Aitamit

Manual de Administrador

Despliegue y operación de la plataforma

European Digital Stores SL

Agosto de 2026 · aitamit.com

Índice

1. Alcance de este manual

Este manual está dirigido al **operador de la plataforma AITAMIT**: quien la despliega, la mantiene, la actualiza y da soporte a los tenants. No cubre el uso funcional de la consola (para eso existe el Manual de Usuario), sino todo lo que ocurre por debajo: infraestructura, identidad, facturación, agentes, copias de seguridad, seguridad y operación diaria.

La documentación técnica de referencia vive en el repositorio (`docs/`): ARQUITECTURA.md, API.md, DESPLIEGUE.md, SEGURIDAD.md, AGENTES.md y ONPREM.md. Este manual los complementa con el detalle operativo.

2. Arquitectura y stack

AITAMIT es un monorepo con cuatro aplicaciones y una infraestructura declarada en Docker Compose:

Servicio	Tecnología	Función
landing	Next.js 14	Web pública (12 idiomas), signup, installers, manuales
console	Next.js 14 + OIDC	Consola operativa de los tenants
backend-api	Go 1.23 + pgx/v5	API REST de control (~50 endpoints), migraciones embebidas
backend-ingest	Go 1.23	Ingesta de agentes (registro, inventario, comandos)
keycloak	Keycloak 25	Identidad OIDC, realm aitamit
postgres	Postgres 16	Datos (esquemas core y aitamit) + BD keycloak
redis	Redis 7	Caché / rate limiting
nats	NATS 2	Mensajería interna

Delante de todo, **Traefik** termina TLS (HTTP-01 y wildcard DNS-01), aplica cabeceras de seguridad, redirección 308 a HTTPS y rate limiting perimetral. Los dominios públicos son aitamit.com, console.aitamit.com, api.aitamit.com, ingest.aitamit.com, auth.aitamit.com y el comodín {slug}.aitamit.com.

3. Requisitos

- **Servidor**: Linux (Ubuntu 22.04+), 4 vCPU / 8 GB RAM / 80 GB SSD como punto de partida razonable; Docker y Docker Compose.
- **Traefik v2** en la red externa `traefik_web` con un resolver ACME operativo.
- **Dominio** aitamit.com con API DNS del registrador (imprescindible para el certificado wildcard DNS-01).
- **Stripe** (cuenta live de European Digital Stores SL) con claves y webhook.
- **SMTP** transaccional propio (mailcow de EDS) con el buzón noreply@aitamit.com.

4. Despliegue paso a paso

4.1 Preparación

```
git clone https://github.com/faseds/aitamit.git /opt/aitamit
cd /opt/aitamit/infra/docker
cp .env.example .env && chmod 600 .env   # rellenar TODOS los valores
```

4.2 Variables del .env

Variable	Contenido
POSTGRES_PASSWORD	Contraseña de Postgres. El init crea las BD aitamit y keycloak.
KEYCLOAK_ADMIN_USER / _PASSWORD	Credenciales del admin master de Keycloak.
KEYCLOAK_BASE_URL	URL INTERNA http://aitamit-keycloak:8080 (nunca la pública: el IPAllowList del admin la bloquearía).
KEYCLOAK_REALM	aitamit
STRIPE_API_KEY / STRIPE_WEBHOOK_SECRET	Clave live y signing secret del webhook.
ALLOW_ANON	false SIEMPRE en producción (la ingesta exige token).

4.3 Arranque

```
docker compose up -d
docker ps --filter name=aitamit- --format "table {{.Names}}\t{{.Status}}"
```

Los ocho contenedores declaran healthcheck; espere a que todos estén healthy. Las migraciones SQL (001–011) se aplican solas al arrancar backend-api.

> **Nota:** Los healthchecks usan 127.0.0.1, no localhost: las imágenes Alpine no siempre resuelven localhost.

4.4 DNS y TLS

1. Cree los registros A: @, www, api, ingest, auth, console y el comodín * apuntando al VPS.
2. Configure en Traefik el resolver DNS-01 con la API key del registrador (fichero .env de Traefik).
3. El router de la consola usa HostRegexp con tls.domains[0].main=aitamit.com y sans=*.aitamit.com.
4. Verifique: https://aitamit.com, https://console.aitamit.com y https://cualquiercosa.aitamit.com deben servir con certificado válido.

