

Universidad Mariano Gálvez de Guatemala
Facultad de Ingeniería en Sistemas y Ciencias de la Computación
Arquitectura de computadoras II
Inge. Alejandro García

Informe: YOLO-Edge Pi Cluster

Guatemala, 11 de octubre de 2026

Nombre	No. Carné
Rogelio Andreé Ical Zuñiga	0900 23 23236
Leonel Alejandro Lira Ventura	0900 23 1349
Fátima Palacios Ordoñez	0900 21 1810
Pablo David Pérez del Aguila	0900 23 2032

Índice

Introducción.....	pág. 3
Definiciones generales.....	pág. 4-5
Objetivo general.....	pág. 6
Objetivos específicos.....	pág. 7
Arquitectura de red.....	pág. 8
Diseño general.....	pág. 8
Topología de red.....	pág. 8-9
Direccionamiento IP.....	pág. 9-10
Servicios y puertos de comunicación.....	pág. 10
Segmentación lógica.....	pág. 11
Alta disponibilidad y tolerancia a fallos.....	pág.11-12
Configuración técnica.....	pág. 12
Preparación de la Raspberry Pi 400.....	pág. 12
Configuración de red en la Raspberry Pi.....	pág.12-13
Instalación de K3s Server en el nodo Master.....	pág. 13
Instalación de K3s Agent en el nodo Worker.....	pág.13-14
Despliegue del modelo de inteligencia artificial.....	pág. 14
Instalación de las herramientas de monitoreo.....	pág.14-15
Configuración del dashboard de monitoreo.....	pág. 15
Tolerancia a fallos y alta disponibilidad.....	pág.16-17
Conclusión de la prueba.....	pág. 17
Dashboard de monitoreo.....	pág. 18
Componentes instalados.....	pág. 18
Dashboards importados.....	pág. 18
Métricas visualizadas.....	pág.18-19
Capturas del dashboard.....	pág.19-22
Inferencia de visión por computadora en tiempo real.....	pág. 22
Implementación.....	pág.22-23
Funcionamiento del sistema.....	pág. 23
Evidencias de funcionamiento.....	pág.24-25
Conclusión de la implementación.....	pág. 25
Capturas de Formulario de registro.....	pág.26-30
Análisis legal.....	pág.31-33
Conclusiones.....	pág. 34
Recomendaciones.....	pág. 35

Introducción

Actualmente, muchas aplicaciones de inteligencia artificial dependen de servicios en la nube para poder procesar la información. Sin embargo, este modelo puede generar retrasos en la respuesta, un mayor uso del ancho de banda y la necesidad de mantener una conexión estable a internet. Estas limitaciones dificultan el uso de soluciones de IA en tiempo real, especialmente en lugares donde la conectividad es limitada.

Como respuesta a esta situación, el proyecto plantea el diseño e implementación de un clúster de Edge Computing de bajo consumo, capaz de ejecutar tareas de IA de manera local. La arquitectura utiliza una Raspberry Pi 400 como nodo Worker, encargada de ejecutar el modelo de visión por computadora YOLO-nano, mientras que una computadora personal funciona como nodo Master, administrando los recursos del clúster mediante K3s, una versión ligera de Kubernetes.

La infraestructura se conecta a través de una red privada local con direccionamiento IP estático dentro del segmento 192.168.100.X y utiliza un switch Gigabit Ethernet para la comunicación entre los dispositivos. También se contempla la tolerancia a fallos para mantener disponible el servicio de inferencia cuando uno de los nodos presenta una desconexión, siempre que el nodo restante tenga los recursos y la configuración necesarios para asumir la tarea. Para supervisar el funcionamiento de la infraestructura, se incorporan Prometheus y Grafana, herramientas que permiten recopilar métricas y visualizar el estado del clúster mediante un dashboard.

Este proyecto integra conocimientos de redes de computadoras, sistemas operativos, administración de contenedores, orquestación e inteligencia artificial en una solución práctica. Además, permite evaluar las ventajas de procesar datos cerca de su origen, reduciendo la dependencia de servicios externos. De esta manera, se busca demostrar que es posible desarrollar una infraestructura de inferencia de IA local utilizando hardware accesible y herramientas de código abierto, con un enfoque en la eficiencia, la autonomía y la disponibilidad del servicio.

Definiciones generales

- Edge Computing: Modelo de computación que permite procesar información cerca del dispositivo que la genera, reduciendo la necesidad de enviar datos a servidores remotos o a la nube.
- Clúster: Conjunto de equipos conectados entre sí que colaboran para ejecutar aplicaciones y compartir recursos de procesamiento.
- K3s: Versión ligera de Kubernetes, desarrollada para funcionar en equipos con recursos limitados y entornos donde se necesita una infraestructura sencilla de administrar.
- Kubernetes: Plataforma de código abierto que permite automatizar la instalación, ejecución, distribución y administración de aplicaciones basadas en contenedores.
- Pod: Unidad básica de ejecución en Kubernetes que agrupa uno o varios contenedores que comparten recursos de red y, cuando se configura, almacenamiento.
- Nodo Master: Equipo encargado de administrar el clúster, coordinar sus componentes y tomar decisiones sobre la programación y ejecución de las aplicaciones.
- Nodo Worker: Equipo que proporciona recursos de procesamiento para ejecutar las aplicaciones y los contenedores asignados por el plano de control del clúster.
- YOLO (You Only Look Once): Familia de modelos de inteligencia artificial utilizada para detectar y localizar objetos en imágenes o videos. Sus versiones pequeñas están diseñadas para reducir el consumo de recursos y facilitar la inferencia en dispositivos con capacidad limitada.
- Inferencia: Etapa en la que un modelo de inteligencia artificial previamente entrenado analiza datos nuevos y genera resultados o predicciones.
- Cuantización: Método de optimización que disminuye la precisión numérica utilizada para representar los parámetros de un modelo de IA, con el objetivo de reducir su tamaño y consumo de memoria.
- Tolerancia a fallos: Capacidad de una infraestructura para mantener sus funciones esenciales cuando ocurre un problema en alguno de sus componentes.
- Alta disponibilidad (HA): Propiedad de un sistema diseñado para mantener los servicios accesibles y reducir el tiempo de interrupción ante fallos o tareas de mantenimiento.
- Switch Gigabit Ethernet: Dispositivo de comunicación que conecta equipos dentro de una red local y permite transmitir datos a velocidades de hasta 1 Gbps por puerto compatible.
- Subred privada: Segmento de una red que utiliza direcciones IP reservadas para comunicaciones internas. Por ejemplo, 192.168.100.0/24 permite organizar dispositivos dentro de una red local.
- Direccionamiento IP estático: Configuración mediante la cual se asigna una

dirección IP fija a un dispositivo para mantener una identificación constante dentro de la red.

- Prometheus: Herramienta de monitoreo de código abierto que recopila y almacena métricas de sistemas y aplicaciones para facilitar el análisis de su rendimiento.
- Grafana: Plataforma que transforma datos de monitoreo en gráficos y paneles visuales para consultar el estado y el comportamiento de una infraestructura.
- WSL2 (Windows Subsystem for Linux 2): Funcionalidad de Windows que utiliza tecnologías de virtualización para ejecutar un entorno Linux con un kernel real, permitiendo utilizar herramientas y comandos de Linux desde Windows.
- Raspberry Pi 400: Computadora compacta basada en arquitectura ARM, integrada en un teclado y equipada con 4 GB de memoria RAM. Su tamaño y consumo energético la hacen útil para proyectos de Edge Computing y aprendizaje de sistemas informáticos.

Objetivo general

Diseñar e implementar un clúster de Edge Computing de bajo consumo que permita ejecutar modelos de inteligencia artificial localmente, reduciendo la dependencia de internet. Para ello, se utilizará hardware de bajo costo y herramientas ligeras de orquestación, con el propósito de mejorar la disponibilidad del servicio y permitir la continuidad de las operaciones ante posibles fallos en alguno de los nodos.

Objetivos específicos

1. Configurar una red local privada con direccionamiento IP estático dentro del segmento 192.168.100.0/24 para garantizar la comunicación entre los nodos del clúster.
2. Instalar y configurar K3s en los nodos Master y Worker para administrar y coordinar la ejecución de aplicaciones mediante contenedores.
3. Implementar un modelo de visión por computadora YOLO-nano en el clúster para realizar la detección de objetos en tiempo real mediante inferencia local.
4. Configurar mecanismos de recuperación ante fallos que permitan mantener disponible el servicio cuando un nodo Worker deje de funcionar, siempre que existan recursos suficientes en los nodos restantes.
5. Implementar un sistema de monitoreo con Prometheus y Grafana para recopilar métricas y visualizar el estado y rendimiento del clúster.
6. Elaborar documentación técnica de la arquitectura, los procedimientos de instalación, las configuraciones realizadas y los resultados obtenidos, facilitando su reproducción en entornos similares.
7. Investigar y describir el marco legal guatemalteco relacionado con las redes de telecomunicaciones, la protección de datos y los aspectos normativos aplicables a la implementación de soluciones de Edge Computing.

Arquitectura de red

Diseño general

El clúster de Edge Computing utiliza una red local privada para permitir la comunicación entre los equipos que forman parte de la infraestructura. La conexión física se realiza mediante un switch Gigabit Ethernet TP-Link TL-SG105 de cinco puertos, mientras que los dispositivos utilizan direcciones IP estáticas dentro de la subred 192.168.100.0/24.

La arquitectura se basa en un modelo Master-Worker, en el que cada equipo cumple una función específica:

- Nodo Master: computadora personal con WSL2 y Ubuntu, encargada de administrar el clúster mediante K3s y coordinar la ejecución de las aplicaciones.
- Nodo Worker: Raspberry Pi 400 que proporciona recursos de procesamiento para ejecutar el modelo de inteligencia artificial y realizar la detección de objetos localmente.

