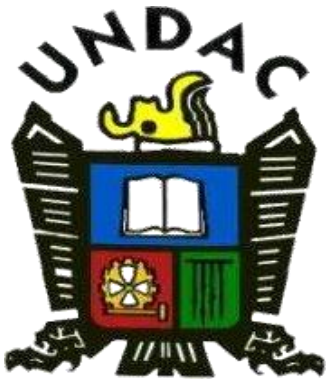
UNIVERSIDAD NACIONAL DANIEL ALCIDES CARRIÓN

FACULTAD DE INGENIERÍA

ESCUELA DE FORMACIÓN PROFESIONAL DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN



TESIS

Análisis de Docker utilizando Odoo, para optimizar la gestión en la librería del Virrey SAC - Lima

Para optar el título profesional de:

Ingeniero de Sistemas y Computación

Autor: Bach. Carlos Alberto LEON PAITAN

Asesor: Mg. Teodoro ALVARADO RIVERA

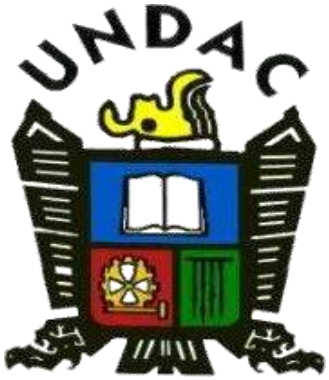
Cerro de Pasco – Perú – 2019

UNIVERSIDAD NACIONAL DANIEL ALCIDES CARRIÓN

FACULTAD DE INGENIERÍA

ESCUELA DE FORMACIÓN PROFESIONAL DE INGENIERÍA

DE SISTEMAS Y COMPUTACIÓN



TESIS

Análisis de Docker utilizando Odoo, para optimizar la gestión en la librería del Virrey SAC - Lima

Sustentada y aprobada ante los miembros del jurado:

Dr. Angel Claudio NUÑEZ MEZA	**Mg. Oscar Clevorio CAMPOS SALVATIERRA**
PRESIDENTE	**MIEMBRO**

Ing. Melquiades Arturo TRINIDAD MALPARTIDA
MIEMBRO

DEDICATORIA

A nuestra alma mater Universidad Nacional Daniel Alcides Carrión por permitirnos lograr el desarrollo de nuestra formación profesional.

A mis padres Abel y Marleny por apoyarme y alentarme a cumplir mis metas y ser un ejemplo de vida.

A Gina por su tenacidad y lucha insaciable que han hecho de ello un gran ejemplo.

RECONOCIMIENTO

Deseo agradecer a Dios, asimismo expresar mi gratitud y sincero agradecimiento a las instituciones y personas que hicieron posible el logro del objetivo propuesto.

- A la Universidad Nacional Daniel Alcides Carrión – Facultad de Ingeniería. Escuela de Formación Profesional de Ingeniería de Sistemas y Computación
- A la Librería El Virrey SAC, por el apoyo unánime en el desarrollo, la aplicación de los instrumentos y la culminación de la presente investigación.
- Al gerente, empleados y colaboradores de la Librería El Virrey SAC por su apoyo desinteresado.
- A mis familiares, amigos y demás personas, que me apoyaron directa e indirectamente.

EL AUTOR.

RESUMEN

Me permito poner a vuestra consideración el presente trabajo que se realizó en la librería El Virrey SAC, que se encuentra ubicado en la ciudad de Lima. En el cual se tuvo como objetivo Determinar si el análisis Docker utilizando Odoo, optimizará la gestión de la Librería del Virrey SAC. Planteamos la hipótesis general: El análisis de Docker utilizando Odoo, optimiza la gestión de la Librería del Virrey SAC, donde los resultados obtenidos fueron significativos y relevantes.

Se tuvo como población a los colaboradores de la Librería del Virrey SAC específicamente a los colaboradores del área de sistemas y al Gerente General, que se encuentra en la ciudad de Lima. Es un estudio de tipo explicativo pues se recolectaron datos o componentes sobre los procesos de funcionamiento del sistema de la Librería El Virrey SAC, se realizó un análisis que permitió conocer los eventos del Docker al utilizar Odoo en la Librería del Virrey SAC. Así mismo, el estudio es de tipo Aplicada, que permitió analizar los resultados de las encuestas que se aplicaron para la recolección de datos de la organización a estudiar. La Investigación es Cuasi Experimental, porque los sujetos o grupos de sujetos de estudio no están asignados aleatoriamente. Se aplicó con el cuestionario de pre test y el cuestionario del post test, en dos momentos con un solo grupo.

Se utilizó el método de análisis funcional donde se analizó la función que cumple Docker para su funcionamiento como un todo y describe como se debe utilizar asimismo también se utilizó el método análisis técnico para describir Docker y

sus propiedades y se explicó el proceso de creación de contenedores para la consolidación de cada uno de los conceptos y proposiciones planteadas en el presente trabajo. Llegando a comprender que Docker con sus beneficios optimiza la Gestión de la Librería del Virrey SAC y considero que también en otras empresas.

Palabras clave: Análisis de Docker; contenedores.

ABSTRACT

I permit getting to your consideration present work that was done in The Virrey SAC bookstore, that it finds itself located in town of Lima. In where had for aimed determinate if the analysis Docker utilizing Odoo will optimize the management of the Virrey SAC'S bookstore. We present general hypothesis: Docker's analysis utilizing Odoo, optimize the management of the Virrey SAC'S bookstore, where the aftermaths founded were significant and relevant.

The collaborators of the Virrey SAC'S bookstore Were had as population specifically to the collaborators of the area of systems and to the General manager, that he meets in town of Lima. A study Belongs to explicative fellow because The Virrey SAC recollected data or components on the functioning processes of the bookstore's system, it was accomplished an analysis that he permitted knowing the Docker's events to the utilizing Odoo in the Virrey SAC'S bookstore. Likewise, study belongs to fellow it is type applied, that permitted to examine the aftermaths of opinion polls that were applicable in order to the anthology of data of the organization to study. Investigation is Quasi Experimental, because the subjects or subjects groups of study are not assigned at random. In this work was applied with pre and post questionnaire, in two moments with an alone group.

The functional- analysis method where analyzed the show that Docker in order to his functioning does his job as a whole and it describes how as It must be utilized and also technical analysis to describe how was utilized method Docker and their properties and it understood the creation process of containers in order to the consolidation out of every one of the concepts and advances presented in the present work. Coming to understand than Docker with their gainings it

optimizes the Steps of the Virrey SAC'S bookstore and I consider that also in another companies.

Keywords: Docker's Analysis; Containers

INTRODUCCIÓN

Es honroso poner a vuestra consideración la tesis titulada “ANÁLISIS DE DOCKER UTILIZANDO ODOO, PARA OPTIMIZAR LA GESTIÓN EN LA LIBRERIA DEL VIRREY SAC - LIMA” que fue realizado con la finalidad de determinar si el análisis Docker utilizando Odoo, optimizará la gestión de la Librería del Virrey SAC.

El propósito de esta investigación, fue analizar la función que cumple Docker para la gestión como un todo y la descripción como se debe utilizar, asimismo también se utilizó el análisis técnico para describir Docker y sus propiedades y se explicó el proceso de creación de contenedores. Adoptar la tecnología Docker en las instituciones y empresas en cuando al alojamiento de informaciones a través de sus aplicaciones y servicios.

En esa perspectiva, el presente trabajo ha sido estructurado en cuatro capítulos: El capítulo I, dedicado al Problema de investigación, donde hacemos referencia a cómo se tiene los beneficios que nos ofrece la tecnología Docker, tenemos la formulación del problema; en el que detallamos los problemas, objetivos, seguido de la justificación desde diversos puntos de vista, importancia y limitaciones de la investigación.

El capítulo II, se presenta el marco teórico, en donde encontramos los antecedentes, bases teórico – científicos con la exposición teórica de los diversos autores y nuestros análisis y comentarios referentes a la tecnología Docker, de manera clara y detallada, según el planteamiento científico, la definición de términos, hipótesis y variables.

El capítulo III, comprende la metodología y técnicas de investigación; donde está el tipo y diseño de investigación utilizada, población y muestra, método de investigación, técnicas e instrumentos de recolección de datos, y los métodos de análisis de datos.

El capítulo IV, está destinada a los resultados y discusión; la información de la empresa, desde la reseña histórica, visión y misión, valores, objetivos, organigramas, localización. Seguido de la presentación de resultados, con la descripción del trabajo de campo, diseño de la organización y procesamiento de los datos. Aquí también presentamos los resultados de la aplicación del pre test y post test, continuando con la prueba de hipótesis sobre la investigación. Y para concluir encontramos las conclusiones y recomendaciones.

Esperando que el presente trabajo sea un aporte al vasto estudio de las tecnologías en la vida de la persona y su entorno.

EL AUTOR.

INDICE

Páginas

CAPÍTULO III

METODOLOGÍA Y TÉCNICAS DE INVESTIGACIÓN

CAPITULO IV

RESULTADOS Y DISCUSIÓN

CAPITULO I

PROBLEMA DE INVESTIGACIÓN

1.1 Identificación y determinación del problema:

En la actualidad las PyMES abarcan una gran cantidad de organizaciones en el Perú, las cuales la mayoría hace uso de tecnologías que faciliten su crecimiento, pero desconocen algunas herramientas que faciliten el uso de dichas tecnologías, muchas PyMES usan plataformas Cloud para alojar páginas web, aplicaciones y servicios web, pero como puede optimizar el uso de recursos tecnológicos Cloud, es así que en este punto se define el término Docker un término que vienen generando cambios paradigmáticos, esta herramienta permite aislar aplicaciones web y servicios e incluso facilita la migración de una infraestructura física a una infraestructura Cloud.

El uso de la tecnología Docker puede ser un factor transformador de sus estructuras y funciones, un instrumento para mejorar su cobertura, calidad, pertinencia y equidad de acceso como una manera de construir una nueva

identidad en las Organizaciones. Actualmente el uso de Docker está asociada a conceptos como optimización, seguridad, escalabilidad y facilidad de la gestión.

Los avances en procesadores han llevado a una situación en la que sólo se aprovecha el 20-30% de su potencia, Docker nos permitir ejecutar varios servicios o aplicaciones en un mismo entorno y aprovechar los recursos del hardware directamente ello se lleva a cabo debido a que a diferencia de la virtualización Docker hace uso de la infraestructura física y a la vez aísla las diversas aplicaciones o servicios según sean sus requerimientos es donde se convierte en una alternativa innovadora.

1.2 Delimitación de la investigación

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software proporcionando una capa adicional de abstracción y automatización de virtualización a nivel de sistemas operativos en Linux. Este trabajo estuvo orientado a generar conocimiento práctico que sea válido para la especialidad de Ingeniería de Sistemas y Computación específicamente de administración de servidores. El alcance que tiene este trabajo fue determinar el análisis de Docker para la optimización de la gestión de la Librería del Virrey SAC.

1.3 Formulación del problema

1.3.1 Problema principal:

¿De qué manera el análisis de Docker utilizando Odoo, optimizará la gestión de la Librería del Virrey SAC?

1.3.2 Problemas específicos:

a) ¿Cómo la implementación del Docker-compose para la escalabilidad de Odoo, optimizará en la gestión de la Librería del Virrey SAC?

b) ¿De qué manera Docker-compose contribuirá en los despliegues de Odoo, en la gestión de la Librería del Virrey SAC??

c) ¿Cómo la ejecución de la migración de Odoo de un servidor físico en la nube optimizará la gestión de la Librería del Virrey SAC?

1.4 Formulación de objetivos

1.4.1 Objetivo General

Determinar si el análisis Docker utilizando Odoo, optimizará la gestión de la Librería del Virrey SAC.

1.4.2 Objetivos específicos

a) Verificar si la Implementación Docker-compose para la escalabilidad de Odoo optimizará la gestión en la Librería del Virrey SAC.

b) Determinar si Docker-compose contribuirá en los despliegues de Odoo, en la gestión de la Librería del Virrey SAC.

c) Verificar si la ejecución de la migración de Odoo optimizará la gestión de la Librería del Virrey SAC

1.5 Justificación de la investigación

A medida que va aumentando la tecnología se han implementado una serie de herramientas y servicios nuevos, y entre ellas actualmente es Docker, tecnología que está siendo adoptada por muchas empresas al momento de alojar tanto como aplicaciones y servicios.

Muchos desarrolladores ven a Docker como un aliado la herramienta que facilita al desarrollador desplegar copias exactas de los servidores, muchas organizaciones están implementado Docker, y por eso en este estudio propusimos hacer uso de la herramienta Docker con la finalidad de analizar los beneficios de dicha tecnología utilizando Odoo y permitió sugerir a los desarrolladores de Odoo optar por el uso de esta tecnología a nuestro alrededor.

Docker es una herramienta que puede empaquetar una aplicación y sus dependencias en un contenedor virtual que se puede ejecutar en cualquier servidor. Esto ayuda a permitir la flexibilidad y portabilidad en donde la aplicación se puede ejecutar, ya sea en las instalaciones físicas, la nube pública, nube privada, etc.

1.6 Limitaciones de la investigación

Este proceso de analizar Docker utilizando Odoo, siendo una tecnología nueva, usualmente, tiene funciones que automatizan procesos, teniendo la opción de especificar ciertas limitaciones.

Algunas de estas limitaciones pueden ser:

1. Desgraciadamente, Docker no es capaz de sustituir a cualquier tipo de software.

2. No tendría sentido usar Docker, si el aplicativo que vamos a usar es capaz de escalar horizontalmente de manera nativa.
3. Docker se requiere Kernel 3.8 mínimo.

CAPITULO II

MARCO TEÓRICO

2.1 Antecedentes de estudio

A nivel local

De acuerdo a la naturaleza de la investigación se han revisado la biblioteca universitaria, bibliotecas públicas y especializadas a nivel local y no se han encontrado trabajos de investigación con tecnología Docker o similares por lo que no se mencionan ninguna en este nivel.

A nivel nacional

Tesis intitulada: "Implementación de una Infraestructura Tecnológica Virtual con alta disponibilidad basada en Clústers para los servidores de la Universidad Señor de Sipán-Lambayeque.", desarrollado por César CARRILLO GUEVARA de la Universidad Nacional "Pedro Ruiz Gallo" – Facultad de Ingeniería Civil, Sistemas y Arquitectura, Escuela Profesional de Ingeniería de Sistemas, para optar el título de Ingeniero de Sistemas – Lambayeque, 2016.

Resumen:

Este trabajo muestra una solución tecnológica de alta disponibilidad para servidores basado en clúster de software propietario, bastante madura como para ser implementada a nivel empresarial. Propuesta que nace de la necesidad de minimizar el tiempo muerto de los servicios ante la caída de los servidores, de la Universidad Señor de Sipán, en los cuales están almacenados. Se analiza el estado actual de la infraestructura de red a nivel físico y lógico, así como los equipos servidores y los servicios que brinda la Universidad Señor de Sipán y a partir de ello se realiza un estudio detallado de las soluciones tecnológicas a nivel de software existentes en el mercado, analizando y comparando las características que estos poseen, dando énfasis a: networking, clustering y high Availability.

Se cuantifican las características de los diferentes SO para servidores y se elige el sistema operativo que garantice alta disponibilidad y que funcionalmente minimice los tiempos de caídas no programados en ellos. De esta manera se diseña un clúster de servidores en base a Windows Server 2016 que garantiza la disponibilidad de los servicios críticos como lo son el servicio WEB y el servicio de base de datos las 24x7. Se implementa un clúster de alta disponibilidad en las instalaciones de la Universidad Señor de Sipán siguiendo los lineamientos del diseño propuesto. Finalmente, se analizó el estado actual de los servidores y el estado posterior a la implementación mediante encuestas realizadas al Área de Integración de Tecnologías perteneciente a la Dirección de Tecnologías de la Información, el 80% de ellos aprueban como “Muy buena” al nivel de disponibilidad de la solución propuesta y comparándolo con el Estándar ANSI/TIA-942 la posicionan en un “TIER I”, garantizando

de esta manera alta disponibilidad y una mejora en la disponibilidad de los servicios brindados.

Tesis intitulada: "Sistema de gestión documental para la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano Puno – 2017.", desarrollado por Yésica Magaly RAMÍREZ ESTRELLA y Joseph Josafat RAMOS ARPASI de la Universidad Nacional Del Altiplano – Puno, Facultad de Ingeniería Estadística e Informática, Escuela Profesional de Ingeniería Estadística e Informática, Para Optar el Título Profesional de Ingeniero Estadístico e Informático – Puno, 2017.

Resumen:

Este trabajo se desarrolló en la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano - Puno. El objetivo principal fue evaluar la gestión documental de la oficina a través de un sistema, que les permitió mejorar las deficiencias en su gestión de documentos a través de la automatización de procesos y actividades como: el acceso y búsqueda de convenios; la publicación y difusión de convocatorias, programas de movilidad, becas y otros; que se realizan en dicha oficina. Para el desarrollo del sistema se consideró un modelo que combina la metodología ágil Extreme Programming, el framework de administración de proyectos Scrum y los principios de la filosofía Lean Software Development. Además, para alcanzar el objetivo principal de la investigación, se analizó la satisfacción de los usuarios respecto a la facilidad del sistema y la utilidad del sistema por el personal de la oficina desde el punto de vista organizacional y funcional. Los resultados de la evaluación determinaron que el nivel de satisfacción de los usuarios fue

alto, alcanzando una media de 28.1 puntos, y de igual manera el nivel de utilidad del sistema también fue alto, obteniendo una media de 28 puntos, en una escala de 7 a 35 puntos para ambos casos. En conclusión, se confirmó que la implantación del sistema mejoró favorablemente la gestión documental de la Oficina de Cooperación Nacional e Internacional.

A nivel internacional

Tesis intitulada: Estudio del contenedor cloud Docker y propuesta de implementación para la plataforma Cloud Fica, desarrollado por Milton Andrés SIMBAÑA ALARCÓN de la Universidad Técnica del Norte - Facultad De Ingeniería Ciencias Aplicadas Carrera de Ingeniería de Sistemas Computacionales, para optar el título de Ingeniero en Sistemas Computacionales - Ecuador, 2016" Resumen:

Este trabajo se realizó utilizando conceptos basados en la nube y de programación en el cual se efectuó un estudio de contenedores Cloud con el fin de administrar aplicaciones y crear infraestructuras virtuales aisladas en un servidor. Docker es un proyecto de código abierto con el que se creó contenedores que se denominan máquinas virtuales ligeras que serán menos exigentes en cuanto a recursos de hardware y software, una de las características de estos contenedores es brindar portabilidad, ligereza y autosuficiencia a las aplicaciones que se desplieguen utilizando contenedores Cloud. La característica principal de Docker es la de brindar la opción de crear infraestructuras virtuales, esto depende de las aplicaciones que vayamos a desplegar, conjuntamente con esta herramienta se usó a Bitnami como un complemento de Docker el cual ofrece soporte a Docker, Bitnami facilita crear Dockerfiles que contienen

instalaciones independientes de herramientas necesarias para el alojamiento de aplicaciones, de este modo también se logró realizar el aislamiento de aplicaciones sin dependencias del sistema anfitrión.

También se realizó una aplicación de prueba en Java server faces (JSF) que conjuntamente con la base de datos PostgreSQL demostró que la herramienta Docker funciona y además brinda un despliegue de aplicaciones rápida, óptima y segura a comparación de una instalación virtual tradicional.

Al finalizar dicho estudio se realizó una propuesta a la plataforma Cloud FICA con el objetivo de implementar esta tecnología y aprovechar los beneficios que brinda la herramienta Docker en un servidor real.

Tesis intitulada: Teoría y Aplicación de la Informática 2 Creación, despliegue y ejecución de Aplicaciones mediante contenedores en Docker de la Universidad Católica "Nuestra señora de la Asunción" Facultad de Ciencias y Tecnologías; para optar el Título de Ingeniero Informático, Paraguay, octubre de 2016.

Resumen:

Desarrollar aplicaciones actualmente en definitiva conlleva una gran cantidad de configuraciones previas de todo tipo, en la práctica, en la fase inicial del desarrollo de un software, inicializar un entorno de trabajo no es una tarea trivial.

En la inicialización se establecen varias características que componen a la base del proyecto, como el/los lenguajes/s de programación, las librerías a

usar en los mismos, tipos base de datos, frameworks, etc. Es sabido que en varias ocasiones estos pueden presentar problemas de compatibilidad con el sistema operativo, o problemas con versiones diferentes instaladas, dependencias de librerías ya utilizadas, etc.

Estos problemas son causados por compartir un único entorno de trabajo para varios proyectos con diferentes necesidades básicas. La interdependencia de librerías en la actualidad, genera un ambiente caótico y estos aumentan con la cantidad de proyectos realizados.

Agilizar este proceso tedioso es posible mediante el encapsulamiento de contenedores que presenta la herramienta Docker. Estos son como pequeñas máquinas virtuales que contienen lo esencial para levantar un entorno de trabajo portable, ligero y autosuficiente.

Proyecto Final de Carrera: VDI low cost implementation with KVM and Linux Dockers elaborado por Eric COLLAZO VALLDURIOLA de la Universidad Politécnica de Cataluña Barcelona, España 2015.

Resumen:

Este trabajo ha implementado un entorno de escritorios virtuales, como una alternativa económica al producto comercial de VMWare.

La solución integrada del entorno de escritorios virtuales que se ha creado para este proyecto cumple perfectamente las expectativas de su funcionamiento, al hacerlo prácticamente con la misma calidad de experiencia de usuario que mediante el entorno comercial de VMWare, Horizon View. Además, la implementación desarrollada supuso una reducción del coste económico de hasta el 85%, lo que lo convierte en una alternativa mucho más económica que Horizon View. Con la incorporación

al entorno de los ordenadores con bajos recursos y de las Raspberry Pi, se ha conseguido disponer de clientes ligeros mucho más económicos, que los que el Ayuntamiento compraba a la empresa que les instaló el VMWare Horizon View para que tuvieran la máxima compatibilidad. Estos clientes ligeros, disponían de una interfaz centralizada para poder controlarlos y actualizarlos de forma rápida y eficaz. Para el uso de esta interfaz se obligaba al cliente a pagar una licencia por dispositivo que quisiera incorporar al sistema de administración. Con nuestro entorno, también disponemos de una configuración centralizada al tener los sistemas operativos que ejecutan en los clientes, para realizar la conexión, almacenados en una de nuestras máquinas virtuales. De esta forma, si realizamos un cambio en esta máquina, éste se realiza en todos los dispositivos del entorno cuando se reinician. El añadido de la doble autentificación, mediante las etiquetas NFC, en los dispositivos Raspberry Pi, incrementa un nivel más la seguridad. Este entorno de virtualización de escritorios ha sido implementado en una de las aulas de formación del centro cívico del Ayuntamiento de Sant Joan Despí. Así pues esta implementación se considera un éxito en la realización del proyecto.

2.2 Bases teóricas - científicas

2.2.1 Docker

Docker es un proyecto de código abierto que permite automatizar el despliegue de aplicaciones dentro de “contenedores”. (S P Polyakov1, 2018).

Éste contenedor empaqueta todo lo necesario para que uno o más procesos (servicios o aplicaciones) funcionen: código herramientas

del sistema, bibliotecas del sistema, dependencias, etc. Esto garantiza que siempre se podrá ejecutar, independientemente del entorno en el que queramos desplegarlo. No hay que preocuparse de qué software ni versiones tiene nuestra máquina, ya que nuestra aplicación se ejecutará en el contenedor.

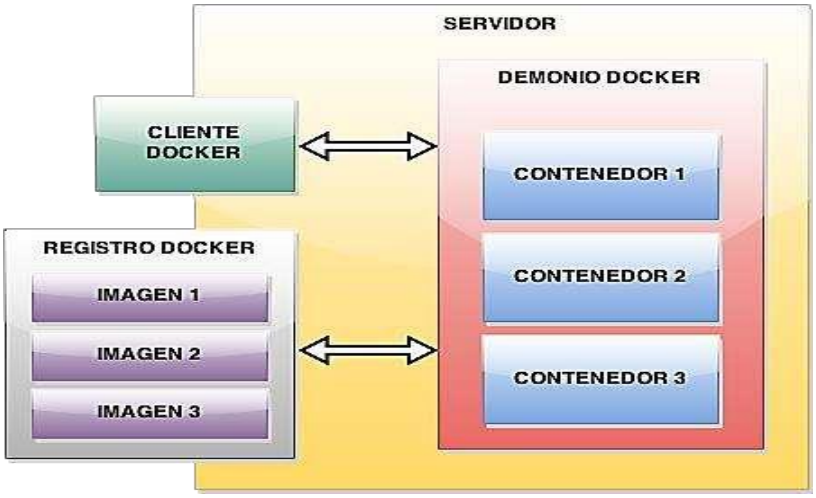


Figura 1: Arquitectura de Docker de un servidor Fuente: (Javier Martin Alonso - https://bit.ly/2Qly9JV)

2.2.2 Un poco de historia

Salomon Hykes comenzó Docker comenzó como un proyecto interno dentro de dotCloud, empresa enfocada a PaaS (plataforma como servicio). Fue liberado como código abierto en marzo de 2013.

Fotografía 1: Dockers trabajando en Bristol, Inglaterra, 1940. (Mouat, 2015)

Con el lanzamiento de la versión 0.9 (en marzo de 2014) Docker

dejó de utilizar LXC como entorno de ejecución por defecto y lo reemplazó con su propia librería, libcontainer (escrita en Go), que se encarga de hablar directamente con el Kernel.

Actualmente es uno de los proyectos con más estrellas en GitHub, con miles de bifurcaciones (forks) y miles de colaboradores.

2.2.3 Requisitos mínimos

Docker funciona de forma nativa en entornos Linux a partir de la versión 3.8 del Kernel. (GÓMEZ DE LA TORRE, 2016).

Algunos Kernels a partir de la versión 2.6.x y posteriores podrían ejecutar Docker, pero los resultados pueden variar, por lo que oficialmente no está soportado.

Para usar Docker en entornos Windows o MAC han creado una herramienta, Boot2Docker, que no es más que una máquina virtual ligera de Linux con Docker ya instalado. Dicha imagen la arrancamos con Virtualbox, Vmware o con la herramienta que tengamos instalada.

Otra manera es con un instalador “todo en uno” para MAC y Windows. Este instalador trae un cliente para Windows, la imagen de una máquina virtual Linux, Virtualbox y msys-git Unix tools. (FLOYD, 2016).

2.2.4 Características

Las principales características de Docker son:

Portabilidad

El contenedor Docker podemos desplegarlo en cualquier sistema, sin necesidad de volver a configurarlo o realizar las instalaciones

necesarias para que la aplicación funcione, ya que todas las dependencias son empaquetadas con la aplicación en el contenedor.

Ligereza

Los contenedores Docker sólo contienen lo que las diferencias del sistema operativo en el que se ejecutan, no se virtualiza un SO completo.

Autosuficiencia

Un contenedor Docker no contiene todo un sistema operativo completo, sólo aquellas librerías, archivos y configuraciones necesarias para desplegar las funcionalidades que contenga. (TURNBULL, 2014).

2.2.5 Ventajas y desventajas de Docker:

Usar contenedores Docker permite a desarrolladores y administradores de sistemas probar aplicaciones o servicios en un entorno seguro e igual al de producción, reduciendo los tiempos de pruebas y adaptaciones entre los entornos de prueba y producción.

Las principales ventajas de usar contenedores Docker son:

Las instancias se inician en pocos segundos.

a) Son fácilmente replicables.
b) Es fácil de automatizar y de integrar en entornos de integración continua.
c) Consumen menos recursos que las máquinas virtuales tradicionales.

d) Mayor rendimiento que la virtualización tradicional ya que corre directamente sobre el Kernel de la máquina en la que se aloja, evitando al hypervisor.

e) Ocupan mucho menos espacio.

f) Permite aislar las dependencias de una aplicación de las instaladas en el host.

g) Existe un gran repositorio de imágenes ya creadas sobre miles de aplicaciones, que además pueden modificarse libremente.

Por todo esto Docker ha entrado con mucha fuerza en el mundo del desarrollo, ya que permite desplegar las aplicaciones en el mismo entorno que tienen en producción o viceversa, permite desarrollarlas en el mismo entorno que tendrán en producción.

Aunque también tiene algunas desventajas:

a) Sólo puede usarse de forma nativa en entornos Unix con Kernel igual o superior a 3.8.

b) Sólo soporta arquitecturas de 64 bits.

Como es relativamente nuevo, puede haber errores de código entre versiones.

