, Google, International Business Machines (IBM) y un gran número de universidades obtienen una nube a gran escala, como proyecto de investigación informática, a mediados de 2008, la computación en la nube alcanzó el estado en los medios, y muchos procedimientos interconectados empezaron a desarrollarse. (Khalid, 2010, p.1).

El concepto general de computación en la nube, aunque tenía un nombre diferente se remonta a 1961. Un conocido científico informático llamado John McCarthy declaró, en el centenario de MIT[10] "Las computadoras del tipo que he defendido se convierten en las computadoras del futuro, luego la informática podría organizarse algún día como una utilidad pública, del mismo modo que el sistema telefónico es una utilidad pública. La utilidad informática podría convertirse en la base de una industria nueva e importante." El término computación de utilidad se refiere a un servicio de computación a pedido, que puede ser utilizado por el público con un modelo financiero de pago solamente por lo que utiliza. El término ha sido evolucionando desde entonces.

La idea fue madurada ligeramente antes de finales de la década de 1990 cuando Salesforce.com introdujo el primer servicio aprovisionado a distancia para la empresa, luego, el concepto comenzó a ser diferente a fines de la década de 1990. Los conceptos luego se enfocaron en una capa de abstracción utilizada para facilitar los métodos de entrega de datos en paquetes conmutados y redes heterogéneas.

[8] John McCarthy, prominente informático que recibió el Premio Turing en 1971 por sus importantes contribuciones en el campo de la Inteligencia Artificial.
[9] (IBM) empresa multinacional de tecnología y consultoría, fabrica y comercializa hardware y software.
[10] (MIT por las iniciales de su nombre en idioma inglés, Massachusetts Institute of Technology) es una universidad privada localizada en Cambridge en donde se exponen desarrollos tecnológicos.

En 2002, Amazon.com presentó la plataforma Amazon Web Services (AWS). La plataforma, en aquel entonces, proporcionaba almacenamiento y recursos informáticos provisionados a distancia. Comercialmente, el término "computación en la nube" surgió cuando Amazon lanzó sus servicios Elastic Compute Cloud (EC2). El modelo de servicio se basó sobre "arrendamiento" de potencia y almacenamiento de procesamiento informático elástico donde las empresas pueden ejecutar sus aplicaciones. Más tarde ese año, Google también comenzó a proporcionar Google Apps. (Alani, 2016, pp. 2–3).

Con la inspección de los anteriores textos de diferentes autores, se tiene los años en los que se han suscitado hechos relevantemente históricos, desde el nacimiento del término nube hasta el completo desarrollo de su funcionamiento actual, para llegar a lo que en el momento se conoce como la computación en la nube, a partir de la información previamente expuesta se pretende realizar una tabla en la que se resuma y organice la información para obtener una mejor comprensión de los sucesos que han ocurrido en tiempo pasado, respecto a la computación en la nube.

En la Tabla 1 se resume los diferentes años y sus respectivos acontecimientos que marcaron la historia de la computación en la nube.

Tabla 1: Historia de la computación en la nube

AÑO	ACONTECIMIENTOS
1960	John McCarthy propuso el modelo de computación como servicio público. Primer uso público del término "Nube" como símbolo de Internet.
1961	John McCarthy declaró, en el centenario de MIT “luego la informática podría organizarse algún día como una utilidad pública”
1996	Primer uso público del término "Nube" como símbolo de Internet, apareció en el periódico de MIT.
1999	Salesforce.com, lideró el camino de la entrega de aplicaciones comerciales a través de Internet.
2002	Amazon.com presentó la plataforma Amazon Web Services (AWS). La plataforma, en aquel entonces, proporcionaba almacenamiento y recursos informáticos brindados a distancia.

2005	Eze Castle construyó e implementó la primera plataforma alojada de computación en la nube.
2006	Google surgió como uno de los principales proveedores de servicios en la nube informática mediante el lanzamiento de Google Docs Services.
2007	Google, IBM y un gran número de universidades obtienen una nube a gran escala, como proyecto de investigación informática.
2008	La computación en la nube alcanzó el estado en los medios, y muchos procedimientos interconectados empezaron a desarrollarse.
2009	Google, Microsoft y otras compañías proporcionaron una empresa basada en navegador como aplicaciones para usuarios finales. Google proporciona su aplicación a través de la nube de Google Apps.
2011	Numerosos proveedores de servicios comenzaron a ofrecer servicios de computación en la nube.
2012	Salesforce.com expone Government Cloud y AppExchange, servicios multiusuario diseñados para el sector público.
2013	La CIA ocupó de Amazon Web Services para construir una nube privada, esto ayudó a aumentar la confianza de los servicios en la nube con respecto a las preocupaciones de seguridad.

Fuente: Propia

Como se puede observar la computación en la nube se desarrolla con la finalidad de lograr que la computación sea visualizada como un servicio que se brinda a través de internet, que sea considerado como una utilidad global como lo era en ese entonces la telefonía pública, el término nube se usa como analogía a lo que representa, el internet al ser difundido en prácticamente todos los lugares y por el ser intangible, fue mediante muchos años de estudios y evolución que se logró tener la tecnología actual que se conoce como computación en la nube.

1.1.3. Ecuador y la computación en la nube

En todos los países del mundo la tecnología se ha convertido en un aspecto muy importante a ser mejorado y adquirido, la tecnología más sobresaliente entre las muchas ha sido el internet, en el Ecuador no se queda atrás cada vez son más los lugares y hogares donde se puede tener acceso al internet y así poder conectarse a los diferentes servicios que se disponen en la web, como las redes sociales, banca móvil, correos electrónicos, almacenamiento de datos o consulta de información, todos estos servicios actualmente forman parte de todo lo que es la computación en la nube, y la forma más fácil y común de hacer uso de la nube es por medio del internet.

El Ecuador posee el Instituto Nacional de Estadística y Censos (INEC) el cual puede brindar información detallada y real de las cifras que se manejan en el país en diferentes ámbitos, una de ellas la de Tecnologías de la Información y Comunicaciones (TIC's), de la cual se tiene información del año 2016 al ser la última realizada por el instituto, de donde se puede obtener la siguiente información, con estos antecedentes:

La encuesta fue realizada con una cobertura nacional, regional, urbana y rural, con un número total de viviendas de 31.092, en el estudio se tuvo como informante del hogar al jefe de familia, la población objetivo de la encuesta fue de todas las personas de 5 años o más de edad, la encuesta hace referencia a diciembre del 2016. (INEC, 2017).

Un índice importante que se puede tomar en cuenta como punto de partida, es las personas que disponen de un equipo por el cual acceder al internet y la nube, a lo cual el INEC indica que, ha habido un incremento considerable en la disponibilidad de una computadora portátil sobre las computadoras de escritorio, con una disminución de 1% del año 2015 al 2016 en computadoras de escritorio y un aumento del 2.8% en la disponibilidad de una computadora portátil. (INEC, 2017).

En la Figura 3 se representa los cambios que se han suscitado en la disponibilidad de una computadora portátil sobre las computadoras de escritorio en los últimos años.

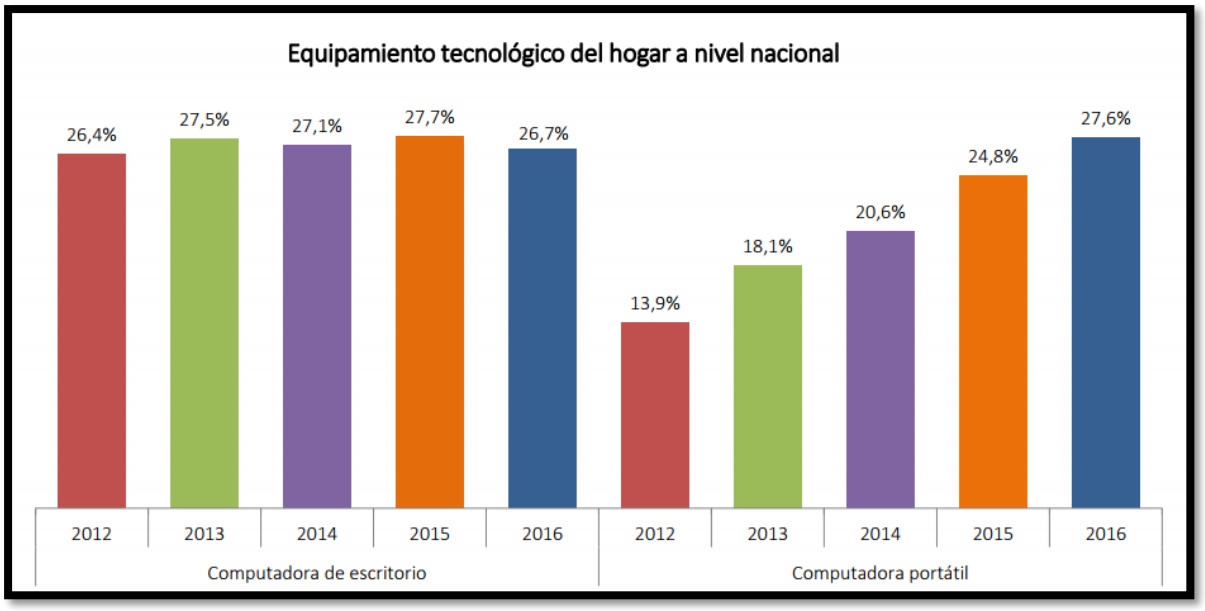

Figura 3: Equipamiento tecnológico del hogar a nivel nacional
Fuente: INEC

El acceso al internet en el ecuador se representa en la Figura 4 en donde se aprecia que se ha obtenido un aumento en el acceso al internet del 3.6% en el área urbana, y del 2.7% en el área rural entre los años 2015 al 2016. (INEC, 2017).

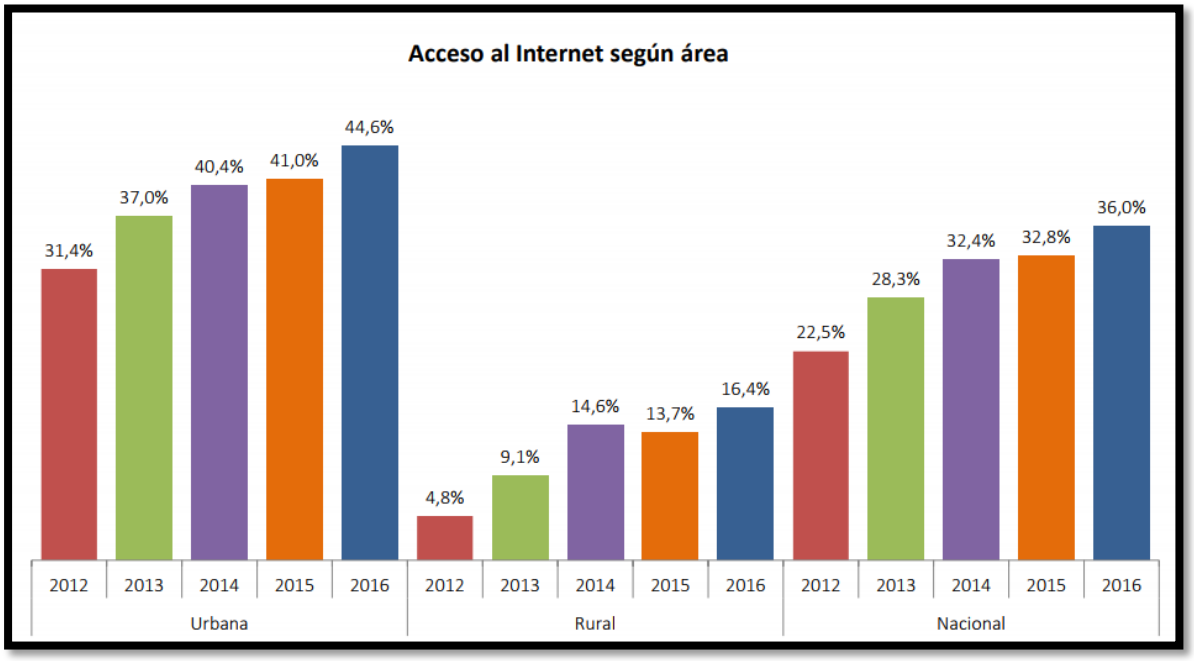

Figura 4: Acceso al internet según área
Fuente: INEC

Se interpreta en la Figura 5 que 8 de cada 10 jóvenes entre 16 y 24 años usaron internet en 2016, le sigue el grupo entre 25 y 34 años con el 67,3% de su población. Siendo estas unas cifras realmente altas. (INEC, 2017).

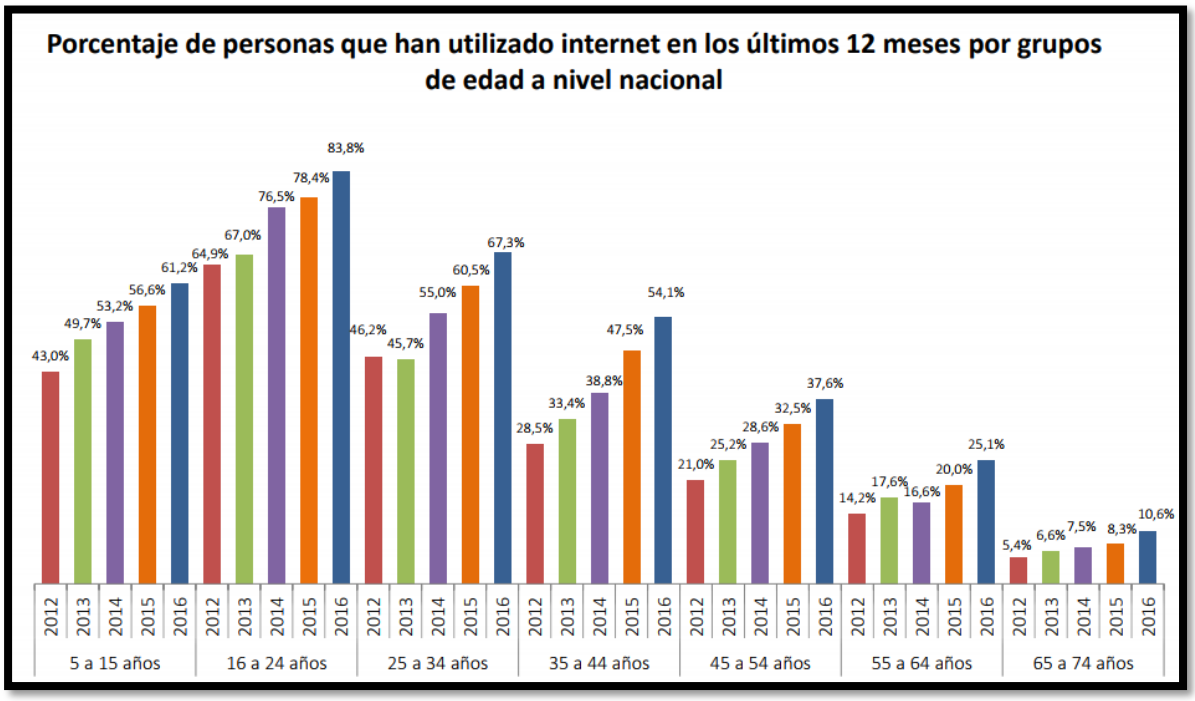

Figura 5: Porcentaje de personas que han utilizado internet en los últimos 12 meses por grupos de edad a nivel nacional
Fuente: INEC

Por último, de esta encuesta se puede obtener unos resultados importantes en lo que respecta al uso que le da la población ecuatoriana al internet en lo que los resultados del INEC arrojan que, en el 2016 a nivel nacional, el 38,0% de las personas usó Internet como fuente de información, mientras el 31,5% lo utilizó como medio de comunicación en general, en lo cual la computación en la nube podría entrar en prácticamente todas estas ramas o usos del internet. (INEC, 2017). En la Figura 6 se representa el uso que le da la población ecuatoriana al internet en los últimos años.

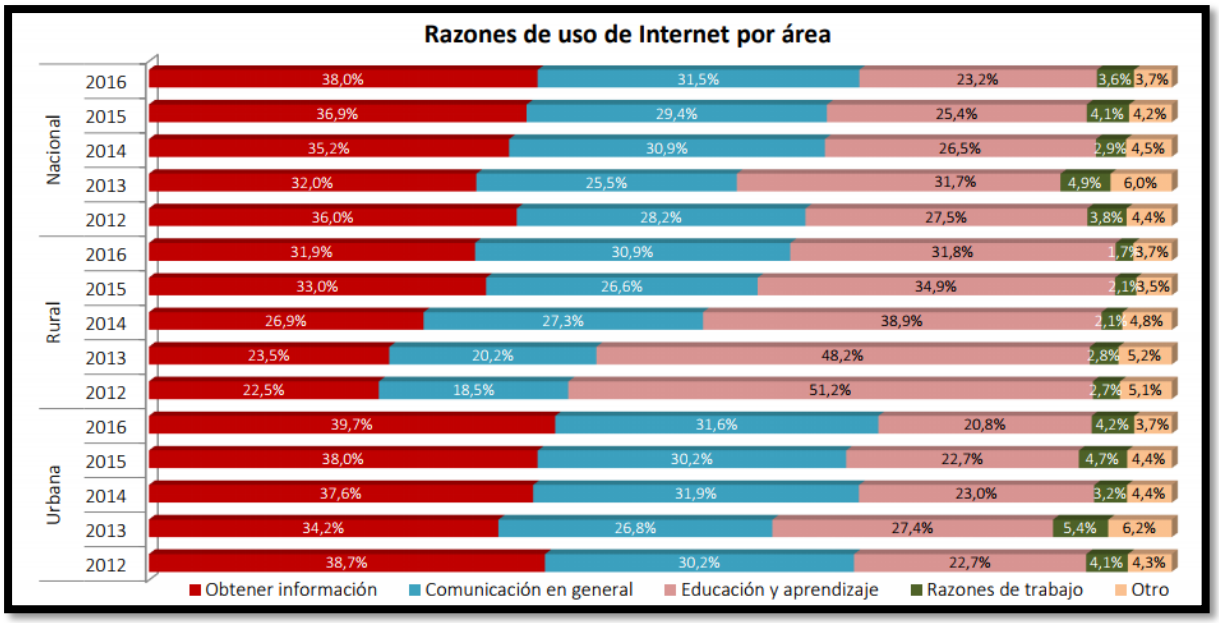

Figura 6: Razones de uso de Internet por área
Fuente: INEC

Se ha encontrado información relevante de la misma manera en la Revista Líderes del Ecuador, una revista del periódico el comercio, en algunos de sus artículos destaca la importancia que ha tenido la computación en la nube sobre algunos proyectos que se han desarrollado en el país, así como las razones que han llegado a optar por una nube como solución.

Artículo publicado en la revista líderes el 22 de abril del 2013, plantea Un sistema contable 100% en línea para todo tipo de empresas y para personas naturales obligadas a llevar contabilidad. Esta es la oferta de Contífico[11], un negocio que tiene casi tres años en el mercado y una cartera de más de 300 clientes. La idea de crear Contífico la tuvo el guayaquileño Óscar Plaza, (…). El software está ligado con lo que se denomina computación en la nube, es decir con la tecnología de alojamiento en la nube y acceso remoto. Se puede utilizar desde una computadora de escritorio o portátil, una tableta o un smartphone.

En el artículo publicado en la revista líderes el 17 de mayo del 2016, informa que ocho servicios corporativos son parte de la nueva propuesta de la operadora Claro. Mediante la plataforma Claro Cloud la firma brindará: servidores virtuales; data center digital, (…). Esta nueva línea de negocios de Claro se enfoca en pequeñas, medianas y grandes empresas del país. El desarrollo de Claro Cloud implicó una inversión de USD 50 millones explicó Gilma Méndez, gerente de Productos y Servicios Corporativos para Claro Ecuador. El objetivo es

[11] Contífico es un sistema propietario desarrollado en lenguaje Python desde. Se comercializa en una modalidad SaaS o software como.

que estas plataformas se operen desde la nube y responde a las nuevas necesidades de las empresas.

En el artículo publicado en la revista líderes el 27 de diciembre del 2016, Rafael Parreño, gerente de Negocios de IBM Ecuador, habla sobre innovación en el sector bancario y cuáles son los nuevos retos que debe afrontar este segmento en los próximos años. El buen uso de datos de los clientes será una de las claves para la banca. Al hablar de qué servicios ofrece IBM actualmente en innovación para la banca Parreño respondió, que en todos los frentes que poseen se tiene muchas plataformas, pero los dos pilares vienen fundamentados en la nube por el mismo hecho de que los bancos deben ser eficientes en soluciones cognitivas, porque tanto en el tema de conocimiento del cliente, en analítica, de monetizar información, de seguridad, de cómo ir mejorando el servicio en las plataformas de pago o en los canales que tienen los bancos.

En el artículo publicado en la revista líderes el 16 de noviembre del 2016, hace 20 años, el ambiente tecnológico en Ecuador y el mundo era otro. Una conexión a Internet era un privilegio de pocos, al igual que los teléfonos móviles. A escala global la situación no era muy diferente. Pero a partir de la primera década de este siglo el panorama tecnológico cambió para siempre. Una larga lista de innovaciones salió a la luz y transformó la manera de entender el mundo y de conectar personas y negocios. Una de las innovaciones de las que se habla aquí en la computación en la nube, se hace mención de ella con: En si la nube, permite almacenar la información particular de una persona o empresa en servidores que están al alcance de los usuarios previo la validación de sus datos y de su contraseña. De esta manera cada usuario puede abrir, revisar y hasta modificar la información o usar sus programas, aunque estos no estén utilizando su computadora personal o del trabajo. Redes sociales como Facebook, Gmail, Pinterest y más almacenan la información de sus usuarios en la nube. Esto permite que acceder a las cuentas desde cualquier dispositivo.

La computación en la nube en el Ecuador, con proyectos y emprendimientos ha ido llenando cada uno de los vacíos que las empresas grandes, medianas y pequeñas han observado en el funcionamiento de sus servicios, empresas de las más importantes no solo en el Ecuador sino a nivel mundial han implementado nubes para los servicios que brindan en este país, como son Claro e IBM, entre otras empresas nacionales que han visto en la computación en la nube una manera eficaz de solucionar ágilmente problemas del negocio, en diferentes ámbitos como el económico y el tecnológico, de esta forma se han ido agregando servicios basados en la nube en el Ecuador, a pesar de que muchos servicios basados en la nube que no están implementados en el país son muy usados por los

ecuatorianos, muchas veces sin que se conozca de esto, lo cierto es que la computación en la nube y los servicios que esta ofrece están implementados por todo el mundo y el Ecuador no es la excepción.

1.1.4. Características de la computación en la nube.

La computación en la nube posee características que la hacen diferente a la computación normal, por el hecho de la analogía con una nube que se utiliza para su nombre, al ser una fuente que brinda servicios y almacenamiento prácticamente ilimitados a la necesidad, para cumplir con los requisitos de los usuarios finales, la computación en la nube debe cumplir ciertos requisitos esenciales para que pueda proporcionar servicios cualitativos eficientes, las características que posee la computación en la nube son las siguientes.

El Instituto Nacional de Estándares y Tecnología, que es una agencia del Departamento de Comercio de los EE. UU., conocido como NIST por sus siglas en inglés, en su definición de computación en la nube, se compone de cinco características esenciales, tres servicios y cuatro modelos de implementación. Las características esenciales son:

a) **Autoservicio bajo demanda:** Un consumidor puede aprovisionar unilateralmente capacidades informáticas, como el tiempo del servidor y el almacenamiento en red, según sea necesario, automáticamente sin requerir interacción del hombre con cada proveedor de servicios.

b) **Múltiples formas de acceder a la red:** Las capacidades están disponibles en la red y se puede acceder a través de mecanismos estándar que promueven el uso de plataformas de clientes heterogéneas delgadas o gruesas (por ejemplo, teléfonos móviles, tabletas, computadoras portátiles y estaciones de trabajo).

c) **Compartición de recursos:** Los recursos informáticos del proveedor se agrupan para servir a múltiples consumidores utilizando un modelo multiusuario, con diferentes recursos físicos y virtuales asignados dinámicamente y reasignados de acuerdo con la demanda del consumidor. Existe una sensación de independencia de ubicación en el sentido de que el cliente no tiene control o conocimiento sobre la ubicación exacta de los recursos proporcionados, pero puede ser capaz de especificar la ubicación en un nivel más alto de abstracción (por ejemplo, país, estado o centro de datos). Ejemplos de recursos incluyen almacenamiento, procesamiento. memoria y ancho de banda de la red.

d) Elasticidad rápida: Las capacidades se pueden aprovisionar y liberar de forma elástica, en algunos casos automáticamente, para escalar rápidamente hacia afuera y hacia adentro de acuerdo con la demanda del consumidor, las capacidades disponibles para aprovisionamiento suelen seguir siendo ilimitadas y puede ser apropiado en cualquier cantidad en cualquier momento.

e) Servicio medido: Los sistemas en la nube controlan y optimizan automáticamente el uso de los recursos mediante la medición del aprovechamiento y capacidad en algún nivel de abstracción apropiado para el tipo de servicio (por ejemplo, almacenamiento, procesamiento, ancho de banda y cuentas de usuario activas). El uso de recursos se puede monitorear, controlar y aplicar, proporcionando transparencia tanto para el proveedor y consumidor del servicio utilizado. (Huang & Wu, 2017).

La Tabla 2 muestra un análisis de las características que ofrece la computación en la nube.

Tabla 2: Análisis de las características de la computación en la nube

Característica	Descripción
Amplio acceso a la red	Un consumidor debe poder acceder a los servicios desde cualquier lugar.
Compartición de recursos	Los recursos informáticos de un proveedor se agrupan para admitir múltiples clientes.
Autoservicio a pedido	Un consumidor debe ser capaz de aprovisionar recursos informáticos (como servidores virtuales) según sea necesario, con una interacción humana mínima.
Servicio medido	Un consumidor debe ser capaz de usar los recursos informáticos en función del pago como uso.
Elasticidad	Un consumidor debería ser capaz de aprovisionar recursos adicionales de forma automática y bajo demanda. Para garantizar esto, el proveedor reúne recursos informáticos para proporcionar escalabilidad horizontal al consumidor.

Fuente: (Mishra, 2017)

Como se puede observar en las características expuestas por el NIST, se podría considerar como unas grandes ventajas para los usuarios finales, en razón de que estas características buscan únicamente facilidad para el usuario, como lo es que el usuario pueda

acceder a sus servicios desde múltiples dispositivos solamente como único requisito el poseer una conexión a internet, el poder asignar más recursos o capacidad a su infraestructura simplemente con solicitarlo, sin necesidad de comprar el respectivo hardware, realizando el pago por lo que está consumiendo en el tiempo que lo haga, esto reduce costos notablemente en cualquier empresa de espacio e infraestructura, a pesar de que el usuario no conoce donde está físicamente su información, la sensación de acceso inmediato le da la confianza necesaria, y por último la elasticidad que le brinda, pudiendo expandir su capacidad prácticamente de forma ilimitada, estas características fueron las que sin duda alguna, dieron paso a que la computación en la nube se adueñara de la industria de servicios informáticos, dejando de lado a la computación tradicional.

1.1.5. Modelos de implementación de la computación en la nube

"Un modelo de implementación responde las siguientes preguntas: ¿Quién puede acceder a un recurso informático?, ¿Cómo puede un usuario acceder a un recurso informático?, ¿Dónde está el hardware físico?"(Mishra, 2017).

"La definición de NIST contiene cuatro modelos de implementación distintos: nubes públicas, privadas, comunitarias e híbridas." (V.K, 2015, p. 4)

- **Nube pública**

"Cuando las infraestructuras proporcionadas por una nube son controladas y operadas por un proveedor de nube y además son de su propiedad, entonces esa nube se llama nube pública. Ejemplos de tales servicios son sitios de redes sociales o correo electrónico. La nube pública también se utiliza para proporcionar servicios para negocios."(V.K, 2015, p. 4).

"En general, estas nubes ofrecen servicios a través de Internet y están operados por un proveedor de servicios en la nube. Por ejemplo, servicios de correo electrónico, sitios de redes sociales, etc., están todos dirigidos al público en general." (Chopra, 2017, p. 19).

