SEO Readiness Portal

Documentation for ALAI SEO Readiness Portal MVP — client audit workflows, GBP local-first mode, reporting.

GBP/Local-First Audit Mode (No-Website Clients)

GBP/Local-First Audit Mode (No-Website Clients)

Implemented: MC #102861 (2026-06-03)
Driver: First real client Smoke House Hadžići (no website, GBP-only) via partner Asmir
Status: PASS — type-check, build, validate:phase4, validate:phase5, validate:gbp-local

Overview

The SEO Readiness Portal now supports clients who do not have a website and rely solely on Google Business Profile (GBP) as their primary online presence. This is a common scenario for small local businesses (restaurants, trades, services).

When a client is detected as **no-website**, the audit runner switches into **GBP-first mode** and runs a different checklist tailored to local optimization best practices.

Detection Logic

No-website mode is activated when:

- `intake.hasWebsite === false` (explicit opt-in), OR
- `site.canonicalUrl` is empty or blank (implicit)

The detection is implemented in `src/lib/audit/runner.ts` function `isNoWebsiteMode(site, intake)`.

New Intake Fields (Optional, Non-Breaking)

Three new fields have been added to the `Intake` type (`src/lib/workspace/types.ts`):

Field	Type	Purpose
`hasWebsite`	`boolean?`	Explicit override: when false, forces no-website mode even if canonicalUrl is not empty
`gbpClaimed`	`boolean?`	Whether the business owner has claimed the GBP listing
`gbpManagerAccess`	`boolean?`	Whether ALAI/partner has been added as a manager to the GBP listing

All three are **optional** and **undefined-safe**. Existing fixtures and validations are unaffected.

GBP-First Checklist (12 Checks)

When no-website mode is active, the audit runs the following checks instead of website-dependent checks:

Check ID	Severity	Category	Description
`intake-not-submitted`	P1	evidence	Intake must be submitted or reviewed before workspace is review-ready
`gbp-first-strategy`	P1	local	**Strategy finding:** recommends GBP-first approach + optional simple-site later
`gbp-not-claimed`	P0	local	Business owner has not claimed the GBP listing → cannot manage or optimize
`gbp-manager-access-missing`	P1	local	ALAI/partner not added as manager → cannot perform optimization work
`gbp-claim-status-unknown`	P1	local	Claim/manager status not confirmed → action recommended to clarify
`gbp-nap-url-missing`	P1	local	GBP URL not recorded → NAP consistency cannot be verified
`gbp-primary-category-missing`	P2	local	Priority services missing → cannot determine correct GBP primary category
`gbp-hours-not-confirmed`	P2	local	Business hours not confirmed → can reduce customer trust

Check ID	Severity	Category	Description
`gbp-photos-not-confirmed`	P2	local	Photos not confirmed → profiles with photos attract more engagement
`gbp-review-routine-not-confirmed`	P2	local	Review response routine not set → important trust/engagement signal
`gbp-posts-not-confirmed`	P2	local	GBP posts activity not confirmed → posts keep profile active and relevant
`missing-priority-services`	P2	content	Priority services missing → needed for page/backlog recommendations
`missing-competitors`	P2	content	Competitor list missing → needed for manual review context
`access-status-unknown`	P2	evidence	Analytics/Search Console access status unknown

Total: 12 checks in no-website mode vs. 6 in standard website mode.

Skipped Website-Dependent Checks

The following checks are **conditionally skipped** in no-website mode:

- `httpsCanonicalCheck` — requires canonicalUrl
- `domainMatchCheck` — requires canonicalUrl and domain

These are replaced by the GBP-specific checks above.

Report/Backlog/Export Flow

GBP-first findings flow through the **existing report/backlog/export workflow** with no changes:

- Findings are categorized (new `local` category introduced)
- Backlog conversion works as normal
- Markdown export includes all GBP findings
- No-ranking disclaimer is preserved

Validation Scripts

A new fixture script validates the no-website path:

```
npm run validate:gbp-local
```

This runs `scripts/validate-gbp-local.ts` which:

- Creates a test client with empty `canonicalUrl`
- Runs the local audit in no-website mode
- Asserts exactly 12 checks are run
- Confirms `gbp-first-strategy` finding is present

Existing validations remain unchanged and passing:

- `npm run validate:phase4` — standard website path (6 checks)
- `npm run validate:phase5` — report generation

GBP API Integration (Deferred)

Important operator note: This is a **manual-input MVP**. The portal does **not** integrate with the Google Business Profile API to fetch live data (photos count, reviews count, hours, etc.).

All GBP checks rely on:

- Intake fields (`gbpClaimed`, `gbpManagerAccess`, `googleBusinessProfileUrl`)
- Intake notes (hours/photos/reviews confirmed via text patterns)

Live GBP API integration is a **follow-on scope**, likely in parallel with Google Search Console/Analytics integration (MC #102806 deferred).

