**Document: 04 – System Context & Architecture**

**Project:** MCPX-KendoBridge Admin Portal  
**Document ID:** TJ-MCPX-DOC-04  
**Version:** 1.1.0  
**Status:** Draft  
**Date:** 2025-09-27  
**Owner:** Systems Architect (Technijian)  
**Confidentiality:** Technijian Internal

**Document Control**

| **Version** | **Date** | **Author** | **Change Summary** | **Status** |
| --- | --- | --- | --- | --- |
| 1.0.0 | 2025-09-26 | Doc-Factory | Initial context & components | Draft |
| 1.1.0 | 2025-09-27 | Systems Architect | Make **Admin UI** in-scope/required; add **MSAL PKCE** flow; codify **JSON vs SSE**; env badge; parity & CORS; error taxonomy | Draft |

**Approvals**

| **Role/Team** | **Name** | **Signature/Date** | **Comment** |
| --- | --- | --- | --- |
| Director of Software Eng. |  |  |  |
| Security/Compliance |  |  |  |
| DBA |  |  |  |
| QA Lead |  |  |  |

**1. Purpose**

Define the system context, trust boundaries, and core architecture of the **MCPX-KendoBridge Admin Portal**, establishing how the **Admin Web (React + KendoReact Fluent 2)**, **Admin API (.NET 8)**, **SQL Server 2022**, **Azure SSO**, and **child MCP bridge (STDIO)** interact across **Alpha → Beta → RTM → Prod** with auditable promotion.

**2. Scope**

In scope: Admin Web SPA, Admin API, DB (SP-only DAL, add-only schema), SSE streaming, error taxonomy, CORS allow-list, Microsoft Graph (optional), CI/CD evidence and config-parity.  
Out of scope: tenant/end-user portals; heavy analytics/BI; non-admin workflows.

**3. References**

* **01 Vision**, **02 Glossary**, **03 Actors & Use Cases**, **04 FR**, **05 NFR**, **10 CI/CD**, **11 Monitoring**, **12 Evidence Pack**, **ADR index**
* **OpenAPI 3.1** — Admin API contract (JSON default; **SSE** when Accept: text/event-stream; legacy /messages & /sse behind feature flag)
* **Design** — design/figma/\*\* prototype; **KendoReact Fluent 2** with **ThemeBuilder** export (design/themebuilder/export/\*\*)
* **Tests** — Gherkin (tests/gherkin/\*.feature), k6 SSE TTFB smoke (tests/perf/k6\_sse\_ttfb.js)

**4. System Overview**

The Admin Portal is a **single, secure control plane** for operating Technijian MCP servers and related services.

* **Admin Web (React + KendoReact Fluent 2)** — SPA that authenticates via **Azure AD (Entra ID) using MSAL PKCE**, reads non-secret **effective config**, renders dashboards, and manages admin operations (config/flags/lookups/jobs/audit/evidence/access/CORS).
* **Admin API (.NET 8)** — Validates **JWT bearer** tokens; enforces **RBAC** (app roles/groups). Returns **JSON** by default; streams **SSE** when requested. Proxies JSON-RPC to the **child MCP** over **STDIO**, normalizing errors into a **standard envelope**.
* **SQL Server 2022** — **Stored-procedure-only** DAL and **add-only** schema. **Non-secrets** may be surfaced via GET /config/effective.
* **Microsoft Graph (optional)** — Directory read/write for access assignments, when enabled by policy.

**5. Actors & External Systems**

* **Portal.Admin** (full control) / **Portal.Viewer** (read-only)
* **Azure AD (Entra ID)** — SSO authority issuing ID/Access tokens with role claims
* **Microsoft Graph** — Optional access management integration
* **GitHub Actions** — CI/CD, evidence artifacts, promotion gates
* **Observability stack** — Logs/metrics/dashboards/alerts