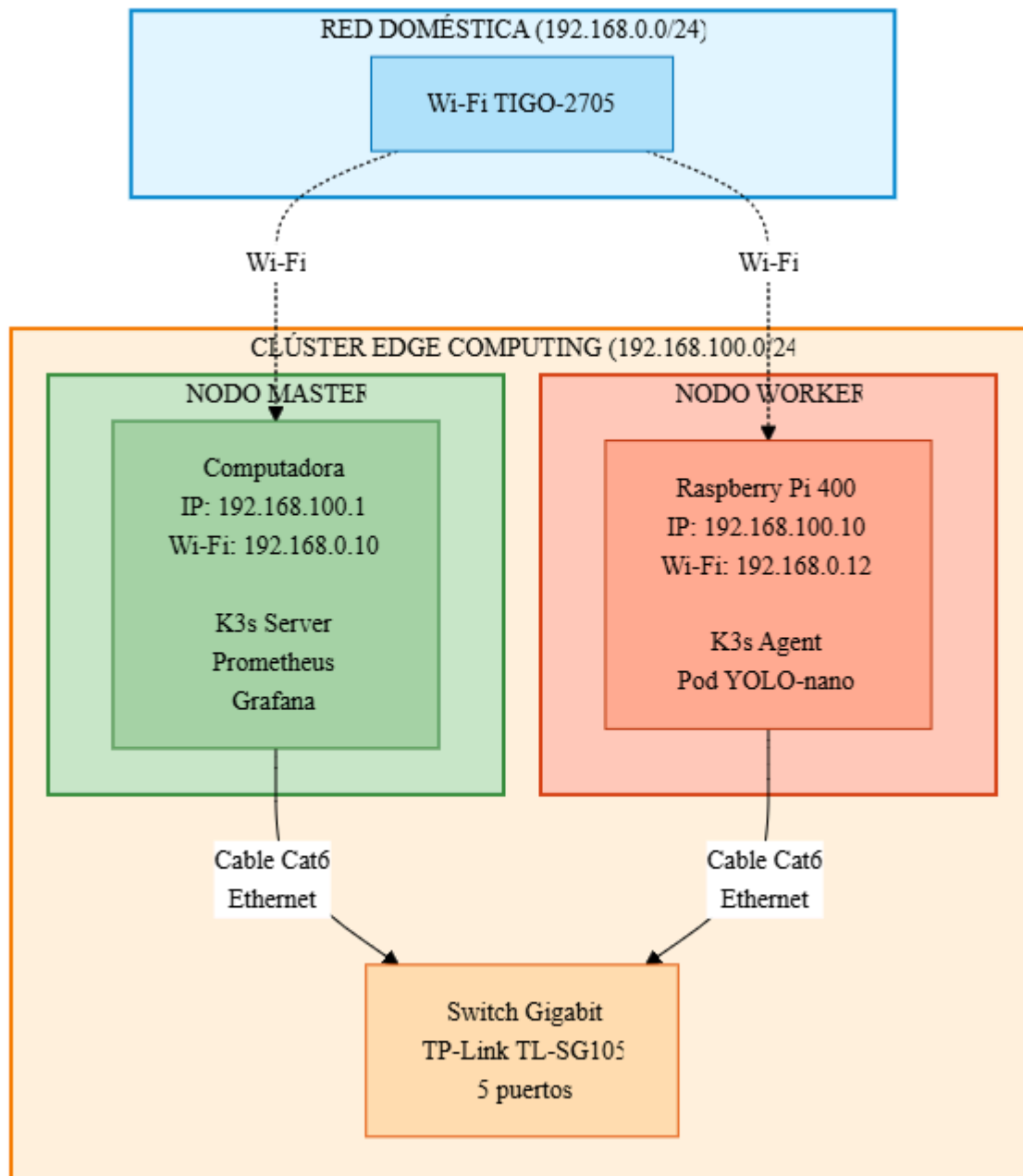
Esta distribución permite aprovechar los recursos de ambos equipos y centralizar la administración de las aplicaciones desde el nodo Master.

Topología de red

La topología conecta los nodos del clúster mediante una conexión Ethernet, utilizando el switch como punto de interconexión. Además, se contempla una red doméstica independiente para acceder a internet y descargar actualizaciones o paquetes necesarios para la instalación.

Elementos principales de la topología:

- Nodo Master: computadora personal con Ubuntu mediante WSL2 y K3s Server.
- Nodo Worker: Raspberry Pi 400 con K3s Agent.
- Switch Gigabit Ethernet: dispositivo encargado de interconectar físicamente los equipos.
- Red del clúster: subred privada 192.168.100.0/24.
- Red doméstica: subred 192.168.0.0/24, utilizada para el acceso a internet.



Direccionamiento IP

Para identificar los dispositivos y mantener una comunicación estable, se propone la siguiente configuración de direcciones IP:

Dispositivo	Interfaz	Dirección IP	Prefijo	Función
Computadora Master	Ethernet USB	192.168.100.1	/24	Administración del clúster
Raspberry Pi 400	eth0	192.168.100.10	/24	Ejecución de inferencia

Switch TP-Link TL-SG105	Ethernet	No aplica	No aplica	Interconexión física
Computadora Master	Wi-Fi	192.168.0.10	/24	Acceso a internet
Raspberry Pi 400	Wi-Fi	192.168.0.12	/24	Acceso a internet

Las direcciones indicadas corresponden al esquema de direccionamiento propuesto y deben verificarse con la configuración real de cada interfaz. El switch TL-SG105 es un dispositivo no administrable, por lo que no requiere una dirección IP para realizar la conmutación de tráfico.

Servicios y puertos de comunicación

Los servicios utilizados por el clúster requieren determinados puertos de comunicación para intercambiar información, administrar los contenedores y consultar las métricas del sistema.

Servicio	Puerto	Protocolo	Nodo	Función
K3s API Server	6443	TCP	Master	Administración del clúster
Kubelet	10250	TCP	Worker	Comunicación y gestión de los Pods
Prometheus	9090	TCP	Master	Consulta y almacenamiento de métricas
Grafana	3000	TCP	Master	Visualización de métricas
Servicio de inferencia YOLO	8000	TCP	Worker	Procesamiento de solicitudes de IA
SSH	22	TCP	Ambos	Administración remota
DNS	53	UDP/TCP	Según configuración	Resolución de nombres

Los puertos indicados deben habilitarse únicamente cuando los servicios correspondientes estén instalados y configurados. En particular, el puerto 53 no implica que el nodo Master funcione como servidor DNS; para ello sería necesario instalar y configurar un servicio específico.

Segmentación lógica

La infraestructura contempla dos segmentos de red con propósitos diferentes:

- **Red del clúster (192.168.100.0/24):** destinada a la comunicación entre los nodos Master y Worker, así como al intercambio de información necesario para ejecutar y administrar las aplicaciones.
- **Red doméstica (192.168.0.0/24):** utilizada para acceder a internet, descargar paquetes y realizar actualizaciones del sistema operativo y las herramientas instaladas.

La separación mediante interfaces de red diferentes permite organizar el tráfico según su función. Sin embargo, para garantizar el aislamiento entre los segmentos, es necesario configurar correctamente las rutas, las reglas de firewall y los servicios de red.

El procesamiento local reduce la dependencia de servicios externos, ya que las solicitudes de inferencia pueden atenderse dentro de la red del clúster. No obstante, las actualizaciones y las descargas de paquetes siguen requiriendo acceso a internet.

Alta disponibilidad y tolerancia a fallos

La arquitectura busca mejorar la disponibilidad del servicio de inferencia mediante la distribución de las aplicaciones entre los nodos del clúster. Para ello, se contempla ejecutar dos réplicas del servicio, procurando que cada una se encuentre en un nodo diferente.

K3s y Kubernetes permiten detectar cuándo un nodo deja de responder y administrar los Pods según el estado de la infraestructura. Si una réplica continúa activa en otro nodo y el servicio está configurado para dirigir las solicitudes hacia ella, es posible mantener la atención de peticiones durante la interrupción de uno de los equipos.

El tiempo de detección del fallo depende de los parámetros de supervisión del clúster, mientras que la recuperación depende de la disponibilidad de las réplicas, la distribución de las aplicaciones y la configuración del acceso al servicio.

Para comprobar la tolerancia a fallos, se deben realizar pruebas de desconexión controlada de un nodo, verificar el estado de los Pods y medir el tiempo durante el cual el servicio deja de responder, si ocurre una interrupción.

Es importante considerar que dos réplicas no garantizan por sí solas la alta disponibilidad. Si el nodo Master deja de funcionar y es el único que ejecuta el plano de control, la administración y recuperación del clúster pueden verse afectadas. Por ello, los resultados deben documentarse a partir de las pruebas realizadas.

Configuración técnica

Preparación de la Raspberry Pi 400

Para preparar el nodo Worker, se instaló Raspberry Pi OS de 64 bits en una tarjeta microSD de 32 GB mediante la herramienta Raspberry Pi Imager. Durante la creación del medio de instalación, se configuraron los parámetros básicos para identificar el equipo y permitir su administración remota.

La configuración inicial contempló los siguientes valores:

- Nombre del equipo: `nodo-worker`
- Usuario: `pi`
- Contraseña: definida por el equipo de trabajo.
- Conexión inalámbrica: red doméstica.
- Acceso remoto: SSH habilitado mediante autenticación por contraseña.

Esta preparación permitió disponer de un entorno Linux funcional para instalar el agente de K3s y ejecutar las aplicaciones del clúster.

Configuración de red en la Raspberry Pi

Para facilitar la comunicación entre los nodos, se asignó a la interfaz Ethernet de la Raspberry Pi la dirección IP estática `192.168.100.10`, correspondiente a la subred privada `192.168.100.0/24`.