2.2.6 Usos y recomendaciones

El uso de Docker está recomendado en:

a) Entornos de integración continua, es decir, cuando el paso de desarrollo a producción en un proyecto sea lo más a menudo posible, para así poder detectar fallos cuanto antes.

b) Para garantizar la integridad de las aplicaciones en diferentes entornos.

c) Cuando necesitemos tener entornos fácilmente desplegables, portables y desechables.

d) Cuando necesitemos un entorno fácilmente escalable.

2.2.7 Arquitectura

Docker usa una arquitectura cliente-servidor. El cliente de Docker habla con el demonio de Docker que hace el trabajo de crear, correr y distribuir los contenedores. Ambos pueden ejecutarse en el mismo sistema, o se puede conectar un cliente a un demonio Docker remoto. El cliente Docker y el demonio se comunican vía sockets o a través de una RESTfull

API. (imagen página oficial). Explicamos un poco por orden la arquitectura o funcionamiento de Docker:

El cliente de Docker (Docker Client) es la principal interfaz de usuario para Docker. Él acepta comandos del usuario y se comunica con el demonio Docker. El demonio Docker (Docker Engine) corre en una máquina anfitriona (host). El usuario no interactúa directamente con el demonio, en su lugar lo hace a través del cliente Docker. (CUESTA & GONZALES, 2016).

El demonio Docker levanta los contenedores haciendo uso de las imágenes, que pueden estar en local o en el Docker Registry.

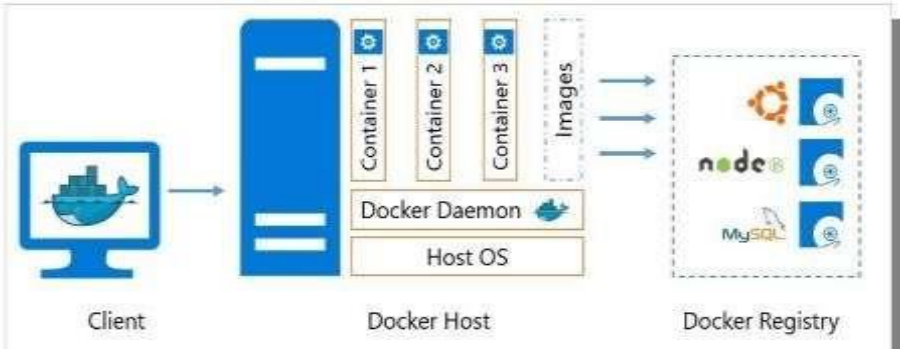


Figura 2: Representación gráfica de la arquitectura de Docker Fuente: Phpflow.com - https://bit.ly/2O6tmdU

Cada contenedor se crea a partir de una imagen y es un entorno aislado y seguro dónde se ejecuta nuestra aplicación. Componentes:

Según la documentación oficial, Docker tiene dos principales componentes:

a) Docker. - Plataforma open source de virtualización con contenedores.
b) Docker Hub. - Plataforma de Software como servicio (SaaS, Software-as-a-Servicie) para compartir y administrar contenedores Docker. Pero también necesitamos conocer otros componentes y conceptos:
c) Docker Engine. - Es el demonio que se ejecuta dentro del sistema operativo (Linux) y que expone una API para la gestión de imágenes, contenedores, volúmenes o redes. Sus funciones principales son:
    - La creación de imágenes Docker.
    - Publicación de imágenes en Docker Registry.

- Descarga de imágenes desde Docker Registry.
- Ejecución de contenedores usando las imágenes.
- Gestión de contenedores en ejecución (pararlo, arrancarlo, ver logs, ver estadísticas).

d) Docker Client. - Cualquier software o herramienta que hace uso de la API del demonio Docker, pero suele ser el comando docker, que es la herramienta de línea de comandos para gestionar Docker Engine. Este cliente puede configurarse para hablar con un Docker local o remoto, lo que permite administrar nuestro entorno de desarrollo local como nuestros servidores de producción.

e) Docker Images. - Son plantillas de sólo lectura que contienen el sistema operativo base (más adelante entraremos en profundidad) dónde correrá nuestra aplicación, además de las dependencias y software adicional instalado, necesario para que la aplicación funcione correctamente. Las plantillas son usadas por Docker Engine para crear los contenedores Docker.

f) Docker Registries. - Los registros de Docker guardan las imágenes. Pueden ser repositorios públicos o privados. El registro público lo provee el Hub de Docker, que sirve tantas imágenes oficiales cómo las subidas por usuarios con sus propias aplicaciones y configuraciones. Así tenemos disponibles para todos los usuarios imágenes oficiales de las principales aplicaciones (Postgres, MongoDB, Odoo, Tomcat, etc.), así como no oficiales de infinidad de aplicaciones y configuraciones.

DockerHub ha supuesto una gran manera de distribuir las aplicaciones. Es un proyecto open source que puede ser instalado en cualquier servidor. Además, nos ofrecen un sistema SaaS de pago.

g) Docker Containers. - El contenedor de Docker aloja todo lo necesario para ejecutar un servicio o aplicación. Cada contenedor es creado de una imagen base y es una plataforma aislada. Un contenedor es simplemente un proceso para el sistema operativo, que se aprovecha de él para ejecutar una aplicación. Dicha aplicación sólo tiene visibilidad sobre el sistema de ficheros virtual del contenedor.

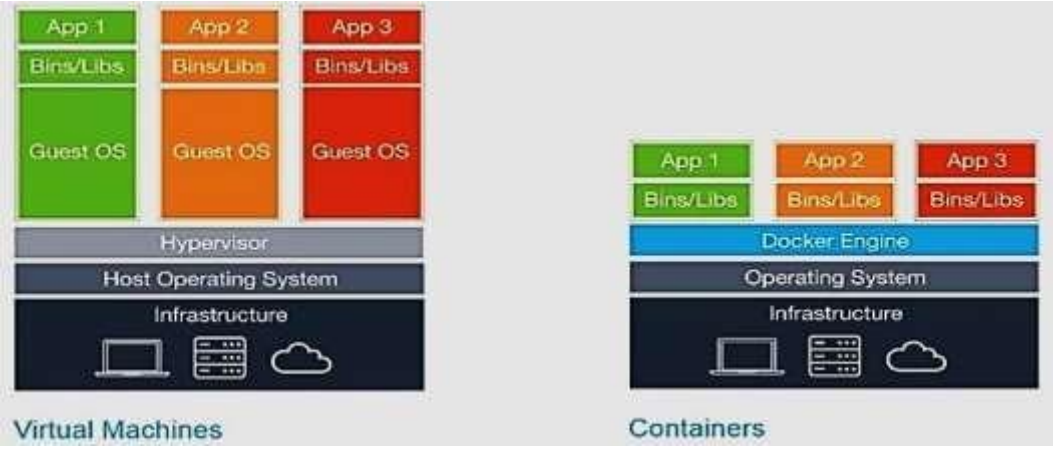
h) Docker Compose. - Es otro proyecto open source que permite definir aplicaciones multi-contenedor de una manera sencilla. Es una alternativa más cómoda al uso del comando docker run, para trabajar con aplicaciones con varios componentes. Es una buena herramienta para gestionar entornos de desarrollo y de pruebas o para procesos de integración continua.

"Docker Compose" es una herramienta para definir y correr aplicaciones Docker multi-contenedor. Docker Compose está pensado, sobre todo para el entorno de desarrollo y para cuando tenemos aplicaciones que empiezan a ser complejas porque raramente una aplicación la podemos correr con único ejecutable, sino que tiene una serie de componentes de los que dependemos alrededor. Docker Compose se basa en un fichero YML en el que se definen todos los contenedores y las relaciones que tienen

entre ellos. Podemos definir una serie de servicios diferentes cada uno con un contenedor diferente de los que podemos enlazar con contenedores locales que compilamos en el tiempo de ejecución o contenedores remotos que descargamos si fuera necesario en el momento que arrancáramos Compose.

Podemos definir todas las opciones de Docker como los puertos que exponemos los volúmenes de donde cogemos los ficheros, etc. y todas las variables de entorno o cualquier configuración que pudiéramos necesitar. No está pensada para producción, pero nos permite con más facilidad que ir lanzando contenedores uno a uno a mano desde el terminal, poder lanzar una serie de aplicaciones complejas; en este caso, simplemente con el fichero de definición lanzamos el "docker-compose up" para levantar la producción se empieza a descargar todas las imágenes que no tengamos ejecuta la instalación de dependencias.

La siguiente vez que levantamos el servicio se hará cargo de traerse todas las dependencias nuevas que tengamos dispuestas en nuestro Docker Compose. De nuevo, con un solo comando nos ha enlazado todas las imágenes y disponemos de todo el servicio en marcha. (Mouat, 2015).



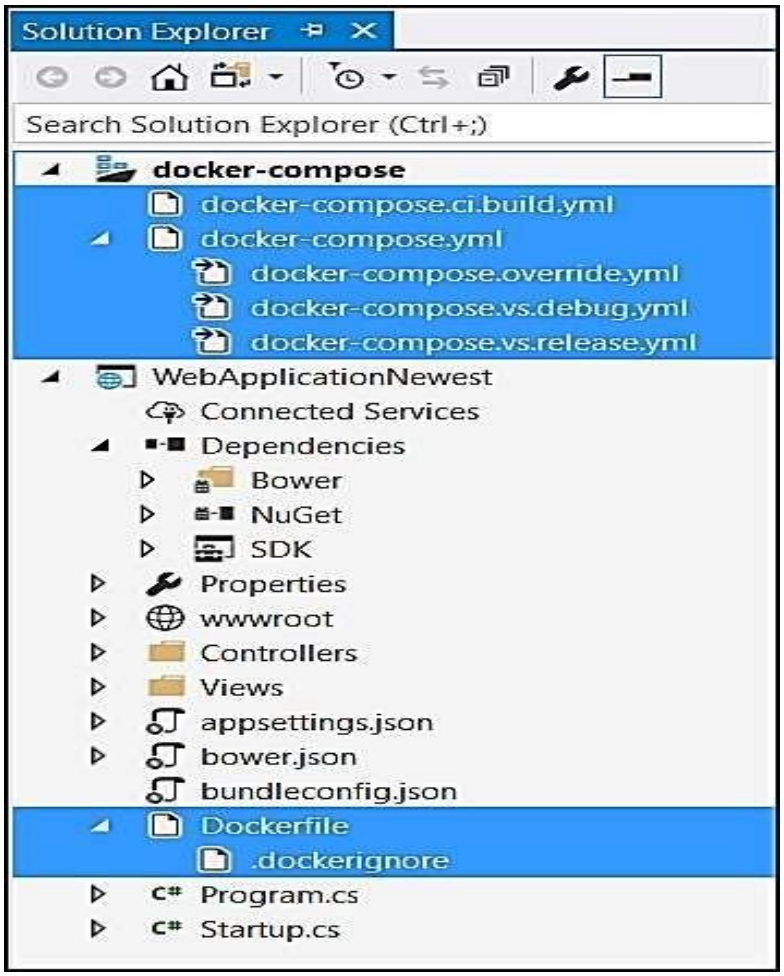


Figura 3: Docker Compose con ASP.NET Fuente: https://bit.ly/2oZfdEz

i) Docker Machine. - Es un proyecto open source para automatizar la creación de máquinas virtuales con Docker instalado, en entornos Mac, Windows o Linux, pudiendo administrar así un gran número de máquinas Docker. Usando el comando docker-machine podemos iniciar, inspeccionar, parar y reiniciar un host administrado, actualizar el Docker client y el Docker daemon, y configurar un cliente para que hable con el host anfitrión. A través de la consola de administración podemos administrar y correr comandos Docker directamente desde el host. Éste comando Docker- machine automáticamente crea hosts, instala

Docker Engine en ellos y configura los clientes Docker.

Diferencias con las máquinas virtuales:

La principal diferencia es que una máquina virtual necesita tener virtualizado todo el sistema operativo, mientras que el contenedor Docker aprovecha el sistema operativo sobre el que se ejecuta, compartiendo el Kernel e incluso parte de sus bibliotecas. Para el SO anfitrión, cada contenedor no es más que un proceso que corre sobre el Kernel. El concepto de contenedor o “virtualización ligera” no es nuevo. En Linux, LXC (Linux Containers) es una tecnología de virtualización a nivel de sistema operativo que utiliza dos características del Kernel, cgroups (que permite aislar y rastrear el uso de recursos) y namespaces (que permite a los grupos separarse, así no pueden verse unos a otros), para poder ejecutar procesos ligeros independientes en una sola máquina, aislados unos de otros, cada uno con su propia configuración de red. Ejemplos de esto son las jaulas de FreeBSD, OpenSolaris o Linux Vservers. Otra diferencia es el tamaño, una máquina virtual convencional puede ocupar bastante, sin embargo, los contenedores Docker sólo contienen lo que las diferencias del sistema operativo en el que se ejecutan, ocupando una media de 150-250 Mb. En cuanto a recursos, el consumo de procesador y memoria RAM es mucho menor al no estar todo el sistema operativo virtualizado.

(GOASGUEN, 2016)

Principales comandos Docker

Antes de empezar a usar Docker en la máquina que hemos preparado, vamos a familiarizarnos con los comandos que nos ofrece Docker. (GONZÁLEZ RODRÍGUEZ, 2017).

Escribiendo docker en la terminal nos aparece una lista de las opciones disponibles:

attach: para acceder a la consola de un contenedor que está corriendo.

build: construye un contenedor a partir de un Dockerfile.

commit: crea una nueva imagen de los cambios de un contenedor.

cp: copia archivos o carpetas desde el sistema de ficheros de un contenedor a el host.

create: crea un nuevo contenedor.

daemon: crea un proceso demonio.

diff: comprueba cambios en el sistema de ficheros de un contenedor.

events: muestra eventos en tiempo real del estado de un contenedor.

exec: ejecuta un comando en un contenedor activo.

export: exporta el contenido del sistema de ficheros de un contenedor a un archivo .tar.

history: muestra el historial de una imagen.

images: lista las imágenes que tenemos descargadas y disponibles.

import: crea una nueva imagen del sistema de archivos vacío e importa el contenido de un fichero .tar.

info: muestra información sobre los contenedores, imágenes, versión de docker.

inspect: muestra informaciones de bajo nivel del contenedor o la imagen.

kill: detiene a un contenedor activo.

load: carga una imagen desde un archivo .tar.

login: para registrarse en un servidor de registro de Docker, por defecto “https://index.docker.io/v1/”.

logout: se desconecta del servidor de registro de Docker.

logs: obtiene los registros de un contenedor.

pause: pausa todos los procesos dentro de un contenedor.

port: busca el puerto público, el cual está nateado, y lo hace

privado.

ps: lista los contenedores

pull: descarga una imagen o un repositorio del servidor de registros Docker.

push: envía una imagen o un repositorio al servidor de registros de Docker.

rename: renombra un contenedor existente.

restart: reinicia un contenedor activo.

rm: elimina uno o más contenedores.

rmi: elimina una o más imágenes.

run: ejecuta un comando en un nuevo contenedor.

save: guarda una imagen en un archivo .tar

search: busca una imagen en el índice de Docker.

start: inicia un contenedor detenido.

stats: muestra el uso de los recursos de los contenedores.

stop: detiene un contenedor.

version: muestra la versión de Docker instalada.

volume create: crea un volumen.

volume inspect: devuelve información de bajo nivel de un volumen.

volume ls: lista los volúmenes.

volume rm: elimina un volumen.

wait: bloquea hasta detener un contenedor, entonces muestra su código de salida.

2.2.8 Eliminar contenedores

Para eliminar un contenedor podemos hacerlo por el nombre o por el ID. Realmente sólo nos hace falta los 3 primeros dígitos del ID. (PETAZZONI, 2015).

Para poder eliminarlo debe estar parado. Si no lo estuviera tendríamos que pararlo con “stop”. Utilizamos el comando rm. Si queremos ahorrarnos el paso de pararlo usamos.

DOCKERFILE

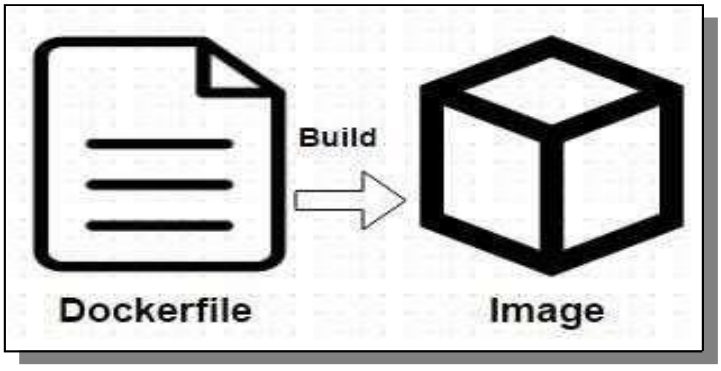


Figura 5: Dockerfile Fuente: https://red.ht/2oZhtf1

Un Dockerfile es un archivo legible por el dominio Docker, que contiene una serie de instrucciones para automatizar el proceso de

creación de un contenedor.

2.2.9 Construir el contenedor

El comando docker build irá siguiendo las instrucciones del Dockerfile y armando la imagen. El Dockerfile puede encontrarse en el directorio en el que estemos o en un repositorio.

El dominio de Docker es el que se encarga de construir la imagen siguiendo las instrucciones, línea por línea y va lanzando los resultados por pantalla. Cada vez que ejecuta una nueva instrucción se hace en una nueva imagen, son imágenes intermedias, hasta que muestra el ID de la imagen resultante de ejecutar todas las instrucciones. El daemon irá haciendo una limpieza automática de las imágenes intermedias.

Estas imágenes intermedias con la caché de Docker. Y ¿para qué sirve que Docker cree una caché de imágenes intermedias? Pues si por alguna razón la creación de la imagen falla (por un comando erróneo, por ejemplo), cuando lo corregimos y volvemos a construir la imagen a partir del dockerfile, el demonio no iniciará todo el proceso, sino que usará las imágenes intermedias y continuará en el punto dónde falló. (BAIRE, 2018).

2.2.10 Dockerfile comandos

FROM

Indicamos una imagen base para construir el contenedor y opcionalmente un tag (si no la indicamos docker asumirá “latest” por defecto, es decir buscará la última versión).

Lo que hace docker al leer esto es buscar en la máquina local una

imagen que se llama, si no la encuentra la descargará de los repositorios.

Sintaxis:

FROM <imagen>

FROM <imagen>:<tag>

MAINTAINER

Es información del creador y mantenedor de la imagen, usuario, correo, etc.

Sintaxis:

MAINTAINER <nombre> <correo> <cualquier_info>

RUN

Ejecuta directamente comandos dentro del contenedor y luego aplica/persiste los cambios creando una nueva capa encima de la anterior con los cambios producidos de la ejecución del comando y se hace un commit de los resultados. Posteriormente sigue con la siguiente instrucción.

Sintaxis:

RUN<commando>→ modo shell, /bin/sh -c

RUN [“ejecutable”,” parámetro1”,” parámetro2”] → modo ejecución, que permite correr comandos en imágenes base que no tengan /bin/sh o hacer uso de otra shell.

ENV

Establece variables de entorno del contenedor. Dichas variables se pasarán a todas las instrucciones RUN que se ejecuten posteriores a la declaración de las variables. Podemos especificar varias

variables de entorno en una sola instrucción ENV.

Sintaxis:

ENV <key> <value> <key> <value>

ENV <key>=<value> <key>=<value>

Si queremos sustituir una variable, aunque esté definida en el Dockerfile, al ejecutar un contenedor podemos especificarla y tomará dicho valor, tenga el que tenga en el Dockerfile.

docker run -env <key>=<valor>

También podemos pasar variables de entorno con el comando "docker run" utilizando el parámetro -e, para especificar que dichas variables sólo se utilizarán en tiempo de ejecución.

Podemos usar estas variables en otras instrucciones llamándolas con *$nombre_var* o *${nombre_var}*. Por ejemplo:

ENV DESTINO_DIR /opt/app WORKDIR $DESTINO_DIR

ADD

Esta instrucción copia los archivos o directorios de una ubicación especificada en <fuente> y los agrega al sistema de archivos del contenedor en la ruta especificada en <destino>.

En fuente podemos poner una URL, Docker se encargará de descargar el archivo y copiarlo al destino.

Si el archivo origen está comprimido, lo descomprime en el destino cómo si usáramos "tar -x".

Sintaxis:

ADD <fuente>...<destino>

ADD ["fuente" ..." destino"]

¿El parámetro <fuente> acepta caracteres comodín tipo?,*, etc.

Una de las cosas a tener en cuenta (entre otras → https://docs.docker.com/engine/reference/builder/#ad) es que el <origen> debe estar donde esté el Dockerfile, no se pueden añadir archivos desde fuera del directorio de construcción.

Si el destino no existe, Docker creará la ruta completa incluyendo cualquier subdirectorio. Los nuevos archivos y directorios se crearán con los permisos 0755 y un UID y GID de 0.
También tenemos que tener en cuenta que, si los archivos o directorios agregados por una instrucción ADD cambian, entonces se invalida la caché para las siguientes instrucciones del Dockerfile.

COPY

Es igual que ADD, sólo que NO admite URLs remotas y archivos comprimidos como lo hace ADD.

WORKDIR

Permite especificar en qué directorio se va a ejecutar una instrucción RUN, CMD o ENTRYPOINT.

Puede ser usada varias veces dentro de un Dockerfile. Si se da una ruta relativa, esta será la ruta relativa de la instrucción WORKDIR anterior.
Podemos usar variables de entorno previamente configuradas, por ejemplo:

ENV rutadir /ruta

WORKDIR $rutadir

USER

Sirve para configurar el nombre de usuario a usar cuando se lanza un contenedor y para la ejecución de cualquier instrucción RUN, CMD o ENTRYPOINT posteriores.

VOLUME

Crea un punto de montaje con un nombre especificado que permite compartir dicho punto de montaje con otros contenedores o con la máquina anfitriona. Es un directorio dentro de uno o más contenedores que no utiliza el sistema de archivos del contenedor, aunque se integra en el mismo para proporcionar varias funcionalidades útiles para que los datos sean persistentes y se puedan compartir con facilidad.

Esto se hace para que cuando usemos el contenedor podamos tener acceso externo a un determinado directorio del contenedor.

Las características de estos volúmenes son:

Los volúmenes pueden ser compartidos y reutilizados entre los contenedores.

a) Un contenedor no tiene que estar en ejecución para compartir sus volúmenes.
b) Los cambios en un volumen se hacen directamente.

c) Los cambios en un volumen no se incluirán al actualizar una imagen.

d) Los volúmenes persisten incluso cuando dejan de usarlos los contenedores.

Esto permite añadir datos, BBDD o cualquier otro contenido sin comprometer la imagen.

El valor puede ser pasado en formato JSON o como un argumento, y se pueden especificar varios volúmenes.

VOLUME [“/var/tmp”] VOLUME /var/tmp

LABEL

LABEL añade metadatos a una imagen Docker. Se escribe en el formato etiqueta=” valor”. Se pueden añadir varios metadatos separados por un espacio en blanco.

LABEL version=”1.0”

LABEL localizacion=” Barbate” tipo=” BBDD”

Podemos inspeccionar las etiquetas en una imagen usando el comando docker inspect. $ docker inspect

<nombre_imagen>/<tag>

ARG

Define una variable que podemos pasar cuando estemos construyendo la imagen con el comando docker build, usando el flag --build-arg <varname>=<value>. Si especificamos un argumento en la construcción que no está definido en el Dockerfile, nos dará un error.

El autor del Dockerfile puede definir una o más variables. Y también puede definir un valor por defecto para una variable, que se usará si en la construcción no se especifica otro.

ARG user1
ARG user1=someuser ARG user2

Se deben usar estas variables de la siguiente forma:

docker build –build-arg <variable>=<valor> ….

Docker tiene un conjunto de variables predefinidas que pueden usarse en la construcción de la imagen sin que estén declaradas en el Dockerfile:

HTTP_PROXY http_proxy HTTPS_PROXY https_proxy

FTP_PROXY ftp_proxy NO_PROXY no_proxy

ONBUILD

Añade triggers a las imágenes. Un disparador se utiliza cuando se usa una imagen como base de otra imagen.

El disparador inserta una nueva instrucción en el proceso de construcción como si se especificara inmediatamente después de la instrucción FROM.

Por ejemplo, tenemos un Dockerfile con la instrucción ONBUILD, y creamos una imagen a partir de este Dockerfile, por ejemplo, IMAGEN_padre.

Si escribimos un nuevo Dockerfile, y la sentencia FROM apunta a IMAGEN_PADRE, cuando construyamos una imagen a partir de

este Dockerfile, IMAGEN_HIJO, veremos en la creación que se ejecuta el ONBUILD que teníamos en el primer Dockerfile.

Los disparadores ONBUILD se ejecutan en el orden especificado en la imagen padre y sólo se heredan una vez, si construimos otra imagen a partir de la IMAGEN_HIJO, los disparadores no serán ejecutados en la construcción de la IMAGEN_NIETO.

Hay algunas instrucciones que no se pueden utilizar en ONBUILD, como son FROM, MAINTAINER y ONBUILD, para evitar recursividad.

Un ejemplo de uso:
FROM Ubuntu:14.04

MAINTAINER mcgomez RUN apt-get update && apt-get install -y apache2 ONBUILD ADD. /var/www/

EXPOSE 80 CMD [“D”,” FOREGROUND”]

EXPOSE

Se utiliza para asociar puertos, permitiéndonos exponer un contenedor al exterior (internet, host, etc.). Esta instrucción le específica a Docker que el contenedor escucha en los puertos especificados. Pero EXPOSE no hace que los puertos puedan ser accedidos desde el host, para esto debemos mapear los puertos usando la opción -p en docker run.

Por ejemplo:

EXPOSE 80 443

docker run centos: centos7 -p 8080:80

CMD

Esta instrucción es similar al comando RUN, pero con la diferencia de que se ejecuta cuando instanciamos o arrancamos el contenedor, no en la construcción de la imagen.

Sólo puede existir una única instrucción CMD por cada Dockerfile y puede ser útil para ejecutar servicios que ya estén instalados o para correr archivos ejecutables especificando su ubicación.

ENTRYPOINT

Cualquier argumento que pasemos en la línea de comandos mediante Docker run serán anexados después de todos los elementos especificados mediante la instrucción ENTRYPOINT, y anulará cualquier elemento especificado con CMD. Esto permite pasar cualquier argumento al punto de entrada. Sintaxis:

ENTRYPOINT [“ejecutable”,” parámetro1”,” parámetro2”] → forma de ejecución

ENTRYPOINT comando parámetro1 parámetro 2 → forma Shell

ENTRYPOINT nos permite indicar qué comando se va a ejecutar al iniciar el contenedor, pero en este caso el usuario no puede indicar otro comando al iniciar el contenedor. Si usamos esta opción es porque no esperamos que el usuario ejecute otro comando que el especificado.

ARCHIVO DOCKERIGNORE

Es recomendable colocar cada Dockerfile en un directorio limpio, en el que sólo agregamos los ficheros que sean necesarios. Pero es posible que tengamos algún archivo en dicho directorio, que cumpla alguna función pero que no queremos que sea agregado a la imagen. Para esto usamos. dockerignore, para que Docker build excluya esos archivos durante la creación de la imagen. Un ejemplo de dockerignore

2.2.11 Utilización de Odoo

Odoo (conocido anteriormente como OpenERP y anteriormente como TinyERP) es un sistema de ERP integrado de código abierto actualmente producido por la empresa belga Odoo S.A. El fabricante declara su producto como una alternativa de código abierto a SAP ERP y Microsoft Dynamics.

Soluciones de industria

Odoo viene provisto de módulos estándar tales como:

- Gestión de compraventa.
- CRM.
- Gestión de proyectos.

- Sistema de gestión de almacenes.
- Manufactura.
- Contabilidad analítica y financiera.
- Puntos de venta.
- Gestión de activos.
- Gestión de recursos humanos.
- Gestión de inventario.
- Ayuda técnica.
- Campañas de marketing.
- Flujos de trabajo.

Licencia e impacto en el modelo de negocios

Los módulos de Odoo, en su mayoría, están cubiertos por la licencia AGPL y algunas partes utilizan una derivada de la licencia Mozilla Public License.1 Como consecuencia directa, OpenERP no requiere ningún pago de licencias para ser utilizado, a diferencia del software más usado del mercado. Esto también implica que, mientras que se respeten los términos de la licencia, la modificación directa del programa es posible.

Arquitectura

Arquitectura WEB

Estructura Odoo está desarrollado sobre una arquitectura web. Hay disponibles múltiples aplicaciones cliente.