Safe Defaults

All GBP checks follow a **fail-closed** philosophy:

- If a field is `undefined` or unknown → emit an "action recommended" finding (P1 or P2)
- Never emit a false pass

Example: if `gbpClaimed` is `undefined`, the check emits `gbp-claim-status-unknown` (P1) asking the operator to confirm with the client.

Implementation Evidence

- **Source:** `src/lib/audit/runner.ts` lines 124-437
- **Types:** `src/lib/workspace/types.ts` lines 54-63
- **Fixture:** `scripts/validate-gbp-local.ts`
- **Validation:** `npm run validate:gbp-local` passes
- **Build:** `npm run type-check` and `npm run build` pass

Summary

The SEO Readiness Portal now handles **two client profiles**:

1. **Website clients** (existing path) — 6 checks, website-dependent validation
2. **No-website / GBP-only clients** (new path) — 12 checks, GBP-first optimization

Detection is automatic based on `canonicalUrl` or explicit `hasWebsite` flag. Both paths use the same report/backlog/export workflow. No breaking changes to existing fixtures or validations.

SEO Client Workflow — Skill (/seo-client) + Intake Auto-Ingest

SEO Client Workflow — Layers 2+3 Architecture

Status: Active (2026-06-03)
Driver: Asmir SEO partner clients (first = Smoke House Hadžići)
MCs: #102865 (skill), #102866 (intake ingest)
Owner: Skillforge (docs), CodeCraft (ingest), FlowForge (launchd)

Architecture Overview

The SEO client workflow is structured in 3 layers:

- **Layer 1:** SEO Readiness Portal tool — local/cloud dual-mode audit, GBP/local-first for no-website clients (MC #102861, see GBP/Local-First Audit Mode)
- **Layer 2:** `/seo-client` skill — generates 5 Bosnian DRAFT deliverables from intake
- **Layer 3:** Intake auto-ingest — parses inbound email into portal workspace

Layer 2: /seo-client Skill

Location

`~/.claude/skills/seo-client/SKILL.md`

Purpose

Given a client SEO intake (from SEO Readiness Portal workspace data **OR** pasted partner email), produce 5 review-ready **DRAFT** deliverables in Bosnian. Never auto-send. All outputs are local drafts under `/tmp/alai/seo-client/<client-slug>/` for human review before delivery.

Inputs

- **Portal workspace:** `/path/to/portal/.data/workspace.json` (client, site, intake objects)
- **Pasted intake email:** numbered questionnaire text (like Asmir partner emails)
- **Email ID:** `--email-id <id>` (fetches via `email-inbox.js`, read-only)

Deliverables (5 Bosnian DRAFTS)

1. **gbp-content.md** — GBP optimization content:
   - Business description (max 750 chars, keyword-rich, natural BS)
   - Primary + 2-3 secondary categories (from GBP taxonomy)
   - Suggested attributes (wheelchair accessible, parking, etc.)
   - 3-5 sample GBP posts (seasonal offers, service updates, local events)
   - Review-ask template (polite, brief, BS)
2. **competitor-benchmark.md** — Competitor analysis:
   - For each competitor (up to 3): GBP completeness, review count/rating, categories, notable strengths
   - Client gap analysis + quick wins
   - Actionable recommendations
   - *Note:* No live GBP API scraping (Phase 10+); manual research or intake-provided competitor URLs
3. **client-report.md** — Client-facing SEO readiness report (BS):
   - Executive summary (current status + top opportunity)
   - Goals (from intake: target location, priority services, conversion action)
   - Current status table (Technical SEO, On-page, Content, Local SEO, Conversion — RAG status)
   - Top findings (P0/P1/P2 prioritized)
   - Recommended action plan (Week 1-2, Month 1, Month 2-3)
   - What we need from client (access, approvals, content)
   - Measurement (KPIs: organic traffic, GBP interactions, contact forms)
   - **Disclaimer:** SEO does NOT guarantee ranking/traffic
   - *Evidence grounding:* portal audit findings (if website exists) or GBP checklist (if GBP-first mode)
4. **client-email-draft.md** — Draft client email (BS):
   - Subject line suggestion
   - Summary: what was reviewed, top findings
   - Immediate asks: missing access, info gaps, priorities
   - Next steps: what ALAI/partner will do after client responds
   - Disclaimer: readiness review only, no ranking/traffic guarantee
   - **Mark as DRAFT** — never auto-send
5. **owner-gbp-claim-guide.md** — Step-by-step GBP claim + add-ALAI-as-Manager guide (BS):
   - Go to business.google.com
   - Search for business name + location

- Claim existing listing (if unclaimed)
- Verify ownership (postcard/phone/email)
- Complete profile: hours, photos, services, description
- Add ALAI/partner as Manager (Settings → Users → Add user, role: Manager)
- Notify ALAI/partner when access granted
- *Screenshot placeholders:* `[Screenshot: GBP claim button]`