La configuración de red debe realizarse mediante el administrador de red correspondiente al sistema operativo instalado. En algunas configuraciones de Raspberry Pi OS, la asignación de una IP estática puede gestionarse mediante NetworkManager o dhcpcd, según la versión y los servicios activos.

También se verificó la disponibilidad de los grupos de control (*cgroups*), necesarios para la administración de recursos de los contenedores ejecutados por K3s.

En sistemas que requieren habilitar explícitamente estos parámetros, se deben configurar de acuerdo con la versión del kernel y el método de arranque utilizado. Después de aplicar los cambios, se comprueba que la memoria y los demás recursos puedan ser gestionados correctamente por el entorno de contenedores.

Finalmente, se verificó la conectividad entre la Raspberry Pi y el nodo Master para confirmar que ambos equipos pudieran comunicarse a través de la red local.

Instalación de K3s Server en el nodo Master

En la computadora principal se preparó un entorno Ubuntu 22.04 mediante Windows Subsystem for Linux 2 (WSL2). Posteriormente, se instaló K3s Server, encargado de ejecutar el plano de control y administrar los recursos del clúster.

El procedimiento de instalación utilizado fue el siguiente: `curl -sfL https://rancher-mirror.rancher.cn/k3s/k3s-install.sh | INSTALL_K3S_MIRROR=cn sh -`

Este comando descarga y ejecuta el instalador de K3s utilizando un espejo de Rancher. Para emplearlo, es necesario disponer de conectividad con el servidor de descarga y verificar la procedencia del script antes de ejecutarlo.

Una vez finalizada la instalación, se consultó el estado del nodo mediante: `sudo kubectl get nodes`

Esta comprobación permite verificar que el plano de control esté disponible y que el nodo Master haya sido registrado correctamente. Para continuar con la instalación del Worker, también es necesario identificar la dirección IP accesible del servidor K3s y obtener el token de incorporación del agente.

Instalación de K3s Agent en el nodo Worker

Con el servidor K3s en funcionamiento, se instaló el agente en la Raspberry Pi 400 para incorporar al clúster. El agente permite que el equipo reciba instrucciones del plano de control y ejecute las aplicaciones asignadas.

El procedimiento empleado fue:

```
curl -sfL https://rancher-mirror.rancher.cn/k3s/k3s-install.sh | INSTALL_K3S_MIRROR=cn  
K3S_URL=https://192.168.100.1:6443 K3S_TOKEN=<TOKEN>  
INSTALL_K3S_EXEC="--node-ip 192.168.100.10" sh -
```

En este comando, `K3S_URL` identifica el servidor de K3s, mientras que `K3S_TOKEN` contiene el valor de autenticación necesario para incorporar el agente al clúster. El marcador `<TOKEN>` debe reemplazarse por el token real obtenido del nodo Master, sin publicarlo en el informe.

La opción `--node-ip` permite indicar la dirección IP que utilizará el agente para identificarse dentro de la red del clúster.

Después de completar la instalación, se verificó el registro y el estado de los nodos con: `sudo kubectl get nodes -o wide`

El resultado esperado es que ambos nodos aparezcan con estado `Ready`, lo cual indica que están registrados y disponibles para ejecutar cargas de trabajo. Esta condición debe confirmarse mediante la salida real del comando.

Despliegue del modelo de inteligencia artificial

Para ejecutar la inferencia de visión por computadora, se preparó un archivo de configuración YAML denominado `yolo-ha.yaml`. Este archivo define los recursos de Kubernetes necesarios para desplegar las aplicaciones de inferencia en los equipos disponibles.

Se contempló utilizar imágenes de contenedor compatibles con cada arquitectura de procesador: ARM64 para la Raspberry Pi 400 y la arquitectura correspondiente a la computadora utilizada como nodo Master.

El despliegue se aplica mediante el siguiente comando: `sudo kubectl apply -f yolo-ha.yaml`

La configuración debe especificar las imágenes de contenedor, el número de réplicas, los puertos de comunicación y las reglas de asignación de los Pods. Para distribuir las réplicas entre los nodos, es necesario utilizar mecanismos de programación de Kubernetes que eviten que ambas se ejecuten en el mismo equipo.

Una vez desplegados los recursos, se puede consultar su estado con: `sudo kubectl get deployments,pods -o wide`

Esta verificación permite revisar la disponibilidad de las réplicas, identificar el nodo donde se ejecuta cada Pod y detectar posibles problemas durante la descarga de imágenes o el inicio de los contenedores.

Instalación de las herramientas de monitoreo

Para supervisar el comportamiento del clúster, se incorporaron Prometheus y Grafana mediante Helm. Prometheus permite recopilar métricas del sistema y de los componentes compatibles, mientras que Grafana facilita su consulta mediante paneles gráficos.

La instalación se realizó con el paquete `kube-prometheus-stack`:

```
helm install monitoring prometheus-community/kube-prometheus-stack \
```

```
--namespace monitoring \  
--create-namespace \  
--set grafana.adminPassword='<CONTRASEÑA_SEGURA>' \  
--set prometheusOperator.admissionWebhooks.enabled=false \  
--set nodeExporter.enabled=false
```

El parámetro `--namespace monitoring` define el espacio de nombres destinado a los componentes de monitoreo, mientras que `--create-namespace` permite crearlo si todavía no existe.

La contraseña de Grafana debe reemplazarse por una credencial segura y no debe almacenarse en archivos públicos ni compartirse en el informe. Además, las opciones que deshabilitan los admission webhooks y Node Exporter modifican las capacidades y los mecanismos de funcionamiento del paquete, por lo que deben utilizarse únicamente cuando exista una razón técnica para ello.

Después de la instalación, se debe comprobar que los componentes estén en funcionamiento mediante: `sudo kubectl get pods -n monitoring`

Configuración del dashboard de monitoreo

Para consultar las métricas del clúster, se habilitó temporalmente el acceso al servicio de Grafana mediante un túnel de reenvío de puertos (port-forward).

El comando utilizado fue: `sudo kubectl port-forward -n monitoring svc/monitoring-grafana 3000:80 --address 0.0.0.0`

Esta configuración permite acceder a Grafana a través del puerto 3000 del entorno donde se ejecuta el comando. La opción `--address 0.0.0.0` hace que el puerto escuche en todas las interfaces disponibles, por lo que debe utilizarse con precaución y limitarse mediante reglas de firewall a los equipos autorizados.

Posteriormente, se configuraron paneles de monitoreo de Kubernetes para visualizar información sobre el estado de los nodos, los recursos disponibles y el comportamiento de los Pods.

Los dashboards predefinidos pueden importarse mediante sus identificadores correspondientes, siempre que sean compatibles con la versión instalada y dispongan de las fuentes de datos necesarias. Para comprobar el funcionamiento del sistema, se recomienda verificar que las métricas se actualicen correctamente y que los paneles reflejen los cambios en el estado de los nodos.

Tolerancia a fallos y alta disponibilidad

Prueba de fallo del nodo Worker

Para evaluar la tolerancia a fallos del clúster de Edge Computing, se realizó una prueba de desconexión del nodo Worker, con el objetivo de comprobar si el servicio de inferencia de inteligencia artificial podía continuar funcionando cuando uno de los equipos dejara de estar disponible.

Inicialmente, se verificó que ambos Pods estuvieran en ejecución y distribuidos entre los dos nodos del clúster:

- **yolo-pc:** ejecutándose en `laptop-pufurtb9`, correspondiente al nodo Master.
- **yolo-pi:** ejecutándose en `nodo-worker`, correspondiente a la Raspberry Pi 400.

A continuación, se desconectó la fuente de alimentación de la Raspberry Pi 400 para simular una falla física del equipo. Después de aproximadamente 40 segundos, K3s detectó la pérdida de comunicación y el nodo Worker cambió al estado `NotReady`.

Durante la prueba, el Pod `yolo-pc`, ubicado en la computadora principal, permaneció en ejecución y continuó atendiendo las solicitudes de inferencia. Esto permitió mantener disponible el servicio mientras la Raspberry Pi estaba desconectada.

Finalmente, se restableció la alimentación eléctrica de la Raspberry Pi. Una vez que el equipo inició nuevamente y recuperó la comunicación con el clúster, K3s reconoció el nodo y permitió que sus cargas de trabajo volvieran a ejecutarse según la configuración de Kubernetes.

Resultados de la prueba: Los resultados registrados durante la prueba se resumen en la siguiente tabla:

Métrica	Resultado
Tiempo de detección del fallo	Aproximadamente 40 segundos
Interrupción observada del servicio	0 segundos, según la prueba realizada
Tiempo de recuperación del nodo	Aproximadamente 60 segundos
Réplicas disponibles durante el fallo	1 (yolo-pc)

Los resultados indican que el clúster pudo mantener una réplica del servicio de inferencia activa mientras el nodo Worker permanecía desconectado. El tiempo de recuperación corresponde al período aproximado que necesitó la Raspberry Pi para volver a estar disponible dentro de la infraestructura.

Para confirmar que la interrupción del servicio fue realmente de cero segundos, es necesario verificar que las solicitudes de inferencia se hayan mantenido exitosas durante toda la prueba, mediante peticiones continuas o registros de disponibilidad.

Conclusión de la prueba

La prueba permitió comprobar que el servicio de inferencia podría continuar funcionando desde el nodo Master cuando la Raspberry Pi 400 dejó de estar disponible. La permanencia de la réplica **yolo-pc** en estado de ejecución evitó que la caída del Worker provocara la interrupción del servicio observado durante el ensayo.