**6. Trust Boundaries & Data Classification**

| **Boundary** | **Entry/Protocol** | **Controls** | **Data Class** |
| --- | --- | --- | --- |
| Browser ↔ API | HTTPS; Authorization: Bearer <JWT>; Origin | JWT validation (issuer/audience/signature/exp), **CORS allow-list** per env, **rate limits**, error envelope | Internal |
| API ↔ DB | SQL over secure network | **SP-only** execution, least privilege, schema **add-only**, audited mutations | Internal |
| API ↔ Child MCP (STDIO) | Process spawn/pipe | Timeouts, backpressure, lifecycle health, error normalization | Internal |
| API ↔ Azure AD/Graph | OIDC/MSAL; Graph REST | Minimal scopes, retries with jittered backoff, circuit breaker | Internal |
| RTM ↔ Prod parity | /ready checks; config diff | **No critical diffs** gate for promotion; evidence retained ≥ 1 year | Internal |

**Secrets** are never returned by API endpoints; **/config/effective** exposes **non-secret** keys only.

**7. Context Diagram (PlantUML)**

@startuml

left to right direction

actor "Portal Admin" as Admin

actor "Portal Viewer" as Viewer

rectangle "Azure AD (Entra ID)" as AAD

rectangle "Microsoft Graph" as Graph

node "Admin Web\n(React + KendoReact Fluent 2)" as Web

node "Admin API\n(.NET 8)" as API

database "SQL Server 2022\nSP-only, add-only" as DB

cloud "Child MCP\n(STDIO)" as MCP

Admin --> Web

Viewer --> Web

Web --> AAD : MSAL PKCE (OIDC)

Web --> API : HTTPS (JWT Bearer)

API --> DB : Stored Procedures

API --> MCP : JSON-RPC via STDIO

API --> Graph : Directory ops (optional)

@enduml

**8. Component Responsibilities**

**Admin Web**

* Auth via **MSAL PKCE**; role-aware routing/guards
* **Environment badge** from /config/effective
* Dashboards (health/ready, p50/p95, error rate, queue depth, version)
* CRUD UI for config/flags/lookups (audited), Jobs with **SSE** progress, Audit & Evidence views, Access & CORS admin

**Admin API**

* JWT authN/Z; **CORS allow-list**; **rate limiting**
* **JSON default; SSE** when Accept: text/event-stream
* Session header issuance/echo as required by SSE notifications
* SP-only DAL & error envelope; feature-flagged **legacy endpoints OFF** by default

**SQL Server**

* sp\_Config\_\*, sp\_Feature\_\*, sp\_Lookup\_\*, sp\_Audit\_\* (canonical patterns)
* Migrations use **add-only** rule (no destructive DDL); VYYYYMMDDHHmm\_\_desc.sql

**9. Key Sequences**

**9.1 Authentication (MSAL PKCE)**

1. User hits portal → MSAL redirects to AAD.
2. AAD authenticates and returns **ID token + Access token** (app role claims).
3. Web stores tokens (session/secure cookie), calls API with Authorization: Bearer.
4. API validates JWT, issues response; Web routes to **/dashboard** with environment badge.

**9.2 Tool Execution (JSON vs SSE)**

1. Client POST /mcp with JSON-RPC body.
2. **Default:** API returns **application/json**.
3. **Streaming:** if Accept: text/event-stream, API opens **SSE**, emitting progress events plus **heartbeats** at configured cadence; first event **TTFB ≤ 200 ms**.
4. Client reconnects with Last-Event-ID (if supported) on transient errors.

**9.3 Background Notifications**

* Client GET /mcp with Mcp-Session-Id opens SSE channel for server-initiated notifications. Missing/invalid session returns **400** with standard envelope.

**9.4 Effective Config Snapshot**

* Client GET /config/effective → API returns **non-secret** merged config + requestId for correlation.

**9.5 Readiness & RTM Parity**

* GET /ready in **RTM** validates dependencies (including **Prod DB read-only**) and blocks promotion on failure or **critical config drift**.

