Diseño de Infraestructura

Índice

Diseño de infraestructura segura y escalable

Objetivo del diseño

Este apartado describe una arquitectura técnica preparada para un despliegue futuro en entornos productivos, con énfasis en seguridad, escalabilidad y mantenibilidad. El diseño está pensado para un proyecto académico con proyección realista de crecimiento, integrando buenas prácticas de infraestructura cloud, separación de responsabilidades y control de accesos.

La arquitectura propuesta es modular y evolutiva, permitiendo inicialmente un despliegue mixto (Render + AWS) y contemplando una futura consolidación en una única plataforma cloud.

1. Arquitectura general

La aplicación se compone de los siguientes bloques principales:

- Frontend web (cliente): interfaz de usuario que consume la API.
- Backend / API: lógica de negocio, control de acceso y orquestación.
- Chatbot con IA: interfaz conversacional integrada con el backend.
- Base de datos PostgreSQL: persistencia de datos de RRHH y marketing.
- Servicios de soporte: caché, colas, auditoría y observabilidad.

- Capa perimetral: proxy inverso y, en despliegues futuros, firewall cloud.

El diseño sigue un modelo de capas bien definidas, reduciendo acoplamiento y facilitando la escalabilidad horizontal.

1.1 Propuesta de despliegue de firewall en entorno cloud (Cloudflare)

1.1.1. Propuesta de despliegue de firewall en entorno cloud (Cloudflare)

De cara a una futura escalabilidad del proyecto y a un despliegue en un entorno productivo con dominio propio, se propone la incorporación de un Web Application Firewall (WAF) gestionado en la capa perimetral de la aplicación.

Dado que el proyecto está actualmente desplegado sin dominio registrado, esta medida no es aplicable en el estado actual, pero resulta viable y recomendable en una fase posterior.

1.1.2. Justificación del uso de Cloudflare

Cloudflare ofrece un servicio de firewall y protección perimetral que puede integrarse sin modificar la arquitectura interna de la aplicación.
La versión gratuita proporciona:

- Protección básica frente a ataques comunes (DDoS, escaneo, bots simples)

- Filtrado de tráfico HTTP/HTTPS

- Gestión centralizada desde la nube
- Escalabilidad automática

Esto lo convierte en una solución adecuada para un proyecto académico con proyección a crecimiento.

1.1.3. Arquitectura propuesta (visión futura)

1. El dominio público de la aplicación apunta a Cloudflare.
2. Cloudflare actúa como proxy inverso, recibiendo todo el tráfico entrante.
3. El tráfico legítimo es reenviado al backend desplegado en Render.
4. El firewall filtra y bloquea peticiones maliciosas antes de que alcancen la aplicación.

De este modo, la superficie de ataque se reduce sin añadir complejidad al backend.

1.1.4. Guía básica de despliegue (escenario con dominio registrado)

1. **Registro del dominio**
    - Adquirir un dominio público para la aplicación.
    - Acceder al panel de Cloudflare y crear una cuenta.

2. **Añadir el dominio a Cloudflare**

    - Introducir el dominio en Cloudflare.

    - Cloudflare detectará automáticamente los registros DNS existentes.

3. **Actualizar servidores DNS**

    - Sustituir los servidores DNS del proveedor del dominio por los proporcionados por Cloudflare.

    - A partir de este punto, todo el tráfico pasará por Cloudflare.

4. **Activación del proxy**

    - En los registros DNS, activar el modo proxy (icono de nube).

    - Esto habilita la inspección de tráfico y el firewall.

5. **Configuración básica del firewall**

    - Activar las reglas de seguridad por defecto.

    - Habilitar protección contra:

        - Peticiones sospechosas

        - Rutas comunes de ataque

        - Tráfico automatizado básico

6. **Configuración HTTPS**

    - Activar HTTPS automático gestionado por Cloudflare.

    - Forzar redirección HTTP → HTTPS.

1.1.5. Beneficios para el proyecto

- Añade una capa de seguridad perimetral sin modificar código

- Facilita el crecimiento del sistema sin rediseño

- Protege especialmente:

    - Endpoints del chatbot

    - APIs de acceso a datos

    - Superficie expuesta a Internet

1.1.6. Consideración final

Esta propuesta no sustituye a los controles de seguridad internos, pero permite desacoplar la protección externa del desarrollo de la aplicación, alineándose con buenas prácticas de diseño de infraestructura segura y escalable.