Servidor y módulos

El módulo del servidor está escrito en el lenguaje Python. El cliente se comunica con éste a través de interfaces XML-RPC y JSON. La funcionalidad del negocio se organiza en módulos. Los módulos no

son más que meras carpetas con una estructura predefinida, con código en Python y archivos XML en su interior. Un módulo define la estructura de los datos, formularios, informes, menús, procedimientos, flujos de trabajo, etc. Los módulos se definen mediante una sintaxis independiente del cliente, de tal forma que añadir nuevos objetos, como menús y formularios los hace disponibles para cualquier cliente.

Aplicaciones cliente

Los clientes son livianos porque no contienen la lógica del negocio. Se da soporte a dos aplicaciones oficialmente:

Una aplicación web implementada como un servidor HTTP que permite a los usuarios conectarse mediante un navegador de internet.

Una aplicación de escritorio escrita en Python utilizando el kit de herramientas GTK+ (obsoleta a partir de la versión 7).

No obstante, la comunidad ha desarrollado otros clientes alternativos.

Base de datos

OpenERP usa PostgreSQL que es un sistema gestor de bases de datos.

Informes

Odoo también cuenta con un sistema de reportes propio utilizando Webkit, y permite integración con otros motores como LibreOffice.org o Jaspersoft.

Código fuente y contribuciones

El código fuente de OpenERP se aloja en GitHub,2 utilizando el sistema de control de versiones Git. Las contribuciones y la documentación también se administran mediante GitHub. Un sitio

web dedicado a recopilar toda la documentación fue lanzado en 2009.

Software como servicio

A partir de la versión 6.0, la actual Odoo S.A. distribuye una versión de OpenERP como servicio.

Aplicaciones OpenERP

La empresa Odoo mantiene un sitio web en el que hace referencia a los módulos oficiales, así como aquellos contribuidos por la comunidad de desarrolladores en un concepto similar a las tiendas de aplicaciones de Apple y Google. Los módulos comunitarios pueden ser referenciados de forma gratuita siempre y cuando se respeten las normas de envío.

Entorno de desarrollo

El desarrollo de módulos se realiza editando archivos Python y XML. No hay un editor oficial, aunque en los tutoriales existe preferencia por Eclipse o PyCharm + PyDev. Parte de la lógica de la aplicación puede ser cambiada desde la interfaz del cliente.

Historial de versiones

Nombre del Programa	Versión	Fecha lanzamiento	Cambios Significativos	Tipo Licencia de Software
Tiny ERP	1.0	Febrero de 2005	Primera versión	GNU GPL
Tiny ERP	2.0	Mayo de 2005	Segunda versión	GNU GPL
Tiny ERP	3.0	Septiembre de 2005	Tercera versión	GNU GPL
Tiny ERP	4.0	Diciembre de 2006	Cuarta versión	GNU GPL
OpenERP	5.0	Abril de 2009	Primera versión	GNU GPL
OpenERP	6.0	Enero de 2011	Primer cliente web	GNU AGPL5

OpenERP	6.1	Febrero de 2012	Primer cliente web Ajax, se descontinúa cliente GTX	GNU AGPL
OpenERP	7.0	22 de diciembre de 2012	Se mejora el cliente web y la usabilidad	GNU AGPL
Odoo	8.0	8 de septiembre de 2014	Soporte para CMS: Generador de sitios web, comercio electrónico (ecommerce), puntos de venta einteligencia de negocio (business intelligence)	GNU AGPL
Odoo	9.0	1 de octubre de 2015	Edición comunitaria	GNU LGPL v3
Odoo	9.0	1 de octubre de 2015	Edición Empresarial	Odoo Enterprise Edition License v1.07
Odoo	10.0	5 de octubre de 2016	Edición comunitaria	GNU LGPL v37
Odoo	10.0	5 de octubre de 2016	Edición Empresarial	Odoo Enterprise Edition License v1.07
Odoo	11.0	5 de octubre de 2017	Edición comunitaria	GNU LGPL V3
Odoo	11.0	5 de octubre de 2017	Edición Empresarial	Edition License V1.0

2.2.12 Introducción a Docker Compose

Las aplicaciones basadas en microservicios se prestan a usar múltiples contenedores cada uno con un servicio, uno puede contener la base de datos PostgreSQL, otro una base de datos clave/valor redis o de documentos como elasticsearch para hacer búsquedas, otro un sistema de mensajería como rabbitmq, otro tomcat o wildfly que use los anteriores y un servidor web como Nginx.

Teniendo múltiples contenedores usar el comando docker run para cada uno de ellos nos resultará incómodo. En este punto entra Docker Compose permitiéndonos definir nuestra aplicación multicontenedor en un archivo con las mismas propiedades que

indicaríamos con el comando docker run individualmente. Con un único comando podremos iniciar todos los contenedores y en el orden que los especifiquemos.

El archivo descriptor nos puede servir no solo como forma de iniciar los contenedores en un entorno de desarrollo sino como de documentación de la aplicación en la que veremos qué contenedores, imágenes, volúmenes, enlaces y demás propiedades tienen.

a) Instalar Docker Compose

Tenemos varias formas de instalar Docker Compose. La que más me gusta y la que recomiendo por ser sencilla es descargar el binario de Docker compose según nuestra plataforma GNU/Linux o Mac. Descargando el binario de Docker Compose deberemos darle permisos de ejecución y si nos interesa colocarlo en la variable de entorno PATH del sistema:

```
$ chmod +x docker-compose
PATH=$PATH: ~/Software/scripts/docker-compose
```

Con el siguiente comando veremos que Docker Compose funciona correctamente y la versión del mismo.

```
$ docker-compose –version
docker-compose version: 1.3.1
Python version: 2.7.9
OpenSSL version: OpenSSL 1.0.1e 11 Feb 2013
```

b) El descriptor de contenedores

El descriptor de los contenedores a usar con Docker Compose es un archivo de texto con formato yml en la que especificamos los diferentes contenedores y sus propiedades, básicamente podemos indicar las mismas propiedades que indicamos arrancando los contenedores individualmente con el comando docker run. En el siguiente ejemplo vemos varios contenedores, dos contenedores de datos para redis y PostgreSQL, los contenedores de redis y PostgreSQL y un contenedor para la aplicación usando tomcat enlazado con los contenedores de redis y PostgreSQL definidos previamente.

```
redisdb:
 image: busybox volumes: /var/lib/redis

postgresqldb:
 image: busybox volumes:
/var/lib/postgresql/data
redis:
      image: redis:3 volumes_from:
- redisdb  ports:
- "6379:6379"
postgresql:
image: postgres:9 volumes_from:
- postgresqldb environment:
    POSTGRES_USER:
posgresql
POSTGRES_PASSWORD:
posgresql ports:
```

```
- "5432:5432"
  apps:

  image:

  library/tomcat:8-

  jre8 links:

- redis

- postgresql

    ports:

- "8080:8080"
```

La imagen de los contenedores se indica con la propiedad *image*, los contenedores de datos, *redisdb* y *posgresqldb*, usan la propiedad *volumes* con los datos que guardarán y la imagen de *busybox* (se suele usar esta para los contenedores de datos porque es muy pequeña), con la propiedad *hostname* podemos indicar el nombre de la máquina que al usar la propiedad *link* docker hará visible al contenedor que los usen, con *volumes_from* podemos usar volúmenes, con *links* enlazar contenedores y con *ports*asociar puertos entre los contenedores y la propia máquina anfitrión, en el ejemplo he usado los puertos por defecto de cada uno de los servicios.

La descripción completa del formato del archivo de Docker Compose nos da una idea de las opciones que podemos usar, está bastante bien explicado y con ejemplos que nos resultará sencillo entender conociendo los parámetros que usamos con *docker run.*

c) Iniciar los contenedores con Docker Compose

Escrito el archivo de los contenedores y llamándolo *docker-compose.yml* podemos iniciar los contenedores con el comando docker-compose up estando en el mismo directorio de trabajo donde esté ubicado del archivo yml (y previamente habiendo iniciado el servicio de docker). Con *docker-compose ps* podremos ver el estado de los contenedores y de cuales está compuesta la aplicación. Con la opción *–help* podemos ver la lista completa de comandos que podemos usar.

Figura 13: Iniciando servicio con el comando docker-compose up -d y listando los servicios activos
Fuente: Elaboracion propia

Docker-compose inicia los contenedores en el orden que hemos indicado en el archivo de definición, las trazas emitidas de los servicios de los contenedores aparecerán en la terminal si iniciamos los contenedores en primer plano y con *Ctrl+C* se pararán los contenedores. Indicando la opción *-d* los contenedores se iniciarán en segundo plano, con *docker-compose stop* podremos pararlos, con *docker-compose restart* reiniciarlos, *docker-compose rm* para eliminar completamente los contenedores y con *docker-compose logs*

veremos las trazas emitidas por los servicios que nos serán de utilizar si iniciamos los contenedores en segundo plano. (RODRIGUEZ GAYOSO, 2017).

2.3 Definición de términos básicos

2.3.1 ERP

Los sistemas de planificación de recursos empresariales ('ERP', por sus siglas en inglés, enterprise resource planning) son los sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía en la producción de bienes o servicios.

2.3.2 Contenedores

Un contenedor1 es un recipiente de carga para el transporte marítimo o fluvial, transporte terrestre y transporte multimodal.

2.3.3 Odoo

Odoo (conocido anteriormente como OpenERP y anteriormente como TinyERP) es un sistema de ERP integrado de código abierto actualmente producido por la empresa belga Odoo S.A. El fabricante declara su producto como una alternativa de código abierto a SAP ERP y Microsoft Dynamics.

2.3.4 Hardware

La palabra hardware en informática se refiere a las partes físicas tangibles de un sistema informático; sus componentes eléctricos, electrónicos, electromecánicos y mecánicos. Cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico

involucrado componen el hardware; contrariamente, el soporte lógico e intangible es el llamado software.

2.3.5 Cloud

La computación en la nube, conocida también como servicios en la nube, informática en la nube, nube de cómputo o nube de conceptos (del inglés cloud computing), es un paradigma que permite ofrecer servicios de computación a través de una red, que usualmente es Internet.

2.3.6 Dockerfiles

Docker puede construir imágenes automáticamente leyendo las instrucciones de a Dockerfile. A Dockerfile es un documento de texto que contiene todos los comandos que un usuario puede llamar en la línea de comandos para montar una imagen. El uso de los docker build usuarios puede crear una compilación automatizada que ejecute varias instrucciones de la línea de comandos sucesivamente. Esta página describe los comandos que puede utilizar en un archivo Dockerfile. Cuando haya terminado de leer esta página, consulte las Dockerfile mejores Prácticas para obtener una guía orientada a la punta.

2.3.7 PaaS

Plataforma como servicio (PaaS) o plataforma de aplicaciones como servicio (PaaS) es una categoría de servicios de cloud computing que proporciona una plataforma que permite a los clientes desarrollar, ejecutar y administrar aplicaciones sin la complejidad

de construir y mantener la infraestructura típicamente asociada con el desarrollo Y lanzar una aplicación. (Por ejemplo, Java runtime), y la provisión de las redes,servidores, almacenamiento, sistema operativo, middleware (por ejemplo, tiempo de ejecución Java), tiempo de ejecución de .NET, integración, etc.), base de datos y otros servicios para alojar la aplicación del consumidor; O un servicio privado (software o dispositivo) dentro del cortafuegos o software desplegado en una infraestructura pública como servicio.

2.3.8 Host

El término host o anfitrión se usa en informática para referirse a las computadoras u otros dispositivos conectados a una red que proveen y utilizan servicios de ella. Los usuarios deben utilizar anfitriones para tener acceso a la red. En general, los anfitriones son computadores monousuario o multiusuario que ofrecen servicios de transferencia de archivos, conexión remota, servidores de base de datos, servidores web, etc. Los usuarios que hacen uso de los anfitriones pueden a su vez pedir los mismos servicios a otras máquinas conectadas a la red.

2.3.9 Framework

Un framework, entorno de trabajo o marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

2.3.10 Servidor

Un servidor es una aplicación en ejecución (software) capaz de atender las peticiones de un cliente y devolverle una respuesta en concordancia. Los servidores se pueden ejecutar en cualquier tipo de computadora, incluso en computadoras dedicadas a las cuales se les conoce individualmente como «el servidor». En la mayoría de los casos una misma computadora puede proveer múltiples servicios y tener varios servidores en funcionamiento. La ventaja de montar un servidor en computadoras dedicadas es la seguridad. Por esta razón la mayoría de los servidores son procesos diseñados de forma que puedan funcionar en computadoras de propósito específico.

2.3.11 LXC

LXC (Linux Containers) es una tecnología de virtualización en el nivel de sistema operativo (SO) para Linux. LXC permite que un servidor físico ejecute múltiples instancias de sistemas operativos aislados, conocidos como Servidores Privados Virtuales (SPV o VPS en inglés) o Entornos Virtuales (EV). LXC no provee de una máquina virtual, más bien provee un entorno virtual que tiene su propio espacio de procesos y redes.

2.3.12 DotCloud

DotCloud era una Plataforma como una compañía de servicios propiedad de cloudControl. Era el desarrollador original de Docker. En enero de 2016 la empresa envió una carta a sus clientes que estaba cerrando.

2.3.13 Forks

Forks es una ciudad en el estado de Washington, EE. UU. La población era de 3.120 habitantes según el censo del 2007. Durante muchos años su economía se basó en la industria maderera. Con recientes declinaciones en la industria, Forks ha tenido que depender de correccionales cercanas, como Clallam Bay Corretional Center y Olympic Correction Center, como fuentes de trabajos. Forks es un destino popular para los pescadores deportivos que pescan salmón y trucha arco iris en los ríos cercanos. También lo es para los visitantes del Parque nacional Olympic.

2.3.14 GitHub

GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub, Inc. (anteriormente conocida como Logical Awesome). Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. El código se almacena de forma pública, aunque también se puede hacer de forma privada, creando una cuenta de pago.

2.3.15 Hipervisor

Un hipervisor (en inglés hypervisor) o monitor de máquina virtual (virtual machine monitor) es una plataforma que permite aplicar diversas técnicas de control de virtualización para utilizar, al mismo tiempo, diferentes sistemas operativos (sin modificar o modificados, en el caso de para virtualización) en una misma

computadora. Es una extensión de un término anterior, «supervisor», que se aplicaba a los kernels de los sistemas operativos de computadora.

2.3.16 API

La interfaz de programación de aplicaciones, abreviada como API del inglés: Application Programming Interface,1 es un conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

2.3.17 LOG'S

En informática, se usa el término log, historial de log o registro a la grabación secuencial en un archivo o en una base de datos de todos los acontecimientos (eventos o acciones) que afectan a un proceso particular (aplicación, actividad de una red informática, etc.). De esta forma constituye una evidencia del comportamiento del sistema. Por derivación, el proceso de generación del log se le suele llamar guardar, registrar o loguear (un neologismo del inglés logging) y al proceso o sistema que realiza la grabación en el log se le suele llamar logger o registrador.

2.3.18 AWS

Amazon Web Services (AWS abreviado) es una colección de servicios de computación en la nube (también llamados servicios web) que en conjunto forman una plataforma de computación en la nube, ofrecidas a través de Internet por Amazon.com. Es usado en aplicaciones populares como Dropbox, Foursquare, HootSuite. Es

una de las ofertas internacionales más importantes de la computación en la nube y compite directamente contra servicios como Microsoft Azure y Google Cloud Platform. Es considerado como un pionero en este campo.

2.3.19 DigitalOcean

DigitalOcean es un proveedor estadounidense de servidores virtuales privados, basado en la ciudad de Nueva York. La compañía alquila facilidades de centros de cómputo existentes, incluyendo sitios como Nueva York, Toronto, Bangalore, Amsterdam, San Francisco, Londres y Singapur.

2.3.20 Script

En informática, un script, archivo de órdenes, archivo de procesamiento por lotes o, cada vez más aceptado en círculos profesionales y académicos, guion es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Los guiones son casi siempre interpretados, pero no todo programa interpretado es considerado un guion. El uso habitual de los guiones es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario. Por este uso es frecuente que los intérpretes de órdenes sean a la vez intérpretes de este tipo de programas.

2.4 Formulación de hipótesis

2.4.1 Hipótesis general

El análisis de Docker utilizando Odoo, optimiza la gestión de la Librería del Virrey SAC.

2.4.2 Hipótesis específicas

a) La implementación del Docker-compose para la escalabilidad de Odoo optimiza la gestión a la Librería del Virrey SAC.

b) El Docker-compose contribuye en los despliegues de Odoo, en la gestión de la Librería del Virrey SAC.

c) La ejecución de la migración de Odoo optimiza la gestión de la Librería del Virrey SAC.

2.5 Identificación de variables

2.5.1 Variable independiente

Análisis Docker utilizando Odoo

2.5.2 Variable dependiente

Optimizar la gestión de la Librería del Virrey SAC.

2.6. Definición operacional de variables e indicadores

Variables	Definición	Dimensiones	Indicadores	Escala de Medición	Definición Operacional
Análisis Docker utilizando Odoo	Docker es un proyecto de código abierto que permite automatizar el despliegue de aplicaciones dentro de “contenedores”. (S P Polyakov1, 2018). Docker usa una arquitectura cliente-servidor. El cliente de Docker habla con el demonio de Docker que hace el trabajo de crear, correr y distribuir los contenedores. Ambos pueden ejecutarse en el mismo sistema, o se puede conectar un cliente a un demonio Docker remoto. El cliente Docker y el demonio se comunican vía sockets o a través de una RESTfull API. (imagen página oficial). Explicamos un poco por orden la arquitectura o funcionamiento de Docker: El cliente de Docker (Docker Client) es la principal interfaz de usuario para Docker. (CUESTA & GONZALES, 2016).	Docker utilizando Odoo	Eficiente utilización de herramientas tecnológicas Eficiente Servicio de atención Eficiente Servicio de sistema	Discreta Discreta Discreta	1=Si(100%); 2=No(0%) 1=Si(32%); 2=No(68%) 1=Si(26%); 2=No(74%)

Optimizar la gestión de la Librería del Virrey SAC	Librería El Virrey se fundó a finales de 1973 y su nombre obedece a que su fundador, Eduardo Sanseviero, era librero anticuario y pretendía instalar una librería de libros “raros, viejos y curiosos”, junto a su esposa, Chachi Sanseviero. Optimización de la gestión de la Librería del Virrey SAC constituye una potente herramienta que permite automatizar un gran número de procesos asociados a esta Librería. Por ende, invertirse menos tiempo, esfuerzo y recursos en el desarrollo de sus tareas habituales.	Gestión de la Librería	Eficiente Servicio de atención a trabajadores. Eficiente servicio de atención a clientes. Eficiente sistematización de reportes de la librería	Discreta Discreta Discreta	1=Si(100%); 2=No(0%) 1=Si(32%); 2=No(68%) 1=Si(26%); 2=No(74%)

CAPÍTULO III

METODOLOGÍA Y TÉCNICAS DE INVESTIGACIÓN

3.1 Tipo de investigación

Para el presente trabajo de investigación, es un estudio de tipo explicativo porque permite dar a conocer todos los procesos de funcionamiento de los sistema de la Librería El Virrey SAC para lo cual se recolectaron datos y componentes sobre diferentes aspectos del personal de la organización a estudiar y se realizó un análisis.

Permitieron describir los eventos del análisis de Docker al utilizar Odoo en la Librería del Virrey SAC.

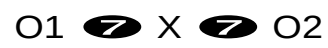
Así mismo, el estudio es de tipo Aplicada, ya que es necesario para poder analizar los resultados de las encuestas que se aplicaron.

3.2 Métodos de investigación

Se utilizó el método de análisis funcional donde se analizó la función que cumple Docker para su funcionamiento como un todo y describe como se debe utilizar asimismo también se utilizó el método análisis técnico para describir Docker y sus propiedades y se explicó el proceso de creación de contenedores para la consolidación de cada uno de los conceptos y proposiciones planteadas en el presente trabajo.

3.3 Diseño de investigación

El término "cuasi-experimento" se refiere a diseños de investigación experimentales en los cuales los sujetos o grupos de sujetos de estudio no están asignados aleatoriamente. Los diseños cuasi- experimentales, principales instrumentos de trabajo dentro del ámbito aplicado, son esquemas de investigación no aleatorios. Dado la no aleatorización, no es posible establecer de forma exacta la equivalencia inicial de los grupos, como ocurre en los diseños experimentales. Cook y Campbell (1986) consideran los cuasi- experimentos como una alternativa a los experimentos de asignación aleatoria, en aquellas situaciones sociales donde se carece de pleno control experimental. La Investigación es Cuasi Experimental, con el cuestionario de pre test y el cuestionario del post test, en dos momentos con un solo grupo que tiene el siguiente esquema:



Donde:

M = Muestra

O1 = Observación de los resultados del pre test

O2 = Observación de los resultados del post test

X = Aplicación del experimento

Nos permitió explicar la observación de los resultados obtenidos con el Docker utilizando Odoo en la Librería del Virrey SAC.

3.4 Población y muestra

3.4.1 Población

La población para el presente trabajo de investigación fue de 13 personas las cuales 12 son colaboradores y 1 el gerente de la Librería del Virrey *SAC.*

3.4.2 Muestra

En vista que la naturaleza de la presente investigación está analizando un sistema y se requiere tener la mayor cantidad de elementos en el estudio de modo que garanticen los resultados obtenidos, se optó por trabajar con una muestra intencionada de 7 personas donde 6 son colaboradores y 1 el gerente de la Librería El Virrey SAC.

3.5 Técnicas e instrumentos de recolección de datos

3.5.1 Técnicas

La presente tesis se apoyó en las siguientes técnicas:

a) Observación: Se realizó visitas permanentes a los ambientes de la Librería del Virrey SAC, de acuerdo a la programación de las actividades de la Investigación.

b) Encuesta: Esta técnica se aplicó con un cuestionario formulado

para nuestro estudio, mediante visitas a la Librería del Virrey SAC - Lima.

c) Fichaje: Mediante esta técnica se pudo abstraer información teórica procedente de la bibliografía sobre los Análisis de Docker.

3.5.2 Instrumentos

a) Hoja de Registro: Se consolidaron datos de la Liberia del Virrey SAC – Lima, de los cuales se obtuvieron las interpretaciones y comentarios para el presente trabajo

b) Escala de Actitudes: Esta escala nos permitió conocer las capacidades de adaptación en el Análisis de Docker.

c) Fichas estructuradas: Bibliográficas, hemerográficas y de resumen. Las fichas estuvieron presentes durante todo el desarrollo de la Investigación.

3.6 Técnicas de procesamiento y análisis de datos

Se realizó un consolidado a fin de depurar datos innecesarios, seguido la evaluación de los mismos ayudados con herramientas de procesamiento estadístico para los datos que recolectamos de la población y de la muestra. Todo se realizó apoyado por un trabajo de gabinete muy estricto.

3.7 Tratamiento Estadístico

a) Uso de Hoja de cálculo (Excel): Se ha utilizado el programa Excel para la tabulación de los datos y la representación gráfica de cada uno de los datos del análisis Docker, todoslos datos se muestran en el capítulo siguiente.

b) SPSS Software estadístico: En la contratación de los datos utilizamos

el SPSS que ha permitido encontrar los datos exactos, que ha fortalecido la interpretación.

3.8 Selección, validación y confiabilidad de los instrumentos de investigación

En el presente trabajo de investigación se utilizó la técnica de la encuesta y el instrumento de cuestionario en preguntas cerradas para obtener los resultados esperados. La validación utilizada fue descriptiva en la recolección de datos ya que el investigador no modifica el entorno ni controla el proceso, sino es un estudio observacional. La encuesta para asegurar la confiabilidad se realizó mediante una prueba piloto y el Coeficiente Alfa Cronbach, que permitió el desarrollo y consolidación de la presente investigación.

3.9. Orientación ética

El presente trabajo de investigación, contiene información autorizada de personas quienes participaron en la aplicación de los instrumentos y que estos fueron incorporados durante el proceso de sistematización de datos. Se realizó la encuesta a la muestra de forma aleatoria dentro la población de estudio. La información contenida en esta investigación considera todos los aspectos éticos afín de cumplir los propósitos de esta investigación.

CAPITULO IV

RESULTADOS Y DISCUSIÓN

4.1 Descripción del trabajo de campo

Como parte del trabajo de campo nos ubicamos en el lugar:

4.1.1 Reseña histórica

Librería El Virrey se fundó a finales de 1973 y su nombre obedece a que su fundador, Eduardo Sanseviero, era librero anticuario y pretendía instalar una librería de libros “raros, viejos y curiosos”, junto a su esposa, Chachi Sanseviero.

Pero, mientras los libros viejos se iban acumulando en la trastienda de la librería, su actividad se inicia como una librería convencional, en la calle Miguel Dasso del distrito de San Isidro. Eduardo Sanseviero, gran jugador de ajedrez y político empedernido, organizaba amenas tertulias literarias y políticas intercaladas con veloces partidas de ajedrez. En breve tiempo fue frecuentada por escritores, políticos e intelectuales. Desde entonces, mantiene su perfil de librería tradicional destacándose en el ambiente librero limeño.

4.1.2 Misión y Visión

a) Misión:

La Librería “El Virrey” propicia el libre acceso a la información, promoviendo la creación y difusión de cultura, en búsqueda permanente de la generación del conocimiento. Contribuye a la comprensión del entorno social y a la formación de sujetos críticos e

independientes, al mejoramiento de la calidad de vida y a la transformación social de la comunidad. Fomenta la diversidad cultural, sosteniendo espacios para la investigación, la recreación y la divulgación de identidad y memoria documental local y regional.

b) Visión:

Aspiramos, a ser un espacio cultural, particular y referente en constante crecimiento, haciéndolo sostenible, accesible e incluyente que ponga en contexto lo público, y puede servir de referente inspirador local y regionalmente.

Continuar con la consolidación de un proyecto cultural integral, en red, enfocado al desarrollo de las colecciones cualquiera sea su soporte; contribuyendo a la formación de públicos autónomos y a la generación de acciones y servicios en espacios adecuados y con tecnologías vigentes.

4.1.3 Valores:

Vocación de servicio. Pretendemos satisfacer las inquietudes y

necesidades culturales de los clientes, con educación y respeto.

Responsabilidad y formalidad ante los clientes, proveedores y colaboradores en todos nuestros compromisos.

Experiencia. El personal lleva trabajando en la misma empresa varias décadas.

4.1.4 Objetivos

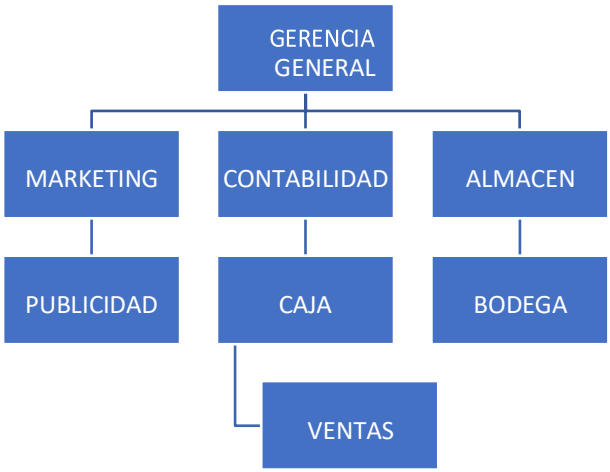
El Virrey es en la actualidad la librería más emblemática de Lima, “Construimos nuestros cimientos poco a poco, apostando por la valorización de cliente sobre todas las cosas y por la potenciación de la cultura en un mundo sobre informado, pero, sin embargo, carente de cultura”.

Nuestro objetivo, finalmente conquistado, al enfrentarnos a este proyecto era fomentar una costumbre: que, en el Día Internacional del Libro, todos regalasen uno. (En tal sentido, el Día Internacional del Libro no es necesariamente un día para leer, sino para que otros lean) y posicionar a la Librería El Virrey como la marca de la literatura construyendo una comunidad activa en las redes sociales con nuestro target (lectores).

Él éxito viene de la mano de una revolucionaria estrategia de comunicación que se asienta en el patrocinio deportivo, el apoyo a la cultura urbana, la organización de eventos y la producción de contenidos audiovisuales con una estética propia y muy reconocible. Para ello diseñamos una promoción de descuento, con una campaña viral que podría rebotar fácilmente en medios. Para ello contamos con el apoyo de líderes de opinión para grabar un video

en el que le dedican o regalan un libro a alguien; ese “alguien” podría ser un colega, amigo, pariente o rival.

