**MCPX‑KendoBridge — Data & DB Contracts**

**Document:** docs/07\_data\_contracts.docx  
**Project Code:** MCPX‑KendoBridge  
**Version:** 1.0.0 (Draft)  
**Last Updated:** 2025‑09‑23  
**Document Owner:** DoSE (Accountable) — DocFactory (Responsible)

**Purpose.** Define the canonical **database schema**, **stored procedures**, **seed data**, **interface contracts**, and **governance rules** for MCPX‑KendoBridge. All data access is **SP‑only**, schema evolution is **add‑only**, and all **dynamic behavior** (child process command/args, timeouts, allowed origins, flags) is sourced from the DB—**never hard‑coded**. **Secrets are not stored in the DB** and must be configured in GitHub Environments or vendor portals.

**DB COMPLIANCE (applies to this entire document):** **Add‑only** schema; **Stored‑procedure‑only** DAL; **No‑Hard‑Coding**. Configuration & feature flags via AppConfig/FeatureFlag and SPs: sp\_Config\_\*, sp\_Feature\_IsEnabled, sp\_Lookup\_Get. Secrets (SQL connection string, Telerik license) live only in **GitHub Environments** or vendor portals.

**Table of Contents**

1. Scope & Data Classification
2. Logical Model (Overview)
3. Physical Schema (Tables, Keys, Constraints)
4. Configuration Keys (Canonical Catalog)
5. Stored Procedure Contracts (Signatures, I/O, Semantics)
6. Security Model (Users/Roles/Grants)
7. Performance & Operational Considerations
8. Migrations & Change Governance
9. SP‑Only DAL Usage Patterns (C#)
10. Validation & Test Plan (SQL + Integration)
11. Examples & Quick Verification Queries
12. Assumptions & Next Steps

**1) Scope & Data Classification**

* **Scope:** Non‑secret configuration and feature flags that control runtime behavior of the MCP proxy; no business or customer data.
* **Classification:** **Internal / Non‑PII** configuration only.
* **Secrets:** **Never** stored in DB (e.g., SQL connection strings, Telerik license). Configure in **GitHub Environments** / platform vaults; redact in logs and in /config/effective.

**2) Logical Model (Overview)**

**Entities**

* **AppConfig** — string‑keyed configuration values used by the proxy (e.g., child process args, timeouts, allowed origins).
* **FeatureFlag** — boolean toggles (e.g., EnableLegacyHttpSse).
* **Lookup** — reserved abstraction (future); exposed via sp\_Lookup\_Get with a stable interface.

**Relationships**

* Independent tables. Configuration read paths only (no joins required). Updates occur via controlled migrations or admin tooling.

**3) Physical Schema (Tables, Keys, Constraints)**

Naming: dbo schema; identifiers in **PascalCase**; additive evolution only (no destructive DDL). All temporal columns are UTC (DATETIME2).

**3.1 Tables**

**Table: dbo.AppConfig**

| **Column** | **Type** | **Nullable** | **Default** | **Notes** |
| --- | --- | --- | --- | --- |
| Key | NVARCHAR(200) | NO | — | Primary Key; case‑insensitive logical semantics. |
| Value | NVARCHAR(MAX) | YES | — | Canonical string representation (see §4 for per‑key semantics). |
| UpdatedAt | DATETIME2 | NO | SYSUTCDATETIME() | Last update timestamp (set by DB). |

**Constraints**

* PRIMARY KEY on [Key].
* Consider a filtered nonclustered index on keys with large values if needed for read perf (defer until measured).

**Table: dbo.FeatureFlag**

| **Column** | **Type** | **Nullable** | **Default** | **Notes** |
| --- | --- | --- | --- | --- |
| Name | NVARCHAR(200) | NO | — | Primary Key (flag name). |
| Enabled | BIT | NO | — | 1 = on, 0 = off. |
| UpdatedAt | DATETIME2 | NO | SYSUTCDATETIME() | Last update timestamp. |

**Constraints**

* PRIMARY KEY on [Name].

**No secrets in these tables.** All secret material is managed outside the DB.

**4) Configuration Keys (Canonical Catalog)**

**All values are stored as strings** (NVARCHAR(MAX)) in AppConfig. The application enforces types at the edge. **Never hard‑code** these values in code or tests—always fetch via sp\_Config\_\*.

| **Key** | **Type (expected)** | **Default (seed)** | **Valid Range / Format** | **Purpose** |
| --- | --- | --- | --- | --- |
| Mcp:ChildCommand | string | npx | Non‑empty | Executable used to spawn child process. |
| Mcp:ChildArgs | string | -y @progress/kendo-react-mcp@latest | Non‑empty | Arguments for child. |
| Mcp:ChildCwd | string | "" | Path or empty | Working directory for child. |
| Security:AllowedOrigins | csv<string> | https://chat.openai.com,https://platform.openai.com | CSV, absolute HTTPS URLs | CORS/Origin allow‑list checked on every request. |
| Network:SseKeepAliveSeconds | int | 15 | 5..120 | Heartbeat interval for SSE streams. |
| Network:RequestTimeoutSeconds | int | 120 | 10..600 | Overall request timeout (server‑side). |

**Environment overrides:** Each environment (Alpha → Beta → RTM → Prod) **may override** values via migrations or controlled admin tooling. RTM validates on **Prod DB**; ensure parity on these keys before promotion.