Por lo tanto, los resultados respaldan la tolerancia a fallos frente a la desconexión de un nodo Worker, siempre que exista otra réplica operativa capaz de atender las solicitudes. Sin embargo, esta prueba no demuestra por sí sola una alta disponibilidad completa, ya que el nodo Master continúa siendo un posible punto único de fallo.

Para evaluar con mayor precisión la disponibilidad de la solución, se recomienda realizar pruebas adicionales con solicitudes continuas, registrar los tiempos de respuesta y comprobar el comportamiento del servicio ante la caída de cada nodo.

Dashboard de monitoreo

Componentes instalados

Para supervisar el funcionamiento del clúster de Edge Computing, se implementó un sistema de monitoreo basado en Prometheus y Grafana. Estas herramientas permiten recopilar información sobre los recursos utilizados y consultar el estado de los componentes de la infraestructura.

Los principales componentes son:

- **Prometheus:** recopila y almacena métricas del clúster con un intervalo de consulta configurado de 15 segundos.
- **Grafana:** presenta las métricas mediante paneles gráficos que facilitan el análisis del rendimiento de los nodos y las aplicaciones.
- **Alertmanager:** permite gestionar y distribuir alertas generadas a partir de reglas de monitoreo. En este proyecto no se configuraron notificaciones.
- **kube-state-metrics:** recopila información sobre el estado de los objetos de Kubernetes, como los nodos, los Pods y los Deployments.

Dashboards importados

Para facilitar la interpretación de los datos, se utilizaron paneles de Grafana orientados al monitoreo de recursos de Kubernetes. Estos permiten consultar información general del clúster y analizar el comportamiento de sus componentes.

Los paneles considerados son:

- **Kubernetes / Compute Resources / Cluster:** presenta una vista general del consumo de recursos del clúster.
- **Kubernetes / Compute Resources / Node (Pods):** permite revisar el uso de recursos asociado a los nodos y sus Pods.
- **Kubernetes / Compute Resources / Namespace (Pods):** facilita el análisis de los recursos utilizados por las aplicaciones dentro de cada espacio de nombres.

La información disponible en cada panel depende de las fuentes de datos, los exportadores y las métricas que estén correctamente configurados.

Métricas visualizadas

El dashboard permite consultar diferentes indicadores para evaluar el comportamiento de la infraestructura y detectar posibles problemas de rendimiento.

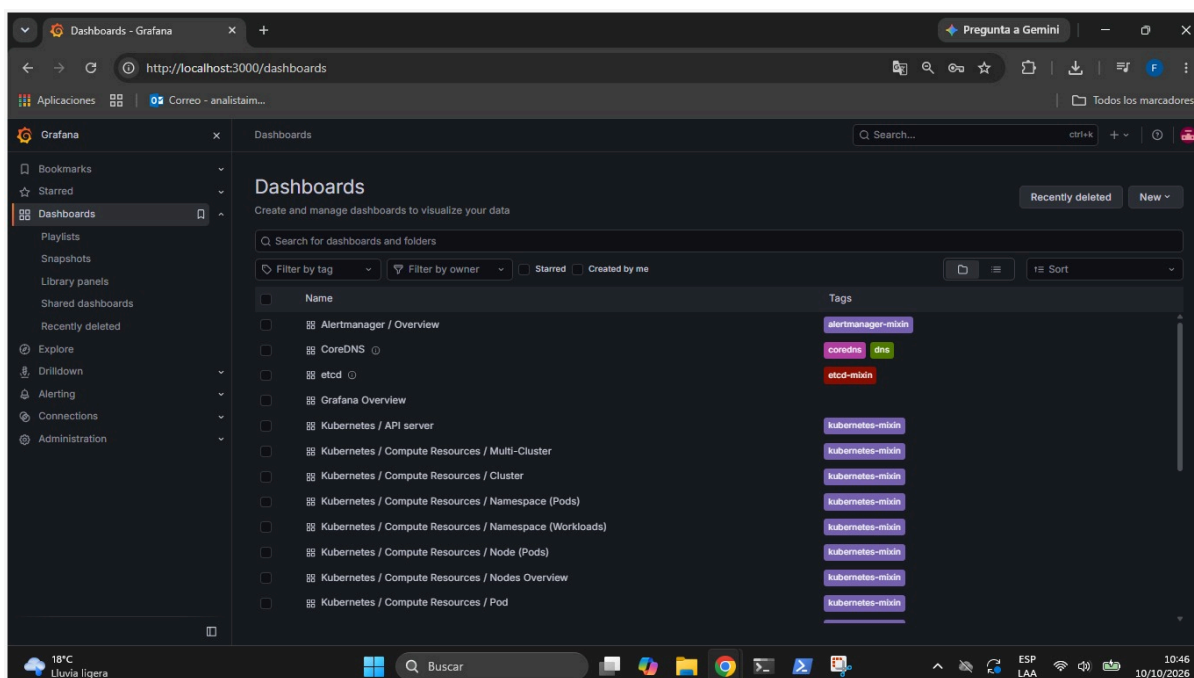
Las principales métricas consideradas son:

- **Uso de CPU:** permite observar el consumo del procesador en los nodos y las aplicaciones.
- **Uso de memoria RAM:** muestra la cantidad de memoria utilizada por los componentes supervisados.
- **Tráfico de red:** permite analizar el volumen de datos entrantes y salientes cuando las métricas correspondientes están disponibles.
- **Estado de los nodos:** facilita identificar si los equipos se encuentran disponibles (**Ready**) o presentan problemas de comunicación (**NotReady**).
- **Cantidad y estado de los Pods:** permite revisar las aplicaciones desplegadas y comprobar cuáles están en ejecución.
- **Tiempo de actividad:** ayuda a evaluar la continuidad operativa de los componentes supervisados.

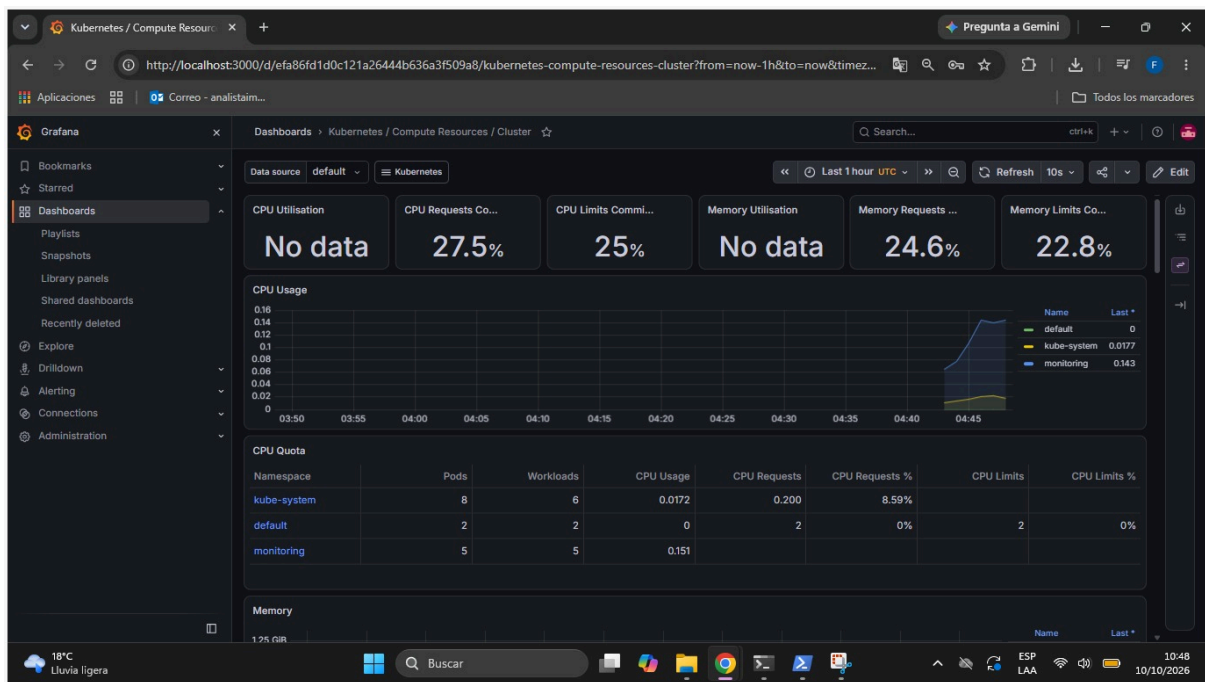
Estas métricas son útiles para identificar cambios en el consumo de recursos, comprobar el comportamiento de las aplicaciones y analizar los efectos de las pruebas de tolerancia a fallos.

