Planning Individual - Desarrollador B (Infraestructura y Backend)

👤 Perfil y Responsabilidades Principales

Rol: Infrastructure & Backend Engineer
Especialización: Arquitectura de sistemas, performance, confiabilidad y escalabilidad
Áreas de Dominio:

- Infraestructura como código (IaC)
- Sistemas distribuidos y alta disponibilidad
- Optimización de bases de datos
- Procesamiento asíncrono y colas
- Monitoring y observabilidad
- Seguridad y compliance

🎯 Objetivos Clave del Rol

1. **Garantizar 99.9% uptime** del sistema
2. **Optimizar costos** de infraestructura (-30% mediante caché y optimización)
3. **Reducir latencia** de procesamiento de archivos a <30s
4. **Automatizar** todo el deployment y scaling
5. **Asegurar** recuperación ante desastres <5 minutos

🗓️ Planning Detallado por Fases

FASE 0: Análisis y Preparación (Semana 0)

Día	Tareas	Entregables	Criterios de Éxito
Lunes	• Análisis de proveedores cloud (AWS vs GCP vs Hetzner) • Estimación de costos por escenarios • Definir arquitectura de red	• Comparison matrix de providers • Calculadora de costos	• Decisión fundamentada de hosting • Costos estimados ±20%
Martes	• Diseñar arquitectura de microservicios • Definir estrategia de datos • Planificar seguridad y compliance	• Diagrama de arquitectura • Documento de seguridad	• Arquitectura sin SPOF • Compliance GDPR definido
Miércoles	• Setup entorno local de desarrollo • Instalar herramientas necesarias • Configurar accesos y permisos	• README de setup • Scripts de instalación	• Entorno funcional • Documentación clara
Jueves	• Definir estrategia de CI/CD • Diseñar pipeline de deployment • Planificar estrategia de branching	• Diagrama de CI/CD • Git workflow guide	• Pipeline diseñado e2e • Estrategia clara
Viernes	• Workshop con Dev A sobre interfaces • Definir contratos entre servicios • Acordar estándares de código	• API contracts • Coding standards doc	• Interfaces acordadas • Standards definidos

FASE 1: Infraestructura Base (Semanas 1-2)

Semana 1: Fundación de Servicios

Tarea	Descripción Detallada	Dependencias	Riesgos
1.1 Docker Compose Completo	• Configurar todos los servicios base • Health checks para cada servicio • Volumes para persistencia • Networks para aislamiento	Ninguna	Compatibilidad entre versiones
1.2 PostgreSQL HA	• Master con streaming replication • Configurar WAL archiving • Automatic failover con repmgr • Connection pooling con PgBouncer	Docker setup	Pérdida de datos en failover
1.3 Redis Cluster	• Redis Sentinel para HA • Configurar persistencia AOF+RDB • Separar cache de sesiones • Políticas de eviction	Docker setup	Split brain en cluster
1.4 Message Queue	• RabbitMQ con management UI • Definir exchanges y queues • Configurar DLQ y retry policies • Clustering para HA	Docker setup	Pérdida de mensajes
1.5 Object Storage	• MinIO con buckets separados • Políticas de lifecycle • Replicación entre zonas • Presigned URLs para uploads	Docker setup	Límites de almacenamiento

Semana 2: Monitoring y Seguridad

Tarea	Descripción Detallada	Dependencias	Riesgos
2.1 Stack de Monitoring	• Prometheus + Grafana • Exporters para cada servicio • Dashboards personalizados • Alertmanager configurado	Servicios base	Overhead de métricas
2.2 Logging Centralizado	• ELK Stack o Loki • Log shipping desde containers • Parsing y enrichment • Retention policies	Servicios base	Costo de almacenamiento
2.3 Tracing Distribuido	• Jaeger para tracing • Instrumentación OpenTelemetry • Sampling strategies • Trace visualization	Servicios base	Performance overhead
2.4 Seguridad Base	• Network policies • Secrets management (Vault) • TLS entre servicios • RBAC implementation	Todos los servicios	Complejidad de gestión
2.5 CI/CD Pipeline	• GitHub Actions/GitLab CI • Build y test automation • Security scanning • Artifact registry	Git setup	Tiempo de build

FASE 2: Servicios de Procesamiento (Semanas 3-4)