4.1.5 Organigrama Empresarial



4.1.6 Localización:

Figura 41: Mapa de Librería el Virrey SAC Fuente: Google

4.2 Presentación, análisis e interpretación de resultados.

Primeramente, se realizó el trabajo en gabinete, se continuó con el trabajo de campo, lo cual nos permitió contrastar el aspecto teórico científico con la aplicación práctica, cumpliendo las siguientes actividades:

4.2.1 Descripción del Trabajo de Campo

Para efectuar los procesos del trabajo de campo se realizaron visitas a la Librería donde se realizará el estudio de investigación para ello se realizó lo siguiente:

1. Petición de permiso para el estudio a la Librería:

   Se realizó la petición de permisos de ingreso a los ambientes de la librería, así como permiso para observar el funcionamiento del sistema

2. Primera visita:

   Se Realizó la primera visita para observar la cantidad de empleados y la cantidad de usuarios de la librearía lo cual nos ayudara a realizar los instrumentos de investigación estadística

3. Segunda visita:

   Se realizó esta visita para poder realizar un cronograma de visitas posteriores para realizar encuestas a los empleados, observar el funcionamiento del sistema y realizar encuestas a los clientes de la librería. Posteriormente se realizaron visitas técnicas durante dos meses donde trabajamos de manera frecuente revisando la plataforma de la Librería y desarrollando nuestro trabajo de manera más amplia.

Se efectuó asumiendo los siguientes procesos:

- Ejecución del proyecto de investigación:

  La ejecución del proyecto de investigación se realizará en la

Librería del Virrey mediante la recolección de información y elaboración de instrumentos.

- Elaboración de los instrumentos de medición:

  Se realizará la elaboración de 2 instrumentos de medición de datos

- Aplicación de los instrumentos investigativos:

  Se aplicarán los instrumentos según un cronograma de actividades el cual se detallará las actividades a realices para la aplicación de instrumentos

- Revisión analítica-crítica e interpretación de los datos recolectados del Sistema de Librería del Virrey:

  Se realizó las interpretaciones de los datos obtenidos.

Cronograma de actividades de recolección de información de los Empleados, Clientes y observación de funcionamiento del sistema.

Actividades por semanas	**Mes Abril**				**Mes Mayo**			
								4
Elaboración de encuestas y encuesta a primer grupo de clientes								
Encuestas a segundo grupo de clientes								
Encuestas a tercer grupo de clientes								
Encuestas a cuarto grupo de clientes								
Elaboración de encuestas y encuesta a primer grupo de empleados								
Encuesta a segundo grupo de empleados								

Observación de Funcionamiento de sistema								
Observación de Funcionamiento de sistema								**x**

- **Pre-Test**

(Sistema de Librería del Virrey SAC)

Como observamos en las siguientes imágenes la librería cuenta con un sistema de escritorio el cual no es un sistema Online y se pudo observar la dificultad en la obtención de información en tiempo real.

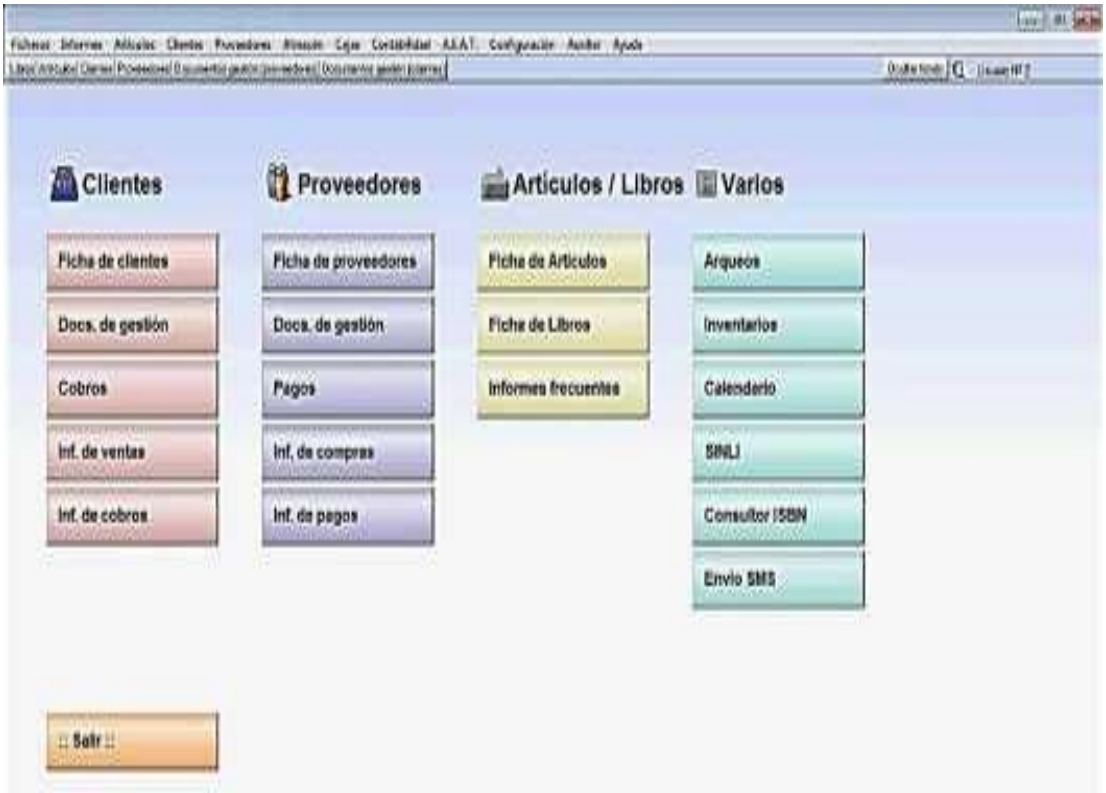


FIGURA 14: Ventana de inicio del sistema de la Librería.

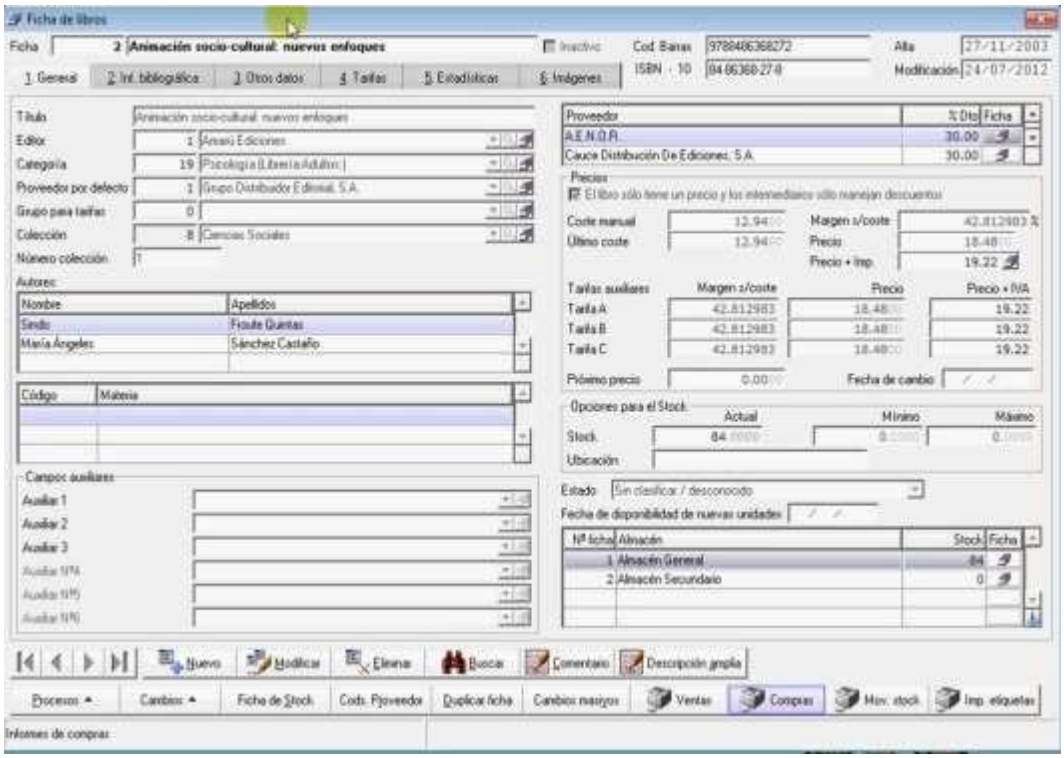


FIGURA 15: Ventana para realizar un proceso de venta.

FIGURA 16: Ventana de búsqueda de libros para una venta.

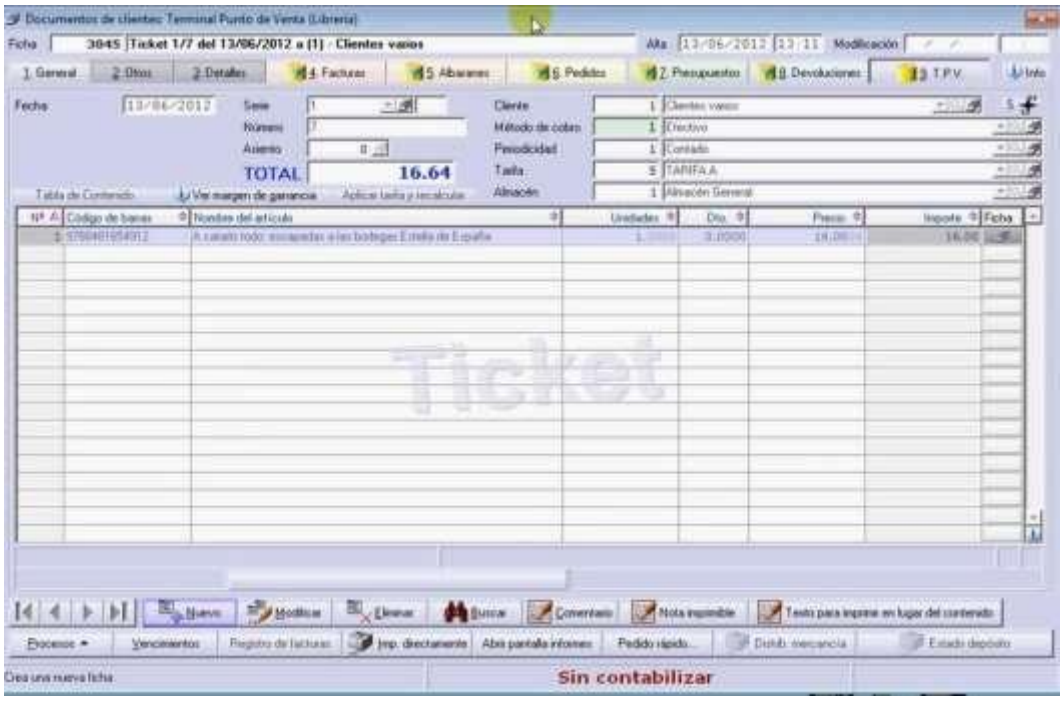


FIGURA 17: Ventana para ingresar los datos para una venta de un libro.

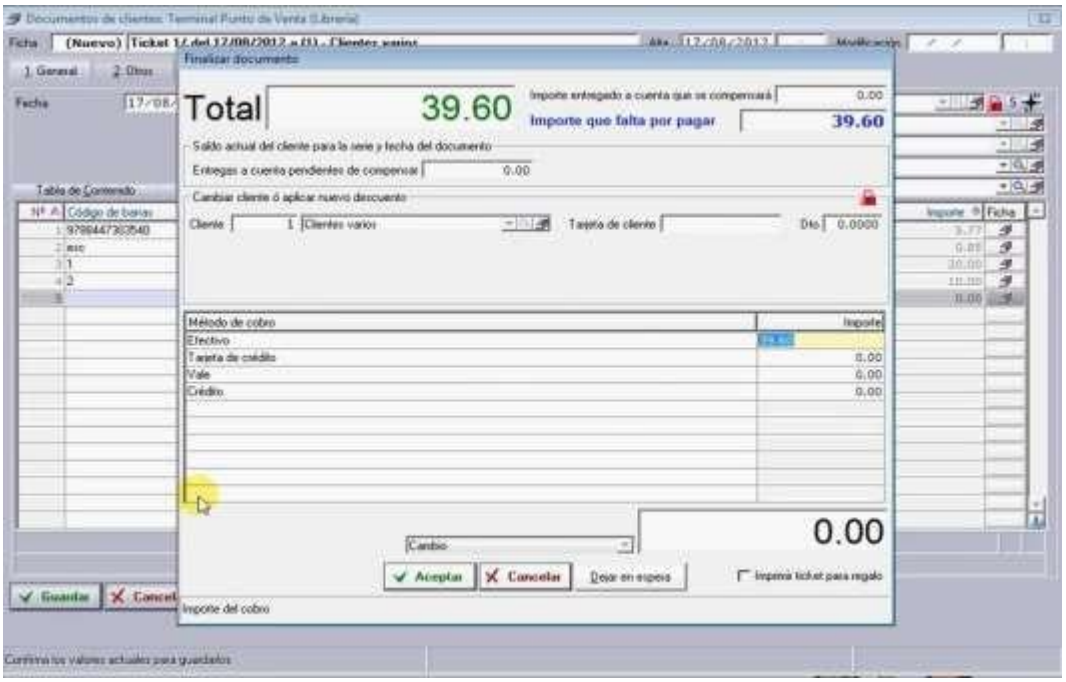


FIGURA 18: Realizando el pago de la venta realizada anteriormente.

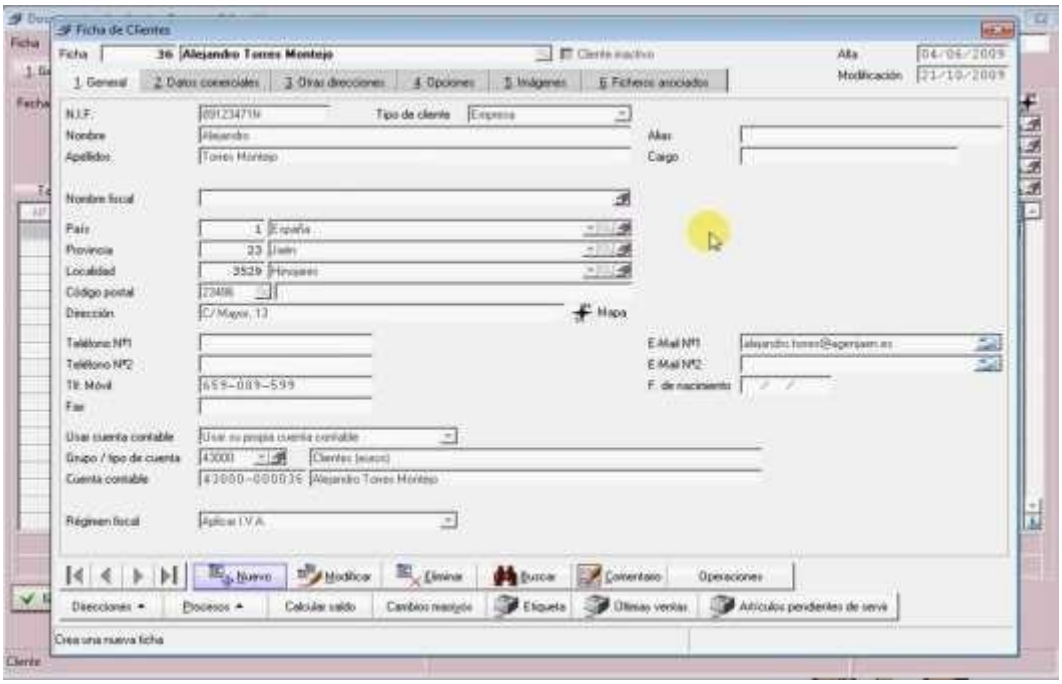


FIGURA 19: Ventana para impresión del ticket de la venta.

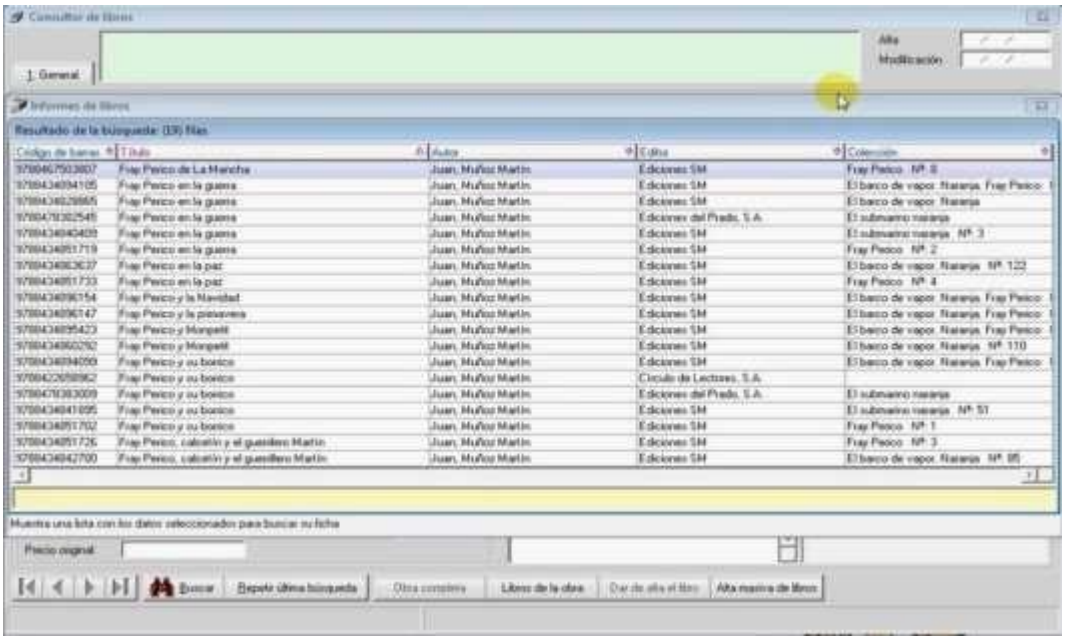


FIGURA 20: Ventana del listado de libros disponibles.

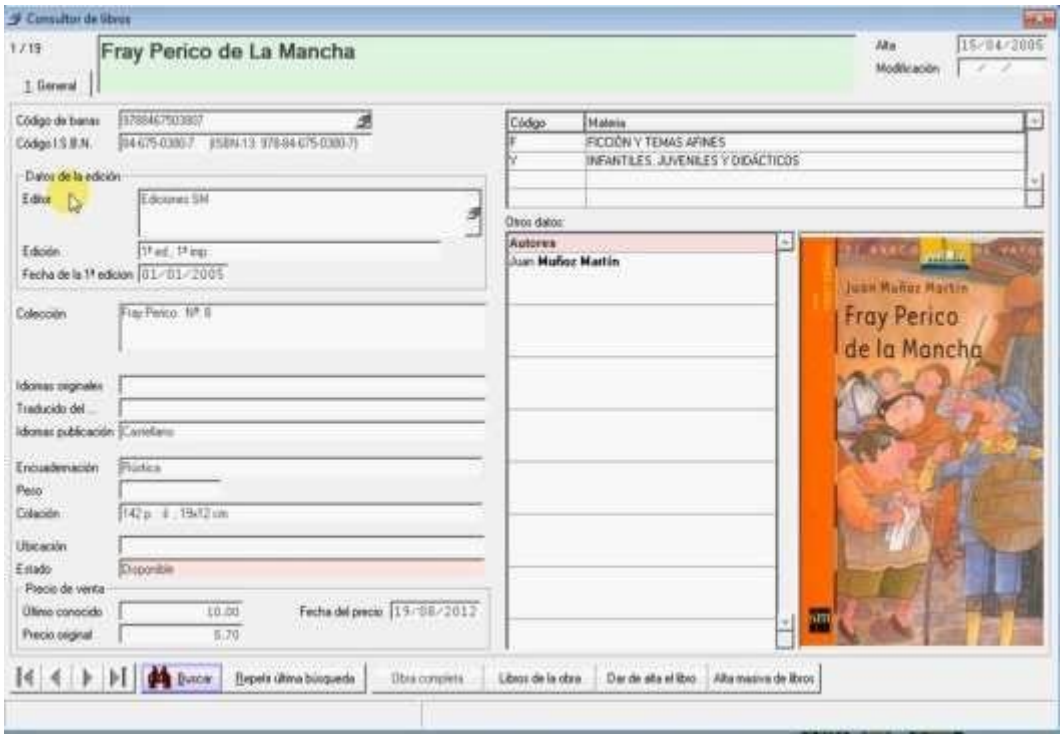


FIGURA 21: Ficha informativa del libro seleccionado.

Fuente: Librería del Virrey SAC

Posterior a realizar la prueba Pre - test se presento la propuesta para la creación de un contenedor Docker con Odoo el cual reemplazara al sistema que se observo en el Pre – test

- **Post – Test**

(Creación de contenedor utilizando Odoo)

Luego de la propuesta de la creación del contenedor Docker utilizando Odoo, se realizó la creación del contenedor paso a paso como se observará en las siguientes imágenes dicho proceso se realizó con la minuciosa observación de los colaboradores de la

librería específicamente del área de sistemas encargados del sistema que cuenta la librería, adicionalmente se explico cada paso de la creación y así poder realizar la prueba Post – Test y poder comparar con la prueba Pre - Test.

Figura 22: Acceso a la cuenta una de Amazon para creación de un servidor en la nube.

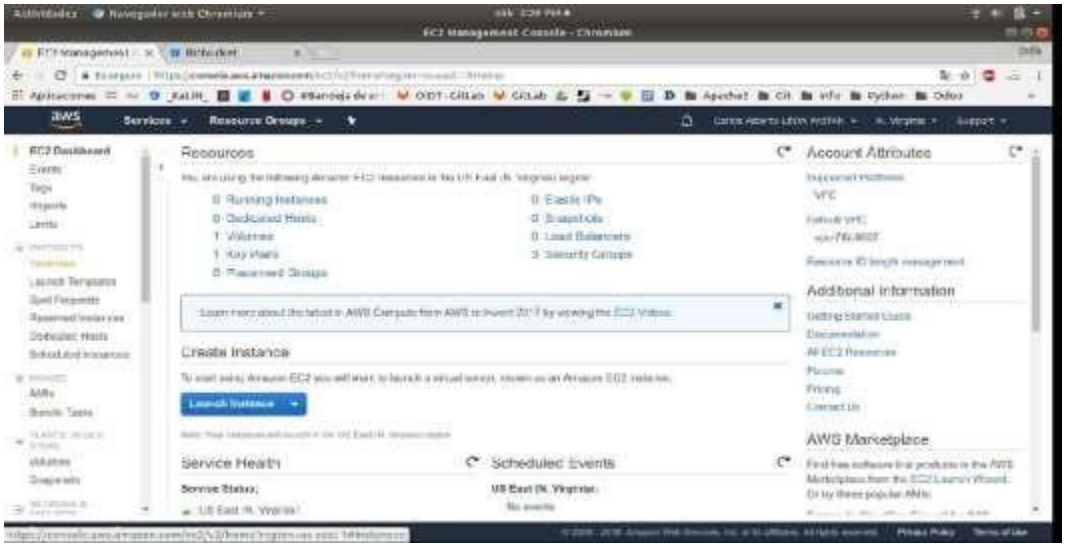


FIGURA 23: Menú de configuración de instancias de Amazon web servicie información de recursos e instancias disponible

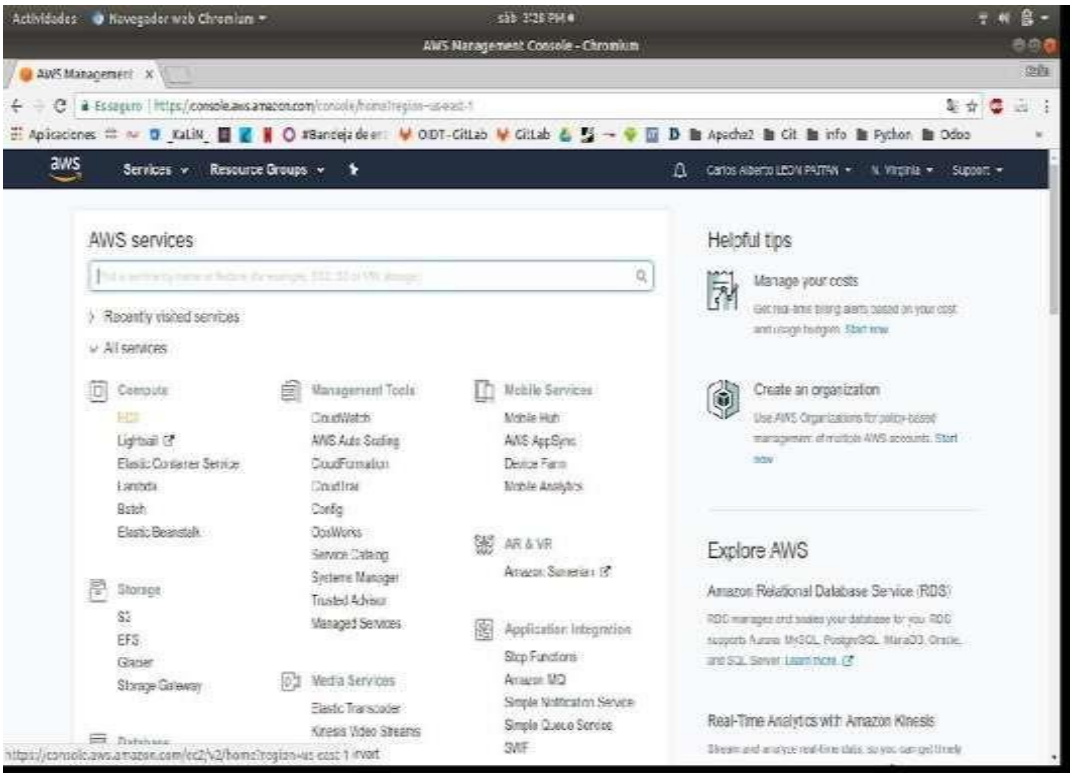


FIGURA 24: Seleccionando el servicio E2 de AWS para crear el servidor que contendrá nuestro servicio Docker.

