UNIVERSIDAD NACIONAL DE SAN CRISTÓBAL DE HUAMANGA

FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

APLICACIÓN PARA GESTIÓN DE SEGURIDAD EN CLÚSTERS KUBERNETES, ENTORNOS PRIVADOS, 2022.

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO DE SISTEMAS

PRESENTADO POR:

Bach. Christian ALTAMIRANO AYALA

ASESOR:

Dr. Ing. Efraín Elías PORRAS FLORES

Ayacucho - Perú

2022

ACTA DE SUSTENTACIÓN DE TESIS

ACTA N° 034-2022-FIMGC

En la ciudad de Ayacucho, en cumplimiento a la **RESOLUCIÓN DECANAL N° 147-2022-FIMGC-D**, siendo los dieciséis días del mes de junio del 2022, a horas 9:00 a.m.; se reunieron los jurados del acto de sustentación, en el Auditórium virtual google meet del Campus Universitario de la Universidad Nacional de San Cristóbal de Huamanga.

Siendo el Jurado de la sustentación de tesis compuesto por el presidente el **Dr. Ing. Manuel Avelino LAGOS BARZOLA,** Jurado el **Mg. Ing. Eloy VILA HUAMÀN,** Jurado el **Ing. José Antonio GUERRERO HINOSTROZA,** Jurado - Asesor el **Dr. Ing. Efraín Elías PORRAS FLORES** y secretario del proceso el **Mg. Ing. Christian LEZAMA CUELLAR,** con el objetivo de recepcionar la sustentación de la tesis denominada "**APLICACIÓN PARA GESTIÓN DE SEGURIDAD EN CLUSTERS KUBERNETES, ENTORNOS PRIVADOS, 2022",** sustentado por el Señor **Christian ALTAMIRANO AYALA,** Bachiller en **Ingeniería de Sistemas**.

El Jurado luego de haber recepcionado la sustentación de la tesis y realizado las preguntas, el sustentante al haber dado respuesta a las preguntas, y el Jurado haber deliberado; califica con la nota aprobatoria de **16 (Dieciséis).**

En fe de lo cual, se firma la presente acta, por los miembros integrantes del proceso de sustentación.

Dr. Ing. Manuel Avelino LAGOS BARZOLA
Presidente

Mg. Ing. Eloy VILA HUAMÀN
Jurado

Dr. Ing. Efraín Elías Porras Flores
Firmado digitalmente por Dr. Ing. Efraín Elías Porras Flores
Fecha: 2022.06.28 08:36:21 -09'00'

Dr. Ing. Efraín Elías PORRAS FLORES
Jurado Asesor

Ing. José Antonio GUERRERO HINOSTROZA
Jurado

Firmado digitalmente por LEZAMA CUELLAR CHRISTIAN

Mg. Ing. Christian LEZAMA CUELLAR
Secretario del Proceso

c.c.:
Bach. Christian ALTAMIRANO AYALA
Jurados (4)
Archivo

DEDICATORIA

A mis padres y hermanos por apoyarme, corregirme, motivarme y guiarme en las distintas etapas de mi vida.

AGRADECIMIENTO

A mi alma mater Universidad Nacional de San Cristóbal de Huamanga por haberme recibido en sus aulas durante mi formación profesional y los docentes de la Escuela de Formación Profesional de Ingeniería de Sistemas por compartir sus invaluables conocimientos en mi formación profesional.

ÍNDICE

Pág.

ÍNDICE DE TABLAS

Pág.

ÍNDICE DE FIGURAS

Pág.

ÍNDICE DE ANEXOS

Pág.

RESUMEN

El uso de contenedores cambió la manera en que las empresas despliegan las aplicaciones, esto debido a las ventajas que supone su uso como la facilidad de despliegue, homogenización de entornos, fácil distribución de los artefactos y sobre todo la facilidad de escalar; Adicionalmente el auge de los servicios cloud en donde se manejan entornos o recursos dinámicos en el cual un grupo de instancias de nuestras aplicaciones pueden incrementar o reducirse de acuerdo a la carga que tenga en momento dado, hace que los contenedores encajen perfectamente en este tipo de necesidades, todo ello provocó que la adopción de contenedores se vea reflejado en estos últimos 5 años en donde el uso de contenedores en entornos productivos pasó de un 23% a un 84%. El objetivo de la investigación fue desarrollar una solución de escritorio para gestionar la seguridad de clúster Kubernetes para entornos privados, 2022. El tipo de investigación es observacional, prospectivo, transversal, descriptivo. La investigación es de tipo observacional porque no se hizo experimento alguno; es prospectivo porque los datos que se obtuvieron fueron mediante entrevistas a expertos para el estudio; es de tipo transversal porque medimos la variable seguridad de clúster en un momento específico; es descriptivo porque desarrollamos un software para gestionar la autorización y autenticación en clústeres Kubernetes. En la presente investigación se utilizó el marco de trabajo Scrum que tiene un ciclo de vida iterativa e incremental, se utilizará Kubeadm y Rancher Kubernetes Engine (RKE) como orquestadores de contenedores.

Palabras clave: RBAC, Autorización, Autenticación, Kubernetes, Seguridad, Control de Accesos.

ABSTRACT

The use of containers changed the way in which companies deploy applications, due to the advantages that their use entails, such as ease of deployment, homogenization of environments, easy distribution of artifacts and, above all, ease of scaling; Additionally, the rise of cloud services where dynamic environments or resources are managed in which a group of instances of our applications can increase or decrease according to the load they have at a given time, makes containers fit perfectly in this type of needs, all this caused the adoption of containers to be reflected in the last 5 years where the use of containers in productive environments went from 23% to 84%. The objective of the research was to develop a desktop solution to manage Kubernetes cluster security for private environments, 2022. The type of research is observational, prospective, cross-sectional, descriptive. The research is of an observational type because no experiment was carried out; it is prospective because the data obtained was through interviews with experts for the study; it is cross-sectional because we measure the cluster security variable at a specific time; it is descriptive because we developed software to manage authorization and authentication in Kubernetes clusters. In the present investigation, the Scrum framework was used, which has an iterative and incremental life cycle, Kubeadm and Rancher Kubernetes Engine (RKE) will be used as container orchestrators.

Keywords: RBAC, Authorization, Authentication, Kubernetes, Security, Access Control.

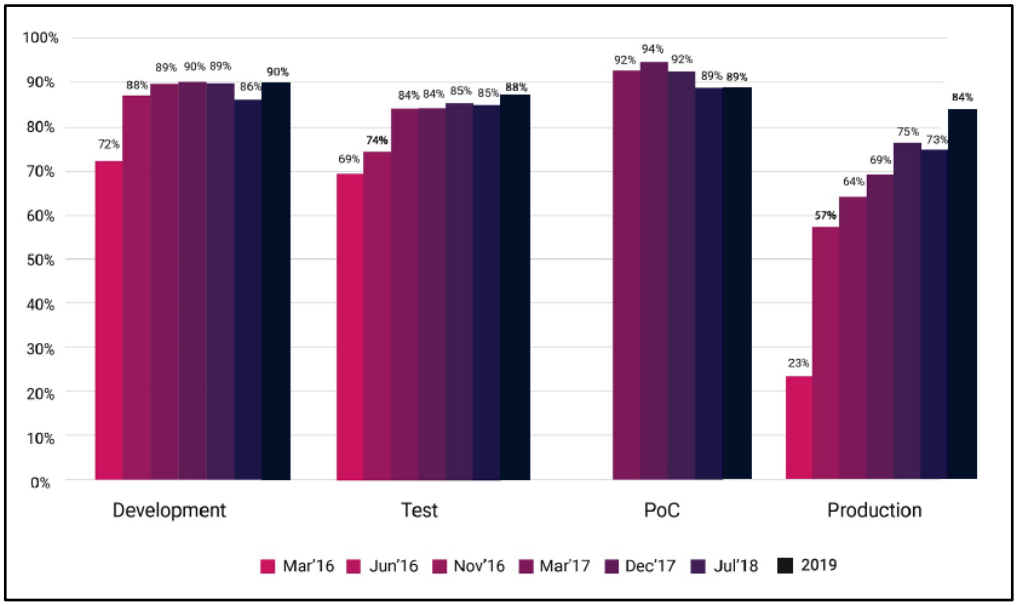
CAPÍTULO I
INTRODUCCIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

El uso de contenedores cambió la manera en que las empresas despliegan las aplicaciones, esto debido a las ventajas que supone su uso como la facilidad de despliegue, homogenización de entornos, fácil distribución de los artefactos y sobre todo la facilidad de escalar; Adicionalmente el auge de los servicios cloud en donde se manejan entornos o recursos dinámicos en el cual un grupo de instancias de nuestras aplicaciones pueden incrementar o reducirse de acuerdo a la carga que tenga en momento dado, hace que los contenedores encajen perfectamente en este tipo de necesidades, todo ello provocó que la adopción de contenedores se vea reflejado en estos últimos 5 años en donde el uso de contenedores en entornos productivos pasó de un 23% a un 84%.

Figura 1

Uso de contenedores por entornos, 2016-2019.

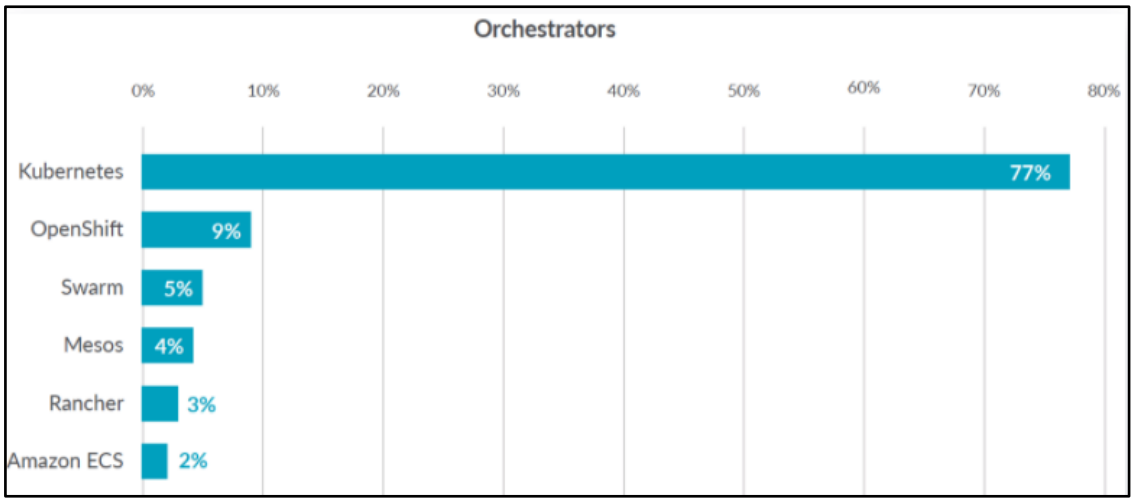


Fuente: Cloud Native Computing Foundation, (2019).

El incremento del uso de los contenedores también se ve reflejado en el uso de plataformas de orquestación de contenedores, esto debido a la facilidad de gestión y control de contenedores que nos ofrecen dichos orquestadores; de entre todas las opciones disponibles en el mercado se puede notar el dominio que tiene Kubernetes con respecto a otras alternativas, esto debido a diversas razones como: la gran cantidad de contribuyentes que posee, es un proyecto open source, tiene releases continuos, etc.

Figura 2

Uso de orquestadores de contenedores.



Fuente: Sysdig, (2019).

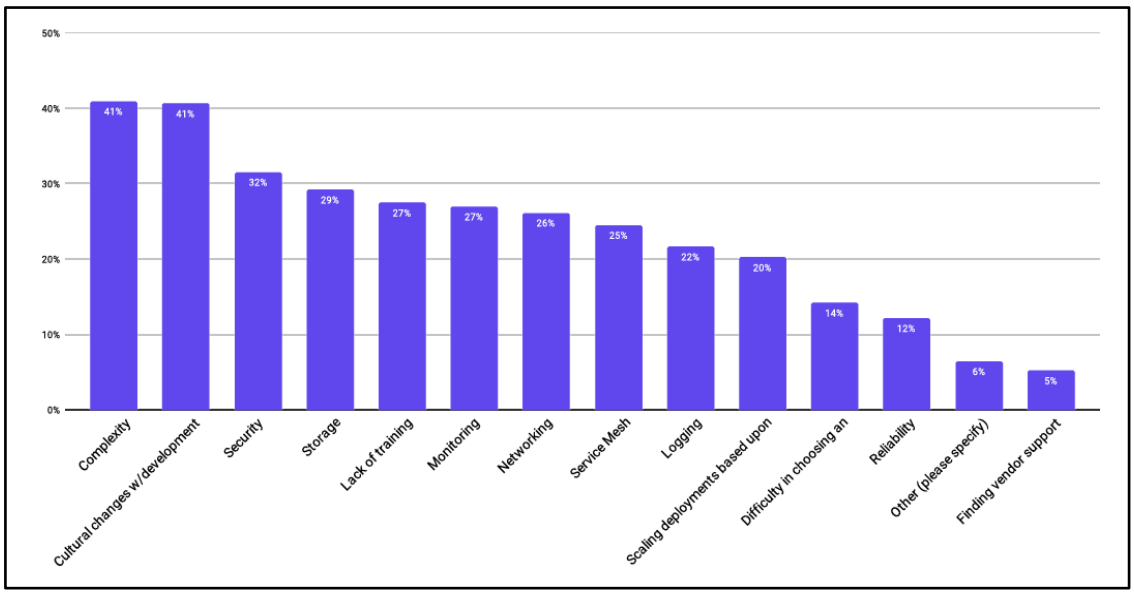
A pesar de los muchos beneficios que nos otorgan Kubernetes a la hora de orquestar los contenedores, existen aún ciertos desafíos a la hora de implementarlos; una de las preocupaciones tiene que ver con la seguridad a nivel de autenticación y autorización a la hora de operarlo, ya que a pesar que Kubernetes cuenta con mecanismos internos que permite implementar validaciones tanto de autenticación y autorización este no es usado del todo o no es usado correctamente, generando así una brecha de seguridad que podrían ocasionar el mal funcionamiento de nuestras aplicaciones así como dañar la infraestructura del clúster.

Una de las principales causas del mal manejo de estos mecanismos de seguridad que tiene Kubernetes es el desconocimiento de la propia herramienta por parte del personal de seguridad, ya que ello implica conocer los recursos que expone y sus respectivos permisos, adicional a ello es importante saber que Kubernetes tiene la capacidad de ser extensible a través de Custom Resource Definitions (CRD) por lo aumenta el nivel de

complejidad, todo este desconocimiento ocasiona que los permisos asignados a los administradores tengan altos privilegios, poniendo en riesgo las aplicaciones y secretos desplegados. Es por ello que la encuesta realizada por la Cloud Native Computing Foundation (2020), resalta que un 32% de los encuestados considera un reto el gestionar la seguridad a la hora usar contenedores.

Figura 3

Retos a la hora de desplegar contenedores.



Fuente: Cloud Native Computing Foundation, (2020).

Esta problemática es aún peor para clúster Kubernetes desplegados en entornos on-premises, ya que los proveedores cloud que también ofrecen kubernetes como servicios auto gestionados resolvieron esta problemática integrando Kubernetes con sus servicios de Gestión de Identidades y Accesos (IAM).

Por lo descrito anteriormente, surge la necesidad de contar con una herramienta que permita al administrador de seguridad definir permisos tanto a nivel de autenticación como autorización sobre los recursos o APIs que expone Kubernetes.

1.2. FORMULACIÓN DEL PROBLEMA

1.2.1. Problema general

¿De qué manera gestionar la seguridad en clúster Kubernetes, para entornos privados, 2022?

1.2.2. Problemas específicos

- ¿Cómo gestionar la autenticación?
- ¿De qué manera gestionar la autorización?

1.3. JUSTIFICACIÓN DE LA INVESTIGACIÓN

Gestionar adecuadamente los permisos mínimos necesarios de los recursos que tiene un clúster Kubernetes permitirá a la organización reducir brechas de seguridad que pudiera generar afectación ya sea a nivel de clúster como a nivel de aplicación, asimismo permitirá a los administradores de seguridad la posibilidad de generar roles y permisos de una manera más fácil y eficiente.

Implementar un mecanismo que permita gestionar la seguridad de clústeres Kubernetes de manera nativa beneficiará a las empresas que cuenten con dichos orquestadores ejecutándose en entornos on-premises.

Tener un componente que gestione la autenticación y autorización de clusters permitirá a la empresa desarrollar fácilmente políticas de control de accesos hacia los recursos que expone Kubernetes, asimismo facilitará al personal de seguridad que no tenga los conocimientos suficientes sobre los mecanismos de seguridad que ofrece Kubernetes a gestionar la seguridad de los clusters sin la necesidad de saber que recursos tenga el cluster, que permisos es posible aplicar sobre los distintos recursos, eliminar roles o permisos no necesarios, etc.

1.4. LIMITACIONES DE LA INVESTIGACIÓN

El estudio considera clúster Kubernetes desde la versión 1.19 en adelante, además, de entre las distintas distribuciones que existen en el mercado se usarán dos de ellas, Kubernetes Nativo y Rancher Kubernetes Engine, no obstante, debería soportar en otras distribuciones siempre en cuando estas implementen las APIs requeridas por la investigación.

1.5. OBJETIVOS

1.5.1. Objetivo general

Desarrollar una aplicación que permita gestionar la seguridad en clúster Kubernetes, para entornos privados, 2022.

1.5.2. Objetivo específico

- Desarrollar una aplicación desktop para gestionar la autenticación.
- Implementar una aplicación desktop para gestionar la autorización

CAPÍTULO II
MARCO TEÓRICO

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

Según el reporte realizado por la Agencia de Seguridad Nacional de los Estados Unidos (2021), la Autenticación y la Autorización son los mecanismos primarios para restringir el acceso a los recursos de clústeres Kubernetes, por lo que ciberactores pueden escanear los puertos conocidos de kubernetes y acceder a la base de datos o realizar llamadas a APIs sin estar autenticados si el clúster está mal configurado.

VMware (2021), realizó un estudio sobre las preocupaciones a nivel de seguridad que tienen las empresas que usan Kubernetes como parte del despliegue de sus aplicaciones, dando como resultado que el 23% indica que aplicar políticas consistentes entre los clústeres y equipos es la principal preocupación y un 17% indica que el control de acceso al clúster es la principal preocupación.

Para OWASP(2021), Tener un fuerte sistema de control de acceso basado en roles (RBAC) es posiblemente uno de los mayores requerimientos críticos en grandes organizaciones, ya que incluso los sistemas más seguros pueden fácilmente ser eludidos por usuarios o trabajadores con mayores privilegios, por lo que restringir los privilegios de los usuarios al mínimo de privilegios necesarios para realizar un trabajo, asegurar que el acceso a los clusters sean configurados a “denegar todo” por defecto, y asegurar una documentación adecuada detallando los roles y responsabilidades es una de las preocupaciones más críticas en las empresas.

La Cloud Native Computing Foundation (2019), realizó una encuesta sobre la adopción de tecnologías cloud-native, en donde el 40% indica que la seguridad es un reto a la hora de usar o desplegar aplicaciones contenerizadas.

2.2. MARCO CONCEPTUAL

2.2.1. Seguridad en Clúster

La seguridad de cluster Kubernetes está basada en la seguridad de Cloud o Datacenters, en donde los proveedores deben de proporcionar las mejores prácticas para alojar el cluster, Cluster en donde se debe configurar las APIs de Kubernetes así como las aplicaciones desplegadas, Contenedor, ya que la imagen a ser usada para el despliegue debe seguir buenas prácticas como escaneos constantes en busca de vulnerabilidades o construir imágenes con los permisos necesarios para ejecutar la aplicación y Código, en donde se debe minimizar las vulnerabilidades posibles, desde realizar un análisis de código estático hasta el manejo de técnicas de encriptación de datos (VMWare, 2021).

La seguridad de cluster kubernetes consiste en principalmente en 2 áreas, la seguridad de componentes del cluster que son configurables y la seguridad de las aplicaciones que se ejecuten sobre el cluster, de cara a la seguridad de componentes configurables del cluster este implica que se debe implementar mecanismos tales como el control de acceso al API de Kubernetes mediante autenticación y autorización, control de acceso al componente Kubelet, control de permisos de las aplicaciones desplegadas. Del lado de las aplicaciones que se ejecutan sobre un clúster se deben implementar mecanismos que permitan reducir la superficie de ataque, dicho mecanismos podrían implicar la configuración de RBAC a las aplicaciones, la gestión de secretos, Políticas de Red y manejo de TLS para peticiones provenientes del exterior (Kubernetes, 2021).

A. Autenticación

Según Shacklett (2021), La autenticación es el proceso de determinar si alguien o algo es quien dice ser, las tecnologías de autenticación proporcionan el control de accesos para sistemas revisando si las credenciales del usuario concuerdan con las credenciales de alguna base de datos de usuarios autorizados o en un servidor de autenticación. Al realizar dicho proceso, se asegura sistemas seguros, procesos seguros y seguridad de la información empresarial.

Según Luksa (2018), el proceso de autenticación en Kubernetes sigue de la siguiente manera: una petición es recibido por el API server, este va a través de una lista de plugins de autenticación, así estos pueden examinar la petición y tratar de determinar quién está enviando dicha petición. El primer plugin que pueda extraer la información

de la petición extrae los valores como usuario, id y los grupos al que pertenece el cliente. El API server deja de invocar a los plugins restantes y continua por la fase de autorización.

Para Tigera (2021), la autenticación en un clúster Kubernetes consiste en el siguiente proceso: el cliente intenta acceder al API, para ello presenta los certificados TLS, una vez establecida conexión el API server ejecuta uno o muchos módulos de autenticación, los módulos de autenticación reciben la petición HTTPS e intentan autenticarse, si la autenticación falla el API retorna un código de estado 401, si la autenticación es exitosa el API server prosigue con el proceso de autorización.

B. Autorización

La autorización es la función de definir el acceso seguro a recursos, el cual está relacionado con la seguridad de la información y seguridad de computadoras, y al control de accesos en particular. Formalmente "autorizar" es definir políticas de acceso. Por ejemplo, el personal de recursos humanos está normalmente autorizado a acceder a registros de los empleados y esta política es a menudo formalizado como una regla de control de acceso sobre un Sistema. Durante cualquier operación, un sistema usa las reglas de control de acceso para decidir si el acceso solicitado debería ser aprobado o rechazado (Audun, 2017).

Kubernetes soporta multiple modulos de autorizacion, como ABAC, RBAC y webhook. Cuando un administrador crea un cluster, configura el modulo de autorizacion que deberia ser usado in el API server. Si mas de un modulo de autorizacion son configurados, Kubernetes revisa cada modulo, y si algun modulo autoriza la peticion, el request es aceptado. Si todos los modulos rechazan alguna peticion, entonces el request es denegado (Kubernetes, 2022).

Una vez autenticado, se espera que cada llamada al API server pase la revision de autorizacion, Kubernetes lleva un componente integrado de Control de Acceso basado en roles que vincula un entre los usuarios o grupos con un lista de roles. Estos permisos (get, post, put, etc) se combinan con los recurso (pod, deployment, ingress, etc.) y pueden se limitados a nivel de namespace o cluster. Es recomendado el uso de los autorizadores de Nodo y RBAC de manera conjunta (Kubernetes, 2022).

2.3. MARCO REFERENCIAL

2.3.1. Control de Accesos basado en Roles

Segun Luksa (2017), el control de acceso basado en roles(RBAC) trata a la autorizacion como permisos asociados a roles mas no asi en usuarios o grupos. Un rol no es nada mas que una colección de permisos.

Según la Agencia de Segurida Nacional (2021), el control de acceso basado en roles (RBAC) es un metodo para controlar el acceso a recursos del cluster basado en los roles de los individuos de una organización, se pueden configurar 2 tipos de permisos, los Roles y los ClusterRoles, en el cual los Roles aplican para un Namespace en particular mientras el ClusterRole aplica para el cluster en general.

Para Auth0(2021), el control de acceso basado en roles nos brinda una ventaja en cuanto a la gestion de la autorización ya que los administradores de sistemas pueden gestionar los permisos de manera masiva en vez de estar configurando uno por uno.

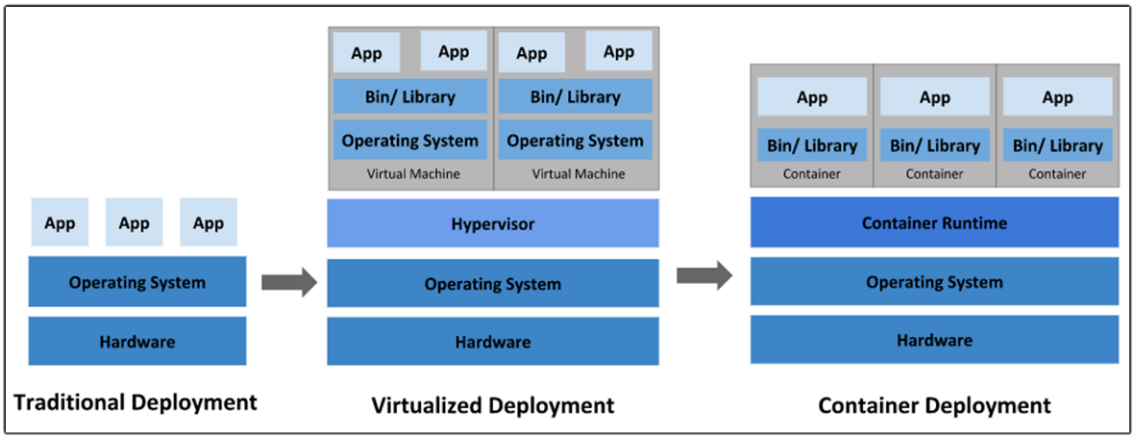
2.3.2. Kubernetes

Para Luksa (2018), Kubernetes permite a los desarrolladores desplegar sus aplicaciones por cuenta propia y a menudo a demanda, sin requerir alguna asistencia de algun equipo de operaciones, sin embargo Kubernetes no solo beneficia a los desarrolladores. Este tambien ayuda al equipo de operaciones en el monitoreo automatico de la progrmacion de las aplicaciones en un evento de falla de hardware. El foco de los administradores de sistemas recae de supervisar aplicaciones individuales a supervisar y administrar Kubernetes, mientras Kubernetes por si mismo se encarga de las aplicaciones.

Kubernetes es una plataforma open-source portable y extensible para gestionar cargas de trabajos y servicios contenerizados que facilita la configuración y automatización declarativa. Asimismo tiene un ecosistema amplio y de rápido crecimiento (Kubernetes, 2022).

Figura 4

Comparación modelo de despliegues.



Fuente: Kubernetes, (2022).

A. Namespace

Según Luksa (2018), los namespaces son un mecanismo para agrupar objetos de Kubernetes de manera organizada, que permite dividir sistemas complejos de numerosos componentes en grupos organizados. Además, permite el de poder objetos con el mismo nombre en distintos namespaces.

B. Service Account

Según Luksa (2018), los Service Account son una manera de autenticación entre una aplicación que se ejecuta dentro del clúster contra el API Server, las aplicaciones usan este mecanismo usando los tokens de los Service Account.

C. Cluster Role y Cluster Role Binding

Según la Agencia de Segurida Nacional (2021), los cluster roles son un conjunto de permisos que se aplica a nivel del Cluster sin importar los namespaces. A su vez el Cluster Role Binding es el mecanimos de asignacion de permisos Cluster Role a usuarios, grupos o service accounts.

D. Role y Role Binding

Según la Agencia de Segurida Nacional (2021), los roles son un conjunto de permisos que se aplica a nivel de namespace. A su vez el Role Binding es el mecanimos de asignacion de permisos Role a usuarios, grupos o service accounts.

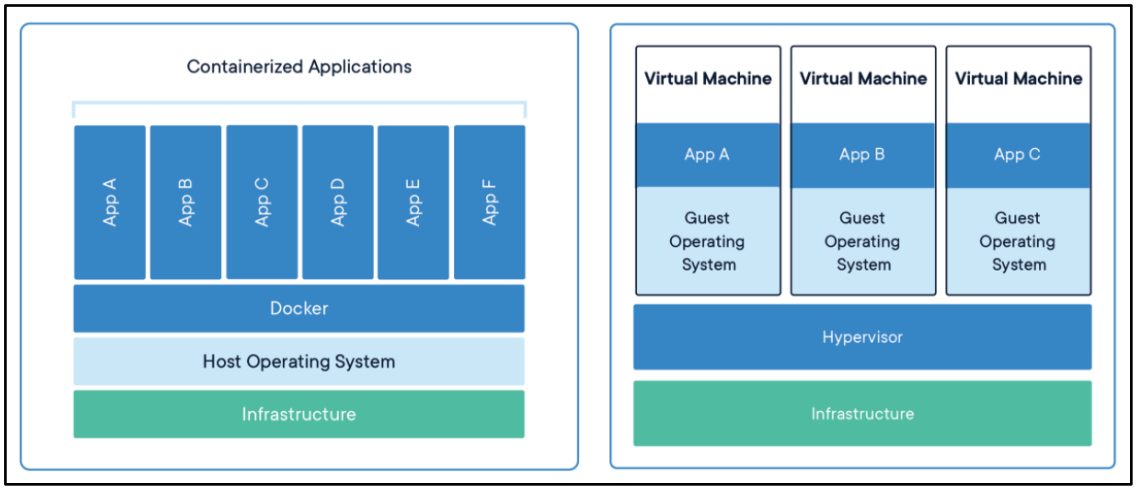
2.3.3. Contenedor

Un contenedor es una unidad estandar de software que enpaqueta codigo y sus respectivas dependencias para que la aplicación pueda ejecutarse de una manera rápida y confiable desde un entorno de computo cualquiera. Un contenedor de imagen es un paquete de software ligero, ejecutable e independiente que incluye todo lo necesario para ejecutar una aplicación: codigo, runtime, herramientas de sistema, librerías de sistemas y ajustes de sistemas (Docker, 2021).

Los contenedores son similares a máquinas virtuales, que comparten el sistema operativo entre las aplicaciones. Por lo tanto, los contenedores son considerados ligeros. Similar a una maquina virtual, un contenedor tiene su propios sistema de archivos, CPU, memoria, espacio de procesos y más. Dado que los contendores son desacoplados de la infraestructura en la que se ejecutan significa que estos son portables a traves de distintas distribuciones de sistemas operativos. (Kubernetes, 2022)

Figura 5

Comparación entre contendor versus máquinas virtuales.