**10. Interfaces (from OpenAPI)**

| **Path** | **Method** | **Semantics** |
| --- | --- | --- |
| /mcp | POST | JSON by default; **SSE** when negotiated |
| /mcp | GET | Open **SSE** stream for background notifications (Mcp-Session-Id required) |
| /healthz, /ready | GET | Liveness / readiness (includes RTM parity signals) |
| /config/effective | GET | **Non-secret** runtime config snapshot + requestId |
| /messages, /sse | VAR | **Legacy** endpoints, **feature-flagged OFF**; 403 feature\_disabled unless enabled |

**Error Envelope (canonical):**  
{ code: string, message: string, details?: object, requestId?: string }  
Representative codes: origin\_forbidden, missing\_session\_id, feature\_disabled, not\_ready, timeout, rate\_limited.

**11. Configuration, Feature Flags & Secrets**

* **Configuration** (non-secret) stored in DB, exposed at /config/effective for UI read; **secrets never surfaced** by API.
* **Feature Flags** (e.g., EnableLegacyHttpSse) gate behavior by env/role; default **OFF**.
* **Canonical expected-config** file used for parity (recommend config/expected/expected-prod.json), compared to RTM.

**12. Security Controls**

* **SSO:** Azure AD **MSAL PKCE**; API validates issuer/audience/signature/exp.
* **RBAC:** App roles/groups (**Portal.Admin**, **Portal.Viewer**); server authoritative; UI hides unauthorized actions.
* **CORS:** **Allow-list** per environment (deny by default).
* **Rate Limiting:** Per principal/IP/session; 429 Retry-After on breach (policy in NFRs).
* **Transport:** TLS 1.2+; HSTS; secure cookies where applicable.
* **Audit:** All admin mutations/write paths logged (who/what/when/before→after).

**13. Observability**

* **Logs:** Structured with requestId, route, role, outcome, latency; SSE start/stop; heartbeat gaps.
* **Metrics:** request rate, error %, p50/p95/p99 latency, queue depth, SSE connections, first-event latency, heartbeat gap alerts.
* **Dashboards:** Health/Ready, latency histograms, error rates, parity status.
* **Alerts:** uptime drop; p95 breach; **SSE heartbeat gap > threshold**; auth failure spikes; parity gate blocked.

**14. Environments & Endpoints**

| **Env** | **Base URL (example)** | **Notes** |
| --- | --- | --- |
| Alpha | https://alpha.example.com/api | Dev/test |
| Beta | https://beta.example.com/api | Pre-RTM |
| RTM | https://rtm.example.com/api | Validates against **Prod DB (read-only)** |
| Prod | https://prod.example.com/api | Live |

**Path casing:** Standardize on api/openapi/mcp-proxy.yaml **or** api/OpenApi/mcp-proxy.yaml and update CI to match—avoid Linux runner case issues.

**15. Risks & Mitigations**

| **Risk** | **Impact** | **Mitigation** |
| --- | --- | --- |
| SSE stalls / slow consumers | User confusion; missed updates | Heartbeats; client backoff & resume; server timeouts; first-event **TTFB ≤ 200 ms** |
| CORS misconfiguration | Legit clients blocked or leakage | DB-backed allow-list; audited edits; optional two-person approval |
| Child process spawn failure | Tool execution unavailable | Map to spawn\_failed; auto-restart; alerting |
| Config drift RTM↔Prod | Broken Prod behaviors | Parity report; **no critical diffs** gate; evidence retained |
| Mixed UI component systems | Inconsistent UX; a11y regressions | **KendoReact Fluent 2 only** in production; ThemeBuilder tokens as SoT |

**16. Open Decisions (ADRs to record)**

* Token storage model (session vs secure cookie) and exact RBAC claim source (app roles vs groups).
* Precise rate-limit thresholds (align to NFRs).
* Access management mode: Graph write vs Change-Request only.
* Canonical OpenAPI folder casing to lock in CI.

**17. Glossary (selected)**

* **Effective Config** — Non-secret, merged runtime configuration exposed to the UI.
* **SSE** — Server-Sent Events (text/event-stream) with progress events and heartbeat comments.
* **Parity Gate** — Release promotion check ensuring RTM matches intended Prod configuration.

**End of Document — TJ-MCPX-DOC-04 v1.1.0**