

## Documento de Análisis de Requisitos y Viabilidad

1. Introducción y Contexto del Proyecto
  - 1.1. Título y Resumen Ejecutivo:
  - 1.2. Justificación y Necesidad (El Problema):
  - 1.3. Objetivos del Proyecto:
2. Análisis de Viabilidad y Requisitos
  - 2.1. Viabilidad Operativa, Técnica y Económica:
  - 2.2. Requisitos Funcionales y No Funcionales:
  - 2.3. Herramientas y Stack Tecnológico Propuesto:
3. Planificación Inicial del Trabajo
  - 3.1. Estructura del Equipo:
  - 3.2. Metodología de Trabajo:
  - 3.3. Hitos Clave de la Fase:
4. Anexos y Referencias
  - 4.1. Webgrafía y Documentación Oficial

---

# Documento de Análisis de Requisitos y Viabilidad

## 1. Introducción y Contexto del Proyecto

### 1.1. Título y Resumen Ejecutivo:

- **Título del Proyecto:** Infraestructura de Alta Disponibilidad y Orquestación de Contenedores con Kubernetes sobre Proxmox.
- **Resumen Ejecutivo:** Este proyecto consiste en el diseño e implementación de una infraestructura *on-premise* virtualizada utilizando Proxmox VE. Sobre esta base, se desplegará un clúster de Kubernetes gestionado mediante automatización con Ansible. El objetivo es proporcionar un entorno resiliente, escalable y de alta disponibilidad para el despliegue de aplicaciones modernas contenerizadas, solucionando problemas de gestión manual y tiempos de inactividad en servicios web críticos.

### 1.2. Justificación y Necesidad (El Problema):

- **Identificación del Cliente/Necesidad:** El cliente es una PYME ficticia de desarrollo de software ("DevOps Solutions S.L.") que actualmente despliega sus aplicaciones web en servidores monolíticos tradicionales.
  - **El problema:** Sufren caídas de servicio cuando un servidor falla, la escalabilidad es manual y lenta, y los entornos de desarrollo no coinciden con los de producción. Necesitan modernizar su infraestructura para garantizar que sus servicios web estén disponibles 24/7 y se desplieguen automáticamente.
- **Motivación Técnica y Profesional:** Este proyecto consolida competencias clave del ciclo ASIR, integrando administración de sistemas (Linux), virtualización (Proxmox), redes

avanzadas (SDN, Balanceo de Carga) y automatización (Ansible). Además, introduce tecnologías de alta demanda en el mercado laboral actual como Kubernetes y metodologías GitOps, diferenciándose de la administración de sistemas clásica.

### 1.3. Objetivos del Proyecto:

- **Objetivo General:** Diseñar y poner en producción un clúster de Kubernetes totalmente funcional sobre hardware físico propio, capaz de orquestar contenedores y exponer servicios al exterior de forma segura y balanceada.
  - **Objetivos Específicos:**
    1. Instalar y configurar el hipervisor **Proxmox VE** en hardware físico.
    2. Automatizar el aprovisionamiento y configuración de máquinas virtuales mediante **Ansible**.
    3. Desplegar un clúster **Kubernetes** con arquitectura de 1 Control Plane y 2 Worker Nodes.
    4. Implementar almacenamiento persistente y gestión de red avanzada con **MetalLB** e **Ingress Controller**.
    5. Desplegar una aplicación web de prueba que demuestre la **Alta Disponibilidad** (self-healing) ante la caída de un nodo.
    6. Documentar todo el proceso utilizando control de versiones con **Git y GitHub**.
  - **Objetivo Deseable:** Implementación de un sistema de monitorización y visualización de métricas del clúster utilizando **Prometheus y Grafana**, condicionado a la disponibilidad de tiempo tras asegurar la estabilidad del núcleo del sistema.
- 