- **Nube privada**

"La infraestructura de la nube se aprovisiona para uso exclusivo de una única organización que comprende múltiples consumidores (por ejemplo, unidades de negocio). Puede ser propiedad, administrada, y operada por la organización, un tercero o una

combinación de ellos, y puede existir en o fuera de las instalaciones.” (Huang & Wu, 2017, p. 8).

“Cuando la infraestructura de la nube se opera exclusivamente para algunos particulares de la organización, y son controladas y administradas por esa organización o por un tercero, entonces esa nube se llama nube privada.” (V.K, 2015, p. 7).

- **Nube comunitaria**

“Cuando el servicio y la infraestructura son utilizados por muchas organizaciones, y son accesibles solo para esas organizaciones, entonces esa nube se llama nube de la comunidad. En nube comunitaria, la infraestructura puede ser administrada, controlada y operada por un proveedor de servicios en la nube o por las organizaciones.” (V.K, 2015, p. 7).

La infraestructura de la nube se aprovisiona para uso exclusivo por parte de una comunidad de consumidores de organizaciones que han compartido inquietudes (por ejemplo, misión, requerimientos de seguridad. política y consideraciones de cumplimiento). Puede ser propiedad, administrado y operado por una o más de las organizaciones en la comunidad, un tercero, o alguna combinación de ellos, y puede existir dentro o fuera de las instalaciones.” (Huang & Wu, 2017, p. 8).

- **Nube híbrida**

“La infraestructura de la nube es una composición de dos o más infraestructuras de nube distintas (privada, comunitaria o pública) que siguen siendo entidades únicas, pero están vinculadas juntas por tecnología estandarizada o patentada que permite datos y aplicaciones portables (por ejemplo, ruptura de nubes para equilibrar la carga entre nubes).”(Huang & Wu, 2017, p. 8).

“Cuando hay una agrupación de nubes diferentes, esa nube se llama nube híbrida. Eso exige más funcionalidad, lo que debería considerarse en el diseño de los sistemas de software. Estas funcionalidades son específicas y admiten la ejecución de aplicaciones en ambientes híbridos y dinámicos. Las nubes híbridas se desarrollan heterogéneamente utilizando recursos tales como infraestructuras virtuales privadas o públicas, clústeres y ordenadores.” (V.K, 2015).

En la Tabla 3 se exponen las diferencias entre una nube pública y una nube privada.

Tabla 3: Diferencias entre nubes públicas y privadas

Nube pública	**Nube privada**
Su propietario es el proveedor de la nube	Su dueño es únicamente la organización.
Implica costos más bajos.	Implica más costos.
La escalabilidad es bajo demanda e ilimitado.	La escalabilidad está limitada a la infraestructura instalada.
Menos seguridad.	Mayor seguridad.
Probarlo es difícil porque todo es público	Las pruebas son más fáciles porque es una nube privada.
El rendimiento es difícil de lograr.	Su rendimiento está garantizado.
Menos gestión y control es necesario porque funciona en el concepto de virtualización.	Se necesita más gestión y control porque tiene un mayor nivel de control sobre los recursos.

Fuente: (Chopra, 2017)

1.1.6. Modelos de servicio de la computación en la nube.

"Las computadoras y la informática se han convertido en una parte integral de nuestra vida cotidiana. Diferentes personas usan diferentes categorías de instalaciones informáticas." (Bhowmik, 2017).

Según la clasificación sugerida por el Instituto Nacional de Estándares y Tecnología (NIST), la computación en la nube se clasifica en modelos de implementación y modelos de servicio. Los modelos de servicio incluyen los tipos de servicios que están disponibles para el usuario en la plataforma de computación en la nube. Hay tres tipos de modelo de servicio, a saber, Infraestructura como un servicio (IaaS), plataforma como servicio (PaaS), software como servicio (SaaS). (K, 2015, p.6).

En la Figura 7 se representan los tres tipos de servicio y los cuatro modelos de implementación que brinda la computación en la nube además de la relación que existe entre ellos.

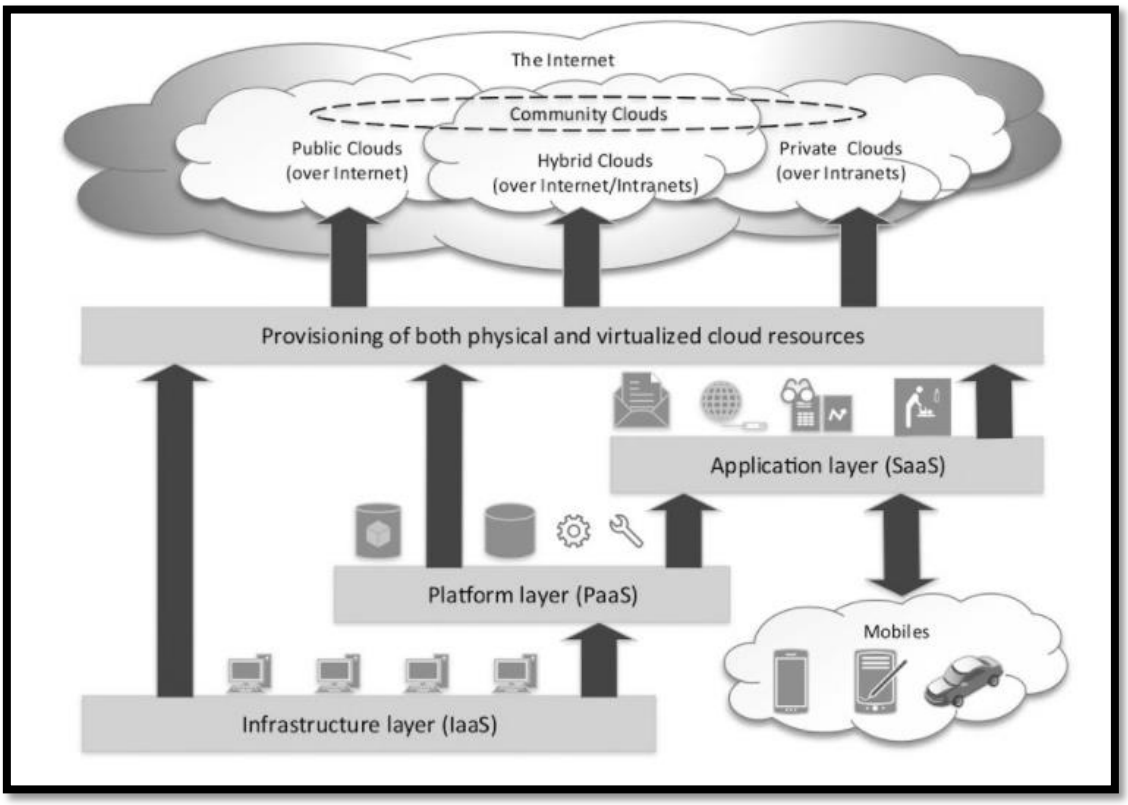

Figura 7: Clasificación de la computación en la nube
Fuente: (Huang & Wu, 2017)

Esta clasificación se basa en el nivel que el usuario necesita o requiere el control de la nube que va a contratar con su proveedor, dependiendo de eso estará haciendo uso de uno de estos modelos de servicio, a pesar de no saber cuál modelo utiliza en el momento.

- **Infraestructura como servicio (IaaS)**

“Usted especifica la esencia del servidor virtual que requiere, incluido la cantidad de CPU, RAM, espacio en disco duro, capacidades de red y sistema operativo. El proveedor de la nube ofrece una máquina virtual para que coincida con estos requisitos. Un escenario en el que se desea almacenar archivos en un disco duro virtual basado en la nube bajo el modelo de servicio IaaS.” (Mishra, 2017).

“El modelo de infraestructura como servicio ofrece hardware y soporte mínimo de software para usuarios que van a desarrollar una aplicación, por ejemplo, máquinas virtuales, almacenamiento virtual e infraestructura virtual. Todas las infraestructuras son administradas

por el proveedor de servicios en la nube. Los usuarios son responsables de la instalación y el control de las aplicaciones, el sistema operativo y las interacciones."(K, 2015, p.9)

En esa capa el usuario tiene el control de todo lo que contiene y se ejecuta dentro de su nube, mediante diferentes accesos que se manejan en este medio, simplemente con solicitarlo a su proveedor de acuerdo con su necesidad.

- **Plataforma como servicio (PaaS)**

El modelo de plataforma como servicio proporciona sistemas operativos, máquinas virtuales, servicios, aplicaciones, marcos para el desarrollo, estructuras de control y transacciones. En este modelo, los usuarios pueden implementar sus aplicaciones en la infraestructura disponible en el ambiente de la nube. Además, le permite al usuario usar esas aplicaciones que son desarrolladas utilizando las herramientas disponibles. En el modelo PaaS en la infraestructura en la nube, los sistemas operativos, y el software es administrado por el proveedor del servicio. Mientras, los usuarios son responsables de instalar y controlar las aplicaciones. (K, 2015, p.9).

La capacidad provista al consumidor es desplegar en la infraestructura en la nube aplicaciones desarrolladas o creadas por el consumidor utilizando idiomas, bibliotecas, servicios y herramientas compatibles con el proveedor. El consumidor no administra ni controla la infraestructura subyacente de la nube, incluida la red, servidores, sistemas operativos o almacenamiento, pero tiene control sobre las aplicaciones implementadas y posiblemente opciones de configuración para el entorno de alojamiento de la aplicación. (Huang & Wu, 2017).

En esta capa se brinda al usuario un entorno de desarrollo y un lugar donde se puede alojar las aplicaciones que necesite, se brindan las herramientas como lenguajes de programación, bases de datos y servidores más comúnmente utilizados para este fin.

- **Software como servicio (SaaS)**

La capacidad provista al consumidor es usar las aplicaciones del proveedor que se ejecutan en una infraestructura en la nube. Se puede acceder a las aplicaciones desde varios dispositivos cliente a través de una interfaz de usuario ligera, como un navegador (por ejemplo, correo electrónico basado en web) o una interfaz de programa. El consumidor no administra ni controla la infraestructura subyacente de la nube, incluida la red, los servidores,

los sistemas que operan, almacenamiento o incluso capacidades de aplicación individuales, con la posible excepción de configuraciones de aplicación específicas para el usuario. (Huang & Wu, 2017, p.7).

"El cliente o usuario es proporcionado de un acceso en base a su solicitud. Él no tiene control sobre el hardware, red, seguridad o sistema operativo." (Chopra, 2017).

Esta es la capa más superficial, en donde el usuario final solamente maneja las aplicaciones o programas que el proveedor le otorga en cuanto a su necesidad o su pedido, sin preocuparse en lo absoluto por la infraestructura que existe detrás de esta, simplemente posee algunas configuraciones personalizables a nivel de software.

Existen otros modelos de servicio que se le han atribuido a la computación en la nube, como el "Proceso empresarial como servicio BPaaS que especifica un proceso comercial que se quiere subcontratar de un proveedor de la nube" (Mishra, 2017), incluso existen autores que han clasificado los modelos de servicio como XaaS, lo que quiere decir, lo que el usuario necesite pero brindado como un servicio, sin embargo en esta investigación realizada solamente se tomará la clasificación dada por el Instituto Nacional de Estándares y Tecnología (NIST).

1.1.7. Ventajas y desventajas del uso de los servicios de la computación en la nube

- **Ventajas de la computación en la nube**

En todo momento la nueva tecnología trae consigo o desata en las personas o países que pretenden hacer uso de ella, un gran temor a lo desconocido, con incertidumbre de qué pros y contras traerá consigo, esto sucede con la computación en la nube, al ser una nueva tecnología sus usuarios temen de ella, especialmente por la seguridad de su información, en solución de este problema se analizarán y citarán las ventajas que posee la computación en la nube desde el punto operativo.

"Las soluciones que brinda la computación en la nube brindan principalmente dos grandes ventajas a las empresas, como son la gran reducción de costo que permite y la alta disponibilidad que ofrece, estos son factores que sin duda alguna puedes ayudar a una empresa u organización a tomar la decisión de optar por la utilización de una nube" (Mishra, 2017). (Ruparelia, 2016)

- **Costos**

El paradigma de la computación en la nube se basa en el intercambio y la utilización óptima de recursos de hardware, una empresa solo necesita pagar el tiempo durante el cual utiliza un recurso. Cuando no se necesita un recurso, la empresa puede renunciar a él y ponerlo a disposición de otra persona para que lo use. Esto reduce tanto el costo de inversión de hardware inicial para un negocio como los costos de mantenimiento continuos. El proveedor de servicios en la nube, no el consumidor, maneja el mantenimiento del subyacente hardware. (Mishra, 2017).

"Al mismo tiempo, se reducen considerablemente no solo las inversiones, sino los plazos necesarios para lanzar al mercado una nueva aplicación o servicio si está basado en el modelo de computación en la nube, facilitando así un mayor dinamismo en la oferta de servicios y habilitando a nuevos actores a entrada en mercados en los que hasta la fecha las barreras de entrada resultaban disuasorias." (Joyanes Aguilar, 2012, p. 42) (Joyanes Aguilar, 2012, p. 42).

- **Disponibilidad**

El tiempo para aprovisionar un recurso listo para usar en la nube es significativamente más bajo que tener que establecer un recurso similar en casa. Por ejemplo, una empresa podría aprovisionar un servidor virtual con un proveedor de la nube en cuestión de segundos, mientras que el proceso real de adquisición de nuevo hardware y software del servidor generalmente lleva unos meses en la mayoría de las organizaciones medianas y grandes. (Mishra, 2017). (Ruparelia, 2016)

A parte de estas dos grandes ventajas que ya son una gran razón para su uso, los siguientes autores plantean muchas más ventajas que puede ofrecer la computación en la nube dentro de una organización sin importar el tamaño de esta.

- **Ahorro**

Tal vez es el beneficio más significativo de la computación en la nube. Las empresas, independientemente del tamaño que tengan, tienen la finalidad de ganar dinero siempre y cuando mantengan sus gastos en unos mínimos asumibles. Con la utilización de los servicios en la nube se eliminan gastos y requisitos como puede ser la adquisición de un servidor interno. La eliminación de estos costes conlleva el ahorro en otras materias como pueden ser los derivados en el gasto energético de estos sistemas.

- **Confiabilidad**

Este tipo de sistema gestionado en la nube es más fiable y consistente que la infraestructura de tecnología de información interna. La computación en la nube ofrece un servicio 24 horas al día, 7 días a la semana, 365 días al año y un 99,99% de disponibilidad.

- **Manejabilidad**

Los servicios en la nube te ofrecen la ventaja de un "todo en uno". La única preocupación que tiene el cliente es el uso del sistema ya que su configuración puesta en marcha y mantenimiento corren a cargo del proveedor de los servicios.

- **Implementación de las últimas tecnologías**

Con la contratación de un servicio de computación en la nube, la empresa se garantiza la constante implementación de sus sistemas a las últimas novedades informáticas sin coste adicional. (PMC Group, 2017).

- **Desventajas de la computación en la nube**

"Se observan las principales desventajas que con frecuencia se atribuyen al modelo de la computación en la nube, que estriban esencialmente en la pérdida de control por parte de los usuarios tanto sobre las aplicaciones y servicios como sobre los datos, en ocasiones muy sensibles, que se suben a nubes, con los consiguientes riesgos relativos tanto a la privacidad como a la integridad de estos. Para evitar estas desventajas, el proveedor de servicios deberá garantizar con transparencia tanto la seguridad como la privacidad de la información a sus clientes. En cualquier caso, estos riesgos no son exclusivos del modelo de la computación en la nube, ya que también están presentes en los sistemas de información dentro de las organizaciones."(Joyanes Aguilar, 2012).

Existen algunas otras desventajas que también es necesario tener en cuenta.

- **Falta de tiempo**

La capacidad de uso de la computación en la nube consigue que el número de llamadas que se reciben puede llegar a colapsar el servicio con el consiguiente corte del servicio lo que acarrea una pérdida de tiempo laboral hasta que se restablece la línea. Lo mismo ocurre en el caso de que la empresa sufra una pérdida de señal de internet.

- **Seguridad**

Aunque la seguridad es una de las principales preocupaciones de los proveedores de los servicios de la computación en la nube, el uso de estos servicios basados en la nube conlleva siempre un riesgo de usurpación de datos sensibles de la empresa que contrata los servicios que proporciona la nube.

- **Dependencia de un proveedor**

Los proveedores de servicios en la nube prometen flexibilidad para usar e integrar, pero no es oro todo lo que reluce. El cambio de servicios es algo que no ha evolucionado todavía por completo, por lo que cualquier empresa que pretendiese migrar servicios de un proveedor a otro puede tener dificultades. Por ejemplo, aplicaciones desarrolladas por Microsoft, probablemente no funcionarán correctamente en una plataforma de Linux.

- **Control limitado**

Dado que la infraestructura de la nube es propiedad en su totalidad, gestionado y supervisado por el proveedor del servicio, el cliente solo tiene un control limitado de las funciones del sistema. Para aquellas funciones que necesiten ser modificadas o actualizadas será el proveedor del servicio el encargado de gestionarlas.(PMC Group, 2017).

1.2. Virtualización

1.2.1. Virtualización de Servidor

"La computación en la nube se basa en la tecnología de virtualización. Fundamentalmente, hay dos tipos de virtualización que son virtualización de servidores y virtualización de aplicaciones" (Mishra, 2017, p. 5).

"También conocida como virtualización de hardware, en este modelo, una sola la máquina física alberga varias máquinas virtuales. Cada máquina virtual puede tener su propio sistema operativo (diferente del sistema operativo de la máquina física subyacente) y su propio conjunto único de aplicaciones." (Mishra, 2017, p. 5).

"La virtualización de servidores usa hardware físico común (redes, almacenamiento o máquinas de computación) para alojar máquinas virtuales. Una máquina física podría tener cualquier número de máquinas virtuales que se ejecutan en ella, para que un conjunto de

hardware se utilice para ejecutar diferentes máquinas. Las máquinas virtuales se pueden instalar con su propio sistema operativo y poseer diferentes conjuntos de aplicaciones."(Ruparelia, 2016, p. 20).

"La virtualización de servidores tiene una importante ventaja en costos, permite consolidar una gran cantidad de máquinas como físicas, en menos máquinas físicas que alojan las máquinas virtuales. Este aumento en la eficiencia de la computación resulta en reducción de costos de espacio, mantenimiento, enfriamiento y electricidad." (Ruparelia, 2016, p. 20).

En la Figura 8 se indica cómo funciona un servidor físico en comparación a un servidor virtual.

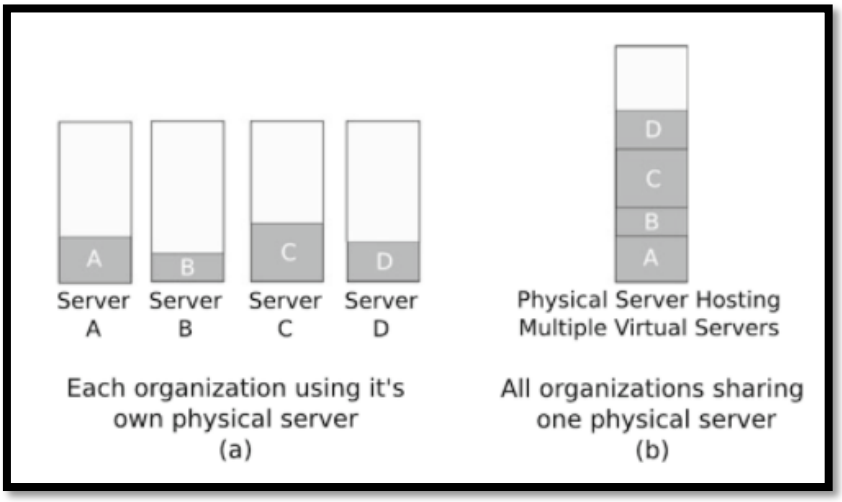

Figura 8: Utilización de un servidor físico versus la utilización de un servidor virtual.

Fuente: (Alani, 2016)

"Como usuario final, espera consumir uno o más servicios de su proveedor de computación en la nube a través de Internet. Estos servicios pueden variar desde máquinas virtuales básicas con un sistema operativo básico hasta suites enteras de aplicaciones." (Mishra, 2017, p. 20).

"Es un método de particionar o dividir un servidor físico en múltiples servidores en modo de que cada uno de ellos tenga la apariencia y capacidades como si fuera una máquina dedicada. De este modo los servidores individuales se dividen en dominios independientes, aislados unos de otros para asegurar que no haya interferencias entre un dominio y otro. Cada dominio asegura a los clientes que los diferentes sistemas no tienen forma de interferir con la integridad de los otros sistemas restantes." (Joyanes Aguilar, 2012, p. 152).

1.2.2. Virtualización de Aplicaciones.

Este tipo de virtualización se basa simplemente en que **“**Una sola máquina alberga una o más aplicaciones que son entregadas a uno o más usuarios a través de Internet.”(Mishra, 2017, p. 5).

“Es un método que describe las tecnologías de software que las separa del sistema operativo fundamental en que se ha ejecutado. Una aplicación virtualizada totalmente no se instala en el sentido tradicional, aunque, sin embargo, se ejecuta como si lo estuviera. La aplicación produce la sensación de que está directamente interconectada con el sistema operativo original y los recursos que gestiona” (Joyanes Aguilar, 2012, p. 154).

Entrega una aplicación alojada en una única máquina a una gran cantidad de usuarios. La aplicación puede estar situado en la nube en máquinas virtuales de alta calidad, pero, debido a que una gran cantidad de usuarios acceden a él, sus costos son compartido por esos usuarios. Esto hace que la aplicación sea más barata para el usuario final. El usuario final no necesita tener hardware de alta calidad para ejecutar la aplicación; una máquina económica, como una estación de trabajo o un terminal de cliente ligero, será suficiente, (…). Por lo general, en tales casos, la aplicación virtual se consume a través de una aplicación móvil o un navegador de Internet del usuario final.(Ruparelia, 2016, p. 20).

CAPÍTULO 2

Desarrollo

2.1. OpenShift

2.1.1. Características de OpenShift

"OpenShift es una PaaS de Red Hat, hay tres versiones diferentes: OpenShift Origin, OpenShift Online y OpenShift Enterprise. OpenShift Origin, la versión de código abierto y gratuito de OpenShift, es el proyecto inicial para las otras dos versiones. Está en GitHub y se lanzó bajo una licencia de Apache 2. Todos los cambios en la base de código pasan por el repositorio público, tanto para Red Hat como para desarrolladores externos. Si se desea utilizar esta versión, se deberá instalarla en una infraestructura propia." (Pousty & Miller, 2014).

En la Figura 9 se muestra la relación que existe entre las versiones de OpenShift y el tipo de infraestructura que usan cada una de ellas.

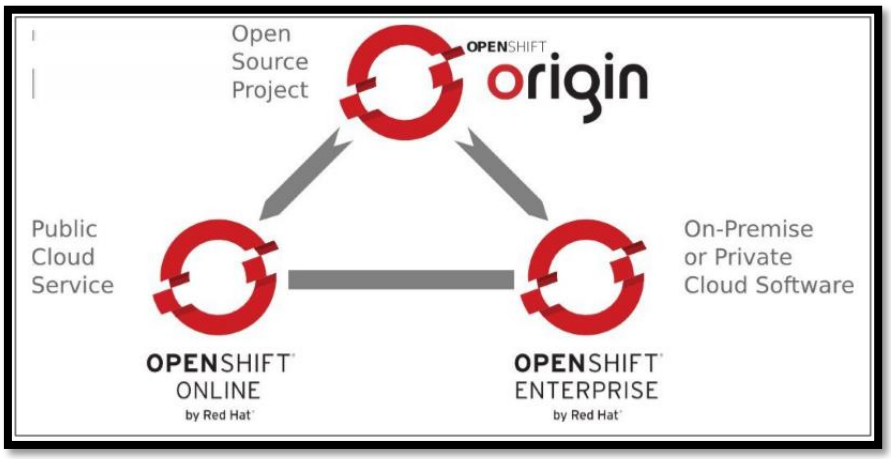

Figura 9: Relación entre las tres versiones de OpenShift

Fuente: (Pousty & Miller, 2014)

OpenShift fue anunciado al mundo en el mes de abril de 2012, como un proyecto de código abierto basado en las tecnologías de Kubernetes y Docker, OpenShift Origin nació con la versión 1.0 en junio de 2015 y avanzó hasta llegar a una versión actual 3.11, en ese proceso contribuyó con muchos proyectos nativos en la nube y en una gran cantidad de proyectos de alto nivel. Después del éxito alcanzado, OpenShift Origin decide realizar un cambio de imagen el 3 de agosto de 2018 en su versión 3.10, pasando a llamarse actualmente OpenShift OKD,

esto con el afán de destacar su trabajo con Kubernetes, como fue anunciado por Diane Mueller en el blog oficial de OpenShift. “Así que es hora de un nuevo logotipo, un nuevo sitio web y un nuevo nombre para nuestro proyecto de código abierto. Estamos cambiando el nombre de nuestro proyecto de código abierto para representar mejor quiénes somos hoy y quién seremos mañana: la distribución de la comunidad de Origen de Kubernetes que impulsa a Red Hat OpenShift.” (“OKD”, 2018)

“Red Hat OpenShift proporciona a los equipos de desarrollo de aplicaciones y operaciones de TI la capacidad de acelerar la entrega de aplicaciones con la velocidad y la coherencia que exige la empresa.” (Red Hat, Inc, 2015b).

“Utilizando las tecnologías de automatización y la arquitectura en la nube de Red Hat, OpenShift puede estandarizar y optimizar los flujos de trabajo de los desarrolladores, las organizaciones de TI pueden construir de manera eficiente y llevar las aplicaciones al mercado de una manera más rápida.” (Red Hat, Inc, 2015b).

- **Plataforma de Autoservicio**

“Los desarrolladores pueden crear rápida y fácilmente aplicaciones bajo demanda directamente desde las herramientas que más se usan. Los operadores pueden aprovechar la ubicación y la política para organizar entornos que cumplan con sus mejores prácticas.”(Red Hat, Inc, 2015b).

- **Soporte Multilenguaje**

“Los desarrolladores tienen la opción y la capacidad de ejecutar múltiples lenguajes de programación, marcos de trabajo y bases de datos en la misma plataforma. Permite a los clientes aprovechar más fácilmente el ecosistema Docker.”(Red Hat, Inc, 2015b)

- **Basado en Contenedores**

“OpenShift proporciona una plataforma inmutable basada en contenedores y Dockers para implementar y ejecutar aplicaciones y microservicios.”(Red Hat, Inc, 2015b).

- **Automatización**

“OpenShift automatiza las compilaciones, implementaciones, administración y más componentes integrados de Kubernetes.”(Red Hat, Inc, 2015b).

- **Persistencia**

“OpenShift les permite a los arquitectos de la plataforma, la opción de incorporar persistencia en sus componentes de aplicación y, al mismo tiempo, ser capaces de ofrecer un diseño nativo de nubes sin estado.”(Red Hat, Inc, 2015b).

- **Portabilidad de la Aplicación**

“Construido alrededor de un modelo de contenedor estandarizado impulsado por Docker. Lo que significa que cualquier aplicación creada en OpenShift puede ejecutarse fácilmente en cualquier lugar que admita el acoplador.” (Red Hat, Inc, 2015b).

- **Colaboración**

“A diferencia de algunas plataformas heredadas, la colaboración en OpenShift no se ve obstaculizada por un proceso inflado. Los desarrolladores pueden agregar o eliminar fácilmente miembros del equipo a un proyecto.”(Red Hat, Inc, 2015b).

- **Fuente Abierta**

“Lo que significa una verdadera transparencia en lugar de algunas de esas opciones de "Núcleo abierto". Tampoco es solo con el código fuente, que incluye las mejores prácticas.”(Red Hat, Inc, 2015b).

- **Escalable**

“Permite que las aplicaciones se escalen fácilmente para manejar un mayor tráfico y demanda en las aplicaciones.”(Red Hat, Inc, 2015b).

Otras de las características adicionales que brinda OpenShift en diferentes ámbitos se describirán en breves rasgos en la Tabla 4.