Fuente: Docker, (2022).

2.3.4. Lenguaje de programación orientado a objetos

Es un método de implementación en el que los programas se organizan como colecciones cooperativas de objetos, cada uno de los cuales representan una instancia de alguna clase, y cuyas clases son todas miembros de una jerarquía de clases unida mediante relaciones de herencia (Booch, 2001).

El paradigma orientado a objetos es útil cuando el sistema se modela de forma casi análoga a la realidad, porque así se simplifica el diseño de alto nivel. Esta analogía permite que los programadores tengan más claro cuál es el papel de cada porción del programa y de los datos, lo que facilita la creación y el mantenimiento del sistema. Además, se promueve la reutilización, pues las similitudes entre objetos se programan sólo una vez en forma abstracta y el programador concentra su esfuerzo en las diferencias concretas (Cervantes et al, 2016).

Características

a. Clase

Definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ellas (Booch, 2001).

Una clase es un tipo de datos que el programador selecciona específicamente para crear objetos. Se dice que cada objeto es un ejemplo concreto de una clase particular de características. Una clase define las propiedades comunes de un grupo de objetos. El programador define la clase como un tipo de datos complejo y le da un nombre. (Cervantes et al, 2016).

b. Método

Algoritmo asociado a un objeto (o a una clase de objetos), cuya ejecución se desencadena tras la recepción de un mensaje. Desde el punto de vista del comportamiento, es lo que el objeto puede hacer. Un método puede producir un cambio en las propiedades del objeto, o la generación de un evento con un nuevo mensaje para otro objeto del sistema (Booch, 2001).

Los métodos dan forma al comportamiento de los objetos de clase. Definir un método es muy similar a definir una función. Los métodos públicos son las operaciones que los objetos externos realizan con el objeto en cuestión. Los métodos privados son procesos internos que no se pueden llamar desde el exterior, pero se pueden llamar desde otro método dentro de la clase. (Cervantes et al, 2016).

c. Objeto

En la POO, un objeto es una entidad virtual (o entidad de software), con datos y funciones que simulan las propiedades del objeto. Los objetos con los que se construyen los programas se ven como si fueran máquinas, las cuales están formadas por un conjunto de elementos autónomos. Las propiedades individuales de estos elementos y las relaciones entre sí definen el funcionamiento general de la máquina (Cervantes et al, 2016).

d. Herencia

Para Deitel y Deitel (2012). Es posible crear una nueva clase de objetos con rapidez y de manera conveniente mediante la herencia; la nueva clase absorbe las características de una clase existente, con la posibilidad de personalizarlas y agregar características únicas propias.

La herencia es una relación entre clases en las que hay una clase padre, llamada superclase, y una o más clases hijas especializadas, a las que se les denomina subclases. La herencia es el mecanismo mediante el cual se implementa la relación de generalización. En la práctica, cuando se codifica un sistema, se habla de herencia en lugar de generalización (Cervantes et al, 2016).

e. Encapsulamiento

Las clases encapsulan (envuelven) los atributos y métodos de objetos; los atributos y métodos de un objeto están muy relacionados entre sí. Pero por lo general no se les permite saber cómo están implementadas otros objetos; los detalles de implementación están ocultos dentro de los mismos objetos (Deitel y Deitel, 2012).

La encapsulación protege los datos de los objetos y se logra declarando las propiedades de una clase como métodos de control de acceso privados y marcados. La forma de acceder a las propiedades desde fuera de la clase es a través de los métodos getter y la forma de modificar las propiedades desde fuera de la clase es a través de los métodos setter. (Cervantes et al, 2016).

f. Polimorfismo

El polimorfismo, en OOP, es una propiedad de que los objetos de las subclases se

representan como superclases. Por ejemplo, si tenemos dos clases X y Z que tienen el mismo padre W, entonces podemos usar una variable de clase W que puede tomar el valor de un objeto de clase X o también el valor de un objeto de clase X. Clase estatua z (Cervantes et al, 2016).

2.3.5. Metodología de desarrollo ágil SCRUM

Según Schwaber y Sutherland (2011), Scrum es un marco de trabajo de procesos que ha sido utilizado para gestionar el desarrollo de productos complejos. Scrum no es un proceso o una técnica para construir productos; en lugar de eso, es un marco de trabajo dentro del cual se puede emplear varios procesos y técnicas. Scrum hace patente la eficacia relativa de tus prácticas de gestión de producto y de desarrollo de modo que puedas mejorarlo.

Según Schwaber y Sutherland (2017), Scrum es un marco de trabajo por el cual las personas pueden abordar problemas complejos adaptativos, a la vez que entregar productos del máximo valor posible productiva y creativamente. Scrum no es un proceso, una técnica o método definitivo. En lugar de eso, es un marco de trabajo dentro del cual se pueden emplear varios procesos y técnicas. Scrum muestra la eficacia relativa de las técnicas de gestión de producto y las técnicas de trabajo de modo que podamos mejorar continuamente el producto, el equipo y el entorno de trabajo.

A. Pilares Scrum

1. Transparencia

Los aspectos importantes del proceso deben ser visibles para los responsables del resultado. La transparencia requiere que estos aspectos se determinen de acuerdo con un estándar común, de modo que los observadores compartan un entendimiento común de lo que están viendo (Schwaber y Sutherland, 2017).

2. Inspección

Los usuarios de Scrum deben verificar regularmente si hay artefactos en Scrum y apuntar a variaciones inesperadas. Sus controles no deben ser tan frecuentes que interfieran con su trabajo. Las inspecciones son más rentables cuando se llevan a cabo diligentemente por inspectores calificados en el sitio (Schwaber y Sutherland, 2017).

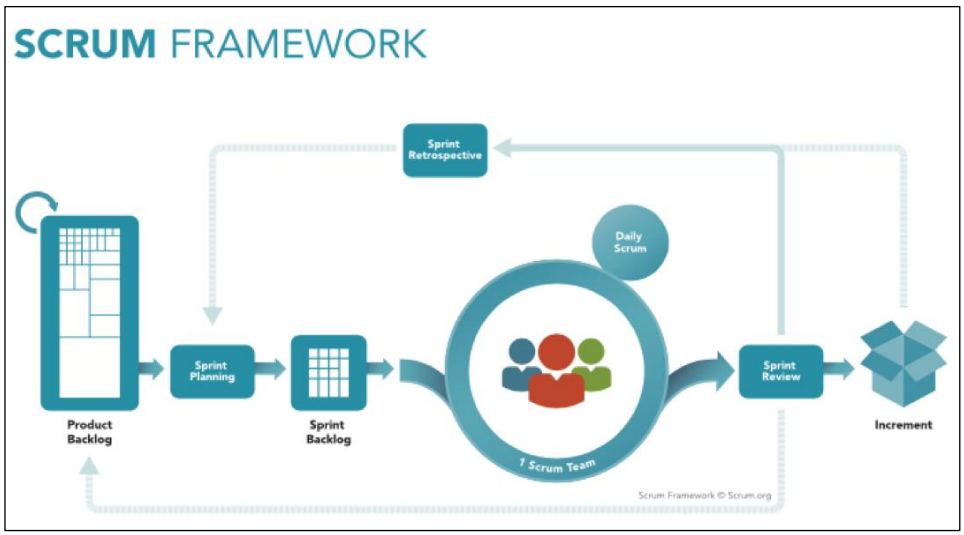
3. Adaptación

Si el auditor determina que uno o más aspectos del proceso se desvían de los límites aceptables y que el producto resultante será inaceptable, se debe modificar el proceso o el material que se maneja. parche. Este ajuste debe hacerse lo más rápido posible para reducir grandes desviaciones (Schwaber y Sutherland, 2017).

B. Proceso Scrum

Figura 6

Diagrama de proceso Scrum.



Fuente: Schwaber y Sutherland, (2017).

La metodología de Scrum pone especial énfasis en diseños limpios y simples. Los conceptos de diseño más importantes en esta metodología son:

1. Sprint

El corazón de Scrum es el Sprint, es un bloque de tiempo (time-box) de un mes o menos durante el cual se crea un incremento de producto "Terminado" utilizable y potencialmente desplegable. Es más conveniente si la duración de los Sprints es consistente a lo largo del esfuerzo de desarrollo. Cada nuevo Sprint comienza inmediatamente después de la finalización del Sprint anterior (Schwaber y Sutherland, 2017).

2. Planificación del Sprint

La Planificación de Sprint tiene un máximo de duración de ocho horas para un Sprint de un mes. Para Sprints más cortos el evento es usualmente más corto. El Scrum Master se asegura de que el evento se lleve a cabo y que los asistentes entiendan su propósito. El Scrum Master enseña al Equipo Scrum a mantenerse dentro del bloque de tiempo. La Planificación de Sprint responde a las siguientes preguntas: ¿Qué puede entregarse en el Incremento resultante del Sprint que comienza?, ¿Cómo se conseguirá hacer el trabajo necesario para entregar el Incremento? (Schwaber y Sutherland, 2017).

3. Scrum Diario

El Daily Scrum es una reunión de 15 minutos para el equipo de desarrollo. El Daily Scrum se lleva a cabo todos los días de la carrera. En este documento, el equipo de desarrollo planea trabajar durante las próximas 24 horas. Esto mejora la colaboración y el rendimiento del equipo al verificar el trabajo avanzado desde la última reunión diaria y predecir el trabajo del próximo sprint. El scrum diario se realiza a la misma hora y se realiza todos los días para reducir la complejidad. (Schwaber y Sutherland, 2017).

El equipo de desarrollo usa scrum a diario para evaluar el progreso hacia el objetivo del sprint y cómo progresar para completar el trabajo involucrado en el backlog del sprint. Daily Scrum mejora las posibilidades del equipo de desarrollo de alcanzar los objetivos de Sprint. En el día a día, el Equipo de Desarrollo debe comprender cómo pretenden trabajar juntos como un equipo autoorganizado para lograr el objetivo del Sprint y crear el aumento esperado al final del Sprint (Schwaber y Sutherland, 2017).

4. Revisión del Sprint

Al final del Sprint se lleva a cabo una Revisión de Sprint para inspeccionar el Incremento y adaptar la Lista de Producto si fuese necesario. Durante la Revisión de Sprint, el Equipo Scrum y los interesados colaboran acerca de lo que se hizo durante el Sprint. Basándose en esto y en cualquier cambio a la Lista de Producto durante el Sprint, los asistentes colaboran para determinar las siguientes cosas que podrían hacerse para optimizar el valor. Se trata de una reunión informal, no una reunión de seguimiento, y la presentación del Incremento tiene como objetivo facilitar la retroalimentación de información y fomentar la colaboración (Schwaber y Sutherland, 2017).

5. Retrospectiva del Sprint

Una retrospectiva de Sprint es una oportunidad para que el equipo de Scrum se examine a sí mismo y planifique mejoras para implementar en el próximo Sprint (Schwaber y Sutherland, 2017).

Un sprint retrospectivo se realiza después de evaluar un sprint y antes de planificar el próximo sprint. Esta es una reunión de tres horas para Sprint por mes. Para carreras cortas, el evento suele ser más corto. El Scrum Master se asegura de que el evento se lleve a cabo y que los participantes entiendan su propósito. El propósito de Sprint Retrospective es:

- Descubrir cómo fue el último Sprint en términos de personas, relaciones, procesos y herramientas.
- Identificar y organizar los factores más importantes que van bien y que se pueden mejorar.
- Hacer un plan para realizar mejoras en la forma en que el equipo Scrum hace su trabajo.

C. Roles Scrum

1. Product Owner

El propietario del producto es responsable de maximizar el valor del producto como resultado del trabajo del equipo de desarrollo. La forma de hacer esto puede variar ampliamente entre diferentes organizaciones, equipos Scrum e individuos (Schwaber y Sutherland, 2017).

El Dueño de Producto es el responsable de gestionar la Lista del Producto (Product Backlog). Ducha gestión incluye:

- Definir claramente los elementos de la Lista del Producto;
- Ordenar los elementos en la Lista del Producto para alcanzar los objetivos y misiones de la mejor manera posible;
- Optimizar el valor del trabajo que el Equipo de Desarrollo realiza;
- Asegurar que la Lista del Producto es visible, transparente y clara para todos y que muestra aquello en lo que el equipo trabajará a continuación; y,
- Asegurar que el Equipo de Desarrollo entiende los elementos de la Lista del Producto al nivel necesario.

2. Equipo de desarrollo

El Equipo de Desarrollo consiste en los profesionales que realizan el trabajo de entregar un Incremento de producto "Terminado" que potencialmente se pueda poner en producción al final de cada Sprint. Un Incremento "Terminado" es obligatorio en la Revisión del Sprint. Solo los miembros del Equipo de Desarrollo participan en la creación del Incremento (Schwaber y Sutherland, 2017).

Los Equipos de Desarrollo tienen las siguientes características:

a. Son auto organizados. Nadie (ni siquiera el Scrum Master) indica al equipo de desarrollo cómo convertir elementos de la Lista del Producto en Incrementos de funcionalidad potencialmente desplegables.
b. Los equipos de desarrollo son multifuncionales, esto es, como equipo cuentan con todas las habilidades necesarias para crear un Incremento de producto.
c. Scrum no reconoce títulos para los miembros de un equipo de desarrollo independientemente del trabajo que realice cada persona.
d. Scrum no reconoce sub equipos en los equipos de desarrollo, no importan los dominios que requieran tenerse en cuenta, como pruebas, arquitectura, operaciones o análisis de negocio.
e. Los miembros individuales del equipo de desarrollo pueden tener habilidades especializadas y áreas en las que estén más enfocados, pero la responsabilidad recae en el equipo de desarrollo como un todo.

3. Scrum Master

El Scrum Master es responsable de promover y apoyar Scrum como se define en la Guía de Scrum. Los Scrum Masters hacen esto ayudando a todos a entender la teoría, prácticas, reglas y valores de Scrum (Schwaber y Sutherland, 2017).

D. Artefactos Scrum

1. Lista de Producto

Una cartera de productos es una lista ordenada de todo lo que se sabe que se requiere en un producto. Esta es la única fuente de solicitudes para cualquier cambio de producto. Los dueños de los productos son responsables de las listas de productos, incluidos el contenido, la disponibilidad y la forma de realizar pedidos. (Schwaber y Sutherland, 2017).

La lista de productos nunca está completa. Su desarrollo inicial solo reflejó los requisitos conocidos y entendidos desde el principio. El Product Backlog evoluciona a medida que evolucionan el producto y el entorno en el que se utiliza. Los listados de productos son dinámicos; Está en constante evolución para identificar productos que deberían ser relevantes, competitivos y ventajosos. Si hay un producto, también hay una lista de sus productos. (Schwaber y Sutherland, 2017).

2. Lista de Pendientes del Sprint

La Lista de Pendientes del Sprint es una predicción hecha por el equipo de desarrollo acerca de qué funcionalidad formará parte del próximo Incremento y del trabajo necesario para entregar esa funcionalidad en un Incremento "Terminado" (Schwaber y Sutherland, 2017).

La Lista de Pendientes del Sprint hace visible todo el trabajo que el Equipo de Desarrollo identifica como necesario para alcanzar el Objetivo del Sprint. Para asegurar el mejoramiento continuo, la Lista de Pendientes del Sprint incluye al menos una mejora de procesos de alta prioridad identificada en la Retrospectiva inmediatamente anterior (Schwaber y Sutherland, 2017).

3. Gráficos de Trabajo Pendiente

Muestra la velocidad a la que se está completando los requisitos. Permite deducir si el tiempo es el requerido para la finalización del proyecto o iteración. Se puede representarlo en intervalos de días u horas pendientes para la culminación de las tareas de iteración (Schwaber y Sutherland, 2011).

2.3.6. Tecnologías de internet

Luján (2001) afirma que a diferencia de otros servicios en línea, que son controlados centralmente, Internet tiene un diseño descentralizado. Porque cada computadora (servidor) en Internet es independiente. Los operadores pueden elegir qué servicio usar y qué servicio local ofrecer.

a. Aplicación Web

Para Luján (2001), una aplicación web es un tipo específico de aplicación cliente/servidor, en la que tanto el cliente (navegador, explorador o visor) como el

servidor (servidor web) y el protocolo al que se conectan (HTTP) están estandarizados y no creados por programadores de aplicaciones.

Una aplicación web generalmente se compone de los siguientes elementos: a) Recursos estáticos: paginas HTML, imágenes, sonidos, hojas de estilo, etc., b) Recursos dinámicos: servlets, JSP, Java Bean, c) Librerías de clases y d) Descriptor de despliegue para definir los parámetros de funcionamiento de la aplicación en el servidor (Groussard, 2010).

b. Protocolo HTTP

HTTP es parte de la familia de protocolos de comunicación TCP/IP (Protocolo de control de transmisión/Protocolo de Internet) que se utiliza en Internet. Estos protocolos permiten interconectar sistemas heterogéneos, facilitando el intercambio de información entre diferentes ordenadores. (Luján, 2000).

c. Protocolo TCP/IP

Según De la Cruz (2013), TCP es en lo más común de los lenguajes, el protocolo que resuelve las comunicaciones entre los diferentes sistemas operativos que existen tales como Windows, Linux, Novell, Unix o Solaris.

CAPÍTULO III
METODOLOGÍA

3.1. TIPO Y NIVEL DE INVESTIGACIÓN

3.1.1. Tipo de investigación

Para Salinas (1993), una investigación de tipo observacional "es aquella que se basa en la observación de los fenómenos, características, situaciones, variaciones, etc. del asunto que se quiere investigar. Sólo se observa, sin manipular, cambiar o variar nada. Luego, las observaciones hechas se pueden registrar para posterior análisis". La investigación es observacional, porque no se realizó ningún experimento para la gestión de la seguridad de los clústeres Kubernetes.

Para José (2012), la planificación potencial de la recopilación de datos es cuando los datos necesarios para el estudio se han recopilado para las necesidades de la investigación. Esta investigación es prospectiva, ya que los datos fueron recolectados a través de entrevistas con expertos en el tema.

Según Palomino et al. (2015), afirma que el escaneo transversal se debe a que los datos se recopilan en un momento y lugar. Su finalidad es describir las variables y analizar sus efectos y correlaciones en un momento dado. Es como tomar una foto de algo que está pasando. La búsqueda es exhaustiva ya que la información relacionada con la seguridad en los clústeres de Kubernetes se recopila en un solo momento.

3.1.2. Nivel de investigación

Bernal (2014) sostiene que la investigación descriptiva muestra, cuenta, describe o identifica eventos, situaciones, características, características de los sujetos o productos de investigación, arquetipos y patrones, pero no se da ninguna explicación o razón para las situaciones o eventos. Asimismo, la investigación descriptiva se basa en gran medida en técnicas como encuestas, entrevistas, observaciones y revisión de literatura.

Según Hernández et al., (2010), la investigación descriptiva busca especificar las propiedades, las características y los perfiles de personas, grupos, comunidades, procesos, objetos o cualquier otro fenómeno que se someta a un análisis. Es decir, únicamente pretenden medir o recoger información de manera independiente o conjunta sobre los conceptos o las variables a las que se refieren, esto es, su objetivo no es indicar cómo se relacionan ésta.

Supo (2020), la investigación descriptiva es un método que utiliza métodos analíticos, que pueden describir las características de un objeto de investigación o una situación específica, destacando sus características y características. Explicar qué son las cosas y describir el estado actual de las cosas. En combinación con ciertos criterios de clasificación, es posible ordenar, agrupar u organizar los elementos relevantes de la investigación. Trabaja sobre los hechos de los acontecimientos, y su objetivo principal es proporcionar una interpretación correcta de los hechos.

Según las referencias mostradas la investigación se enmarca en una investigación de tipo descriptiva ya que se busca brindar información acerca de la seguridad en orquestadores de contenedores.

3.2. DISEÑO DE LA INVESTIGACIÓN

Según Hernández, Fernández y Baptista (2010) mencionan que, las investigaciones no experimentales son aquellas que se realizan sin manipular deliberadamente las variables, es decir, no se varía intencionalmente la variable, simplemente se realiza la observación de las funciones tal y como se dan en su contexto natural para después analizarlas. El diseño es no experimental, debido a que no manipulamos la variable seguridad de clúster, los datos para desarrollar la aplicación desktop son obtenidos mediante la entrevista en un momento específico, aplicados a personal de seguridad que gestionan clúster Kubernetes.

3.3. VARIABLES

3.3.1. Definición conceptual de variables

Variable de Interés

Seguridad de Clúster: La seguridad de clúster Kubernetes consiste en principalmente en 2 áreas, la seguridad de componentes del cluster que son configurables y la seguridad

de las aplicaciones que se ejecuten sobre el cluster, de cara a la seguridad de componentes configurables del cluster este implica que se debe implementar mecanismos tales como el control de acceso al API de Kubernetes mediante autenticación y autorización.

Variables descriptivas

- **Autenticación:** La autenticación es el proceso de determinar si alguien o algo es quien dice ser.
- **Autorización:** La autorización es la función de definir el acceso seguro a recursos, el cual está relacionado con la seguridad de la información.

3.3.2. Definición operacional de variables

Variable de Interés

X: Seguridad de Clúster

Variables descriptivas

- **Autenticación:** Representa las peticiones autenticadas aceptadas, se mide en porcentaje.
- **Autorización:** Representa las peticiones aceptadas debidamente autorizadas, se mide en porcentaje.

3.4. POBLACIÓN Y MUESTRA

3.4.1. Población

Según los autores (Arias-Gómez, Villasís-Keever, & Miranda Novales, 2016), la población de estudio es un conjunto de casos, definido, limitado y accesible, que formará el referente para la elección de la muestra, y que cumple con una serie de criterios predeterminados. Es necesario aclarar que cuando se habla de población de estudio, el término no se refiere exclusivamente a seres humanos, sino que también puede corresponder a animales, muestras biológicas, expedientes, hospitales, objetos, familias, organizaciones, etc.

La población está compuesta por 4 clústeres Kubernetes, en entornos privados, 2022.

3.4.2. Muestra

Según Scharager y Armijo (2001). El muestreo no probabilístico también se conoce como muestreo intencional o de conveniencia; La selección de ítems no se basa en probabilidades sino en condiciones que permitan el muestreo, que puede ser utilizado en poblaciones heterogéneas y variables; También debe decidir cuándo se recopilarán los datos. No existe muestra, es un censo, porque se hará el estudio para los 4 clústeres Kubernetes en entornos privados.

3.5. OPERACIONALIZACIÓN DE LAS VARIABLES

Tabla 1

Operacionalización de variables.

VARIABLE	DIMENSIÓN	INDICADOR	PREGUNTA
Seguridad de Clúster	Autenticación	Certificados Digitales	¿Cómo autenticar un cluster usando certificados digitales?
			¿Cómo obtener un certificado digital?
			¿Cómo crear una solicitud de firma de certificado?
			¿Cómo aprobar una solicitud de firma de certificado?
			¿Cómo rechazar una solicitud de firma de certificado?
		Usuario	¿Cómo crear un usuario?
			¿Cómo asignar permisos a un usuario?
		Grupo	¿Cómo crear un usuario?
			¿Cómo asignar permisos a un grupo?
		Service Account	¿Cómo crear un Service account?
			¿Cómo eliminar un Service account?
			¿Cómo asignar permisos a un Service account?
	Autorización	Cluster Role	¿Cómo crear un Cluster Role?
			¿Cómo editar un Cluster Role?
			¿Cómo eliminar un Cluster Role?
			¿Cómo asociar un Cluster Role a un Grupo?
			¿Cómo asociar un Cluster Role a un Usuario?
			¿Cómo asociar un Cluster Role a un Service Account?
		Role	¿Cómo crear un Role?
			¿Cómo editar un Role?
			¿Cómo eliminar un Role?
			¿Cómo asociar un Role a un Usuario?
			¿Cómo asociar un Role a un Service Account?
			¿Cómo asociar un Cluster Role a un Grupo?
		API	¿Cuáles son los permisos que puede tener una API de Kubernetes?
			¿Cuáles son los alcances que tiene una API de Kubernetes?

Fuente. Elaboración propia.

3.6. TÉCNICAS E INSTRUMENTOS DE LA INVESTIGACIÓN

3.6.1. Técnicas

Arias (2006) señala que se trata de diferentes formas o métodos de recolección de información. El autor continúa mostrando que ejemplos de esta técnica son la observación directa, el escaneo binario (entrevista o cuestionario), el análisis de documentos y el análisis de contenido; Entre otras cosas.

Según lo mencionado anteriormente, el estudio se caracteriza por el uso de la observación como técnica de la entrevista no estructurada, para el desarrollo de la investigación se procedió con la entrevista no estructurada.

3.6.2. Instrumentos

Según, Arias (2006), indica que "los instrumentos son cualquier recurso, dispositivo o formato (papel o digital), que se utiliza para obtener, registrar o almacenar la información. Entre los cuales se puede mencionar: los cuestionarios, entrevistas y otros".

Para el desarrollo de la investigación se utilizaron guías de entrevista para recolectar información.

3.6.3. Validez del instrumento

Es un coeficiente que permite cuantificar la relevancia de los ítems respecto a un dominio de contenido a partir de las valoraciones de N jueces (Aiken, 1985).

El método más común para realizar verificaciones de elegibilidad de contenido contra los criterios del jurado es buscar la aprobación o el rechazo de un elemento para que se incluya en la prueba por parte de varios jueces, esta cantidad de entradas está sujeta a cambios a pedido del autor del instrumento. (Aiken, 1980).

En la investigación se realizó por validez de contenido, el coeficiente V de Aiken es 0.94, esto significar que el instrumento de recolección de datos tiene validez óptima. El coeficiente tiene un valor entre 0 y 1; mientras más se acerque a la unidad, mejor será la validez. El valor de significancia. en el anexo 4 se muestran los datos obtenidos por los 4 jueces.

3.6.4. Confiabilidad de instrumento

Para la encuesta no se aplicaron técnicas de confiabilidad de herramientas, se utilizaron como herramientas entrevistas no estructuradas, y sus criterios no fueron definidos por alternativas u opciones de retroalimentación como escala de calificación tipo Likert o dicotómica.

3.7. PROCEDIMIENTOS

3.7.1. Estrategia de prueba de hipótesis

La investigación no presenta hipótesis al tratarse de una investigación de nivel descriptivo.

3.7.2. Técnicas de procesamiento de datos

A. Técnicas para procesamiento de datos

La elección de las herramientas de ingeniería utilizadas tiene en cuenta su simplicidad y velocidad de desarrollo, lo que facilita la gestión de la complejidad de las aplicaciones, el desarrollo de aplicaciones y los estándares de las aplicaciones, al tiempo que nos permite utilizar la tecnología para proteger la información crítica.

Tabla 2

Herramientas de procesamiento de datos.

SOFTWARE	FABRICANTE	SERVICIO
MAC OS	Apple	Es la versión del sistema operativo Mac para escritorio, ofrece el marco para instalar las herramientas de desarrollo.
Visual Studio Code	Microsoft	Es un entorno de desarrollo integrado para el desarrollo de programas informáticos.
Angular	Google	Angular es una plataforma para crear aplicaciones web, móviles y de escritorio.
Typescript	Microsoft	Typescript es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

Fuente. Elaboración propia.

B. Técnicas para análisis de datos

Los datos obtenidos mediante los instrumentos; encuesta y cuestionario, se analizaron y procesaron usando el marco de trabajo Scrum, con el objetivo de desarrollar los procesos en estudio que nos mostraran datos sobre la seguridad, autenticación y autorización de clúster Kubernetes.

3.7.3. Diseño estadístico

Dado que es una investigación de nivel descriptivo la investigación no posee hipótesis, entonces se presenta la aplicación de gestión de la seguridad de clúster Kubernetes que se muestra en figura y tablas que se muestra en resultados.

3.7.4. Análisis e interpretación de datos

El análisis, recolección e interpretación de la información se realizó utilizando la metodología Scrum.

CAPÍTULO IV
RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

4.1.1. ARTEFACTOS SCRUM

4.1.1.1. PRODUCT BACKLOG

El producto Backlog se determinó en una reunión entre el equipo Scrum y el Product Owner, de dicha reunión se explicaron y detallaron las actividades a realizar para poder tener el producto terminado, dicha lista no está refinada ni priorizada ya que en una primera iteración se busca identificar todas las actividades sin considerar la prioridad ni el puntaje.

Como resultado de la reunión se obtuvo la primera lista de actividades que posteriormente se convertirán en el “Product Backlog”.

Tabla 3

Listado de Actividades.

Nº	HISTORIAS DE USUARIO
01	Como Product Owner, quiero el diseño de la arquitectura de la aplicación para conocer las herramientas y tecnologías que se utilizarán.
02	Como Product Owner, quiero poder crear usuarios a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.
03	Como Product Owner, quiero poder crear grupos a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.
04	Como Product Owner, quiero poder crear Service Accounts a fin de poder desarrollar integraciones automatizadas con nuestro cluster Kubernetes.
05	Como Product Owner, quiero poder aprobar o rechazar las solicitudes de creación de usuarios.
06	Como Product Owner, quiero poder crear roles a nivel de namespace a fin de tener un control y gobierno sobre los objetos Kubernetes existentes.
07	Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del namespace.
08	Como Product Owner, quiero poder editar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.
09	Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.
10	Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.
11	Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.
12	Como Product Owner, quiero poder eliminar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.
13	Como Product Owner, quiero poder asignar los clsuter roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del cluster.
14	Como Product Owner, quiero poder ver los recursos existentes en cluster, así como las versiones existentes y los permisos que se les puede aplicar, a fin de poder crear roles bien definidos.

Fuente. Elaboración propia.

Una vez definido el listado de actividades a realizar, el Product Owner debe de definir prioridades de acuerdo con sus necesidades e importancia, esta priorización es realizado con la ayuda del Scrum Master, de dicha reunión se tuvo como resultado la siguiente tabla:

Tabla 4

Listado de Actividades Priorizadas.

Nº	HISTORIAS DE USUARIO	PRIORIDAD
01	Como Product Owner, quiero el diseño de la arquitectura de la aplicación para conocer las herramientas y tecnologías que se utilizarán.	1
02	Como Product Owner, quiero poder crear usuarios a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	2
03	Como Product Owner, quiero poder crear grupos a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	3
04	Como Product Owner, quiero poder crear Service Accounts a fin de poder desarrollar integraciones automatizadas con nuestro cluster Kubernetes.	4
05	Como Product Owner, quiero poder aprobar o rechazar las solicitudes de creación de usuarios.	5
06	Como Product Owner, quiero poder crear roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	6
07	Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del namespace.	7
08	Como Product Owner, quiero poder editar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	8
09	Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	9
10	Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	10
11	Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	11
12	Como Product Owner, quiero poder eliminar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	12
13	Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del cluster.	13
14	Como Product Owner, quiero poder ver los recursos existentes en cluster, así como las versiones existentes y los permisos que se les puede aplicar, a fin de poder crear roles bien definidos.	14

Fuente. Elaboracion propia.