Capturas del dashboard

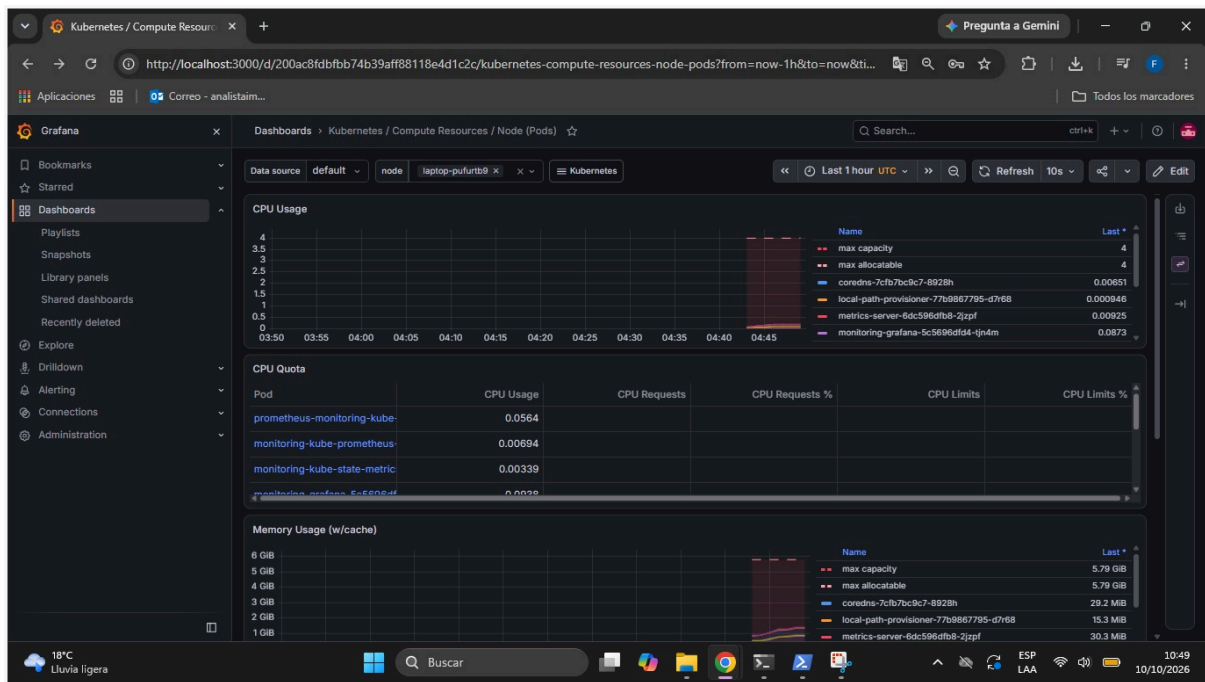
Lista de dashboards



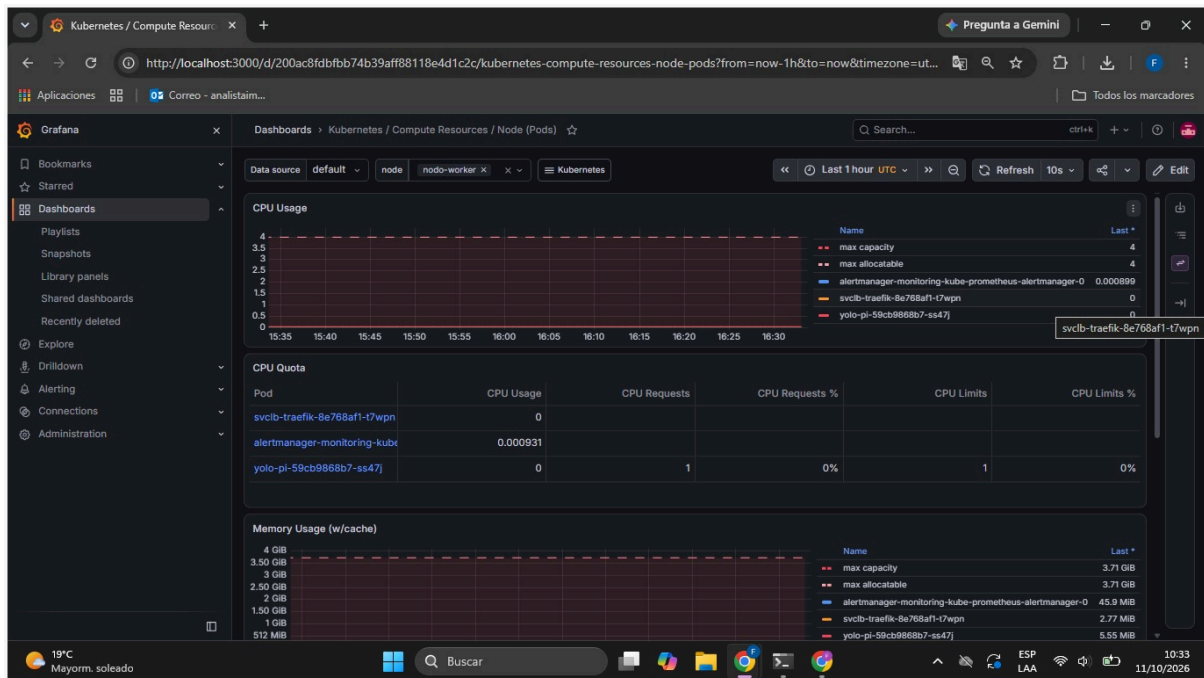
Dashboard "Kubernetes / Compute Resources / Cluster"



Dashboard "Kubernetes / Compute Resources / Node (Pods)"



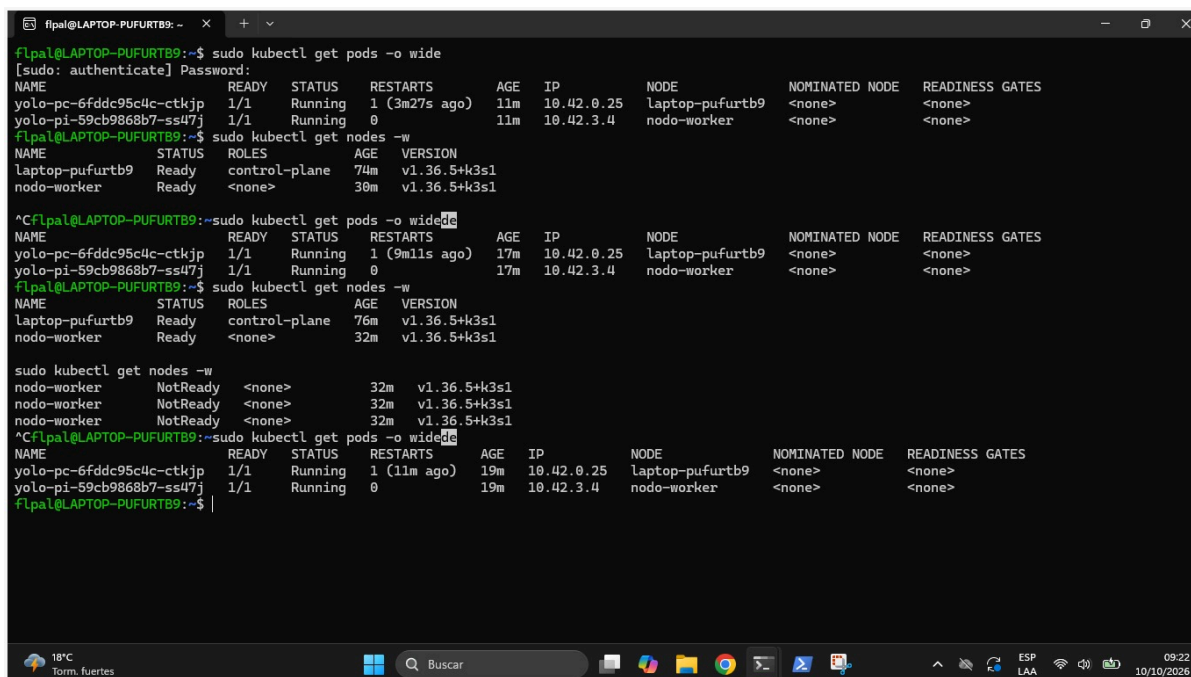
Dashboard del nodo Worker



Port-forward + kubectl get pods -o wide

```
fLpal@LAPTOP-PUFURTB9:~$ sudo kubectl port-forward -n monitoring svc/monitoring-grafana 3000:80 --address 0.0.0.0
Forwarding from 0.0.0.0:3000 -> 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
^CfLpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP              NODE              NOMINATED NODE   READINESS GATES
yolo-pc-6fddc95cd4-ctkjp   1/1     Running   4 (30m ago)   13h   10.42.0.39     laptop-pufurtb9   <none>           <none>
yolo-pi-59cb9868b7-ss47j   1/1     Running   4 (30m ago)   13h   10.42.3.13     nodo-worker       <none>           <none>
fLpal@LAPTOP-PUFURTB9:~$
```

Prueba de tolerancia a fallos completa



```
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
[sudo: authenticate] Password:
NAME                READY   STATUS    RESTARTS   AGE   IP            NODE                NOMINATED NODE   READINESS GATES
yolo-pc-6fddc95c4c-ctkjp   1/1     Running   1 (3m27s ago)    11m   10.42.0.25    laptop-pufurtb9     <none>           <none>
yolo-pi-59cb9868b7-ss47j   1/1     Running   0              11m   10.42.3.4     nodo-worker         <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes -w
NAME                STATUS   ROLES    AGE   VERSION
laptop-pufurtb9     Ready    control-plane   74m   v1.36.5+k3s1
nodo-worker         Ready    <none>        30m   v1.36.5+k3s1

^Cflpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP            NODE                NOMINATED NODE   READINESS GATES
yolo-pc-6fddc95c4c-ctkjp   1/1     Running   1 (9m11s ago)    17m   10.42.0.25    laptop-pufurtb9     <none>           <none>
yolo-pi-59cb9868b7-ss47j   1/1     Running   0              17m   10.42.3.4     nodo-worker         <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes -w
NAME                STATUS   ROLES    AGE   VERSION
laptop-pufurtb9     Ready    control-plane   76m   v1.36.5+k3s1
nodo-worker         Ready    <none>        32m   v1.36.5+k3s1

sudo kubectl get nodes -w
nodo-worker         NotReady    <none>        32m   v1.36.5+k3s1
nodo-worker         NotReady    <none>        32m   v1.36.5+k3s1
nodo-worker         NotReady    <none>        32m   v1.36.5+k3s1
^Cflpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP            NODE                NOMINATED NODE   READINESS GATES
yolo-pc-6fddc95c4c-ctkjp   1/1     Running   1 (11m ago)    19m   10.42.0.25    laptop-pufurtb9     <none>           <none>
yolo-pi-59cb9868b7-ss47j   1/1     Running   0              19m   10.42.3.4     nodo-worker         <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ |
```

Inferencia de visión por computadora en tiempo real

Para desarrollar el Caso 1 del proyecto, se implementó un sistema de visión por computadora utilizando el modelo YOLOv8n, una versión ligera de YOLO diseñada para detectar objetos en imágenes y videos. El procesamiento se realiza en la Raspberry Pi 400, utilizando una cámara USB conectada directamente al equipo.