Tabla 4: Características adicionales OpenShift

Característica	Descripción	Tipo
Aplicación De Imágenes Y Plantillas De Inicio Rápido	“Permite escribir aplicaciones en Java, Node.js, .NET, Ruby, Python, PHP y más. OpenShift incluye plantillas de aplicaciones de inicio rápido creadas previamente que permiten implementar plantillas de aplicaciones, bases de datos y más con un solo clic.” (Red Hat, Inc, 2017).	Aplicaciones bajo pedido
Catálogo De Contenedores, Docker Hub, Y Más	“Permite aprovechar una gran comunidad de contenedores con formato Docker de Linux. Desde contenedores listos para la empresa en Red Hat Container Catalog hasta registros comunitarios como Docker Hub, y trabajar directamente con Docker API.” (Red Hat, Inc, 2017).	Aplicaciones bajo pedido
Despliegue Con Un Solo Clic	“El despliegue se realiza con un solo clic o ingresando un comando git push. OpenShift está diseñado para reducir o eliminar muchos de los dolores de cabeza de administración de sistemas relacionados con la construcción y el despliegue de aplicaciones en contenedores.” (Red Hat, Inc, 2017).	Código
Flujos De Trabajo De Desarrollador Estandarizados	“La organización de desarrollo de aplicaciones puede estandarizar el flujo de trabajo del desarrollador y crear procesos repetibles para la entrega de aplicaciones para agilizar todo el proceso.” (Red Hat, Inc, 2017).	Proceso De Entrega
Escalado Automático	“Permite la elasticidad de la nube al proporcionar escalado horizontal automático de la plataforma a medida que aumenta la carga de la aplicación, eliminando la necesidad de Operaciones para aumentar manualmente el número de instancias de la aplicación.” (Red Hat, Inc, 2017).	Capacidad flexible
Elección De La Infraestructura De La Nube	“Es posible ejecutar en la parte superior de la nube física o virtual, pública o privada, e incluso la infraestructura de nube híbrida. Esto le da a TI la libertad de implementar	Capacidad flexible

	OpenShift de la manera que mejor se adapte a la infraestructura existente.” (Red Hat, Inc, 2017).	
Conjunto De Herramientas Rico En Línea De Comandos	“Incluye un amplio conjunto de herramientas de línea de comandos que brindan acceso completo a la interfaz del desarrollador. Estas herramientas son fáciles de usar y también se pueden programar para interacciones automatizadas.” (Red Hat, Inc, 2017).	Poderosas herramientas
Integración Ide	“Con la integración de la plataforma OpenShift con Eclipse, JBoss Developer Studio y los desarrolladores de Visual Studio pueden mantenerse completamente dentro del IDE con el que se sienten cómodos cuando trabajan con OpenShift.” (Red Hat, Inc, 2017).	Poderosas herramientas
Fuente Abierta	“A diferencia de otros proveedores importantes de tecnología, trabaja y participa en comunidades de meritocracia rigurosa. Red Hat ® es el proveedor líder mundial de soluciones de código abierto. Cuando las mejores ideas ganan, sus clientes no pueden perder.” (Red Hat, Inc, 2017).	Abierto y Extensible
Ecosistema	“Es posible conectarse al mayor ecosistema de socios, clientes y expertos de la industria que respalda y acelera el éxito.” (Red Hat, Inc, 2017).	Abierto y Extensible
Entrenamiento Y Certificación	“Dominar el desarrollo y las operaciones de aplicaciones en contenedores con cursos intensivos de capacitación en el mundo real. Ya sea que se esté actualizando la experiencia o desarrollando nuevas habilidades, aquí es donde comienza todo. Entonces, es posible ser más comercializable certificando el nivel de habilidad.” (Red Hat, Inc, 2017).	Servicios y Soporte

Fuente: Propia

2.1.2. Arquitectura de OpenShift

"OpenShift v3 es un sistema diseñado para exponer la imagen del contenedor con formato Docker subyacente y los conceptos de Kubernetes con la mayor precisión posible, con un enfoque en la fácil composición de las aplicaciones por parte de un desarrollador. Por ejemplo, instalar Ruby, ingresar el código y agregar MySQL." (Red Hat, Inc, 2015a).

- **Capas de OpenShift**

"El servicio Docker proporciona la abstracción para el empaquetado y la creación de imágenes de contenedores livianos basadas en Linux. Kubernetes proporciona la administración del clúster y organiza contenedores en múltiples hosts." (Red Hat, Inc, 2015a).

Los contenedores de la plataforma OpenShift permiten:

a) Gestión de código fuente, compilaciones e implementaciones para desarrolladores
b) Administrar y proporcionar imágenes a escala a medida que fluyen a través de su sistema
c) Gestión de aplicaciones a escala
d) Seguimiento de equipo y usuario para organizar una gran cantidad de desarrolladores
e) Infraestructura de red que admite el clúster

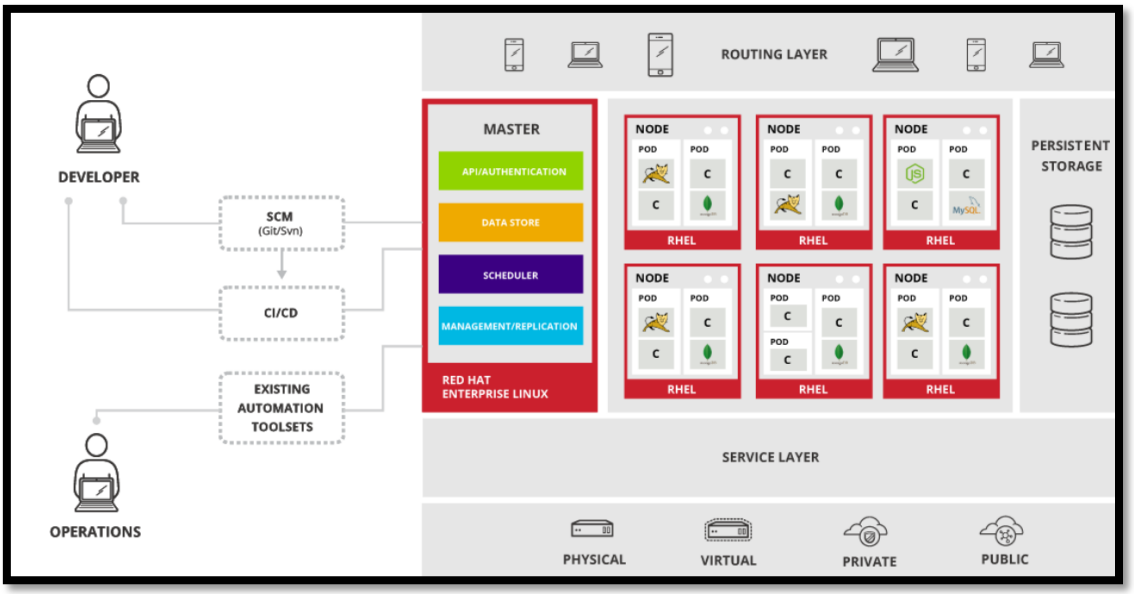

Figura 10: Descripción general de la arquitectura de la plataforma OpenShift

Fuente: (Red Hat, Inc, 2015a).

- **En qué se basa la arquitectura**

OpenShift Origin tiene una arquitectura basada en microservicios de unidades desacopladas más pequeñas que funcionan juntas. Se ejecuta en la parte superior de un clúster de Kubernetes, con datos sobre los objetos almacenados en etcd, una tienda confiable de clave-valor agrupada. Esos servicios están desglosados por función:

a) API[12] REST, que expone cada uno de los objetos centrales.
b) Los controladores, que leen esas API, aplican cambios a otros objetos e informan el estado o escriben de nuevo al objeto.

Los usuarios hacen llamadas a la API REST para cambiar el estado del sistema. Los controladores usan la API REST para leer el estado deseado del usuario, y luego intentan sincronizar las otras partes del sistema. Por ejemplo, cuando un usuario solicita una compilación, crea un objeto de "compilación". El controlador de compilación ve que se ha creado una compilación nueva y ejecuta un proceso en el clúster para realizar esa compilación. Cuando la compilación se completa, el controlador actualiza el objeto de compilación a través de la API REST y el usuario ve que su compilación está completa. (Red Hat, Inc, 2015a).

- **Componentes de la Infraestructura**

  - **Kubernetes**

Kubernetes administra aplicaciones en contenedores a través de un conjunto de contenedores o hosts y proporciona mecanismos para la implementación, el mantenimiento y la escala de la aplicación. El servicio Docker empaqueta, crea instancias y ejecuta aplicaciones en contenedores.

Un clúster de Kubernetes consiste en uno o más maestros y un conjunto de nodos. Opcionalmente, puede configurar sus maestros para alta disponibilidad (HA) para garantizar que el clúster no tenga un solo punto de falla. (Red Hat, Inc, 2015a).

  - **Master**

El maestro es el host o los hosts que contienen los componentes maestros, incluidos el servidor API, el servidor del administrador del controlador y etc. El maestro gestiona los nodos

[12] API. Interfaz de programación de aplicaciones, es un conjunto de subrutinas que ofrecen abstracción para ser usadas por otra aplicación.

en su clúster de Kubernetes y programa los pods[13] para que se ejecuten en los nodos. (Red Hat, Inc, 2015a).

Los componentes del host maestro son los siguientes:

a) **Servidor API:** El servidor API de Kubernetes válida y configura los datos para pods, servicios y controladores de replicación. También asigna pods a los nodos y sincroniza la información del pod con la configuración del servicio. Se puede ejecutar como un proceso independiente.

b) **Etc:** etcd almacena el estado maestro persistente mientras que otros componentes miran etcd para ver los cambios para ubicarse en el estado deseado. etcd se puede configurar opcionalmente para alta disponibilidad.

c) **Servidor de Controlador Maestro:** El servidor del gestor de controlador ETCD realiza cambios en los objetos del controlador de replicación y luego utiliza la API para hacer cumplir el estado deseado. Se puede ejecutar como un proceso independiente. Varios de estos procesos crean un clúster con un maestro activo a la vez.

d) **HAProxy:** Opcional, se usa al configurar maestros de alta disponibilidad con el método nativo para equilibrar la carga entre los puntos finales de los maestros. El método de instalación avanzada puede ser configurado HAProxy con el método nativo. Alternativamente, puede usar el método nativo para configurar su propio equilibrador de carga de su elección. (Red Hat, Inc, 2015a).

- **Nodos**

Un nodo proporciona los entornos de tiempo de ejecución para contenedores. Cada nodo en un clúster de Kubernetes tiene los servicios requeridos para ser gestionados por el maestro. Los nodos también tienen los servicios necesarios para ejecutar pods, incluido el servicio Docker, un kubelet y un proxy de servicio.

OpenShift Origin crea nodos desde un proveedor de la nube, sistemas físicos o sistemas virtuales. Kubernetes interactúa con objetos de nodo que son una representación de esos nodos. El maestro utiliza la información de los objetos del nodo para validar nodos con comprobaciones de estado. Un nodo se ignora hasta que pasa las comprobaciones de estado

[13] POD. Concepto de Kubernetes que es uno o más contenedores implementados juntos en un host.

y el maestro continúa comprobando los nodos hasta que sean válidos. La documentación de Kubernetes contiene más información sobre la administración de nodos.

Los administradores pueden administrar nodos en una instancia de OpenShift Origin utilizando la CLI. Para definir la configuración completa y las opciones de seguridad al iniciar los servidores de nodos, también usar archivos de configuración de nodo dedicados. (Red Hat, Inc, 2015a).

Los componentes de un nodo son los siguientes:

a) **Kubelet:** Cada nodo tiene un kubelet que actualiza el nodo según lo especificado por un manifiesto de contenedor, que es un archivo YAML que describe un pod. El kubelet usa un conjunto de manifiestos para garantizar que sus contenedores se inicien y que sigan funcionando.

b) **Proxy de servicio:** Cada nodo también ejecuta un proxy de red simple que refleja los servicios definidos en la API en ese nodo. Esto permite que el nodo realice envíos simples de flujo TCP y UDP a través de un conjunto de back-end.(Red Hat, Inc, 2015a).

- **Contenedores**

OpenShift Origin puede utilizar cualquier servidor que implemente la API de registro de Docker como fuente de imágenes, incluido Docker Hub, registros privados ejecutados por terceros y el registro integrado de OpenShift Origin.

OpenShift Origin proporciona un registro integrado de contenedores llamado OpenShift Container Registry (OCR) que agrega la capacidad de aprovisionar automáticamente nuevos repositorios de imágenes bajo demanda. Esto proporciona a los usuarios una ubicación incorporada para las compilaciones de sus aplicaciones para impulsar las imágenes resultantes.

Cada vez que se envía una nueva imagen al OCR, el registro notifica a OpenShift Origin sobre la nueva imagen, pasando toda la información sobre ella, como el espacio de nombres, el nombre y los metadatos de la imagen. Diferentes piezas de OpenShift Origin reaccionan a las nuevas imágenes, creando nuevas construcciones e implementaciones.

OCR también se puede implementar como un componente independiente que actúa únicamente como un registro de contenedores, sin la integración de compilación y despliegue.

- **Consola Web**

La consola web de OpenShift Origin es una interfaz de usuario accesible desde un navegador web. Los desarrolladores pueden usar la consola web para visualizar, explorar y administrar los contenidos de los proyectos.

La consola web se inicia como parte del maestro. Todos los archivos estáticos necesarios para ejecutar la consola web se proveen en OpenShift Origin. Los administradores también pueden personalizar la consola web mediante extensiones, que le permiten ejecutar scripts y cargar hojas de estilo personalizadas cuando se carga la consola web. Puede cambiar la apariencia de casi cualquier aspecto de la interfaz de usuario.

- **Conceptos de la Arquitectura**

Los siguientes temas proporcionan información arquitectónica de alto nivel sobre los conceptos básicos y los objetos que se encuentran en OpenShift Origin. Muchos de estos objetos provienen de Kubernetes, que se amplía con OpenShift Origin para proporcionar una plataforma de ciclo de vida de desarrollo más rica en características.

a) **Los contenedores y las imágenes** son los componentes básicos para implementar sus aplicaciones.

b) **Los pods y los servicios** permiten que los contenedores se comuniquen entre sí a través de las conexiones proxy.

c) **Los proyectos y usuarios** proporcionan el espacio y los medios para que las comunidades organicen y administren su contenido en conjunto.

d) **Las compilaciones y las transmisiones de imágenes** permiten construir imágenes en funcionamiento y reaccionar ante nuevas imágenes.

e) **Las implementaciones** agregan soporte ampliado para el desarrollo del software y el ciclo de vida de la implementación.

f) **Las rutas** proporcionan el dominio para el acceso a la aplicación.

g) **Las plantillas** permiten que se creen muchos objetos a la vez en base a parámetros personalizados.

2.1.3. **Instalación de OpenShift**

OpenShift para su instalación necesita muchas configuraciones previas para su instalación y existen cosas que se deben tener en cuenta para dicha configuración ya que depende de escenario en que vaya a ser utilizado, algunas de las cosas que se deben tener en cuenta antes de la instalación son que OpenShift funciona bajo Red Hat Enterprise Linux (RHEL), si se necesita que exista una monitorización del clúster ya que esto requeriría más recursos del sistema y los aspectos mínimos que requiere OpenShift en lo que respecta a hardware .

“En un clúster de un host maestro OpenShift recomienda cumplir con los siguientes requisitos mínimos, tener 1 núcleo de CPU y 1.5 GB de memoria por cada 1000 pods. Por lo tanto, el tamaño recomendado de un host maestro en un grupo OpenShift de 2000 pods es como mínimo de 2 núcleos de CPU y 16 GB de RAM, más 2 núcleos de CPU y 3 GB de RAM, con un total de 4 núcleos de CPU y 19 GB de RAM”. (“Documentation OKD”, 2018)

Después de esas consideraciones se debe crear un archivo de inventario que tendrá detalles del host del clúster y detalles de la configuración del clúster para su posterior instalación como el dominio que usará y su dirección IP, con esta información OpenShift sabrá donde y como instalar el clúster, es necesario revisar la configuración de este archivo en la documentación de OpenShift en donde se detallan todas las variables que se debe configurar.

En este estudio se usó una máquina virtual en VMware 15.0, sobre la cual se levantó el sistema operativo CentOS 7 para llevar a cabo la instalación de OpenShift Origin 3.11, la cual se configuró con las siguientes características:

a) 2 núcleos de CPU
b) 16 GB de RAM
c) 200 GB de almacenamiento

Después de terminada la instalación del sistema operativo se instalan todas las librerías que OpenShift requiere las cuales se deben revisar en la documentación de OpenShift, entre las librerías necesarias están Ansible, Docker y Python y además se configura el dominio que se introdujo en el archivo de inventario en este caso console.dominio.prueba.

Opcionalmente se puede configurar el almacenamiento de Docker para optimizar su uso de disco.

Con estas configuraciones se puede proceder a clonar los archivos necesarios para la instalación de OpenShift Origin, que se encuentran almacenados en GitHub en la siguiente dirección https://github.com/openshift/openshift-ansible.git, se puede dar paso a la instalación de OpenShift por medio de ansible indicando la ruta específica del archivo de inventario con el siguiente comando para ejecutar el archivo de prerrequisitos:

```
$ ansible-playbook -i inventory.ini openshift-ansible/playbooks/prerequisites.yml
```

Y el siguiente para ejecutar el archivo de instalación del clúster:

```
$ ansible-playbook -i inventory.ini openshift-ansible/playbooks/deploy_cluster.yml
```

Si la instalación termina correctamente como se muestra en la Figura 11, se deberá proceder a crear un usuario y asignarle un rol para poder ingresar a OpenShift como se muestra en la sección 2.2.2 de este documento.

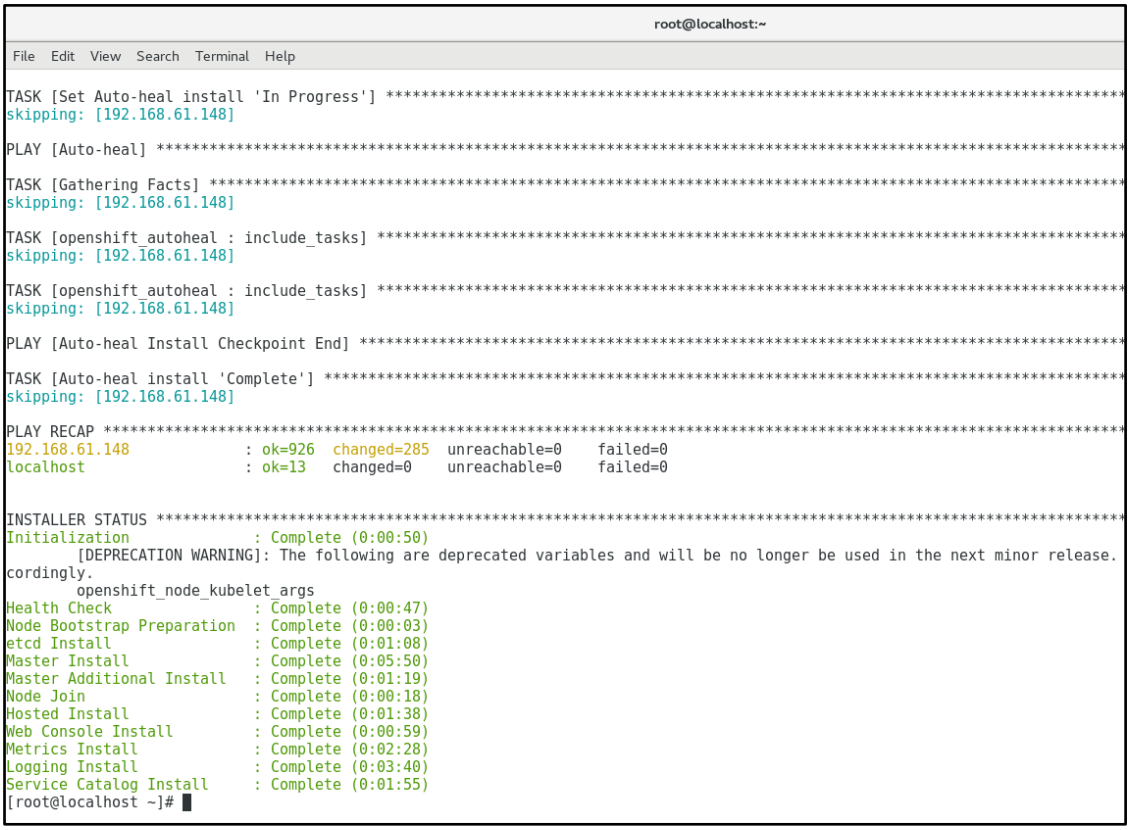

Figura 11 Instalación finalizada de OpenShift en este estudio

Fuente: Propia.

Con estas acciones se puede ingresar ya a la consola web en el dominio configurado y puerto configurado en el archivo de inventario con el usuario creado, en la Figura 12 se muestra la consola web de OpenShift versión 3.11.

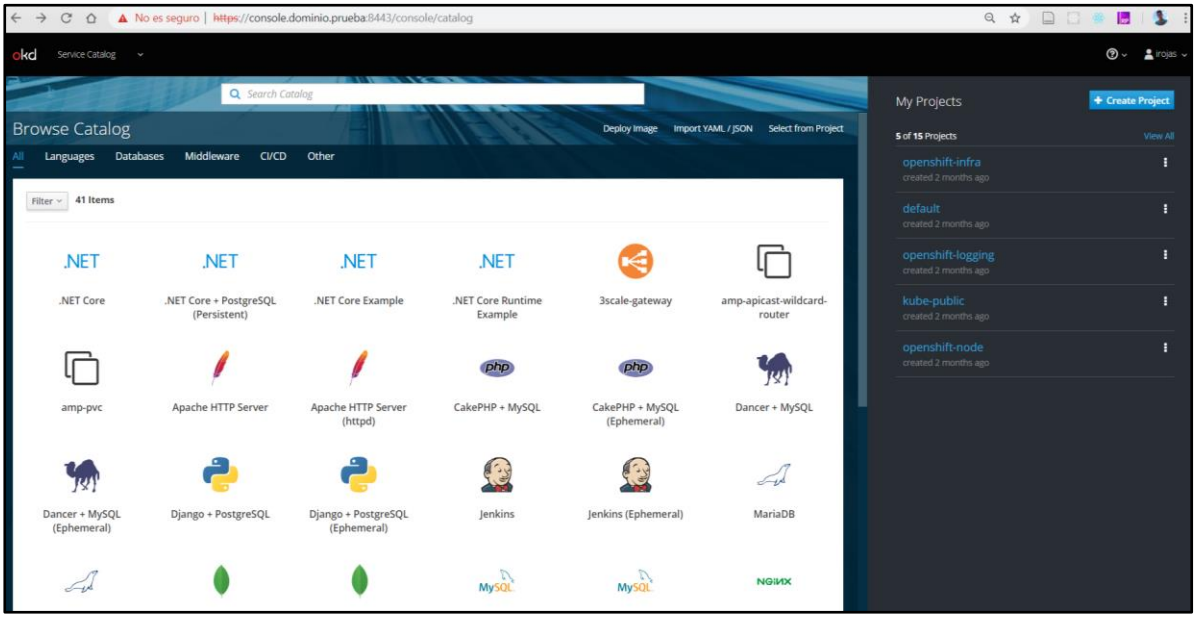

Figura 12 Pantalla Principal de la consola web de OpenShift

Fuente: Propia.

2.1.4. OpenShift en una nube Privada.

Hay muy pocas cosas mejores para un desarrollador de aplicaciones o un programador nato que una infraestructura sólida de una Plataforma como servicio (PaaS), que elimina todos los problemas subyacentes que suele tener el colocar aplicaciones en servicio y que se concentre simple y únicamente en el desarrollo de su aplicación.

Uno de los mejores que existen es OpenShift Online, que puede usar libremente o ampliar con una suscripción para obtener más instancias con más servicios como tamaño, escalabilidad y capacidades de almacenamiento. Este es un PaaS público, y de ninguna manera esta opción es una mala elección, pero se podría estar interesado en armar una propia experiencia privada de PaaS realizando la instalación de OpenShift Origin en una infraestructura o una máquina virtual local y privada. (Eric D. Schabell, 2016).

No está claro cuántas empresas han configurado sus propias nubes privadas de PaaS basadas en OpenShift Origin, que Red Hat abrió en abril del 2012. Sin embargo, lo que está

claro para Red Hat es que las empresas comerciales desean ejecutar sus propias nubes de plataforma como servicio internas debido a problemas de cumplimiento y seguridad con la opción de utilizarlas en nubes públicas basadas en la misma capa de abstracción de PaaS.

Y por eso la compañía RedHat ha estado trabajando para fortalecer OpenShift y desarrollar su equipo de soporte técnico para que se pueda venderlo a las tiendas de TI y a departamentos específicos de TI. (Timothy Prickett Morgan, 2012).

Para realizar la instalación de OpenShift Origin en una nube privada se poseen varios métodos de instalación disponibles, cada uno de los cuales le permite obtener rápidamente su propia instancia de OpenShift Origin en funcionamiento. Según su entorno, puede elegir el método de instalación que mejor se adapte a sus necesidades. Para implementar un clúster completo de OpenShift Origin es necesario consultar la guía de instalación avanzada de la documentación de OpenShift Origin.

Antes de elegir un método de instalación, primero se debe cumplir con los requisitos previos en sus hosts, que incluyen la verificación de los requisitos del sistema y del entorno, así como la instalación y la configuración de Docker. Después de asegurarse de que sus hosts están configurados correctamente, puede continuar eligiendo uno de los métodos de instalación que están disponibles. Docker y OpenShift Origin deben ejecutarse en el sistema operativo Linux. Si desea ejecutar el servidor desde un host Windows o Mac OS X, primero debe iniciar una VM Linux. (Red Hat, Inc., 2015b).

2.2. Administración de OpenShift

La administración del clúster de OpenShift cubre las tareas básicas que un administrador debe realizar cotidianamente y cómo debería reaccionar frente a ciertas circunstancias normales en este ámbito de administración.

La administración del clúster de OpenShift se puede realizar en mayor parte utilizando la CLI[14], y la otra manera disponible es usando la consola web que ofrece una manera sencilla e intuitiva de controlar la nube, obviamente la CLI proporciona muchas más herramientas y opciones de configuración completas a diferencia que la consola web. (Red Hat, Inc., 2015a).

[14] CLI Interfaz de línea de comandos

2.2.1. Administración Básica

En esta sección se incluye la manera de administrar básicamente OpenShift en su interfaz de línea de comandos con las acciones básicas que se requiere conocer, posteriormente se explican algunos de los datos que se pueden visualizar en su interfaz gráfica en la consola Web.

a) Ingreso en OpenShift

```
root@localhost:~
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# oc login
Authentication required for https://console.dominio.prueba:8443 (openshift)
Username: irojas
Password:
Login successful.

You have access to the following projects and can switch between them with 'oc project <projectname>':
```

b) Crear un nuevo proyecto

```
root@localhost:~
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# oc new-project demo-gestionbecas
Now using project "demo-gestionbecas" on server "https://console.dominio.prueba:8443".

You can add applications to this project with the 'new-app' command. For example, try:

    oc new-app centos/ruby-25-centos7~https://github.com/sclorg/ruby-ex.git

to build a new example application in Ruby.
```

c) Crear una nueva aplicación

```
root@localhost:~
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# oc new-app --name=myappphp --template=cakephp-mysql-example
--> Deploying template "openshift/cakephp-mysql-example" to project demo-gestionbecas

     CakePHP + MySQL (Ephemeral)

--> Creating resources ...
    secret "cakephp-mysql-example" created
    service "cakephp-mysql-example" created
    route.route.openshift.io "cakephp-mysql-example" created
    imagestream.image.openshift.io "cakephp-mysql-example" created
    buildconfig.build.openshift.io "cakephp-mysql-example" created
    deploymentconfig.apps.openshift.io "cakephp-mysql-example" created
    service "mysql" created
    deploymentconfig.apps.openshift.io "mysql" created
--> Success
```

La aplicación se crea de una plantilla llamada cakephp-mysql-example que es una aplicación base de PHP junto con una base de datos MySQL, junto con ella se crean las respectivas partes como dos servicios que se abrevia con (svc), los dos tienen un

deploymentConfig que se abrevia con (dc), un buildConfig que se abrevia con (bc) y la respectiva ruta de acceso.

d) Comprobar el estado del proyecto

```
root@localhost:~
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# oc status
In project demo-gestionbecas on server https://console.dominio.prueba:8443

http://cakephp-mysql-example-demo-gestionbecas.apps.dominio.prueba (svc/cakephp-mysql-example)
  dc/cakephp-mysql-example deploys istag/cakephp-mysql-example:latest <-
    bc/cakephp-mysql-example source builds https://github.com/sclorg/cakephp-ex.git on openshift/php:7.1
    deployment #1 deployed 7 minutes ago - 1 pod

svc/mysql - 172.30.227.218:3306
  dc/mysql deploys openshift/mysql:5.7
    deployment #1 deployed 18 minutes ago - 1 pod
```

Se observa los dos pods que se crearon previamente uno de la aplicación PHP y el otro de la base de datos en MySQL.

e) Consultar información de los recursos que son creados

```
root@localhost:~
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# oc describe svc/mysql
Name:              mysql
Namespace:         demo-gestionbecas
Labels:            app=cakephp-mysql-example
                   template=cakephp-mysql-example
Annotations:       description=Exposes the database server
                   openshift.io/generated-by=OpenShiftNewApp
Selector:          name=mysql
Type:              ClusterIP
IP:                172.30.227.218
Port:              mysql  3306/TCP
TargetPort:        3306/TCP
Endpoints:         10.128.0.60:3306
Session Affinity:  None
Events:            <none>
```

Se observa información como la IP del servicio, los puertos que usa en este caso 3306 en TCP, su nombre y el proyecto en el que se encuentra.

f) Listar todos los pods

```
root@localhost:~
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# oc get po --all-namespaces
NAMESPACE                NAME                            READY     STATUS      RESTARTS   AGE
default                  docker-registry-1-kfrpd         1/1       Running     1          10d
default                  registry-console-1-hh7s6        1/1       Running     1          10d
default                  router-1-2k2jt                  1/1       Running     2          10d
demo-gestionbecas        cakephp-mysql-example-1-48crc   1/1       Running     0          17m
```

En la Figura 13 se detalla la información más relevante que es presentada en la consola web que brinda OpenShift acerca de un contenedor con una aplicación desplegada en este caso la aplicación de prueba de Gestión de Becas.