Guardrails

- **No ranking/traffic guarantees:** forbidden words enforced (`ranking|traffic|guarantee|guaranteed|first page|top results`)
- **Drafts only:** all outputs are local files; NO auto-send, NO email trigger
- **Optional Lexicon BS check:** route to Lexicon (Dževad Jahić) for linguistic QA if requested
- **MAX TURNS:** 30 (skill execution timeout)
- **No fabricated metrics:** no Search Console/Analytics data without verified access
- **No hardcoded competitor data:** always from intake or prompt user

Workflow Modes

- **GBP_FIRST mode:** no website detected (`intake.hasWebsite === false` OR `site.canonicalUrl` empty)
  - Focus: `owner-gbp-claim-guide.md` as P0 deliverable
  - Skip website-specific audit findings
  - All deliverables optimized for GBP
- **WEBSITE_AND_GBP mode:** website exists
  - Run portal audit: `npm run audit -- --client <slug>`
  - Parse audit findings for client report evidence
  - Include both website + GBP recommendations

Evidence Bundle

On completion, creates `/tmp/alai/seo-client/<client-slug>/DELIVERABLES-MANIFEST.md` with:

- File checklist (all 5 deliverables)
- Validation: PASS (no forbidden words)
- Linguistic QA: PENDING | REVIEWED_BY_LEXICON
- Approval: PENDING_HUMAN_REVIEW
- Next steps: human review required before client delivery

Registration

Registered in `~/.claude/skill-registry.db`:
- **Name:** seo-client
- **Triggers:** /seo-client, "seo client workflow", "process seo intake", "generate seo deliverables", "asmir seo tip"
- **Domain:** seo-operations
- **Level:** 3
- **Max Turns:** 30
- **Status:** active

Layer 3: Intake Auto-Ingest

Location

`~/business/ALAI-Holding-AS/products/SEO-Readiness-Portal/scripts/ingest-intake-email.ts`

Purpose

Parse inbound client SEO intake email (numbered questionnaire format) and append a DRAFT Client+Site+Intake record to the portal workspace. Idempotent via ingest ledger. Read-only email access; no email send.

Usage

```
# Ingest specific email by ID
npm run ingest:intake -- --email-id <id>

# Scan pending ACTION emails (list only)
npm run ingest:intake -- --scan

# Dry-run mode (no writes)
npm run ingest:intake -- --email-id <id> --dry-run

# Auto-run audit after ingest (deferred; not implemented in current draft)
npm run ingest:intake -- --email-id <id> --run-audit
```

Workflow

1. **Email fetch:** Shell-out to `node ~/system/tools/email-inbox.js show <id>` (read-only, no network calls)

2. **Parse:** Extract numbered intake fields:
   - Business name, location, services, languages
   - Target markets (local/regional/national)
   - Competitors (names or URLs)
   - GBP status (claimed/unclaimed, manager access yes/no)
   - Website status (has website yes/no, URL if yes)
   - Analytics/Search Console status
   - Priority services, conversion goals
3. **Workspace append:** Create draft Client, Site, Intake objects:
   - `client.status = "lead"`
   - `intake.status = "submitted"`
   - `intake.hasWebsite = inferred from question "Do you have a website?"`
   - `site.canonicalUrl = empty if no website (triggers GBP-first mode in skill)`
4. **Idempotency:** Dedup by source email ID via `.data/ingest-ledger.json`:
   - Schema: `{ schemaVersion: 1, updatedAtUtc, ingestedEmailIds: { [emailId]: { emailId, clientId, siteId, intakeId, ingestedAtUtc } } }`
   - If email ID already in ledger → skip (NOOP)
5. **Output:** JSON summary to stdout:
   - Email ID, client ID, site ID, intake ID
   - Client name, status, target markets
   - Site domain (or "no website — GBP/local mode")
   - Intake status, hasWebsite, gbpClaimed, gbpManagerAccess, priorityServicesCount, competitorsCount

Guardrails

- **DRAFT only:** status = "lead"/"submitted" — no active/onboarded promotion
- **No audit auto-run:** unless `--run-audit` flag (not implemented; manual run via portal)
- **No email send:** read-only email access, no outbound mail
- **No production mutation:** writes only to local workspace file (`.data/workspace.json`) + ledger

Launchd Wrapper (Deferred)

Template at `scripts/launchd/com.alai.seo-intake-ingest.plist`:
- NOT installed (manual trigger for now)
- When enabled: periodic scan for ACTION emails, auto-ingest eligible intake questionnaires
- Safe + logged (writes to `~/logs/seo-intake-ingest.log`)