El siguiente paso fue asignar las puntuaciones respectivas a cada historia, esta tarea se realizó en conjunto entre el Scrum Master, Product Owner y el Equipo Scrum. Para el cálculo de la puntuación se usó el sistema de numeración Fibonacci dando como resultado la tabla de esfuerzo.

Tabla 5

Puntuación de Actividades.

Nº	HISTORIAS DE USUARIO	PRIORIDAD	ESFUERZO
01	Como Product Owner, quiero el diseño de la arquitectura de la aplicación para conocer las herramientas y tecnologías que se utilizarán.	1	8
02	Como Product Owner, quiero poder crear usuarios a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	2	5
03	Como Product Owner, quiero poder crear grupos a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	3	5
04	Como Product Owner, quiero poder crear Service Accounts a fin de poder desarrollar integraciones automatizadas con nuestro cluster Kubernetes.	4	5
05	Como Product Owner, quiero poder aprobar o rechazar las solicitudes de creación de usuarios.	5	5
06	Como Product Owner, quiero poder crear roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	6	5
07	Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y Service Accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del namespace.	7	5
08	Como Product Owner, quiero poder editar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	8	5
09	Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	9	5
10	Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	10	5
11	Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	11	5
12	Como Product Owner, quiero poder eliminar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	12	5
13	Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del cluster.	13	5
14	Como Product Owner, quiero poder ver los recursos existentes en cluster, así como las versiones existentes y los permisos que se les puede aplicar, a fin de poder crear roles bien definidos.	14	3

Fuente. Elaboración propia.

El “Product Backlog” obtenido fue usado a lo largo de los 5 Sprints, en la cual al ser de naturaleza flexible este puede cambiarse de acuerdo con las prioridades del Product Owner.

4.1.1.2. ITERACIONES

4.1.1.2.1. SPRINT 01

En este primer sprint se revisaron los elementos del Backlog y se seleccionaron las historias que se realizaran de acuerdo con las necesidades y prioridades del Product Owner, de esta revisión se obtuvo el entregable del sprint 01 (Tabla 6).

Tabla 6

Entregables del Sprint 01.

Nº	HISTORIAS DE USUARIO	ESFUERZO
01	Como Product Owner, quiero el diseño de la arquitectura de la aplicación para conocer las herramientas y tecnologías que se utilizarán.	8
02	Como Product Owner, quiero poder crear usuarios a fin de que pueden ejecutar comandos kubectl de acuerdo con el permiso asignado.	5

Fuente. Elaboración propia.

La reunión del primer Sprint duró aproximada 2 horas, donde se clarificaron los objetivos y el contexto de cada elemento. A continuación, se realizó la planificación detallada de las tareas a fin de determinar cómo implementar los elementos seleccionados del sprint.

Tabla 7

Estimación de tiempo disponible para el Sprint 01.

Duración del Sprint			2 Semanas
Días efectivos del Sprint			10 días
Miembros del equipo	**Días disponibles**	**Horas disponibles por día**	**Total Horas Sprint**
Christian Altamirano Ayala	10	8	80

Fuente Elaboración propia.

El siguiente paso una vez identificado y planificado la estimación de los tiempos, así como los recursos disponibles asignados al proyecto es la elaboración de las tareas individuales por Historia.

Figura 7

Sprint Backlog del Sprint 01.

Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero el diseño de la arquitectura de la aplicación para conocer las herramientas y tecnologías que se utilizarán.	Definir Arquitectura de la Solucion	CAA	3	1									
	Seleccionar Tecnologia a utilizar.	CAA	3										
	Documentar la Arquitectura	CAA	2										
Como Product Owner, quiero poder crear usuarios a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
		Real	13										
		Deseado	13										

Fuente. Elaboración propia.

Una vez definido el Sprint Backlog del Sprint 01, el equipo se encarga de desarrollar las tareas y actualizar el tablero de la Figura 7 una vez concluida una tarea. Para controlar y monitorear el avance de los compromisos del Sprint el equipo realiza revisiones diarias de las tareas a todo el equipo, esta actividad es llamada el “Daily Scrum”, esta actividad tiene como objetivo revisar el avance diario de las historias y resolver posibles impedimentos, para ello se realizan reuniones diarias de no más de 15 minutos donde el equipo básicamente realiza las siguientes 3 preguntas:

a. ¿Qué se hizo desde la última reunión?
b. ¿Qué tiene planificado hacer el día de hoy?
c. ¿Qué impedimentos se tiene para poder culminar las tareas?

A medida que el equipo fue reportando los avances diarios de las tareas, el Scrum Master junto con el Producto Owner mapean dicho avance en el Sprint Backlog, la Figura 8 representa el mapeo de las tareas del Sprint 01 a final de cierre del Sprint, así mismo el avance se puede observar en la gráfica de “Burndown Chart” de la figura 9.

Figura 8

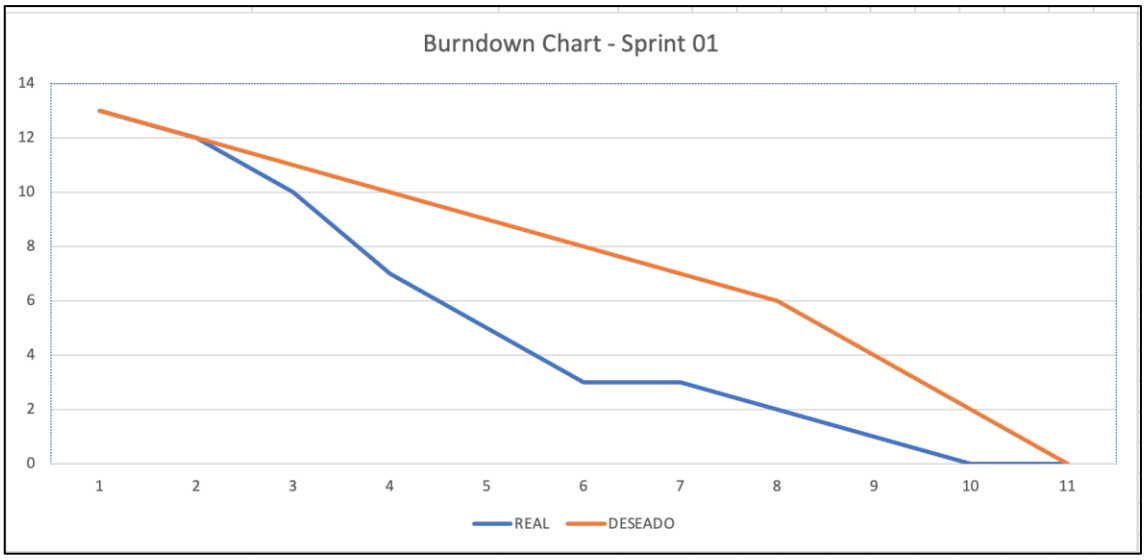
Avance Sprint Backlog del Sprint 01.

Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero el diseño de la arquitectura de la aplicación para conocer las herramientas y tecnologías que se utilizarán.	Definir Arquitectura de la Solucion	CAA	3	1	1	1							
	Seleccionar Tecnologia a utilizar.	CAA	3		1	1	1						
	Documentar la Arquitectura	CAA	2			1	1						
Como Product Owner, quiero poder crear usuarios a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	Elaboracion de interaz	CAA	2					1		1			
	Implementacion de API	CAA	2					1			1		
	Prueba de Funcionalidad	CAA	1									1	
		Real	13	1	2	3	2	2	0	1	1	1	
		Deseado	13	12	10	7	5	3	3	2	1	0	0

Fuente Elaboración propia.

Figura 9

Gráfica Burndown Chart del Sprint 01.

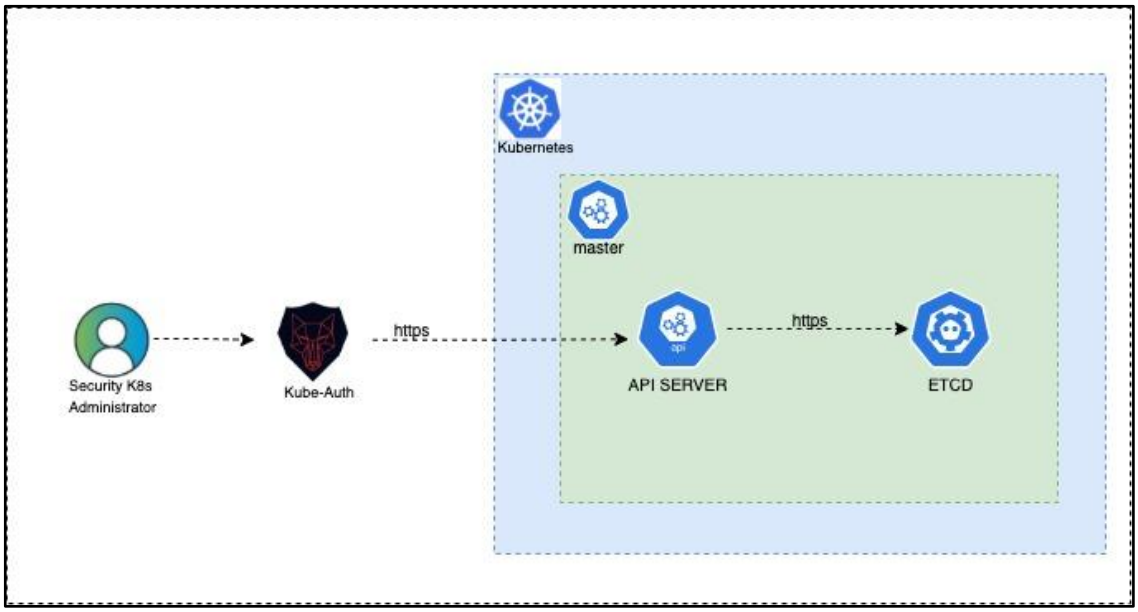


Fuente. Elaboración propia.

Del Gráfico de la Figura 8 y Figura 9 se puede observar que se logró completar las historias dentro de los plazos previstos, esto debido al gran conocimiento que se tiene sobre las tecnologías involucradas. Al final de Sprint se obtuvo como resultados los siguientes: Arquitectura de la solución (Figura 10), estructura del empaquetamiento de la solución (Figura 11) y la implementación de la creación de usuarios (Figuras 12, 13 y 14).

Figura 10

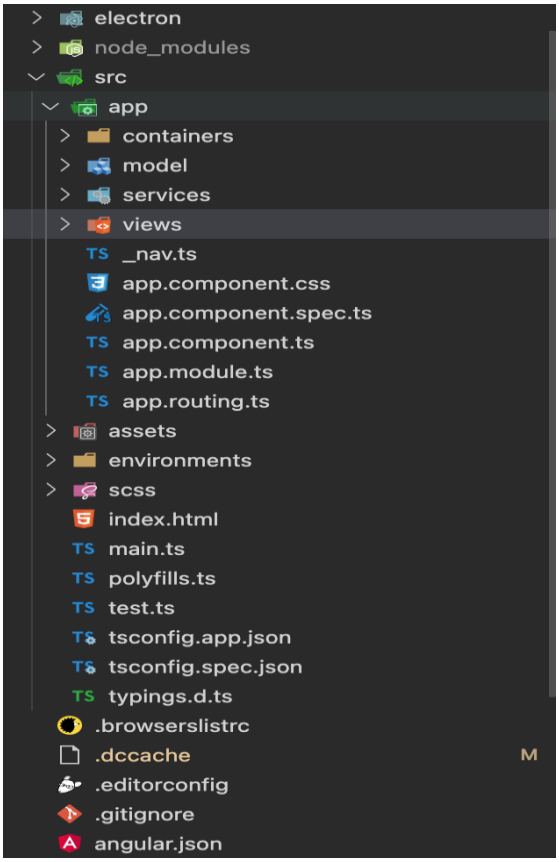
Arquitectura de la Solución.



Fuente. Elaboración propia.

Figura 11

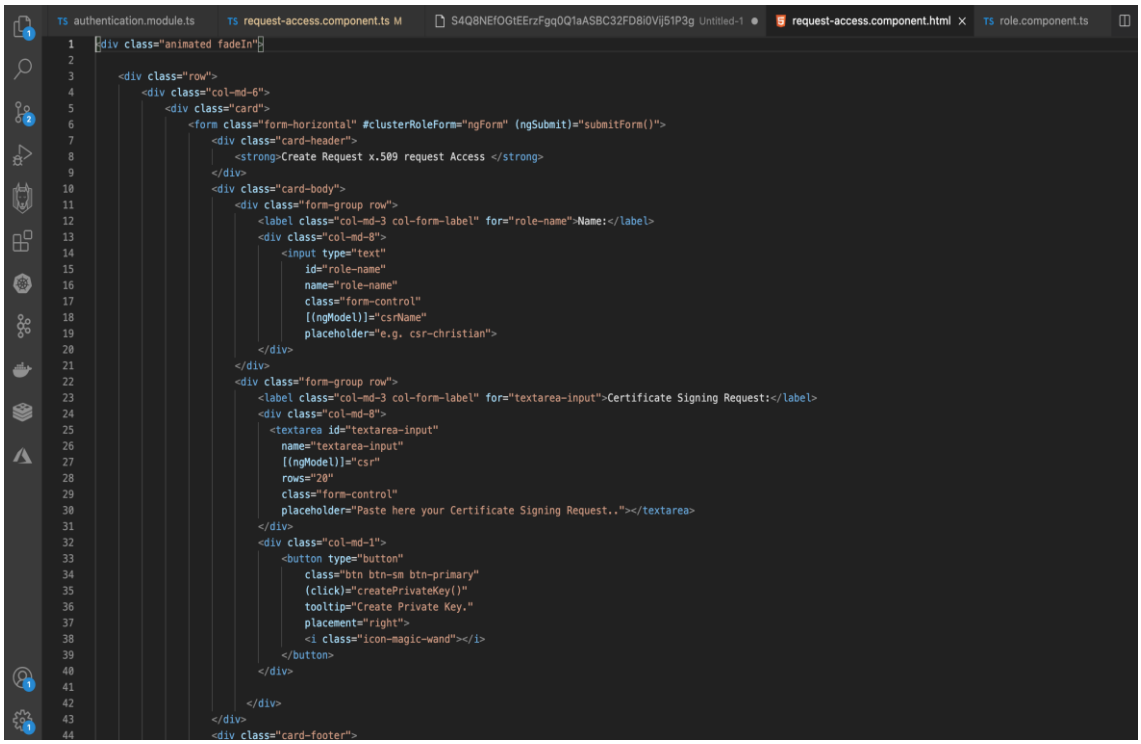
Estructura del proyecto.



Fuente. Elaboración propia.

Figura 12

Implementación Vista de creación de usuarios.



Fuente. Elaboración propia.

Figura 13

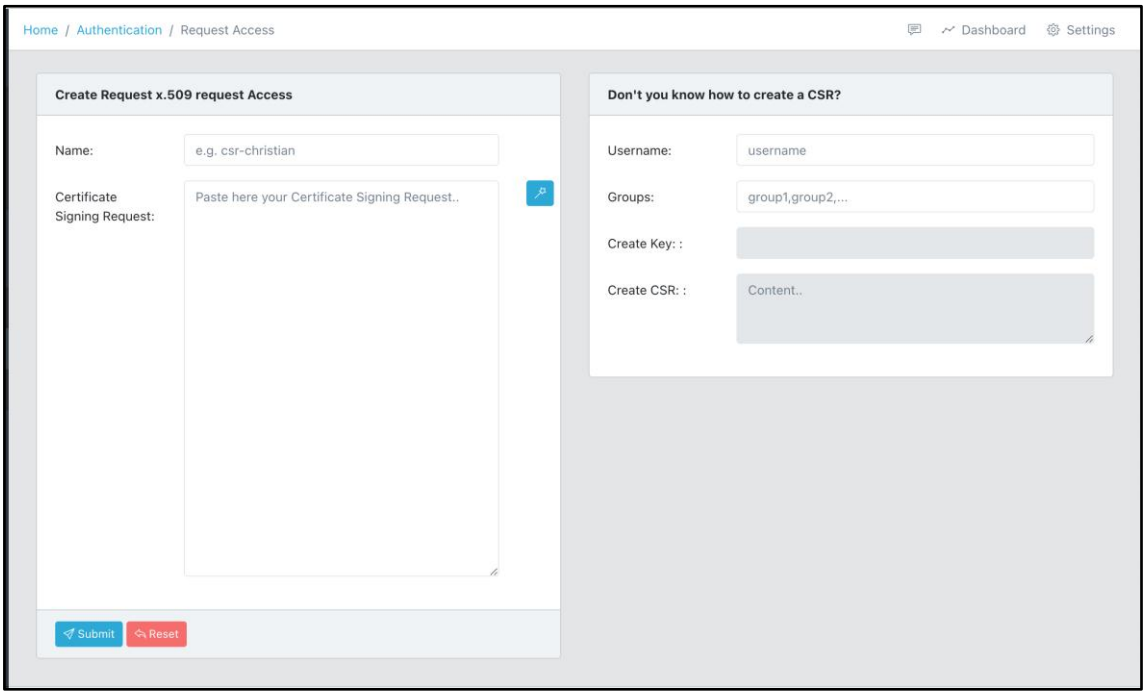
Implementación de controlador de creación de usuarios.

```
import { Component, OnInit } from '@angular/core';
import { ToastrService } from 'ngx-toastr';
import { CertificateSigningRequest } from '../../../../model/certificate-signing-request';
import { CertificateSigningRequestSpec } from '../../../../model/certificate-signing-request-spec';
import { Metadata } from '../../../../model/metadata';
import { CsrService } from '../../../../services/csr.service';

const SIGNER_NAME="kubernetes.io/kube-apiserver-client"

@Component({
  selector: 'app-request-access',
  templateUrl: './request-access.component.html'
})
export class RequestAccessComponent implements OnInit {



  csr: string;
  csrName: string;
  username: string
  groups: string
  opensslcsrcommand: string = ""
  opensslkeycommand: string = ""

  constructor(private toastr: ToastrService,
    private csrService: CsrService) {
  }
  ngOnInit(): void {
  }


  async createPrivateKey(){

  }

  updateCreateKeyCommand(){
    if (this.username){
      this.opensslkeycommand="openssl genrsa -out user-"+this.username+".key 2048"
    }
  }

```

Fuente. Elaboración propia.

Figura 14

Interfaz de creación de usuarios.



Fuente. Elaboración propia.

4.1.1.2.2. SPRINT 02

En este primer sprint se revisaron los elementos del Backlog y se seleccionaron las historias que se realizaran de acuerdo con las necesidades y prioridades del Product Owner, de esta revisión se obtuvo el entregable del sprint 02 (Tabla 8).

Tabla 8

Entregables del Sprint 02.

HISTORIAS DE USUARIO	ESFUERZO
Como Product Owner, quiero poder crear grupos a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	5
Como Product Owner, quiero poder crear Service Accounts a fin de poder desarrollar integraciones automatizadas con nuestro cluster Kubernetes.	5
Como Product Owner, quiero poder aprobar o rechazar las solicitudes de creación de usuarios.	5

Fuente. Elaboración propia.

La reunión del segundo sprint tuvo una duración aproximada de 2 horas, donde se clarifican los objetivos y el contexto de cada elemento. A continuación, se realizan la

planificación detallada de las tareas para saber cómo implementar los elementos seleccionados en el sprint.

Tabla 9

Estimación de tiempo disponible para el Sprint 02.

Duración del Sprint			2 Semanas
Días efectivos del Sprint			10 días
Miembros del equipo	**Días disponibles**	**Horas disponibles por día**	**Total Horas Sprint**
Christian Altamirano Ayala	10	8	80

Fuente. Elaboración propia

El siguiente paso una vez identificado y planificado la estimación de los tiempos, así como los recursos disponibles asignados al proyecto es la elaboración de las tareas individuales por Historia.

Figura 15

Sprint Backlog del Sprint 02.

Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder crear grupos a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder crear Service Accounts a fin de poder desarrollar integraciones automatizadas con nuestro cluster Kubernetes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder aprobar o rechazar las solicitudes de creación de usuarios.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
		Real	15	15	15	15	15	15	15	15	15	15	15
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración propia.

Una vez definido el Sprint Backlog del Sprint 02, el equipo realizó los Dayli Scrum con el fin de revisar el avance de las historias y resolver posibles impedimentos, en las reuniones diarias de no más de 15 minutos el equipo básicamente realiza las siguientes 3 preguntas:

a. Que se hizo desde la última reunión
b. Que tiene planificado hacer el día de hoy
c. Que impedimentos se tiene para poder culminar las tareas

Esta activad se lleva a cabo de manera repetitiva a lo largo del Sprint, que en este caso fue de 10 días. A medida que el equipo fue reportando los avances diarios de las tareas, el Scrum Master junto con el Producto Owner mapean dicho avance en el Sprint Backlog de la Figura 16.

Figura 16

Avance Sprint Backlog del Sprint 02.

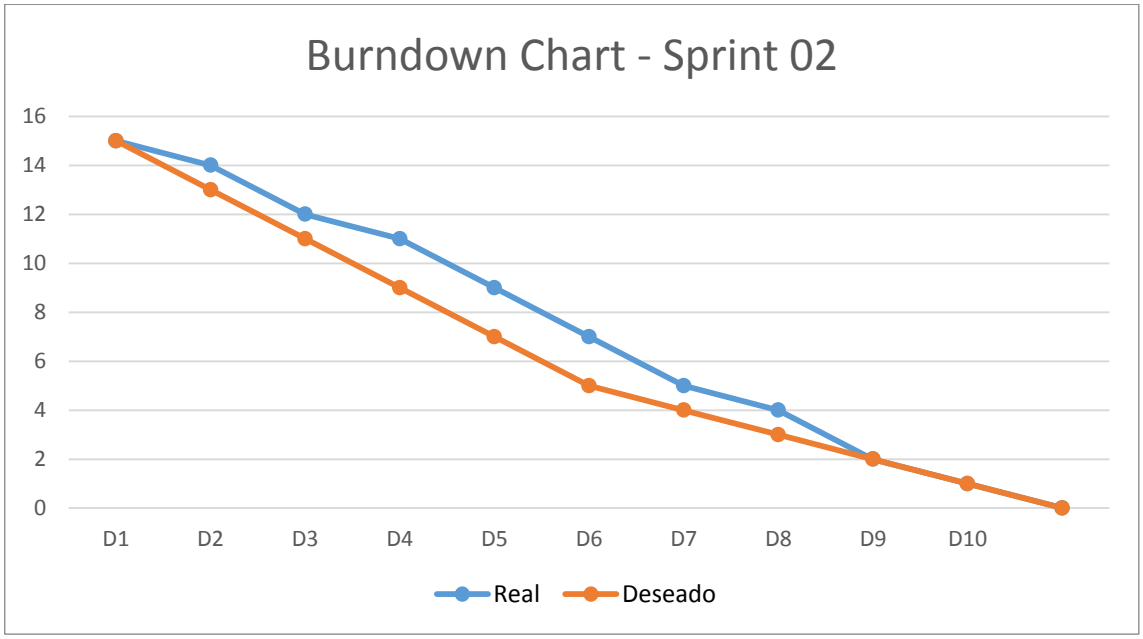
Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder crear grupos a fin de que pueden ejecutar comandos kubectl de acuerdo al permiso asignado.	Elaboracion de interaz	CAA	2	1	1								
	Implementacion de API	CAA	2		1	1							
	Prueba de Funcionalidad	CAA	1				1						
Como Product Owner, quiero poder crear Service Accounts a fin de poder desarrollar integraciones automatizadas con nuestro cluster Kubernetes.	Elaboracion de interaz	CAA	2				1	1					
	Implementacion de API	CAA	2					1	1				
	Prueba de Funcionalidad	CAA	1						1				
Como Product Owner, quiero poder aprobar o rechazar las solicitudes de creación de usuarios.	Elaboracion de interaz	CAA	2							1	1		
	Implementacion de API	CAA	2								1	1	
	Prueba de Funcionalidad	CAA	1										1
		Real	15	14	12	11	9	7	5	4	2	1	0
		Deseado	15	13	11	9	7	5	4	3	2	1	0

Fuente. Elaboración propia

Así mismo el avance se puede observar en la gráfica Burndown Chart de la figura 17.

Figura 17

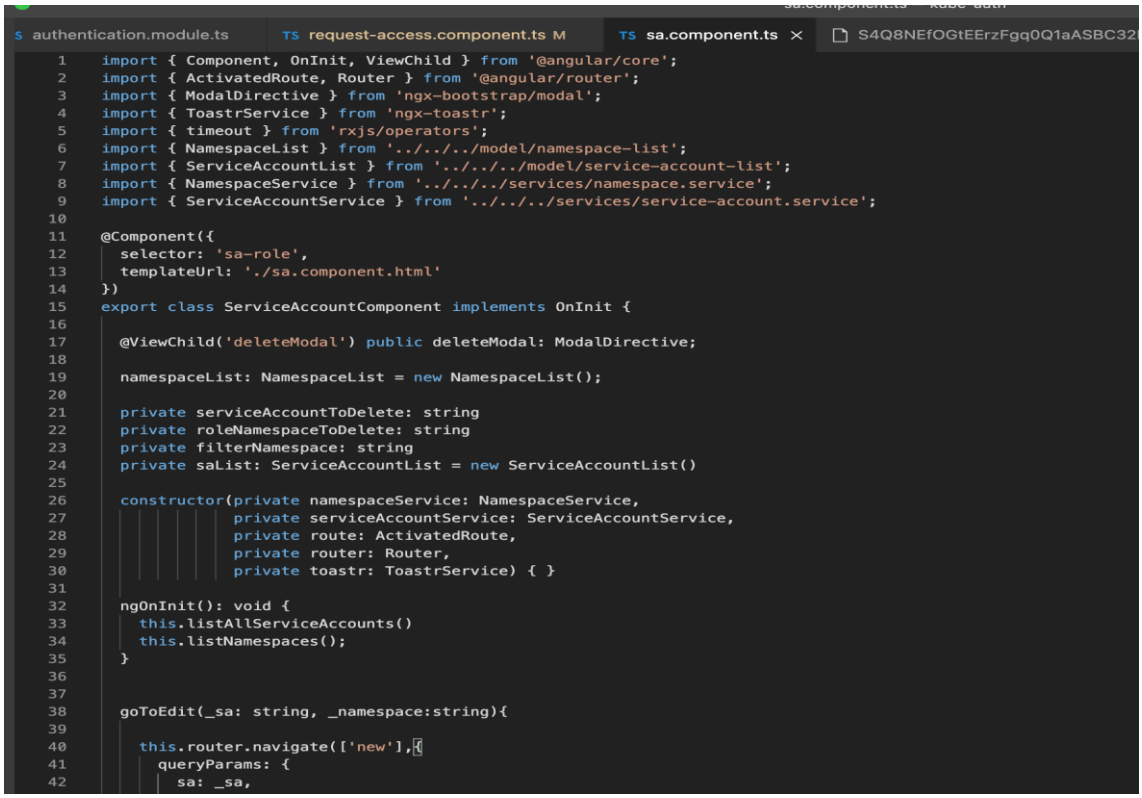
Gráfica BurnDown Chart del Sprint 02.



Fuente. Elaboración propia

Figura 18

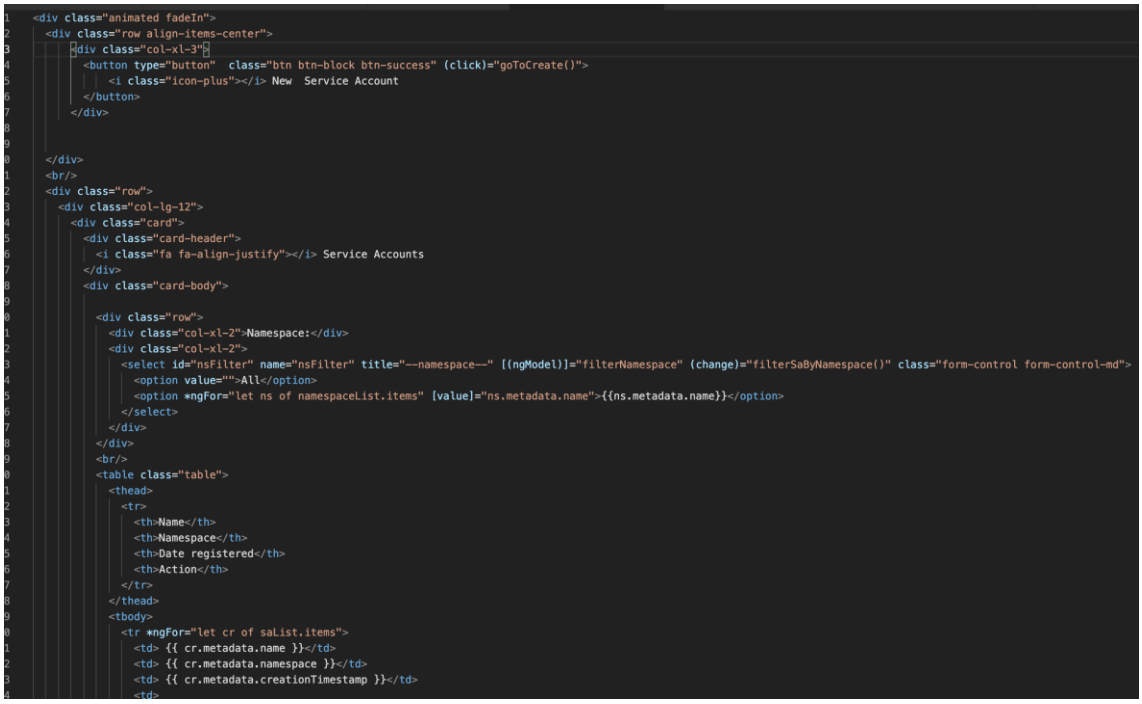
Controlador para gestionar Service Accounts.