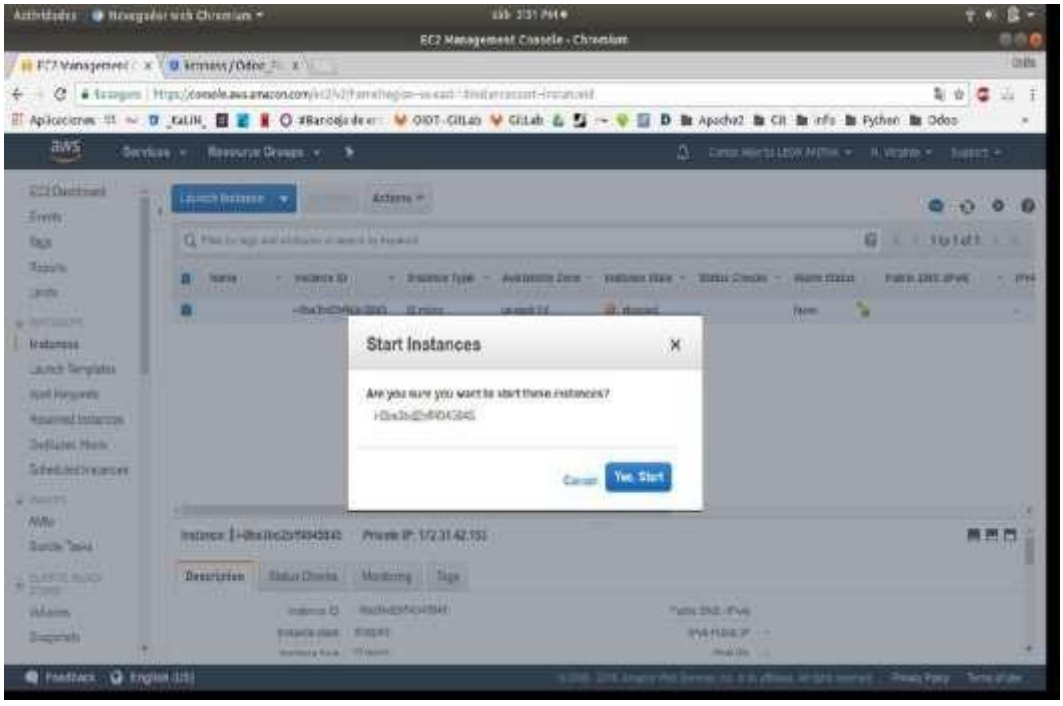


Figura 25: iniciando instancia que contiene el servidor Ubuntu 16.04

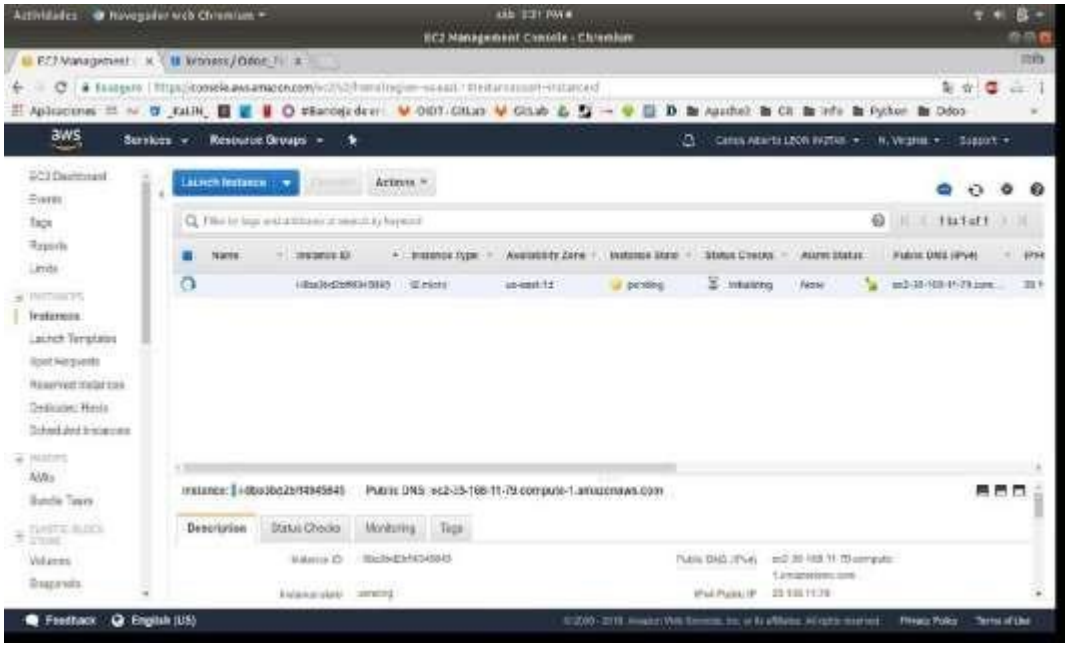


FIGURA 26: Pantalla en espera de inicio de instancia

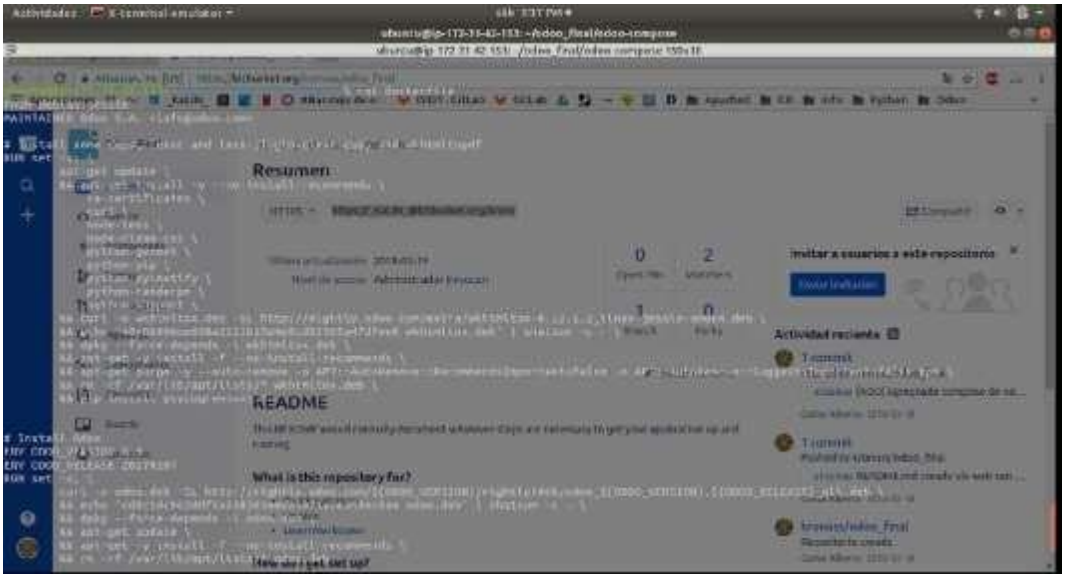


FIGURA 27: Dockerfile de Odoo.

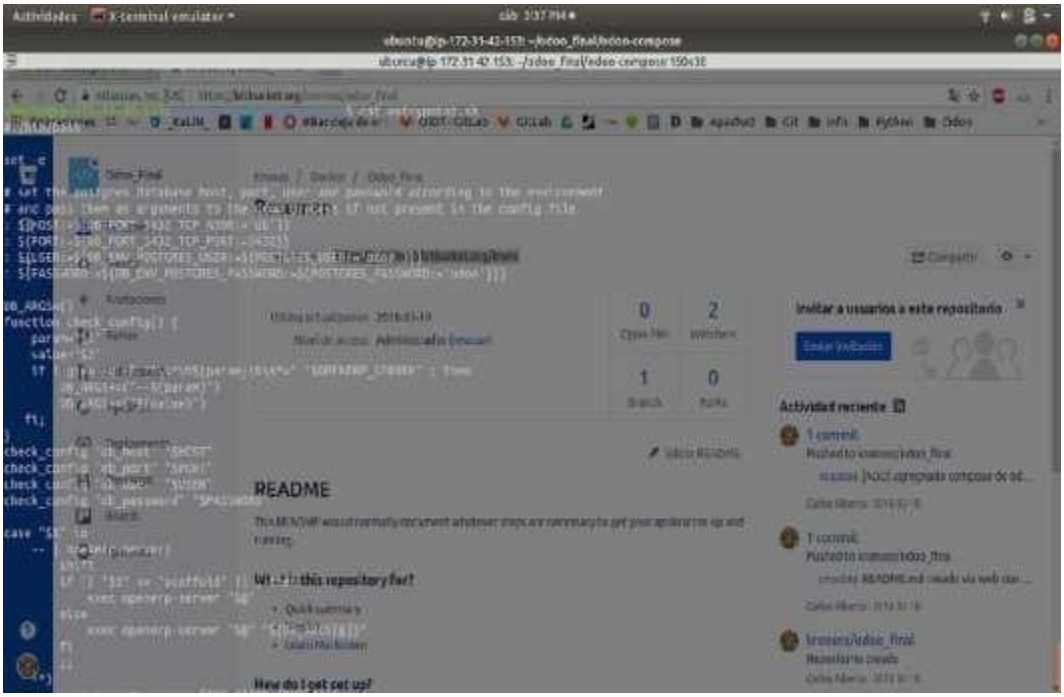


FIGURA 28: Script de Odoo para el contenedor.

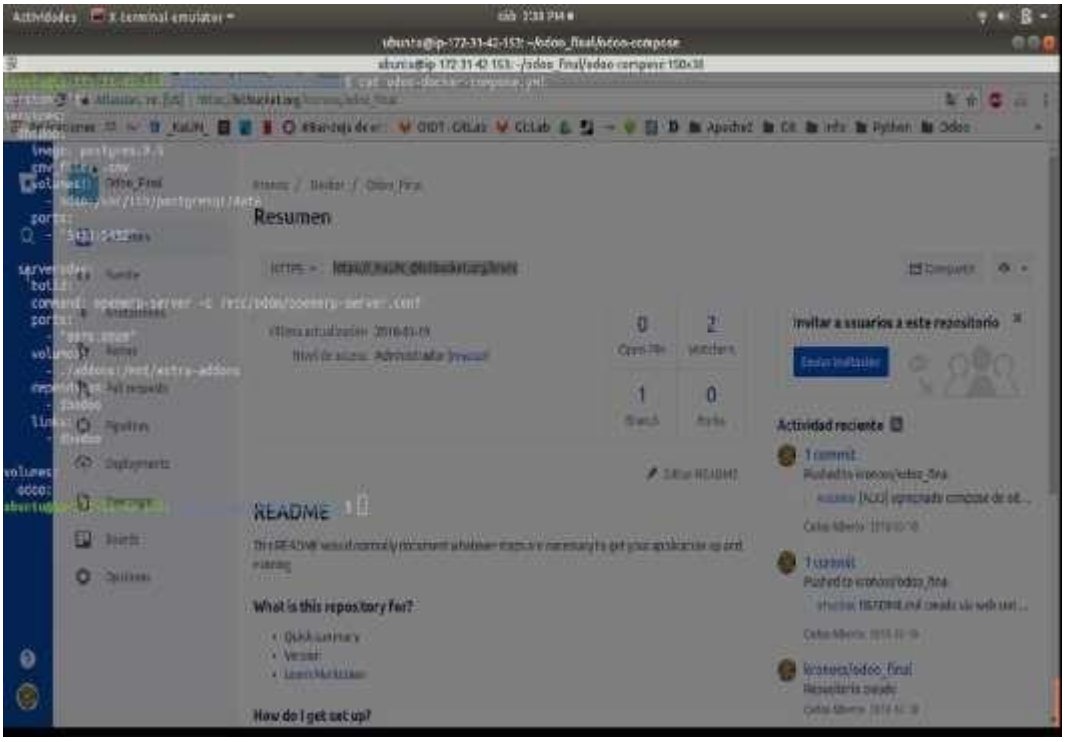


FIGURA 29: Docker compose - aqui configuramos los puertos del servicio y de la Base de datos

FIGURA 30: Archivo de configuración para Odoo donde redireccionaremos modulo para ERP en caso tener uno.

FIGURA 31: Con el comando docker-compose -f odoo-docker-compose.yml up -d empezamos la creación del contenedor y levantamos el servicio Odoo.

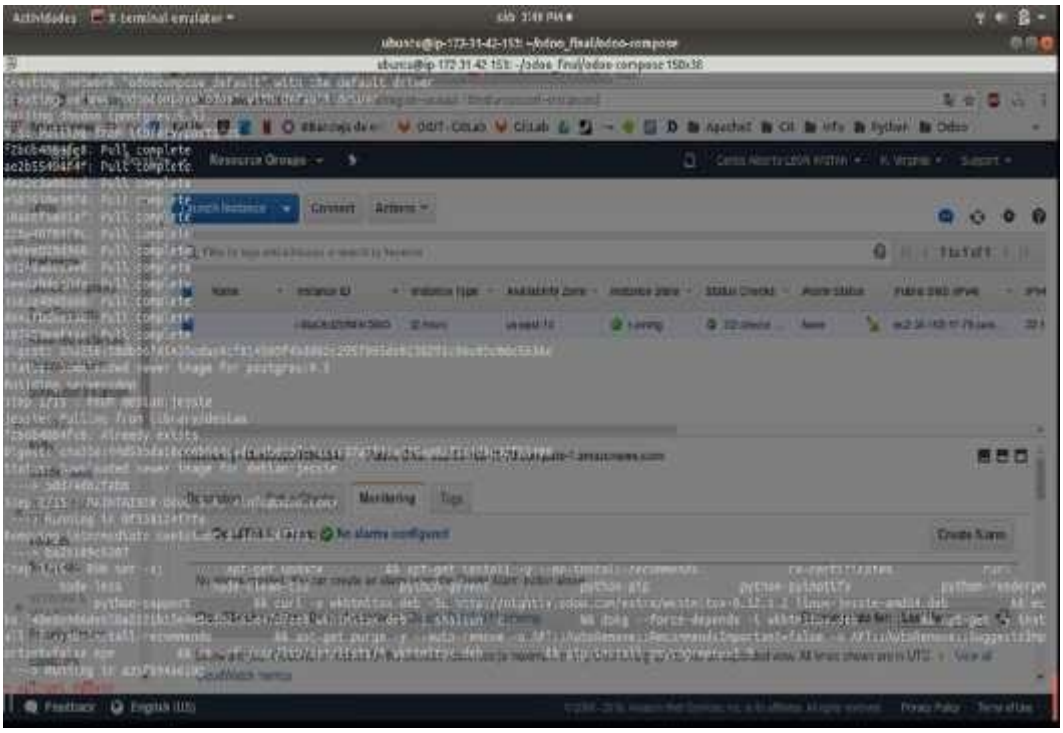


FIGURA 32: Como observamos ya tenemos el servicio levantado y tenemos que crear la Base de datos.

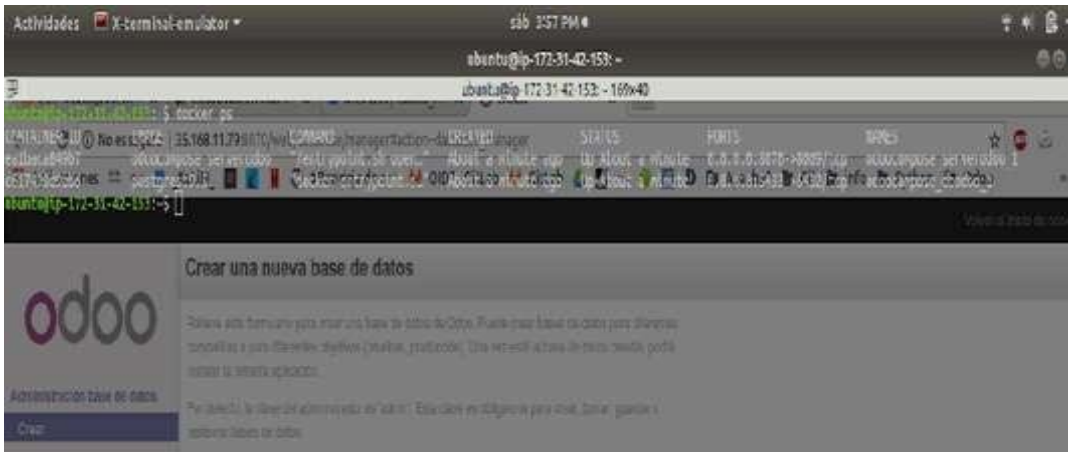


FIGURA 33: Con el comando Docker ps observamos que el servicio Odoo está corriendo con su respectivo base de datos.

FIGURA 34: Procedemos a instalar los módulos necesarios para que puedan realizar sus actividades normalmente

FIGURA 35: Como se observa ya tenemos la instalación del módulo de ventas

FIGURA 36: Importación de productos para la Base de datos

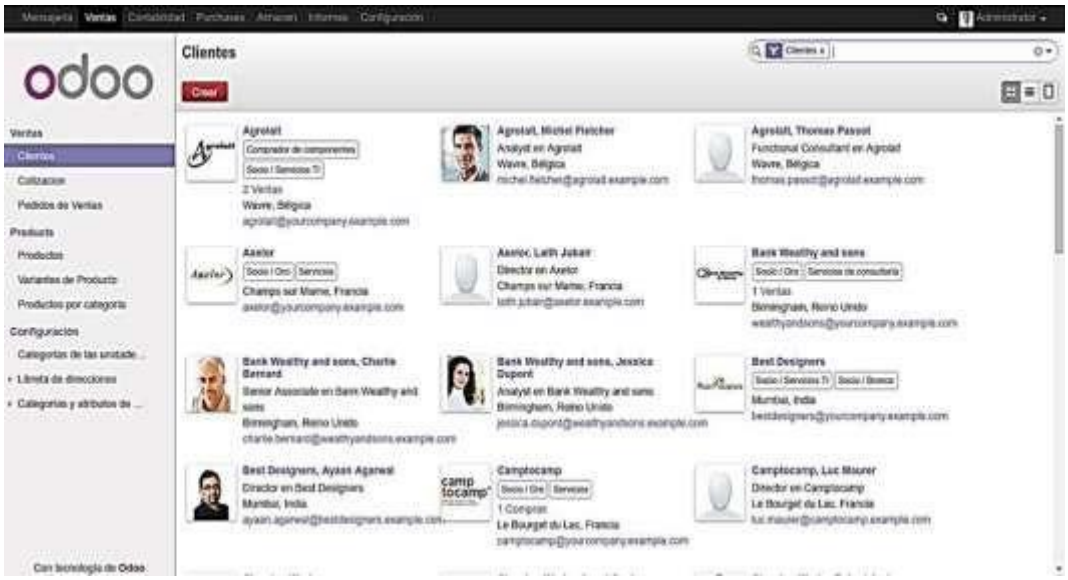


FIGURA 37: Como observamos tenemos la lista de Clientes de la librería

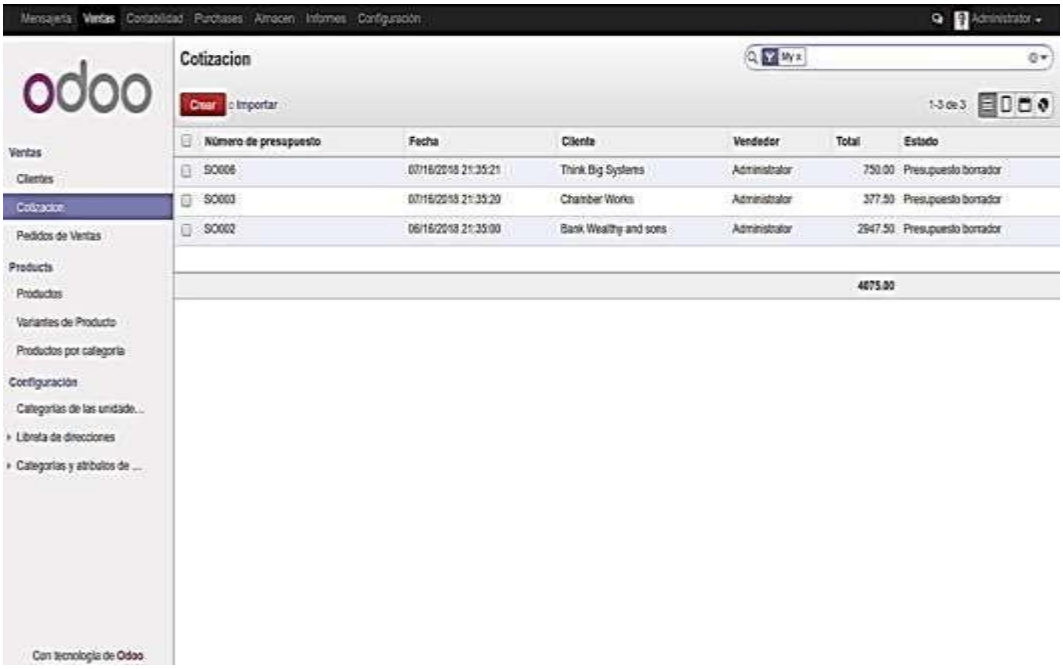


FIGURA 38: Realizando un proceso de venta.

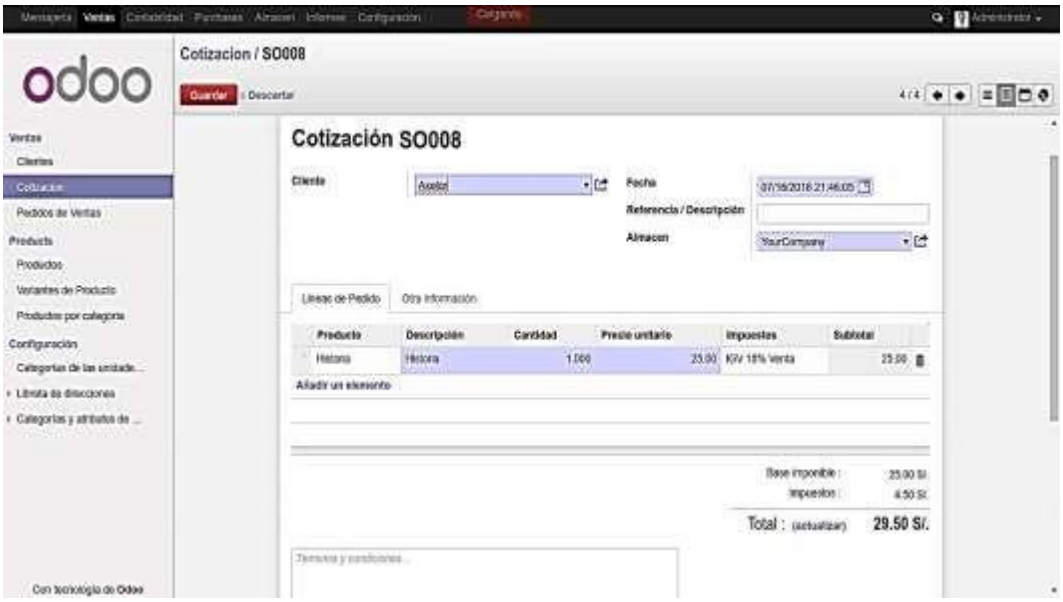


FIGURA 39: Registros de venta en la cual se Ingresa los datos para la venta

FIGURA 40: Observamos que nuestra venta está en estado borrador

FIGURA 41: Como observamos nuestra venta está lista para facturar

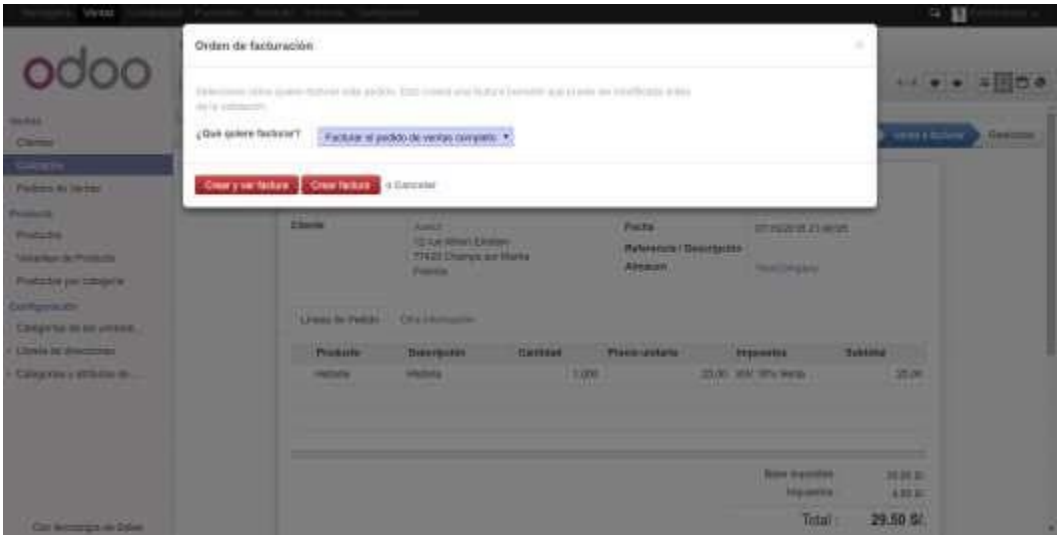


FIGURA 42: Creando la factura de la venta

FIGURA 43: Como observamos tenemos nuestra factura en estado borrador

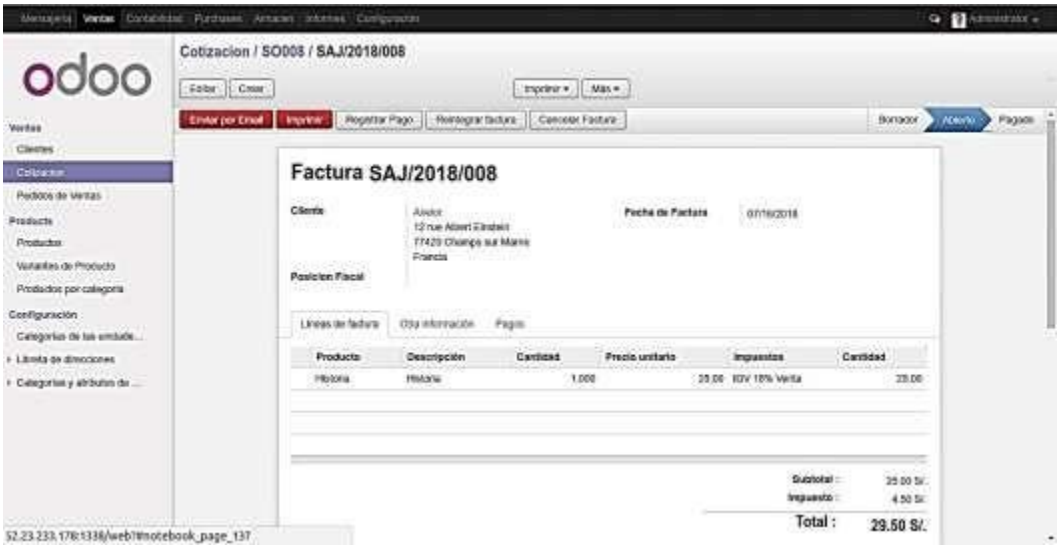


FIGURA 44: Factura en estado abierto listo para realizar el pago posteriormente realizaremos el pago respectivo.

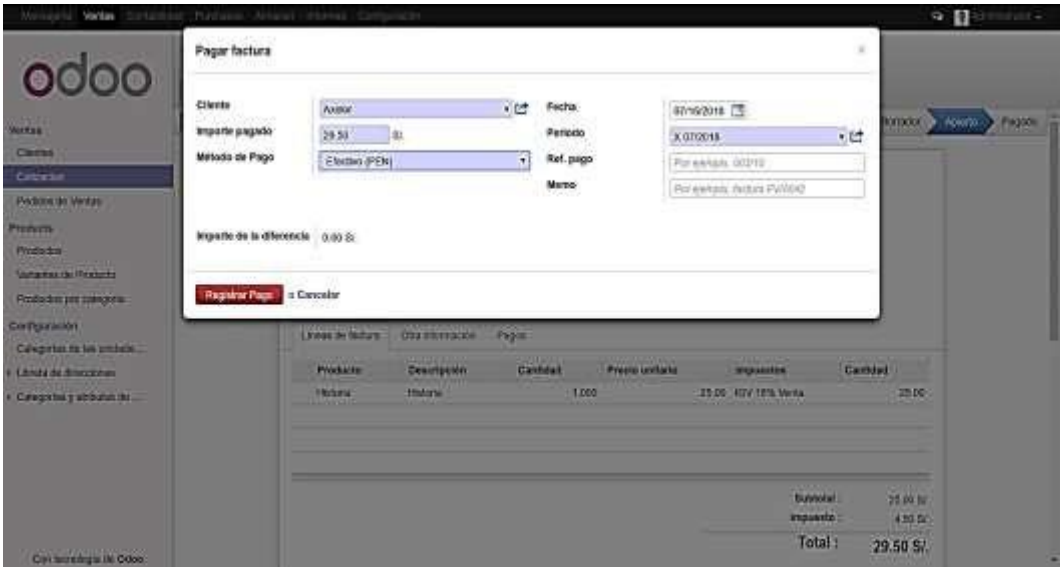


FIGURA 45: Realizamos el pago de la factura respectiva.

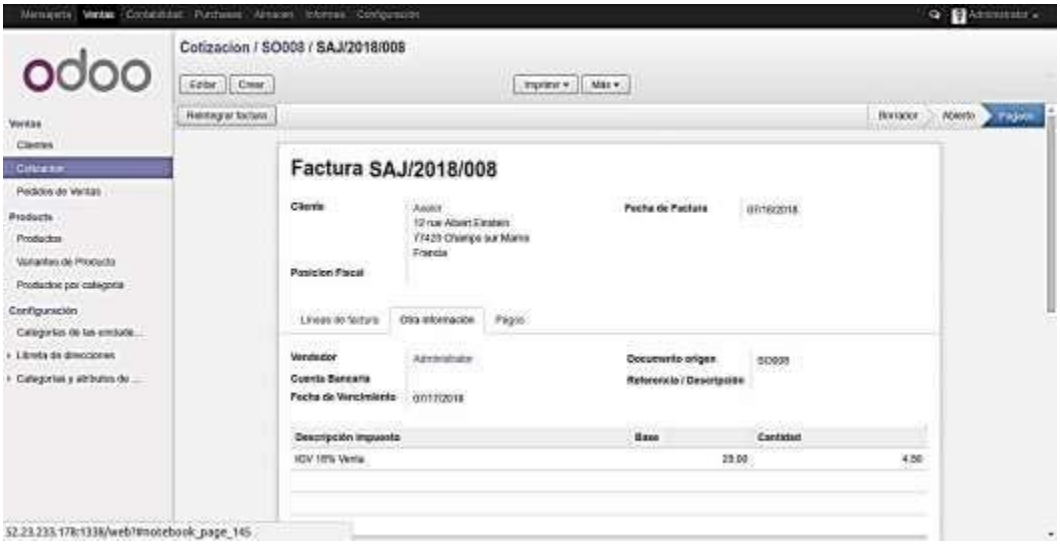


FIGURA 46: Como observamos tenemos la factura en estado pagado.