End-to-End Flow

1. **Intake email arrives** (from Asmir or direct client) → tagged ACTION by email-inbox.js rules
2. **Layer 3 ingest:** `npm run ingest:intake -- --email-id <id>` → creates draft client in portal workspace
3. **Portal GBP audit:** CEO or automated task runs audit (Layer 1) → evidence collected
4. **Layer 2 skill:** `/seo-client` generates 5 Bosnian DRAFT deliverables
5. **CEO review:** human reviews drafts at `/tmp/alai/seo-client/<client-slug>/`
6. **Send:** CEO or agent manually copies content to client email / BookStack client page

Limitations / Deferred

- **Live GBP API:** Google OAuth + Places API deferred pending CEO approval (Phase 10+)
- **Launchd not installed:** manual trigger for intake ingest (no auto-scan yet)
- **No auto-send:** all deliverables require human review gate
- **No Search Console/Analytics API:** access verification deferred
- **Single language:** Bosnian (BS) only; EN/DE deferred
- **No BookStack page auto-creation:** manual copy-paste for client-facing docs

Related Documentation

- Layer 1: GBP/Local-First Audit Mode
- SEO Operations Playbook: `~/business/ALAI-Holding-AS/sales/seo-automation/SEO-OPERATIONS-PLAYBOOK.md`
- Client Report Template: `~/business/ALAI-Holding-AS/sales/seo-automation/seo-client-report-template.md`
- Bosnian Linguistic QA: `~/system/rules/bosnian-linguistic-validation.md`

Evidence

- **MC #102865:** /seo-client skill authored at `~/.claude/skills/seo-client/SKILL.md`
- **MC #102866:** Intake ingest script at `scripts/ingest-intake-email.ts`
- **Test fixture:** Asmir email #8792 (Smoke House Hadžići)
- **Skill registration:** `~/.claude/skill-registry.db`
- **Status:** DRAFT (no deploy, no auto-send, no production data mutation)

SEO Portal intake-audit loop + chatbot P0 security (MC 103100/103105/103101) 2026-06-07

SEO Portal: intake-audit loop + chatbot P0 security

Date: 2026-06-07 | **MCs:** #103100 (intake-audit loop), #103105 (chatbot P0 security), #103101 (chatbot eval) | **Deployed image:** alairegistry.azurecr.io/seo-readiness-portal:20260607-intake-audit-p0 | **Source commit:** c6aed80ab (branch seo-intake-audit-loop-103100)

What changed

#103100 — intake-audit loop (gaps B + C)

- Client intake business context (competitors, priorityServices, targetMarkets, businessSummary) is now threaded from the pipeline into the audit runner, report generator and action-plan generator. Previously this rich intake data was collected but ignored.
- Connected GSC/GA OAuth state is plumbed into the audit runner and the "Access and measurement readiness" report section, overriding the self-reported dropdowns with real connection data, with graceful fallback when OAuth is not connected.
- Files: src/lib/audit/runner.ts, src/lib/reports/generator.ts, src/lib/reports/action-plan.ts, src/lib/workspace/persistence.ts. No-ranking disclaimer and forbiddenClaimWords guard preserved.

#103105 — chatbot P0 security fixes (from #103101 eval)

- **P0-A** Prompt injection: user message newlines are stripped before building the conversation, and the Ollama call was switched from /api/generate (flat prompt) to /api/chat (structured roles) so role boundaries are enforced server-side. All three dispatchChat callers updated (route.ts, action-plan.ts, deliverables.ts).
- **P0-B** Input guard: now scans ALL user-role messages, not just the last one. Previously an injection in message[0] of a multi-turn payload bypassed the guard.
- **P0-C** XSS: LLM output is HTML-escaped before markdown transforms / dangerouslySetInnerHTML in ChatMessage.tsx, so injected tags render inert.

Verification

- type-check EXIT 0; next build compiled clean (resolved a stale middleware.ts/proxy.ts conflict; canonical = middleware.ts).
- P0-B runtime-proven: secret word in message[0] (clean last message) triggered the pre-canned guard reply, no LLM call.
- P0-A runtime-confirmed: newline role-injection did not hijack the model; /api/chat path functional (Ollama replied).
- P0-C code-verified (DOM screenshot deferred — browser offline).
- Live post-deploy probes (seo-tools.snowit.ba): /intake/test 200, /api/health 200, /api/intake-chat (bad token) 401.
- Evidence: /tmp/evidence-103105/runtime-verification.md, /tmp/evidence-103105/deploy-verification.md, /tmp/evidence-103100/validation-results.txt, /tmp/alai/seo-chatbot-review/REPORT.md

Known follow-ups

- #103111 — monorepo alai-holding.git unpushable (>100MB files); source committed locally only, not on remote. Restore push/PR path.
- Remaining chatbot improvements from eval: 6 of 11 intake fields are structurally hard to extract (P1), no confirm-before-prefill step, in-memory rate limiter, unbounded paid-tier cost on Ollama outage.