Fuente. Elaboración propia

Figura 19

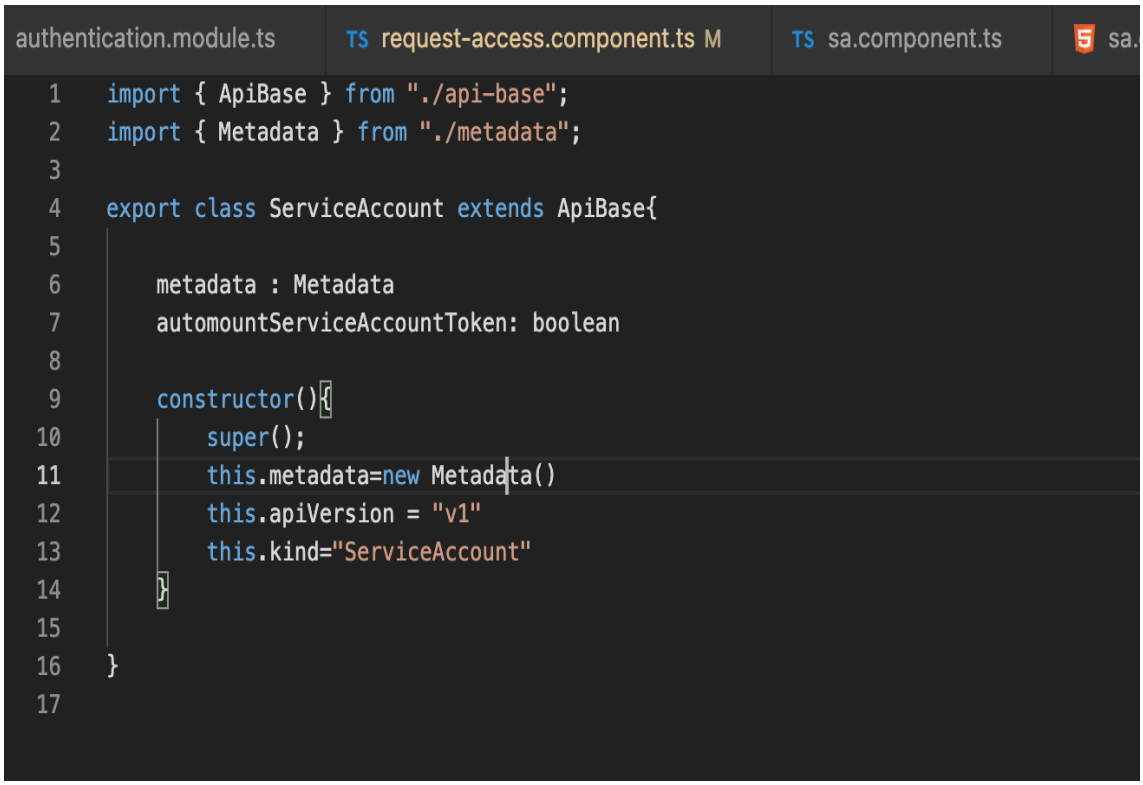
Implementación Vista para gestión de Service Accounts.



Fuente. Elaboración propia

Figura 20

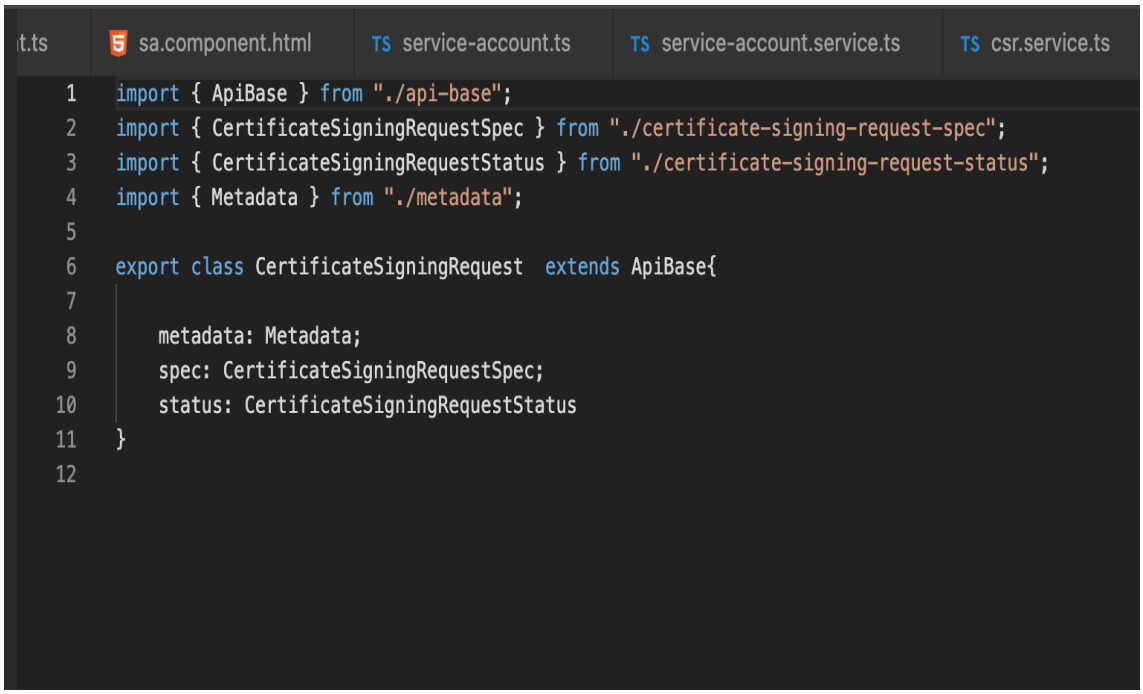
Implementación Modelo de Service Accounts.



Fuente. Elaboración propia.

Figura 21

Implementación Modelo para CSR.



Fuente. Elaboración propia

Figura 22

Implementación cliente REST para Service Account.

```
export class ServiceAccountService {

  constructor(private http:HttpClient) {

  }

  listServiceAccount(): Observable<HttpResponse<ServiceAccountList>> {
    return this.http.get<ServiceAccountList>(API_BASE+'/api/v1/serviceaccounts',
      {headers,observe: 'response'}
    );
  }

  listServiceAccountByNamespace(namespace:string): Observable<HttpResponse<ServiceAccountList>> {
    return this.http.get<ServiceAccountList>(API_BASE+'/api/v1/namespaces/'+namespace+'/serviceaccounts',
      {headers,observe: 'response'}
    );
  }


  findServiceAccount(name:string , namespace: string): Observable<HttpResponse<ServiceAccount>> {
    return this.http.get<ServiceAccount>(API_BASE+'/api/v1/namespaces/'+namespace+'/serviceaccounts/'+name,
      {headers,observe: 'response'}
    );
  }

  deleteServiceAccount(name: string,namespace:string): Observable<HttpResponse<any>> {
    return this.http.delete<any>(API_BASE+'/api/v1/namespaces/'+namespace+'/serviceaccounts/'+name,
      {headers,observe: 'response'}
      );
  }

  createServiceAccount(sa: ServiceAccount): Observable<HttpResponse<ServiceAccount>> {
    return this.http.post<ServiceAccount>(API_BASE+'/api/v1/namespaces/'+sa.metadata.namespace+'/serviceaccounts',
      sa,
      {headers,observe: 'response'}
    );
  }

  updateServiceAccount(sa: ServiceAccount): Observable<HttpResponse<ServiceAccount>> {
    return this.http.put<ServiceAccount>(API_BASE+'/api/v1/namespaces/'+sa.metadata.namespace+'/serviceaccounts/'+sa.metadata.name,
      sa,
      {headers,observe: 'response'}
    );
```

Fuente. Elaboración propia.

Figura 23

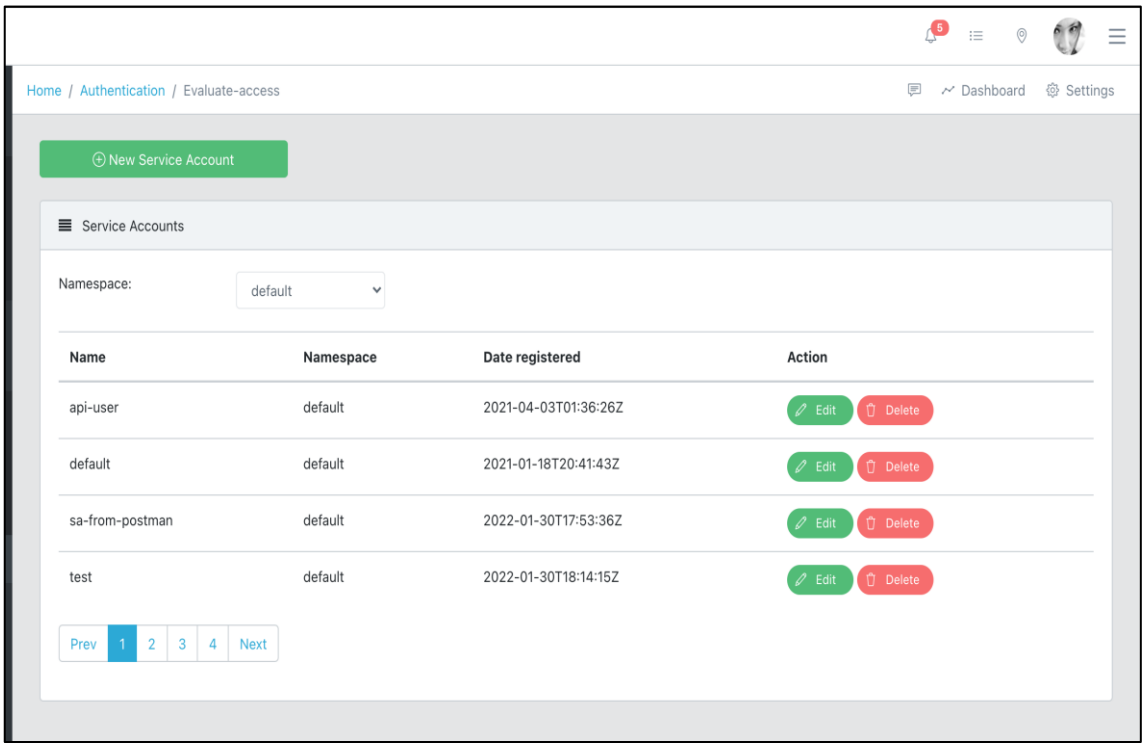
Implementación cliente REST para CSR.

```
@Injectable({
  providedIn: 'root'
})
export class CsrService {

  constructor(private http:HttpClient) {

  }

  listCertificateSigningRequest(): Observable<HttpResponse<CertificateSigningRequestList>> {
    return this.http.get<CertificateSigningRequestList>(API_BASE+'/apis/certificates.k8s.io/v1/certificatesigningrequests',
      {
        headers,
        observe: 'response'
      }
    );
  }

  createCertificateSigningRequest(csr:CertificateSigningRequest): Observable<HttpResponse<CertificateSigningRequest>> {
    return this.http.post<CertificateSigningRequest>(API_BASE+'/apis/certificates.k8s.io/v1/certificatesigningrequests',
      csr,
      {
        headers,
        observe: 'response'
      }
    );
  }

  deleteCertificateSigningRequest(name:string): Observable<HttpResponse<any>> {
    return this.http.delete<any>(API_BASE+'/apis/certificates.k8s.io/v1/certificatesigningrequests/'+name,
      {
        headers,
        observe: 'response'
      }
    );
  }
```

Fuente. Elaboración propia.

Figura 24

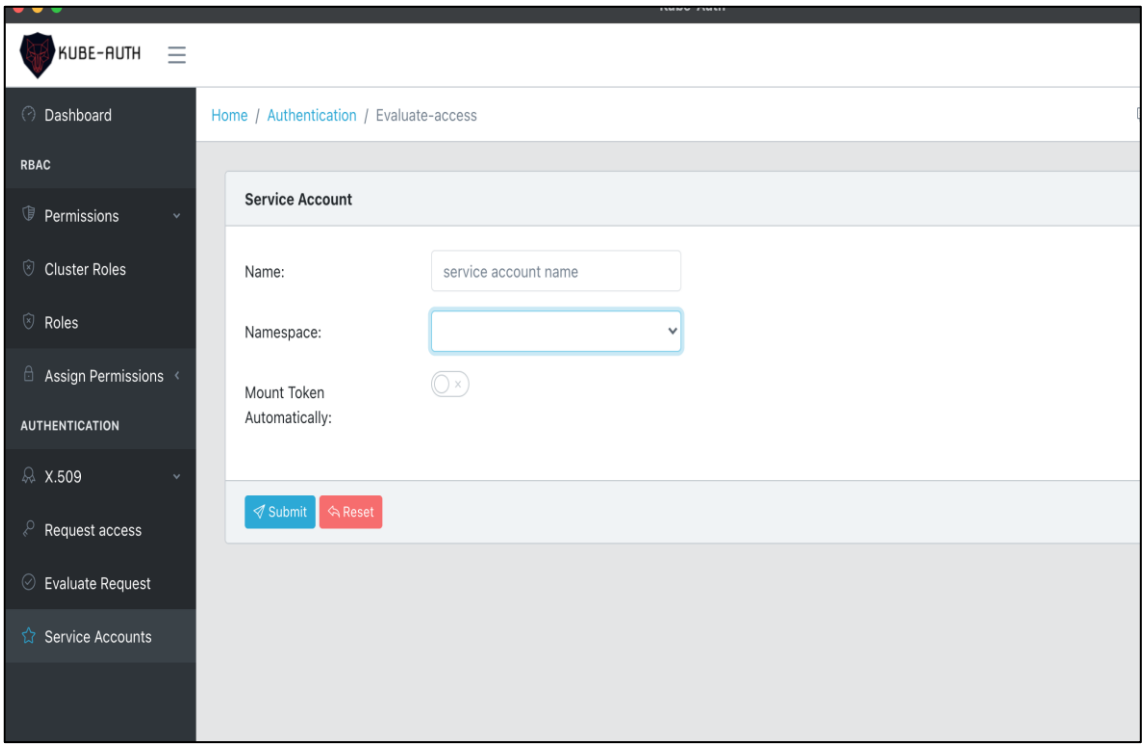
Interfaz de listado de Service Account.



Fuente. Elaboración propia.

Figura 25

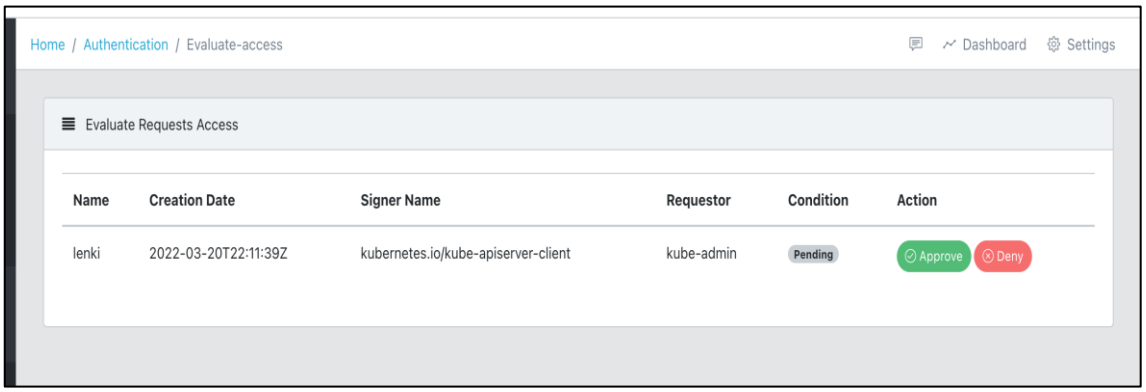
Interfaz creación de Service Account.



Fuente. Elaboración propia.

Figura 26

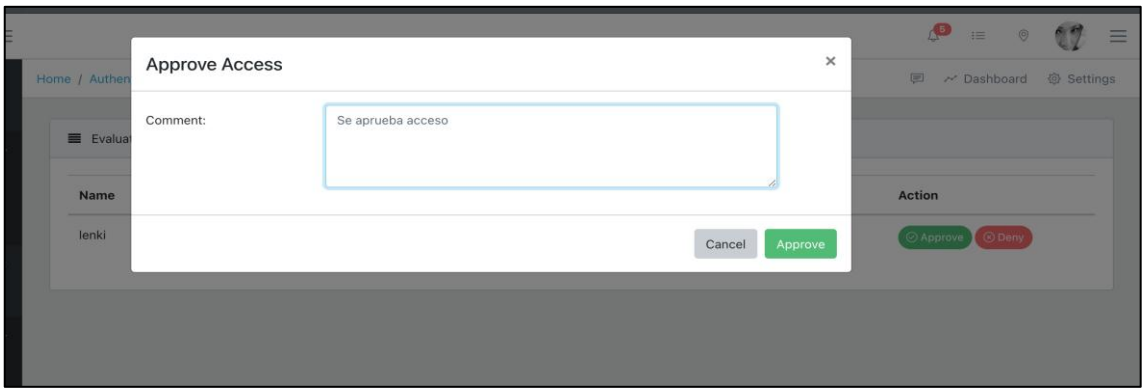
Interfaz listado de CSR.



Fuente. Elaboración propia.

Figura 27

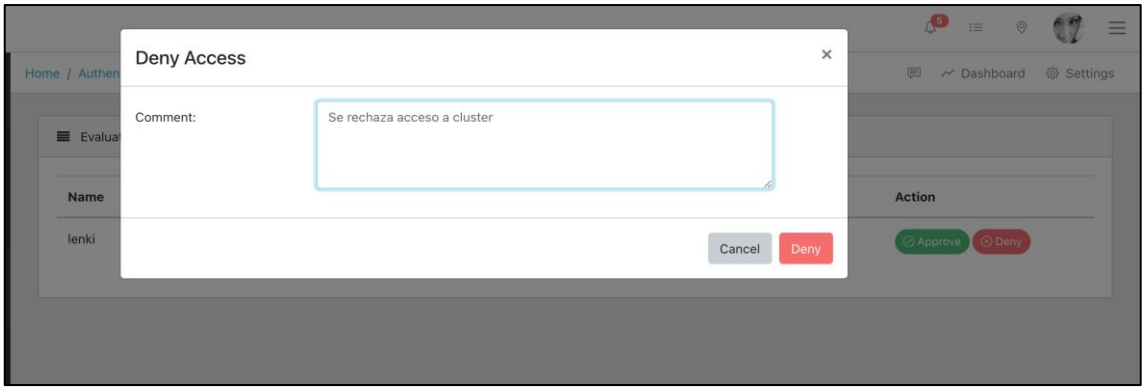
Interfaz aprobación de CSR.



Fuente. Elaboración propia.

Figura 28

Interfaz rechazo CSR.



Fuente. Elaboración propia.

4.1.1.2.3. SPRINT 03

En este primer sprint se revisaron los elementos del Backlog y se seleccionaron las historias que se realizaran de acuerdo con las necesidades y prioridades del Product Owner, de esta revisión se obtuvo el entregable del sprint 03 (Tabla 10).

Tabla 10

Entregables del Sprint 03.

HISTORIAS DE USUARIO	ESFUERZO
Como Product Owner, quiero poder crear roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5
Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del namespace.	5
Como Product Owner, quiero poder editar roles a nivel de Namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5

Fuente. Elaboración propia.

La reunión del tercer sprint tuvo una duración aproximada de 2 horas, donde se clarifican los objetivos y el contexto de cada elemento. A continuación, se realizan la planificación detallada de las tareas para saber cómo implementar los elementos seleccionados en el sprint.

Tabla 11

Estimación de tiempo disponible del Sprint 03.

Duración del Sprint			2 Semanas
Días efectivos del Sprint			10 días
Miembros del equipo	**Días disponibles**	**Horas disponibles por día**	**Total Horas Sprint**
Christian Altamirano Ayala	10	8	80

Fuente. Elaboración propia.

El siguiente paso una vez identificado y planificado la estimación de los tiempos, así como los recursos disponibles asignados al proyecto es la elaboración de las tareas individuales por Historia.

Figura 29

Sprint Backlog del Sprint 03.

Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder crear roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del namespace.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder editar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
		Real	15	15	15	15	15	15	15	15	15	15	15
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración propia.

Una vez definido el Sprint Backlog del Sprint 03, el equipo realizó los Dayli Scrum con el fin de revisar el avance de las historias y resolver posibles impedimentos, en las reuniones diarias de no más de 15 minutos el equipo básicamente realiza las siguientes 3 preguntas:

a. Que se hizo desde la última reunión
b. Que tiene planificado hacer el día de hoy
c. Que impedimentos se tiene para poder culminar las tareas

Esta activad se lleva a cabo de manera repetitiva a lo largo del Sprint, que en este caso fue de 10 días. A medida que el equipo fue reportando los avances diarios de las tareas, el Scrum Master junto con el Producto Owner mapean dicho avance en el Sprint Backlog de la Figura 30.

Figura 30

Avance Sprint Backlog del Sprint 03.

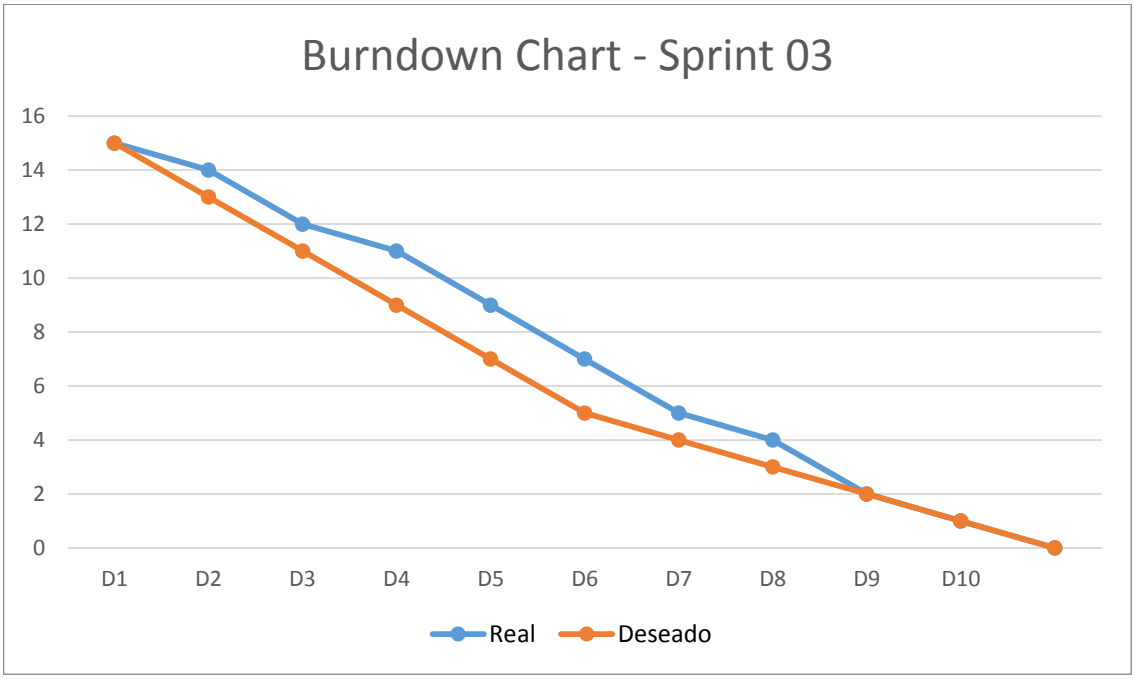
Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder crear roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2	1	1								
	Implementacion de API	CAA	2		1	1							
	Prueba de Funcionalidad	CAA	1				1						
Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del namespace.	Elaboracion de interaz	CAA	2				1	1					
	Implementacion de API	CAA	2					1	1				
	Prueba de Funcionalidad	CAA	1							1			
Como Product Owner, quiero poder editar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2							1	1		
	Implementacion de API	CAA	2								1	1	
	Prueba de Funcionalidad	CAA	1										1
		Real	15	14	12	11	9	7	6	4	2	1	0
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración propia.

Así mismo el avance se puede observar en la gráfica Burndown Chart de la figura 31.

Figura 31

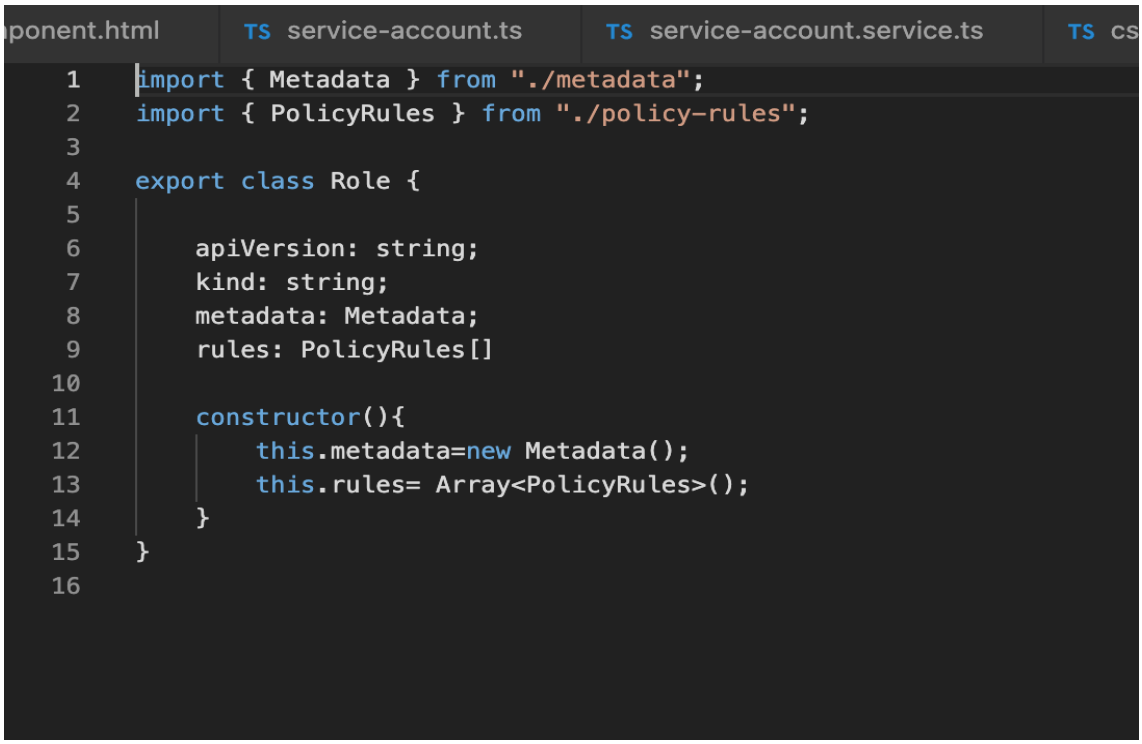
Gráfica BurnDown Chart del Sprint 03.



Fuente. Elaboración propia.

Figura 32

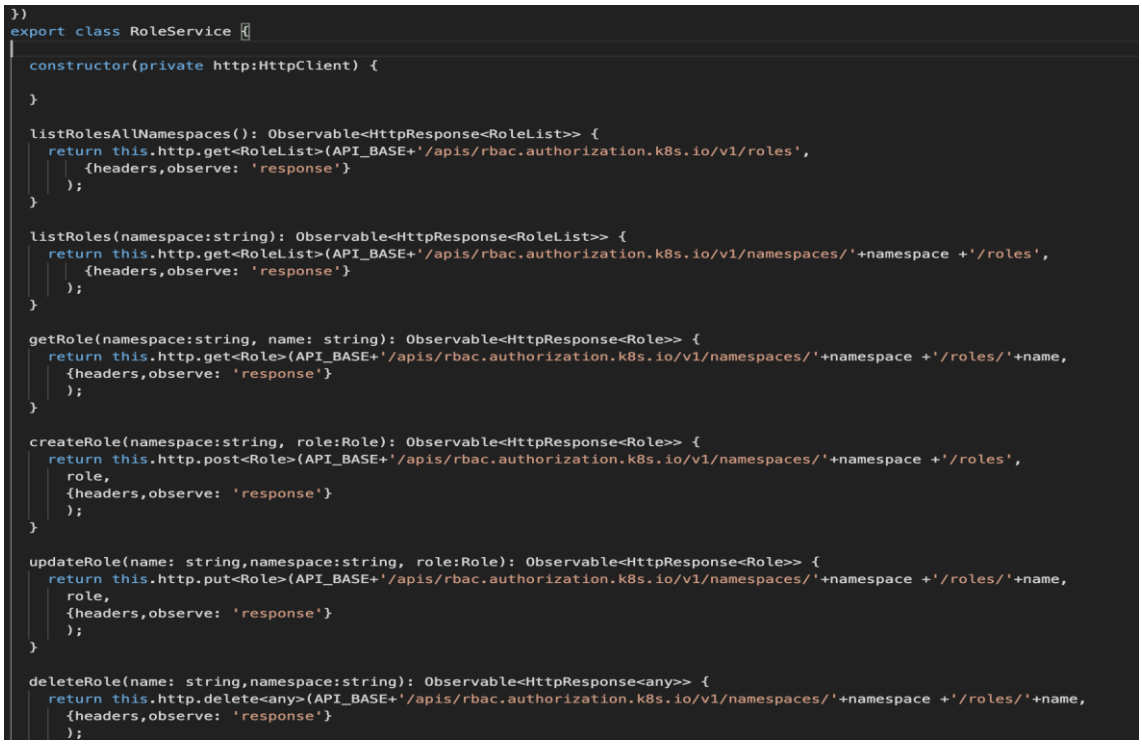
Implementación de clase Role.



Fuente. Elaboración propia.