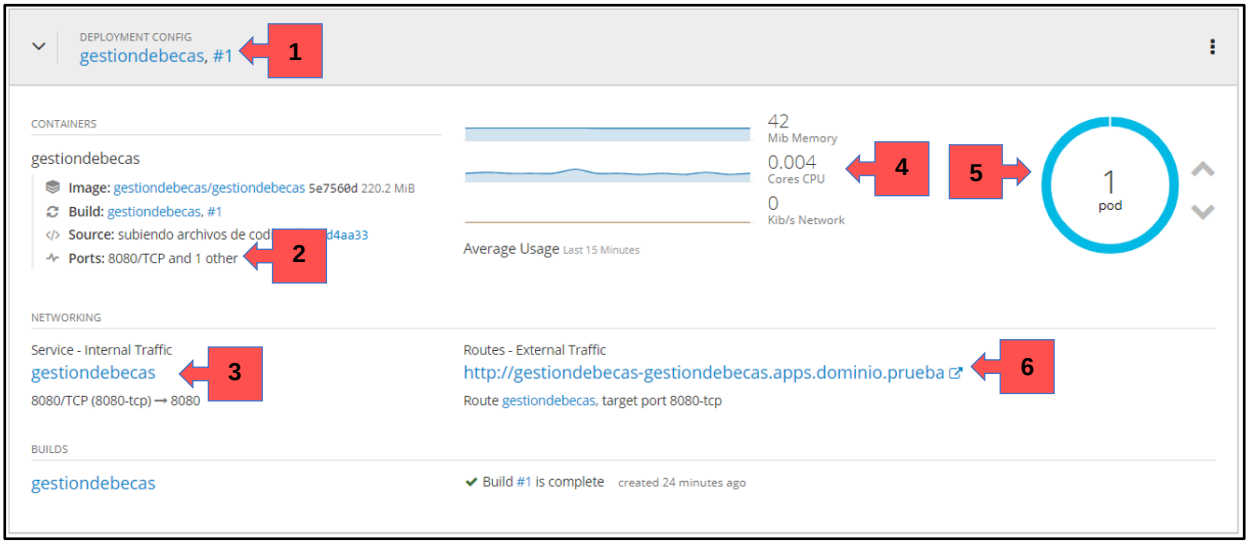

Figura 13: Datos informativos Consola Web OpenShift

Fuente: Propia

1. **Despliegue**: Indica la imagen que se usó para el despliegue e informa el número de despliegue del contenedor en este caso el despliegue número 1.
2. **Puertos:** Indica los puertos que expone el contenedor, en este se expone el puerto 8080 con el protocolo TCP.
3. **Servicio:** Cada aplicación que se tenga en OpenShift generará un servicio, el servicio es quien expone los puertos para la comunicación con el contenedor.
4. **Métricas:** Permite visualizar gráficamente los recursos que están siendo consumidos por el contenedor como RAM, CPU y red.
5. **Pods:** Con cada instancia generada de la aplicación se visualizará una cantidad de Pods en este caso solamente una instancia.
6. **Ruta:** Esta es la dirección por la cual se ingresará a la aplicación, las rutas son creadas con el nombre del servicio unido al nombre del proyecto y finalizado con el dominio que se creó OpenShift.

2.2.2. Administrar nodos

Cuando se realizan operaciones de administración de nodos, la CLI interactúa con objetos del nodo que son representaciones de hosts de nodos reales. El maestro utiliza la información de los objetos del nodo para validar nodos con comprobaciones de estado. (Red Hat, Inc., 2015a).

- **Listado de nodos**

Para enumerar todos los nodos conocidos por el maestro:

```
$ oc get nodes
NAME                 STATUS    ROLES     AGE       VERSION
master.example.com   Ready     master    7h        v1.9.1+a0ce1bc657
node1.example.com    Ready     compute   7h        v1.9.1+a0ce1bc657
node2.example.com    Ready     compute   7h        v1.9.1+a0ce1bc657
```

También se puede consultar la información de un nodo específico usando como clave su nombre, el comando que se emplea es el siguiente, reemplazando <node> con el nombre completo de nodo:

```
$ oc get node <node>
```

La información que se observa en la pantalla incluye parámetros como el STATUS, que es el estado en el que se encuentra cada respectivo nodo, un nodo en OpenShift puede encontrarse en tres condiciones, en la Tabla 5 se lista y describe cada una de las tres condiciones posibles en un nodo:

Tabla 5: Condiciones del nodo

Condición	Descripción
Ready	El nodo pasa las comprobaciones de estado realizadas desde el maestro y devuelve esta condición.
NotReady	El nodo no pasa las comprobaciones de estado realizadas desde el maestro.
SchedulingDisabled	Los pods no pueden programarse para su ubicación en el nodo.

Fuente: (Red Hat, Inc., 2015a)

Para obtener información completa y detallada sobre un nodo como su nombre completo, su dirección IP[15], nombre de su host y sus recursos se puede emplear el comando describe de la siguiente manera.

```
$ oc describe node <node>
```

Existen también opciones de eliminación y edición como son el comando delete y edit respectivamente, en lo que respecta a la eliminación de un nodo hay que tener en cuenta que con la eliminación de este también se borraran con todos los pods que existan en su interior, para ello OpenShift no permitirá su eliminación si existe uno o más pods en su interior, para ello se deberá vaciar el nodo antes de su eliminación, si se desea eliminar el nodo junto con todos los pods y contenedores en su interior se puede ejecutar el archivo uninstall.yml el cual procede a desinstalar totalmente el nodo en el que se ejecuta. (Red Hat, Inc., 2015a).

2.2.3. Administrar usuarios

Esta es otra de las partes importantes en la administración de la nube, en este tema se abarca la administración de cuentas de usuario , que incluye cómo se crean nuevas cuentas de usuario, como se manejan y cómo pueden eliminarse dentro de la plataforma OpenShift Origin.

- **Agregar un usuario**

Después de que los nuevos usuarios inicien sesión en OpenShift Origin, se crea una cuenta para ese usuario por cada proveedor de identidad configurado en el maestro. El administrador del clúster puede administrar el nivel de acceso de cada usuario. (Red Hat, Inc., 2015a).

a) Lo primero que se debe hacer para crear un usuario es crear una cuenta de usuario a la cual después de ser creada se le agregara el rol deseado, que será lo que le permita el ingreso al clúster.

```
$ htpasswd -b /etc/origin/master/htpasswd ${USERNAME} ${PASSWORD}
```

[15] IP es un número que identifica, de manera lógica y jerárquica, a una Interfaz en red

b) Darle al nuevo usuario el rol deseado:

```
$ oc create clusterrolebinding registry-controller --clusterrole=cluster-admin --user=admin
```

Donde la opción --clusterrole es el rol deseado del clúster. Por ejemplo, para otorgar a los nuevos usuarios los privilegios de administrador se usa el rol cluster-admin, lo que le da al usuario acceso a todo dentro de un clúster.

- **Eliminar Usuario**

Para eliminar un usuario primero hay que eliminar el registro del usuario y después se procede a eliminar la identidad del usuario.

a) Eliminar el registro de usuario

```
$ oc delete user demo
  user "demo" deleted
```

b) Eliminar la identidad del usuario

```
$ oc delete identity htpasswd_auth:demo
  identity "htpasswd_auth:demo" deleted
```

La identidad del usuario está relacionada con el proveedor de identidad que utiliza. Se puede obtener el nombre del proveedor de registro de usuario con:

```
$ oc get user
```

En este ejemplo, el nombre del proveedor de identidad es htpasswd_auth:

```
$ oc delete identity htpasswd_auth:demo
  identity "htpasswd_auth:demo" deleted
```

2.2.4. Gestión de proyectos

En OpenShift Origin, los proyectos se usan para agrupar y aislar objetos relacionados. Como administrador, se puede otorgar a desarrolladores el acceso a ciertos proyectos o permitir crear otros proyectos además se puede otorgar derechos administrativos dentro de proyectos individuales a quien lo requiera. (Red Hat, Inc., 2015a).

- **Proyectos de auto aprovisionamiento**

Puede permitir a los desarrolladores crear sus propios proyectos. Hay un medio que asignará un proyecto según una plantilla. La consola web y el comando #oc new-project utilizan este medio cuando un desarrollador crea un nuevo proyecto. (Red Hat, Inc., 2015a).

- **Modificar una plantilla para nuevos proyectos**

El servidor de la API proporciona automáticamente proyectos basados en la plantilla que se identifica mediante el parámetro projectRequestTemplate del archivo master-config.yml, si el parámetro no se encuentra definido, el servidor API genera una plantilla predeterminada que crea un proyecto con el nombre solicitado y asigna al usuario solicitante el rol de "administrador" para ese nuevo proyecto. (Red Hat, Inc., 2015a).

Para crear una propia plantilla de proyecto personalizada se sugiere seguir los siguientes pasos:

a) Comenzar con la plantilla de proyecto predeterminada actual:

```
$ oc adm create-bootstrap-project-template -o yaml> template.yaml
```

b) Usar un editor de texto para modificar el archivo template.yaml agregando objetos o modificando objetos ya existentes.

c) Cargar la nueva plantilla creada:

```
$ oc create -f template.yaml -n default
```

d) Modificar el archivo master-config.yaml en la siguiente sección para hacer referencia a la plantilla ahora cargada:

```
...
projectConfig:
  projectRequestTemplate: "default / project-request"
  ...
```

Cuando se envía una solicitud de proyecto, la API sustituye los siguientes parámetros en la plantilla:

a) **Nombre Del Proyecto:** El nombre del proyecto. Necesario.

b) **Nombre Visible:** El nombre para mostrar del proyecto. Puede estar vacío

c) **Descripción Del Proyecto:** La descripción del proyecto. Puede estar vacío

d) **Administrador Del Proyecto:** El nombre de usuario del usuario administrador.

e) **Usuario Que Solicita El Proyecto:** El nombre de usuario del usuario solicitante.

- **Deshabilitar la autoaprovisionamiento**

Al eliminar la función self-provisioners de los grupos de usuarios autenticados en el clúster, se deniegan los permisos para el autoaprovisionamiento de cualquier proyecto nuevo.

```
$ oc adm policy remove-cluster-role-from-group self-provisioner /
  system:authenticated system:authenticated:oauth
```

Al deshabilitar el auto aprovisionamiento, se debe configurar en el archivo master-config.yaml el parámetro projectRequestMessage para instruir a todos los desarrolladores sobre cómo se debe solicitar un nuevo proyecto. Este parámetro es una cadena que se presentará al desarrollador en la consola web y también en la línea de comandos cuando intente aprovisionar un proyecto por cuenta propia.
Por ejemplo:

a) Póngase en contacto con su administrador en projectname@example.com para solicitar un proyecto.

b) Para solicitar un nuevo proyecto, complete el formulario de solicitud de proyecto ubicado en http://www.sitio.com.

Ejemplo de la sección a modificar del archivo master-config.yaml:

```
...
projectConfig:
  ProjectRequestMessage: "message"
  ...
```

2.2.5. Administración de pods

Este tema describe la administración de pods además de la configuración sobre la cantidad de ancho de banda que pueden usar.

- **Visualizar pods y sus estadísticas**

Puede visualizar las estadísticas de uso de los pods, que proporcionan información de tiempos de ejecución para los contenedores y estadísticas de uso que incluyen CPU, memoria RAM y consumo de almacenamiento, se puede realizar con el siguiente comando:

```
$ oc adm top pods
NAME                         CPU(cores)   MEMORY(bytes)
hawkular-cassandra-1-pqx6l   219m         1240Mi
hawkular-metrics-rddnv       20m          1765Mi
heapster-n94r4               3m           37Mi
```

- **Limitar el ancho de banda disponible a los pods**

Puede aplicar la configuración de tráfico de calidad de servicio a un pod y limitar efectivamente el ancho de banda disponible. El tráfico de salida (desde el pod) se maneja mediante la vigilancia, que simplemente elimina los paquetes que superan la velocidad configurada. El tráfico de ingreso (al pod) se maneja configurando los paquetes en cola para manejar los datos de manera efectiva. Los límites que usted coloca en un pod no afectan el ancho de banda de otros pods.

Para limitar el ancho de banda en un pod:

a) Escribir un archivo JSON de definición de objeto y especificar la velocidad del tráfico de datos del pod mediante las anotaciones kubernetes.io/ingress-bandwidthy y

kubernetes.io/egress-bandwidth. Por ejemplo, para limitar el ancho de banda de ingreso y entrada de pod a 10 M / s:

```
{
    "kind": "Pod",
    "spec": {
        "containers": [
            {
                "image": "openshift/hello-openshift",
                "name": "hello-openshift"
            }
        ]
    },
    "apiVersion": "v1",
    "metadata": {
        "name": "iperf-slow",
        "annotations": {
            "kubernetes.io/ingress-bandwidth": "10M",
            "kubernetes.io/egress-bandwidth": "10M"
        }
    }
}
```

b) Crea el pod usando la definición de objeto:

```
oc create -f <file_or_dir_path>
```

Se pueden realizar muchas más acciones de configuración en los pods cómo establecer la duración o tiempo de vida de este o limitar el acceso a un pod con un firewall que requieren una mayor configuración.

La configuración de OpenShift abarca un sinnúmero de opciones que requieren un amplio conocimiento sobre las tecnologías Kubernetes y Docker que son en las que se basa OpenShift que en su mayoría deben ser realizadas por comandos de consola, en este documento se explica conceptos básicos que son necesarios para el control de proyectos en OpenShift, la mayoría de acciones necesarias para creaciones de proyectos y despliegue de aplicaciones, también se las puede realizar mediante la interfaz gráfica o consola Web que brinda OpenShift.

2.3. Aplicación de prueba a analizar

2.3.1. Metodología SCRUM

"El desarrollo de software ágil es una de las metodologías en el desarrollo de un software más usadas. La palabra ágil significa ser rápido, liviano y libre de movimiento. Ágil es una palabra utilizada para describir un concepto de modelo de proceso que es diferente de los conceptos de modelo de proceso existentes. En el desarrollo de software ágil las interacciones y el personal son más importantes que el proceso y las herramientas, un software en funcionamiento es más importante que una documentación completa, la colaboración con los clientes es más importante que la negociación del contrato y ser más receptivo a los cambios es más importante que seguir el plan. Sin embargo, al igual que otros modelos de procesos, El desarrollo ágil tiene sus propias ventajas y no es adecuado para todo tipo de proyectos, productos, personas y situaciones." (Guna Permana & Stikom Bali, 2015).

- **Definición de SCRUM**

"Scrum fue desarrollado por Jeff Sutherland en 1993 y su objetivo es convertirse en un desarrollo y gestión metodológica que sigue los principios de la metodología ágil. Scrum es un marco de software receptivo para el desarrollo de proyectos de software y gestión de productos o desarrollo de aplicaciones. El foco está en la estrategia, flexibilidad y un desarrollo general de productos donde el equipo de desarrollo trabaje como una unidad para alcanzar objetivos comunes. Scrum tiene un complejo proceso en el que muchos factores afectan el resultado final." (Guna Permana & Stikom Bali, 2015).

Scrum es una metodología dentro del cual las personas pueden abordar complejos problemas de adaptación, productiva y creativamente, entregando productos del mayor valor posible este marco de trabajo es ligero, simple de entender, pero difícil de dominar.

Scrum se ha utilizado para gestionar el desarrollo de productos complejos desde principios de la década de 1990, scrum no es un proceso o una técnica para construir productos; más bien, es un marco dentro del cual se puede emplear varios procesos y técnicas. Scrum deja en claro la eficacia relativa de las prácticas de desarrollo y gestión de su producto para que pueda mejorar.

El marco de Scrum consiste en grupos scrum y sus roles asociados, eventos, artefactos y reglas. Cada componente dentro del marco sirve para un propósito específico y es esencial para el éxito y uso de Scrum. (Schwaber & Sutherland, 2013).

Scrum como una metodología de desarrollo ágil fundamenta la creación de ciclos breves denominados “Sprints”. Por esto, para entender el ciclo de desarrollo Scrum primero se deben conocer las fases que definen el desarrollo ágil, que se ilustra en la Figura 14.

Figura 14: Ciclo de desarrollo ágil
Fuente (Gallego & Domingo Troncho, 2014)

Scrum gestiona estas iteraciones a través de reuniones diarias, uno de los elementos fundamentales de esta metodología, como se observa en la Figura 15.

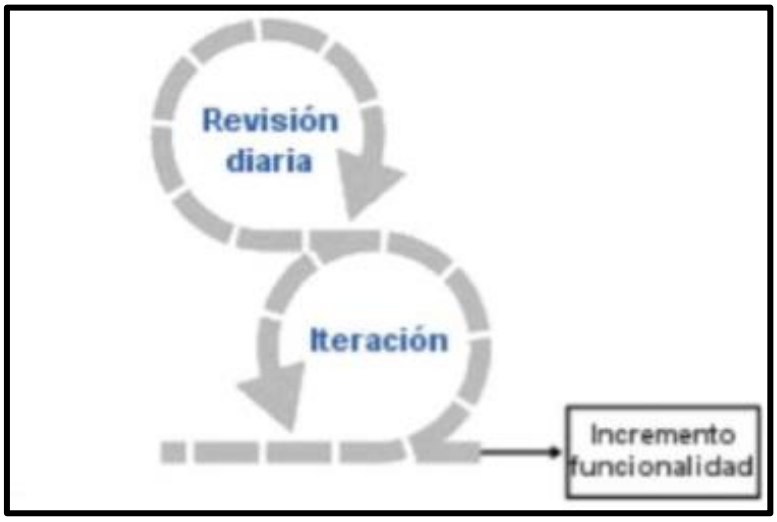

Figura 15: Ciclo principal de Scrum
Fuente (Gallego & Domingo Troncho, 2014)

- **Componentes de SCRUM**

Scrum se puede dividir de forma general en 3 fases como son las reuniones que forman parte de los artefactos de esta metodología junto con los roles y los elementos scrum. (Gallego & Domingo Troncho, 2014)

- **Las Reuniones**

a) Planificación del Backlog: Se define un documento con los requisitos del sistema por prioridades. En esta fase se define también la planificación del Sprint 0, en la que se decidirá cuáles van a ser los objetivos y el trabajo que hay que realizar para esa iteración, además, en esta reunión se obtiene un Sprint Backlog, que es la lista de tareas y que es el objetivo más importante del Sprint.

b) Seguimiento del Sprint: En esta fase se hacen reuniones diarias en las que las 3 preguntas principales para evaluar el avance de las tareas serán:

a) ¿Qué trabajo se realizó desde la reunión anterior?
b) ¿Qué trabajo se hará hasta una nueva reunión?
c) Inconvenientes que han surgido y qué hay que solucionar para poder continuar.

c) Revisión del Sprint: Cuando se finaliza el Sprint se realizará una revisión del incremento que se ha generado.

- **Los Roles**

Los roles se dividen en dos grupos:

Las personas comprometidas con el proyecto y el proceso Scrum:

a) Product Owner
b) Scrum Master
c) Equipo De Desarrollo

Las personas que no son parte del proceso de Scrum, es necesario una retroalimentación dé la salida del proceso para así poder revisar y planear cada sprint:

a) Usuarios
b) Stakeholders
c) Managers

- **Elementos de Scrum**

Los elementos que forman a Scrum son:

a) **Product Backlog:** Es una lista creada y gestionada por el cliente en la cual se especifican los requisitos del producto o los que irá adquiriendo luego de varias iteraciones. Es necesario que antes de empezar el primer Sprint se definan cuáles van a ser los objetivos del producto y tener la lista de los requisitos ya definida. No es necesario que sea muy detallada, simplemente deberá contener los requisitos principales para que el equipo pueda trabajar.

b) **Sprint Backlog:** Es la lista ordenada de tareas que se plantea para la ejecución del producto, donde se busca descomponer en unidades más concretas y poder ejecutar las tareas importantes o eliminar varias tareas irrelevantes. Generalmente, las tareas a completar se suelen gestionar mediante el Scrum Taskboard, a cada objetivo se le asignan las tareas necesarias para llevarlo a cabo, se usan post-its que se van moviendo de una columna a otra para cambiar el estado.

c) **Incremento:** Evalúa los resultados que se van obteniendo luego de completar varias iteraciones, así, según los resultados que se obtengan, el cliente puede ir haciendo los cambios necesarios y replanteando el proyecto. (Gallego & Domingo Troncho, 2014).

- **Descripción General de la Metodología**

  - **Fundamentación**

Las principales razones del uso de un ciclo de desarrollo iterativo e incremental de tipo scrum para la ejecución de este proyecto son:

a) Las funciones del sistema permiten que sea desarrollado sobre una funcionalidad mínima e implementar nuevas funcionalidades sobre esta, así como editar las características de las funcionalidades nuevas o de las que ya están desarrolladas.

b) Es muy probable la implementación de funcionalidades adicionales a las que se ha identificado en un inicio.

c) Al tener como finalidad un estudio, la aplicación se puede desarrollar alrededor de una funcionalidad con complejidad mínima.

- **Personas y roles del proyecto**

Tabla 6: Roles del proyecto

Persona	Contacto	Rol
Iván Darío Rojas Rojas	idrojasr@utn.edu.ec	Coordinador
Iván Darío Rojas Rojas	idrojasr@utn.edu.ec	Gestor del Producto

Fuente Propia

- **Artefactos**

  a) Documentos

    a) Pila de producto o Product Backlog

    b) Pila de sprint o Sprint Backlog

  b) Sprint

  c) Incremento

- **Pila del producto**

En la Tabla 7 se identifican los principales requerimientos del proyecto: prioridad, siendo 1 el menos importante y 10 el más importante, estado que puede ser “Por Hacer”, “En progreso” y “Hecho”, estimación siendo 5 el requerimiento de menor complejidad y 13 el que se espera sea el más complejo por último se presenta el sprint al que pertenece el requerimiento.

Tabla 7: Product Backlog

Id	Funcionalidad Requerida	Prioridad	Estado	Estimación	Sprint
1	Desarrollo de la funcionalidad que permite el ingreso.	8	Hecho	5	1
2	Desarrollo de postulación de beca	10	Hecho	8	1
3	Desarrollo de carga de archivos de requisitos.	7	Hecho	13	2
4	Desarrollo de revisión de postulantes y visualización de archivos cargados.	10	Hecho	13	2
5	Desarrollo de aprobar, rechazar y agregar observación de un requisito.	6	Hecho	7	2
6	Desarrollo de gestión de becas y requisitos.	8	Hecho	6	3
7	Desarrollo de salida del sistema y cabecera de accesos rápidos.	5	Hecho	5	3

Fuente Propia

- **Pila del sprint**

Se identificaron las tareas necesarias para el desarrollo del sistema de gestión de becas de prueba de la Universidad Técnica del Norte por cada sprint establecido previamente, se detallan en la Tabla 8.

Tabla 8: Tareas a implementar por sprint

Sprint	Id	Tema	Funcionalidad
1	1.1	Implementación del formulario de ingreso de gestión de becas.	Ingreso
	1.2	Implementación de validaciones del formulario de ingreso.	Ingreso
	1.3	Implementación de selección de becas disponibles.	Postulación
	1.4	Implementación de registro en la base de datos del estudiante y la beca postulada.	Postulación
2	2.1	Implementación de formulario de carga de archivo, con opción de eliminación.	Carga de archivos
	2.2	Almacenamiento en la base de datos del binario del archivo cargado en el formulario.	Carga de archivos
	2.3	Implementación del formulario de presentación de los postulantes registrados.	Revisión de postulantes
	2.4	Implementación de visualización en el navegador de los archivos cargados por postulantes.	Revisión de postulantes
3	3.1	Implementación de funcionalidades de aprobar, rechazar e ingreso de observación con formulario desplegado.	Aprobar, rechazar y agregar observación de un requisito
	3.2	Implementación de Creación, Edición y Eliminación de becas.	Gestión de becas
	3.3	Implementación de Creación, Edición y Eliminación de requisitos de una beca.	Gestión de becas
	3.4	Implementación de los accesos rápidos en la cabecera del sistema.	Cerrar sesión y accesos rápidos
	3.5	Implementación de cierre de sesión del sistema.	Cerrar sesión y accesos rápidos

Fuente Propia

- **Sprint 1**

Nombre

Desarrollo de formulario de ingreso con sus funciones e implementación de la funcionalidad que permite a un estudiante postular a una beca.

Descripción

El Sprint 1 tuvo como objetivo la creación de las funciones necesarias para permitir el ingreso al sistema incluidas las validaciones respectivas, además de la función de postulación a una beca de un estudiante en el sistema.

Resultado

El usuario puede ingresar en el sistema como estudiante o administrador dependiendo de los permisos que posea, un alumno puede postular a una beca para después proceder a completar los requisitos necesarios.

a) **Equipo de desarrollo:** 1
b) **Días de duración del Sprint:** 5
c) **Fecha de Inicio:** 01/10/2018
d) **Fecha de Finalización:** 06/10/2018
e) **Número óptimo de horas de trabajo por persona (80%):** 30
f) **Número de horas de trabajo del equipo:** 30

- **Sprint 2**

Nombre

Carga y visualización de archivos cargados por un estudiante e implementación de funcionalidad de revisión de los requisitos cargados en formato PDF.

Descripción

En el Sprint 2 se implementó un formulario donde se puede cargar archivos PDF y otro en el que se puede visualizarlo en una pestaña del navegador y la función de eliminarlo.

Resultado

El usuario normal del sistema tiene la opción de cargar archivos PDF uno por cada requisito de la beca postulada y el administrador del sistema procederá a evaluar dichos archivos cargados.

a) **Equipo de desarrollo:** 1
b) **Días de duración del Sprint:** 5
c) **Fecha de Inicio:** 07/10/2018
d) **Fecha de Finalización:** 12/10/2018
e) **Número óptimo de horas de trabajo por persona (80%):** 30
f) **Número de horas de trabajo del equipo:** 30

- **Sprint 3**

Nombre

Gestión de entidades necesarias en el sistema, cierre de sesión y creación de accesos rápidos en la cabecera.

Descripción

El Sprint 3 permitió la implementación de una cabecera en donde se tienen accesos rápidos y la implementación de la opción en ella de cerrar sesión.