SEO Portal Path C — Delegated Service-Account GSC/GA via Workload Identity Federation (MC #103390)

Overview

MC #103390 — SEO Readiness Portal Path C replaces the blocked per-client Google OAuth flow with a delegated read-only service account authenticated via **Workload Identity Federation (WIF)**. No static JSON key is created or stored; the portal authenticates entirely through Azure App Service's managed identity federating into GCP.

Why Path C?

The previous OAuth path required Google to verify the app's OAuth consent screen for restricted scopes (`webmasters.readonly` + `analytics.readonly`). Asmir (pilot client, snowit.ba) hit the hard block: "*Access blocked: snowit.ba has not completed the Google verification process*" on `/api/oauth/google/start`. Google app verification for restricted scopes takes weeks and requires domain validation, brand approval, and security review — not feasible for an MVP launch.

Path C sidesteps this entirely: the portal reads GSC and GA4 data as a single read-only service account that each client explicitly grants access to in their own Google properties. No OAuth consent screen, no Google app verification, no per-user token storage.

Service Account

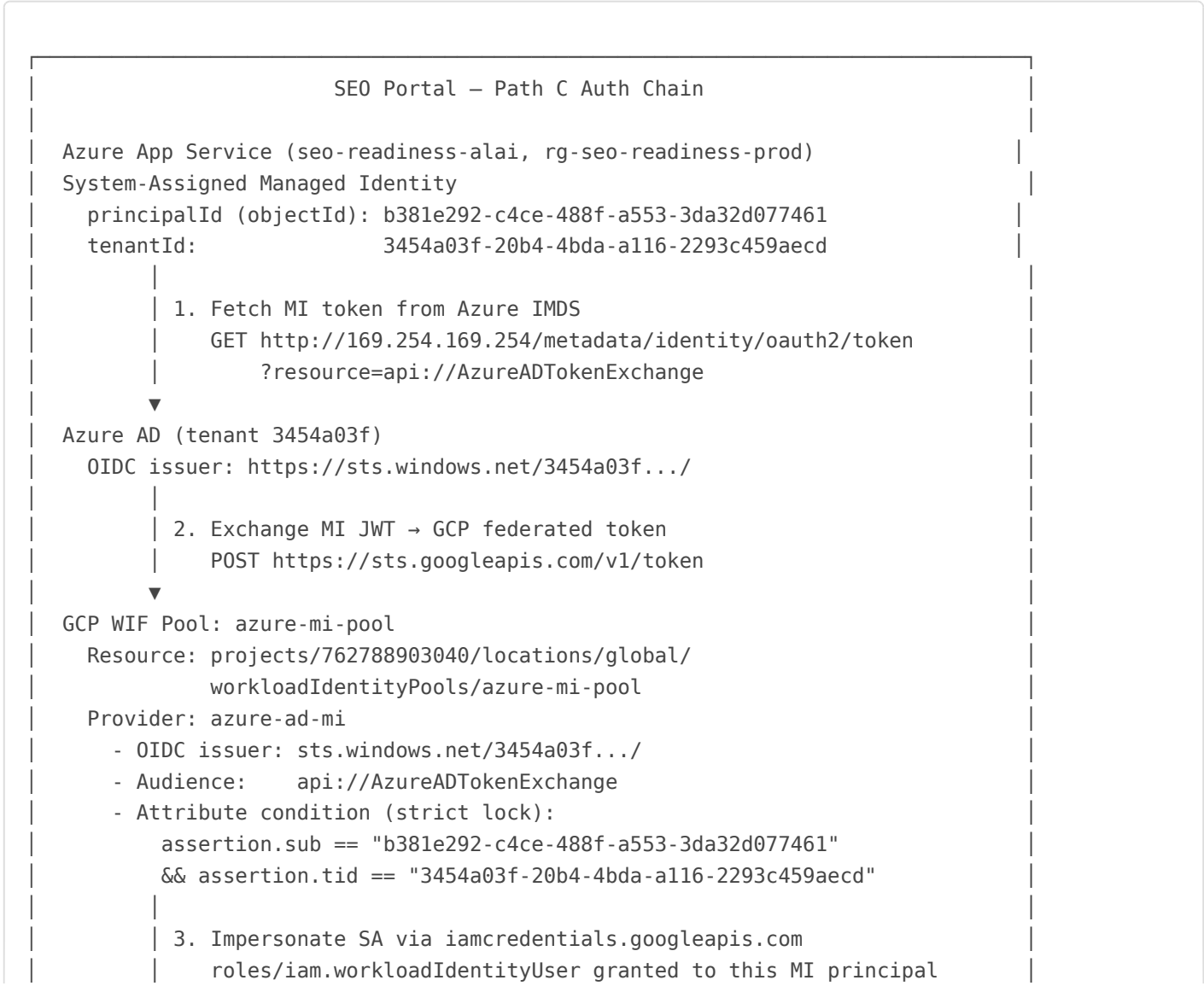
Field	Value
SA email (canonical)	`seo-gsc-reader@tribal-sign-487920-k0.iam.gserviceaccount.com`
GCP project	`tribal-sign-487920-k0` (project number: 762788903040)
Display name	SEO Portal GSC/GA read-only reader
Unique ID	103977132559951298607