La solución está diseñada para ejecutar la inferencia localmente, sin depender de servicios de inteligencia artificial en la nube. De esta manera, las imágenes capturadas se procesan en el dispositivo, reduciendo la necesidad de transmitir información a servidores externos.

Implementación

El desarrollo del sistema se realizó mediante los siguientes pasos:

1. **Conexión de la cámara:** se conectó una cámara USB Adesso CyberTrack H4 a uno de los puertos USB de la Raspberry Pi 400 para capturar imágenes en tiempo real.
2. **Instalación del entorno de desarrollo:** se instalaron OpenCV 4.10 y la biblioteca Ultralytics en un entorno virtual de Python. OpenCV permite

capturar y procesar imágenes, mientras que Ultralytics facilita la ejecución del modelo de detección de objetos.

3. **Descarga del modelo:** se utilizó el modelo preentrenado `yolov8n.pt`, correspondiente a la variante nano de YOLOv8, que requiere menos recursos que las versiones de mayor tamaño.
4. **Desarrollo del script de inferencia:** se creó el archivo `inference.py`, encargado de obtener los fotogramas de la cámara, enviarlos al modelo y procesar los resultados de detección.
5. **Creación de la imagen Docker:** se empaquetaron el código y las dependencias necesarias en una imagen denominada `yolo-inference:latest`, con el propósito de facilitar su ejecución dentro de un contenedor.
6. **Despliegue en K3s:** se configuró la aplicación para ejecutarse como un Pod dentro del clúster. Se habilitó el acceso al dispositivo `/dev/video0`, correspondiente a la cámara, para permitir la captura de imágenes desde el contenedor.

Funcionamiento del sistema

Una vez iniciado el contenedor, el script accede a la cámara y procesa los fotogramas capturados mediante el modelo YOLOv8n. Como resultado, el sistema identifica los objetos reconocidos por el modelo y genera las detecciones correspondientes.

La ejecución local permite realizar el procesamiento sin enviar las imágenes a servicios externos. El rendimiento obtenido depende de la capacidad de procesamiento de la Raspberry Pi 400, la resolución de los fotogramas y la configuración utilizada para ejecutar el modelo.

Esta implementación integra captura de video, procesamiento de imágenes, inferencia de inteligencia artificial y contenerización, aplicando los conceptos de Edge Computing al procesamiento de datos cerca de su fuente de origen.

Evidencias de funcionamiento

Contenedor de inferencia corriendo en la Raspberry Pi 400

```
Last login: Sun Oct 11 12:42:58 2026 from 192.168.188.1
pi@nodo-worker:~$ sudo k3s crictl ps | grep yolo-real
e00d6df5314ca      23a4788bd341f      13 minutes ago      Running      yolo      0      3fe7b266433d7      yolo-rea
l-7d45789d-l4mfz      default
pi@nodo-worker:~$ |
```

Logs de YOLO-nano detectando objetos en tiempo real

```
pi@nodo-worker:~$ sudo k3s crictl logs e00d6df5314ca
WARNING ⚠ user config directory '/root/.config/Ultralytics' is not writable, using '/tmp/Ultralytics'. Set YOLO_CONFIG_DIR to override.
Creating new Ultralytics Settings v0.0.8 file ✓
View Ultralytics Settings with 'yolo settings' or at '/tmp/Ultralytics/settings.json'
Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=path/to/dir'. For help see https://docs.ultralytics.com/usage/settings.
YOLO-nano inference iniciado
[18:37:45] Sin detecciones
[18:37:46] Sin detecciones
[18:37:47] Sin detecciones
[18:37:48] Sin detecciones
[18:37:49] Sin detecciones
[18:37:50] Detectado: [{'clase': 'person', 'confianza': 0.35}]
[18:37:51] Detectado: [{'clase': 'person', 'confianza': 0.47}]
[18:37:52] Detectado: [{'clase': 'person', 'confianza': 0.38}]
[18:37:53] Detectado: [{'clase': 'person', 'confianza': 0.56}]
[18:37:54] Detectado: [{'clase': 'person', 'confianza': 0.57}]
[18:37:55] Detectado: [{'clase': 'person', 'confianza': 0.44}, {'clase': 'person', 'confianza': 0.3}]
[18:37:56] Detectado: [{'clase': 'person', 'confianza': 0.5}, {'clase': 'person', 'confianza': 0.34}]
[18:37:57] Detectado: [{'clase': 'person', 'confianza': 0.51}, {'clase': 'person', 'confianza': 0.38}]
[18:37:58] Detectado: [{'clase': 'person', 'confianza': 0.53}]
[18:37:59] Detectado: [{'clase': 'person', 'confianza': 0.5}]
[18:38:00] Detectado: [{'clase': 'person', 'confianza': 0.46}, {'clase': 'person', 'confianza': 0.38}]
[18:38:01] Detectado: [{'clase': 'person', 'confianza': 0.48}, {'clase': 'person', 'confianza': 0.35}]
[18:38:02] Detectado: [{'clase': 'person', 'confianza': 0.52}, {'clase': 'person', 'confianza': 0.36}]
[18:38:03] Detectado: [{'clase': 'person', 'confianza': 0.53}, {'clase': 'person', 'confianza': 0.34}]
[18:38:04] Detectado: [{'clase': 'person', 'confianza': 0.48}, {'clase': 'person', 'confianza': 0.36}]
```

Estado de los Pods en el clúster K3s

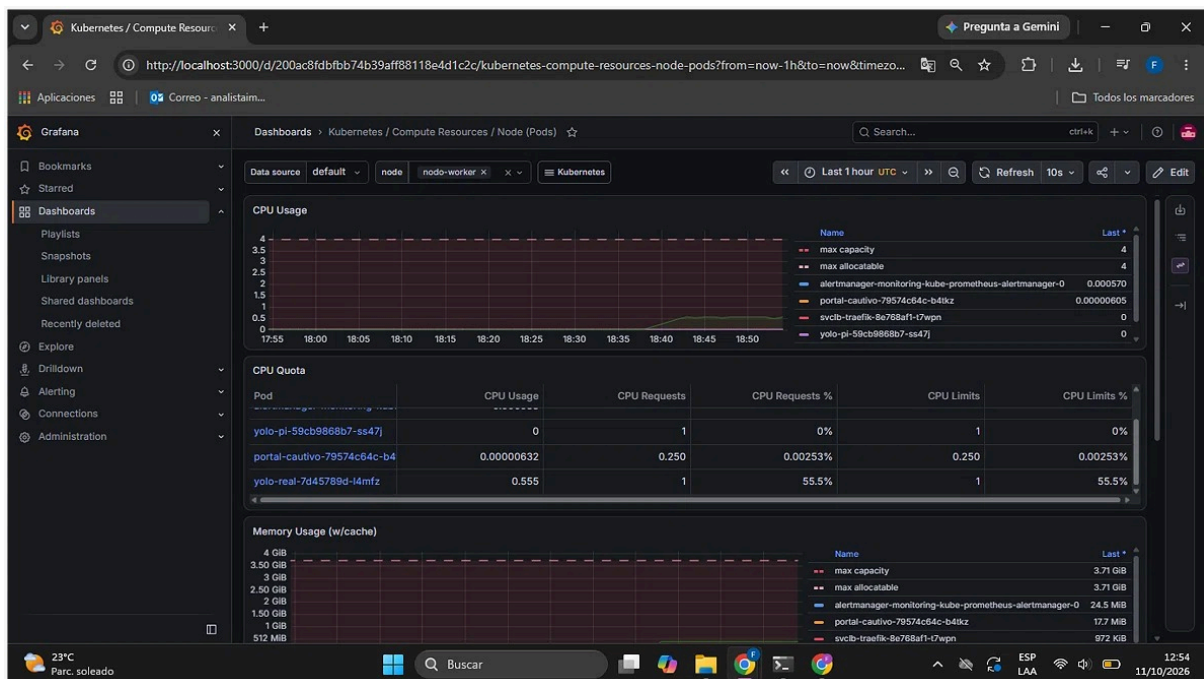
```
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
portal-cautivo-79574c64c-b4tkz 1/1 Running 0 15h 10.42.3.17 nodo-worker <none> <none>
yolo-pc-6fddc95c4c-ctkjp 1/1 Running 6 (38m ago) 15h 10.42.0.39 laptop-pufurtb9 <none> <none>
yolo-pi-59cb9868b7-ss47j 1/1 Running 6 (48m ago) 15h 10.42.3.13 nodo-worker <none> <none>
yolo-real-7d45789d-l4mfz 1/1 Running 0 15m 10.42.3.18 nodo-worker <none> <none>
flpal@LAPTOP-PUFURTB9:~$ |
```

Estado de los nodos del clúster

```
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION
laptop-pufurtb9 Ready control-plane 16h v1.36.5+k3s1 192.168.0.10 <none> Ubuntu 26.04.1 LTS 6.18.40.1-microsoft-st
andard-WSL2 (amd64) containerd://2.3.4-k3s1.36
nodo-worker Ready <none> 16h v1.36.5+k3s1 192.168.100.10 <none> Debian GNU/Linux 13 (trixie) 6.18.50+rpt-rpi-v8 (ar
m64) containerd://2.3.4-k3s1.36
flpal@LAPTOP-PUFURTB9:~$ |
```

Dashboard de Grafana mostrando el consumo de recursos del Pod de inferencia

```
FLpa1@LAPTOP-PUFURT89:~$ sudo kubectl port-forward -n monitoring svc/monitoring-grafana 3000:80 --address 0.0.0.0
Forwarding from 0.0.0.0:3000 -> 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
```



Conclusión de la implementación

El sistema de visión por computadora se ejecutó correctamente en el clúster de Edge Computing. La cámara USB permitió capturar imágenes y detectar objetos como personas, teléfonos celulares y vehículos. El procesamiento se realizó de forma local, sin depender de servicios en la nube, demostrando la capacidad del sistema para ejecutar tareas de inteligencia artificial directamente en los dispositivos del clúster.