Resultado

El usuario normal y el usuario administrador tienen diferentes opciones dependiendo del su rol, el administrador puede gestionar las becas y los requisitos, la opción de cerrar sesión es común para los dos roles.

a) **Equipo de desarrollo:** 1

b) **Días de duración del Sprint:** 5

c) **Fecha de Inicio:** 13/10/2018

d) **Fecha de Finalización:** 18/10/2018

e) **Número óptimo de horas de trabajo por persona (80%):** 30

f) **Número de horas de trabajo del equipo:** 30

En la Tabla 9 se listan las tareas realizadas por cada Sprint, la funcionalidad a la que pertenece, el estado de dicha tarea y el resultado que se obtuvo al finalizar la misma.

Tabla 9: Revisión de resultados alcanzados

Sprint	Tarea	Estado de Tarea	Tarea cumplida	Funcionalidad	Resultado alcanzado en el sprint	Estado del Sprint
1	Implementación del formulario de ingreso de gestión de becas.	F	SI	Ingreso	El usuario puede ingresar en el sistema como estudiante o administrador dependiendo de los permisos que posea, un alumno puede postular a una beca para después proceder a completar los requisitos necesarios.	T
	Implementación de validaciones del formulario de ingreso.	F	SI	Ingreso		
	Implementación de selección de becas disponibles.	F	SI	Postulación		
	Implementación de registro en la base de datos del estudiante y la beca postulada.	F	SI	Postulación		
2	Implementación de formulario de carga de archivo, con opción de eliminación.	F	SI	Carga de archivos	El usuario normal del sistema tiene la opción de cargar archivos PDF uno por cada requisito de la beca postulada y el administrador del sistema procederá a evaluar dichos archivos cargados.	T
	Almacenamiento en la base de datos del binario del archivo cargado en el formulario.	F	SI	Carga de archivos		
	Implementación del formulario de presentación de los postulantes registrados.	F	SI	Revisión de postulantes		

	Implementación de visualización en el navegador de los archivos cargados por postulantes.	F	SI	Revisión de postulantes		
3	Implementación de funcionalidades de aprobar, rechazar e ingreso de observación con formulario desplegado.	F	SI	Aprobar, rechazar y agregar observación de un requisito	El usuario normal y el usuario administrador tienen diferentes opciones dependiendo del su rol, el administrador puede gestionar las becas y los requisitos, la opción de cerrar sesión es común para los dos roles.	T
	Implementación de Creación, Edición y Eliminación de becas.	F	SI	Gestión de becas		
	Implementación de Creación, Edición y Eliminación de requisitos de una beca.	F	SI	Gestión de becas		
	Implementación de los accesos rápidos en la cabecera del sistema.	F	SI	Cerrar sesión y accesos rápidos		
	Implementación de cierre de sesión del sistema.	F	SI	Cerrar sesión y accesos rápidos		

Fuente Propia

2.3.2. Desarrollo de la aplicación

La aplicación de prueba se desarrolló en el lado del servidor en el lenguaje PHP v7.0 y un servidor de base de datos PostgreSQL 9.6 para el almacenamiento de los datos que requiere la aplicación, para la parte del usuario o interfaz gráfica se usó HTML 5, CSS3 y JavaScript, incluido el framework Materialize 1.0 de Google.

La funcionalidad de la aplicación está basada en la administración de becas dentro la Universidad Técnica del Norte en la ciudad de Ibarra, en la cual se pretende controlar qué estudiantes están postulando a una determinada beca, así como los estudiantes que hayan sido aprobados para ser acreedores de una beca, brinda los respectivos CRUD de las entidades Universidades, Becas, Provincias, Becario y Postulante, además de algunos resúmenes de los datos más relevantes del sistema como resumen de acreditaciones por beca, cabe mencionar que esta aplicación es netamente para pruebas de rendimiento en OpenShift con el fin de obtener información con la cual analizar estadísticamente.

Se manejó la arquitectura MVC (Modelo, Vista y Controlador) para lograr manejar de una mejor manera la lógica del negocio y se posea un mejor manejo de cambios en el mantenimiento o solución de problemas, por esta razón la estructura de directorios es muy simple de entender y administrar como se presenta en la Figura 16.

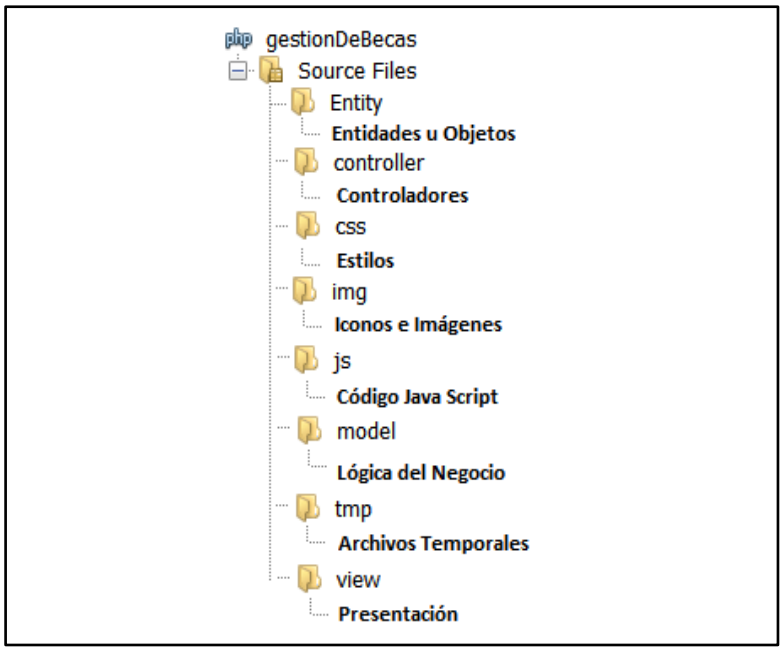

Figura 16: Estructura de archivos de la aplicación de prueba

Fuente: Propia

En la base de datos se han creado las entidades consideradas necesarias para el correcto funcionamiento del flujo de gestión de becas, entre las cuales estarían las principales como becas la cual almacena las diferentes opciones a las cuales un estudiante puede postular, usuarios que contendrá los alumnos que pueden ingresar al sistema y postulantes que son los usuarios quienes ya han postulado a una beca específica, en la Figura 17 se muestra el diagrama lógico.

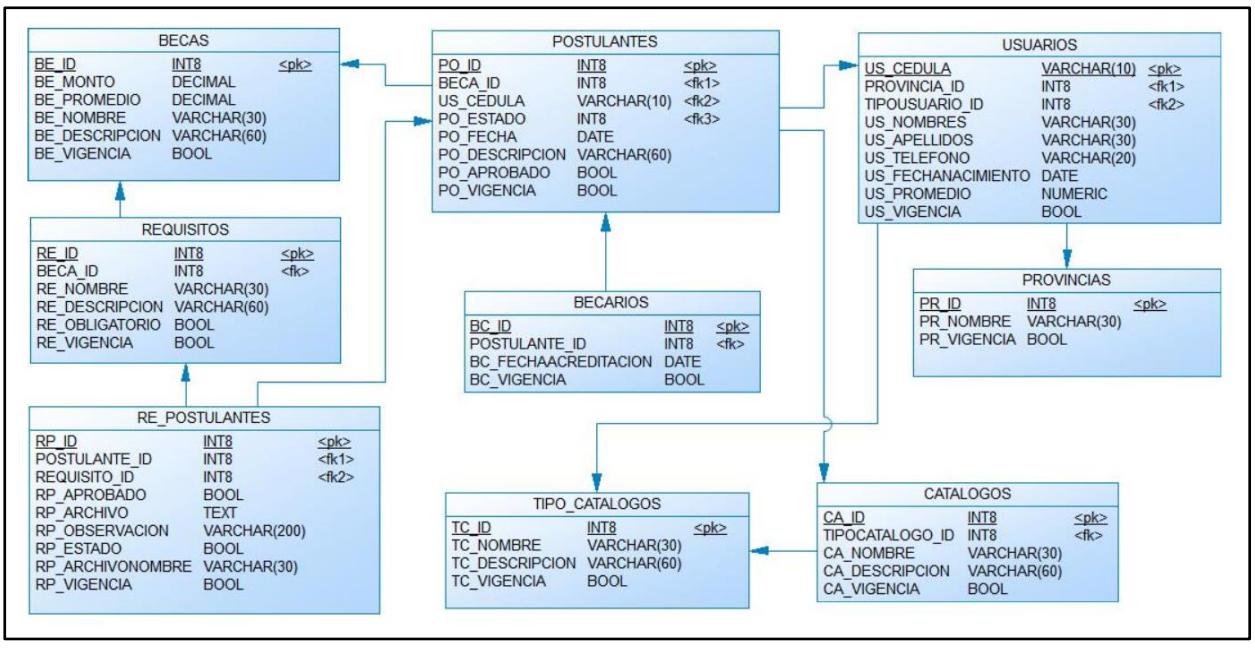

Figura 17: Diagrama Lógico de Base de Datos Desarrollado en PowerDesigner

Fuente: Propia

Las pantallas de interfaz gráfica de usuario en el sistema están basadas todas en un mismo formato, con una ocupación total de la pantalla del 80%, un color rojo en su mayoría y azul para botones de acciones, además contienen una cabecera con el logotipo de la Universidad Técnica del Norte a la izquierda y accesos rápidos a la derecha.

Este sistema de prueba para la gestión de becas de Universidad Técnica del Norte contiene los siguientes módulos:

a) Postulación del estudiante a la beca
b) Administración de Becas y Requisitos
c) Revisión de la beca por parte del Administrador
d) Opciones del rol Administrador

En la Figura 18 se detalla una de las pantallas principales dividida en seis secciones, en este caso la del Administrador, secciones que son detalladas posteriormente en la Tabla 10.

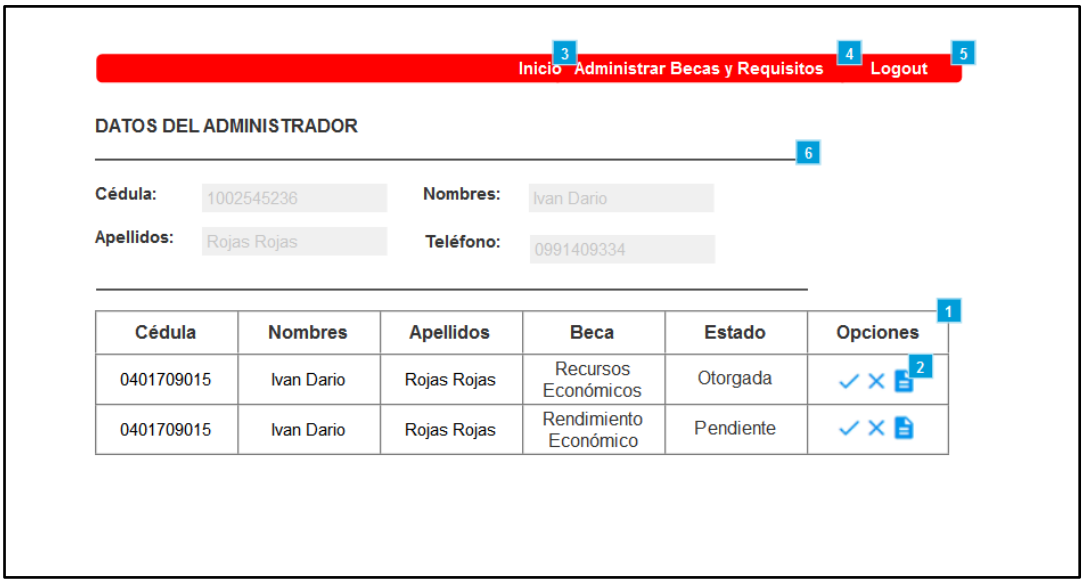

Figura 18: Prototipo Pantalla de Administración desarrollada en AXURE

Fuente: Propia

Tabla 10: Detalles de secciones para pantalla del administrador

No.	Tabla / Opción / Campo	Descripción
1	Tabla que lista las diferentes becas postuladas.	Toda la información de la tabla se presenta de manera no Editable.
2	Opciones	El sistema permite: **a)** Aprobar la beca **b)** Rechazar la beca **c)** Revisar Requisitos

3	Inicio	Permite regresar al home del sistema.
4	Administrar Becas y Requisitos	Permite redireccionar a la pantalla de Administración de Becas y Requisitos.
5	Logout	Permite salir del sistema.
6	Datos del Administrador	Muestra la información del Administrador conectado

Fuente: Propia

2.3.3. Despliegue de la aplicación

Para poder desplegar una aplicación en OpenShift es necesario tener un repositorio con los archivos de la misma como GitHub, BitBucket o SourceForge, ya que OpenShift utiliza el repositorio para descargar el código fuente de la aplicación que se desea desplegar en la plataforma, se puede usar cualquier medio para subir los archivos de la aplicación que se va a desplegar, en este caso se usó Git un software el cual está desarrollado para almacenar y controlar los cambios que se realizan en el código fuente mediante la creación de versiones, en la Figura 19 se presenta el código fuente almacenado en GitHub.

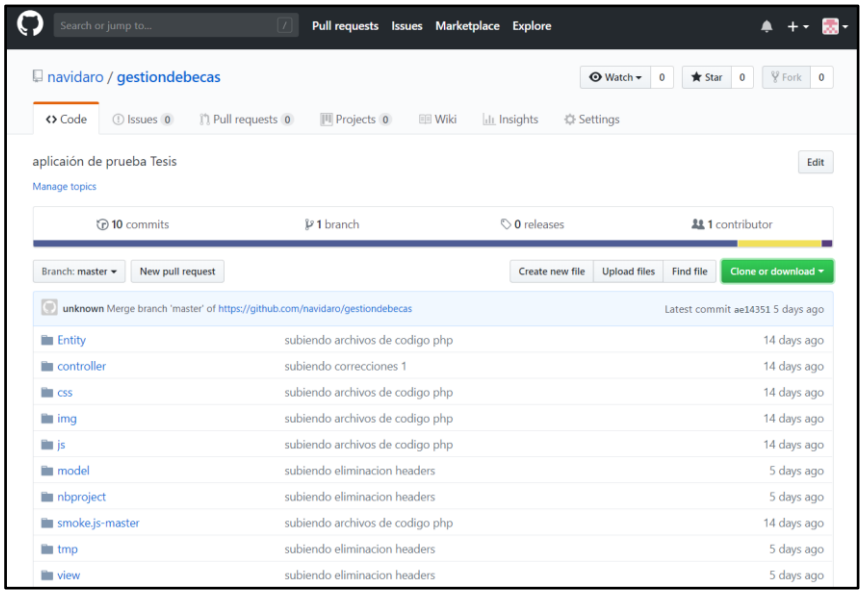

Figura 19: Código Fuente de la Aplicación almacenada en GitHub
Fuente: Propia

Se procede a la creación y despliegue del servicio de base de datos necesario, en este caso PostgreSQL que se elige del extenso catálogo que posee OpenShift que nos permite configurarlo a nuestras necesidades, aspectos como la versión a usar y datos necesarios para la conexión que después pueden ser modificados si se requiere como: usuario,

contraseña y nombre de la base de datos, se visualiza el servicio de la base de datos creada en la Figura 20.

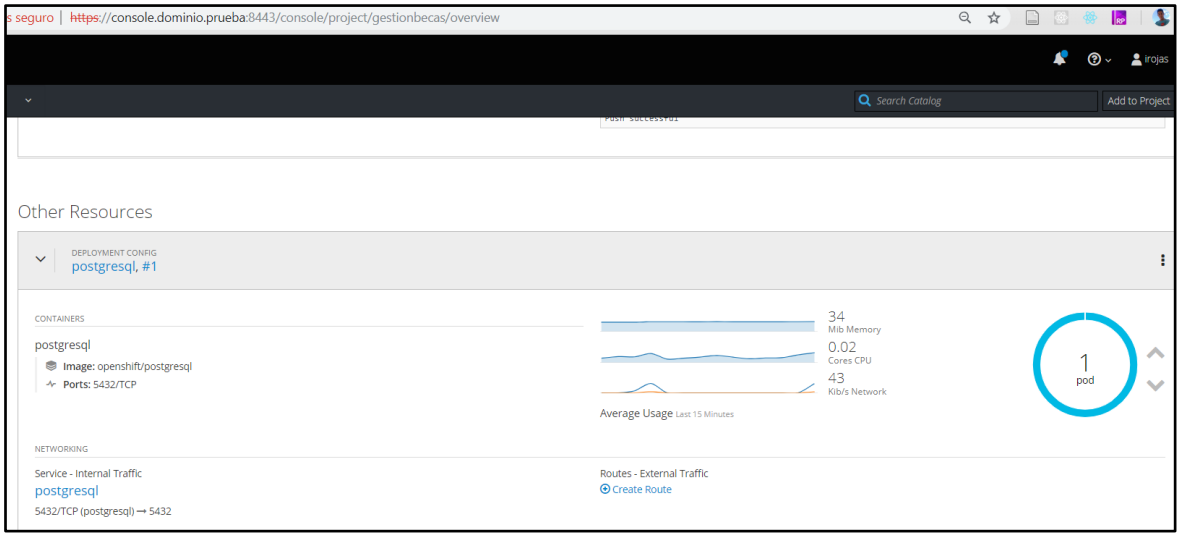

Figura 20: Servicio de Base de Datos Desplegado en OpenShift
Fuente: Propia

Posteriormente a paso este paso se deberá configurar la el código fuente que realizará la conexión a la base de datos con los datos adecuados que se configurarán en OpenShift, como nombre de usuario, contraseña, nombre de la base de datos y la IP del servicio de la base de datos, que se la puede obtener desde la consola web de OpenShift en la opción Applications > Services e ingresando en el servicio de la base de datos en la cual se hará conexión, las credenciales de acceso se las obtiene de Resources > Secrets y buscando el secreto que hace referencia a nuestro servicio, con estos datos correctamente configurados la aplicación tendría acceso al servicio de la base de datos.

La creación del contenedor que almacena la aplicación es igual a la creación del servicio de base de datos que se realiza por medio de la consola Web, se elige el lenguaje en el que está desarrollado el código fuente y se procede a ingresar la dirección del repositorio en donde se encuentran los archivos de código y el nombre que se desea tenga la aplicación, los contenedores de la base de datos y de apache creados y desplegados se visualizan en la Figura 21.

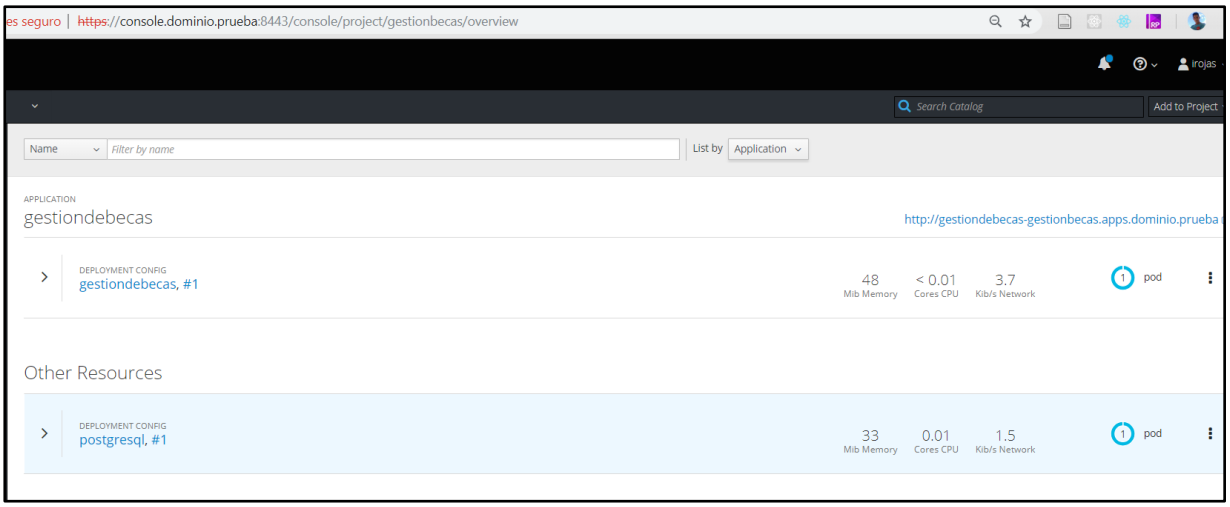

Figura 21: Contenedores Apache y PostgreSQL desplegados en OpenShift

Fuente: Propia

El Despliegue de la aplicación como se mira en la sección 2.2 del documento se puede realizar por la CLI de OpenShift o por la consola web, la manera más sencilla de hacerlo es con el uso de la consola web que brinda OpenShift si se necesitan configuraciones más avanzadas se requerirá usar la CLI de OpenShift.

2.4. Rendimiento de aplicaciones en Servidores Tradicionales.

Para esta evaluación de rendimiento que de la aplicación de Gestión de Becas que ha sido desplegada en un servidor CentOS 7 bajo el servidor de aplicaciones de Apache y con el servidor de base de datos PostgreSQL, se le sometió a una carga de peticiones en diferentes cantidades, su comportamiento se evaluó midiendo tiempos de respuesta y porcentajes de error, para este fin se usó la herramienta JMeter[16] misma que se usa para cargas de usuarios virtuales en aplicaciones web, los valores de prueba de esta evaluación son los siguientes:

Servidor: 192.168.61.134

a) Puerto: 80
b) Ruta: /controller/controllerLogin.php
c) solicitud: Listar Postulantes con becas postuladas

[16] JMeter herramienta Java usada para simular una carga pesada en un servidor, grupo de servidores, red u objeto para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga. https://jmeter.apache.org

La prueba de carga se realizó con usuarios virtuales en intervalos de 100, partiendo del número 100 hasta el número 1000, se ejecutó la prueba tres veces por cada número de usuarios y haciendo uso del promedio de estos tres, con el fin de mejorar la precisión de los resultados, recopilando por cada uno los parámetros antes mencionados de evaluación, los resultados de la prueba de carga son listados en la Tabla 11.

Tabla 11: Resultado de rendimiento obtenidos con servidor CentOS

Número de Usuarios	Media (ms)	Desviación Estándar	Porcentaje de Error	Rendimiento (peticiones/min)
100	693,33	257,25	0,00	60.00
200	3855,33	5344,61	1,83	55,10
300	7514,66	7603,41	6,11	53,80
400	10057,66	8669,63	13,66	54,96
500	12309,00	9616,34	20,87	54,83
600	13211,00	9402,22	23,72	56,13
700	14409,00	9590,04	29,14	56,16
800	15498,00	9660,67	34,21	56,83
900	15974,00	9655,52	35,93	57,60
1000	16789,33	9537,00	40,56	57,50

Fuente: Propia

Variables analizadas:

a) **Número de Usuarios:** Número de peticiones enviadas a la aplicación en el servidor.
b) **Media:** Tiempo promedio en recibir respuesta a las peticiones.
c) **Desviación Estándar:** Valor en que cada tiempo de respuesta se aleja del promedio.
d) **Porcentaje de error:** respuestas fallidas a las peticiones por cada 100 peticiones.
e) **Rendimiento:** Transacciones o peticiones que han podido ser procesadas por minuto.

Con los datos obtenidos se realiza un análisis estadístico para determinar cuán correlacionados están, con una prueba de normalidad se verifica cuáles de los parámetros evaluados encajan en lo habitual o corriente, es decir incrementan o decrecen de manera normal sin excesos, esto permitirá saber cuál de ellos puede tener correlación con la variable de incremento en 100,es decir el número de usuarios.

En la Tabla 12 se visualiza la prueba de normalidad, como se observa la cantidad de datos analizados es 10 lo que lleva a tomar los resultados de la prueba de Shapiro Wilk que se usa cuando el análisis se realiza en muestras menores de 50, y se toma los parámetros en donde la columna de significación es mayor a 0.05 lo que demuestra normalidad en ellos (Meyers L, Gamst G, Guarino A, 2016), en este caso se trabaja con usuarios, media y error en razón de ser normales.

Tabla 12: Prueba de normalidad con Resultados obtenidos en servidor CentOS

	Pruebas de normalidad					
	Kolmogorov-Smirnov			**Shapiro-Wilk**		
	Estadístico	**gl**	**Sig**	**Estadístico**	**gl**	**Sig**
Usuario	0,096	10	0,200	0,970	10	0,892
Media	0,193	10	0,200	0,900	10	0,220
Error	0,138	10	0,200	0,934	10	0,489
Rendimiento	0,478	10	0,000	0,432	10	0,000
Desviación Estándar	0,296	10	0,013	0,657	10	0,000

Fuente: Propia

Basado en el análisis de correlación de Pearson de la Tabla 13 se demuestra que existe una fuerte correlación positiva de 0,955 entre los usuarios y la media además también una fuerte correlación positiva de 0,993 entre usuarios y porcentaje de error, teniendo en cuenta que una correlación positiva máxima es de 1 y una mínima es de 0 (Meyers L, Gamst G, Guarino A, 2016), lo que significa para este estudio que mientras más peticiones se hagan a la aplicación Gestión de Becas en el servidor de CentOS aumenta proporcionalmente junto con ellas el tiempo de respuesta de cada petición y el porcentaje de error de peticiones con un error como respuesta.

Tabla 13: Resultados de prueba de correlación de Pearson

		Usuarios	**Media**	**Error %**
Usuarios	**Correlación de Pearson**	1	0,955	0,993
	Sig. (Bilateral)		0,000	0,000

	N	10	10	10
Media	**Correlación de Pearson**	0,955	1	0,964
	Sig. (Bilateral)	0,000		0,000
	N	10	10	10
Error %	**Correlación de Pearson**	0,993	0,964	1
	Sig. (Bilateral)	0,000	0,000	
	N	10	10	10

Fuente: Propia

Como se mira en la Figura 22 el rendimiento es casi constante en este caso, variando al inicio de la prueba en pequeña escala entre 50 y 60 peticiones por minuto, y estabilizándose al final con esto se demuestra que para el caso de esta prueba con la aplicación desplegada el rendimiento en el servidor CentOS 7 no fue el máximo ni fue constante.

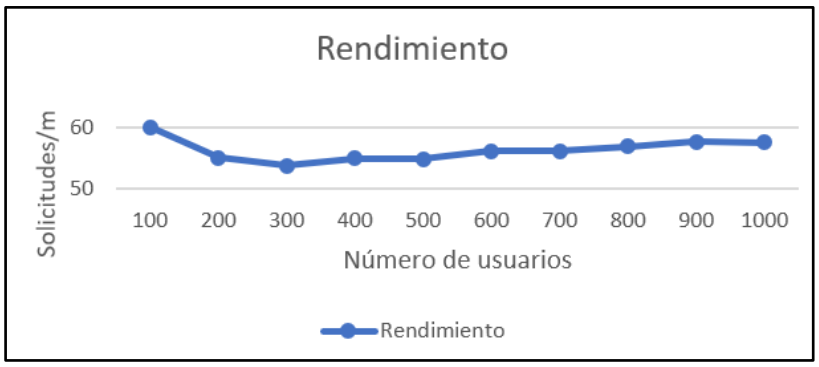

Figura 22: Gráfica del Rendimiento vs el Número de Usuarios

Fuente: Propia

En la Figura 23 se observa la relación que existe entre la media de tiempo de respuesta y el porcentaje de error, llegando a una variación equitativa entre los dos parámetros, con un mayor número de usuarios el porcentaje de error aumenta proporcionalmente a la media de respuesta, llegando el error a casi un 45 por ciento y la media a 17 segundos, manteniendo un incremento normal a la variación de usuarios.