1.2 Capas y flujos de tráfico

El flujo de tráfico se estructura de la siguiente forma:

- Cliente (React/Vite): Servido como contenido estático desde CDN o detrás del proxy. Posibilidad de pre-render para mejorar rendimiento y SEO.
- Proxy inverso (Traefik / Nginx):
  - Terminación TLS/HTTPS obligatoria.
  - Enrutado por rutas: / (frontend), /api (backend), /metrics (observabilidad).
  - Controles de borde: CORS restrictivo, rate limiting, cabeceras de seguridad.
  - Posibilidad de múltiples réplicas para evitar puntos únicos de fallo.
- Backend / API: Expone endpoints controlados y encapsula el acceso a datos.
- Base de datos: Accesible únicamente desde el backend autorizado, aislada de acceso externo.
- Separación de entornos: Configuraciones diferenciadas para desarrollo, staging y producción.

1.3 Backend y API

- API Node/Express organizada por dominios funcionales.
- Versionado explícito de API (/api/v1).
- Posibilidad de incorporar un API Gateway interno para centralizar:
  - Autenticación y autorización.
  - Rate limiting.
  - Versionado y contratos.

Este diseño facilita la evolución futura hacia microservicios sin reescritura completa.

1.4 Base de datos (PostgreSQL)

- Usuarios diferenciados por función (aplicación, migraciones, auditoría).
- Migraciones versionadas con rollback controlado.
- Índices alineados con patrones de consulta.
- Backups periódicos y pruebas de restauración.
- Acceso restringido mediante reglas de red y credenciales específicas.

2. Seguridad

La seguridad se aplica de forma transversal:

- Autenticación: JWT con expiración corta y refresh tokens.
- Autorización (RBAC): roles diferenciados (admin, auditor, operador, viewer).
- Validación de entrada: esquemas estrictos y sanitización.
- Gestión de secretos: variables de entorno seguras y gestores externos en producción.
- Transporte seguro: HTTPS obligatorio y cabeceras de seguridad.
- Seguridad específica del chatbot:
    - Rate limiting por usuario e IP.
    - Límite de tamaño de prompt.
    - Timeouts y circuit breakers.
    - Logging controlado sin datos sensibles.

2.1 Seguridad avanzada

Se incorporan medidas de seguridad de nivel empresarial para reforzar la protección transversal:

- Zero Trust: verificación continua de identidades y accesos, segmentación de red y principio de mínimo privilegio.
- Threat modeling: identificación y priorización de amenazas en cada capa de la arquitectura.
- Seguridad en CI/CD: commits firmados, imágenes verificadas (cosign) y enforcement de políticas en pipelines.

3. Modularidad

3.1 Dominios funcionales

La aplicación se divide en dominios independientes:

- Autenticación
- Usuarios
- Gestión funcional (RRHH / marketing)
- Auditoría
- Salud y operaciones

Cada dominio puede evolucionar de forma independiente.

3.2 Capas internas

- Rutas
- Controladores
- Servicios
- Repositorios
- Esquemas de validación

- Middlewares

Esta separación mejora la mantenibilidad y la seguridad.

3.3 Modularidad y escalabilidad avanzada

Se refuerza la modularidad con patrones de arquitectura evolutiva:

- Diagrama de dominios funcionales con dependencias mínimas.
- Evolución hacia event-driven architecture (Kafka, SNS/SQS).
- Patrones de caching definidos (Redis, CDN) para mejorar rendimiento y escalabilidad.

4. Jerarquía operativa y de acceso

- RBAC centralizado con política de “deny by default”.
- Jerarquía clara de cambios:
  - Infraestructura
  - API
  - Datos
- Auditoría completa de acciones críticas.

4.1 Gobernanza y acceso

La gobernanza se formaliza con políticas claras de auditoría y seguridad de datos:

- Registro y revisión de eventos críticos con trazabilidad completa.
- Cifrado de datos sensibles en reposo (KMS) y en tránsito.
- Auditoría periódica de roles y accesos.

5. Datos y auditoría

- Registro de todas las acciones relevantes.
- Relaciones controladas mediante claves foráneas.
- Preparado para escalar a modelos de event sourcing si fuera necesario.

6. Observabilidad

- Logs estructurados.
- Métricas expuestas para monitorización.
- Dashboards y alertas.
- Tracing distribuido para análisis de incidencias.