Field	Value
Project IAM roles	None — access granted per-property by each client only
APIs enabled	searchconsole.googleapis.com, analyticsdata.googleapis.com, iamcredentials.googleapis.com, sts.googleapis.com

Workload Identity Federation Chain

Because the GCP organisation (ID: 897698145517) enforces the org policy `constraints/iam.disableServiceAccountKeyCreation`, no static JSON key can be created. WIF allows the Azure App Service's managed identity to impersonate the SA at runtime without any long-lived credential.

Architecture Diagram



```
|         ▼                                                                  |
|  Service Account: seo-gsc-reader@tribal-sign-487920-k0.iam.gserviceaccount.com |
|    Short-lived access token (1h TTL, auto-refreshed)                       |
|         |                                                                  |
|         | 4a. GSC Search Analytics API                                     |
|         |     searchconsole.searchanalytics.query(siteUrl)                 |
|         | 4b. GA4 Data API                                                 |
|         |     analyticsdata.runReport(properties/NNNN)                     |
|         ▼                                                                  |
|  Client's GSC + GA4 property (client must grant SA email as Viewer/Restricted) |
└────────────────────────────────────────────────────────────────────────────┘
```

WIF Resource Names

Resource	Name / Value
WIF Pool ID	`azure-mi-pool`
WIF Pool resource	`projects/762788903040/locations/global/workloadIdentityPools/azure-mi-pool`
WIF Provider ID	`azure-ad-mi`
WIF Provider resource	`projects/762788903040/locations/global/workloadIdentityPools/azure-mi-pool/providers/azure-ad-mi`
OIDC issuer	`https://sts.windows.net/3454a03f-20b4-4bda-a116-2293c459aecd/`
Allowed audience	`api://AzureADTokenExchange`
Attribute condition	assertion.sub == "b381e292-..." && assertion.tid == "3454a03f-..."
SA impersonation member	`principal://iam.googleapis.com/projects/762788903040/locations/global/workloadIdentityPools/azure-mi-pool/subject/b381e292-c4ce-488f-a553-3da32d077461`

WIF Credential Config (non-secret)

Stored as Azure App Service app setting `GOOGLE_APPLICATION_CREDENTIALS_JSON` on `seo-readiness-alai`. This JSON contains no private key material — it is safe to log at the config level (but not to expose to browsers). Local copy: `/tmp/evidence-103390/wif-credential-config.json`.


```
{
  "type": "external_account",
  "audience": "//iam.googleapis.com/projects/762788903040/locations/global/workloadIdentityPools
  "subject_token_type": "urn:ietf:params:oauth:token-type:jwt",
  "token_url": "https://sts.googleapis.com/v1/token",
  "service_account_impersonation_url": "https://iamcredentials.googleapis.com/v1/projects/-/serv
  "credential_source": {
    "url": "http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resourc
```

```
    "headers": { "Metadata": "true" },
    "format": { "type": "json", "subject_token_field_name": "access_token" }
  }
}
```

Connection-Verify Flow (verifyAndSaveGrant)

The server action `verifyAndSaveGrant()` in `src/app/intake/[token]/access-grant/actions.ts` handles the client grant confirmation:

1. **Token gate:** clientId is extracted from the HMAC intake token payload only — never from user input.
2. **Field validation:** GSC requires `siteUrl`; GA4 requires `ga4PropertyId`.
3. **Live probe:** Calls `verifyAccess()` from `sa-reader.ts`, which attempts one real API call using the WIF-authenticated SA. A 403 from Google means the client has not yet added the SA as a user.
4. **Persist on success:** Calls `saveGoogleConnectionSa()` — stores `connectionMode="delegated_sa"`, sentinel token values, `grantVerified=true`, `verifiedAt=now`. Idempotent on platform.
5. **Error isolation:** `not_granted` → nothing persisted, SA email returned for display. `api_error` / `misconfigured` → generic error, nothing persisted. Token value never included in any returned detail.

Security Posture

Property	Details
No static key	Org policy `constraints/iam.disableServiceAccountKeyCreation` is respected. No JSON key was created or stored anywhere.
Read-only SA	SA holds no project IAM roles. Access is granted only per-property by the client in their own GSC/GA4 settings.
Strict WIF attribute condition	Only the specific Azure MI (objectId b381e292) from the specific tenant (3454a03f) can impersonate the SA. No other identity can use this WIF provider.
Short-lived tokens	WIF-generated access tokens have a 1-hour TTL and are fetched at runtime by the google-auth-library ADC. No long-lived credential on disk.
clientId isolation	verifyAndSaveGrant() derives clientId from HMAC-signed token only. A client cannot probe another client's property.