Figura 33

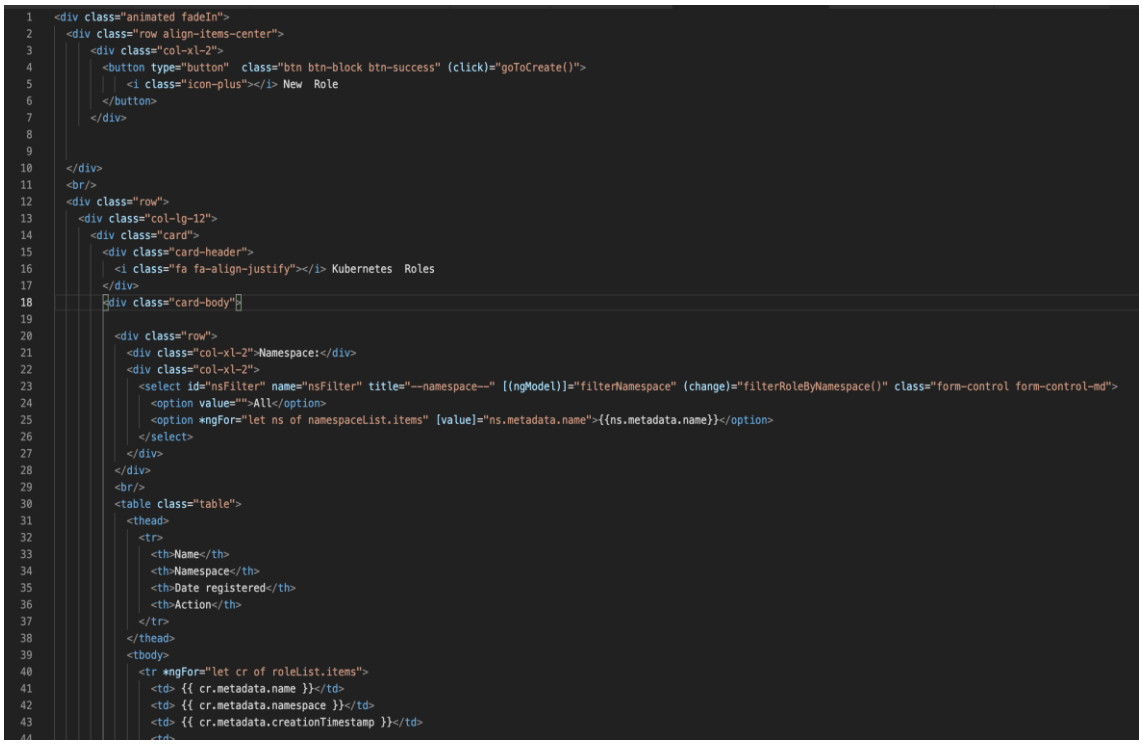
Implementación cliente REST para Roles.



Fuente. Elaboración propia.

Figura 34

Implementación Vista de Roles.



```
<div class="animated fadeIn">
  <div class="row align-items-center">
      <div class="col-xl-2">
        <button type="button"  class="btn btn-block btn-success" (click)="goToCreate()">
            <i class="icon-plus"></i> New  Role
        </button>
      </div>


  </div>
  <br/>
  <div class="row">
    <div class="col-lg-12">
      <div class="card">
        <div class="card-header">
          <i class="fa fa-align-justify"></i> Kubernetes  Roles
        </div>
        <div class="card-body">

          <div class="row">
            <div class="col-xl-2">Namespace:</div>
            <div class="col-xl-2">
              <select id="nsFilter" name="nsFilter" title="--namespace--" [(ngModel)]="filterNamespace" (change)="filterRoleByNamespace()" class="form-control form-control-md">
                <option value="">All</option>
                <option *ngFor="let ns of namespaceList.items" [value]="ns.metadata.name">{{ns.metadata.name}}</option>
              </select>
            </div>
          </div>
          <br/>
          <table class="table">
            <thead>
              <tr>
                <th>Name</th>
                <th>Namespace</th>
                <th>Date registered</th>
                <th>Action</th>
              </tr>
            </thead>
            <tbody>
              <tr *ngFor="let cr of roleList.items">
                <td> {{ cr.metadata.name }}</td>
                <td> {{ cr.metadata.namespace }}</td>
                <td> {{ cr.metadata.creationTimestamp }}</td>
```

Fuente. Elaboración propia.

Figura 35

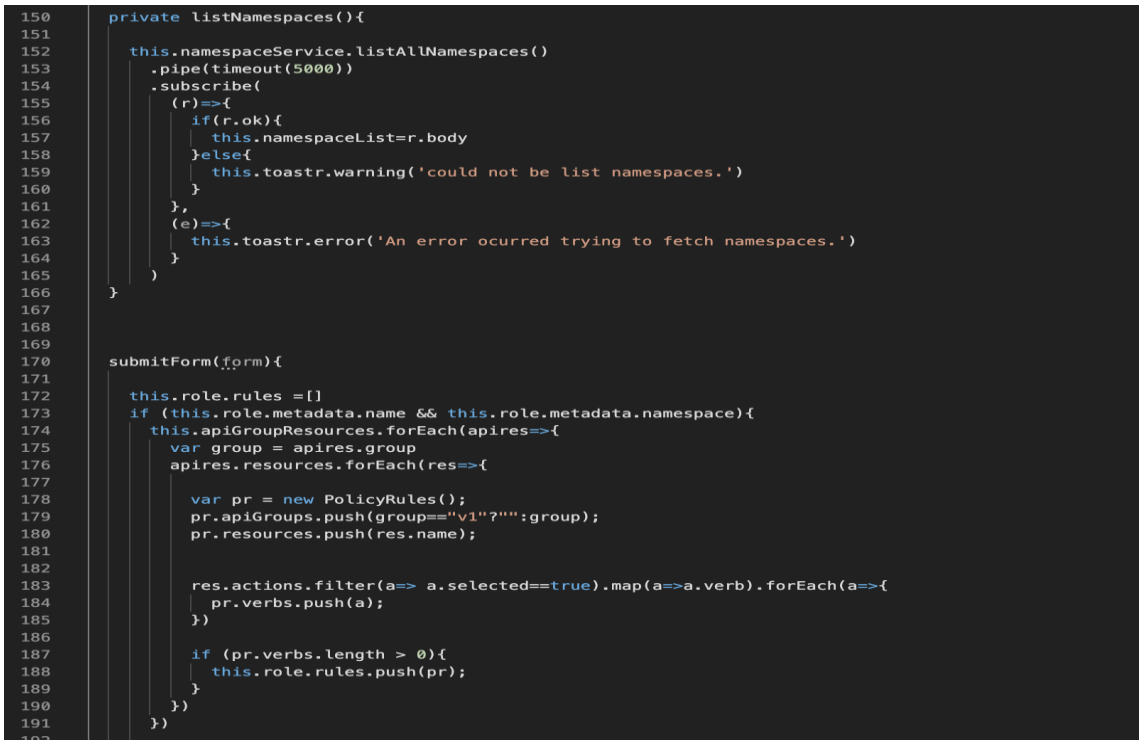
Implementación de controlador de Roles.

```
@Component({
  selector: 'app-role-new',
  templateUrl: './role-new.component.html',
})

export class RoleNewComponent implements OnInit {

  apiGroupList: ApiGroupList;
  apiGroupResources:  ApiGroupResource[] = []
  apiGroupResourcesTmp:  ApiGroupResource[] = []
  role: Role=new Role();
  private isEditRole: boolean =false
  filterNamespace: string
  namespaceList: NamespaceList = new NamespaceList();

  constructor(private roleService: RoleService,
    private namespaceService: NamespaceService,
    private apiGroupService: ApiGroupService,
    private toastr: ToastrService,
    private route: ActivatedRoute) { }



  async ngOnInit(){

    var cr = this.route.snapshot.queryParamMap.get('role')
    var ns = this.route.snapshot.queryParamMap.get('namespace')

    this.listNamespaces();

    if (cr && ns){
      this.isEditRole=true
      await this.getAllClusterResources(true,cr,ns);
    }else{
      await this.getAllClusterResources(false,cr,null);
    }

  }

  async  getAllClusterResources(isEdit:boolean,_clusterRole:string,_namespace:string){
```

Fuente. Elaboración propia.

Figura 36

Implementación de controlador de Roles II.

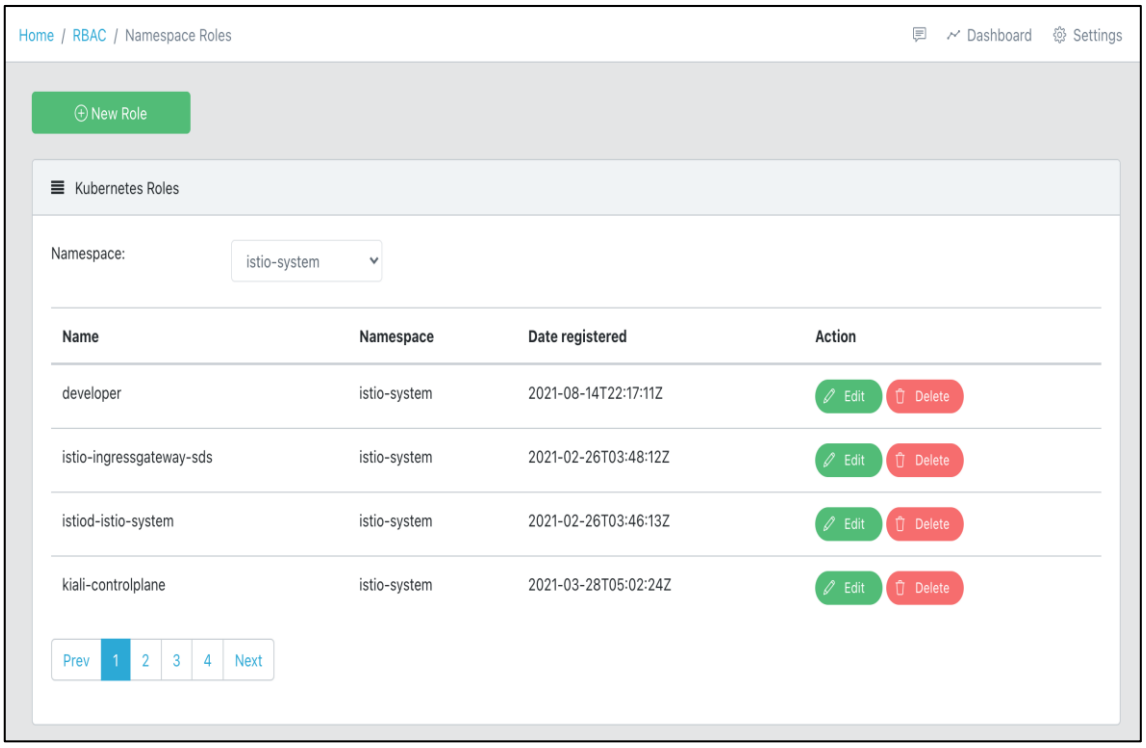


```
150   private listNamespaces(){
151
152     this.namespaceService.listAllNamespaces()
153       .pipe(timeout(5000))
154       .subscribe(
155         (r)=>{
156           if(r.ok){
157             this.namespaceList=r.body
158           }else{
159             this.toastr.warning('could not be list namespaces.')
160           }
161         },
162         (e)=>{
163           this.toastr.error('An error ocurred trying to fetch namespaces.')
164         }
165       )
166   }
167
168
169
170   submitForm(form){
171
172     this.role.rules =[]
173     if (this.role.metadata.name && this.role.metadata.namespace){
174       this.apiGroupResources.forEach(apires=>{
175         var group = apires.group
176         apires.resources.forEach(res=>{
177
178           var pr = new PolicyRules();
179           pr.apiGroups.push(group=="v1"?"":group);
180           pr.resources.push(res.name);
181
182
183           res.actions.filter(a=> a.selected==true).map(a=>a.verb).forEach(a=>{
184             pr.verbs.push(a);
185           })
186
187           if (pr.verbs.length > 0){
188             this.role.rules.push(pr);
189           }
190         })
191       })
```

Fuente. Elaboración Propia.

Figura 37

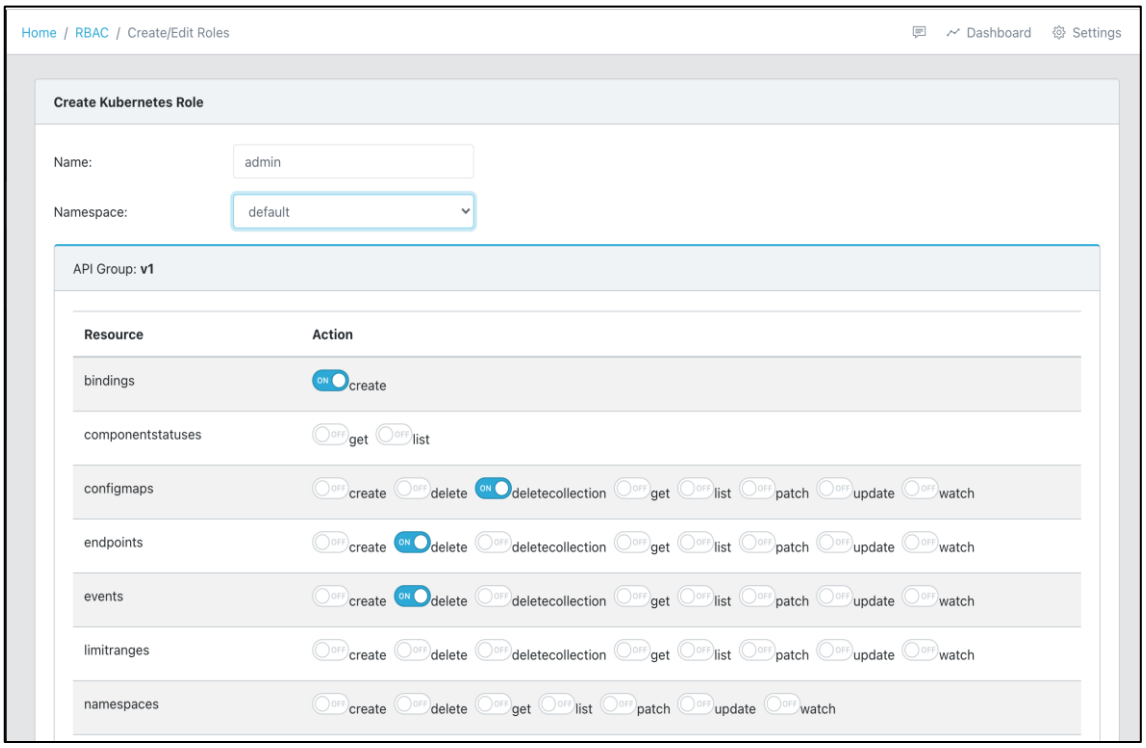
Interfaz listado de Roles.



Fuente. Elaboración Propia.

Figura 38

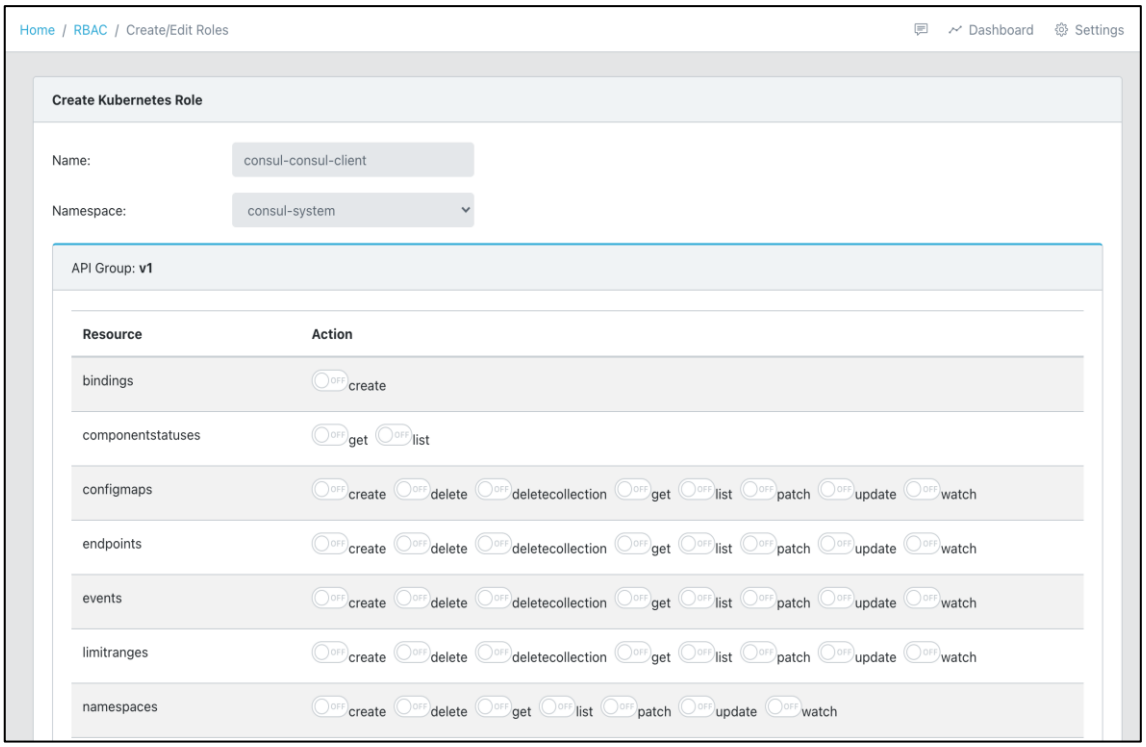
Interfaz creación de Roles.



Fuente. Elaboración Propia.

Figura 39

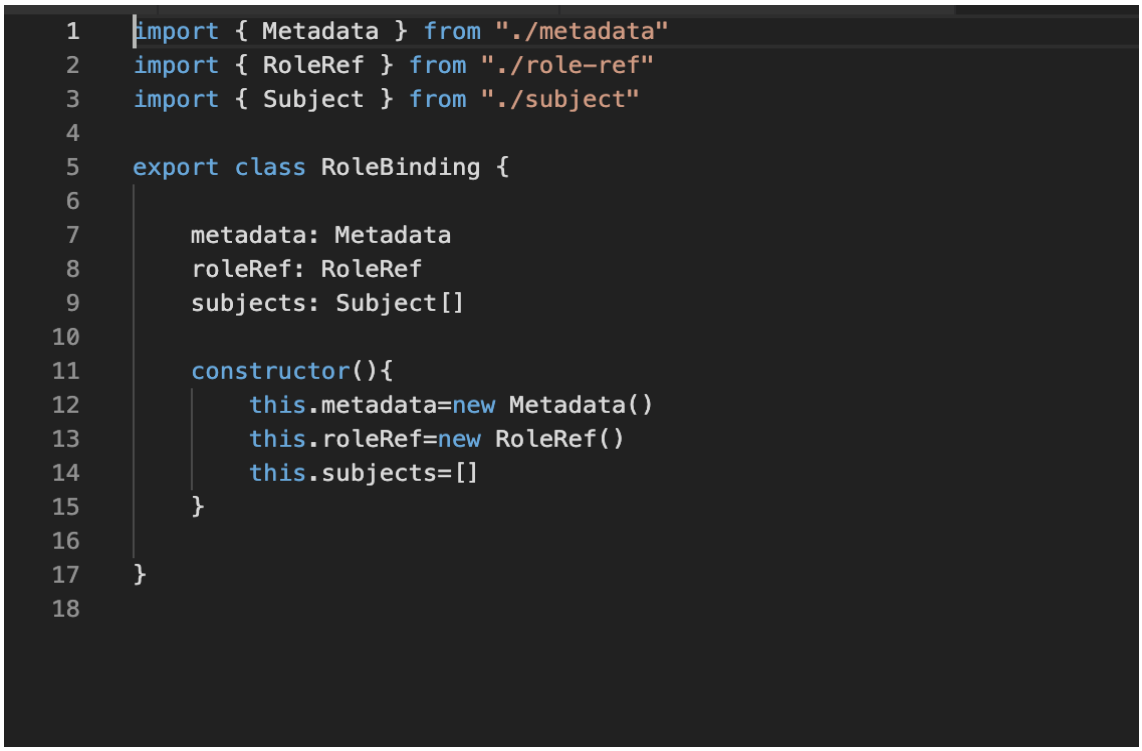
Interfaz actualización de Roles.



Fuente. Elaboración Propia.

Figura 40

Implementación Modelo para Asignación de Roles.

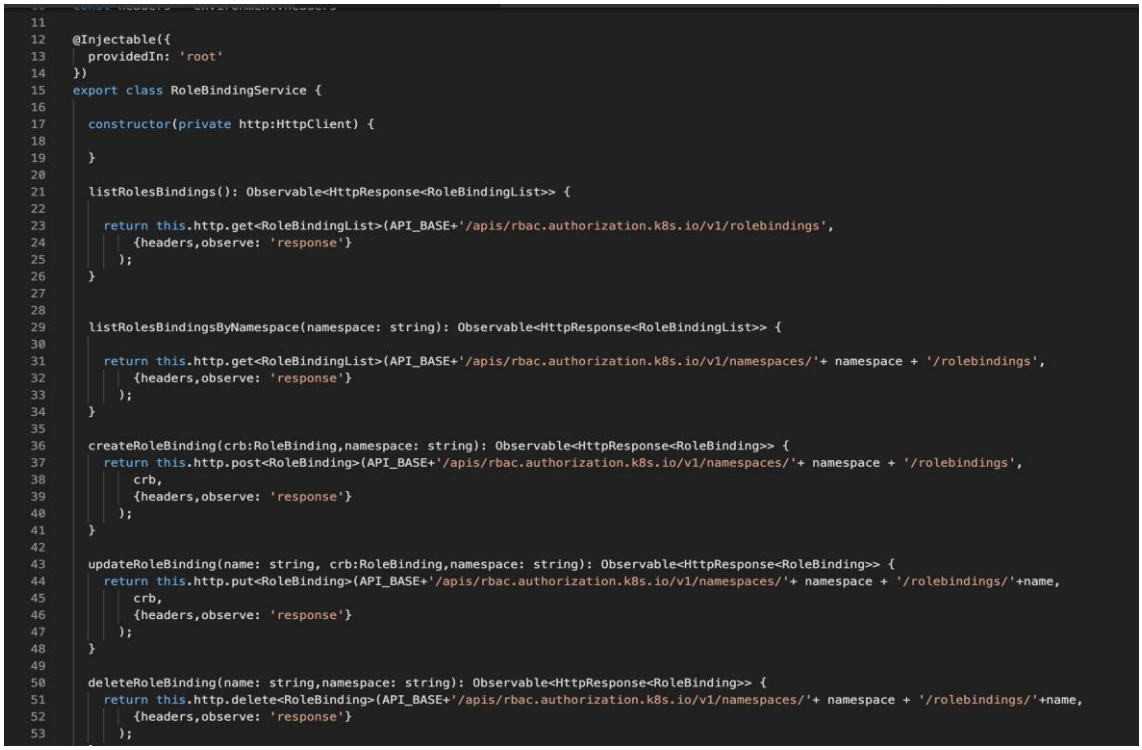


```
import { Metadata } from "./metadata"
import { RoleRef } from "./role-ref"
import { Subject } from "./subject"

export class RoleBinding {

    metadata: Metadata
    roleRef: RoleRef
    subjects: Subject[]

    constructor(){
        this.metadata=new Metadata()
        this.roleRef=new RoleRef()
        this.subjects=[]
    }

}
```

Fuente. Elaboración Propia.

Figura 41

Implementación cliente REST para asignación de Roles.

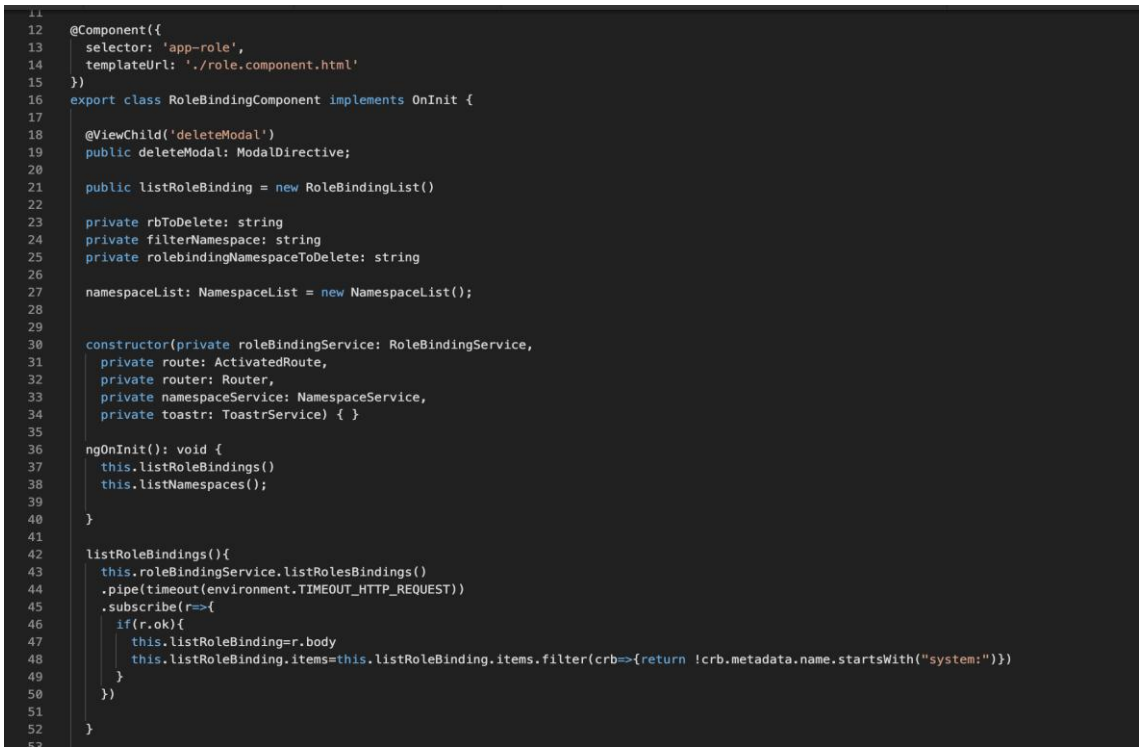


```
@Injectable({
  providedIn: 'root'
})
export class RoleBindingService {

  constructor(private http:HttpClient) {

  }

  listRolesBindings(): Observable<HttpResponse<RoleBindingList>> {

    return this.http.get<RoleBindingList>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/rolebindings',
      {headers,observe: 'response'}
    );
  }


  listRolesBindingsByNamespace(namespace: string): Observable<HttpResponse<RoleBindingList>> {

    return this.http.get<RoleBindingList>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+ namespace + '/rolebindings',
      {headers,observe: 'response'}
    );
  }

  createRoleBinding(crb:RoleBinding,namespace: string): Observable<HttpResponse<RoleBinding>> {
    return this.http.post<RoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+ namespace + '/rolebindings',
      crb,
      {headers,observe: 'response'}
    );
  }

  updateRoleBinding(name: string, crb:RoleBinding,namespace: string): Observable<HttpResponse<RoleBinding>> {
    return this.http.put<RoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+ namespace + '/rolebindings/'+name,
      crb,
      {headers,observe: 'response'}
    );
  }

  deleteRoleBinding(name: string,namespace: string): Observable<HttpResponse<RoleBinding>> {
    return this.http.delete<RoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+ namespace + '/rolebindings/'+name,
      {headers,observe: 'response'}
    );
```

Fuente. Elaboración Propia.

Figura 42

Implementación de controlador de asignación de Roles.

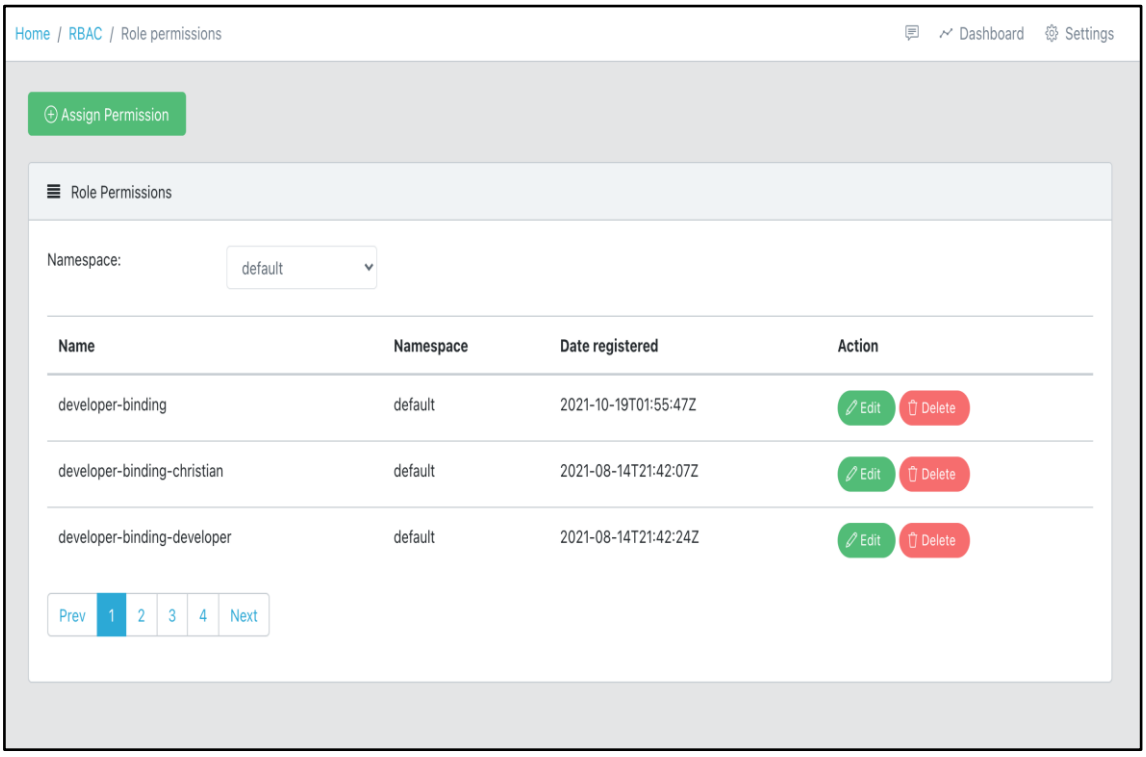


```
@Component({
  selector: 'app-role',
  templateUrl: './role.component.html'
})
export class RoleBindingComponent implements OnInit {

  @ViewChild('deleteModal')
  public deleteModal: ModalDirective;

  public listRoleBinding = new RoleBindingList()

  private rbToDelete: string
  private filterNamespace: string
  private rolebindingNamespaceToDelete: string

  namespaceList: NamespaceList = new NamespaceList();


  constructor(private roleBindingService: RoleBindingService,
    private route: ActivatedRoute,
    private router: Router,
    private namespaceService: NamespaceService,
    private toastr: ToastrService) { }

  ngOnInit(): void {
    this.listRoleBindings()
    this.listNamespaces();

  }

  listRoleBindings(){
    this.roleBindingService.listRolesBindings()
    .pipe(timeout(environment.TIMEOUT_HTTP_REQUEST))
    .subscribe(r=>{
      if(r.ok){
        this.listRoleBinding=r.body
        this.listRoleBinding.items=this.listRoleBinding.items.filter(crb=>{return !crb.metadata.name.startsWith("system:")})
      }
    })

  }
```

Fuente. Elaboración Propia.