## 2. Análisis de Viabilidad y Requisitos

### 2.1. Viabilidad Operativa, Técnica y Económica:

- **Viabilidad Técnica:** El proyecto es viable ya que se dispone de un equipo físico con recursos suficientes (CPU/RAM) para soportar la virtualización anidada o bare-metal necesaria. Se cuenta con el *know-how* adquirido durante el ciclo formativo en administración Linux y redes, complementado con la investigación autodidacta en orquestación de contenedores.
  - **Justificación:** Se ha elegido **Kubernetes** frente a Docker Swarm por ser el estándar de facto en la industria, y **Proxmox** por su flexibilidad frente a soluciones propietarias como VMware ESXi.
- **Viabilidad Operativa y Temporal:** El alcance se ha ajustado para ser completado dentro del curso académico, priorizando la estabilidad del núcleo del clúster (Core) y el despliegue de aplicaciones. **La implementación de la pila de observabilidad (Prometheus y Grafana) se considera un hito de ampliación**, que se abordará únicamente si el cronograma permite

finalizar los objetivos prioritarios con antelación, o quedará documentada como la primera línea de actuación para futuras mejoras del sistema.

- **Viabilidad Económica (FOSS):** El coste de licencias es **cero**, ya que el 100% del stack tecnológico es Software Libre (Open Source).
  - Hipervisor: Proxmox (GNU AGPL, v3).
  - SO: Debian/Ubuntu (GPL).
  - Orquestador: Kubernetes (Apache 2.0). La viabilidad económica se garantiza al no requerir inversión en software, limitando el coste al hardware existente y al consumo eléctrico.

## 2.2. Requisitos Funcionales y No Funcionales:

- **Requisitos Funcionales:**
  - El sistema debe permitir el despliegue de aplicaciones web mediante ficheros YAML.
  - El sistema debe asignar automáticamente IPs de la red local a los servicios mediante MetalLB.
  - El sistema debe redirigir el tráfico HTTP basándose en nombres de dominio (Ingress).
  - El sistema debe recuperar automáticamente un contenedor (Pod) si este falla o se elimina.
  - La automatización (Ansible) debe ser capaz de configurar un nuevo nodo desde cero.
- **Requisitos No Funcionales:**
  - **Alta Disponibilidad:** El servicio web debe seguir respondiendo aunque uno de los nodos Worker se apague.
  - **Escalabilidad:** Debe permitir añadir nuevos nodos al clúster sin detener el servicio.
  - **Mantenibilidad:** Toda la configuración debe estar declarada en código (IaC) en el repositorio de GitHub.
  - **Usabilidad:** Acceso a los servicios web a través de URLs amigables, no por IPs y puertos.

## 2.3. Herramientas y Stack Tecnológico Propuesto:

- **Selección Tecnológica:**
  - **Hipervisor:** Proxmox VE.
  - **Sistemas Operativos:** Ubuntu Server 22.04 LTS (Nodos) y Ubuntu/Windows (Cliente).
  - **Orquestación y Contenedores:** Kubernetes (K8s) v1.28+, Containerd.
  - **Red y Balanceo:** MetalLB (Load Balancer L2), Nginx Ingress Controller.
  - **Automatización:** Ansible.
  - **Control de Versiones:** Git y GitHub.

- **Justificación de la Elección:** Se utilizan estas herramientas por ser líderes en el ecosistema **Cloud Native** y **FOSS**.
    - **Kubernetes** permite una gestión de cargas mucho más eficiente que las máquinas virtuales tradicionales.
    - **Ansible** elimina el error humano en configuraciones repetitivas (idempotencia).
    - **MetalLB** es indispensable para simular un entorno de producción real en una red doméstica (*on-premise*) donde no existen balanceadores de carga de proveedores cloud.
- 

### 3. Planificación Inicial del Trabajo

#### 3.1. Estructura del Equipo:

- **Administrador del Proyecto (Alumno):** Asume todos los roles técnicos y de gestión (SysAdmin, DevOps Engineer, Documentador).
  - Responsable de la infraestructura física y virtual.
  - Desarrollador de scripts de automatización e integración.