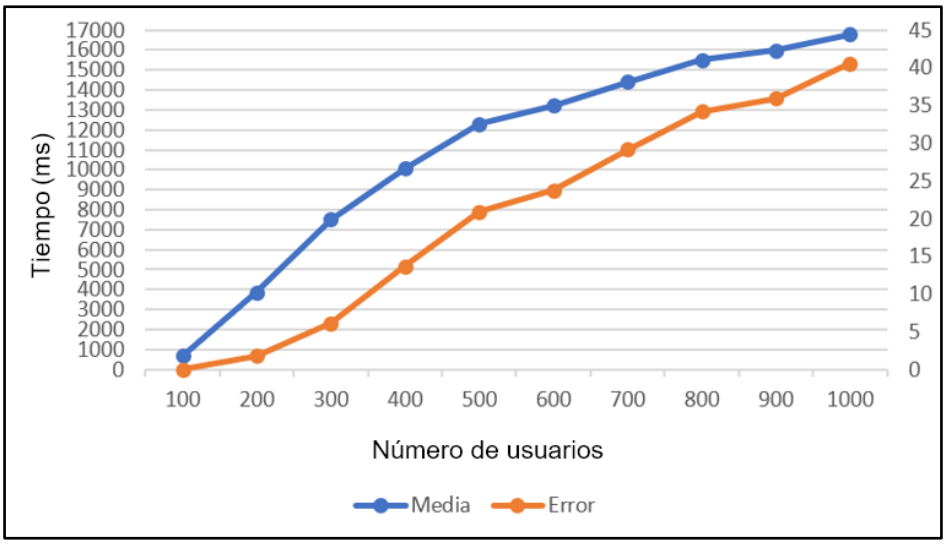

Figura 23: Variación de la media y el error en razón de los Usuarios

Fuente: Propia

2.5. Rendimiento de aplicaciones en OpenShift

Para evaluar el rendimiento que de la aplicación de prueba Gestión de Becas, que ha sido desplegada en OpenShift en dos contenedores separados uno para Apache y otro para PostgreSQL, se sometió al mismo procedimiento de carga que el servidor de CentOS, realizando peticiones en diferentes cantidades, su comportamiento se evaluó midiendo parámetros como tiempos de respuesta y porcentajes de error, para este fin se usó la herramienta JMeter, los valores de prueba de esta evaluación son los siguientes:

Servidor: gestiondebecas-gestionbecas.apps.dominio.prueba

- **a)** Puerto: 80
- **b)** Ruta: /controller/controllerLogin.php
- **c)** solicitud: Postulaciones

La prueba de carga se realizó con la misma cantidad de usuarios virtuales que en la anterior prueba en intervalos de 100, partiendo de 100 usuarios hasta el número 1000 usuarios, se ejecutó la prueba tres veces por cada número de usuarios y haciendo uso del promedio de estos tres, con el fin de mejorar la precisión de los resultados, recopilando por cada uno los datos antes mencionados en la evaluación, los resultados de la prueba de carga son listados en la Tabla 14.

Tabla 14: Resultado de rendimiento obtenidos con OpenShift Origin

Número de Usuarios	Media (ms)	Desviación Estándar	Porcentaje de Error	Rendimiento (peticiones/min)
100	519,33	221,59	0,00	59,7
200	563,33	214,89	0,00	59,9
300	885,33	508,11	0,00	59,8
400	1083,00	619,95	0.00	59,8
500	1590,33	1045,93	0.00	59,6
600	1571,00	880,75	0,00	59,6
700	3901,66	3740,63	0,18	59,1
800	2111,66	1239,80	0,00	59,6
900	3240,00	1979,95	0,00	59,6
1000	4065	2748,59	0,00	59,5

Fuente: Propia

Variables analizadas:

a) **Número de Usuarios:** Número de peticiones enviadas a la aplicación en el servidor.
b) **Media:** Tiempo promedio en recibir respuesta a las peticiones.
c) **Desviación Estándar:** Valor en que cada tiempo de respuesta se aleja del promedio.
d) **Porcentaje de error:** respuestas fallidas a las peticiones por cada 100 peticiones.
e) **Rendimiento:** Transacciones o peticiones que han podido ser procesadas por minuto.

Con los datos obtenidos se realiza el análisis estadístico para determinar cuán correlacionados están, con una prueba de normalidad se verifica cuáles de los parámetros evaluados encajan en lo habitual o corriente, es decir incrementan o decrecen de manera normal sin excesos, esto permitirá saber cuál de ellos puede tener correlación con la variable constante es decir el número de usuarios.

En la Tabla 15 se visualiza la prueba de normalidad, como se observa la cantidad de datos analizados es 10 lo que lleva a tomar los resultados de la prueba de Shapiro Wilk que se usa cuando el análisis se realiza en muestras menores de 50, y se toma los parámetros en donde la columna de significación es mayor a 0.05 lo que demuestra normalidad en ellos (Meyers L,

Gamst G, Guarino A, 2016), en este caso se los parámetros normales son: usuarios, media, rendimiento y desviación.

Tabla 15 Prueba de Normalidad con resultados en OpenShift Origin

	Pruebas de normalidad					
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estadístico	gl	Sig	Estadístico	gl	Sig
Usuario	0,096	10	0,200	0,970	10	0,892
Media	0,227	10	0,152	0,868	10	0,094
Error	0,524	10	0,000	0,366	10	0,000
Rendimiento	0,207	10	0,200	0,880	10	0,131
Desviación	0,264	10	0,047	0,861	10	0,079

Fuente: Propia

Basado en el análisis de correlación de Pearson de la Tabla 16 se demuestra que existe una fuerte correlación positiva de 0,895 entre los usuarios y la media además también una fuerte correlación positiva de 0,756 entre usuarios y la desviación estándar, teniendo en cuenta que una correlación positiva máxima es de 1 y una mínima es de 0 (Meyers L, Gamst G, Guarino A, 2016), lo que significa para este estudio que mientras más peticiones se hagan a la aplicación Gestión de Becas en el servidor de CentOS aumenta proporcionalmente junto con ellas el tiempo de respuesta de cada petición y la desviación estándar.

Tabla 16: Análisis de Correlación de Pearson con Resultados de OpenShift Origin

		Usuarios	Tiempo	Desviación Estándar	Rendimiento	Error %
Usuarios	**Correlación de Pearson**	1	0,895	0,756	-0,584	0,174
	Sig. (Bilateral)		0,000	0,011	0,631	0,077
	N	10	10	10	10	10
Tiempo	**Correlación de Pearson**	0,895	1	0,958	0,512	-0,805
	Sig. (Bilateral)	0,000		0,000	0,130	0,005
	N	10	10	10	10	10
	Correlación de Pearson	0,174	0,512	0,731	1	-0,830

Error %	**Sig. (Bilateral)**	0,631	0,130	0,016		0,003
	N	10	10	10	10	10
Desviación Estándar	**Correlación de Pearson**	0,756	0,958	1	0,731	-0,910
	Sig. (Bilateral)	0,011	0,000		0,016	0,000
	N	10	10	10	10	10
Rendimiento	**Correlación de Pearson**	-0,584	-0,805	-0,910	-0,830	1
	Sig. (Bilateral)	0,077	0,005	0,000	0,003	
	N	10	10	10	10	10

Fuente: Propia

Como se mira en la Figura 24 el rendimiento es prácticamente constante manteniéndose durante toda la prueba entre 60 y 59 peticiones por minuto, prácticamente estable en el transcurso de la prueba, con esto se demuestra que en el caso de esta prueba el rendimiento fue cercano al máximo y constante en el servidor OpenShift Origin.

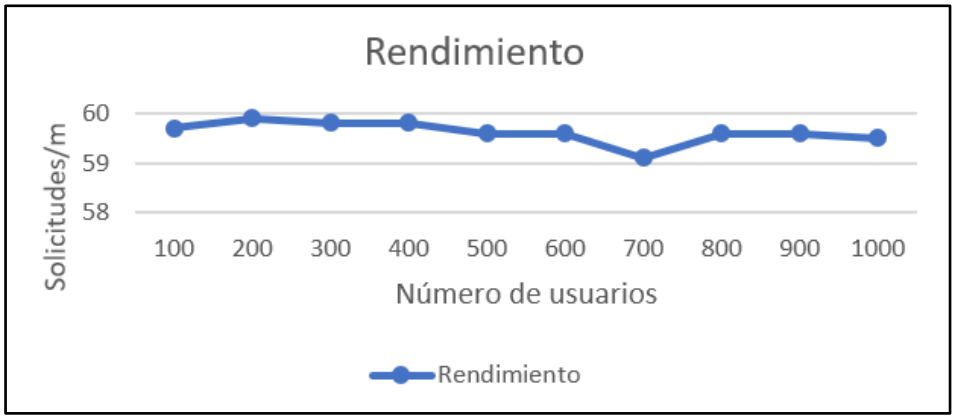

Figura 24: Rendimiento en OpenShift Origin

Fuente: Propia

Como se representa en la Figura 25 el porcentaje de respuestas con error de esta prueba en OpenShift fue prácticamente de 0 por ciento a excepción de 700 usuarios donde hubo 1.18 por ciento de errores.

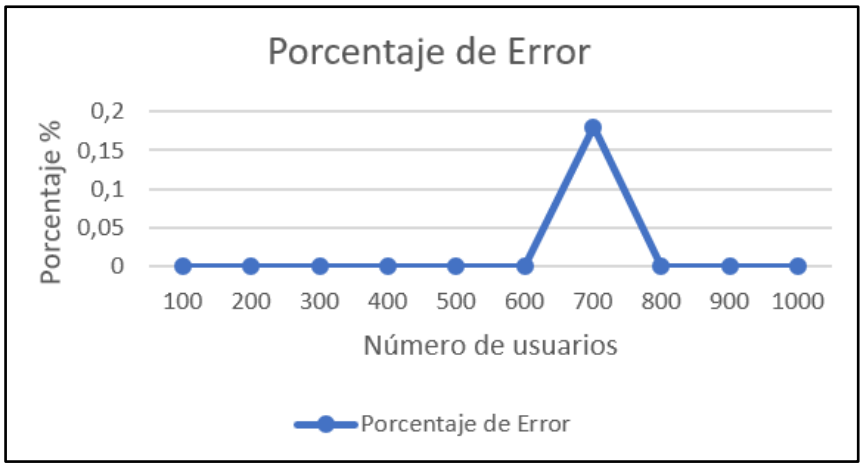

Figura 25: Variabilidad de Error en OpenShift Origin

Fuente: Propia

En la Figura 26 se observa la relación que existe entre la media de tiempo de respuesta y la desviación estándar, llegando a una variación equitativa entre los dos parámetros, con un mayor porcentaje de error la media de respuesta aumenta proporcionalmente manteniendo un incremento normal a la variación de usuarios y error.

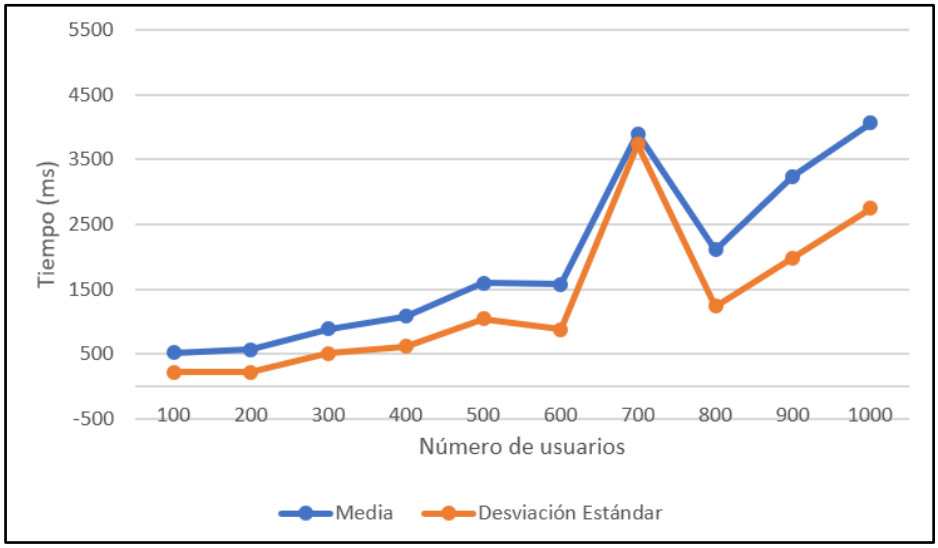

Figura 26: Representación de relación entre media y desviación

Fuente: Propia

2.6. Evaluación de Resultados

En las pruebas realizadas en los dos diferentes ambientes de prueba, todo se llevó a cabo en condiciones iguales, con las mismas características de hardware, la misma aplicación de prueba y las mismas cargas de peticiones, en la Figura 27 se visualizan los datos recopilados del promedio de tiempo de respuesta (media) y la desviación estándar en los dos ambientes, por un lado un servidor CentOS 7 y por el otro lado un servidor CentOS 7 en el cual se desplegó la plataforma como servicio OpenShift Origin.

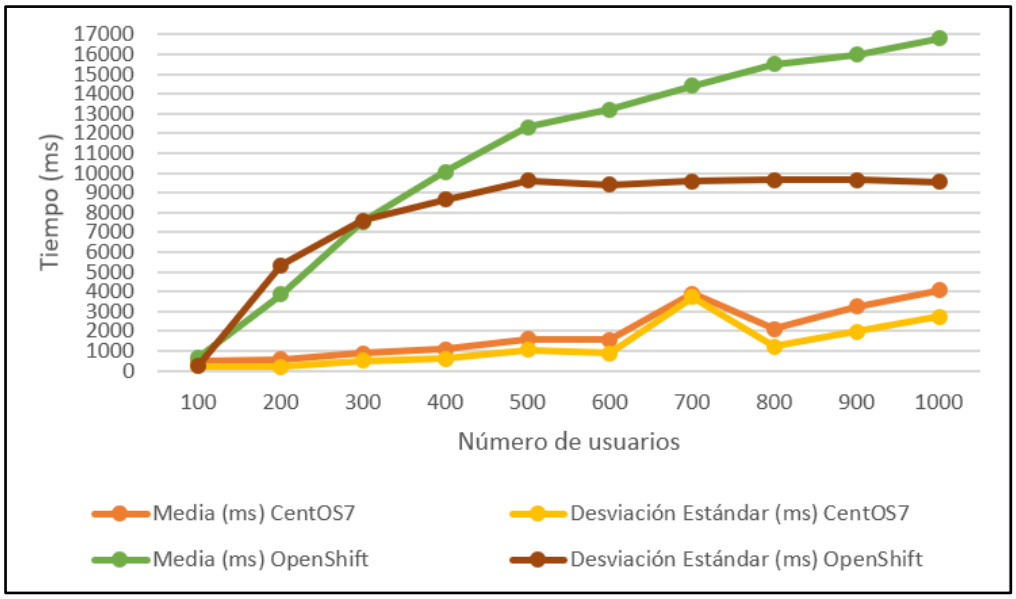

Figura 27: Vista comparativa resultados

Fuente: Propia

Los resultados de las pruebas de correlación arrojan que en los dos ambientes de prueba la media de tiempo de respuesta aumenta conforme se cargan peticiones a la aplicación en el servidor, llegando en CentOS 7 a una media de 693.33 milisegundos con 100 peticiones de carga enviadas y a una media de 16789.33 milisegundos de respuesta con 1000 peticiones, en los resultados de OpenShift se obtuvo una media de 519,33 milisegundos de tiempo de respuesta con 100 peticiones y una media de tiempo de respuesta de 4065 milisegundos, lo que indica que para el caso de esta prueba OpenShift mostró notablemente un menor tiempo promedio en respuesta de peticiones en las diferentes cargas efectuadas.

Los resultados que se obtuvo en porcentaje de error de los dos ambientes fue diferente por un lado en el servidor CentOS se observó un incremento de porcentaje proporcional al número de peticiones enviadas llegando con 1000 peticiones a un porcentaje de 40.56%, en los resultados de la plataforma OpenShift Origin se obtuvo en casi todas las cargas de

peticiones un 0% de error. Lo que indica para esta prueba que OpenShift se desempeñó mejor en cargas elevadas respondiendo correctamente a las peticiones.

En la Tabla 17 se muestra un cuadro comparativo de resumen de los valores con los resultados mínimos y máximos obtenidos en este estudio en los dos ambientes de prueba, como se observa, el desempeño de la aplicación desplegada en OpenShift en el caso de esta prueba fue significativamente mejor en tiempos de respuesta, porcentaje de error y rendimiento.

Tabla 17: Resumen Resultados Rendimiento

	CentOS 7		**OpenShift Origin**	
	Min	**Max**	**Min**	**Max**
Media (ms)	693,33	16789,33	519,33	4065,00
Desviación Estándar (ms)	257,25	9660,67	221,59	3740,63
Error (%)	0,00	40,56	0,00	0,18
Rendimiento (peticiones/min)	53,80	60	59,1	59,9

Fuente: Propia

En la Figura 28 se visualiza una comparación de los resultados mínimos obtenidos en los dos ambientes y en la Figura 29 una comparación de los resultados máximos.

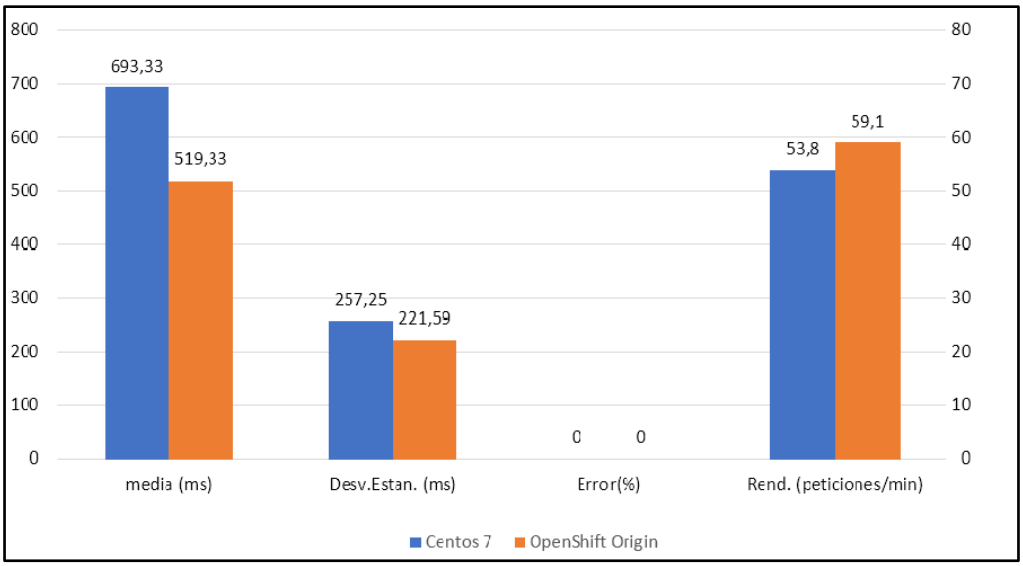

Figura 28 Resumen resultados mínimos

Fuente: Propia

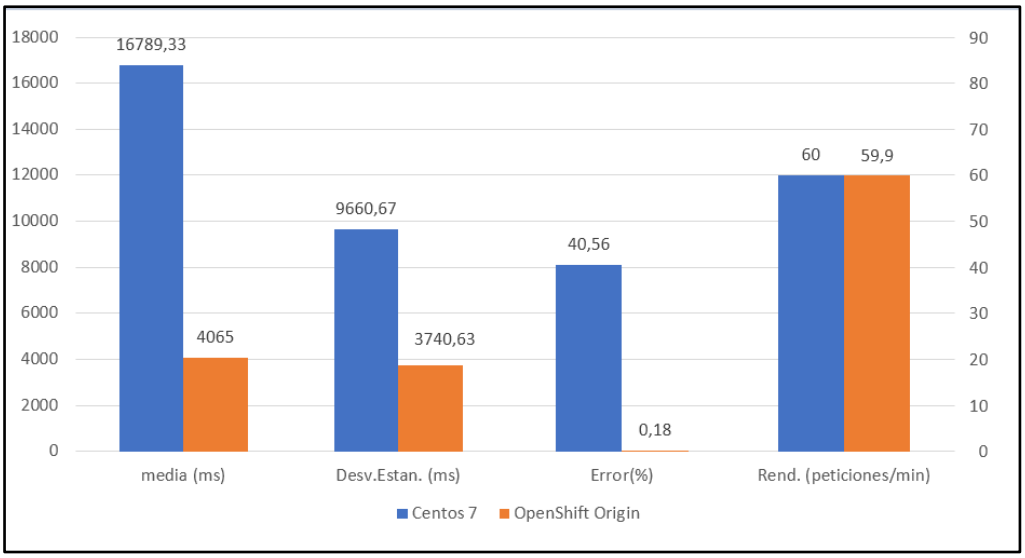

Figura 29 Resumen resultados máximos

Fuente: Propia

El rendimiento demostrado por los dos ambientes fue bueno sin embargo, en pequeña cantidad OpenShift obtuvo mejores resultados, teniendo en promedio de la prueba un rendimiento de 59.62 peticiones procesadas por minuto por otro lado CentOS tuvo un rendimiento promedio de 56.30 siendo este más bajo que en OpenShift, en los parámetros de media y desviación los valores promedio obtenidos son notablemente más bajos en OpenShift, sobresaliendo en la media con 1953,06 contra 7782,06 ms obtenidos en CentOS 7 y en la desviación estándar con 1320,02 ms contra un 7933,67 ms respectivamente.
En la Tabla 18 y Figura 30 se muestra una comparación de los valores promedio obtenidos en esta prueba en cada uno de los diferentes parámetros evaluados.

Tabla 18 Resumen de promedios de resultados

	CentOS 7	**OpenShift Origin**
	Promedio	**Promedio**
Media (ms)	7782,06	1953,06
Desviación Estándar (ms)	7933,67	1320,02
Error (%)	20,60	0,02
Rendimiento (Peticiones/min)	55,88	59,62

Fuente: Propia

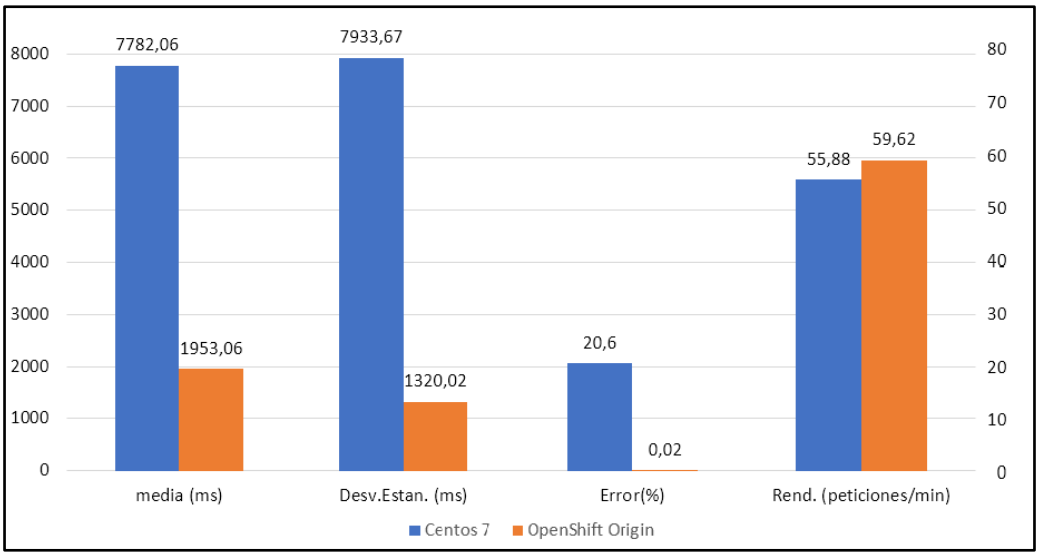

Figura 30 Resumen de promedio de resultados

Fuente: Propia

CONCLUSIONES

Después de concluir el presente estudio cumpliendo todos los objetivos anteriormente planteados se puede llegar a las siguientes conclusiones:

- La computación en la nube es un área muy extensa, que abarca muchos campos de conocimiento de la informática, cuyo objetivo principal es facilitar la adquisición de servicios informáticos por personas o empresas ya sea como software, plataforma o infraestructura.

- La principal ventaja que ofrece la computación en la nube es la disminución de costos en el consumo de servicios informáticos, ya que permite adquirirlos con los recursos exactos que se necesiten, cuando se requiera y sin tiempos establecidos, dejando de hacer uso del servicio cuando se desee.

- OpenShift es una excelente alternativa en lo que refiere a una plataforma como servicio, ya que integra tecnologías que han sido mejoradas durante años, esto permite que sea una plataforma como servicio estable y confiable.

- OpenShift Origin brinda la facilidad en el proceso de desarrollo de software de que el personal que realiza esta tarea consuma su tiempo solamente en ésta, dejando de lado las tareas conjuntamente realizadas con el desarrollo, OpenShift realiza y abstrae totalmente las configuraciones necesarias para el funcionamiento de la aplicación que sea desplegada.

- OpenShift gestiona de manera más eficiente los recursos del servidor brindando como resultado un mejor rendimiento las aplicaciones desplegadas en comparación con servidores tradicionales.

- Al mantener el código de una aplicación que se encuentra en OpenShift almacenado en un repositorio se puede manejar un control de versiones de este código, teniendo también un respaldo o copia de seguridad.

RECOMENDACIONES

Al concluir esta investigación se recomienda tener en cuenta los siguientes puntos:

- Se debe tener muy en cuenta los requisitos mínimos que son indicados en la documentación de instalación de OpenShift, ya que suele necesitar de amplias capacidades a nivel de infraestructura, dependiendo del escenario al que se lo vaya a exponer o de la disponibilidad que se necesite.

- Con la existencia de nubes de cómputo privadas y públicas, procurar mantener la información más confidencial para la empresa, organización o persona en un lugar resguardado o en una nube privada donde se le pueda dar el respectivo monitoreo y control de seguridad que necesite.

- La instalación y administración de OpenShift Origin requiere amplios conocimientos en el manejo de las tecnologías de Kubernetes y Docker además como conocimientos en la administración de Linux, por lo que se debe analizar previamente cuáles son las necesidades reales de la empresa y evaluar si los beneficios que brinda OpenShift son realmente necesarios.

- OpenShift cuenta con herramientas predeterminadas para el control de los proyectos como Prometheus para él envió de alertas por email o Grafana que permite recoger métricas básicas del sistema y los contenedores, las cuales se puede optar por instalar o no, por medio del archivo de inventario, se recomienda la instalación de ellas para sacar un mejor provecho de estas herramientas.

- Previamente a la instalación de OpenShift se debe configurar y confirmar correctamente los datos de instalación en el archivo de inventario, de lo contrario esto puede traer errores durante o después de la instalación.

REFERENCIAS

Alani, M. M. (2016). *Elements of Cloud Computing Security: A Survey of Key Practicalities*. Springer.

Bhowmik, S. (2017). *Cloud Computing*. Cambridge University Press.

Chopra, R. (2017). *Cloud Computing: An Introduction*. Stylus Publishing, LLC.

Documentation OKD. (2018, octubre 11). Recuperado el 4 de febrero de 2019, de https://docs.okd.io/3.11/install/prerequisites.html

Eric D. Schabell. (2016, febrero 11). Cómo instalar OpenShift como su PaaS privado | Desarrollador Planet JBoss. Recuperado el 6 de febrero de 2018, de http://planet.jboss.org/post/how_to_install_openshift_as_your_private_paas

Gallego, T., & Domingo Troncho, A. C. (2014, febrero 6). Gestión de Proyectos Informático Metodología SCRUM. Recuperado de http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf

Guna Permana, P. A., & Stikom Bali, S. (2015). Scrum Method Implementation in a Software Development Project Management (Vol. 6). Presentado en International Journal of Advanced Computer Science and Applications, Recuperado de https://thesai.org/Downloads/Volume6No9/Paper_27-Scrum_Method_Implementation_in_a_Software_Development_Project_Management.pdf

Huang, D., & Wu, H. (2017). *Mobile Cloud Computing: Foundations and Service Models*. Morgan Kaufmann.

Joyanes Aguilar, L. (2012). *Computación en la nube: estrategias de cloud computing en la empresa*. Mexico D.F: Alfaomega.

Khalid, A. (2010). Cloud Computing: Applying Issues in Small (p. 4). Presentado en 2010 International Conference on Signal Acquisition and Processing, Bahawalpur,

Pakistan: Department of Computer Science & IT. Recuperado de http://sci-hub.cc/http://ieeexplore.ieee.org/abstract/document/5432738/

Meyers L, Gamst G, Guarino A. (2016). *Applied multivariate research: design and interpretation* (3ra ed.). California: SAGE Publications.

Mishra, A. (2017). *Amazon Web Services for Mobile Developers: Building Apps with AWS*. John Wiley & Sons.

OKD: Renaming of OpenShift Origin with 3.10 release. (2018, agosto 3). Recuperado el 6 de febrero de 2019, de https://developers.redhat.com/blog/2018/08/03/okd-renaming-of-openshift-origin-with-3-10-release/

PMC Group. (2017, junio 6). Cloud Computing. Ventajas y desventajas. Recuperado el 5 de enero de 2018, de https://www.pmclatam.com/cloud-computing-ventajas-y-desventajas/

Pousty, S., & Miller, K. (2014). *Getting Started with OpenShift: A Guide for Impatient Beginners*. O'Reilly Media, Inc.