Property	Details
No secret logging	Verified via grep in ST2 evidence: no console calls, no token/auth/credential values logged anywhere in sa-reader.ts.
Scope	Reads are scoped to GSC Search Analytics and GA4 Data API. No write scopes, no Admin SDK, no domain-wide delegation.

No-Key Rotation / How WIF Differs from Key Rotation

Traditional SA Key Rotation

With a static JSON key (`GOOGLE_SA_KEY`), the operational runbook is:

1. Generate a new key via GCP IAM Console.
2. Update the Azure App Service secret `GOOGLE_SA_KEY`.
3. Restart the App Service to pick up the new value.
4. Verify the old key is revoked in GCP IAM → Service Accounts → Keys.
5. Repeat every 90 days (or on suspected compromise).

This path is not available because `constraints/iam.disableServiceAccountKeyCreation` is enforced at org level (org ID: 897698145517). No account with org-level `orgpolicy.policyAdmin` is currently credentialed.

WIF — There Is Nothing to Rotate

The WIF credential config (`GOOGLE_APPLICATION_CREDENTIALS_JSON`) contains no private key material. It is a routing config that tells the google-auth-library how to reach the Azure IMDS endpoint and which WIF pool to exchange against. It is not a secret. Tokens are minted at runtime by Azure MI (renewed automatically by the platform) and exchanged for short-lived GCP access tokens.

Operational actions that DO require maintenance:

- **If the Azure App Service is recreated / MI changes:** The new MI objectId must be added as a new `roles/iam.workloadIdentityUser` binding on the SA, and the WIF provider attribute condition must be updated to the new objectId. Update `GOOGLE_APPLICATION_CREDENTIALS_JSON` with the new MI's IMDS URL if the subscription/tenantId changes.
- **If the GCP WIF pool/provider needs to be re-created:** Re-run ST1 FlowForge steps B–D (pool create, provider create, binding). Update `GOOGLE_APPLICATION_CREDENTIALS_JSON`

audience URL with the new pool resource name.
- **If the SA email changes:** Update `service_account_impersonation_url` in the WIF config, update all client-facing onboarding instructions with the new SA email.
- **Audit / verify the chain is intact (periodic check):** From Azure App Service Kudu console, run the 4-step ST6 probe sequence (documented in st1-flowforge.md): IMDS token fetch → GCP STS exchange → ADC token → tokeninfo SA email confirmation.

Summary: WIF eliminates the 90-day key rotation burden and removes the attack surface of a long-lived credential leaking from a secrets store. The tradeoff is that the WIF configuration binds to infrastructure identities (Azure MI objectId) rather than a portable key — so infra changes require config updates, not just secret rotation.

Evidence Files

- `/tmp/evidence-103390/st1-flowforge.md` — SA creation, API enablement, WIF provisioning (Phase 1 + Phase 2)
- `/tmp/evidence-103390/st2-codecraft.md` — sa-reader.ts implementation, 19/19 Vitest tests PASS
- `/tmp/evidence-103390/st3-codecraft.md` — verifyAndSaveGrant action, GoogleConnection model pivot, 31/31 tests PASS
- `/tmp/evidence-103390/wif-credential-config.json` — non-secret WIF credential config (reference copy)
- `/tmp/evidence-103390/client-onboarding-grant-access-bs.md` — Bosnian client onboarding instructions (LEXICON-PENDING)
- `/Users/makinja/business/ALAI-Holding-AS/products/SEO-Readiness-Portal/SPEC-PATH-C-DELEGATED-SA-103390.md` — original specification

Client Onboarding Summary

Each client must add the SA email as a user in their own Google properties before the portal can read data. The steps are documented in the Bosnian client-facing document (file path above, Lexicon validation pending). In brief:

1. **Google Search Console:** Settings → Users and permissions → Add user → paste `seo-gsc-reader@tribal-sign-487920-k0.iam.gserviceaccount.com` → permission Restricted or Full → Add.
2. **Google Analytics 4:** Admin → Property Access Management → + → paste same email → role Viewer → Add.
3. **Portal grant page:** Paste GSC site URL + GA4 property ID → click Verify. The `verifyAndSaveGrant()` action probes the live APIs and persists the connection on success.