Capturas de Formulario de registro

Portal Cautivo
Clúster de Edge Computing - Inferencia de IA Local

Paso 1: Registro

Nombre completo

Correo electrónico

Teléfono

Continuar al pago

Portal Cautivo
Clúster de Edge Computing - Inferencia de IA Local

Paso 1: Registro

Fatima Palacios

fpalacios2604@gmail.com

53673364

Continuar al pago

Formulario de pago

The screenshot shows a web browser window with the address bar displaying 'http://192.168.100.10:30080'. The page title is 'Portal Cautivo - Clúster Edge Computing - Inferencia de IA Local'. The main content area is a white card on a blue background. The card has a lock icon and the text 'Portal Cautivo' and 'Clúster de Edge Computing - Inferencia de IA Local'. Below this, it says 'Paso 2: Pago'. There is a summary box with 'Plan: Acceso al clúster de IA', 'Duración: 24 horas', and 'Precio: Q50.00'. Below the summary are three input fields: a long one with '5620001205208905', a date field with '02/28', and a short one with '123'. At the bottom of the card is a blue button labeled 'Pagar Q50.00'. The browser's taskbar at the bottom shows the Windows logo, a search bar with 'Buscar', and various application icons. The system tray shows '19°C Mayorm. nublado' and the date '12/10/2026'.

Portal Cautivo
Clúster de Edge Computing - Inferencia de IA Local

Paso 2: Pago

Plan: Acceso al clúster de IA
Duración: 24 horas
Precio: Q50.00

5620001205208905
02/28
123

Pagar Q50.00

Acceso liberado

The screenshot shows the 'Portal Cautivo' interface after payment. The card displays a green checkmark icon and the text 'Acceso Liberado'. Below this, it says 'Tu acceso al clúster de IA ha sido activado.' There is a green box with 'IP asignada: 192.168.100.20' and 'Válido hasta: 14/10/2026, 2:25:48 a. m.'. At the bottom of the card is a blue button labeled 'Acceder al clúster'.

Portal Cautivo
Clúster de Edge Computing - Inferencia de IA Local

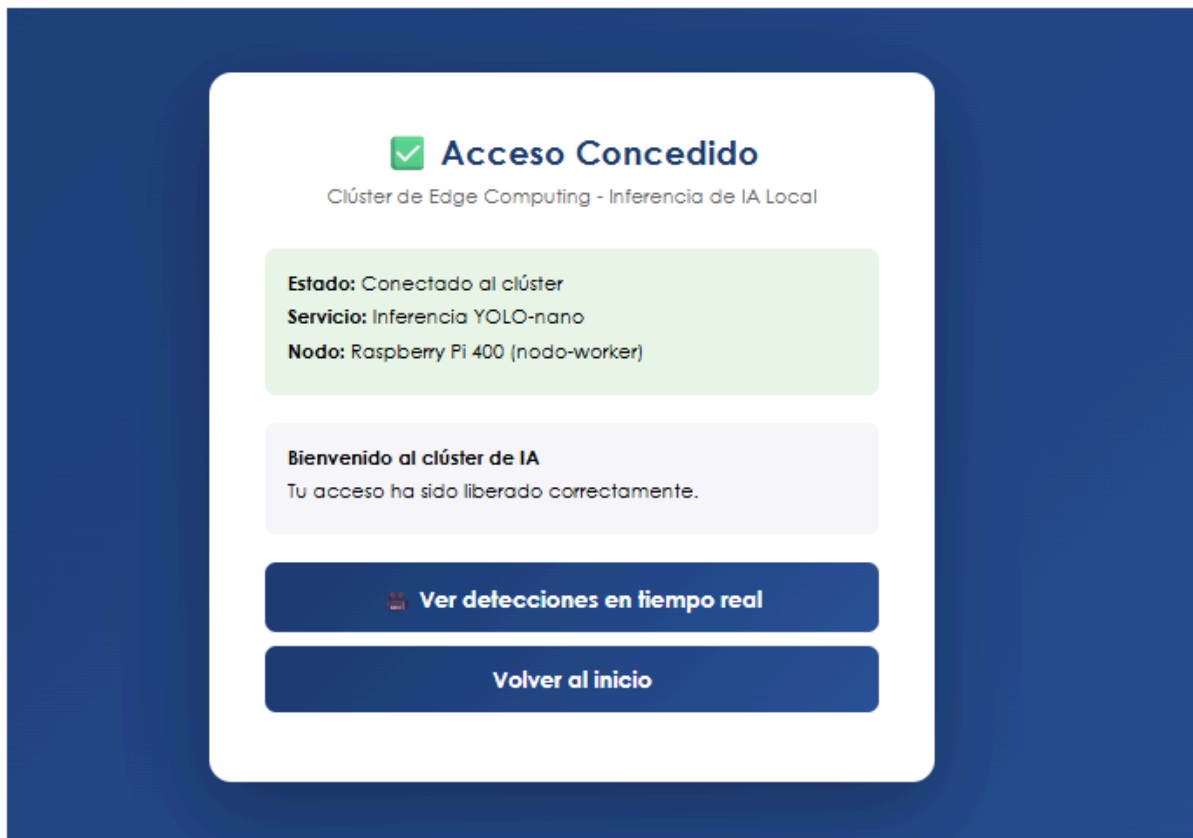
✓ **Acceso Liberado**

Tu acceso al clúster de IA ha sido activado.

IP asignada: 192.168.100.20
Válido hasta: 14/10/2026, 2:25:48 a. m.

Acceder al clúster

Acceso concedido



Galería de detecciones en tiempo real



El nodo Worker cambia a **NotReady**

```
edge1234
flpal@LAPTOP-PUFURTB9:~$
flpal@LAPTOP-PUFURTB9:~$ edge1234
edge1234: command not found
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes
[sudo: authenticate] Password:
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         Ready     <none>      47h   v1.36.5+k3s1
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP          NODE          NOMINATED NODE   READINESS GATES
portal-cautivo-7b99f8f857-nq6dd  0/1     ContainerStatusUnknown  7 (20h ago)  21h   <none>     nodo-worker   <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl delete pod portal-cautivo-7b99f8f857-nq6dd --force --grace-period=0
Warning: Immediate deletion does not wait for confirmation that the running resource has been terminated. The resource may continue to run on the cl
uster indefinitely.
pod "portal-cautivo-7b99f8f857-nq6dd" force deleted from default namespace
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
portal-cautivo      0/0     0            0           21h
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl scale deployment portal-cautivo --replicas=1
deployment.apps/portal-cautivo scaled
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP          NODE          NOMINATED NODE   READINESS GATES
portal-cautivo-74bd95ddd8-qt7lp  1/1     Running    0           5s    10.42.3.41  nodo-worker   <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes -w
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         Ready     <none>      47h   v1.36.5+k3s1
```

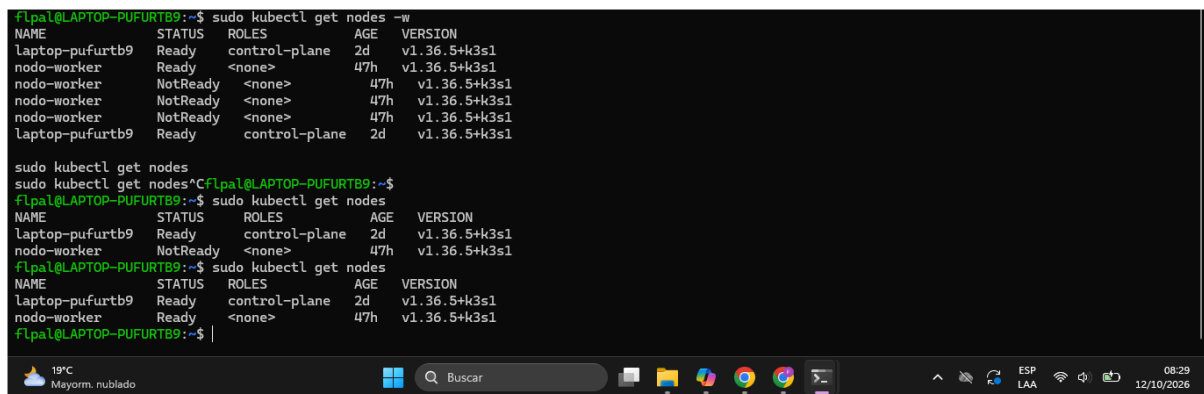
El Pod sobreviviente sigue en **Running**

```
edge1234
flpal@LAPTOP-PUFURTB9:~$
flpal@LAPTOP-PUFURTB9:~$ edge1234
edge1234: command not found
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes
[sudo: authenticate] Password:
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         Ready     <none>      47h   v1.36.5+k3s1
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP          NODE          NOMINATED NODE   READINESS GATES
portal-cautivo-7b99f8f857-nq6dd  0/1     ContainerStatusUnknown  7 (20h ago)  21h   <none>     nodo-worker   <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl delete pod portal-cautivo-7b99f8f857-nq6dd --force --grace-period=0
Warning: Immediate deletion does not wait for confirmation that the running resource has been terminated. The resource may continue to run on the cl
uster indefinitely.
pod "portal-cautivo-7b99f8f857-nq6dd" force deleted from default namespace
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
portal-cautivo      0/0     0            0           21h
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl scale deployment portal-cautivo --replicas=1
deployment.apps/portal-cautivo scaled
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP          NODE          NOMINATED NODE   READINESS GATES
portal-cautivo-74bd95ddd8-qt7lp  1/1     Running    0           5s    10.42.3.41  nodo-worker   <none>           <none>
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes -w
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         Ready     <none>      47h   v1.36.5+k3s1
nodo-worker         NotReady  <none>      47h   v1.36.5+k3s1
nodo-worker         NotReady  <none>      47h   v1.36.5+k3s1
nodo-worker         NotReady  <none>      47h   v1.36.5+k3s1
```