5. Keycloak

5.1 Realm y clients

1. Cree el realm **aitamit** con bruteForceProtected=true (5 fallos, espera incremental hasta 15 min) y verifyEmail=true.
2. Client **aitamit-console**: public, PKCE, redirect URIs https://console.aitamit.com/* y https://*.aitamit.com/*.
3. Client **aitamit-api**: bearer-only.
4. Client scope **tenant** con mapper oidc-usermodel-attribute-mapper que emite el atributo tenant_id como claim en id/access/userinfo. Asígnelo como default scope a ambos clients.
5. Roles realm: admin, manager, worker, employee, tenant-admin, platform-admin.

> **Importante:** Sin el mapper de tenant_id la plataforma es inoperante: todo token sin ese claim recibe 403. Es la primera comprobación ante un «no funciona nada tras reinstalar Keycloak».

5.2 SMTP del realm

Configure el email del realm contra el mailcow de EDS: puerto **465 con SSL implícito** (587/STARTTLS suele dar timeout en mailcow), remitente noreply@aitamit.com. Pruebe con el botón de test y después con un signup real: el VERIFY_EMAIL debe llegar en el idioma del usuario.

5.3 Acceso al admin de Keycloak

La ruta /admin/ está protegida por IPAllowList en Traefik (localhost + IP del propio VPS). Para administrar desde fuera:

```
ssh -L 9999:localhost:8080 root@<vps>
# navegador: http://localhost:9999/admin
```

6. Stripe y facturación

6.1 Principio: Stripe solo cobra

La fuente de verdad del catálogo comercial es la tabla **core.plans**, administrable desde la consola (rol platform-admin, sección Planes). Al guardar un plan con requires_stripe=true, el backend crea automáticamente el Product/Price en Stripe si no existe o si el precio cambió. Los Prices de Stripe son inmutables: las suscripciones existentes conservan su precio antiguo, que es el comportamiento correcto.

6.2 Configuración inicial

1. Cree el webhook en Stripe apuntando a https://api.aitamit.com/api/v1/billing/webhook con los eventos customer.subscription.created/updated/deleted, invoice.payment_succeeded e invoice.payment_failed.
2. Copie el signing secret al .env (STRIPE_WEBHOOK_SECRET) y reinicie backend-api.
3. Defina el catálogo en Consola → Planes: precio, mínimo de agentes, cuota de alta, días de trial, visibilidad y si requiere Stripe.
4. Verifique el ciclo completo con un signup de prueba de un plan de pago: tenant trialing + customer y subscription en Stripe + evento registrado en core.billing_events.

> **Nota:** Los eventos del webhook son idempotentes por stripe_event_id: reintentos de Stripe no duplican efectos.

6.3 Asignar platform-admin

El rol realm **platform-admin** habilita la administración del catálogo. Asígnelo únicamente a los operadores de la plataforma; ningún usuario de tenant debe tenerlo.

7. Gestión de tenants

- **Alta**: normalmente self-service desde la landing. El signup crea tenant + usuario Keycloak (con atributo tenant_id) + install token y devuelve la URL de bienvenida en {slug}.aitamit.com.
- **Alta manual**: inserte el tenant en core.tenants y cree el usuario en Keycloak con el atributo tenant_id; útil para migraciones o ventas asistidas.
- **Suspensión**: un tenant con status disabled deja de poder ingestar y sus install tokens quedan inservibles.
- **Límites**: asset_limit se fija en el alta desde el plan elegido. Cambiar después el límite del plan NO afecta a los tenants existentes (conservan el suyo, por diseño).
- **Slugs reservados**: core.reserved_slugs impide registrar admin, api, www, etc.

8. Publicación de agentes

La matriz completa de plataformas y las recetas de compilación cruzada están en **docs/AGENTES.md**. Operativamente:

1. Compile el target (Rust; Android con Gradle) y genere el .sha256 de cada binario.
2. Copie binario + checksum a apps/landing/public/agent/ con la nomenclatura aitamit-agent-<os>-<arch>.
3. Rebuild + recreate del contenedor landing (los public/* van embebidos en la imagen standalone). En caliente vale `docker cp` a aitamit-landing:/app/public/agent/ + `docker restart aitamit-landing`.
4. Verifique la descarga y el checksum desde https://aitamit.com/agent/ y una instalación real de cada installer.