Semana 3: OCR y Procesamiento de Archivos

Tarea	Descripción Detallada	Skills Necesarios	Métricas de Éxito
3.1 Servicio OCR	• Tesseract en containers • Queue consumers escalables • Preprocessing de imágenes • Multi-idioma support	Docker, Python, Tesseract	95% accuracy rate
3.2 Pipeline de Procesamiento	• Workflow con estados • Reintentos inteligentes • Progress tracking • Error handling robusto	RabbitMQ, Python	<30s por documento
3.3 Optimización de Recursos	• GPU support para OCR • Batch processing • Memory optimization • Concurrent processing	Performance tuning	10x throughput
3.4 Storage Optimization	• Compresión de archivos • Deduplicación • Tiered storage • CDN integration	S3, Compression	50% reducción storage

Semana 4: API Gateway y Rate Limiting

Tarea	Descripción Detallada	Skills Necesarios	Métricas de Éxito
4.1 API Gateway	• Kong/Envoy setup • Route configuration • Request/Response transformation • API versioning	API Gateway, Lua	<10ms overhead
4.2 Rate Limiting Distribuido	• Token bucket implementation • Redis-based limiting • Per-user/Per-IP limits • Graceful degradation	Redis, Algorithms	0 false positives
4.3 Circuit Breakers	• Hystrix patterns • Fallback strategies • Recovery policies • Monitoring integration	Resilience patterns	<1s detection time
4.4 Load Balancing	• HAProxy/Nginx config • Health checks • Session affinity • A/B routing	Load balancing	Equal distribution

FASE 3: Optimización y Performance (Semanas 5-6)

Semana 5: Optimización de Base de Datos

Tarea	Descripción Detallada	Herramientas	Objetivo
5.1 Query Optimization	• EXPLAIN ANALYZE todas las queries • Índices optimizados • Query rewriting • Prepared statements	pg_stat_statements	<50ms P95
5.2 Particionamiento	• Particionamiento por fecha • Archivado automático • Vacuum strategies • Statistics update	PostgreSQL 15	Linear scaling
5.3 Read Replicas	• Load balancing de lecturas • Lag monitoring • Automatic failover • Connection pooling	PgBouncer, HAProxy	<100ms lag
5.4 Database Caching	• Query result caching • Materialized views • Redis integration • Cache invalidation	Redis, PostgreSQL	90% hit rate

Semana 6: Caché Multi-nivel

Tarea	Descripción Detallada	Herramientas	Objetivo
6.1 CDN Setup	• CloudFlare/Fastly • Cache rules • Purge strategies • Origin shield	CDN Provider	Global <100ms
6.2 Application Cache	• In-memory caching • Distributed cache • Cache warming • TTL strategies	Redis, Memcached	95% hit rate
6.3 Database Cache	• Query cache tuning • Buffer pool sizing • Working set analysis • Cache metrics	PostgreSQL, Redis	Reduced DB load 70%
6.4 Cache Invalidation	• Event-driven invalidation • Cache tags • Cascading updates • Consistency guarantees	Redis, RabbitMQ	<1s propagation

FASE 4: Kubernetes y Producción (Semanas 7-8)

Semana 7: Migración a Kubernetes

Tarea	Descripción Detallada	Complejidad	Prioridad
7.1 K8s Cluster Setup	• EKS/GKE/k3s decisión • Multi-zone setup • Network policies • Storage classes	Alta	P0
7.2 Helm Charts	• Crear charts para cada servicio • Values para environments • Dependencies management • Secrets handling	Media	P0
7.3 Auto-scaling	• HPA configuration • VPA setup • Cluster autoscaler • Custom metrics	Alta	P0
7.4 Service Mesh	• Istio/Linkerd evaluation • mTLS entre servicios • Traffic management • Observability	Alta	P1
7.5 GitOps	• ArgoCD/Flux setup • Repository structure • Sync policies • Rollback strategies	Media	P1

Semana 8: Confiabilidad y DR

Tarea	Descripción Detallada	Complejidad	Prioridad
8.1 Backup Strategy	• Automated DB backups • File storage backups • Configuration backups • Test restores	Media	P0
8.2 Disaster Recovery	• Runbooks creation • Failover procedures • Data recovery plans • Communication plans	Alta	P0
8.3 Chaos Engineering	• Chaos Monkey setup • Failure scenarios • Game days • Improvement tracking	Media	P1
8.4 Security Hardening	• Vulnerability scanning • Penetration testing • Security policies • Compliance checks	Alta	P0