6.1 Observabilidad y resiliencia

La observabilidad se amplía con métricas de calidad de servicio y pruebas de resiliencia:

- Definición de SLIs/SLOs/SLA (ejemplo: latencia <200ms, disponibilidad 99.9%).
- Chaos testing para validar tolerancia a fallos y recuperación automática.
- Alertas proactivas con umbrales definidos y escalado automático de incidentes.

7. CI/CD y entornos

- Pipelines automatizados con controles de calidad y seguridad.
- Escaneo de dependencias e imágenes.
- Despliegues controlados mediante versionado y feature flags.
- Rollback seguro en todos los niveles.

7.1 Infraestructura como código (IaC)

La infraestructura se gestiona de forma reproducible y auditable:

- Uso de Terraform/Pulumi para despliegues consistentes y trazables.
- Preparación para multi-cloud readiness, manteniendo portabilidad del diseño.
- Gestión de secretos con rotación automática y auditoría de accesos.

8. Plan de ampliación y escalación

- Escalado horizontal y vertical.
- Uso de caché y colas asíncronas.
- Separación progresiva en microservicios.
- Mejora de resiliencia y alta disponibilidad.

8.1 Plan de crecimiento y métricas de éxito

Se establece un roadmap de migración y criterios de validación:

Roadmap de despliegue

- Fase 1 – Entorno on-premise controlado
    - Despliegue limitado a intranet corporativa.
    - Acceso restringido por roles.
    - Monitorización básica de uso y errores.

- Fase 2 – Entorno híbrido (intranet + cloud)
    - Integración progresiva con servicios cloud.
    - Uso de AWS WAF para protección frente a:
        - Inyección de prompts
        - Abuso de peticiones
        - Tráfico anómalo
    - Comunicación segura mediante APIs autenticadas.
- Fase 3 – Consolidación en cloud
    - Despliegue completo en entorno cloud securizado.
    - Segmentación de servicios y datos sensibles.
    - Escalado automático controlado.

Métricas de éxito

- Reducción de latencia en respuestas bajo carga.
- Porcentaje de tests de seguridad superados (SAST/DAST).
- Porcentaje de dependencias auditadas y actualizadas.
- Número de accesos bloqueados por WAF.
- Incidentes detectados y correlacionados por el SIEM.

Monitorización y detección

- Integración de logs de:
    - Chatbot
    - APIs internas
    - AWS WAF

- Centralización en un SIEM corporativo para:
    - Detección de comportamientos anómalos
    - Auditoría de accesos
    - Respuesta temprana ante incidentes

9. Estrategia de despliegue: entorno mixto y consolidación futura

Actualmente, el proyecto contempla un despliegue mixto, combinando:

- Render: para frontend y backend en fases iniciales.
- AWS: para servicios gestionados (bases de datos, almacenamiento, secretos).

De cara a un despliegue futuro más maduro, se propone la consolidación de toda la aplicación en una única plataforma cloud, como AWS, lo que permitiría:

- Unificación de red y seguridad.
- Uso de servicios gestionados (RDS, ECS/EKS, ALB, Secrets Manager).
- Mejora de observabilidad centralizada.
- Reducción de latencias y complejidad operativa.

Esta transición puede realizarse de forma progresiva, manteniendo el diseño actual compatible con ambos escenarios.

10. Despliegue de referencia (Docker Compose)

Se incluye un despliegue de referencia local/autoalojado basado en Docker Compose que replica la arquitectura propuesta y facilita:

- Pruebas de escalabilidad.
- Validación de flujos.
- Preparación para despliegues en entornos cloud gestionados.

Este despliegue sirve como base técnica para migraciones futuras a orquestadores más avanzados.

11. Conclusiones

El diseño presentado integra buenas prácticas de seguridad, modularidad y escalabilidad, con una visión evolutiva hacia entornos productivos en cloud. La arquitectura propuesta permite un despliegue inicial mixto y contempla una consolidación futura en una plataforma única, garantizando mantenibilidad y resiliencia.

Las medidas de seguridad avanzada, observabilidad y gobernanza refuerzan la confianza en el sistema, mientras que la modularidad y el plan de crecimiento aseguran su capacidad de adaptación.

Este documento constituye una base sólida para la transición hacia infraestructuras empresariales, manteniendo la claridad, la reproducibilidad y la alineación con objetivos académicos y profesionales.