8.1 Agentes de escritorio con control remoto

El control remoto interactivo (capítulo 9) requiere el agente compilado con la feature `remote` (arrastra la captura de pantalla y la inyección de input). Se distribuye como binario aparte del agente de inventario base: `aitamit-agent-{linux-x64,windows-x64,macos-arm64,macos-x64}-remote(.exe)`. Recetas de compilación cruzada (mingw para Windows, cargo-zigbuild + SDK para macOS) en docs/CONTROL-REMOTO.md.

8.2 Móviles

- **Android**: publicado en Google Play (cuenta EDS) y como APK en aitamit.com/agent. Para flota grande, Android Enterprise / Managed Google Play.
- **iOS/iPadOS**: no hay agente que publicar — Apple lo prohíbe. Se gestionan por MDM nativo de AITAMIT (capítulo 9); no hay binario que compilar ni firmar.

> **Importante:** Estado de firma: **Android ya firmado** (keystore de EDS en CI) y publicado en Google Play. **Pendientes**: code-signing Windows (SmartScreen) y Developer ID + notarización macOS (Gatekeeper). Sin ellos los instaladores de escritorio funcionan pero el SO muestra advertencias.

9. Móviles (MDM) y control remoto

AITAMIT es su propio **servidor MDM de Apple** (endpoints públicos `/mdm/enroll`, `/mdm/checkin`, `/mdm/connect`) y ofrece **control remoto de escritorio** mediante un relay WebSocket. Detalle técnico en docs/AGENTES.md y docs/CONTROL-REMOTO.md.

9.1 iOS: enrolamiento y gestión

- Enrolamiento manual: la consola (Móviles → Enrolar) genera la URL/QR del perfil; el usuario lo instala por Safari. Inventario y gestión (bloquear, borrar, apps, perfiles, restricciones) inmediatos.
- **Push y zero-touch a escala** (opcional): requiere, por tenant, un **certificado push MDM** (Apple Push Certificates Portal, identity.apple.com) y un **token de servidor de Apple Business Manager**. Ambos se obtienen con el Apple ID corporativo del cliente (login + 2FA; ningún software lo hace headless) y se cargan una vez en Móviles → Apple Business Manager. Sin ellos, inventario y comandos se aplican en el siguiente check-in del dispositivo en lugar de al instante.

9.2 Conectores de solo lectura (Jamf / Intune)

Para clientes que ya gestionan iOS con otro MDM (Apple no admite dos perfiles por dispositivo). Se configuran en Móviles → Conectar Jamf/Intune:

- **Jamf Pro**: crear un API Role & Client de solo lectura (Settings → API Roles and Clients); introducir URL de la instancia, client_id y client_secret.
- **Microsoft Intune**: registrar una app en Entra ID con permiso de aplicación `DeviceManagementManagedDevices.Read.All`; introducir directory (tenant) id, application (client) id y secret.
- La sincronización vuelca la flota iOS a inventario (type=mobile). Puede lanzarse bajo demanda («Sincronizar ahora») o dejarse periódica.

9.3 Control remoto de escritorio

El relay del backend (`/rmm/remote/agent` y `/rmm/remote/operator`, WSS) empareja agente y operador por sesión. El agente inicia la conexión saliente (atraviesa NAT); el operador se autentica con un ticket de un solo uso; toda sesión queda auditada (remote_session.start/.end). No requiere configuración adicional de infraestructura salvo desplegar el agente con feature `remote` en los equipos (capítulo 8.1). Verificación E2E documentada en docs/CONTROL-REMOTO.md.

10. Operación diaria

10.1 Salud

```
curl -s https://api.aitamit.com/healthz && curl -s https://ingest.aitamit.com/healthz
docker ps --filter name=aitamit- --format "table {{.Names}}\t{{.Status}}"
```

10.2 Métricas

backend-api y backend-ingest exponen **/metrics** en formato Prometheus. Conéctelos a su Prometheus/Grafana y vigile como mínimo: tasa de errores 5xx, latencia de ingesta, agentes activos y colas de comandos.

10.3 Logs

```
docker logs -f aitamit-backend-api
docker logs -f aitamit-backend-ingest | grep -i error
```

El daemon de Docker debe tener rotación configurada (50 MB × 5 ficheros) para que los logs no llenen el disco.