Prueba a tolerancia a fallos

```
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes -w
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         Ready     <none>     47h   v1.36.5+k3s1
nodo-worker         NotReady  <none>     47h   v1.36.5+k3s1
nodo-worker         NotReady  <none>     47h   v1.36.5+k3s1
nodo-worker         NotReady  <none>     47h   v1.36.5+k3s1
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1

sudo kubectl get nodes
sudo kubectl get nodes^Cflpal@LAPTOP-PUFURTB9:~$
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         NotReady  <none>     47h   v1.36.5+k3s1
flpal@LAPTOP-PUFURTB9:~$ sudo kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
laptop-pufurtb9     Ready     control-plane  2d   v1.36.5+k3s1
nodo-worker         Ready     <none>     47h   v1.36.5+k3s1
flpal@LAPTOP-PUFURTB9:~$
```



Análisis legal

Ley General de Telecomunicaciones de Guatemala: La Ley General de Telecomunicaciones, Decreto 94-96 del Congreso de la República de Guatemala, establece el marco jurídico para las actividades de telecomunicaciones y regula aspectos relacionados con el uso y aprovechamiento del espectro radioeléctrico.

En relación con el proyecto, se consideran los siguientes aspectos:

- **Uso del espectro radioeléctrico:** la comunicación principal entre los nodos del clúster se realiza mediante conexiones Ethernet por cable. Por lo tanto, este intercambio de datos no requiere utilizar el espectro radioeléctrico. Sin embargo, si se utiliza Wi-Fi para conectarse a la red doméstica, sí existe comunicación inalámbrica que opera en bandas de radiofrecuencia.
- **Prestación de servicios de telecomunicaciones:** el proyecto tiene una finalidad académica y está orientado a ejecutar inferencia de inteligencia artificial dentro de una red local. No se plantea ofrecer servicios de telecomunicaciones a terceros. Por ello, según el alcance descrito, no se identifica una necesidad evidente de autorización para prestar este tipo de servicios; cualquier actividad comercial futura tendría que evaluarse de acuerdo con la normativa aplicable.

Regulaciones de la Superintendencia de Telecomunicaciones (SIT)

La Superintendencia de Telecomunicaciones (SIT) es la entidad vinculada a la aplicación del marco normativo guatemalteco en materia de telecomunicaciones y administración del espectro radioeléctrico.

La infraestructura propuesta utiliza equipos de red convencionales para conectar una computadora personal y una Raspberry Pi 400, sin ofrecer conectividad pública ni comercializar servicios de telecomunicaciones. En consecuencia, las actividades descritas no parecen corresponder a la operación de un proveedor comercial de telecomunicaciones.

No obstante, esta evaluación se limita al alcance académico del prototipo. Si la solución se ampliara para ofrecer servicios a usuarios externos, utilizar recursos regulados o funcionar como parte de una actividad comercial, sería necesario revisar los requisitos legales correspondientes.

Protección de datos y privacidad

El proyecto utiliza un modelo de visión por computadora para detectar objetos mediante inferencia local. Aunque el procesamiento se realiza dentro del clúster, el uso de una cámara puede involucrar información personal si permite identificar a

las personas captadas.

En Guatemala, el Decreto 57-2008, Ley de Acceso a la Información Pública, incluye disposiciones relacionadas con la información pública y los datos personales en poder de los sujetos obligados por dicha ley. Su aplicación depende de la naturaleza de la entidad responsable y del contexto en el que se traten los datos.

Asimismo, durante 2026 continuaba el análisis legislativo de propuestas relacionadas con la protección de datos personales, incluida la iniciativa 6464. Por ello, no debe presentarse una iniciativa de ley como si ya fuera una norma vigente.

Para reducir los riesgos de privacidad, el proyecto contempla las siguientes medidas:

- Procesar las imágenes localmente, sin enviarlas a servicios externos de inteligencia artificial.
- Evitar el almacenamiento permanente de imágenes o videos cuando no sea necesario.
- Restringir el acceso al sistema y a los resultados de la inferencia.
- Informar sobre el uso de la cámara cuando corresponda y evitar captar información ajena a los objetivos de la prueba.

Estas medidas contribuyen a reducir la exposición de información. Sin embargo, el procesamiento local no garantiza por sí solo que no exista tratamiento de datos personales.

Conclusiones del análisis legal: A partir del alcance académico y de la arquitectura descrita, se identifican las siguientes conclusiones:

1. La comunicación principal entre los nodos se realiza mediante Ethernet, por lo que el tráfico cableado del clúster no depende del uso del espectro radioeléctrico.
2. El prototipo no contempla la comercialización de servicios de telecomunicaciones ni la prestación de conectividad pública.
3. El procesamiento local de imágenes reduce la necesidad de transmitir información a terceros, aunque deben considerarse los riesgos de privacidad si se captan personas identificables.
4. La Ley de Acceso a la Información Pública debe analizarse según su ámbito de aplicación, mientras que las iniciativas sobre protección de datos personales deben distinguirse de las leyes que ya están vigentes.
5. Si el proyecto se amplía a un entorno comercial o a una implementación que trate datos personales de terceros, será necesario realizar una revisión jurídica específica.

En conclusión, el uso académico de una red local y de herramientas de inteligencia artificial no permite afirmar, por sí solo, que el proyecto esté exento de todas las obligaciones legales. Su evaluación debe considerar la finalidad, los servicios ofrecidos, los equipos utilizados y el tipo de información procesada.

Conclusiones

1. El desarrollo del proyecto permitió implementar una arquitectura de Edge Computing que integra una Raspberry Pi 400 y una computadora personal para ejecutar y administrar aplicaciones de inteligencia artificial mediante K3s.
2. La configuración de una red privada con direccionamiento IP estático facilitó la comunicación entre los nodos y permitió organizar el tráfico necesario para el funcionamiento del clúster.
3. La utilización de K3s demostró que es posible administrar aplicaciones contenerizadas mediante una plataforma ligera, adecuada para entornos con recursos de hardware limitados.
4. El despliegue del modelo YOLO-nano permitió explorar la ejecución de tareas de detección de objetos de forma local, reduciendo la dependencia de servicios de procesamiento en la nube.
5. La prueba de desconexión de la Raspberry Pi permitió observar que la réplica de inferencia alojada en la computadora personal podía continuar ejecutándose. Este resultado demuestra tolerancia a la falla del Worker en las condiciones evaluadas, aunque no garantiza la disponibilidad ante la caída del nodo Master.
6. La integración de Prometheus y Grafana proporcionó herramientas para consultar métricas y supervisar el estado de los componentes del clúster, facilitando el análisis de su funcionamiento.
7. El análisis del marco legal guatemalteco permitió identificar aspectos relacionados con las telecomunicaciones, el uso de redes locales y la privacidad de los datos. La aplicación de las disposiciones legales debe evaluarse según las características y el alcance de cada implementación.
8. El proyecto permitió aplicar de manera práctica conocimientos de redes de computadoras, sistemas operativos, virtualización, orquestación de contenedores e inteligencia artificial, estableciendo una base técnica que puede utilizarse para desarrollar futuras soluciones de Edge Computing.

Recomendaciones

1. Mejorar la capacidad de procesamiento: considerar una Raspberry Pi 5 con 8 GB de RAM u otro dispositivo con mayores recursos para ejecutar modelos de inteligencia artificial más exigentes.
2. Ampliar la infraestructura: incorporar un segundo nodo Worker para aumentar la capacidad de procesamiento y facilitar la distribución de las aplicaciones entre varios equipos.
3. Evaluar el almacenamiento compartido: implementar una solución como NFS cuando sea necesario compartir modelos, archivos de configuración o datos entre los nodos, evitando duplicaciones innecesarias.
4. Fortalecer la seguridad de los servicios: utilizar HTTPS con certificados válidos en los servicios expuestos y limitar el acceso a los puertos de administración mediante reglas de firewall.
5. Automatizar los respaldos: establecer copias de seguridad periódicas de las configuraciones, manifiestos YAML, modelos y demás archivos necesarios para recuperar el entorno ante una falla.
6. Mejorar la autenticación remota: configurar el acceso SSH mediante claves públicas y deshabilitar la autenticación por contraseña cuando se haya comprobado que el acceso mediante claves funciona correctamente.
7. Utilizar control de versiones: almacenar los archivos de configuración, scripts y documentación en un repositorio Git para registrar los cambios y facilitar la instalación del proyecto en otros equipos.
8. Explorar hardware especializado en IA: evaluar dispositivos como NVIDIA Jetson Nano u otras plataformas con aceleración de GPU, comparando su rendimiento, consumo energético y compatibilidad con los modelos utilizados.
9. Realizar pruebas de rendimiento: medir el tiempo de inferencia, la cantidad de solicitudes procesadas, el consumo de CPU y memoria, y el comportamiento del servicio durante las pruebas de desconexión.
10. Fortalecer la alta disponibilidad: evaluar mecanismos adicionales de redundancia para el plano de control, la conectividad y los servicios de inferencia, con el propósito de reducir los puntos únicos de fallo y mejorar la continuidad operativa.