Red Hat, Inc. (2015a, febrero 1). Overview | Architecture | OpenShift Container Platform 3.6. Recuperado el 27 de noviembre de 2017, de https://docs.openshift.com/container-platform/3.6/architecture/index.html

Red Hat, Inc. (2015a, febrero 8). Managing Nodes | Cluster Administration | OpenShift Origin Latest. Recuperado el 2 de febrero de 2018, de https://docs.openshift.org/latest/admin_guide/manage_nodes.html

Red Hat, Inc. (2015b, febrero 8). Setting Up a Cluster | Getting Started | OpenShift Origin Latest. Recuperado el 7 de febrero de 2018, de https://docs.openshift.org/latest/getting_started/administrators.html

Red Hat, Inc. (2015b, junio 24). Características y ventajas: Red Hat OpenShift Container Platform. Recuperado el 27 de noviembre de 2017, de https://www.openshift.com/container-platform/features.html

Red Hat, Inc. (2017, mayo 3). Funciones - Red Hat OpenShift. Recuperado el 28 de noviembre de 2017, de https://www.openshift.com/features/index.html

Ruparelia, N. B. (2016). *Cloud Computing*. MIT Press.

Schwaber, K., & Sutherland, J. (2013, julio 1). The Definitive Guide to Scrum: The Rules of the Game. Recuperado de https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf

Timothy Prickett Morgan. (2012, noviembre 27). Red Hat takes OpenShift platform cloud private. Recuperado el 6 de febrero de 2018, de https://www.theregister.co.uk/2012/11/27/redhat_openshift_enterprise_paas/

V.K, P. (2015). *CLOUD COMPUTING*. PHI Learning Pvt. Ltd.

ANEXOS

Anexo A: Manual de Instalación de OpenShift Origin

Creación de Máquina virtual

Seleccionar Configuración avanzada y posteriormente agregar la imagen ISO de CentOS 7

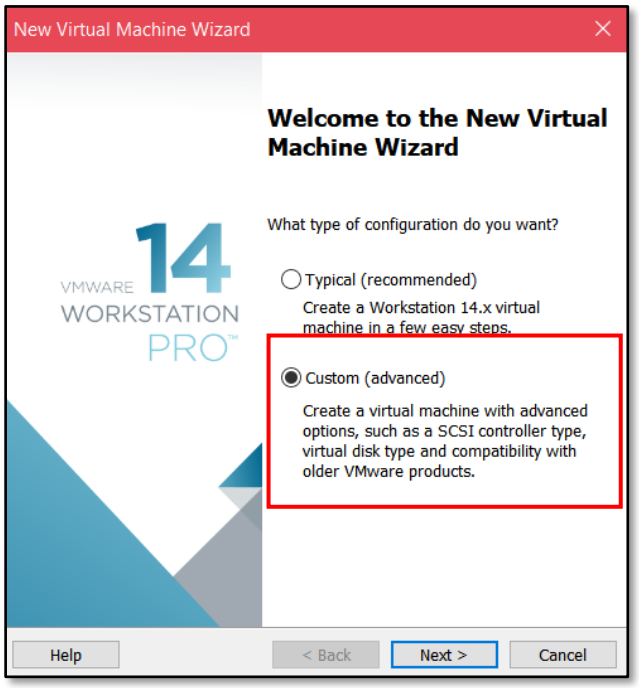

Figura A-1

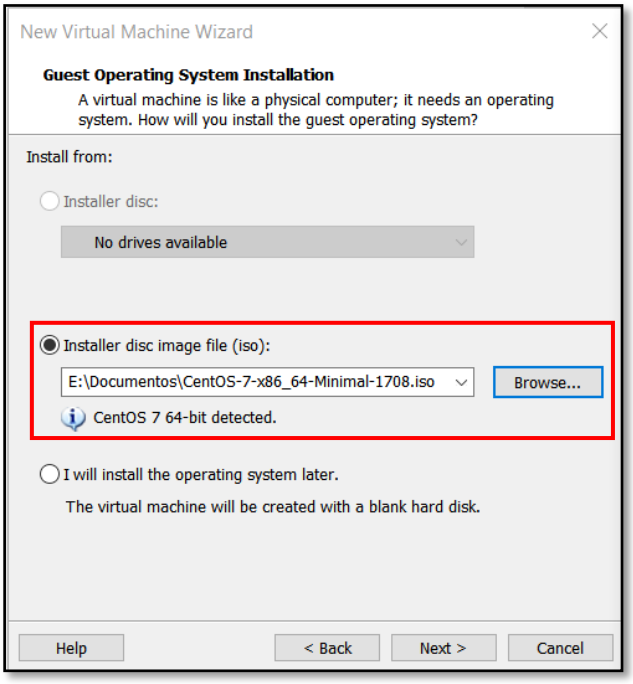

Figura A-2

Finalizar la creación de la máquina virtual

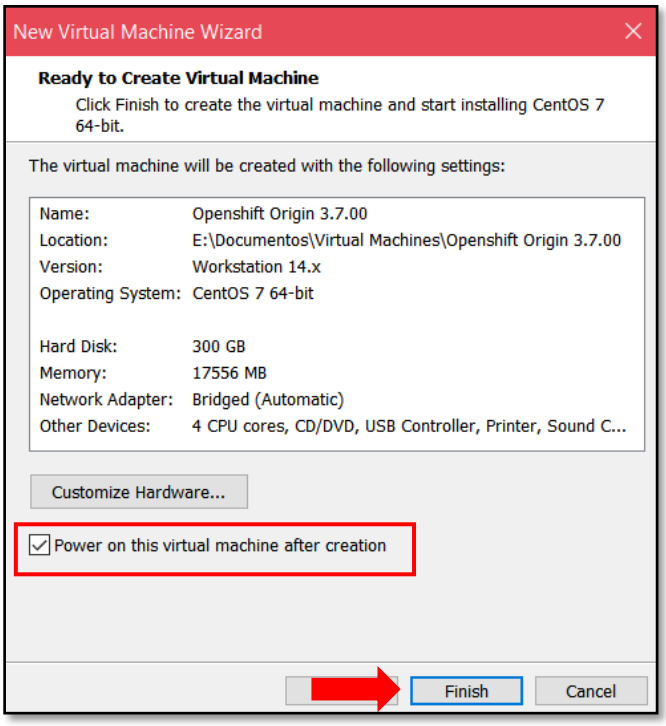

Figura A-3

Automáticamente se iniciará la máquina virtual creada, por lo que se procede a la instalación del sistema operativo. Se deberá seleccionar Install CentOS 7

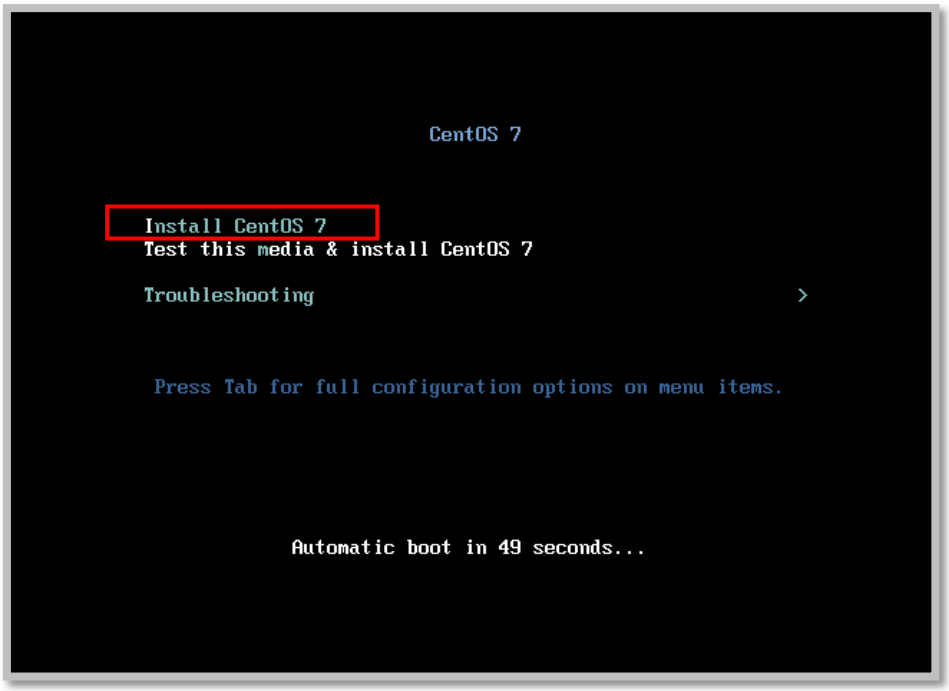

Figura A-4

Luego de la instalación, se debe reiniciar la máquina virtual.

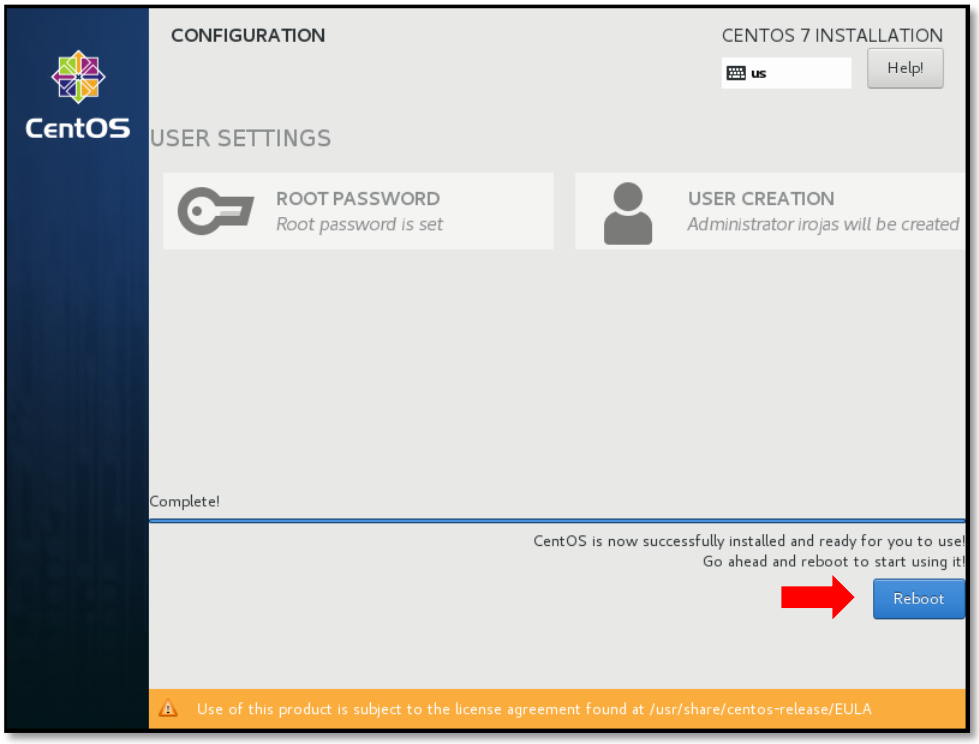

Figura A-5

A continuación, se procede a acceder a la máquina virtual con las credenciales configuradas anteriormente.

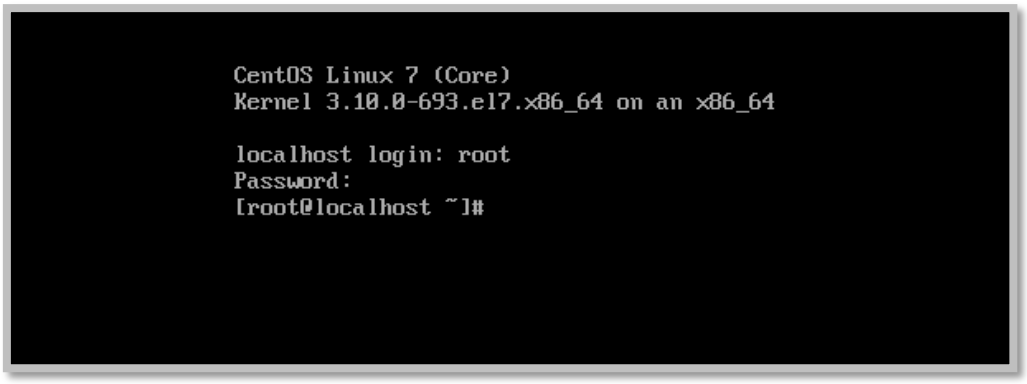

Figura A-6

Para un fácil manejo de la terminal, se recomienda acceder desde una herramienta externa como PuTTy. Aquí se ingresa la dirección IP configurada en la red y el puerto de acceso como SSH.

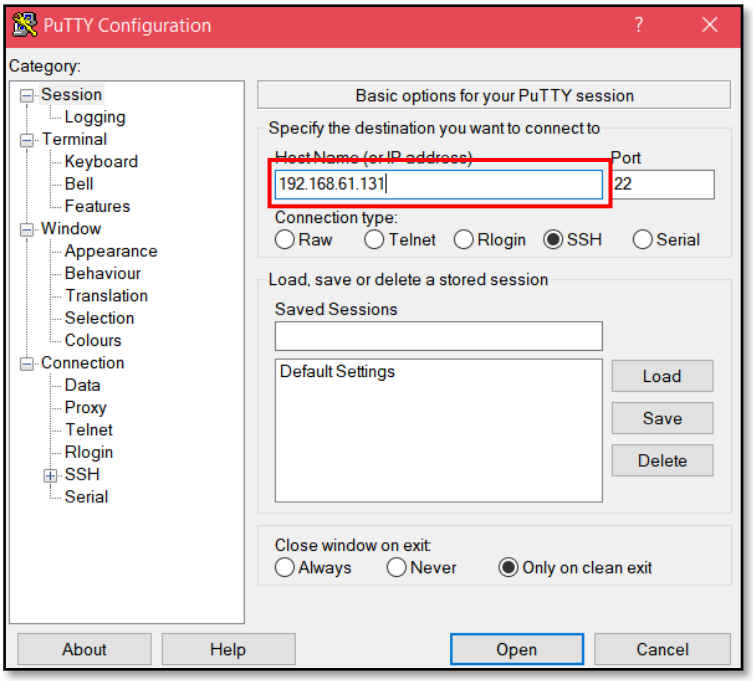

Figura A-7

Instalación de los paquetes base

Los paquetes y librerías que se deben instalar son los siguientes:

- yum install -y epel-release
- yum install -y git
- yum install -y wget
- yum install -y zile
- yum install -y nano
- yum install -y net-tools docker
- yum install -y python-cryptography
- yum install -y pyOpenSSL.x86_64
- yum install -y python2-pip
- yum install -y openssl-devel
- yum install -y python-devel
- yum install -y httpd-tools
- yum install -y NetworkManager
- yum install -y python-passlib
- yum install -y java-1.8.0-openjdk-headless

```
[root@localhost ~]#      yum install -y  wget git zile nano net-tools docker-1.13.1 \
>              bind-utils iptables-services \
>              bridge-utils bash-completion \
>              kexec-tools sos psacct openssl-devel \
>                       httpd-tools NetworkManager \
                      python-cryptography python2-pip python-devel  python-passlib \
                      java-1.8.0-openjdk-headless "@Development Tools"
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.uce.edu.ec
 * extras: mirror.uce.edu.ec
 * updates: mirror.uce.edu.ec
Package wget-1.14-18.el7.x86_64 already installed and latest version
Package git-1.8.3.1-20.el7.x86_64 already installed and latest version
No package      available.
Package nano-2.3.1-10.el7.x86_64 already installed and latest version
Package net-tools-2.0-0.24.20131004git.el7.x86_64 already installed and latest version
Package 32:bind-utils-9.9.4-73.el7_6.x86_64 already installed and latest version
Package iptables-services-1.4.21-28.el7.x86_64 already installed and latest version
Package bridge-utils-1.5-9.el7.x86_64 already installed and latest version
Package 1:bash-completion-2.1-6.el7.noarch already installed and latest version
```

Figura A-8

Clonar el repositorio de OpenShift

git clone http://github.com/openshift/openshift-ansible

```
root@localhost:~
[root@localhost ~]# [ ! -d openshift-ansible ] && git clone https://github.com/openshift/openshift-ansible.git
Cloning into 'openshift-ansible'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 135694 (delta 1), reused 9 (delta 0), pack-reused 135672
Receiving objects: 100% (135694/135694), 36.42 MiB | 62.00 KiB/s, done.
Resolving deltas: 100% (84896/84896), done.
[root@localhost ~]#
```

Figura A-9

Para agregar el dominio se ingresa al fichero /etc/hosts y se configura con el dominio:

```
root@localhost:~
[root@localhost ~]# cat <<EOD > /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.61.139 localhost  console console.dominio.prueba
EOD
[root@localhost ~]#
```

Figura A-10

Se habilita Docker

```
root@localhost:~
[root@localhost ~]# systemctl restart docker
[root@localhost ~]#     systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[root@localhost ~]#
```

Figura A-11

Ahora se genera un conjunto de claves empleando el algoritmo RSA. Para configurar el acceso seguro mediante SSH.

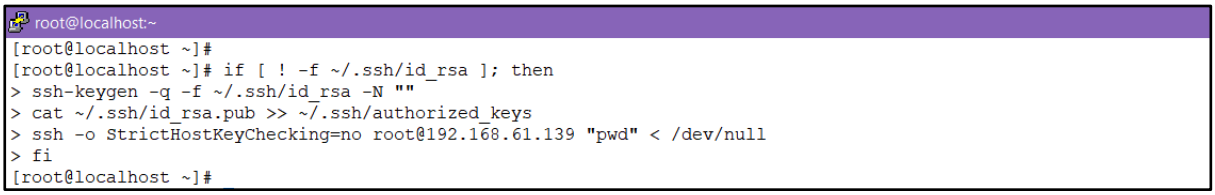

Figura A-12

Se configura el archivo de inventario

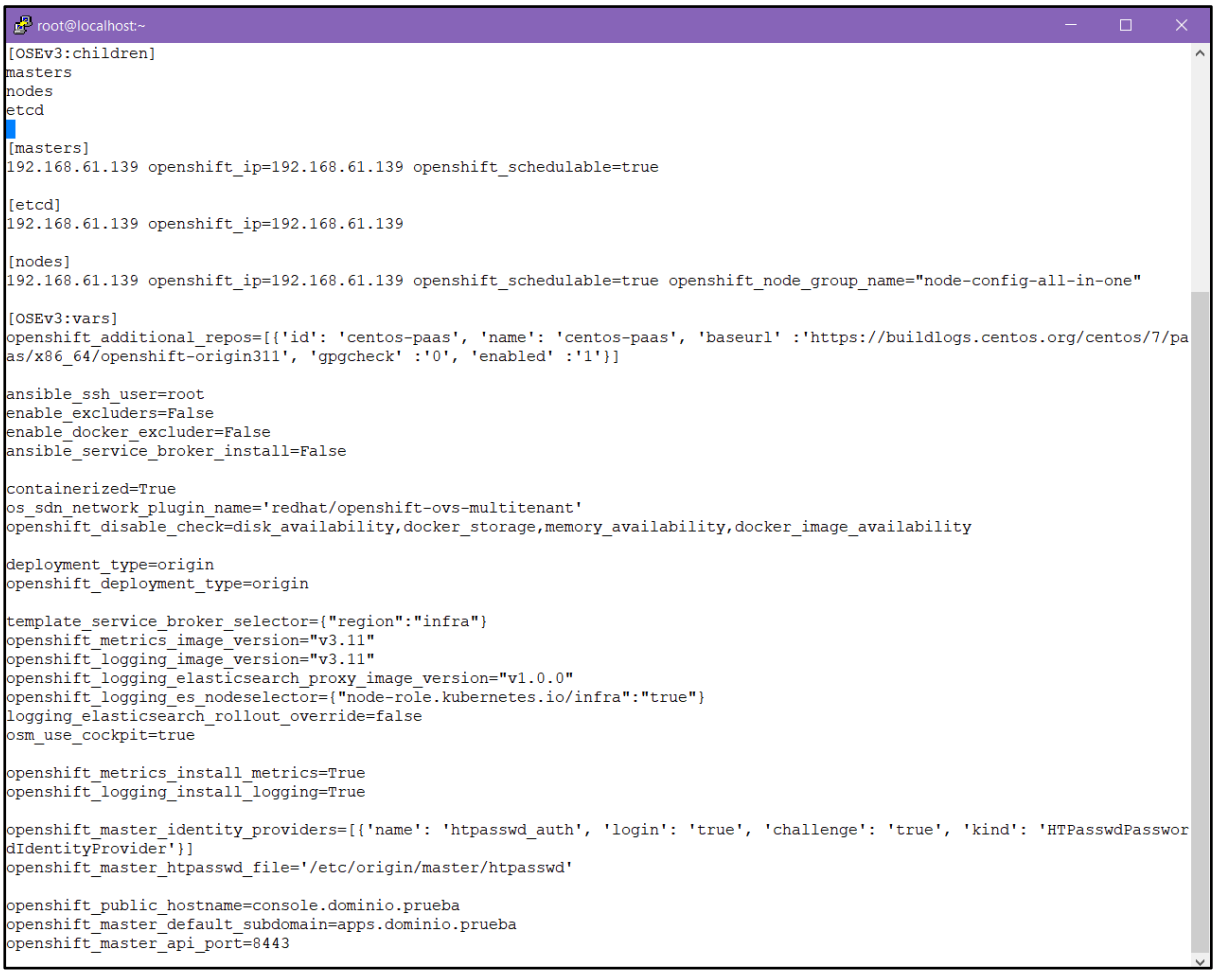

Figura A-14

Se inicia la instalación de OpenShift Origin

```
root@localhost:~
[root@localhost ~]# ansible-playbook -i inventory.ini openshift-ansible/playbooks/deploy_cluster.yml

PLAY [Initialization Checkpoint Start] ****************************************************************************************************************

TASK [Set install initialization 'In Progress'] *******************************************************************************************************
ok: [192.168.61.139]

PLAY [Populate config host groups] ********************************************************************************************************************

TASK [Load group name mapping variables] **************************************************************************************************************
ok: [localhost]

TASK [Evaluate groups - g_nfs_hosts is single host] ***************************************************************************************************
skipping: [localhost]

TASK [Evaluate oo_all_hosts] **************************************************************************************************************************
ok: [localhost] => (item=192.168.61.139)

TASK [Evaluate oo_masters] ****************************************************************************************************************************
ok: [localhost] => (item=192.168.61.139)
```

Figura A-15

Se termina la Instalación

```
PLAY [Auto-heal Install Checkpoint End] ***************************************************************************************************************

TASK [Auto-heal install 'Complete'] *******************************************************************************************************************
skipping: [192.168.61.148]

PLAY RECAP ********************************************************************************************************************************************
192.168.61.148             : ok=926  changed=285  unreachable=0    failed=0
localhost                  : ok=13   changed=0    unreachable=0    failed=0


INSTALLER STATUS **************************************************************************************************************************************
Initialization              : Complete (0:00:50)
        [DEPRECATION WARNING]: The following are deprecated variables and will be no longer be used in the next minor release. Please update your inventory a
cordingly:
        openshift_node_kubelet_args
Health Check                : Complete (0:00:47)
Node Bootstrap Preparation  : Complete (0:00:03)
etcd Install                : Complete (0:01:08)
Master Install              : Complete (0:05:50)
Master Additional Install   : Complete (0:01:19)
Node Join                   : Complete (0:00:18)
Hosted Install              : Complete (0:01:38)
Web Console Install         : Complete (0:00:59)
Metrics Install             : Complete (0:02:28)
Logging Install             : Complete (0:03:40)
Service Catalog Install     : Complete (0:01:55)
[root@localhost ~]#
```

Figura A-16

Para obtener una lista de todos los pods en el clúster, se lo hace con el comando:

```
[root@localhost ~]# oc get po --all-namespaces
NAMESPACE                           NAME                                      READY     STATUS    RESTARTS   AGE
default                             docker-registry-1-9q7k4                   1/1       Running   0          2h
default                             registry-console-2-lffv9                  1/1       Running   0          55m
default                             router-2-xd6rs                            1/1       Running   0          40m
kube-service-catalog                apiserver-57zlt                           1/1       Running   0          29m
kube-service-catalog                controller-manager-bnv76                  1/1       Running   1          29m
logging                             logging-curator-2-gf9jm                   1/1       Running   0          54m
logging                             logging-es-data-master-j2oqttct-3-fqjbj   2/2       Running   0          30m
logging                             logging-fluentd-fsx6r                     1/1       Running   0          2h
logging                             logging-kibana-2-6f6jb                    2/2       Running   0          54m
openshift-ansible-service-broker    asb-2-stqc2                               1/1       Running   0          2m
openshift-ansible-service-broker    asb-etcd-2-wj4rq                          1/1       Running   0          2m
openshift-infra                     hawkular-cassandra-1-tn476                1/1       Running   1          37m
openshift-infra                     hawkular-metrics-npdqk                    1/1       Running   2          36m
openshift-infra                     heapster-hl5zl                            1/1       Running   0          36m
openshift-template-service-broker   apiserver-mrtlt                           1/1       Running   0          24m
[root@localhost ~]#
```

Figura A-17

Si todos los contenedores están corriendo se puede acceder a la consola web

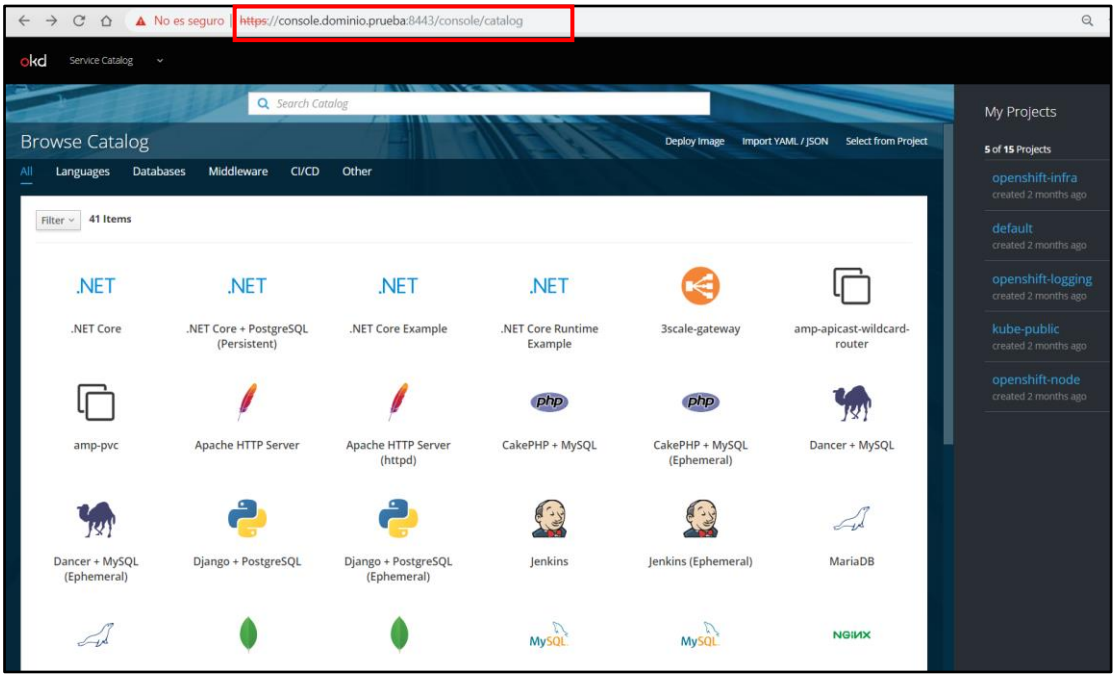

Figura A-18

Guía completa de Instalación OpenShift Origin adjunta en el CD

Anexo B: Manual de despliegue de una aplicación en OpenShift Origin

Se crea un repositorio con el código fuente de la aplicación

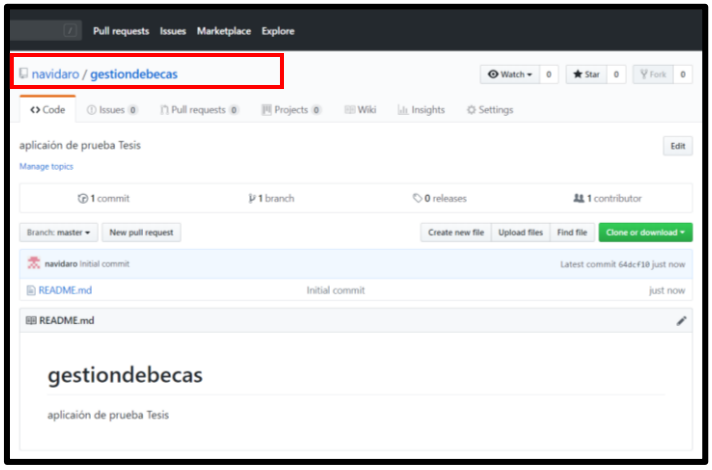

Figura B-1

Se agrega el código fuente con la herramienta Git.