#### 3.2. Metodología de Trabajo:

- **Metodología:** Kanban.
- **Herramienta:** GitHub Projects (Tablero integrado en el repositorio).
- **Justificación:** Al ser un proyecto individual, Kanban permite una visualización directa del flujo de trabajo ("Por hacer", "En progreso", "Hecho") sin la sobrecarga administrativa de las ceremonias de Scrum (Sprints, Dailies), permitiendo flexibilidad según la carga de trabajo de otras asignaturas.

#### 3.3. Hitos Clave de la Fase:

| Fase     | Tarea                                            | Estado     |
|----------|--------------------------------------------------|------------|
| Análisis | Definición de la arquitectura del sistema.       | Completado |
| Análisis | Selección de Hardware y Stack de Software.       | Completado |
| Análisis | Creación del Repositorio y Estructura en GitHub. | Completado |
| Diseño   | Instalación de Proxmox en hardware físico.       | Pendiente  |
| Diseño   | Desarrollo de Playbooks de Ansible iniciales.    | Pendiente  |

| Fase           | Tarea                                            | Estado    |
|----------------|--------------------------------------------------|-----------|
| Implementación | Despliegue del Cluster Kubernetes (Init & Join). | Pendiente |

---

## 4. Anexos y Referencias

### 4.1. Webgrafía y Documentación Oficial

A continuación se detallan las fuentes de información primarias utilizadas para el análisis, diseño e implementación de la infraestructura:

#### A. Virtualización y Sistema Base

- **Proxmox VE Administration Guide:** Documentación oficial para la instalación, configuración de almacenamiento y gestión de redes en el hipervisor.
  - Enlace: <https://pve.proxmox.com/pve-docs/>
- **Ubuntu Server Documentation:** Guías de referencia para la configuración del sistema operativo base de los nodos.
  - Enlace: <https://ubuntu.com/server/docs>

#### B. Orquestación y Contenedores (Kubernetes)

- **Kubernetes Documentation:** Fuente principal para los conceptos de arquitectura (Control Plane, Nodes), despliegue de Pods y Servicios.
  - Enlace: <https://kubernetes.io/docs/home/>
- **Containerd Getting Started:** Referencia para la configuración del runtime de contenedores compatible con CRI.
  - Enlace: <https://github.com/containerd/containerd/blob/main/docs/getting-started.md>
- **Herramienta Kubeadm:** Documentación específica para la inicialización del clúster (bootstrapping).
  - Enlace: <https://kubernetes.io/docs/reference/setup-tools/kubeadm/>

#### C. Redes y Balanceo de Carga

- **MetalLB Documentation:** Guía de implementación para el balanceador de carga en entornos *bare-metal* (capa 2).
  - Enlace: <https://metallb.universe.tf/>
- **Ingress Nginx Controller:** Documentación de la comunidad de Kubernetes para la configuración del controlador de entrada y reglas de enrutamiento.
  - Enlace: <https://kubernetes.github.io/ingress-nginx/>

#### D. Automatización y Gestión

- **Ansible Documentation:** Manuales de referencia para la creación de inventarios, Playbooks y uso de módulos.
  - Enlace: <https://docs.ansible.com/>
- **Proxmox Ansible Collection:** Documentación de los módulos comunitarios para gestionar Proxmox vía Ansible.
  - Enlace: [https://docs.ansible.com/ansible/latest/collections/community/general/proxmox\\_kvm\\_module.html](https://docs.ansible.com/ansible/latest/collections/community/general/proxmox_kvm_module.html)

## E. Metodología y Control de Versiones

- **Git SCM:** Referencia de comandos y flujos de trabajo para el control de versiones.
  - Enlace: <https://git-scm.com/doc>
- **Markdown Guide:** Guía de sintaxis para la elaboración de la documentación técnica en el repositorio.
  - Enlace: <https://www.markdownguide.org/>