FIGURA 47: También podemos observar el historial de pagos de la factura.

Fuente: Elaboración propia.

odoo-docker-compose.yml

```
version: '2' services:
  dbodoo:
    image:
    postgres:9
    .5
    env_file:
    .env
    volumes:
      - db_odoo:/var/lib/postgre
    sql/data ports:
      - "5433:5432"

                              serverodoo: build: . volumes:
      - ./src:/opt/odoo/server
    command: python odoo.py -c odoo-
    server.conf ports:
      - "8070:8069"
    depends_on:
      dbodoo links:
      - dbodoo

      volumes: db_odoo:
```

Dockerfile

```
FROM python:2.7

ENV ODOO_USER=odoo
ENV BASE_HOME=/opt
ENV ODOO_HOME=/odoo/server

RUN set -x; \
        apt-get update \
        && apt-get install -y --no-install-
            recommends \ ca-certificates \
                    curl \ vim \
            node-less \
            node-clean-css \ python-gevent \ python-renderpm \
```

```
python-pip \ python-dev \
python-setuptools \ libevent-dev \ libsasl2-dev \ libldap2-dev \ libssl-dev \
                        postgresql
        -client \ && curl -o
        wkhtmltox.deb -SL
http://nightly.odoo.com/extra/wkhtmltox-0.12.1.2_linux-
jessie- amd64.deb \
        && echo
'40e8b906de658a2221b15e4e8cd82565a47d7ee8
wkhtmltox.deb' | sha1sum -c - \
        && dpkg --force-depends -i wkhtmltox.deb \
        && apt-get -y install -f --no-install-
        recommends \ && apt-get purge -y --auto-
        remove -o
APT::AutoRemove::RecommendsImportant=
false -o
APT::AutoRemove::SuggestsImportant=fa
lse npm \
        && rm -rf /var/lib/apt/lists/*
        wkhtmltox.deb \ && pip install
        psycogreen==1.0

RUN groupadd -r $ODOO_USER && useradd -r -g $ODOO_USER
$ODOO_USER
RUN mkdir -p $BASE_HOME$ODOO_HOME

WORKDIR $BASE_HOME$ODOO_HOME
ADD ./src $BASE_HOME$ODOO_HOME
RUN pip install -r
    requirements.txt \ &&
    pip install pyinotify \
    && pip install XlsxWriter

RUN chown $ODOO_USER:$ODOO_USER -R
    $BASE_HOME/* \ && mkdir -p
    /var/lib/odoo \
    && chown $ODOO_USER:$ODOO_USER -R /var/*
USER $ODOO_USER
```

.env

```
POSTGRES_USER=odoo
POSTGRES_PASSWORD=odoo
```

Fuente: Elaboración propia.

Posterior la creación del contenedor Docker con Odoo con la observación de los colaboradores de la librería y al realizar la prueba Post – test.

POST – TEST para el Gerente de la Librería en el cual se presentarán capturas de pantalla de datos que ayuden amejorar la gestión de la librería

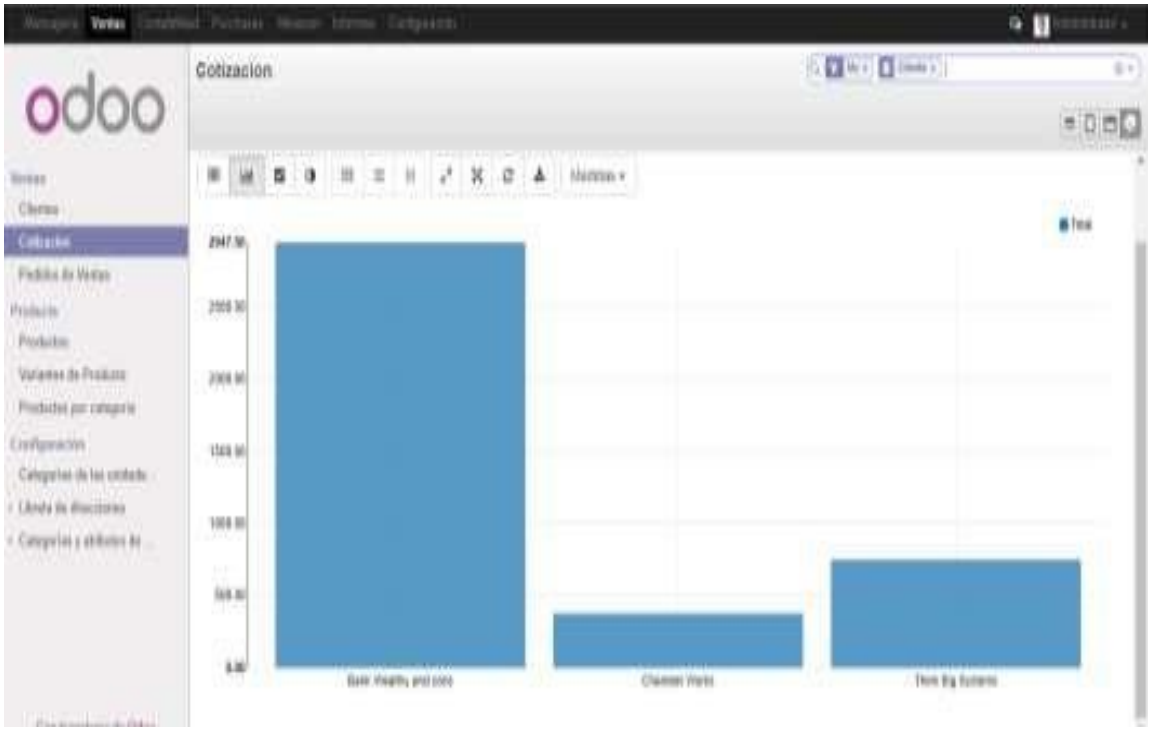


FIGURA 48: El Gerente de la librería observa el crecimiento de las ventas en tiempo real y con graficas que facilitan su trabajo.

Fuente: Elaboración propia.

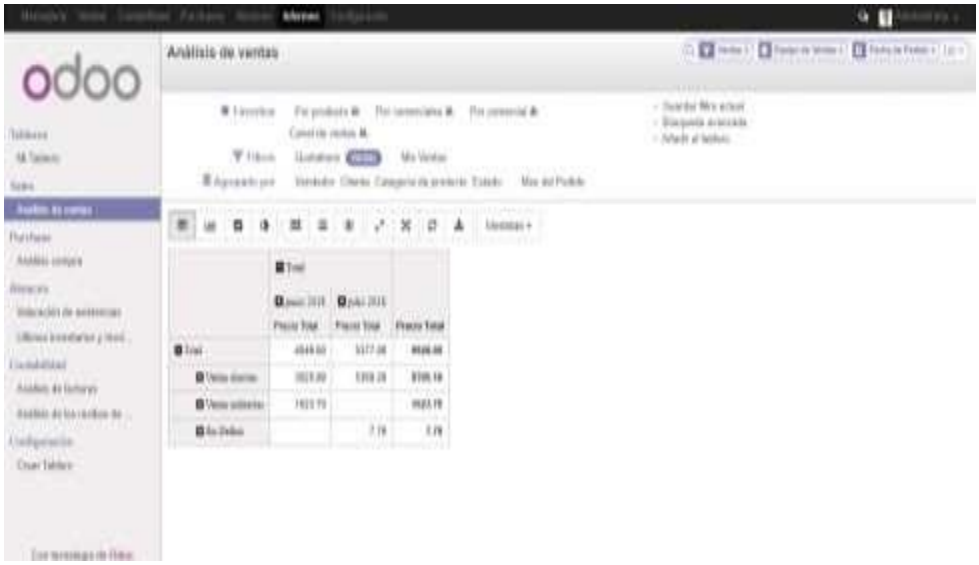


FIGURA 49: El Gerente de la librería observa el crecimiento de las ventas en tiempo real y en tablas que facilitan su trabajo.

Fuente: Elaboración propia.

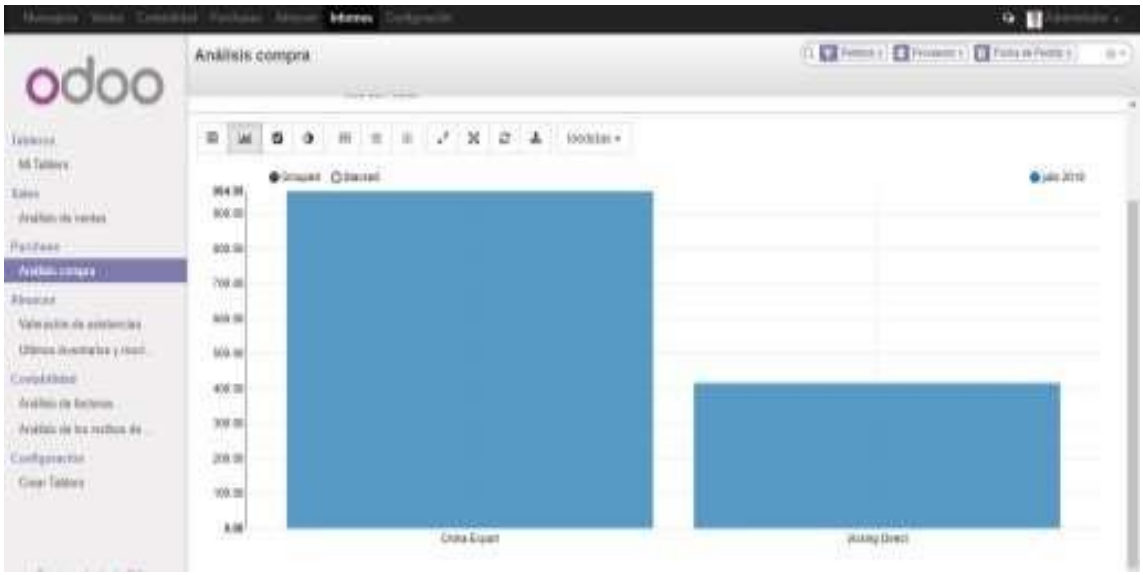


FIGURA 50: El Gerente de la librería observa el crecimiento de las compras en tiempo real y en tablas que facilitan su trabajo.

Fuente: Elaboración propia.

4.2.2 Diseño de la Organización y Procesamiento de los Datos

La acción de recolección de información de la investigación, para dichos resultados se previó estrategias de organización y procesamiento de los datos.

Los datos obtenidos se organizaron e interpretaron, según los siguientes pasos:

- Organización de los datos obtenidos, según actores y criterios para ser analizados.
- Procesamiento de los datos obtenidos utilizando técnicas estadísticas de conteo, codificación y organización con sus respectivos instrumentos como: tablas o cuadros y gráficos estadísticos.
- Las acciones de procesamiento se organizaron a través de una tabla estadística en la que se representa la frecuencia de la alternativa de los ítems propuestos en nuestra encuesta, ello luego se procesó en forma porcentual.
- Esto nos permitió el análisis e interpretación de los resultados obtenidos asumiendo las técnicas estadísticas de inferencia y de descripción cuantitativa e interpretativa.

4.2.3 Presentación de resultados:

Aquí presentamos las tablas, gráficos, figuras y otros que nos permite realizar la interpretación de los resultados, para ello se tuvo la ayuda del programa SPSS para el análisis.

a) Confiabilidad del instrumento aplicado

	N	%
Casos Válido	13	100,0
Excluido	0	,0
Total	13	100,0

Alfa de Cronbach	N de elementos
,871	13

Interpretación: Como se observa el valor para verificar la consistencia interna del instrumento el cual mediante el procedimiento Alfa de Cronbach fue de 0.871, y que representa un valor óptimo para la aplicación del instrumento y garantizar la confiabilidad de los resultados obtenidos. De acuerdo a esto, se puede afirmar existe confiabilidad y consistencia interna en la composición del cuestionario aplicado.

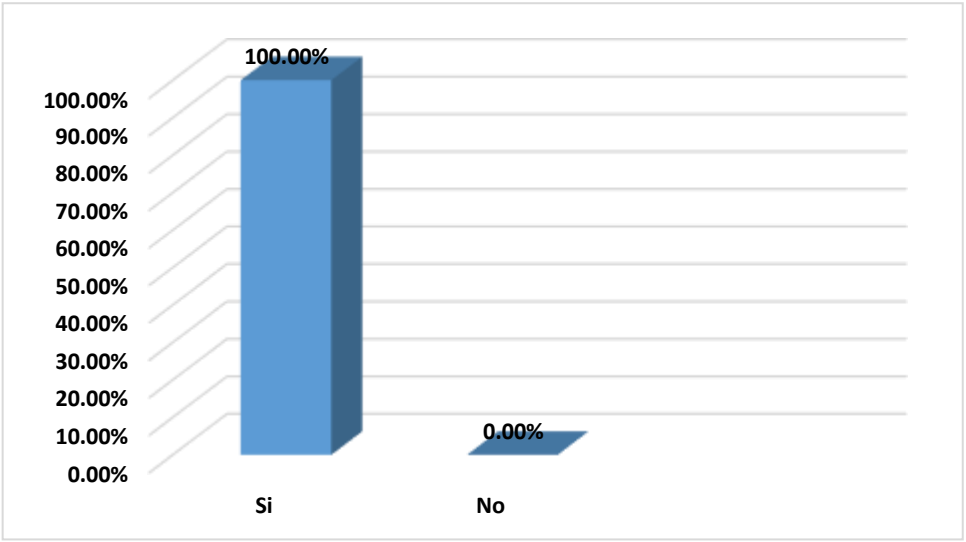
b) Resultados de la Aplicación del Pre Test - Gerente General

Cuadro Nº 1: La Librería El Virrey en la actualidad utiliza herramientas tecnológicas para manejo y gestión de información

Criterio	Nº de respuestas	Porcentaje
Si	1	100,0%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como se observa en el cuadro el porcentaje con mayor aceptación es que la Librería no utiliza diferentes herramientas tecnológicas que alcanza un 100% frente a un 0,00% menciona que no utiliza al ítem. Con esto podemos afirmar que la Librería del Virrey no brinda otras opciones para manejo de herramientas tecnológicas, dándonos a entender que es un sistema sin muchas posibilidades para la atención a los clientes.

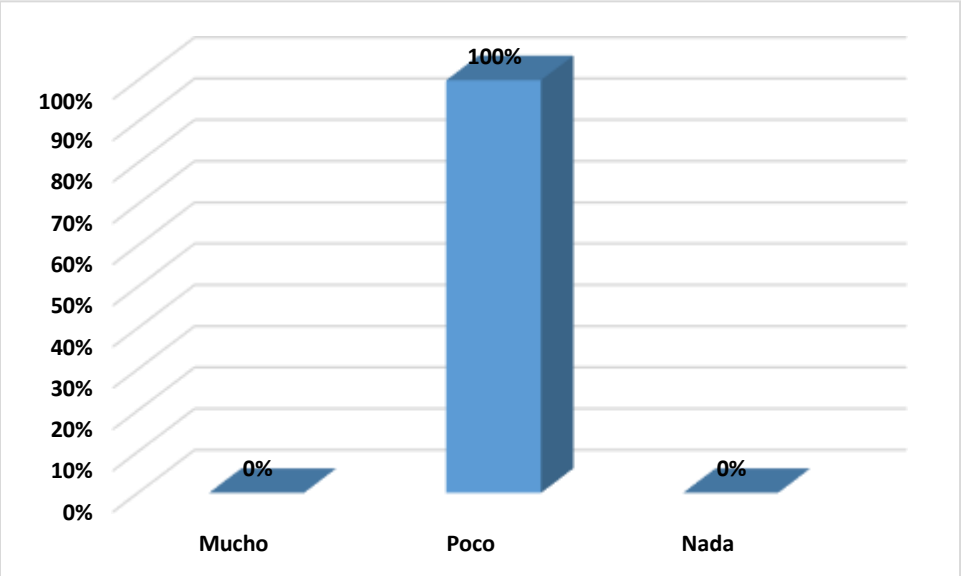


Cuadro Nº 2: Cuánto de información te brinda el sistema de la Librería del Virrey SAC para su uso

Criterio	Nº de respuestas	Porcentaje
Mucho	0	0,0%
Poco	1	100,0%
Nada	0	0.0%
Total	1	**100,0%**

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: según observamos el 0,00% de encuestados afirma que la información del sistema es mucho, seguido del 100,0% quienes afirman que la información es poco y el 0,0% mencionan que el sistema no brinda nada de información. Esto nos permite comprender que el sistema es limitado para el administrador y clientes.

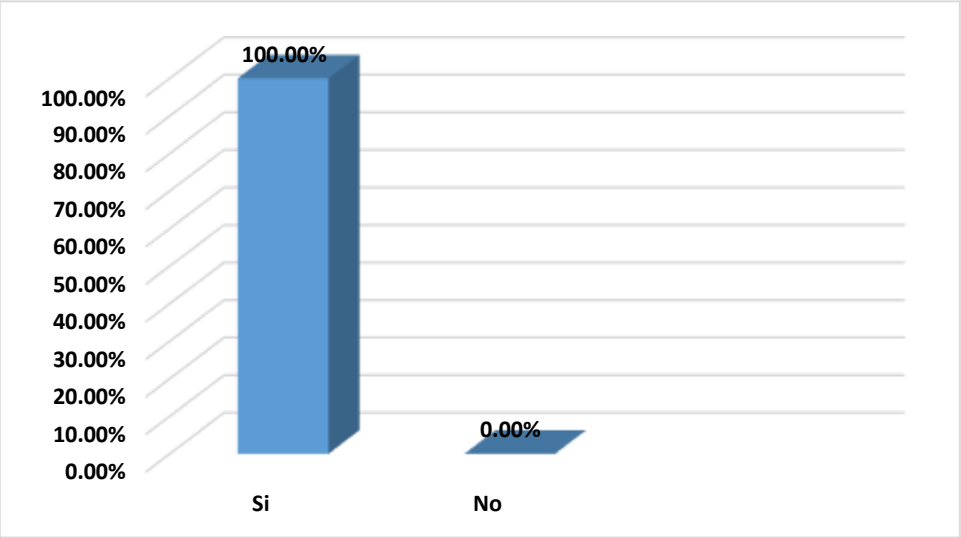


Cuadro Nº 3: Estás conforme con las herramientas de gestión actuales con la que trabaja la Librería.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,0%
No	0	0,0%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Información: Según los datos del presente cuadro el 100,0 afirma que, si está conforme con las herramientas tecnológicas, seguido del 0,00% no está conforme. Con esto podemos comprender que el entrevistado se encuentran conforme con el sistema de la Librería.

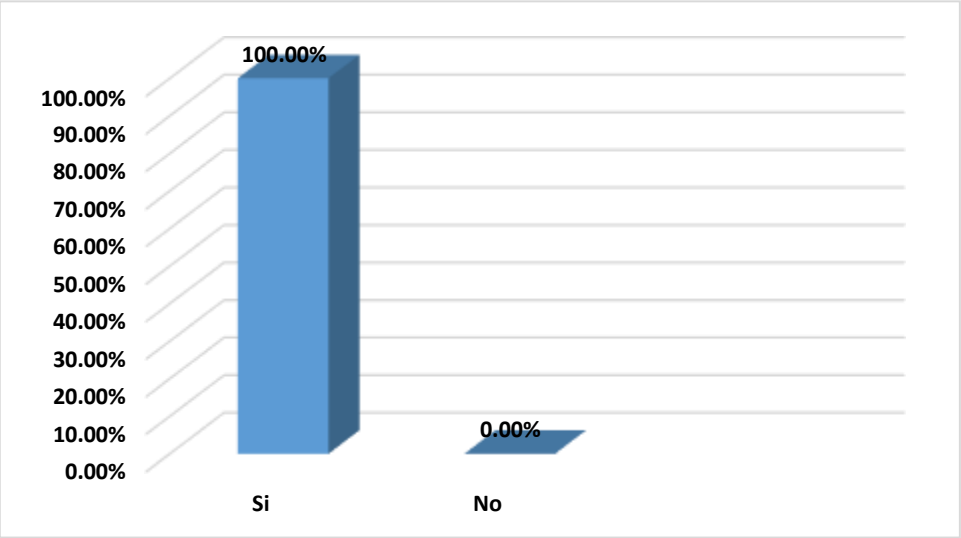


Cuadro Nº 4: El acceso a la información por el usuario al sistema es de manera fácil

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como encontramos en el cuadro el 100,00% afirma que si es fácil el uso del sistema y un 0,00% afirma que es fácil. Dándonos a entender la complejidad del sistema.

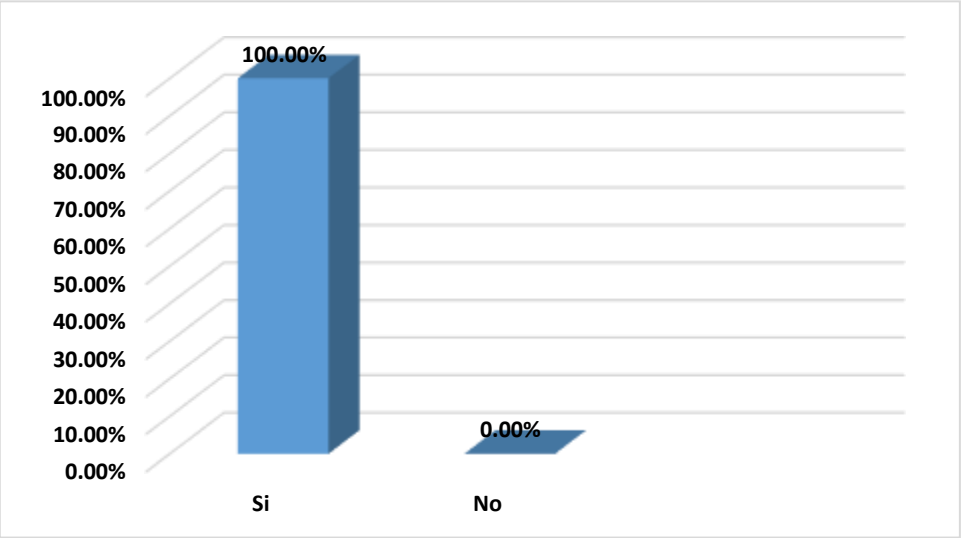


Cuadro Nº 5: Considera usted que los procesos de gestión de la Librería a su disposición.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: De acuerdo al cuadro sobre los entrevistados el 100,00% menciona que, si están integrados los procesos de la Librería y el 0,00% que no están integrados. Con ello podemos afirmar que la mayoría del encuestado considera que están integrados los procesos de la Librería.

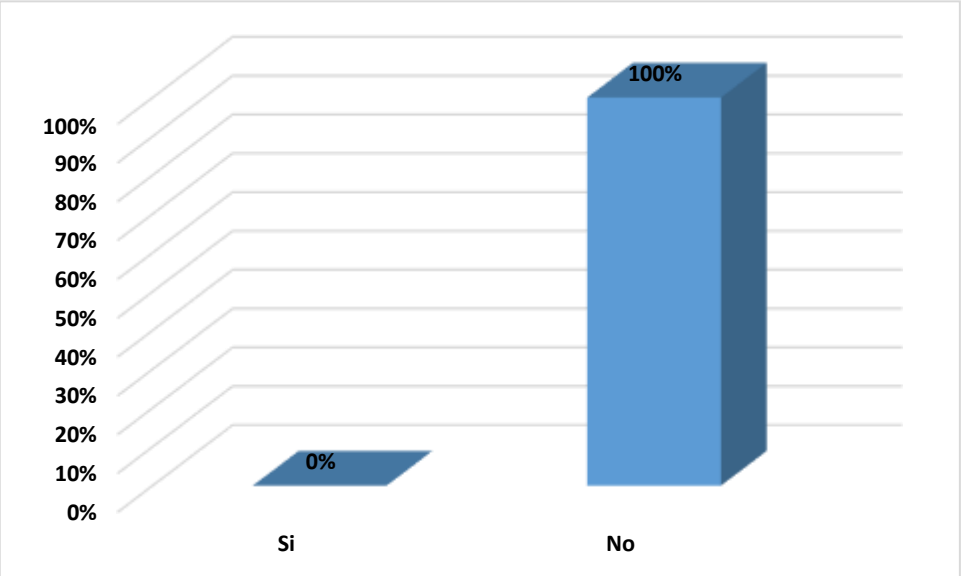


Cuadro Nº 6: Consideras que el sistema que usa la Librería del Virrey como la mejor opción para mejorar la gestión

Criterio	Nº de respuestas	Porcentaje
Si	0	0,00%
No	1	100,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: en el cuadro anterior encontramos respuestas sobre la mejora de la gestión con ayuda del sistema de la Librería, donde el 0,00% si considera la mejor opción, el 100,00% no lo considera, entendiéndose que el sistema de la Librería requiere una cambio para lograr optimizar la gestión.



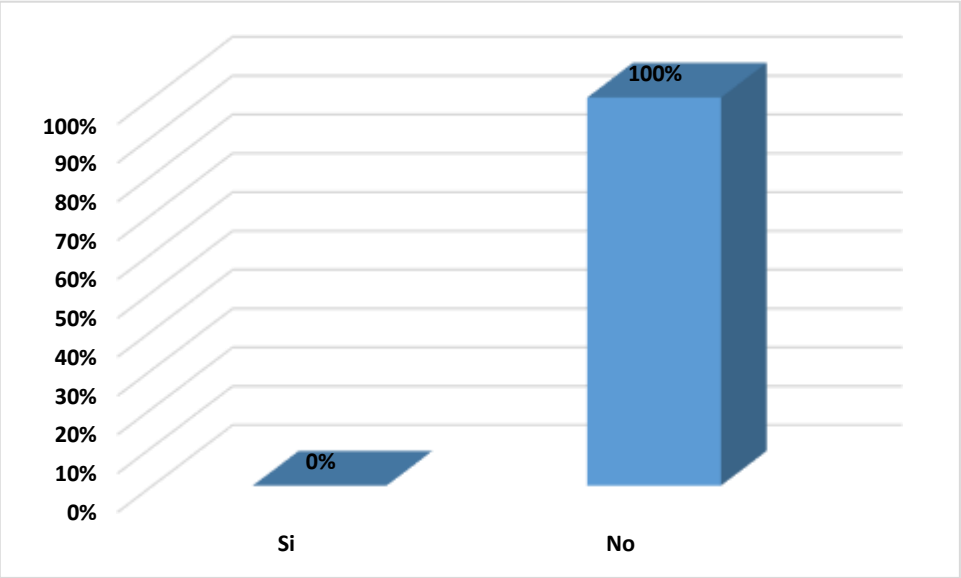
Fuente: Elaboración propia

Cuadro Nº 7: Considera que el Sistema de la Librería apoya a su persona en el manejo de la información para su gestión.

Criterio	Nº de respuestas	Porcentaje
Si	0	0,00%
No	1	100,00%
Total	1	100,00%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Observamos que en el cuadro un 0,00% si considera que el sistema de la Librería es suficiente, el 100,00% menciona que no. Surgiendo la necesidad de cambio en el sistema o del mismo para mejora de los procesos de acceso a la información.



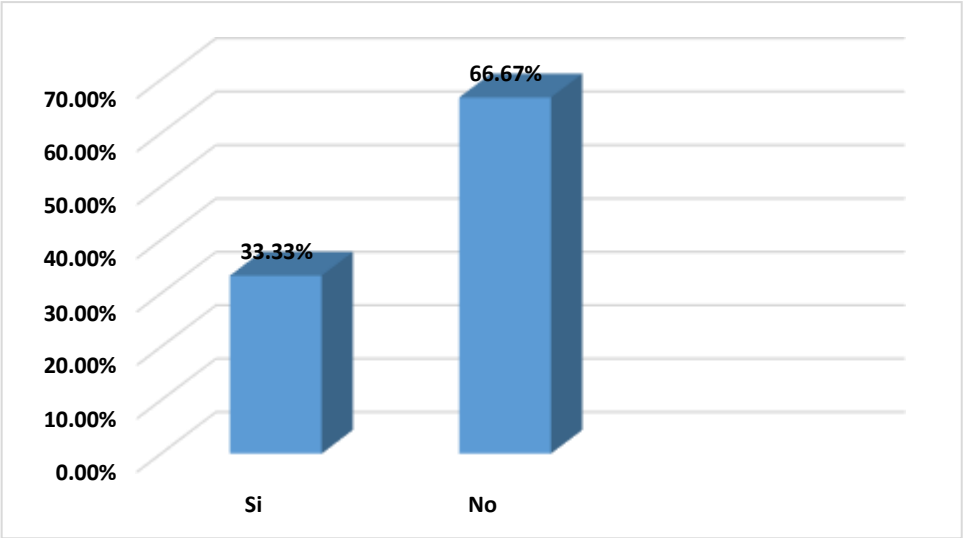
c) Resultadosde la Aplicación del Pre Test - colaboradores librería

Cuadro Nº 8: Usted cree que el sistema actual es escalable para nuevas sedes de la librería.