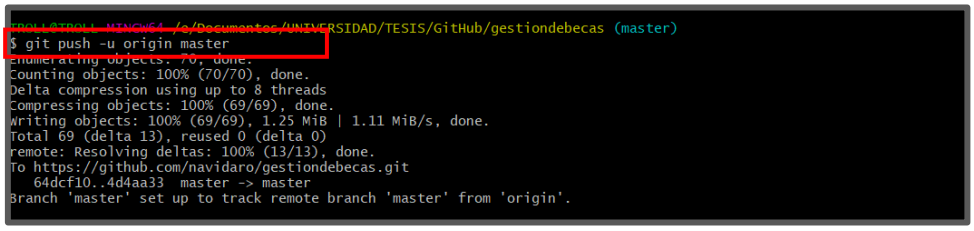

Figura B-2

Se crea un proyecto en OpenShift Origin

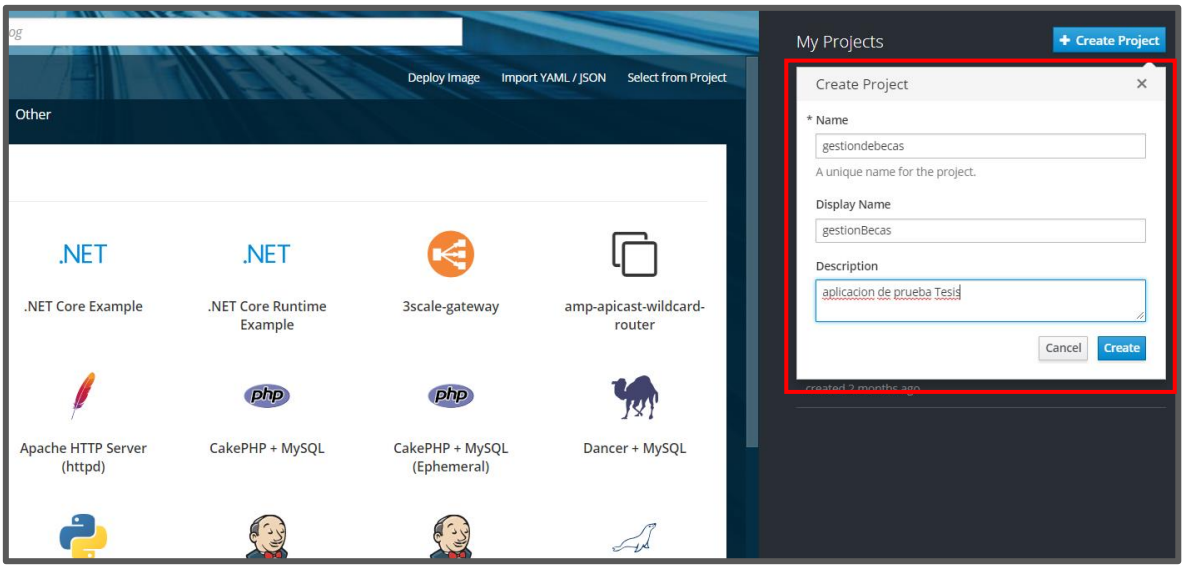

Figura B-3

Se crea un contenedor de PHP y se agrega el repositorio creado

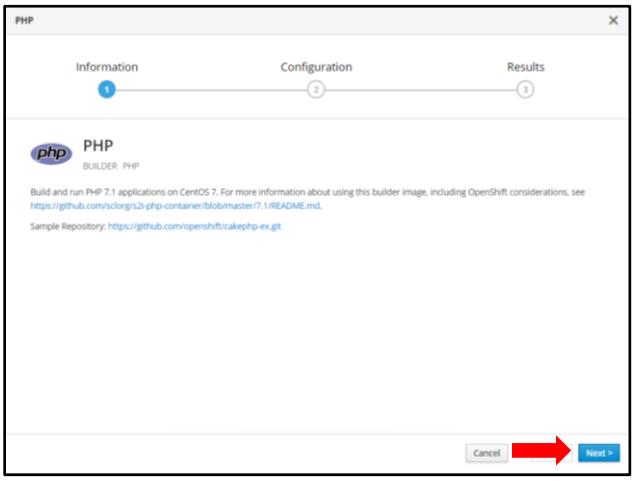

Figura B-4

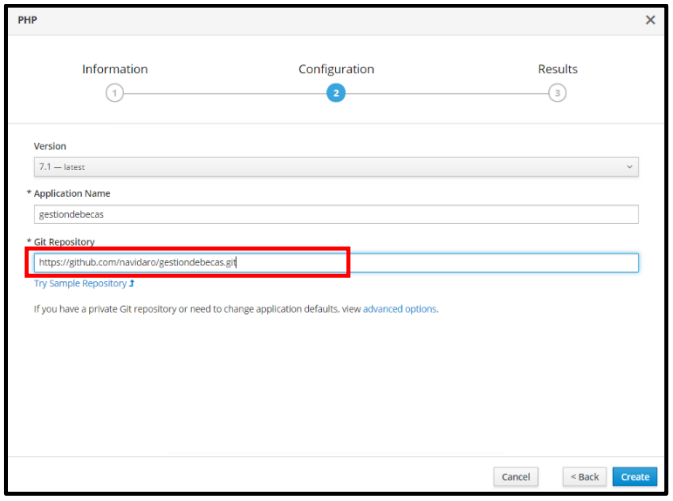

Figura B-5

Figura B-6

El contenedor empieza a descargar el código fuente del repositorio y se crea Correctamente

Figura B-7

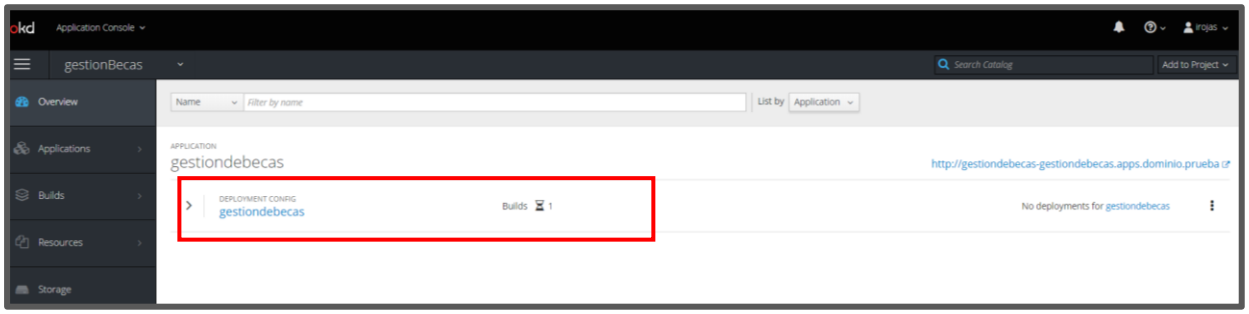

Figura B-8

El contenedor se crea correctamente

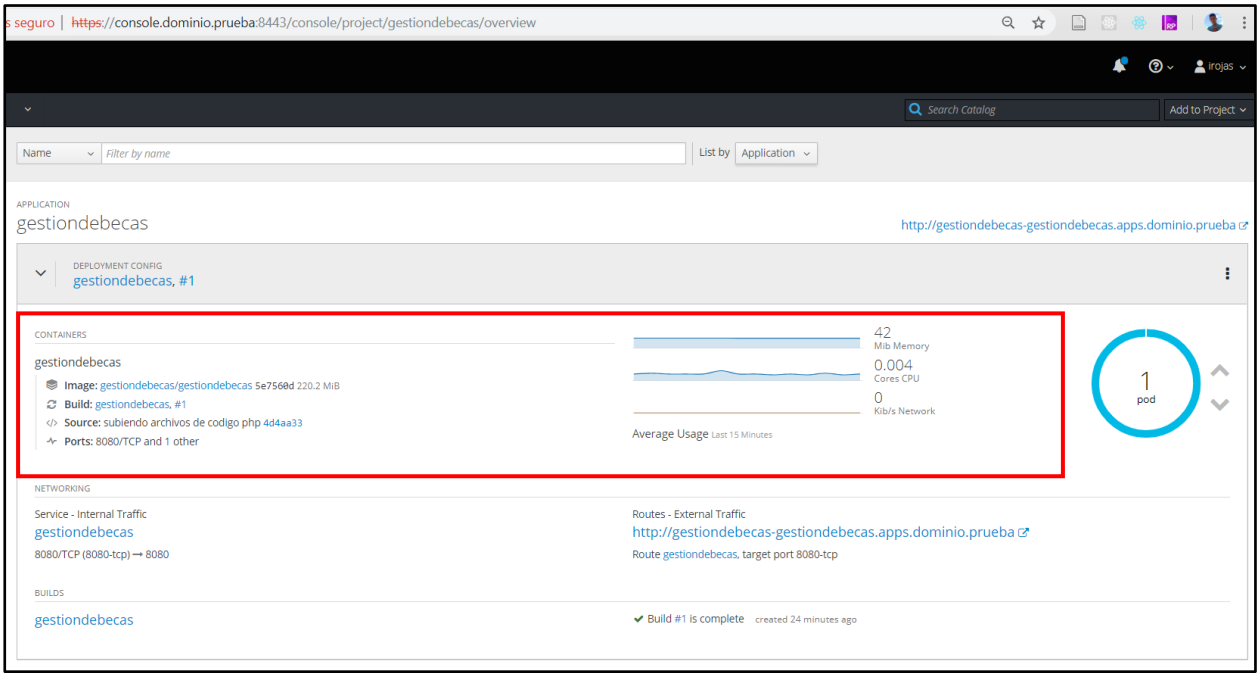

Figura B-9

Se comprueba que la aplicación está en línea para proceder a crear la base de datos

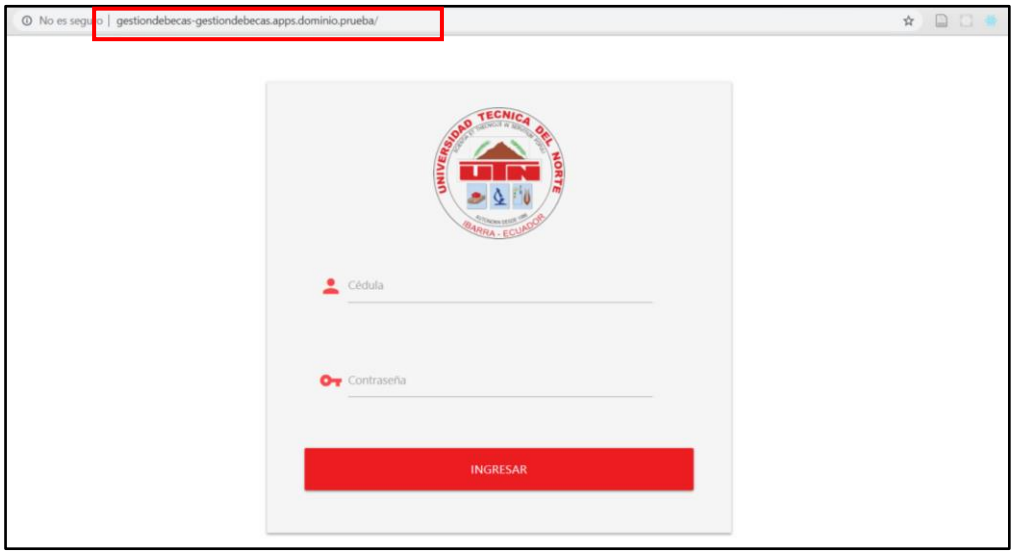

Figura B-10

Se procede a crear el contenedor de la base de datos

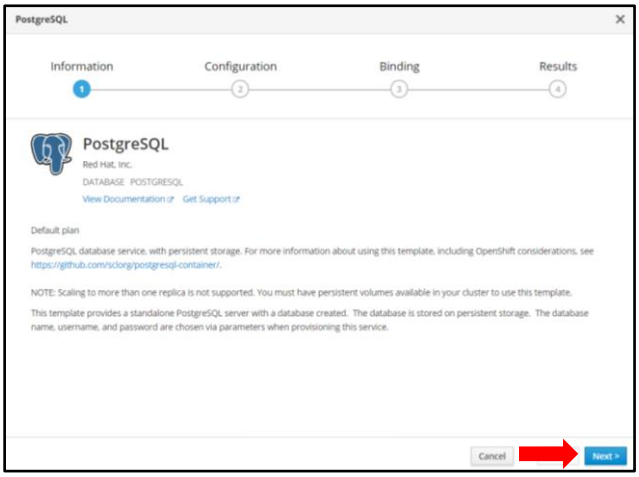

Figura B-11

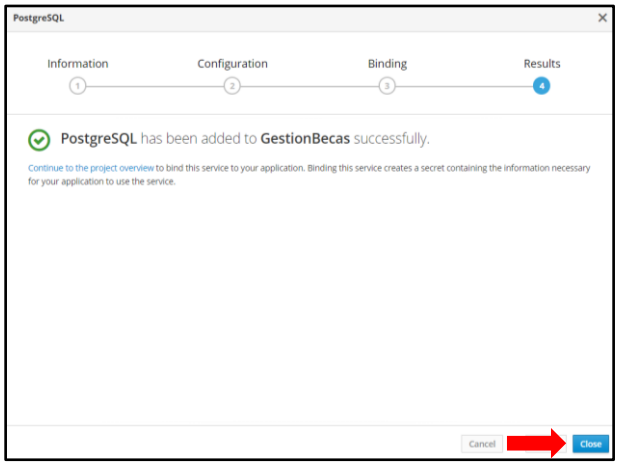

Figura B-12

El contenedor se crea correctamente

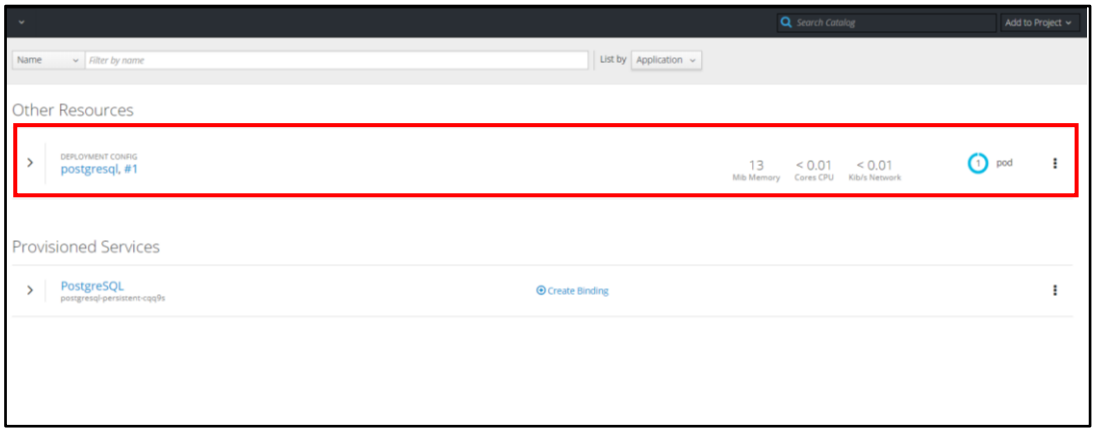

Figura B-13

Verificar la IP del servicio de base de datos

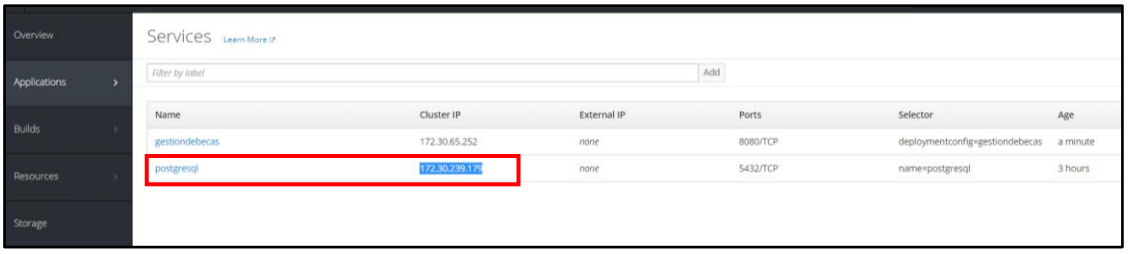

Figura B-14

Revisar las variables de conexión

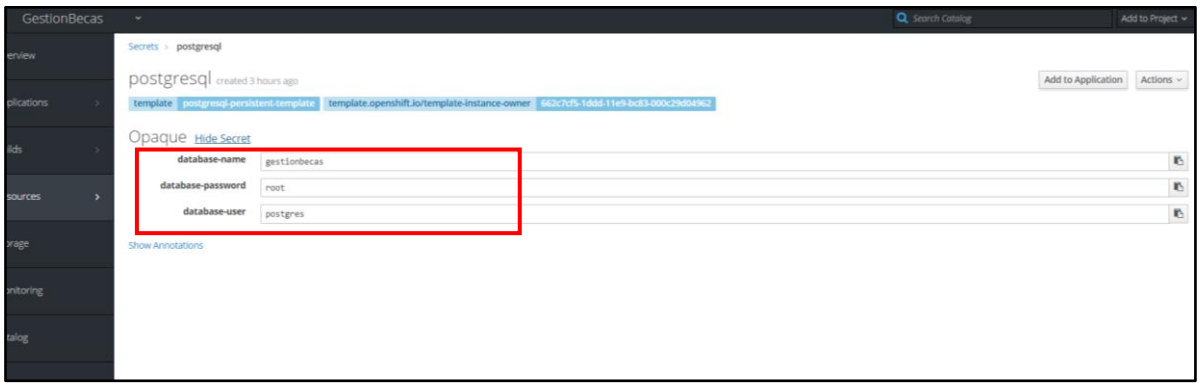

Figura B-15

Se modifica la conexión de la aplicación de acuerdo al contenedor creado

navidaro Update Database.php 180bfe3 3 hours ago

1 contributor

46 lines (41 sloc) 1.42 KB Raw Blame History

```
<?php

class Database {

//Propiedades estaticas con la informacion de la conexion (DSN):
    private static $dbName = 'gestionbecas';
    private static $dbHost = '172.30.239.179';
    private static $dbUsername = 'postgres';
    private static $dbUserPassword = 'root';
//Propiedad para control de la conexion:
    private static $conexion = null;
    /**
     * No se permite instanciar esta clase, se utilizan sus elementos
     * de tipo estatico.
     */
    public function __construct() {
        exit('No se permite instanciar esta clase.');
    }
```

Figura B-16

Se realiza una restauración de la base de datos

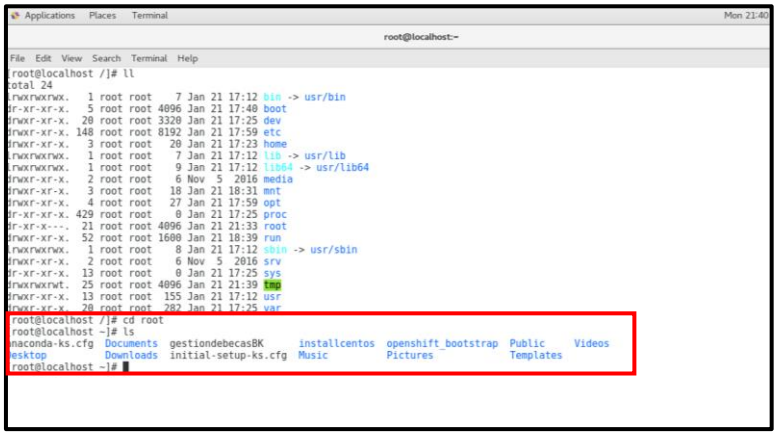

Figura B-17

Copiamos el archivo al contenedor

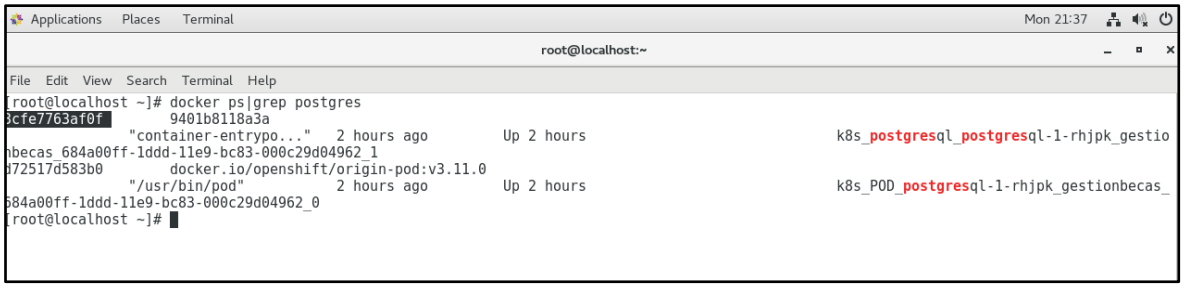

Figura B-18

```
[root@localhost /]# cd root
[root@localhost ~]# ls
anaconda-ks.cfg  Documents  gestiondebecasBK      installcentos  openshift_bootstrap  Public     Videos
Desktop          Downloads  initial-setup-ks.cfg  Music          Pictures             Templates
[root@localhost ~]# docker ps|grep postgres
3cfe7763af0f        9401b8118a3a
            "container-entrypo..."   2 hours ago         Up 2 hours                              k8s_postgresql_postgresql-1-rhjpk_gestio
nbecas_684a00ff-1ddd-11e9-bc83-000c29d04962_1
d72517d583b0        docker.io/openshift/origin-pod:v3.11.0
            "/usr/bin/pod"           2 hours ago         Up 2 hours                              k8s_POD_postgresql-1-rhjpk_gestionbecas_
684a00ff-1ddd-11e9-bc83-000c29d04962_0
[root@localhost ~]# docker cp gestiondebecasBK 3cfe7763af0f:/gestiondebecasBK
```

Figura B-19

Se realiza una consulta a la base de datos para saber que todo está correcto

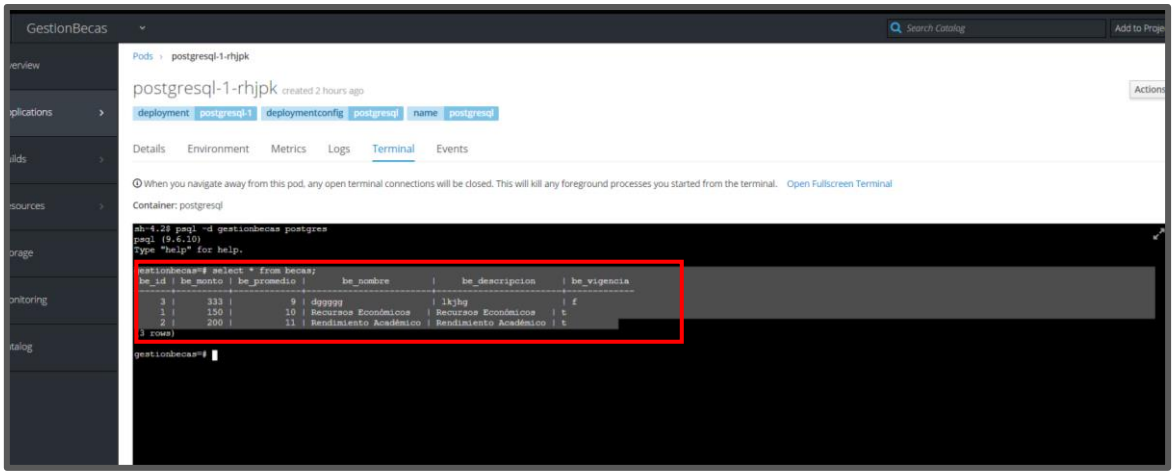

Figura B-20

Los contenedores están creados correctamente

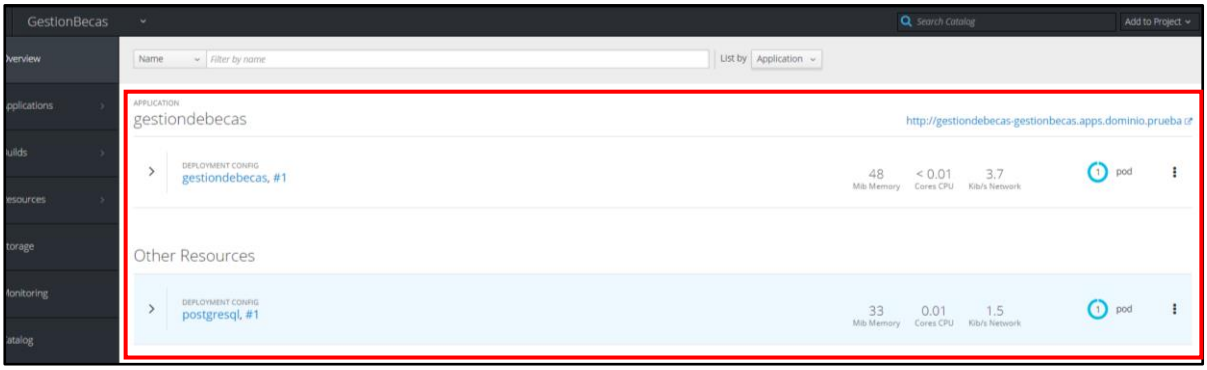

Figura B-21

Se puede acceder a los datos de la aplicación

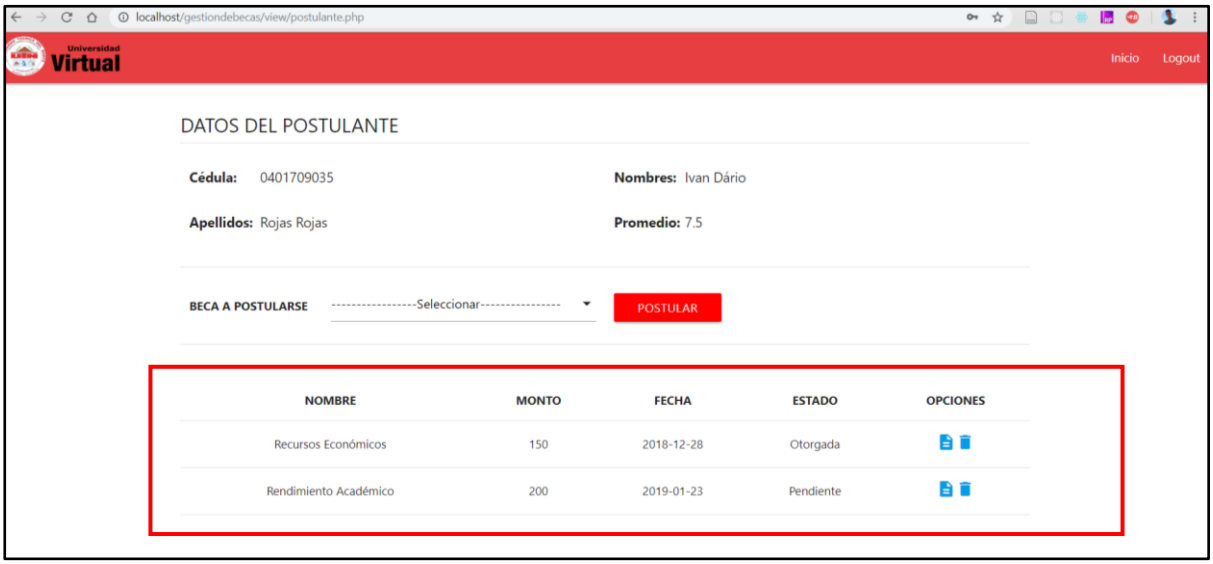

Figura B-22

Guía completa de Despliegue en OpenShift Origin adjunta en el CD