FASE 5: Features Avanzadas (Semanas 9-10)

Semana 9: Real-time y Streaming

Tarea	Descripción Detallada	Stack Técnico	Casos de Uso
9.1 WebSocket Gateway	• Socket.io/native WS • Scaling con Redis Pub/Sub • Connection management • Heartbeat/reconnection	Node.js, Redis	Chat, notificaciones
9.2 Event Streaming	• Kafka/Redis Streams • Event sourcing • CQRS patterns • Event replay	Kafka, Redis	Audit, analytics
9.3 Real-time Analytics	• Streaming aggregations • Time-series data • Real-time dashboards • Alerting rules	InfluxDB, Grafana	Métricas live

Semana 10: Optimización Final

Tarea	Descripción Detallada	Stack Técnico	Impacto
10.1 Performance Profiling	• APM integration • Bottleneck analysis • Memory profiling • CPU optimization	DataDog, New Relic	20% mejora
10.2 Cost Optimization	• Reserved instances • Spot instances • Right-sizing • Usage analysis	Cloud tools	30% reducción
10.3 Documentation	• Architecture docs • Runbooks • Troubleshooting guides • API documentation	Confluence, Swagger	100% coverage
10.4 Knowledge Transfer	• Team training • Video tutorials • Best practices • Handover sessions	-	Team ready

📊 KPIs y Métricas de Éxito

Métricas de Infraestructura

- **Disponibilidad**: 99.9% (43 min downtime/mes)
- **Latencia P95**: <200ms API, <30s procesamiento
- **Error Rate**: <0.1%
- **Costo por usuario**: <$0.10/mes

Métricas de Desarrollo

- **Deployment frequency**: 5+ por día
- **Lead time**: <2 horas
- **MTTR**: <30 minutos
- **Change failure rate**: <5%

Métricas de Calidad

- **Test coverage**: >80%
- **Technical debt ratio**: <5%
- **Security vulnerabilities**: 0 críticas
- **Documentation coverage**: 100%

🛠️ Stack Tecnológico Principal

Infrastructure

- **Containers**: Docker, Kubernetes
- **IaC**: Terraform, Helm
- **CI/CD**: GitHub Actions, ArgoCD

Data Layer

- **Primary DB**: PostgreSQL 15
- **Cache**: Redis 7
- **Object Storage**: MinIO/S3
- **Search**: Elasticsearch

Monitoring

- **Metrics**: Prometheus + Grafana
- **Logs**: ELK/Loki
- **Traces**: Jaeger
- **APM**: DataDog/New Relic

Languages & Frameworks

- **Primary**: Python (FastAPI)
- **Performance Critical**: Go
- **Scripts**: Bash, Python
- **Config**: YAML, HCL

🚨 Riesgos y Mitigaciones

Riesgo	Probabilidad	Impacto	Mitigación
Complejidad excesiva	Alta	Alto	Adopción incremental, documentación exhaustiva
Costos cloud elevados	Media	Alto	Monitoring de costos, alertas, optimización continua
Deuda técnica	Media	Medio	20% tiempo para refactoring, code reviews
Falta de expertise K8s	Media	Alto	Training, consultoría externa, start simple
Downtime en migración	Baja	Alto	Blue-green deployment, rollback plan

📈 Plan de Crecimiento Personal

Skills a Desarrollar

1. **Kubernetes**: CKA certification
2. **Cloud Architecture**: AWS/GCP certification
3. **SRE Practices**: Google SRE books
4. **Performance**: Systems Performance book

Mentoría y Colaboración

- Pair programming con Dev A semanalmente
- Code reviews cruzados
- Sesiones de arquitectura quincenales
- Documentación como primera prioridad

✅ Definition of Done

Para cada tarea:

- [ ] Código en producción
- [ ] Tests automatizados (unit + integration)
- [ ] Documentación actualizada
- [ ] Monitoring configurado
- [ ] Runbook para troubleshooting
- [ ] Security review pasado
- [ ] Performance benchmarked

- [ ] Knowledge transferred