11. Copias de seguridad y recuperación

> **Importante:** No poner en producción con tráfico real sin copias. Es el único bloqueante absoluto de la lista de pendientes.

11.1 Patrón recomendado (estándar EDS)

1. **Espejo caliente**: VPS espejo con snapshot de datos cada 15 minutos (rsync o ZFS send). Objetivo: RPO 15 min.
2. **Backup nocturno**: pg_dump completo + volúmenes Docker a NAS/offsite con retención de 30 días.
3. **Vigilancia**: ambos procesos monitorizados por el panel de infraestructura de EDS con alerta si no se ejecutan.

11.2 Qué respaldar

- Postgres completo (BD aitamit y BD keycloak — la segunda contiene usuarios e identidad).
- El fichero .env (secretos) y la configuración de Traefik (certificados ACME).
- apps/landing/public/agent/ y manual/ si se publican binarios/documentos no regenerables desde el repo.

11.3 Restauración

1. Provisionar VPS con Docker + Traefik y clonar el repo.
2. Restaurar .env y volúmenes/BD desde la última copia.
3. docker compose up -d y verificación E2E completa (checklist del capítulo 15).
4. Cambiar los DNS al nuevo servidor (TTL bajo si es una migración planificada).

12. Seguridad de la plataforma

El modelo completo está en docs/SEGURIDAD.md. Resumen operativo de lo que debe permanecer siempre activo:

Capa	Control
Host	UFW solo 22/80/443 · fail2ban sshd · rotación de logs
Perímetro	HSTS 2 años + preload · X-Frame DENY · CSP · rate limit 30 req/s burst 100 · redirección 308
Identidad	Brute force protection en el realm · admin de Keycloak tras IPAllowList · errores JWT genéricos
Aplicación	CORS restringido con Vary: Origin · signup 5/h por IP · validación de entrada estricta
Ingesta	Token obligatorio (ALLOW_ANON=false) · 2 MB máx · 60 req/min por IP
Datos	Postgres/Redis/NATS solo en la red interna · .env chmod 600 · BYOK por tenant disponible
Integridad	Auditoría hash-chain (verificable vía API) · informes firmados SHA-256 · webhooks HMAC

13. On-prem y air-gapped

- **docker-compose.onprem.yml** (infra/onprem/): despliegue autoalojado en la infraestructura del cliente, con su propio .env.onprem.
- **Helm chart** (infra/helm/aitamit/): despliegue en Kubernetes.
- **Bundle air-gapped** (airgapped-bundle.sh): empaqueta imágenes y artefactos para instalar sin salida a Internet.
- El detalle de cada modalidad está en docs/ONPREM.md.

14. Actualizaciones y rollback

1. Pull del repo y revisión del changelog de migraciones (nuevos ficheros en cmd/api/migrations/).
2. Backup previo (o snapshot del espejo verificado).
3. docker compose build --no-cache <servicios cambiados> && docker compose up -d --force-recreate <servicios>.
4. Verificación E2E (capítulo 15). Ante regresión: volver a la imagen anterior (docker compose up -d con el tag previo). Las migraciones son forward-only: restaurar BD desde backup si una migración debe deshacerse.

15. Runbooks de incidentes

Incidente	Diagnóstico y acción
La consola no carga / bucle de login	Keycloak caído o mapper tenant_id ausente. docker logs aitamit-keycloak; verificar client scope tenant.
Todo devuelve 403	Tokens sin claim tenant_id: revisar el default scope de los clients tras cambios en Keycloak.
Los agentes no reportan	ingest caído o ALLOW_ANON/token mal: healthz de ingest, logs, y probar un register con token válido.
Signup falla con 500	Stripe (clave/price) o Keycloak admin API: logs de backend-api; verificar credenciales del .env.
No llegan emails	SMTP del realm: probar test de Keycloak; verificar 465/SSL y credenciales del buzón en el mailcow.
Certificado wildcard no renueva	API key del registrador caducada o rate limit ACME: logs de Traefik; regenerar API key.
Webhook Stripe en rojo	Firma incorrecta (secret rotado): actualizar STRIPE_WEBHOOK_SECRET y reiniciar backend-api.
Disco lleno	Logs sin rotar o imágenes huérfanas: docker system prune -a --volumes tras verificar backups.