Figura 43

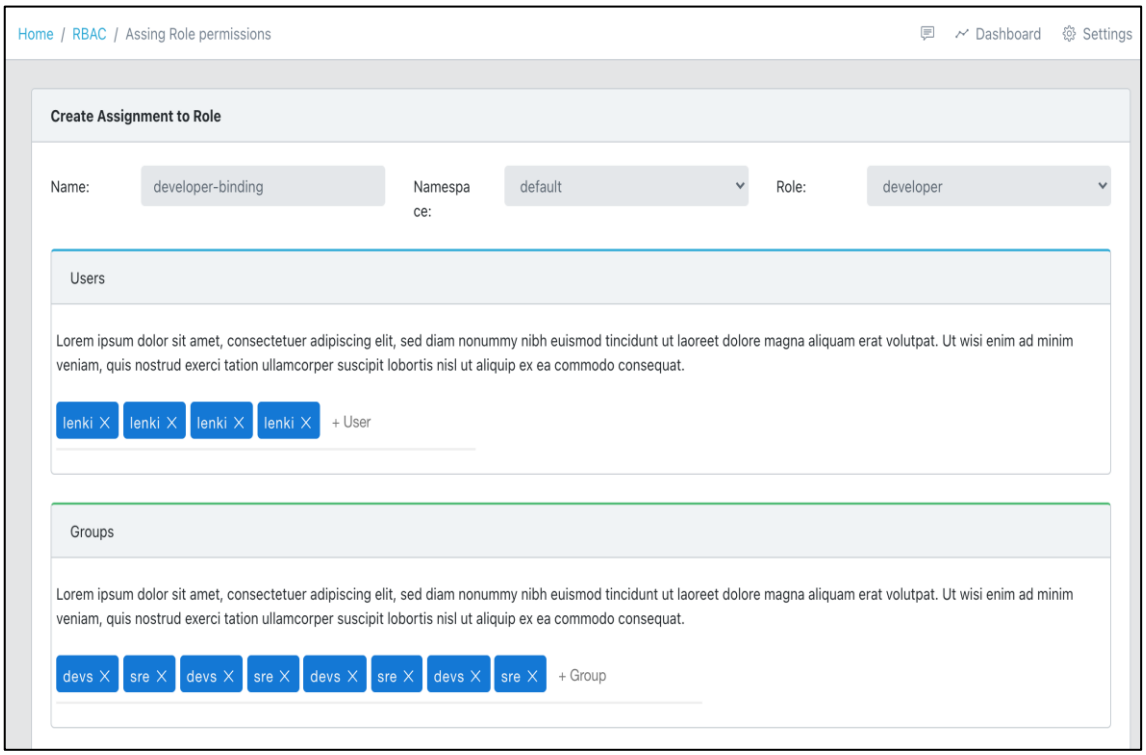
Interfaz listado de asignación de Roles.



Fuente. Elaboración Propia.

Figura 44

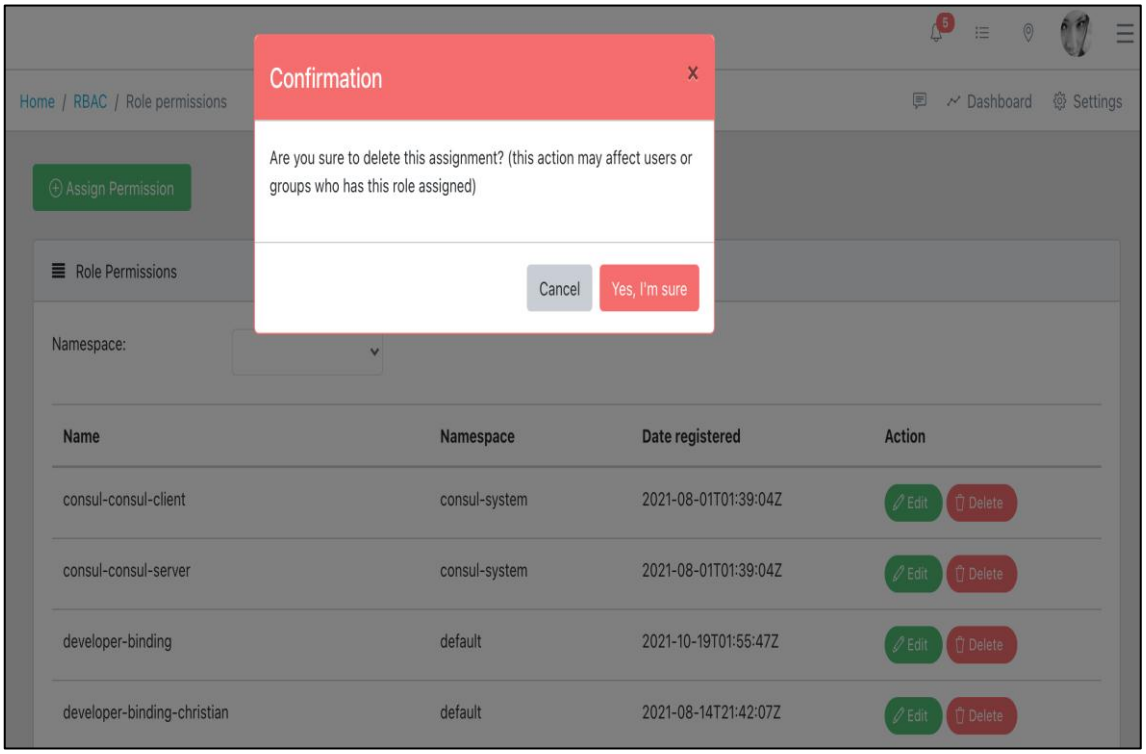
Interfaz creación de asignación de Roles.



Fuente. Elaboración Propia.

Figura 45

Interfaz eliminación de asignación de Roles.



Fuente. Elaboración Propia.

4.1.1.2.4. SPRINT 04

En este primer sprint se revisaron los elementos del Backlog y se seleccionaron las historias que se realizaran de acuerdo con las necesidades y prioridades del Product Owner, de esta revisión se obtuvo el entregable del sprint 04 (Tabla 12).

Tabla 12

Entregables del Sprint 04.

HISTORIAS DE USUARIO	ESFUERZO
Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5
Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5
Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5

Fuente. Elaboración Propia.

La reunión del cuarto sprint tuvo una duración aproximada de 3 horas, donde se clarifican los objetivos y el contexto de cada elemento. A continuación, se realizan la planificación detallada de las tareas para saber cómo implementar los elementos seleccionados en el sprint.

Tabla 13

Estimación tiempo disponible Sprint 04.

Duración del Sprint			2 Semanas
Días efectivos del Sprint			10 días
Miembros del equipo	**Días disponibles**	**Horas disponibles por día**	**Total Horas Sprint**
Christian Altamirano Ayala	10	8	80

Fuente. Elaboración Propia.

El siguiente paso una vez identificado y planificado la estimación de los tiempos, así como los recursos disponibles asignados al proyecto es la elaboración de las tareas individuales por Historia.

Figura 46

Sprint Backlog del Sprint 04.

Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
		Real	15	15	15	15	15	15	15	15	15	15	15
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración Propia.

Una vez definido el Sprint Backlog del Sprint 04, el equipo realizó los Dayli Scrum con el fin de revisar el avance de las historias y resolver posibles impedimentos, en las reuniones diarias de no más de 15 minutos el equipo básicamente realiza las siguientes 3 preguntas:

a. Que se hizo desde la última reunión
b. Que tiene planificado hacer el día de hoy
c. Que impedimentos se tiene para poder culminar las tareas

Esta activad se lleva a cabo de manera repetitiva a lo largo del Sprint, que en este caso fue de 10 días. A medida que el equipo fue reportando los avances diarios de las tareas, el Scrum Master junto con el Producto Owner mapean dicho avance en el Sprint Backlog de la Figura 47.

Figura 47

Avance Sprint Backlog del Sprint 04.

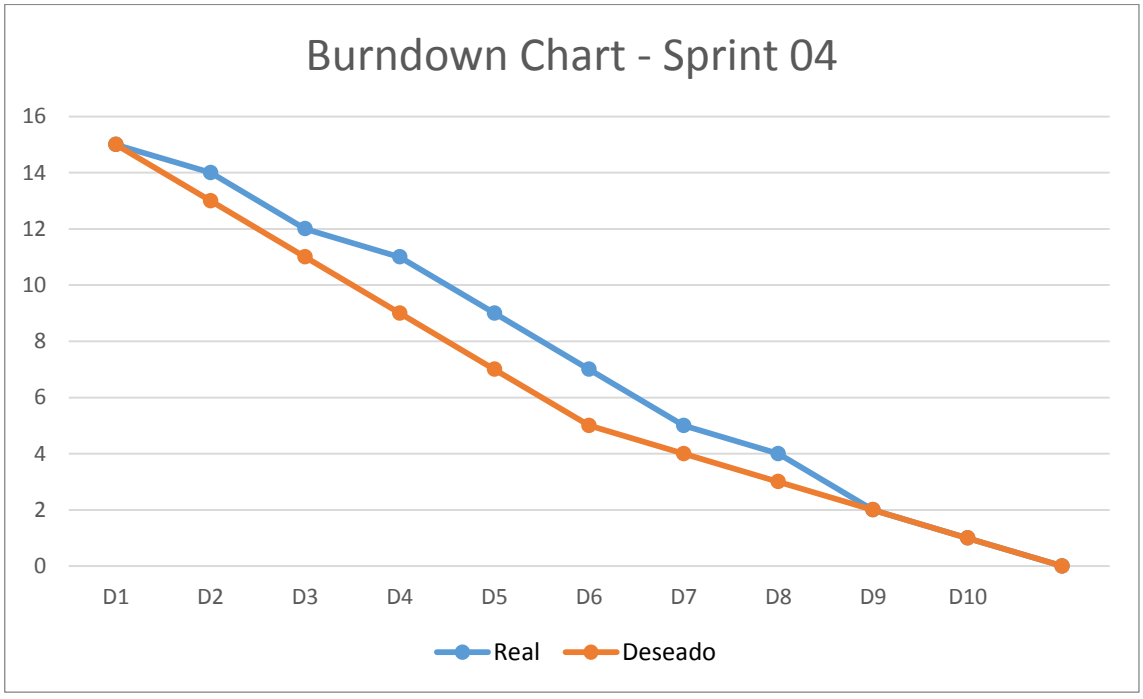
Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2	1	1								
	Implementacion de API	CAA	2		1	1							
	Prueba de Funcionalidad	CAA	1				1						
Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2				1	1					
	Implementacion de API	CAA	2					1	1				
	Prueba de Funcionalidad	CAA	1							1			
Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2							1	1		
	Implementacion de API	CAA	2								1	1	
	Prueba de Funcionalidad	CAA	1										1
		Real	15	14	12	11	9	7	6	4	2	1	0
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración Propia.

Así mismo el avance se puede observar en la gráfica Burndown Chart de la figura 48.

Figura 48

Gráfica BurnDown Chart del Sprint 04.



Fuente. Elaboración Propia.

Figura 49

Implementación vista para asignación de roles.

```
<div bsModal #deleteModal="bs-modal" class="modal fade" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-danger" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h4 class="modal-title">Confirmation</h4>
        <button type="button" class="close" (click)="deleteModal.hide()" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <p>Are you sure to delete this assignment? (this action may affect users or groups who has this role assigned)</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" (click)="deleteModal.hide()">Cancel</button>
        <button type="button" class="btn btn-danger" (click)="deleteRole()">Yes, I'm sure</button>
      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->

</div>
```

Fuente. Elaboración Propia.

Figura 50

Implementación de controlador para eliminar Roles.

```
deleteRole(){
  if (this.rbToDelete && this.rolebindingNamespaceToDelete){
    this.roleBindingService.deleteRoleBinding(this.rbToDelete,this.rolebindingNamespaceToDelete)
    .subscribe((r)=>
      {
        if (r.ok){
          this.listRoleBinding.items = this.listRoleBinding.items.filter(i=>{return i.metadata.name!==this.rbToDelete})
          this.deleteModal.hide()
          this.toastr.success('Assignment deleted succesfully.')
        }else{
          this.toastr.error('An error ocurred trying to delete the role.')
        }
      }, (e)=>{
        this.toastr.error('An error ocurred trying to delete the role.')
    });
  }else{
    this.toastr.warning('no role to be deleted was specified.')
  }
}
```

Fuente. Elaboración Propia.

Figura 51

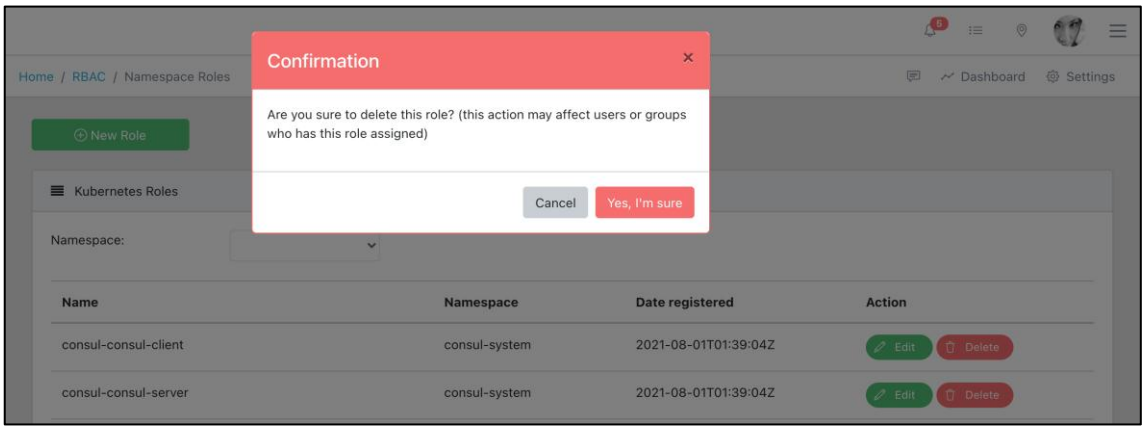
Implementación cliente REST para Roles.

```
createRole(namespace:string, role:Role): Observable<HttpResponse<Role>> {
  return this.http.post<Role>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+namespace +'/roles',
    role,
    {headers,observe: 'response'}
    );
}

updateRole(name: string,namespace:string, role:Role): Observable<HttpResponse<Role>> {
  return this.http.put<Role>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+namespace +'/roles/'+name,
    role,
    {headers,observe: 'response'}
    );
}

deleteRole(name: string,namespace:string): Observable<HttpResponse<any>> {
  return this.http.delete<any>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/namespaces/'+namespace +'/roles/'+name,
    {headers,observe: 'response'}
    );
}
}
```

Fuente. Elaboración Propia.

Figura 52

Interfaz eliminación de Roles.



Fuente. Elaboración Propia.

Figura 53

Implementación modelo de Cluster Roles.

```
1   import { Metadata } from "./metadata";
2   import { PolicyRules } from "./policy-rules";
3
4   export class ClusterRole {
5
6       apiVersion: string;
7       kind: string;
8       metadata: Metadata;
9       rules: PolicyRules[]
10
11      constructor(){
12          this.apiVersion="v1"
13          this.kind="ClusterRole"
14          this.metadata=new Metadata();
15          this.rules= Array<PolicyRules>();
16      }
17  }
18
```

Fuente. Elaboración Propia.

Figura 54

Implementación cliente REST para Cluster Roles.

```
@Injectable({
  providedIn: 'root'
})
export class ClusterRoleService {

  constructor(private http:HttpClient) {

  }

  listClusterRoles(): Observable<HttpResponse<ClusterRoleList>> {

    return this.http.get<ClusterRoleList>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterroles',
      {headers,observe: 'response'}
      );
  }

  createClusterRole(clusterRole:ClusterRole): Observable<HttpResponse<ClusterRole>> {
    return this.http.post<ClusterRole>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterroles',
      clusterRole,
      {headers,observe: 'response'}
      );
  }

  updateClusterRole(name: string, clusterRole:ClusterRole): Observable<HttpResponse<ClusterRole>> {
    return this.http.put<ClusterRole>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterroles/'+name,
      clusterRole,
      {headers,observe: 'response'}
      );
  }

  getClusterRole(name: string): Observable<HttpResponse<ClusterRole>> {
    return this.http.get<ClusterRole>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterroles/'+name,
      {headers,observe: 'response'}
      );
  }

  deleteClusterRole(name: string): Observable<HttpResponse<any>> {
    return this.http.delete<any>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterroles/'+name,
      {headers,observe: 'response'}
      );
  }
}
```

Fuente. Elaboración Propia.

Figura 55

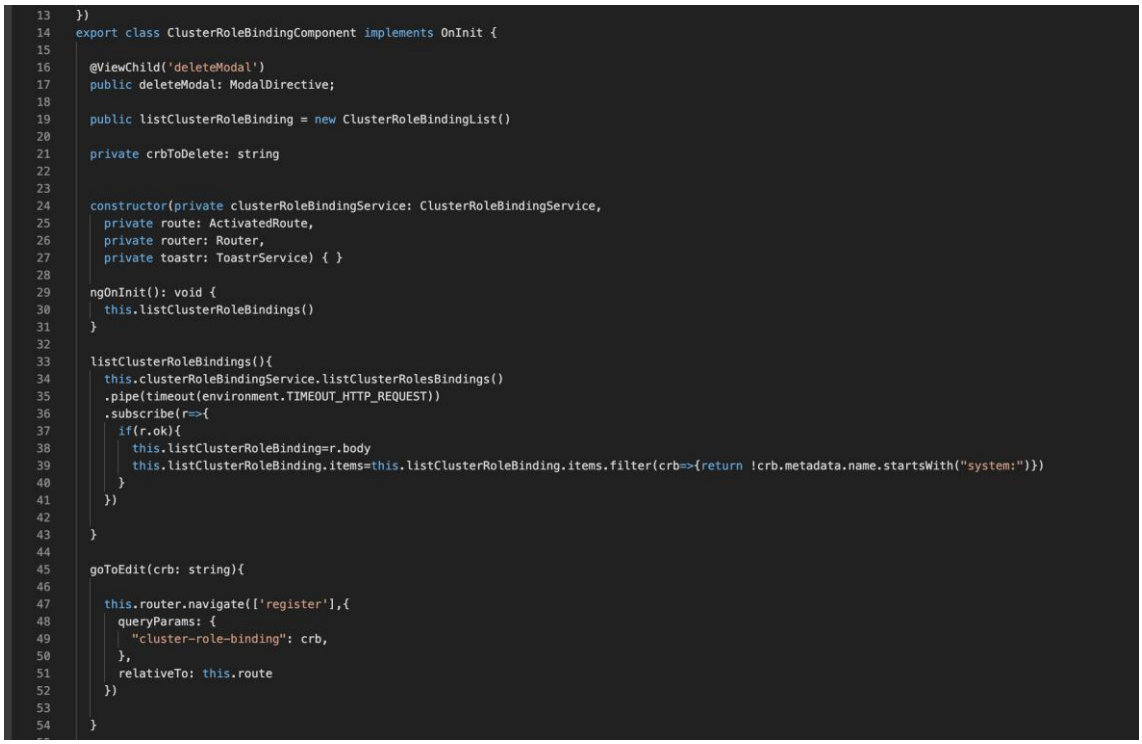
Implementación de controlador para creación de Cluster Roles.

```
submitForm(){

  if (this.clusterRole){
    this.clusterRoleBinding.roleRef.apiGroup="rbac.authorization.k8s.io"
    this.clusterRoleBinding.roleRef.kind="ClusterRole"
    this.clusterRoleBinding.roleRef.name=this.clusterRole.metadata.name
    this.clusterRoleBinding.subjects = []

    this.users.forEach(u=>{
      this.clusterRoleBinding.subjects.push(new Subject(u["value"],"rbac.authorization.k8s.io","User"))
    })

    this.groups.forEach(g=>{
      this.clusterRoleBinding.subjects.push(new Subject(g["value"],"rbac.authorization.k8s.io","Group"))
    })

    if (this.isEdit){
      this.clusterRoleBindingService.updateClusterRoleBinding(this.clusterRoleBinding.metadata.name,this.clusterRoleBinding)
        .subscribe(r=>{
          if(r.ok ){
            this.toastr.success("Cluster Role permission updated.")
            this.fetchClusterRoleBinding(this.clusterRoleBinding.metadata.name)
            return
          }
          this.toastr.error("An error ocurred while trying to update permission assignment.")
        }, e=>{
          this.toastr.error("Unexpected error ocurred.")
        })
    }else{
      this.clusterRoleBindingService.createClusterRoleBinding(this.clusterRoleBinding)
        .subscribe(r=>{
          if(r.ok ){
            this.toastr.success("Cluster Role permission created.")
            return
          }
          this.toastr.error("An error ocurred while trying to create permission assignment.")
        }, e=>{
          this.toastr.error("Unexpected error ocurred.")
        })
    }
  }
```

Fuente. Elaboración Propia.

Figura 56

Implementación de controlador para asignación de permisos Cluster Roles.

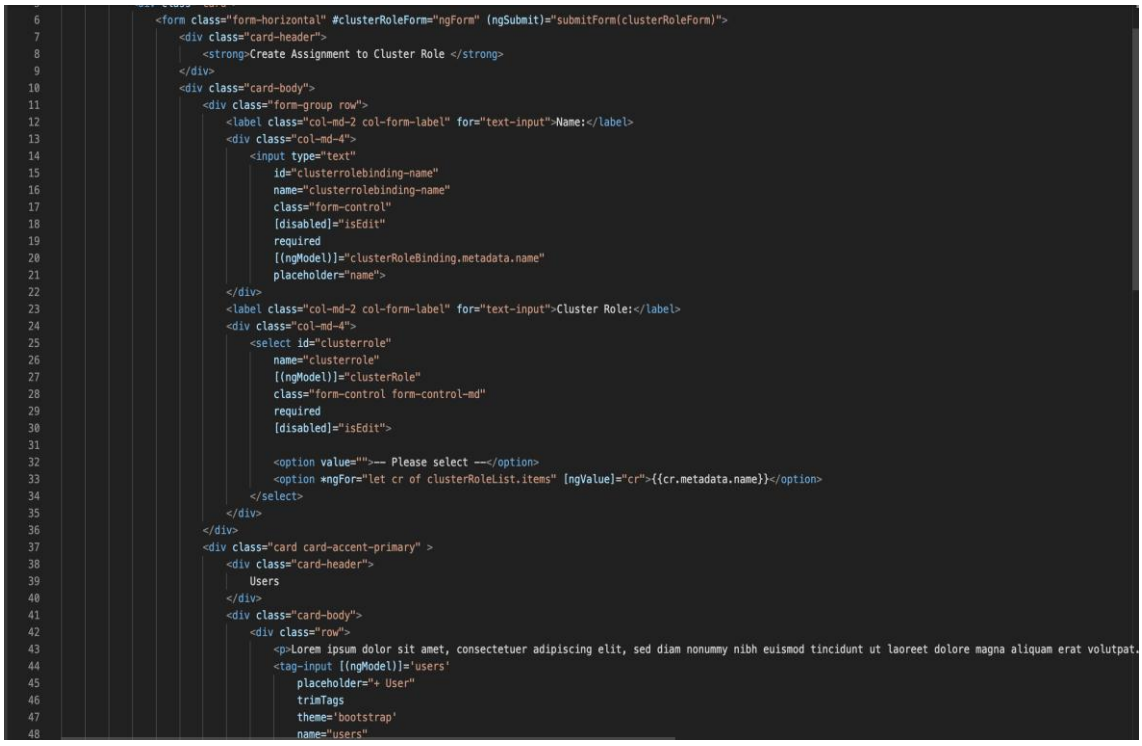


```
})
export class ClusterRoleBindingComponent implements OnInit {

  @ViewChild('deleteModal')
  public deleteModal: ModalDirective;

  public listClusterRoleBinding = new ClusterRoleBindingList()

  private crbToDelete: string


  constructor(private clusterRoleBindingService: ClusterRoleBindingService,
    private route: ActivatedRoute,
    private router: Router,
    private toastr: ToastrService) { }

  ngOnInit(): void {
    this.listClusterRoleBindings()
  }

  listClusterRoleBindings(){
    this.clusterRoleBindingService.listClusterRolesBindings()
    .pipe(timeout(environment.TIMEOUT_HTTP_REQUEST))
    .subscribe(r=>{
      if(r.ok){
        this.listClusterRoleBinding=r.body
        this.listClusterRoleBinding.items=this.listClusterRoleBinding.items.filter(crb=>{return !crb.metadata.name.startsWith("system:")})
      }
    })

  }

  goToEdit(crb: string){

    this.router.navigate(['register'],{
      queryParams: {
        "cluster-role-binding": crb,
      },
      relativeTo: this.route
    })

  }
```

Fuente. Elaboración Propia.

Figura 57

Implementación vista asignación Cluster Roles.



```
<form class="form-horizontal" #clusterRoleForm="ngForm" (ngSubmit)="submitForm(clusterRoleForm)">
    <div class="card-header">
        <strong>Create Assignment to Cluster Role </strong>
    </div>
    <div class="card-body">
        <div class="form-group row">
            <label class="col-md-2 col-form-label" for="text-input">Name:</label>
            <div class="col-md-4">
                <input type="text"
                    id="clusterrolebinding-name"
                    name="clusterrolebinding-name"
                    class="form-control"
                    [disabled]="isEdit"
                    required
                    [(ngModel)]="clusterRoleBinding.metadata.name"
                    placeholder="name">
            </div>
            <label class="col-md-2 col-form-label" for="text-input">Cluster Role:</label>
            <div class="col-md-4">
                <select id="clusterrole"
                    name="clusterrole"
                    [(ngModel)]="clusterRole"
                    class="form-control form-control-md"
                    required
                    [disabled]="isEdit">

                    <option value="">-- Please select --</option>
                    <option *ngFor="let cr of clusterRoleList.items" [ngValue]="cr">{{cr.metadata.name}}</option>
                </select>
            </div>
        </div>
        <div class="card card-accent-primary" >
            <div class="card-header">
                Users
            </div>
            <div class="card-body">
                <div class="row">
                    <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.
                    <tag-input [(ngModel)]='users'
                        placeholder="+ User"
                        trimTags
                        theme='bootstrap'
                        name="users"
```

Fuente. Elaboración Propia.

Figura 58

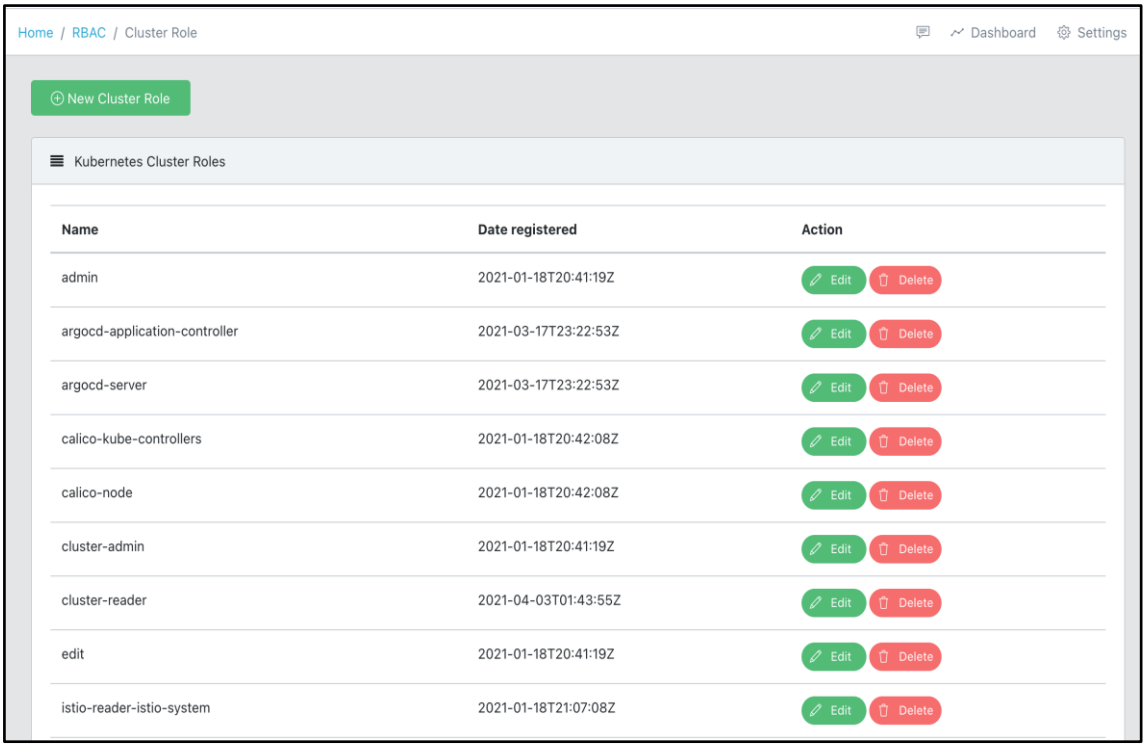
Implementación vista listado Cluster Roles.

```
<div class="row">
  <div class="col-lg-12">
    <div class="card">
      <div class="card-header">
        <i class="fa fa-align-justify"></i> Cluster Role Permissions
      </div>
      <div class="card-body">

        <table class="table">
          <thead>
            <tr>
              <th>Name</th>
              <th>Date registered</th>
              <th>Action</th>
            </tr>
          </thead>
          <tbody>
            <tr *ngFor="let crb of listClusterRoleBinding.items">
              <td> {{ crb.metadata.name }}</td>
              <td> {{ crb.metadata.creationTimestamp }}</td>
              <td>
                <button type="button" class="btn btn-sm btn-pill btn-success" (click)="goToEdit(crb.metadata.name)">
                  <i class="icon-pencil"></i> Edit  
                </button>
                <button type="button" class="btn btn-sm btn-pill btn-danger" data-toggle="modal" (click)="openConfirmationModal(crb.metadata.name)">
                  <i class="icon-trash"></i>  Delete  
                </button>
              </td>
            </tr>

          </tbody>
        </table>
        <ul class="pagination">
          <li class="page-item"><a class="page-link" href="#">Prev</a></li>
          <li class="page-item active">
            <a class="page-link" href="#">1</a>
          </li>
          <li class="page-item"><a class="page-link" href="#">2</a></li>
          <li class="page-item"><a class="page-link" href="#">3</a></li>
          <li class="page-item"><a class="page-link" href="#">4</a></li>
          <li class="page-item"><a class="page-link" href="#">Next</a></li>
        </ul>
      </div>
```

Fuente. Elaboración Propia.