Criterio	Nº de respuesta	Porcentaje
Si	2	33,33%
No	4	66,67%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como se observa en el cuadro el 33,33% mencionaron que el sistema actual es escalable y el 66,67% asegura que no es escalable. Esto nos permite proponer sistemas con más característica. Con ello vemos que es necesario un cambio de verdad en el sistema de la Librería para buscar la integración del uso por los clientes.

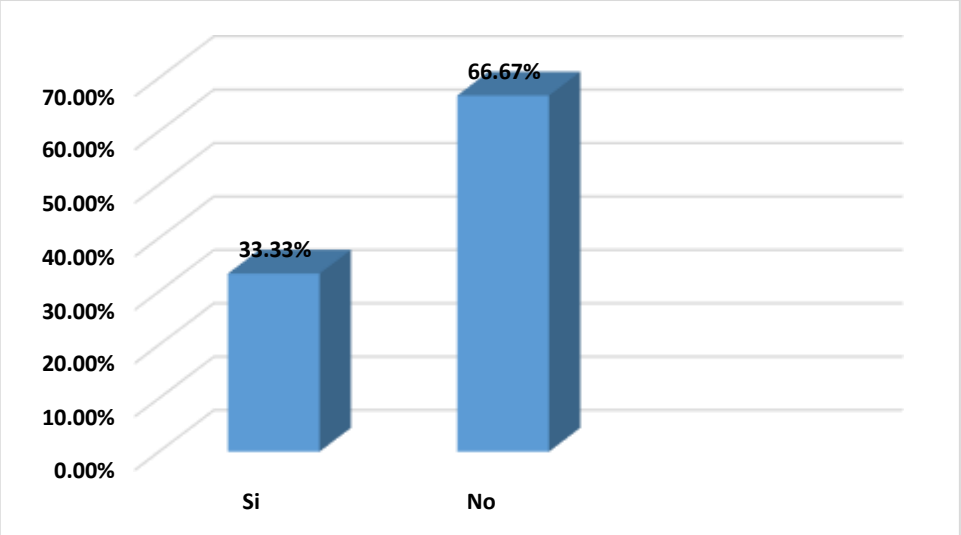


Cuadro Nº 9: Usted cree que el sistema actual optimiza la gestión de la librería.

Criterio	Nº de respuesta	Porcentaje
Si	2	33,33%
No	4	66,67%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como se observa en el cuadro el 33,33% afirmaron que el sistema actual optimiza la gestión y el 66,67% asegura que no se optimiza la gestión. Esto nos permite entender la necesidad de implementar sistemas con más opciones que facilitan el mejoramiento de la gestión.

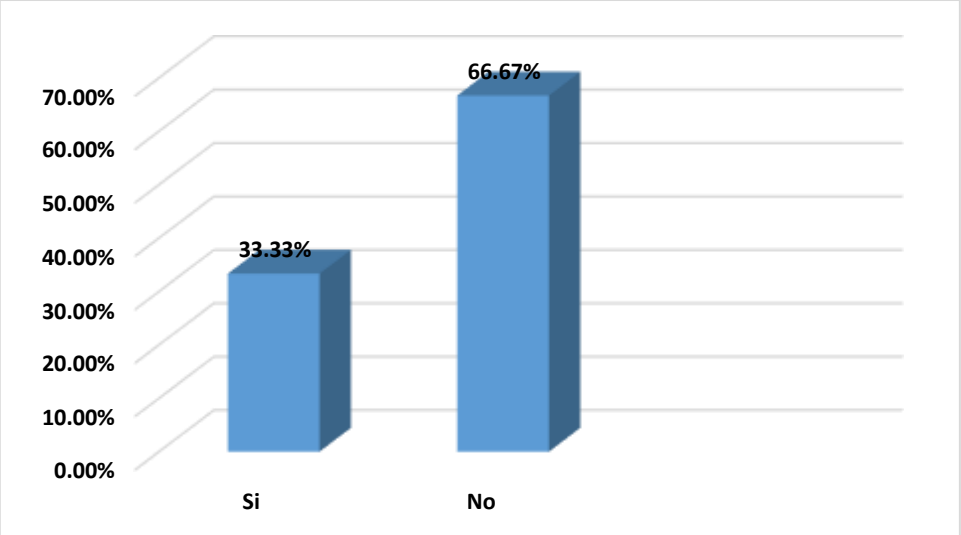


Cuadro Nº 10: Usted cree que el sistema actual es de fácil despliegue en nuevos puntos de venta.

Criterio	Nº de respuesta	Porcentaje
Si	2	33,33%
No	4	66,67%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Según observamos el 33,33% de los encuestados manifestaron que el sistema actual es de fácil despliegue y el 66,67% asegura que no es fácil. Esto nos permite encontrar la funcionalidad del sistema con más recursos para desplazarse correctamente en los puntos de venta.

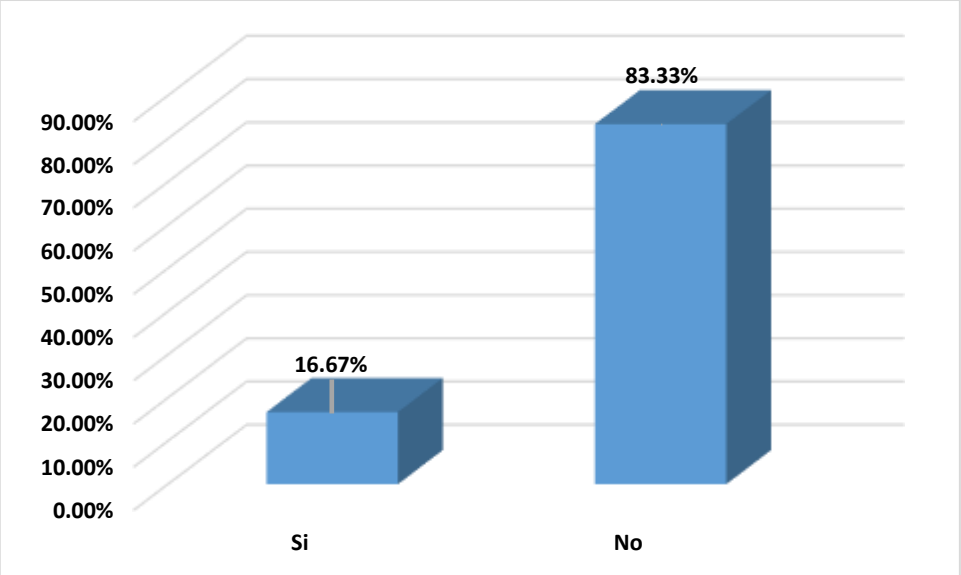


Cuadro Nº 11: Usted cree que la gestión es óptima por los despliegues del sistema.

Criterio	**º de respuesta**	**Porcentaje**
Si	1	16,67%
No	5	83,33%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como observamos en el cuadro el 16,67% tiene la gestión óptima y el 83,33% asegura que no es óptima por los despliegues. Esto nos permite comprender que este sistema tiene desventajas en ampliar su accesibilidad en distintos espacios.

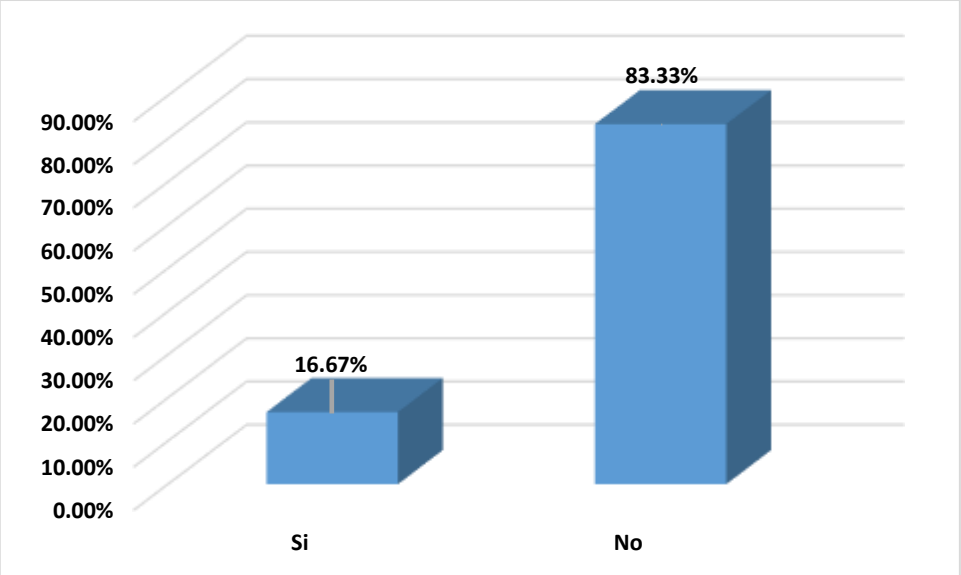


Cuadro Nº 12: Usted cree que la migración del sistema optimiza la gestión de la librería.

Criterio	Nº de respuesta	Porcentaje
Si	1	16,67%
No	5	83,33%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Se observa en el cuadro el 16,67% manifiestan que la migración del sistema optimiza la gestión y el 83,33% asegura que no cree que la migración favorece la optimización. Dándonos a comprender que es necesario implementar sistemas con muchos recursos de fácil accesibilidad en su manejo.

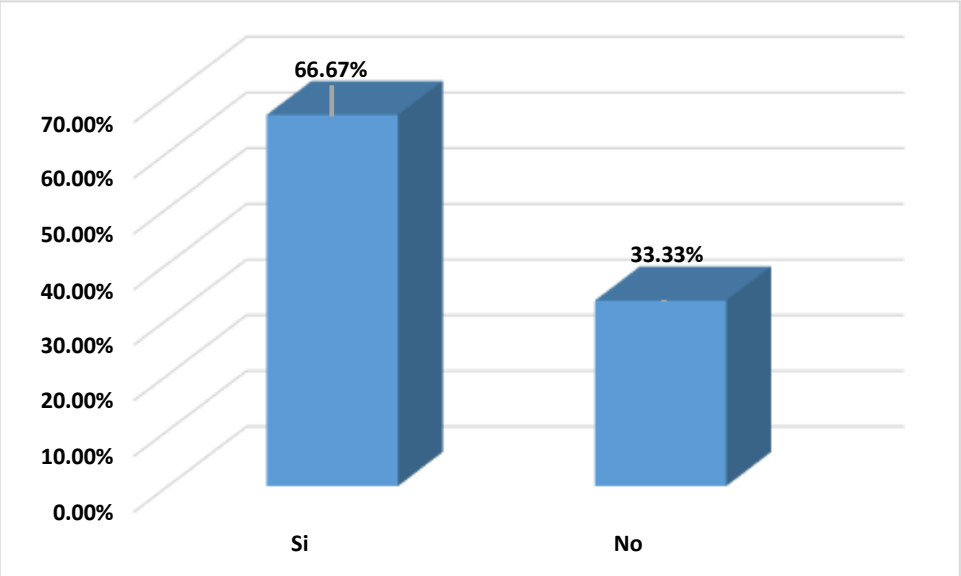


Cuadro Nº 13: Usted cree que la migración a un entorno en la nube optimizaría la gestión de la librería.

Criterio	º de respuesta	Porcentaje
Si	4	66,67%
No	2	33,33%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como se observa en el cuadro el 66,67% manifiesta que la migración a un entorno en la nube optimizaría la gestión y el 33,33% asegura que no optimizaría. Con ello vemos que es necesario un sistema de amplios recursos facilitando la accesibilidad y portabilidad del mismo.



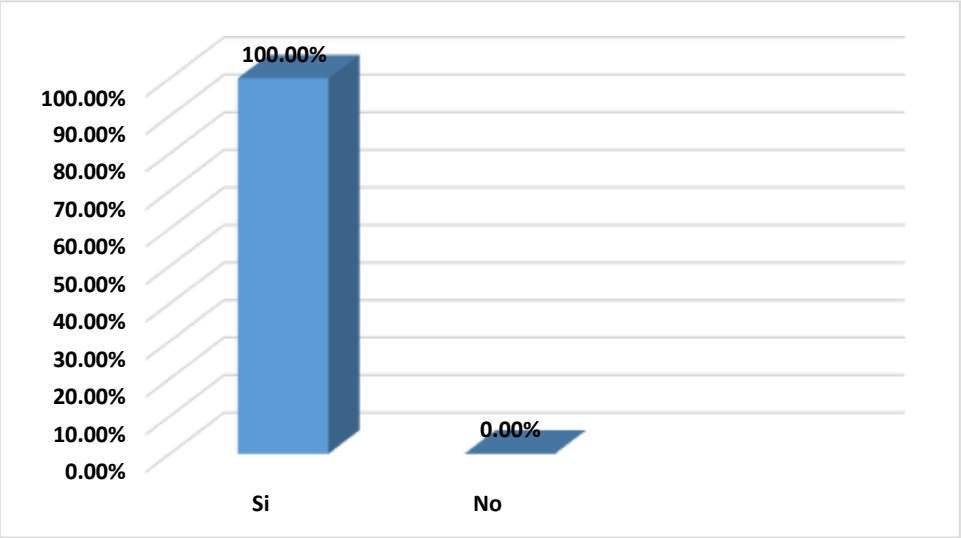
d) Resultados de la Aplicación del Pos Test - Gerente general

Cuadro Nº 14: El uso de Docker para el nuevo sistema optimiza la gestión de La Librería.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: de acuerdo con los datos obtenidos el 100,00% menciona que el uso del nuevo sistema optimiza la gestión de la librería, el 0,00% que no. Con ello podemos afirmar que es necesario la implementación del sistema Docker.

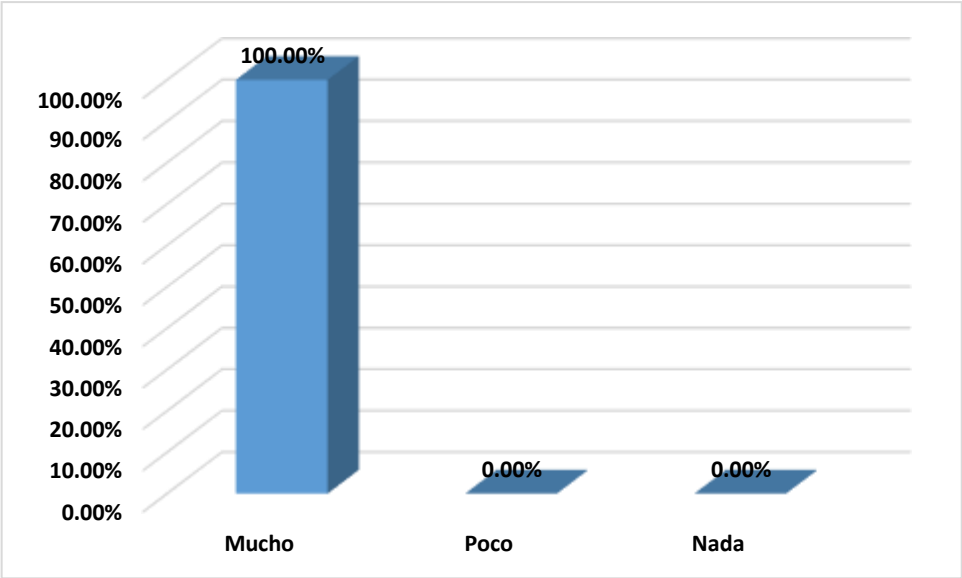


Cuadro Nº 15: La implementación del nuevo sistema con Docker-compose optimiza la gestión de la librería.

Criterio	Nº de respuestas	Porcentaje
Mucho	1	100,0%
Poco	0	0,0%
Nada	0	0.0%
Total	1	**100,0%**

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: según observamos el 100,00% afirma que la implementación del sistema Docker, seguido del 0,00% mencionan que el sistema no optimiza la gestión. Esto nos permite comprender que el nuevo sistema le brinda nuevas posibilidades en su uso con el administrador y clientes.

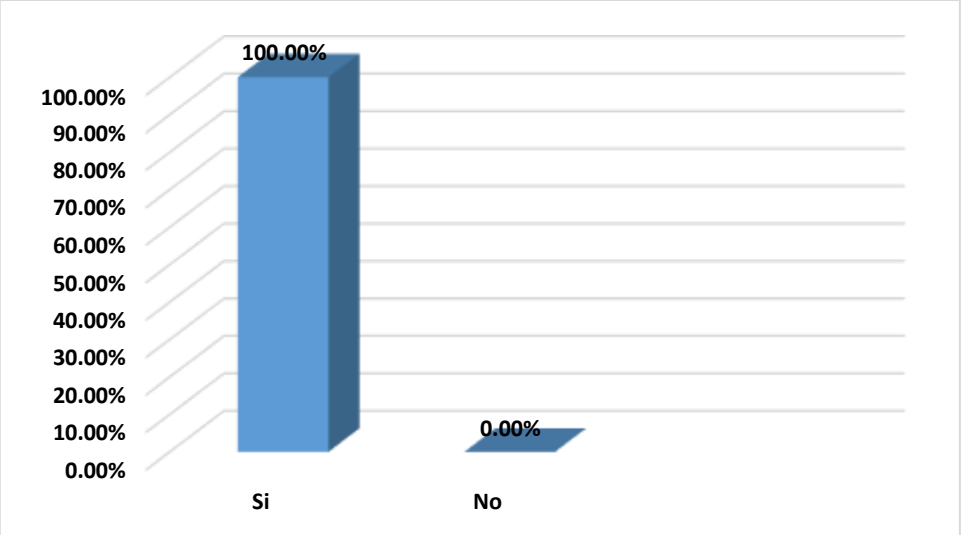


Cuadro Nº 16: El despliegue del nuevo sistema con Docker-compose apoya en la gestión de la Librería.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: como se observa en la consolidación de datos recopilados el 100,00% afirma que si está conforme con el despliegue del nuevo sistema, el 0,00% menciona que no optimiza. Esto nos permite afirmar que el sistema Docker-compose mejora la gestión de la librería.

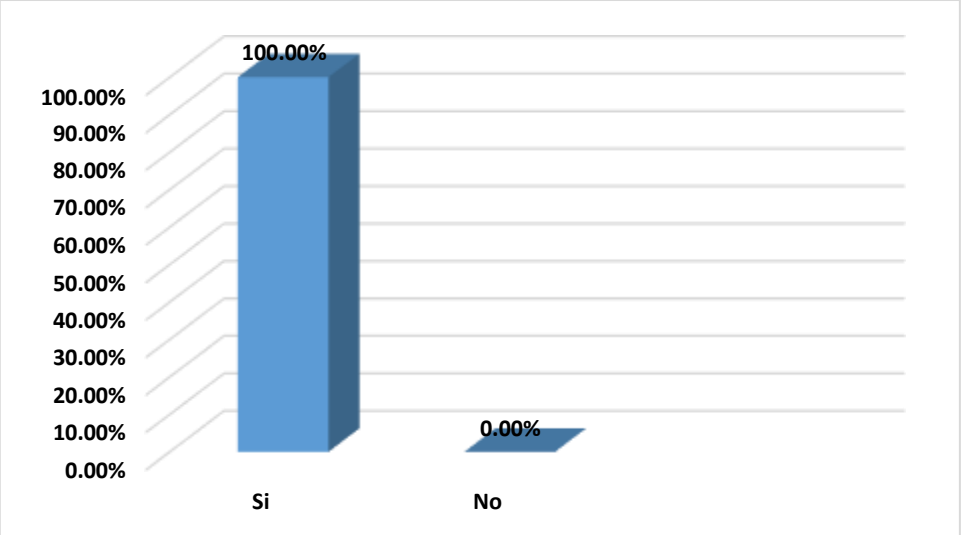


Cuadro Nº 17: La ejecución de la migración a un entorno en la nube optimiza la gestión de la librería.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Al observar el cuadro podemos interpretar que el 100,00% cree que la migración al entorno nube del sistema, el 0,00% cree que no optimiza la gestión. El sistema Docker tiene mayores herramientas para los clientes que lo hacen más fácil el acceso a la información.

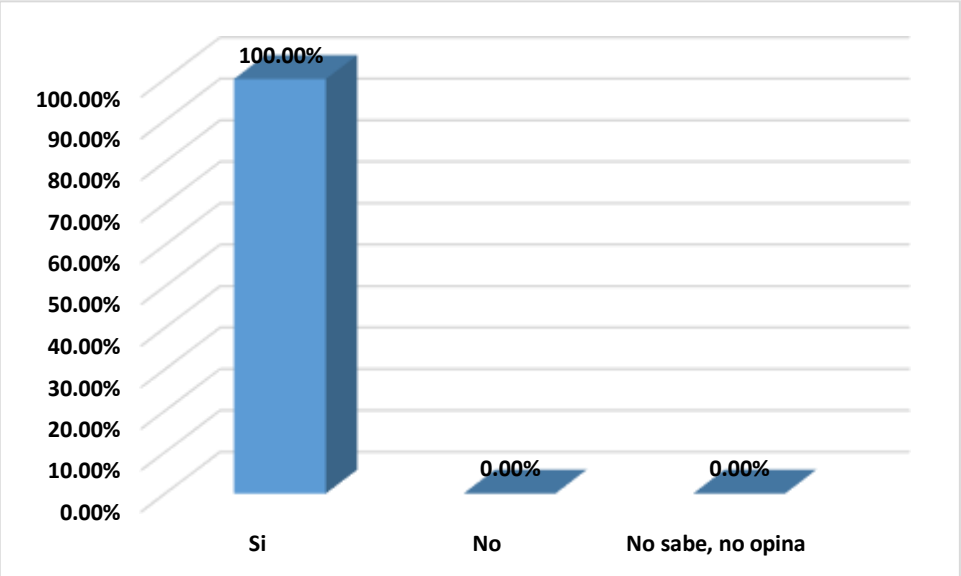


Cuadro Nº 18: Considera usted que con el nuevo sistema implementado con Docker-compose optimiza los procesos de la Librería y están integrados.

Criterio	Nº de respuestas	Porcentaje
Si	1	100%
No	0	0%
No sabe, no opina	0	0%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como encontramos en los datos que presenta el cuadro, el 100% si considera que el sistema Docker optimiza los procesos de la Librería, el 0% menciona que no y el 0% no sabe, no opina. La cantidad de recursos que ofrece el nuevo sistema permite comprender que todo está integrado.

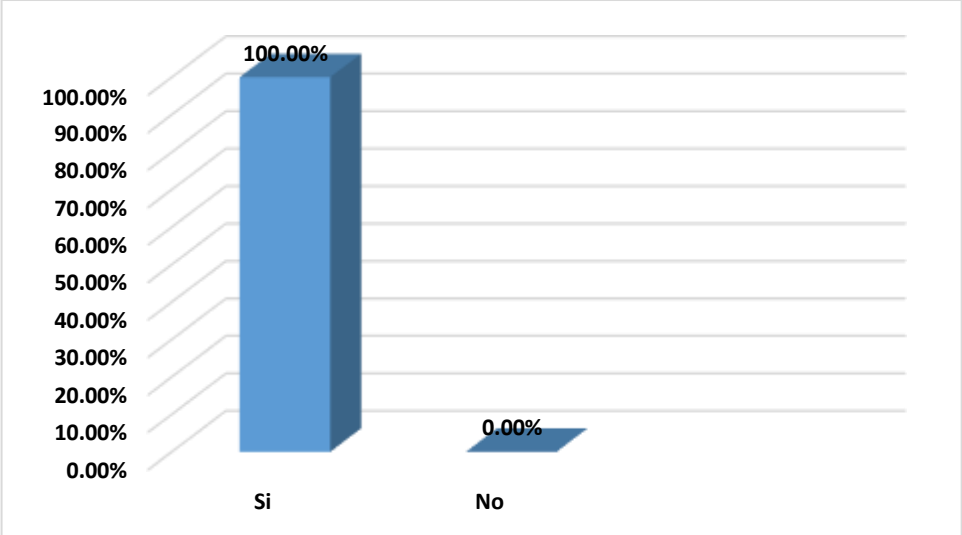


Cuadro Nº 19: Considera que el nuevo Sistema implementado con Docker-compose es suficiente para el manejo de los procesos de acceso a la información.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: en el cuadro anterior encontramos respuestas frente al considerar el nuevo sistema de la Librería como la mejor opción, un 100,00% afirma que sí con toda seguridad, 0,00% no considera, entendiéndose que el nuevo sistema implementado en la Librería busca optimizar la gestión.



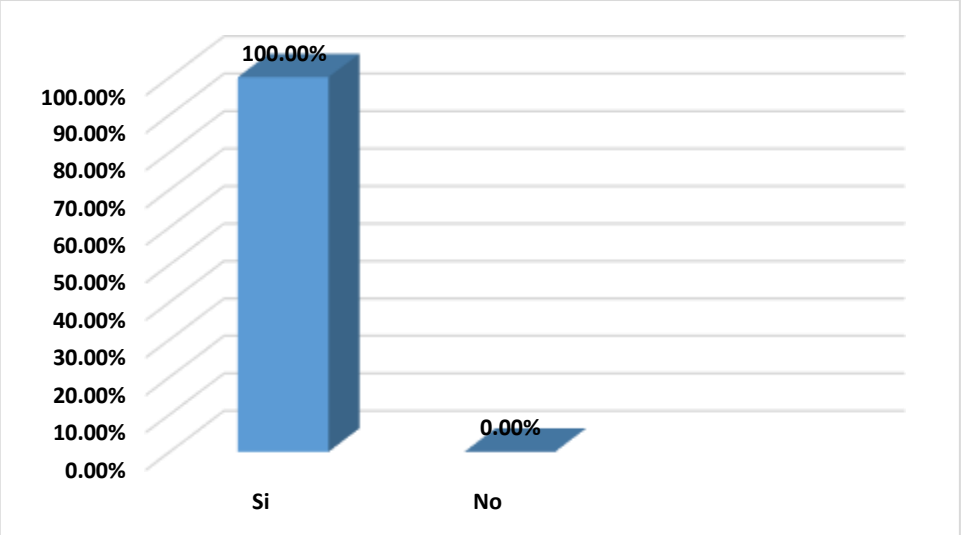
Fuente: Elaboración propia

Cuadro Nº 20: Cree que los cambios que se han dado con el nuevo sistema Docker con Odoo desde su implementación y funcionamiento han sido beneficiosos.

Criterio	Nº de respuestas	Porcentaje
Si	1	100,00%
No	0	0,00%
Total	1	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: De acuerdo a los datos del cuadro el 100,00% si considera que el nuevo sistema ha sido beneficioso, el 0,00% dice que no. Esto evidencia el cambio significativo la implementación del sistema Docker que es percibido de manera inmediata por la gerencia y los clientes de la Librería.



e) Resultados de la Aplicación del Post Test - Colaboradores librería

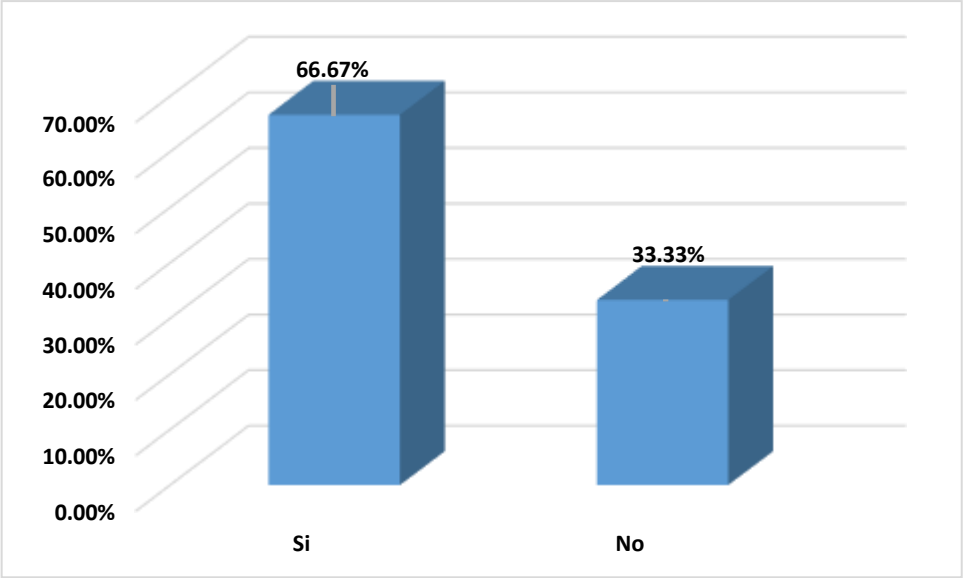
Cuadro Nº 21: Usted cree que Docker-compose contribuye a la escalabilidad del nuevo sistema de la librería.