Open Items / Follow-On

- **ST4/Vizu:** UX for the access-grant page (paste siteUrl/propertyId + verify button) — not yet built.
- **ST5/Securion:** OAuth retirement — disable per-client OAuth start/callback, confirm no dead secret exposure.
- **ST6/Proveo:** Real E2E against live snowit.ba GSC + GA4 property — final verification gate.
- **Lexicon:** Bosnian client onboarding text requires Dževad Jahić validation before client send.
- **Push blocker:** ST2/ST3 code on branch `seo-path-c-sa-reader-103390` could not be pushed (network error from build context). Orchestrator to retry push from network-capable context.

SEO Pipeline — Portal Intake ? John Deep-Report (agent runbook)

Purpose

The SnowIT SEO offering uses the portal for intake and crawl at scale (zero partner effort). John (agents) produces the quality deep report. The portal’s own auto-report is a shallow teaser; the real deliverable is John’s deep-report play. This runbook is the canonical reference for any agent session picking up SEO work.

Architecture / Flow

```
Asmir (partner) adds client via /partners in portal
  OR John mints magic-link directly
        |
        v
System emails HMAC-bound magic-link to end client
        |
        v
Client opens https://seo-tools.snowit.ba/intake/<token>
  fills smart form + AI chatbot
        |
        v
Portal auto-crawls site
  stores intakeSubmission + crawl summary in workspace.json
  fires notify_audit_ready webhook/email
        |
        v
seo-intake-watcher (LaunchAgent) polls workspace every 10 min
  detects NEW submission
  emails john@alai.no
```

```
        |
        v
John agent session runs ~/system/prompts/seo-deep-report-play.md
        |
        v
Quality deep report delivered to Asmir / client
```

Key Live Facts (verified 2026-06-22)

Item	Value
Portal app	Azure App Service `seo-readiness-alai`, RG `rg-seo-readiness-prod`
Current known-good image	`20260622-audit-depth` (v1.1.0 crawl audit engine)
Public URLs	seo-tools.snowit.ba / seo-tools.alai.no
Deploy method	`az acr build` + `az webapp config container set` (DEPLOY-MAP.md in repo root; ZAKON PI2)
Workspace read (read-only)	Kudu VFS + AAD token: `az account get-access-token --resource https://management.azure.com` then `GET https://seo-readiness-alai.scm.azurewebsites.net/api/vfs/data/workspace.json`
Handoff watcher script	`~/system/tools/seo-intake-watcher.js`
LaunchAgent	`com.john.seo-intake-watcher` (10-min poll; emails john@alai.no on NEW intake only)
Watcher state	`~/system/state/seo-intake-watcher-seen.json`
Deep-report play	`~/system/prompts/seo-deep-report-play.md`
GSC/GA layer	OPTIONAL / separate — pending OAuth verification (MC #103428). Crawl-based audit needs NO Google credentials.

Trigger Checklist — Receiving a New Intake

1. Email arrives at john@alai.no with subject containing "SEO intake" or "audit_ready".
2. Open workspace.json via Kudu VFS (read-only; see table above) and confirm `intakeSubmission` is present for the client domain.

3. Read `~/system/prompts/seo-deep-report-play.md` — follow it exactly. Do not improvise the report structure.
4. Deliver completed report to Asmir (partner) and optionally to client email on record.
5. Mark intake as processed in seen-state: add domain + timestamp to `~/system/state/seo-intake-watcher-seen.json`.

When to Use the Portal vs Do It Directly

For a small number of one-off clients, John running the audit directly (without the portal intake flow) is faster and produces a deeper initial result. The portal earns its keep at intake *volume* — when Asmir is referring multiple clients per week and cannot afford to coordinate each one manually. (CEO note 2026-06-22.)

Anti-Pitfalls for Agents

- **Portal auto-report can produce false positives.** Example verified 2026-06-22: bilko.cloud robots.txt and sitemap.xml were reported as 200 OK by the shallow auto-report when both URLs actually returned homepage HTML. Always verify by live outcome (`curl -sI <url>`), not the portal's own summary.
- **One deploy per task.** Build agents must perform exactly one deploy cycle. Do not iterate deploys within a single MC task unless explicitly unblocked.
- **No new endpoints/scope without a new MC.** If the task is a runbook write, do not add API routes or portal features as a side-effect.
- **Never hand-write expected results into workspace.json to fake verification.** Evidence must be machine-generated from actual portal behaviour.
- **Deploy path is ZAKON PI2.** Always check DEPLOY-MAP.md in repo root before any `az acr build` or `az webapp` command.

Evidence & Memory Links

- Memory project: `project_seo_pipeline_portal_plus_john_2026-06-22`
- Memory project: `project_seo_portal_live_autonomous_2026-06-22`
- Related pages:
  - SEO Readiness Portal MVP — Status and Evidence (new-client-template book)
  - SEO Readiness Portal Cloud Migration — 2026-06-01 (SnowIT book)
  - SEO Client Workflow — Skill + Intake Auto-Ingest
  - SEO Portal intake-audit loop + chatbot P0 security (2026-06-07)