Figura 59

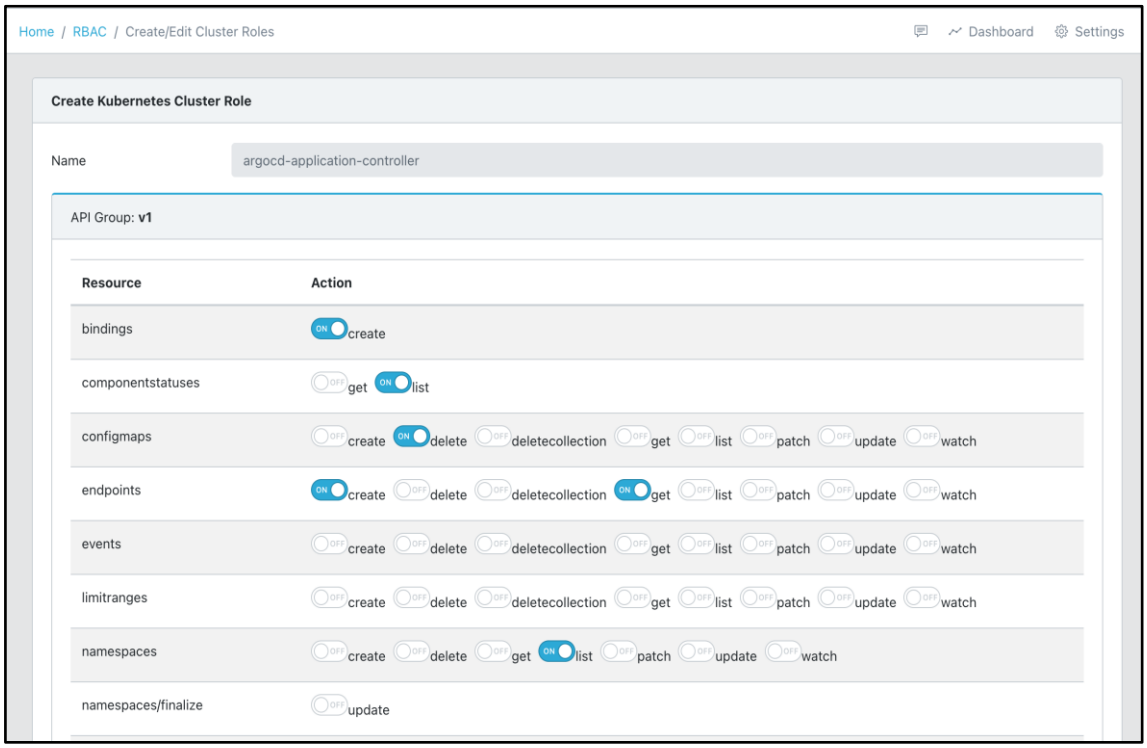
Interfaz listado de Cluster Roles



Fuente. Elaboración Propia.

Figura 60

Interfaz creación y edición de asignación permisos Cluster Roles.



Fuente. Elaboración Propia.

4.1.1.2.5. SPRINT 05

En este primer sprint se revisaron los elementos del Backlog y se seleccionaron las historias que se realizaran de acuerdo con las necesidades y prioridades del Product Owner, de esta revisión se obtuvo el entregable del sprint 05 (Tabla 14).

Tabla 14

Entregables del Sprint 05.

HISTORIAS DE USUARIO	ESFUERZO
Como Product Owner, quiero poder eliminar roles a nivel de namespace a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5
Como Product Owner, quiero poder crear roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5
Como Product Owner, quiero poder editar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	5

Fuente. Elaboración Propia.

La reunión del cuarto sprint tuvo una duración aproximada de 3 horas, donde se clarifican los objetivos y el contexto de cada elemento. A continuación, se realizan la planificación detallada de las tareas para saber cómo implementar los elementos seleccionados en el sprint.

Tabla 15

Estimación tiempo disponible Sprint 05.

Duración del Sprint	2 Semanas
Días efectivos del Sprint	10 días

Miembros del equipo	**Días disponibles**	**Horas disponibles por día**	**Total Horas Sprint**
Christian Altamirano Ayala	10	8	80

Fuente. Elaboración Propia.

El siguiente paso una vez identificado y planificado la estimación de los tiempos, así como los recursos disponibles asignados al proyecto es la elaboración de las tareas individuales por Historia.

Figura 61

Sprint Backlog del Sprint 05.

Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder eliminar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del cluster.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
Como Product Owner, quiero poder ver los recursos existentes en cluster, así como las versiones existentes y los permisos que se les puede aplicar, a fin de poder crear roles bien definidos.	Elaboracion de interaz	CAA	2										
	Implementacion de API	CAA	2										
	Prueba de Funcionalidad	CAA	1										
		Real	15	15	15	15	15	15	15	15	15	15	15
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración Propia.

Una vez definido el Sprint Backlog del Sprint 05, el equipo realizó los Dayli Scrum con el fin de revisar el avance de las historias y resolver posibles impedimentos, en las reuniones diarias de no más de 15 minutos el equipo básicamente realiza las siguientes 3 preguntas:

a. ¿Qué se hizo desde la última reunión?
b. ¿Qué tiene planificado hacer el día de hoy?
c. ¿Qué impedimentos se tiene para poder culminar las tareas?

Esta activad se lleva a cabo de manera repetitiva a lo largo del Sprint, que en este caso fue de 10 días. A medida que el equipo fue reportando los avances diarios de las tareas, el Scrum Master junto con el Producto Owner mapean dicho avance en el Sprint Backlog de la figura 62.

Figura 62

Avance Sprint backlog del Sprint 05.

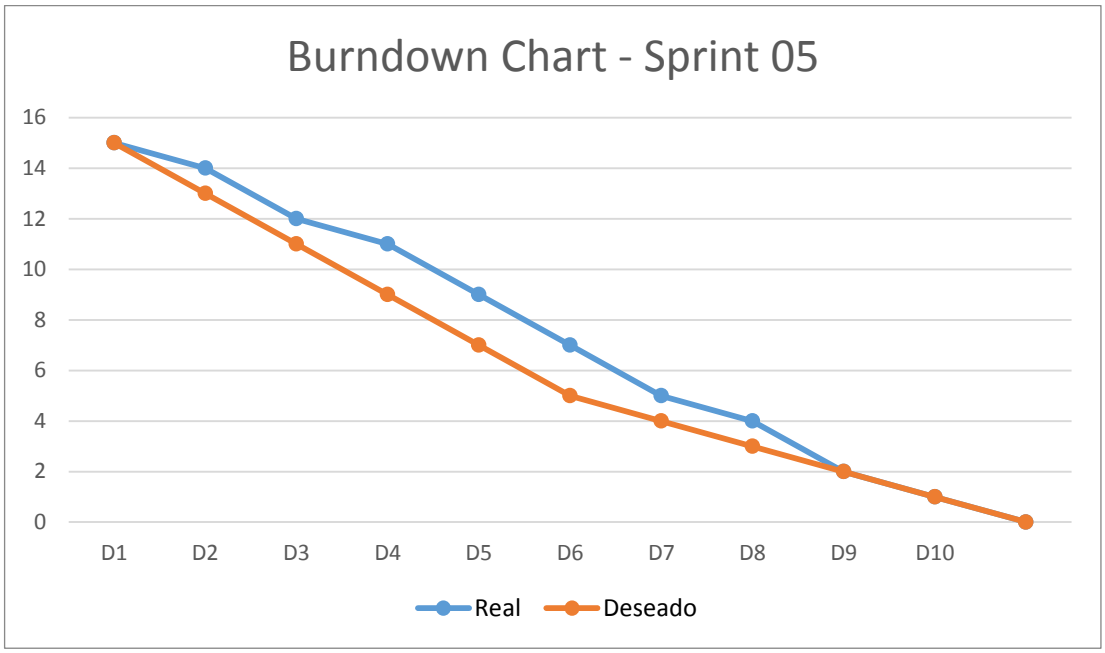
Sprint Backlog	Tareas	Responsable	Puntos	PUNTOS ENTREGADOS AL FINAL DEL DIA									
				D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Como Product Owner, quiero poder eliminar roles a nivel del cluster a fin de tener un control y gobiernos sobre los objetos Kubernetes existentes.	Elaboracion de interaz	CAA	2	1	1								
	Implementacion de API	CAA	2		1	1							
	Prueba de Funcionalidad	CAA	1				1						
Como Product Owner, quiero poder asignar los roles a los grupos, usuarios y service accounts a fin de garantizar los permisos mínimos necesarios para los usuarios del cluster.	Elaboracion de interaz	CAA	2				1	1					
	Implementacion de API	CAA	2					1	1				
	Prueba de Funcionalidad	CAA	1							1			
Como Product Owner, quiero poder ver los recursos existentes en cluster, así como las versiones existentes y los permisos que se les puede aplicar, a fin de poder crear roles bien definidos.	Elaboracion de interaz	CAA	2							1	1		
	Implementacion de API	CAA	2								1	1	
	Prueba de Funcionalidad	CAA	1										1
		Real	15	14	12	11	9	7	6	4	2	1	0
		Deseado	15	12	11	10	9	8	7	6	4	2	0

Fuente. Elaboración Propia.

Así mismo el avance se puede observar en la gráfica Burndown Chart de la figura 63.

Figura 63

Gráfica BurnDown Chart del Sprint 05.



Fuente. Elaboración Propia.

Figura 64

Implementación cliente REST para Cluster Roles.

```
47
48    deleteClusterRole(name: string): Observable<HttpResponse<any>> {
49      return this.http.delete<any>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterroles/'+name,
50        {headers,observe: 'response'}
51        );
52    }
53  }
54
```

Fuente. Elaboración Propia.

Figura 65

Implementación de controlador para eliminar permisos Cluster Roles.

```
66
67  deleteRole(){
68    if (this.crbToDelete){
69      this.clusterRoleBindingService.deleteClusterRoleBinding(this.crbToDelete).subscribe((r)=>{
70        if (r.ok){
71          this.listClusterRoleBinding.items = this.listClusterRoleBinding.items.filter(i=>{return i.metadata.name!==this.crbToDelete})
72          this.deleteModal.hide()
73          this.toastr.success('Assignment deleted succesfully.')
74        }else{
75          this.toastr.error('An error ocurred trying to delete the role.')
76        }
77      }, (e)=>{
78        this.toastr.error('An error ocurred trying to delete the role.')
79    });
80    }else{
81      this.toastr.warning('no role to be deleted was specified.')
82    }
83  }
84
85  }
86
```

Fuente. Elaboración Propia.

Figura 66

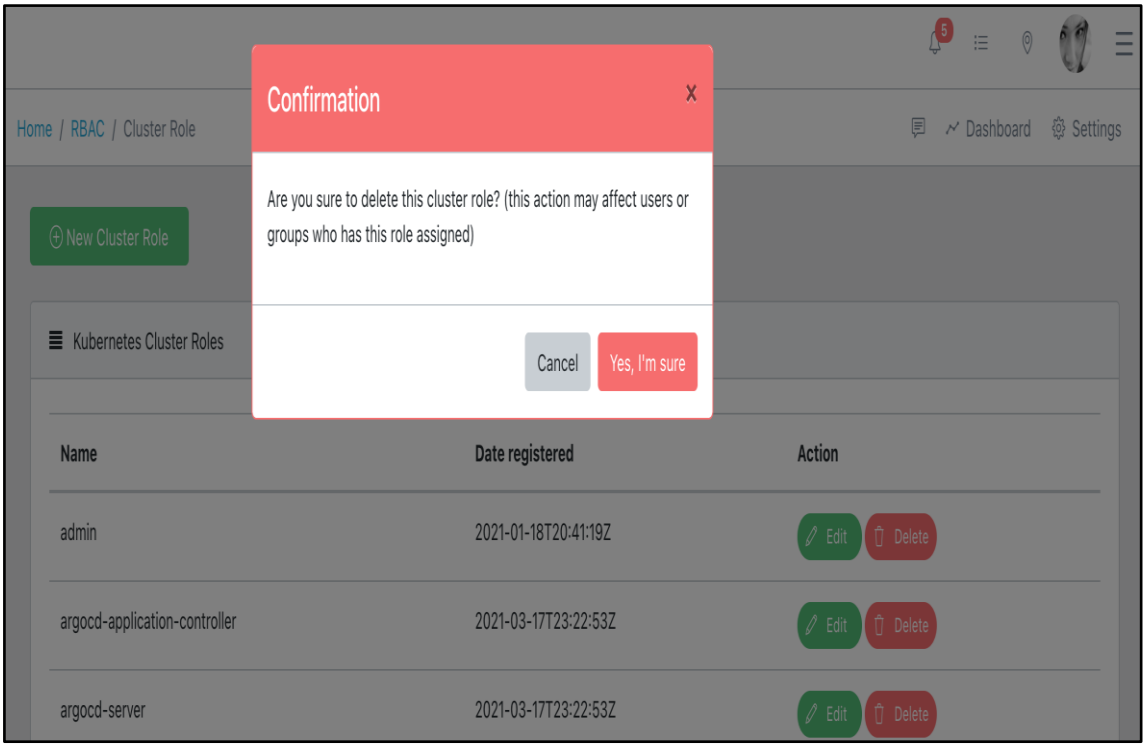
Implementación vista eliminación de Cluster Roles.

```
<div bsModal #deleteModal="bs-modal" class="modal fade" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-danger" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h4 class="modal-title">Confirmation</h4>
        <button type="button" class="close" (click)="deleteModal.hide()" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <p>Are you sure to delete this assignment? (this action may affect users or groups who has this role assigned)</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" (click)="deleteModal.hide()">Cancel</button>
        <button type="button" class="btn btn-danger" (click)="deleteRole()">Yes, I'm sure</button>
      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->

</div>
```

Fuente. Elaboración Propia.

Figura 67

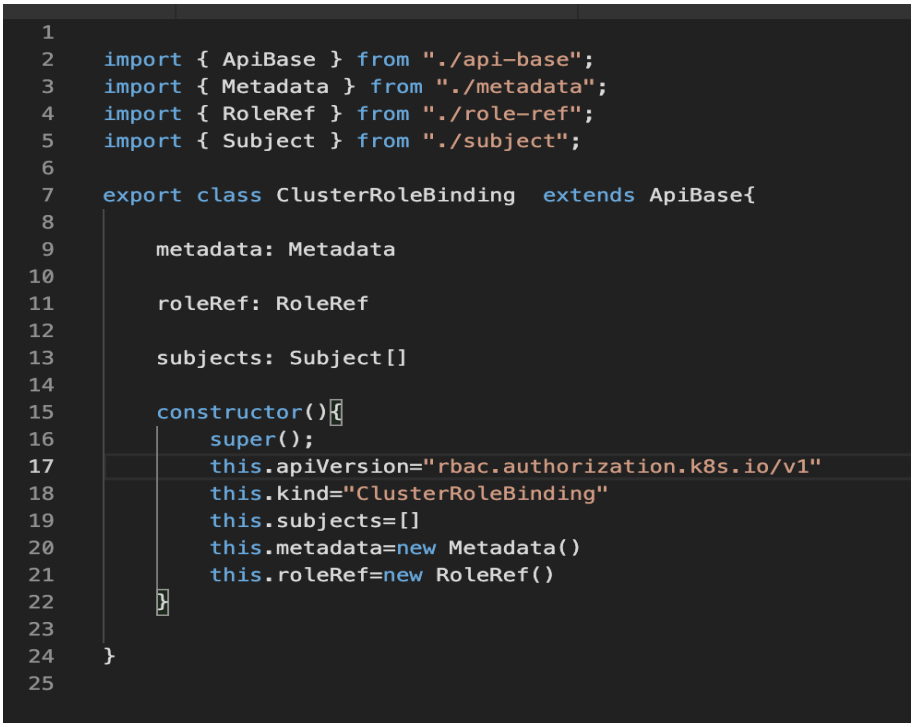
Interfaz eliminación de Cluster Roles.



Fuente. Elaboración Propia.

Figura 68

Implementación modelo para Cluster Role.

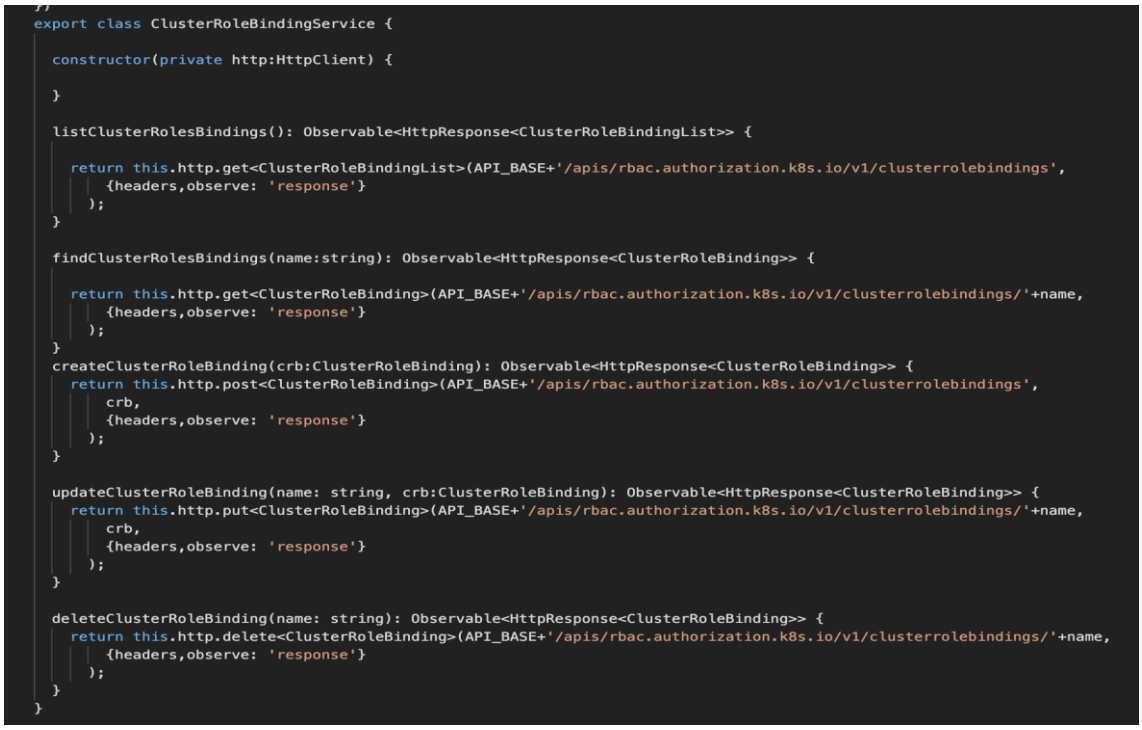


```
import { ApiBase } from "./api-base";
import { Metadata } from "./metadata";
import { RoleRef } from "./role-ref";
import { Subject } from "./subject";

export class ClusterRoleBinding  extends ApiBase{

    metadata: Metadata

    roleRef: RoleRef

    subjects: Subject[]

    constructor(){
        super();
        this.apiVersion="rbac.authorization.k8s.io/v1"
        this.kind="ClusterRoleBinding"
        this.subjects=[]
        this.metadata=new Metadata()
        this.roleRef=new RoleRef()
    }

}
```

Fuente. Elaboración Propia.

Figura 69

Implementación cliente REST para asignación permisos Cluster Role.

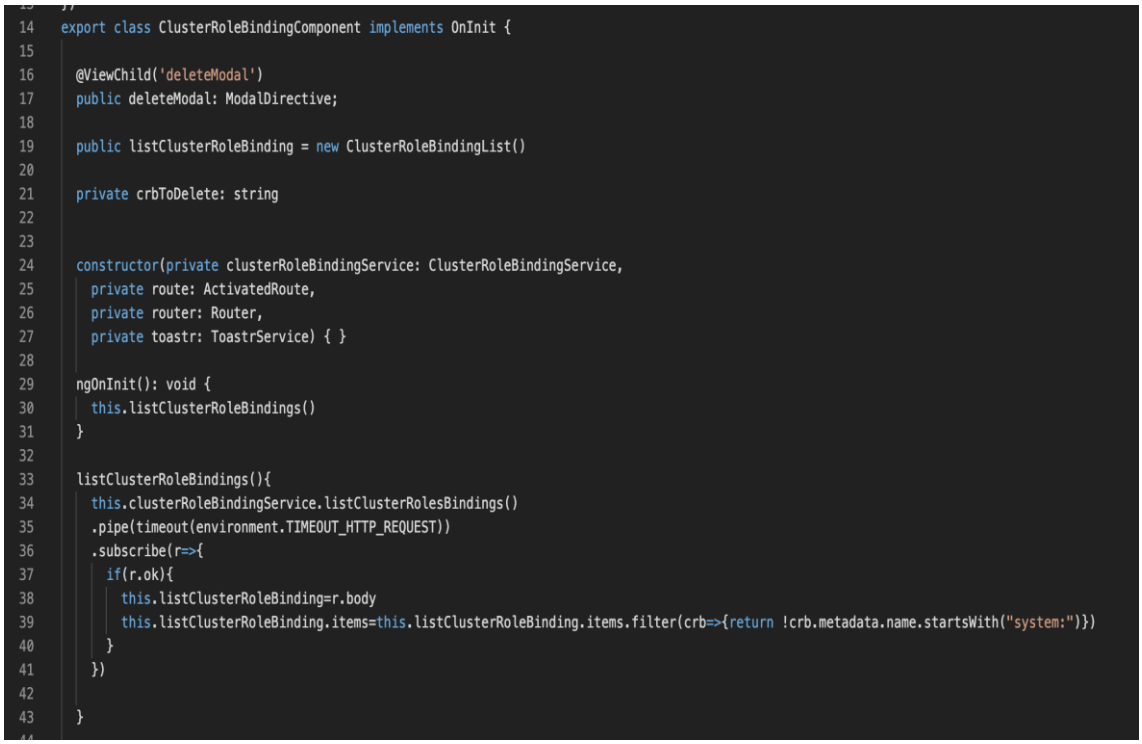


```
export class ClusterRoleBindingService {

  constructor(private http:HttpClient) {

  }

  listClusterRolesBindings(): Observable<HttpResponse<ClusterRoleBindingList>> {

    return this.http.get<ClusterRoleBindingList>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterrolebindings',
      {headers,observe: 'response'}
    );
  }

  findClusterRolesBindings(name:string): Observable<HttpResponse<ClusterRoleBinding>> {

    return this.http.get<ClusterRoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterrolebindings/'+name,
      {headers,observe: 'response'}
    );
  }
  createClusterRoleBinding(crb:ClusterRoleBinding): Observable<HttpResponse<ClusterRoleBinding>> {
    return this.http.post<ClusterRoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterrolebindings',
      crb,
      {headers,observe: 'response'}
    );
  }

  updateClusterRoleBinding(name: string, crb:ClusterRoleBinding): Observable<HttpResponse<ClusterRoleBinding>> {
    return this.http.put<ClusterRoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterrolebindings/'+name,
      crb,
      {headers,observe: 'response'}
    );
  }

  deleteClusterRoleBinding(name: string): Observable<HttpResponse<ClusterRoleBinding>> {
    return this.http.delete<ClusterRoleBinding>(API_BASE+'/apis/rbac.authorization.k8s.io/v1/clusterrolebindings/'+name,
      {headers,observe: 'response'}
    );
  }
}
```

Fuente. Elaboración Propia.

Figura 70

Implementación controlador asignación permisos Cluster Role.

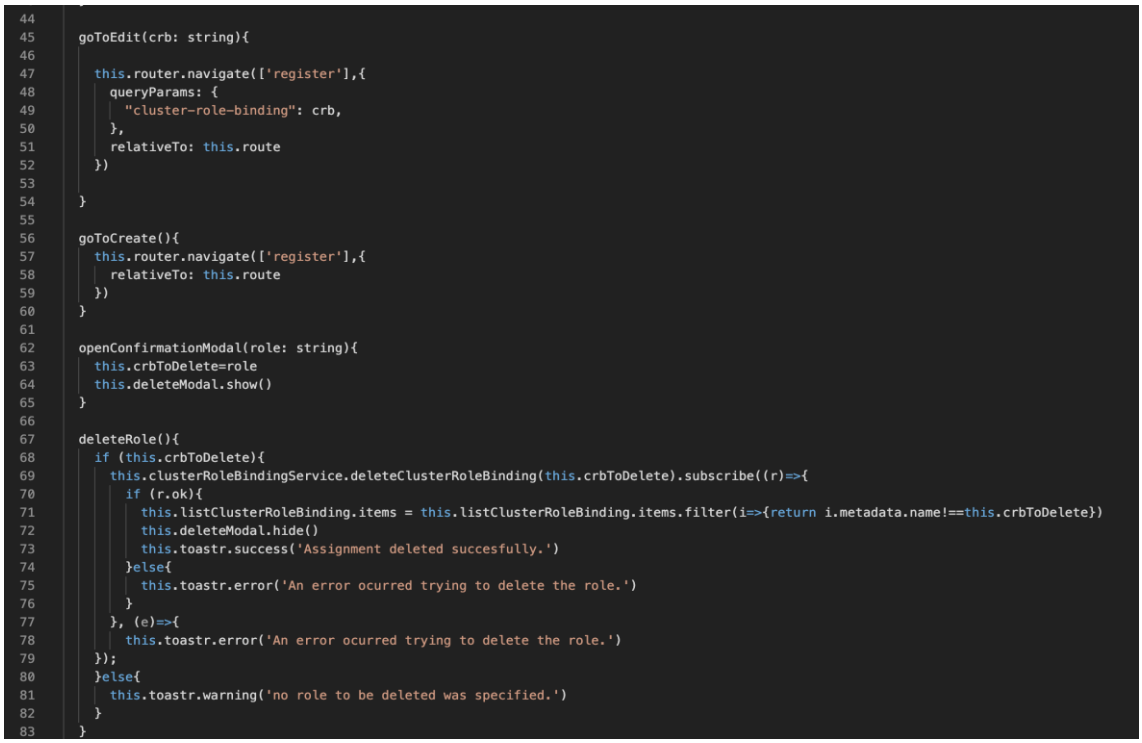


```
export class ClusterRoleBindingComponent implements OnInit {

  @ViewChild('deleteModal')
  public deleteModal: ModalDirective;

  public listClusterRoleBinding = new ClusterRoleBindingList()

  private crbToDelete: string


  constructor(private clusterRoleBindingService: ClusterRoleBindingService,
    private route: ActivatedRoute,
    private router: Router,
    private toastr: ToastrService) { }

  ngOnInit(): void {
    this.listClusterRoleBindings()
  }

  listClusterRoleBindings(){
    this.clusterRoleBindingService.listClusterRolesBindings()
    .pipe(timeout(environment.TIMEOUT_HTTP_REQUEST))
    .subscribe(r=>{
      if(r.ok){
        this.listClusterRoleBinding=r.body
        this.listClusterRoleBinding.items=this.listClusterRoleBinding.items.filter(crb=>{return !crb.metadata.name.startsWith("system:")})
      }
    })

  }
```

Fuente. Elaboración Propia.

Figura 71

Implementación controlador edición y eliminación de asignación de permisos Cluster Roles



```
  goToEdit(crb: string){

    this.router.navigate(['register'],{
      queryParams: {
        "cluster-role-binding": crb,
      },
      relativeTo: this.route
    })

  }

  goToCreate(){
    this.router.navigate(['register'],{
      relativeTo: this.route
    })
  }

  openConfirmationModal(role: string){
    this.crbToDelete=role
    this.deleteModal.show()
  }

  deleteRole(){
    if (this.crbToDelete){
      this.clusterRoleBindingService.deleteClusterRoleBinding(this.crbToDelete).subscribe((r)=>{
        if (r.ok){
          this.listClusterRoleBinding.items = this.listClusterRoleBinding.items.filter(i=>{return i.metadata.name!==this.crbToDelete})
          this.deleteModal.hide()
          this.toastr.success('Assignment deleted succesfully.')
        }else{
          this.toastr.error('An error ocurred trying to delete the role.')
        }
      }, (e)=>{
        this.toastr.error('An error ocurred trying to delete the role.')
    });
    }else{
      this.toastr.warning('no role to be deleted was specified.')
    }
  }
```

Fuente. Elaboración Propia.

Figura 72

Implementación de controlador de asignación de permisos a nivel Cluster.

```
export class ClusterRoleBindingRegisterComponent implements OnInit {


  clusterRoleBinding : ClusterRoleBinding= new ClusterRoleBinding()
  clusterRoleList: ClusterRoleList=new ClusterRoleList()
  clusterRole: ClusterRole=new ClusterRole()
  clusterRoleBindingName : string
  users : Object[] = [];
  groups : Object[] = [];
  isEdit: boolean = false
  constructor(private clusterRoleService: ClusterRoleService,
    private clusterRoleBindingService: ClusterRoleBindingService,
    private toastr: ToastrService,
    private route: ActivatedRoute) { }

  ngOnInit(){
    var crb = this.route.snapshot.queryParamMap.get('cluster-role-binding')
    if (crb){
      this.isEdit=true
    }
    this.listClusterRole()

  }

  private fetchClusterRoleBinding(name:string){
    this.clusterRoleBindingService.findClusterRolesBindings(name)
      .subscribe(r=>{
        if (r.ok){
          this.clusterRoleBinding=r.body
          this.users=[]
          this.groups=[]
          this.clusterRoleBinding.subjects.forEach(sub=>{
            if (sub.kind=="Group"){
              this.groups.push({"value":sub.name,"display":sub.name})
            }
            if (sub.kind=="User"){
              this.users.push({"value":sub.name,"display":sub.name})
            }
          })
```

Fuente. Elaboración Propia.

Figura 73

Implementación vista listado de Cluster Roles.

```
submitForm(){

  if (this.clusterRole){
    this.clusterRoleBinding.roleRef.apiGroup="rbac.authorization.k8s.io"
    this.clusterRoleBinding.roleRef.kind="ClusterRole"
    this.clusterRoleBinding.roleRef.name=this.clusterRole.metadata.name
    this.clusterRoleBinding.subjects = []

    this.users.forEach(u=>{
      this.clusterRoleBinding.subjects.push(new Subject(u["value"],"rbac.authorization.k8s.io","User"))
    })

    this.groups.forEach(g=>{
      this.clusterRoleBinding.subjects.push(new Subject(g["value"],"rbac.authorization.k8s.io","Group"))
    })

    if (this.isEdit){
      this.clusterRoleBindingService.updateClusterRoleBinding(this.clusterRoleBinding.metadata.name,this.clusterRoleBinding)
        .subscribe(r=>{
          if(r.ok ){
            this.toastr.success("Cluster Role permission updated.")
            this.fetchClusterRoleBinding(this.clusterRoleBinding.metadata.name)
            return
          }
          this.toastr.error("An error ocurred while trying to update permission assignment.")
        }, e=>{
          this.toastr.error("Unexpected error ocurred.")
        })
    }else{
      this.clusterRoleBindingService.createClusterRoleBinding(this.clusterRoleBinding)
        .subscribe(r=>{
          if(r.ok ){
            this.toastr.success("Cluster Role permission created.")
            return
          }
          this.toastr.error("An error ocurred while trying to create permission assignment.")
        }, e=>{
          this.toastr.error("Unexpected error ocurred.")
        })
    }
  }
```

Fuente. Elaboración Propia.