16. Checklist de verificación E2E

Ejecute esta lista tras cada despliegue, actualización o restauración. «Hecho» significa **desplegado y verificado**; ningún punto se da por bueno sin comprobarlo de verdad:

1. Landing en https://aitamit.com correcta en varios idiomas.
2. Signup completo: tenant creado, usuario en Keycloak con tenant_id, install token emitido, email de verificación recibido.
3. {slug}.aitamit.com sirve la consola con TLS válido.
4. Instalación real de un agente Linux con el token: activo visible con inventario completo en < 15 min.
5. Comando remoto de prueba entregado y completado con salida.
6. Plan de pago: suscripción trialing en Stripe y evento en billing_events.
7. /api/v1/audit/verify devuelve la cadena íntegra.
8. healthz/readyz de api e ingest en 200; /metrics sirviendo.
9. Backup de la noche anterior presente y restaurable (prueba de restauración mensual).

17. Modelo de soporte

Nivel	Quién	Qué resuelve
L1	Soporte funcional	Uso de la consola, altas de usuarios, dudas de módulos. Herramientas: Manual de Usuario.
L2	Operador de plataforma	Incidencias de tenants, agentes que no reportan, emails, facturación. Herramientas: este manual, runbooks del cap. 15.
L3	Ingeniería	Bugs del producto, migraciones, rendimiento, seguridad. Herramientas: repositorio y docs/ técnicas.

Recomendación operativa: todo incidente L2 resuelto que no esté en los runbooks del capítulo 14 debe añadirse a ellos. El manual es un documento vivo.

18. Pendientes conocidos de plataforma

Estado a la fecha de este manual. Lo ya resuelto se marca como referencia.

Pendiente	Impacto	Prioridad
Code-signing Windows	SmartScreen advierte al instalar el agente de escritorio.	Alta
Notarización macOS (Developer ID de EDS)	Gatekeeper exige autorización manual.	Alta
Cert push MDM + token ABM por cliente (iOS)	Sin ellos, comandos/inventario iOS se aplican en el siguiente check-in en vez de al instante (push). Son piezas de Apple: login+2FA del cliente, no automatizable.	Media
Control remoto en Android	Ver/tocar la pantalla del móvil (MediaProjection + AccessibilityService). El de escritorio ya está.	Media
Verificar agente remoto macOS en Mac real	Compila para arm64/x64; falta ejecutarlo con permisos TCC en hardware.	Media
Rate limiter en Redis	El límite in-memory se resetea al reiniciar.	Media
CI de builds de escritorio	Android ya en CI; los binarios de escritorio se compilan a mano (cross en el VPS).	Baja

Nota: Ya resuelto (referencia): copias de seguridad espejo+nocturno; keystore Android y publicación en Google Play; **MDM iOS nativo con gestión + conectores Jamf/Intune**; **control remoto de escritorio (Windows/macOS/Linux)**; cobros recurrentes con fin de trial y portal de facturación.

19. Glosario técnico

Término	Definición
ACME / DNS-01	Protocolo de emisión de certificados; el reto DNS-01 permite wildcard.
Claim	Atributo dentro de un token JWT (aquí, tenant_id).
Hash chain	Auditoría donde cada entrada incluye el hash de la anterior.
Ingesta	Superficie HTTP exclusiva para agentes (ingest.aitamit.com).
JWKS	Conjunto de claves públicas de Keycloak contra el que la API valida los JWT.
MDM / ADE	Gestión de móviles Apple; ADE (Automated Device Enrollment) = enrolamiento zero-touch vía Apple Business Manager.
Conector MDM	Integración de solo lectura con Jamf Pro / Microsoft Intune para inventariar iOS sin re-enrolar.
Relay de control remoto	Servicio WSS del backend que empareja agente y operador por sesión para el control remoto de escritorio.
Migración forward-only	Cambio de esquema sin script de reversa; se deshace restaurando backup.
PKCE	Extensión OAuth para clients públicos (la consola).
Realm	Espacio de identidad de Keycloak (aitamit).
RPO / RTO	Pérdida máxima de datos / tiempo máximo de recuperación asumidos.
Standalone (Next.js)	Build autocontenida; los public/* viajan dentro de la imagen.