Criterio	Nº de respuestas	Porcentaje
Si	4	66,67%
No	2	33,33%
Total	6	100,00%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: como se observa en el cuadro el 66,67% cree que el sistema con Docker favorece la escalabilidad para la Librería, el 33,33% no cree. Los beneficios que ofrece el sistema Docker Odoo es observable y manejable facilitando

la interacción con el cliente haciéndolo más cómodo en el manejo de información.

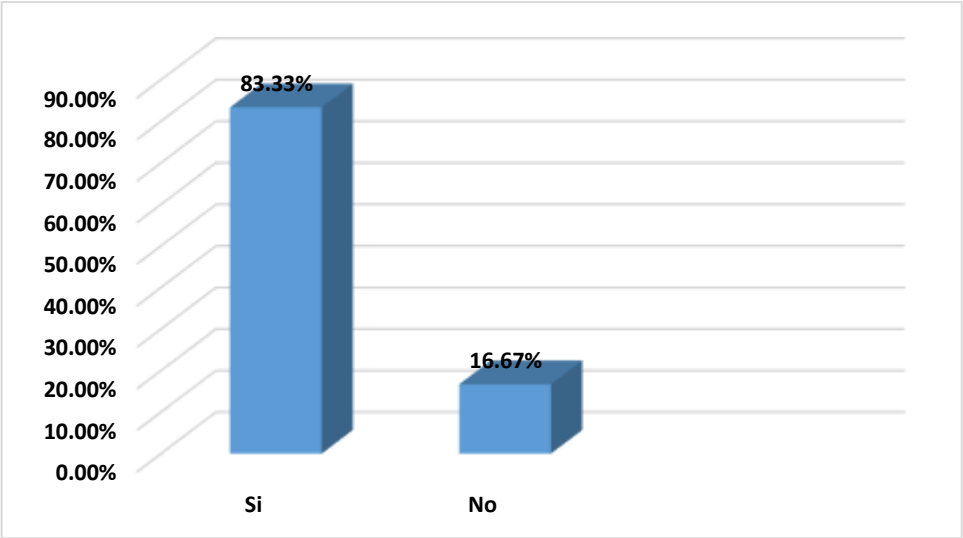


Cuadro Nº 22: Usted cree que la implementación del Docker-compose optimiza la gestión de la librería.

Criterio	Nº de respuesta	Porcentaje
Si	5	83,33%
No	1	16,67%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como se observa en el cuadro el 83,33% afirmaron que creen en la implementación del sistema actual optimiza la gestión y el 16,67% asegura que no se optimiza la gestión. Esto nos permite entender la necesidad de implementar sistemas con más opciones como el Docker que facilitan el mejoramiento de la gestión.

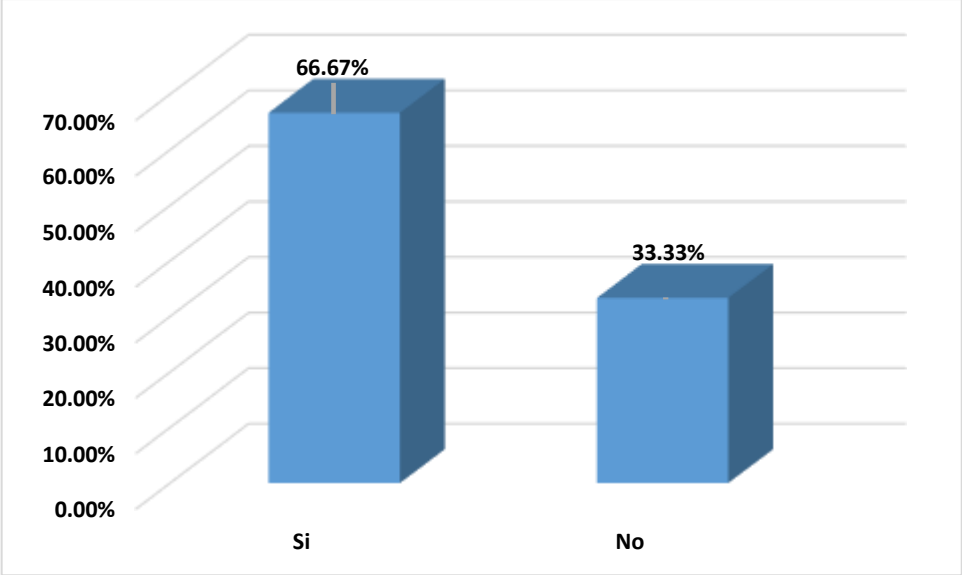


Cuadro Nº 23: Usted cree que Docker-compose facilita el despliegue en nuevos puntos de venta.

Criterio	Nº de respuesta	Porcentaje
Si	5	66,67%
No	1	33,33%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Como observamos el 66,67% de los encuestados manifestaron que el sistema actual es de fácil despliegue y el 33,33% asegura que no es fácil. Esto nos permite encontrar la funcionalidad del sistema con más recursos para desplazarse correctamente en los puntos de venta.

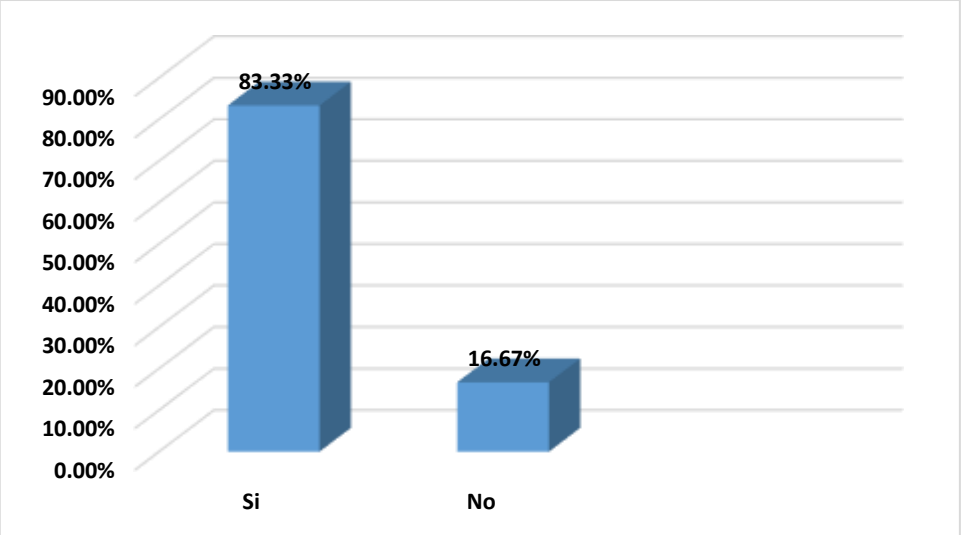


Cuadro Nº 24: Usted cree que Docker-compose facilita la migración del sistema a un entorno en la nube.

Criterio	Nº de respuesta	Porcentaje
Si	5	83,33%
No	1	16,67%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Tal como observamos en el cuadro el 83,33% tiene la gestión óptima y el 16,67% asegura que no facilita la migración en la nube. Esto nos permite comprender que este sistema Docker tiene ventajas en ampliar su accesibilidad en distintos espacios.

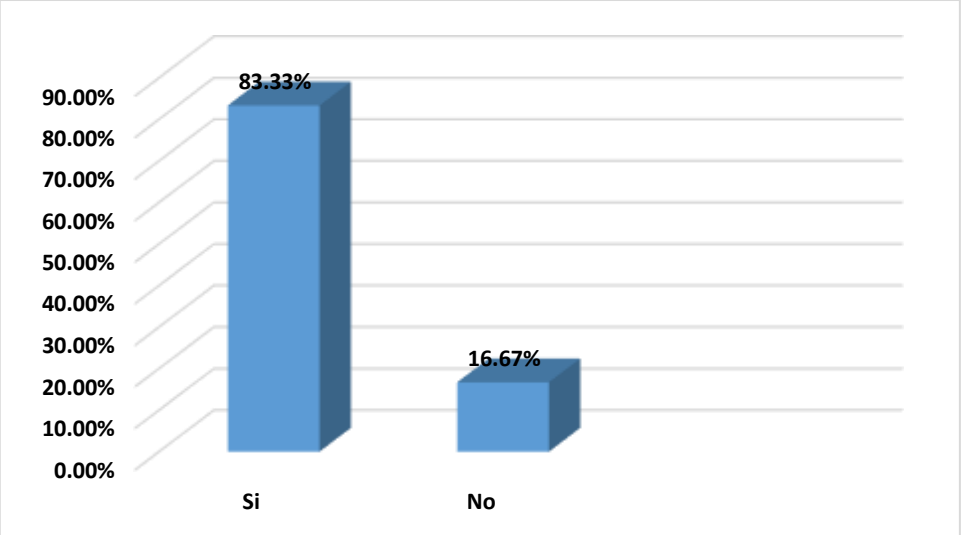


Cuadro Nº 25: Usted cree que Docker optimiza la gestión de procesos en su área.

Criterio	Nº de respuesta	Porcentaje
Si	5	83,33%
No	1	16,67%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Se observa que el 83,33% manifiestan que Docker optimiza la gestión y el 16,67% asegura que no favorece la optimización. Por lo que es necesario implementar sistemas con muchos recursos para el mejoramiento de los procesos de las áreas de la librería.

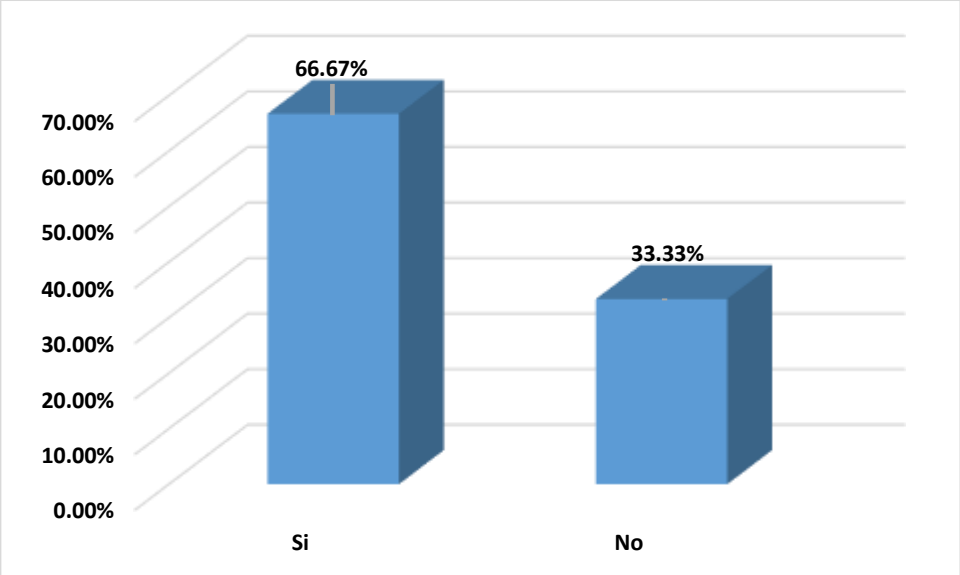


Cuadro Nº 26: Usted cree que Docker y Odoo son herramientas de fácil aprendizaje para su implementación.

Criterio	Nº de respuesta	Porcentaje
Si	4	66,67%
No	2	33,33%
Total	6	100,0%

Fuente: Encuesta abril 2018. Elaboración propia.

Interpretación: Según el cuadro de los resultados de los encuestados el 66,67% manifiesta que Docker y Odoo optimizaría la gestión por su fácil implementación y uso y el 33,33% asegura que no son fáciles. Con ello vemos que es necesario una capacitación sobre el manejo del sistema Docker donde se pueda aprovechar la accesibilidad y portabilidad del mismo.



4.3 Prueba de hipótesis

4.3.1 Prueba de Hipótesis.

Hipótesis Alterna (H1)

El análisis de Docker utilizando Odoo, optimiza la gestión de la Librería del Virrey SAC.

Hipótesis Nula (Ho)

El análisis de Docker utilizando Odoo, no optimiza la gestión de la Librería del Virrey SAC

4.3.2 Hipótesis estadística

$H_{i=r} \geq 0.5$

$H_o = r < 0.5$

$\alpha = 0.05$

Denota:

Hi: El índice de correlación entre las variables es mayor o igual a 0.5

Ho: El índice de correlación entre las variables es menor a 0.5

El valor de significancia está asociado al valor α = 0.05

Determinación de la zona de rechazo de la hipótesis nula

Nivel de confianza al 95%; Valor de significancia: α = 0.05

Resultados:

			Análisis de Docker utilizando Odoo	Gestión de la Librería del Virrey SAC
	Análisis de Docker utilizando Odoo	**Coeficiente de correlación (bilateral** **Sig.**	1,000 13	0,86 0,793 13
	Gestión de la Librería del Virrey SAC	**Coeficiente de correlación** **Sig. (bilateral)**	0,86 0,793 13	1,000 13

Fuente: Elaboración propia.

En el cuadro precedente podemos observar que p – valor = 0,86 ($p < 0.5$), por lo tanto, rechazamos la hipótesis nula y podemos decir que el análisis de Docker utilizando Odoo, optimiza la gestión de la Librería del Virrey SAC

4.4 Discusión de resultados.

- El instrumento de medición aplicado (a = 0.871) presenta una buena Confiabilidad y cada uno de sus ítems muestra consistencia interna, lo cual nos permite afirmar que el instrumento de 13 ítems tiene confiabilidad.
- Cada uno de los ítems del instrumento están estrechamente vinculados y la validación empírica nos dice que hay unicidad del mismo y que cada uno de sus ítems buscan la medición de las variables, es decir, que existe unicidad de los ítems. Por lo tanto, podemos afirmar que el instrumento sí presenta validez.
- Los colaboradores están convencidos que la utilización de herramientas tecnológicas para despliegues con Docker implementando Odoo han fortalecido enormemente la gestión de la Librería del Virrey debió a que la información se obtiene en tiempo real.
- Los colaboradores confirman que la satisfacción por la facilidad de aprendizaje de la utilización de Docker y manifiestan que un buen servicio es un factor importante para la mejora de la gestión en su área.
- Los colaboradores confirman que la migración de Odoo de un servidor físico a la nube es muy fácil y ayudará en la gestión de la librería debido

a que se podrá gestionar nuevas cedes con Docker utilizando Odoo.

- Los colaboradores confirman que, al utilizar Docker, Odoo es mas escalable para nuevos puntos de venta de la librería.

- El Gerente de la librería confirma que la información que observa con el sistema Odoo que fue implementado con Docker lo ayuda en la toma de decisiones que es lo más importante para su gerencia.

- El Gerente confirma que utilizar Docker ayuda a la gestión de la librería debido a que la apertura de nuevo puntos de venta lo cual es una gran ayuda para la librería.

CONCLUSIONES

Al concluir el proceso investigativo asumimos las siguientes conclusiones:

- Ha quedado comprobado que existe relación significativa entre el análisis de Docker Odoo y la gestión óptima de la Librería del Virrey; es decir, a través de Docker Odoo se mejoran los servicios de la librería El Virrey SAC incrementando su calidad de servicio, cantidad de clientes y su productividad.

- El Docker-compose contribuye a la ejecución de aplicaciones que fortalecen y optimizan las funciones y servicios implantados con Odoo.
- Docker-compose permite utilizar varias imágenes y comunicarlas, para obtener los requisitos necesarios para hacer funcionar Odoo correctamente además incorpora una base de datos que fortalece la administración de la información importante para la Librería.
- La migración de Odoo transfiere los datos de forma simplificada para una mejor gestión de la información ya que son necesarios para la toma de decisiones para gerencia de la Librería.

- Finalmente se concluye que Docker y Odoo son herramientas que fortalecen la gestión de la librería debido a que dichas herramientas incrementaron la calidad de servicio que ofrece cada día la Librería.

RECOMENDACIONES

Hacemos llegar las siguientes recomendaciones:

1. **A las diversas empresas**

   - Se adopte una política responsable y con mirada holística para implementar y desarrollar los servicios que brindan, haciendo uso de Docker y sus aplicaciones afines para lograr su óptimo funcionamiento.
   - Se debe realizar talleres Virtuales de Capacitación de Docker, a través de la red, para que todos los empresarios puedan optimizar sus servicios globales.

2. **A los estudiantes**

   Sugerimos que incorporen como hábito profesional la utilización de tecnologías sistémicas para optimizar el trabajo, que realicen prototipos de empresas modelos incorporando tecnologías y que se capaciten permanentemente, considerando que nos encontramos en la Sociedad de la información.

BIBLIOGRAFÍA

1) BAIRE, A. (2018). *Docker Tutorial.* France: UMR IRISA.

2) CUESTA , B., & GONZALES, S. (2016). *Introducción a Docker.* España: Open Canarias S.L.

3) DEVELOPERS, S. (2017). *docker Documentation.* Release.

4) FLOYD, M. (2016). *Docker From Code to Container.* SUMOLOGIC.

5) GOASGUEN, S. (2016). *Docker Cookbook.* USA: O'Reilly.

6) GÓMEZ DE LA TORRE, M. (2016). *Gestión de Contenedores Docker.* ASIR.

7) GONZÁLEZ RODRÍGUEZ, A. (2017). *Docker: guía práctica.* España: RC Libros.

8) Mouat, A. (2015). *Using Docker.* Boston: O'Reilly.

9) PETAZZONI, J. (2015). *Introduction to Docker.* Docker Inc.

10) RODRIGUEZ GAYOSO, R. (2017). *Recetas Docker.* GR.

11) S P Polyakov1, A. P. (2018). Docker Container Manager: A Simple Toolkit for Isolated Work with Shared Computational, Storage, and Network Resources. Computer Simulations in Physics and beyond.

12) TURNBULL, J. (2014). The Docker book.

LINKOGRAFÍA

1. (Docker, 2018, págs. https://www.docker.com/what-docker)
2. (Docker, Documentacion Docker, 2018, pág. https://docs.docker.com/)
3. (DockerHub, 2016, pág. https://hub.docker.com/)
4. (Dockerdocs, 2018, págs. https://docs.docker.com/get-started/)
5. (Wikihow, s.f., págs. http://es.wikihow.com/escribir-una-tesis)
6. (Sabino Carlos, 1994, pág. http://www.sld.cu/galerias/pdf/sitios/bmn/como_hacer_una_tesis.pdf.
7. (UPHM, págs. http://uphm.edu.mx/manuales/Manual-paraelaboracion- de-tesis-y-trabajos-de-investigacion.pdf)
8. (WordPress, s.f., págs. https://javierjeronimo.es/2014/11/15/kubernetes-cluster- serviciosdocker-facil/)
9. (Digital Ocean, 2015, págs. https://www.digitalocean.com/community/tutorials/el-ecosistema- dedocker-una-introduccion-a-los-componentes-mas-comunes-es)

ANEXOS

ANEXO Nº 1

Matriz de Consistencia

ANÁLISIS DE DOCKER UTILIZANDO ODOO, PARA OPTIMIZAR LA GESTIÓN EN LA LIBRERIA DEL VIRREY SAC - LIMA					
PROBLEMA GENERAL	***OBJETIVO GENERAL***	***HIPÓTESIS GENERAL***	***VARIABLE INDEPENDIENTE***	***TIPO***	***POBLACIÓN***
¿De qué manera el análisis de Docker utilizando Odoo, optimizará la gestión de la Librería del Virrey SAC?	Determinar si el análisis Docker utilizando Odoo, optimizará la gestión de la Librería del Virrey SAC.	El análisis de Docker utilizando Odoo, optimiza la gestión de la Librería del Virrey SAC.	Análisis Docker utilizando Odoo	Explicativo de enfoque cuantitativo	La población fue de 13 personas las cuales 12 son colaboradores y 1 el gerente de la Librería del Virrey SAC
PROBLEMAS ESPECIFICOS	***OBJETIVOS ESPECÍFICOS***	***HIPÓTESIS ESPECÍFICOS***	***VARIABLE DEPENDIENTE***	***DISEÑO***	***MUESTRA***
¿Cómo la implementación del Docker-compose para la escalabilidad de Odoo, optimizará en la gestión de la Librería del Virrey SAC? ¿De qué manera Docker compose contribuirá en los despliegues de Odoo, en la gestión de la Librería del Virrey SAC.? ¿Cómo la ejecución de la migración de Odoo de un servidor físico en la nube optimizará la gestión de la Librería del Virrey SAC?	Verificar si la Implementación Docker-compose para la escalabilidad de Odoo optimizará la gestión en la Librería del Virrey SAC. Determinar si Docker-compose contribuirá en los despliegues de Odoo, en la gestión de la Librería del Virrey SAC. Verificar si la ejecución de la migración de Odoo optimizará la gestión de la Librería del Virrey SAC.	La implementación del Docker compose para la escalabilidad de Odoo optimiza la gestión a la Librería del Virrey SAC. El Docker-compose contribuye en los despliegues de Odoo, en la gestión de la Librería del Virrey SAC. La ejecución de la migración de Odoo optimiza la gestión de la Librería del Virrey SAC.	Optimizar la gestión de la Librería del Virrey SAC	Cuasi experimental	Para el presente estudio a realizar se consideraron 7 personas donde 6 son colaboradores y 1 el gerente de la Librería del Virrey SAC, el cual permitirá la recolección de datos.

ANEXO Nº 2
Instrumentos de Recolección de datos

UNIVERSIDAD NACIONAL DANIEL ALCIDES CARRIÓN

Facultad de Ingeniería

Escuela de Ingeniería de Sistemas y Computación

"***ANÁLISIS DE DOCKER UTILIZANDO ODOO, PARA OPTIMIZAR LA GESTIÓN EN LA LIBRERIA DEL VIRREY SAC - LIMA***"

Encuesta 1 (PRE TEST – Para el Gerente General)

INSTRUCCIONES: Marque una alternativa con las que más se identifica:

1) La Librería El Virrey en la actualidad utiliza herramientas tecnológicas para manejo y gestión de información

a) Si ( )

b) No ( )

2) Cuánto de información te brinda el sistema de la Librería del Virrey SAC para su uso

a) Mucho ( )

b) Poco ( )

c) Nada ( )

3) Estás conforme con las herramientas de gestión actuales con la que trabaja la Librería.

a) Si ( )

b) No ( )

4) El acceso a la información para usted al sistema es de manera fácil.

a) Si ( )

b) No ( )

5) Considera usted que los procesos de gestión de la Librería a su disposición.

a) Si ( )

b) No ( )

6) Consideras que el sistema que usa la Librería del Virrey como la mejor opción para mejorar la gestión.

a) Si ( )

b) No ( )

7) Considera que el Sistema de la Librería apoya a su persona en el manejo de la información para su gestión.

a) Si ( )

b) No ( )

ANEXO Nº 3

UNIVERSIDAD NACIONAL DANIEL ALCIDES CARRIÓN

Facultad de Ingeniería

Escuela de Ingeniería de Sistemas y Computación

"***ANÁLISIS DE DOCKER UTILIZANDO ODOO, PARA OPTIMIZAR***

LA GESTIÓN EN LA LIBRERIA DEL VIRREY SAC - LIMA"

Encuesta 2 (PRE TEST – Para los colaboradores de la librería)

INSTRUCCIONES: Marque una alternativa con las que más se identifica:

1. Usted cree que el sistema actual es escalable para nuevas sedes de la librería

    a. Si ( )

    b. No ( )

2. Usted cree que el sistema actual optimiza la gestión de la librería

    a) Si ( )

    b) No ( )

3. Usted cree que el sistema actual es de fácil despliegue en nuevos puntos de venta

    a) Si ( )

    b) No ( )

4. Usted cree que la gestión es óptima por los despliegues del sistema

    a) Si ( )

    b) No ( )

5. Usted cree que la migración del sistema optimiza la gestión de la librería

    a) Si ( )

    b) No ( )

6. Usted cree que la migración a un entorno en la nube optimizaría la gestión de la librería

    a) Si ( )

    b) No ( )

ANEXO Nº 4

UNIVERSIDAD NACIONAL DANIEL ALCIDES CARRIÓN

Facultad de Ingeniería

Escuela de Ingeniería de Sistemas y Computación

"***ANÁLISIS DE DOCKER UTILIZANDO ODOO, PARA OPTIMIZAR***

LA GESTIÓN EN LA LIBRERIA DEL VIRREY SAC - LIMA"

Encuesta 1 (POS TEST – Para el Gerente General)

INSTRUCCIONES: Marque una alternativa con las que más se identifica:

1) El uso de Docker para el nuevo sistema optimiza la gestión de La Librería.
    a) Si ( )
    b) No ( )

2) La implementación del nuevo sistema con Docker-compose optimiza la gestión de la librería
    a) Mucho ( )
    b) Poco ( )
    c) Nada ( )

3) El despliegue del nuevo sistema con Docker-compose apoya en la gestión de la Librería.
    a) Si ( )

b) No ( )

4) La ejecución de la migración a un entorno en la nube optimiza la gestión de la librería.

a) Si ( )

b) No ( )

5) Considera usted que con el nuevo sistema implementado con Docker-compose optimiza los procesos de la Librería y están integrados

a) Si ( )

b) No ( )

c) No sabe, no opina ( )

6) Considera que el nuevo Sistema implementado con Docker- compose es suficiente para el manejo de los procesos de acceso a la información.

a) Si ( )

b) No ( )

7) Cree que los cambios que se han dado con el nuevo sistema Docker con Odoo desde su implementación y funcionamiento han sido beneficiosos.

a) Si ( )

b) No ( )

ANEXO Nº 5

UNIVERSIDAD NACIONAL DANIEL ALCIDES CARRIÓN

Facultad de Ingeniería

Escuela de Ingeniería de Sistemas y Computación

"***ANÁLISIS DE DOCKER UTILIZANDO ODOO, PARA OPTIMIZAR***

LA GESTIÓN EN LA LIBRERIA DEL VIRREY SAC - LIMA"

Encuesta 2 (POS TEST – Para los colaboradores de la librería)

INSTRUCCIONES: Marque una alternativa con las que más se identifica:

1. Usted cree que Docker-compose contribuye a la escalabilidad del nuevo sistema de la librería
   a. Si ( )
   b. No ( )
2. Usted cree que la implementación del Docker-compose optimiza la gestión de la librería
   a. Si ( )
   b. No ( )
3. Usted cree que Docker-compose facilita el despliegue en nuevos puntos de venta
   a. Si ( )
   b. No ( )

4. Usted cree que Docker-compose facilita la migración del sistema a un entorno en la nube
    a. Si ( )
    b. No ( )
5. Usted cree que Docker optimiza la gestión de procesos en su área
    a. Si ( )
    b. No ( )
6. Usted crees que Docker y Odoo son herramientas de fácil aprendizaje para su implementación
    a. Si ( )
    b. No ( )

ANEXO N° 06: FOTOGRAFÍAS

Realizando la visita a los ambientes de la Librería El Virrey SAC

ANEXO N° 07: GERENTE Y LISTA DE COLABORADORES

Nombre del Gerente de la librería: Koffler de Sanseviero Brunilda Viviana

APELLIDOS Y NOMBRES	CARGO	EDAD	TURNO
ESCOBEDO HUANQUI EDWIN PABLO	Jefe de Departamento de sistemas	30	Mañana
LUIS ENRIQUE ARENAS SOTO	Auxiliar de instalaciones	25	Tarde
MARCO ANTONIO VARA VARA	Supervisor de control de procesos	29	Mañana
JORGE DIAZ INFANTE	Analista de Soporte	29	Mañana
ROBERTO VALLE CRISOSTOMO	Auxiliar de instalaciones	24	Tarde
ABEL SALAZAR PINEDA	Auxiliar de instalaciones	25	Tarde
VICTOR COICA ELESCANO	Analista de Soporte	28	Mañana
SARCO VALDEZ LUIS ERNESTO	Auxiliar de instalaciones	25	Tarde
MARCELO CASTRO CESAR LEON	Auxiliar de instalaciones	28	Mañana
EMILIO SALDAÑA GUTIERREZ	Supervisor de control de procesos	24	Tarde
JESUS GUERRA SALAS	Analista de Soporte	24	Tarde
JUAN ALBERTO RETIS TITO	Auxiliar de instalaciones	24	Tarde
ALEJANDRO MEZA LOPEZ	Auxiliar de instalaciones	26	Mañana