Figura 74

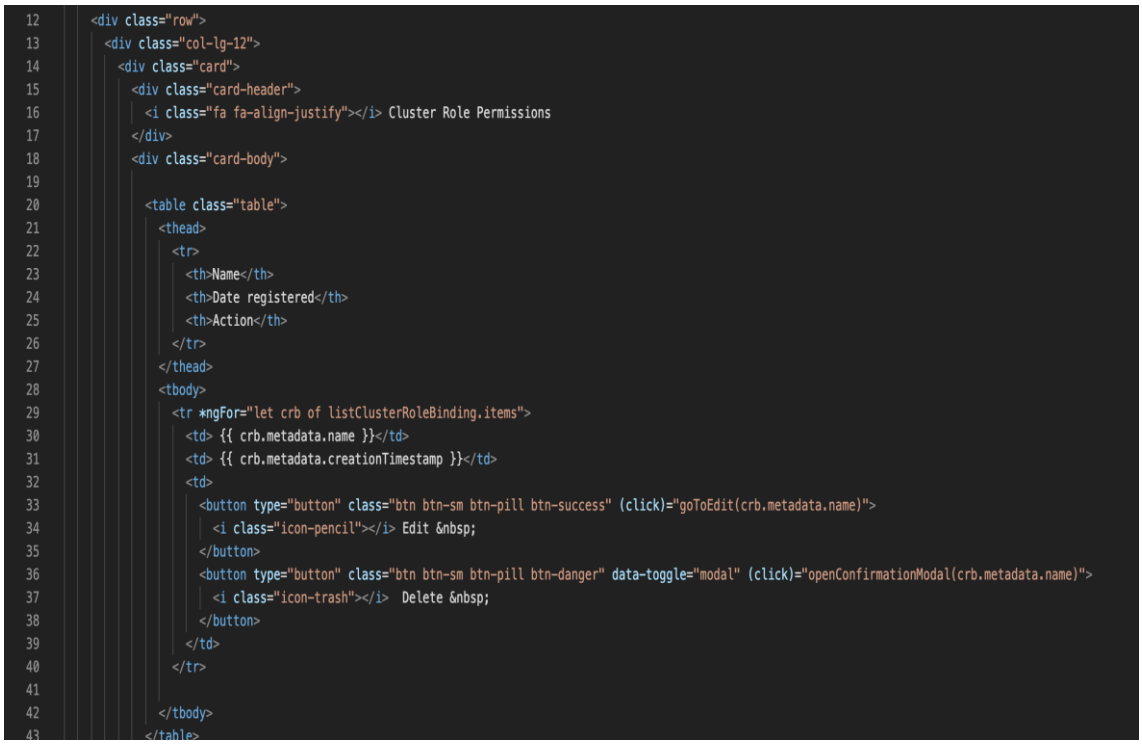
Implementación vista creación permisos Cluster Roles.

```
<div class="animated fadeIn">

    <div class="row">
        <div class="col-md-12">
            <div class="card">
                <form class="form-horizontal" #clusterRoleForm="ngForm" (ngSubmit)="submitForm(clusterRoleForm)">
                    <div class="card-header">
                        <strong>Create Assignment to Cluster Role </strong>
                    </div>
                    <div class="card-body">
                        <div class="form-group row">
                            <label class="col-md-2 col-form-label" for="text-input">Name:</label>
                            <div class="col-md-4">
                                <input type="text"
                                    id="clusterrolebinding-name"
                                    name="clusterrolebinding-name"
                                    class="form-control"
                                    [disabled]="isEdit"
                                    required
                                    [(ngModel)]="clusterRoleBinding.metadata.name"
                                    placeholder="name">
                            </div>
                            <label class="col-md-2 col-form-label" for="text-input">Cluster Role:</label>
                            <div class="col-md-4">
                                <select id="clusterrole"
                                    name="clusterrole"
                                    [(ngModel)]="clusterRole"
                                    class="form-control form-control-md"
                                    required
                                    [disabled]="isEdit">

                                    <option value="">-- Please select --</option>
                                    <option *ngFor="let cr of clusterRoleList.items" [ngValue]="cr">{{cr.metadata.name}}</option>
                                </select>
                            </div>
                        </div>
                        <div class="card card-accent-primary" >
                            <div class="card-header">
                                Users
```

Fuente. Elaboración Propia.

Figura 75

Implementación vista listado de asignación de permisos Cluster Roles.



```
<div class="row">
  <div class="col-lg-12">
    <div class="card">
      <div class="card-header">
        <i class="fa fa-align-justify"></i> Cluster Role Permissions
      </div>
      <div class="card-body">

        <table class="table">
          <thead>
            <tr>
              <th>Name</th>
              <th>Date registered</th>
              <th>Action</th>
            </tr>
          </thead>
          <tbody>
            <tr *ngFor="let crb of listClusterRoleBinding.items">
              <td> {{ crb.metadata.name }}</td>
              <td> {{ crb.metadata.creationTimestamp }}</td>
              <td>
                <button type="button" class="btn btn-sm btn-pill btn-success" (click)="goToEdit(crb.metadata.name)">
                  <i class="icon-pencil"></i> Edit  
                </button>
                <button type="button" class="btn btn-sm btn-pill btn-danger" data-toggle="modal" (click)="openConfirmationModal(crb.metadata.name)">
                  <i class="icon-trash"></i>  Delete  
                </button>
              </td>
            </tr>

          </tbody>
        </table>
```

Fuente. Elaboración Propia.

Figura 76

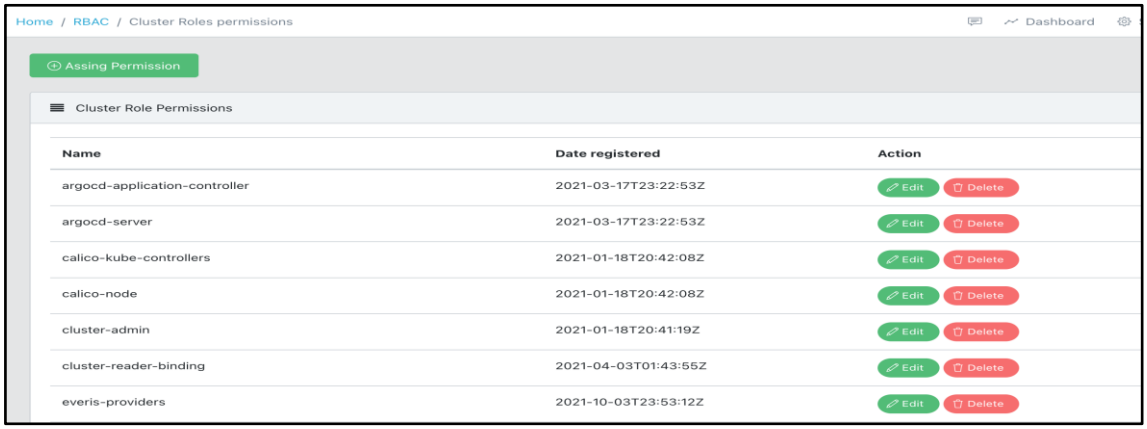
Implementación vista eliminación de asignación de permisos Cluster Roles.

```
<div bsModal #deleteModal="bs-modal" class="modal fade" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-danger" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h4 class="modal-title">Confirmation</h4>
        <button type="button" class="close" (click)="deleteModal.hide()" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <p>Are you sure to delete this assignment? (this action may affect users or groups who has this role assigned)</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" (click)="deleteModal.hide()">Cancel</button>
        <button type="button" class="btn btn-danger" (click)="deleteRole()">Yes, I'm sure</button>
      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->

</div>
```

Fuente. Elaboración Propia.

Figura 77

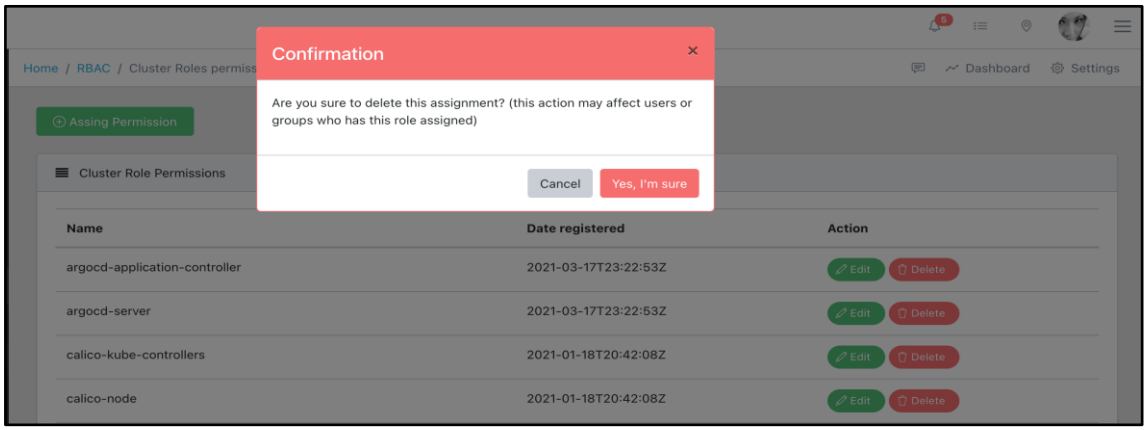
Interfaz listado de asignación permisos Cluster Roles.



Fuente. Elaboración Propia.

Figura 78

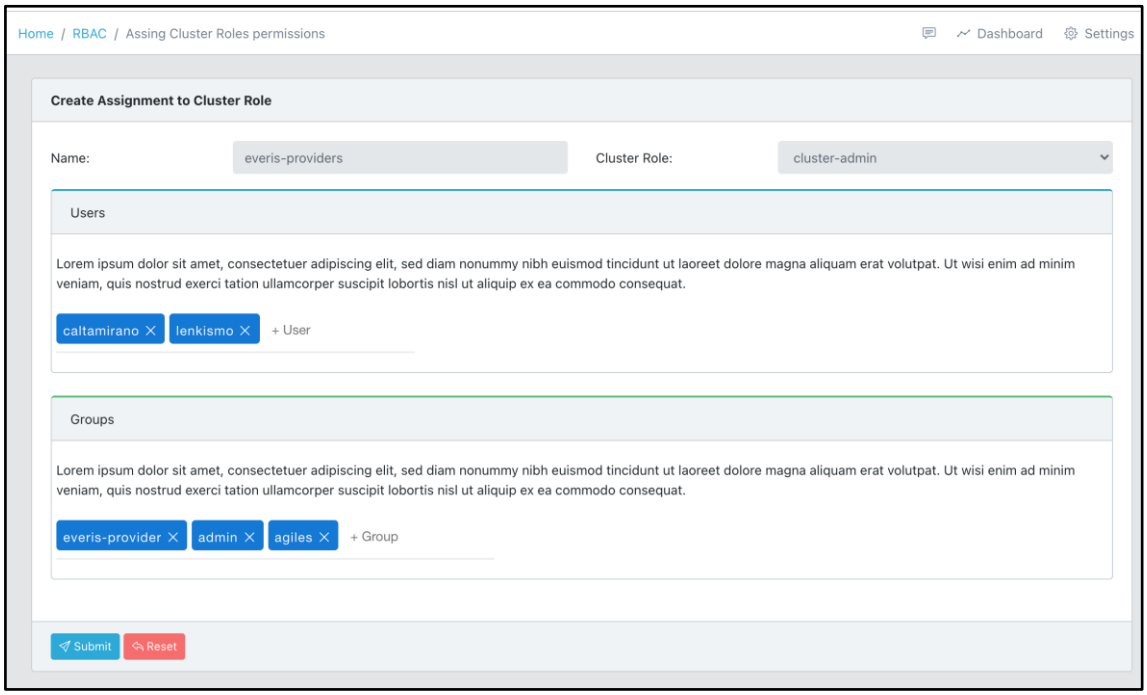
Interfaz eliminación de asignación de permisos Cluster Roles.



Fuente. Elaboración Propia.

Figura 79

Interfaz listado de asignación permisos Cluster Roles.



Fuente. Elaboración Propia.

4.2. DISCUSIÓN

Según la Agencia de Seguridad Nacional (2021), la autenticación y la autorización son los mecanismos primarios para restringir el acceso a recursos del clúster Kubernetes, De acuerdo con la investigación se implementó mecanismos que permitan configurar la autenticación y la autorización hacia los clúster Kubernetes.

Según la Agencia de Seguridad Nacional (2021), el Control de Acceso Basado en Roles es un método para controlar el acceso a los recursos del clúster basado en Roles de los individuos de una organización. De acuerdo con los resultados de la investigación, se ha creado los distintos módulos que permitan crear los permisos a nivel de namespace y clúster a distintas identidades que pudiera manejar una organización.

CONCLUSIONES

1. Se ha logrado desarrollar mecanismos de autenticación a fin de tener un control adecuado de quien o quienes puede acceder a los recursos que exponen en clúster Kubernetes.

2. Se ha logrado desarrollar mecanismos de autorización a fin de tener un control adecuado sobre si pueden ejecutar acciones sobre los recursos que exponen en clúster Kubernetes una vez estén autenticados.

RECOMENDACIONES

- Realizar un estudio a fin de integrar la autenticación con proveedores OIDC.

- Investigar un Webhook a fin de lanzar mensaje o alertas en casos de comportamiento anómalos a las APIs de Kubernetes.

- Realizar un estudio para modificar la aplicación a fin de tener soporte para múltiples clústeres.

REFERENCIAS BIBLIOGRÁFICAS

Arias, F. (2006). El proyecto de Investigación. Introducción a la metodología científica. Caracas, Editorial Episteme

Arias-Gómez, J., Villasís-Keever, M. Á., & Miranda Novales, M. G. (2016). El protocolo de investigación III. Alergia México, 63(2), 201-206. Obtenido de https://www.redalyc.org/articulo.oa?id=486755023011

Auth0 (2021), What is Authorization? obtenido de https://auth0.com/intro-to-iam/what-is-authorization/

Bernal C. (2006). Metodología de la Investigación. México. Editorial Pearson Prentice Hall.

Booch, G. (2001). Análisis y Diseño Orientado a Objetos. México: S.A. ALHAMBRA.

Cloud Native Computing Foundation (2019), CNCF Survey 2019, Estados Unidos.

Deitel, Paul; Deitel, Harvey;. (2012). How to Program Java (9na ed.). México: Pearson Educación.

Docker (2021). What is a Container. Obtenido de https://www.docker.com/resources/what-container

Duran, F., Gutiérrez, F., Pimentel, E. (2007). Programación orientada a objetos con java. Madrid, España: Paraninfo s. a.

Groussard, T. (2012). Java 7: los fundamentos del lenguaje java. Barcelona: ediciones ENI.

Jorge Cervantes, María Gómez Pedro Gonzales y Abel García (2016). Introducción a la programación orientada a objetos. Mexico: Unida Cuajimalpa.

Jøsang, Audun (2017), A Consistent Definition of Authorization, Proceedings of the 13th International Workshop on Security and Trust Management (STM 2017)

José Supo (2020). Metodología de la investigación científica. Perú.

Ken Schwaber y Jeff Sutherland (2017), La guía definitiva de scrum: Las reglas del juego. Estados Unidos

Kroenke, d. (2003). Procesamiento de base de datos: fundamentos, diseño e implementación. (8va ed.). Juárez, México: Pearson.

Kubernetes (2021), What is Kubernetes. Obtenido de https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

Lujan, S. (2001). Programación en internet: cliente web. Alicante. España: club universitario.

Marco Luksa (2018). Kubernetes in Action. Shelter Island. Manning Publications

Mary E. Shacklett (2021), What is authentication? obtenido de https://www.techtarget.com/searchsecurity/definition/authentication

OWASP (2021), Kubernetes Security Cheat Sheet, Obtenido de https://cheatsheetseries.owasp.org/cheatsheets/Kubernetes_Security_Cheat_Sheet.html

Roberto Hernández, Carlos Fernadez y Pilar Baptista (2014). Metodología de la investigación (6ta. Edicion). México. Editorial McGraw-Hill.

Sabino C. (2006). Como hacer una tesis y elaborar toda clase de escritos. Venezuela. Editorial Panapo.

Scharager, J y Armijo, I. (2001). Metodología de la Investigación para las Ciencias Sociales. Santiago. Escuela de Psicología, SECICO Pontificia Universidad Católica de Chile.

Seguridad Nacional de los Estados Unidos (2021). Kubernetes Hardening Guidance. Estados Unidos.

Tamayo y Tamayo, M. (2006). El proceso de la investigación científica. México. Limusa Noriega Editores.

Tigera (2021), Kubernetes Security: Risks, Security Controls, and Best Practices, obtenido de https://www.tigera.io/learn/guides/kubernetes-security

Vmware (2021), The State of Kubernetes 2021. Estados Unidos

LISTA DE ABREVIATURAS

OIDC	: Open ID Connect
NSA	: Agencia de Seguridad Nacional
OWASP	: The Open Web Application Security Project
CNCF	: Cloud Native Computing Foundation
API	: Application Programming Interface
RBAC	: Control de Acceso Basado en Roles

GLOSARIO

Namespace: Separación lógica dentro de un clúster Kubernetes que facilita la organización de los objetos o recursos de acuerdo con la necesidad del administrador.

Role: Define un conjunto de permisos sobre los objetos que expone un clúster Kubernetes y que se aplican a nivel de Namespace.

Cluster Role: Define un conjunto de permisos sobre los objetos que expone un cluster Kubernetes y que se aplican a nivel del cluster.

Service Account: Representa una identidad asignada a una aplicación, es usada para establecer comunicación entre la aplicación y el servidor de APIs que expone el clúster.

ANEXOS

Anexo 1. Matriz de consistencia

Título: Aplicación para Gestión de Seguridad en Clúster Kubernetes, Entornos Privados, 2022.

PROBLEMAS	OBJETIVOS	VARIABLES	MÉTODO DE INVESTIGACIÓN
PROBLEMA GENERAL ¿De qué manera gestionar la seguridad en clúster Kubernetes, para entornos privados, 2022? **PROBLEMAS ESPECÍFICOS** a. ¿Cómo gestionar la autenticación? b. ¿De qué manera gestionar la autorización?	**OBJETIVO GENERAL** Desarrollar una aplicación que permita gestionar la seguridad en clúster Kubernetes, para entornos privados, 2022 **OBJETIVOS ESPECÍFICOS** a. Desarrollar una aplicación desktop para gestionar la autenticación. b. Implementar una aplicación desktop para gestionar la autorización.	**VARIABLE DE INTERÉS** X: Seguridad en clúster **VARIABLES DESCRIPTIVAS** X1: Autenticación X2: Autorización	**TIPO DE INVESTIGACIÓN** Observacional, prospectivo, transversal, descriptivo. **NIVEL DE INVESTIGACIÓN** Descriptivo **DISEÑO** No experimental, prospectivo transversal. **POBLACIÓN** La población está compuesta por 4 clústeres Kubernetes, en entornos privados, 2022. **MUESTRA** No existe muestra, es un censo, porque se hará el estudio para los 4 clústeres Kubernetes en entornos privados. **TÉCNICA** Entrevista **INSTRUMENTO** Guía para entrevista

Anexo 2. Certificado de validez de contenido del instrumento que mide la variable seguridad de clúster Kubernetes.

Señor juez se requiere su colaboración para determinar la validez del contenido del procedimiento de registro, también puede recomendar el retiro de artículos o la modificación del contenido de un artículo. Gracias por su cooperación.

Nº	**DIMENSIONES/ items**	**Relevancia**		**Pertinencia**		**Claridad**		**Observaciones**
Dimensión: Autenticación		**SI**	**NO**	**SI**	**NO**	**SI**	**NO**	
01	¿Cómo autenticar un clúster usando certificados digitales?							
02	¿Cómo obtener un certificado digital?							
03	¿Cómo crear una solicitud de firma de certificado?							
04	¿Cómo aprobar una solicitud de firma de certificado?							
05	¿Cómo rechazar una solicitud de firma de certificado?							
06	¿Cómo crear un usuario?							
07	¿Cómo asignar permisos a un usuario?							
08	¿Cómo crear un grupo?							
09	¿Cómo asignar permisos a un grupo?							
10	¿Cómo crear un Service account?							
11	¿Cómo eliminar un Service account?							
12	¿Cómo asignar permisos a un Service account?							
Dimensión: Autorización								
13	¿Cómo crear un Cluster Role?							
14	¿Cómo editar un Cluster Role?							
15	¿Cómo eliminar un Cluster Role?							

16	¿Cómo asociar un Cluster Role a un Grupo?							
17	¿Cómo asociar un Cluster Role a un Usuario?							
18	¿Cómo asociar un Cluster Role a un Service Account?							
19	¿Cómo crear un Role?							
20	¿Cómo editar un Role?							
21	¿Cómo eliminar un Role?							
22	¿Cómo asociar un Role a un Usuario?							
23	¿Cómo asociar un Role a un Service Account?							
24	¿Cómo asociar un Cluster Role a un Grupo?							
25	¿Cuáles son los permisos que puede tener una API de Kubernetes?							
26	¿Cuáles son los alcances que tiene una API de Kubernetes?							

Apellidos y nombres del juez validador Dr./Mg: ……

DNI: …….

Especialidad del validador: ……

1. Relevancia: El ítem es apropiado para representar al componente o dimensión específica del constructo.
2. Pertinencia: El ítem corresponde al concepto teórico formulado.
3. Claridad: Se entiende sin dificultad alguna el enunciado del ítem, es conciso, exacto y directo.

Nota: Suficiencia, se dice suficiencia cuando los ítems planteados son suficientes para medir la dimensión.

15 de mayo del 2022

Anexo 3. Instrumento de registro

GUÍA DE ENTREVISTA PARA EL EXPERTO EN KUBERNETES

Consentimiento informado:
La presente entrevista es a fin de poder contribuir con la investigación de tesis del alumno Christian Altamirano Ayala, dicha investigación se titula "Aplicación para Gestión de Seguridad en Clúster Kubernetes, Entornos Privados, 2022", que tiene como objetivo recabar información acerca de la seguridad en clúster Kubernetes on-premises. Por ello, la información proporcionada será solo para uso académico y se mantendrá la confidencialidad del entrevistado.

DATOS GENERALES DE LA ENTREVISTA	
Nombre:	Fecha:
Puesto:	Lugar:
Disposición:	Tiempo utilizado:
Nº de interrupciones:	Nº de Observaciones:

01. ¿Cómo autenticar un clúster usando certificados digitales?
02. ¿Cómo obtener un certificado digital?
03. ¿Cómo crear una solicitud de firma de certificado?
04. ¿Cómo aprobar una solicitud de firma de certificado?
05. ¿Cómo rechazar una solicitud de firma de certificado?
06. ¿Cómo crear un usuario?
07. ¿Cómo asignar permisos a un usuario?
08. ¿Cómo crear un grupo?
09. ¿Cómo asignar permisos a un grupo?
10. ¿Cómo crear un Service Account?
11. ¿Cómo eliminar un Service Account?
12. ¿Cómo asignar permisos a un Service Account?
13. ¿Cómo crear un Cluster Role?
14. ¿Cómo editar un Cluster Role?
15. ¿Cómo eliminar un Cluster Role?
16. ¿Cómo asociar un Cluster Role a un Grupo?
17. ¿Cómo asociar un Cluster Role a un Usuario?
18. ¿Cómo asociar un Cluster Role a un Service Account?
19. ¿Cómo crear un Role?
20. ¿Cómo editar un Role?
21. ¿Cómo eliminar un Role?
22. ¿Cómo asociar un Role a un Usuario?
23. ¿Cómo asociar un Role a un Service Account?
24. ¿Cómo asociar un Cluster Role a un Grupo?
25. ¿Cuáles son los permisos que puede tener una API de Kubernetes?
26. ¿Cuáles son los alcances que tiene una API de Kubernetes?

Anexo 4. Cálculo de la V de Aiken con intervalos de confianza

Inserte valores

min	***0***
max	***1***
k	***1***
n	**4**
sig	**1.96**

95%

									Intervalo de Confianza	
		Juez 1	**Juez 2**	**Juez 3**	**Juez 4**	**Media**	**DE**	**V de Aiken**	**Interpretación V**	**Inferior**	**Superior**
Item 1	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 2	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 3	Relevancia	1	1	0	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 4	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 5	Relevancia	1	1	1	0	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	0	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
Item 6	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 7	Relevancia	1	0	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00

	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 8	Relevancia	1	0	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 9	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 10	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	0	1.00	0.00	0.75	VALIDO	0.57	1.00
Item 11	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 12	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	0	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 13	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 14	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	0	1.00	0.00	0.75	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 15	Relevancia	0	1	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 16	Relevancia	1	1	0	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	0	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 17	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00

	Representatividad	1	1	0	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 18	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 19	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	0	0	1	1.00	0.00	0.5	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 20	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 21	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 22	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	0	1	1.00	0.00	0.75	VALIDO	0.57	1.00
Item 23	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 24	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	1	0	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 25	Relevancia	1	1	1	0	1.00	0.00	0.75	VALIDO	0.57	1.00
	Representatividad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
Item 26	Relevancia	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00
	Representatividad	0	1	1	1	1.00	0.00	0.75	VALIDO	0.57	1.00
	Claridad	1	1	1	1	1.00	0.00	1	VALIDO	0.57	1.00

CONSTANCIA DE ORIGINALIDAD DE TRABAJO DE INVESTIGACIÓN

CONSTANCIA N° 035-2022-FIMGC

El que suscribe; responsable verificador de originalidad de trabajos de tesis de pregrado en segunda instancia para las **Escuelas Profesionales** de la **Facultad de Ingeniería de Minas, Geología y Civil**; en cumplimiento a la Resolución de Consejo Universitario N° 039-2021-UNSCH-CU, Reglamento de Originalidad de Trabajos de Investigación de la UNSCH y Resolución Decanal N° 158-2021-FIMGC-UNSCH-D, deja constancia que Sr./Srta.

Apellidos y Nombres	:	ALTAMIRANO AYALA, Christian
Escuela Profesional	:	INGENIERÍA DE SISTEMAS
Título de la Tesis	:	APLICACIÓN PARA GESTIÓN DE SEGURIDAD EN CLÚSTERS KUBERNETES, ENTORNOS PRIVADOS, 2022
Evaluación de la Originalidad	:	**27 %** Índice de Similitud
Identificador de la entrega	:	1855261762

Por tanto, según los Artículos 12, 13 y 17 del Reglamento de Originalidad de Trabajos de Investigación, es **PROCEDENTE** otorgar la **Constancia de Originalidad** para los fines que crea conveniente.

Ayacucho, 12 de junio del 2022

Firmado digitalmente por LEZAMA CUELLAR CHRISTIAN

Mg. Ing. Christian LEZAMA CUELLAR
Verificador de Originalidad de Trabajos de Tesis de Pregrado de la FIMGC

(X) Con depósito para Sustentación y Tramite de Titulo

APLICACIÓN PARA GESTIÓN DE SEGURIDAD EN CLÚSTERS KUBERNETES, ENTORNOS PRIVADOS, 2022

por Christian Altamirano Ayala

Fecha de entrega: 12-jun-2022 08:20a.m. (UTC-0500)
Identificador de la entrega: 1855261762
Nombre del archivo: Tesis_Christian,_ALTAMIRANO_AYALA_EPIS.pdf (14.51M)
Total de palabras: 15979
Total de caracteres: 81746

APLICACIÓN PARA GESTIÓN DE SEGURIDAD EN CLÚSTERS KUBERNETES, ENTORNOS PRIVADOS, 2022

INFORME DE ORIGINALIDAD

27%	20%	2%	19%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	Submitted to Universidad Nacional de San Cristóbal de Huamanga Trabajo del estudiante	8%
2	repositorio.unsch.edu.pe Fuente de Internet	2%
3	vsip.info Fuente de Internet	1%
4	1library.co Fuente de Internet	1%
5	upc.aws.openrepository.com Fuente de Internet	1%
6	www.cua.uam.mx Fuente de Internet	1%
7	Submitted to tec Trabajo del estudiante	1%
8	proyectoingesoft.blogspot.com Fuente de Internet	1%

9	repositorio.ucv.edu.pe Fuente de Internet	1 %
10	www.slideshare.net Fuente de Internet	1 %
11	www.scribd.com Fuente de Internet	1 %
12	repositorio.unan.edu.ni Fuente de Internet	1 %
13	Submitted to Universidad Internacional de la Rioja Trabajo del estudiante	1 %
14	Submitted to Universidad Cesar Vallejo Trabajo del estudiante	1 %
15	repositorio.uta.edu.ec Fuente de Internet	1 %
16	repositorio.untels.edu.pe Fuente de Internet	<1 %
17	Submitted to Universidad Abierta para Adultos Trabajo del estudiante	<1 %
18	dokumen.pub Fuente de Internet	<1 %
19	repositorio.utc.edu.ec Fuente de Internet	<1 %

20	repositorio.urp.edu.pe Fuente de Internet	<1 %
21	autosist.blogspot.com Fuente de Internet	<1 %
22	docs.citrix.com Fuente de Internet	<1 %
23	www.scrum.org Fuente de Internet	<1 %
24	repositorio.uca.edu.ni Fuente de Internet	<1 %
25	www.coursehero.com Fuente de Internet	<1 %
26	es.scribd.com Fuente de Internet	<1 %
27	www.research.manchester.ac.uk Fuente de Internet	<1 %
28	Submitted to Instituto Madrileno de Formacion Trabajo del estudiante	<1 %
29	rd.udb.edu.sv:8080 Fuente de Internet	<1 %
30	Submitted to COLEGIO NACIONAL EXPERIMENTAL AMBATO Trabajo del estudiante	<1 %

31	repo.uta.edu.ec Fuente de Internet	<1 %

Excluir citas	Activo	Excluir coincidencias	< 30 words
Excluir bibliografía	Activo		