Aquí tienes el **stack tecnológico completo** para implementar SentinelCore, optimizado para escalabilidad, seguridad y el enfoque en contenedores que mencionaste:

**Stack Principal**

| **Capa** | **Tecnologías** | **Justificación** |
| --- | --- | --- |
| **Frontend** | Vue.js 3 (Composition API) + Tailwind CSS + Vite | Alta reactividad + diseño modular compatible con microfrontends |
| **Backend** | Python FastAPI + Uvicorn (ASGI) | Alto rendimiento para APIs + soporte nativo para async (escaneos largos) |
| **Toquens** | Contenedores Docker (Alpine Linux) + gVisor (sandboxing) | Aislamiento seguro y portabilidad |
| **Orquestación** | Kubernetes (K3s para entornos pequeños) + Helm | Gestión automatizada de toquens escalables |
| **Comunicación** | RabbitMQ (cola de mensajes) + WebSockets (notificaciones en tiempo real) | Coordinación entre módulos sin acoplamiento |
| **Bases de Datos** | PostgreSQL (transaccional) + TimescaleDB (métricas) | Escalabilidad vertical/horizontal + análisis temporal de vulnerabilidades |
| **Seguridad** | Vault (gestión secrets) + OpenPolicyAgent (control de acceso) | Cumplimiento de regulaciones y Zero Trust |
| **Monitorización** | Prometheus + Grafana Loki + Alertmanager | Observabilidad de toquens y detección de anomalías |

**Detalle por Módulo**

**1. Toquen de Antesala**

dockerfile

# Ejemplo Dockerfile para Toquen PDF

FROM python:3.11-alpine

RUN apk add --no-cache clamav clamav-libunrar

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt # PyMuPDF, python-magic

COPY pdf\_sanitizer.py .

HEALTHCHECK --interval=30s CMD pgrep -f "python pdf\_sanitizer.py" || exit 1

CMD ["python", "pdf\_sanitizer.py"]

**Comunicación**:

* **Entrada**: Cola RabbitMQ (scan\_requests)
* **Salida**: API REST (POST /results) con firmas JWT

**2. Integración WhatsApp**

python

# Ejemplo con Twilio API

from fastapi import APIRouter

from twilio.rest import Client

router = APIRouter()

client = Client(settings.TWILIO\_SID, settings.TWILIO\_TOKEN)

@router.post("/whatsapp/webhook")

async def handle\_message(from: str, body: str):

if "scan" in body:

await rabbit.publish("scan\_requests", {"target": body.split()[-1], "user": from})

return {"status": "queued"}

**3. Dashboard Web**

**Stack Específico**:

* **Mapa de Servicios**: D3.js + Sigma.js (visualización de redes)
* **Auth**: Ory Hydra (OIDC) + permisos basados en roles
* **Editor de Reglas**: Monaco Editor (VSCode en navegador)

**Infraestructura como Código**

terraform

# Ejemplo AWS EKS + VPC

module "vpc" {

source = "terraform-aws-modules/vpc/aws"

cidr = "10.0.0.0/16"

enable\_nat\_gateway = true

}

resource "aws\_eks\_cluster" "sentinel" {

vpc\_config {

endpoint\_private\_access = true

subnet\_ids = module.vpc.private\_subnets

}

}

**Pipeline CI/CD**

yaml

# GitHub Actions ejemplo

jobs:

deploy-toquens:

runs-on: ubuntu-latest

steps:

- uses: docker/build-push-action@v4

with:

push: **true**

tags: ghcr.io/sentinelcore/toquen-pdf:latest

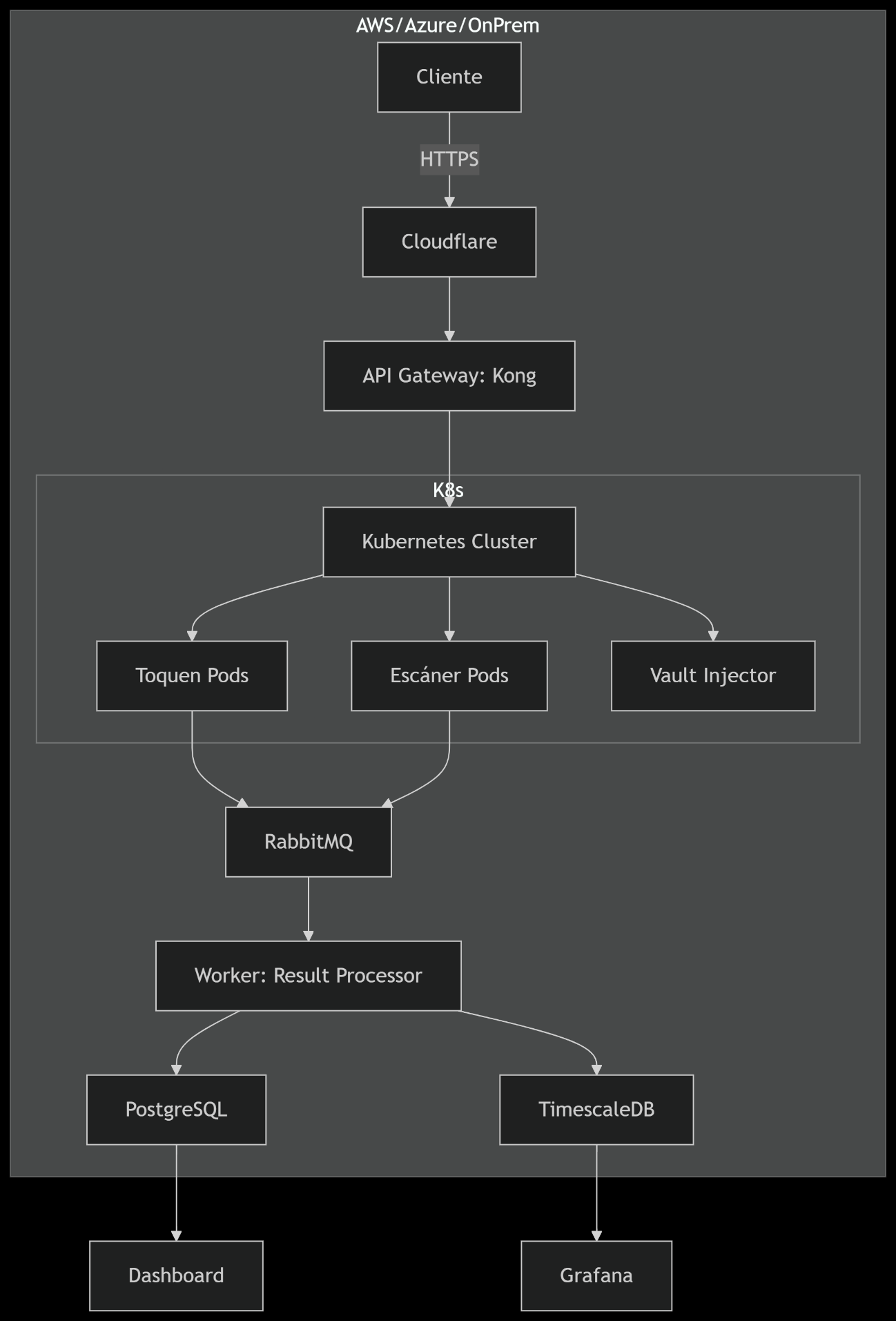
file: ./toquens/pdf/Dockerfile

- run: kubectl rollout restart deployment/toquen-pdf -n production

**Herramientas Adicionales Clave**

| **Propósito** | **Herramienta** | **Configuración Recomendada** |
| --- | --- | --- |
| Escaneo Web | OWASP ZAP en Docker (+API) | Helm Chart oficial con custom policies |
| Análisis Cloud | Prowler + Scout Suite | Ejecución en contenedores efímeros (AWS Fargate) |
| Gestión de Secretos | HashiCorp Vault | Inyección dinámica en runtime (no env vars) |
| Logs Estructurados | FluentBit + ELK | Filtrado por módulo/toquen |
| Pruebas de Seguridad | Trivy (escaneo imágenes) + Kube-bench | Integrado en pipeline CI |

**Diagrama de Arquitectura Completa**



**Checklist de Implementación**

1. Configurar VPC con subnets públicas/privadas
2. Desplegar EKS/GKE con auto-scaling (mínimo 3 nodos)
3. Implementar HashiCorp Vault para secrets
4. Crear Helm Charts para cada tipo de toquen
5. Configurar CI/CD con escaneos de seguridad integrados
6. Implementar backup automatizado para PostgreSQL