**5) Stored Procedure Contracts**

**Policy:** Every DB access from the application uses **stored procedures** (SqlCommand.CommandType = StoredProcedure). SP interfaces are **stable**; changes are **add‑only** (e.g., new optional parameters, new result columns appended after existing ones). Return shapes documented below are **normative**.

**5.1 dbo.sp\_Config\_GetValue**

* **Purpose:** Retrieve a single configuration value by key.
* **Signature:**
* CREATE OR ALTER PROCEDURE dbo.sp\_Config\_GetValue
* @Key NVARCHAR(200)
* AS
* BEGIN
* SET NOCOUNT ON;
* SELECT [Value] FROM dbo.AppConfig WHERE [Key] = @Key;
* END
* **Input:** @Key (exact key, case‑sensitive by DB collation).
* **Output:** **Result set** with a single column [Value] (nullable NVARCHAR(MAX)); no rows if key not found.
* **Error Semantics:** No explicit RAISERROR. Missing key yields empty result; caller handles defaulting.
* **Usage:** Low‑latency reads on hot path; cache in memory with short TTL.

**5.2 dbo.sp\_Config\_GetAll**

* **Purpose:** Enumerate all non‑secret configuration pairs.
* **Signature:**
* CREATE OR ALTER PROCEDURE dbo.sp\_Config\_GetAll
* AS
* BEGIN
* SET NOCOUNT ON;
* SELECT [Key],[Value],[UpdatedAt]
* FROM dbo.AppConfig
* ORDER BY [Key];
* END
* **Output:** Rows with [Key] NVARCHAR(200), [Value] NVARCHAR(MAX), [UpdatedAt] DATETIME2.
* **Usage:** Powers /config/effective endpoint; the service **must redact** any values considered sensitive (none expected here by policy).

**5.3 dbo.sp\_Feature\_IsEnabled**

* **Purpose:** Check a boolean feature flag.
* **Signature:**
* CREATE OR ALTER PROCEDURE dbo.sp\_Feature\_IsEnabled
* @Name NVARCHAR(200)
* AS
* BEGIN
* SET NOCOUNT ON;
* SELECT CAST(COALESCE((SELECT [Enabled] FROM dbo.FeatureFlag WHERE [Name] = @Name), 0) AS BIT) AS [Enabled];
* END
* **Input:** @Name = flag name.
* **Output:** One row, one column [Enabled] BIT (0/1).
* **Usage:** Gates legacy endpoints (/messages, /sse).

**5.4 dbo.sp\_Lookup\_Get (Reserved)**

* **Purpose:** Future‑proofed lookup interface for type/key pairs.
* **Signature:**
* CREATE OR ALTER PROCEDURE dbo.sp\_Lookup\_Get
* @Type NVARCHAR(100),
* @Key NVARCHAR(200)
* AS
* BEGIN
* SET NOCOUNT ON;
* -- Reserved for future lookup tables; keep SP interface stable (add-only).
* SELECT CAST(NULL AS NVARCHAR(MAX)) AS [Value];
* END
* **Output:** One row with [Value] NVARCHAR(MAX) (currently NULL).
* **Evolvability:** Add backends later without changing the SP signature.

**6) Security Model (Users/Roles/Grants)**

**Least privilege:** Application identity can **execute specific SPs only**; it **cannot** SELECT from tables directly. No dynamic SQL inside SPs.

**Recommended pattern (illustrative):**

-- One-time DBA operations (per environment)

-- 1) App login & user (names are examples; actual names managed outside docs)

CREATE USER [mcp\_proxy\_app] FOR LOGIN [mcp\_proxy\_app\_login];

-- 2) Role that can EXEC specific SPs only

CREATE ROLE [app\_executor];

GRANT EXECUTE ON OBJECT::dbo.sp\_Config\_GetValue TO [app\_executor];

GRANT EXECUTE ON OBJECT::dbo.sp\_Config\_GetAll TO [app\_executor];

GRANT EXECUTE ON OBJECT::dbo.sp\_Feature\_IsEnabled TO [app\_executor];

GRANT EXECUTE ON OBJECT::dbo.sp\_Lookup\_Get TO [app\_executor];

-- 3) Assign app user to role

EXEC sp\_addrolemember N'app\_executor', N'mcp\_proxy\_app';

-- 4) DO NOT grant SELECT on tables to the app role/user

**Auditing & Change Control**

* Changes to AppConfig/FeatureFlag occur via **migrations** or controlled admin tooling.
* All releases attach an **Evidence Pack** (OpenAPI diff/lint, CodeQL SARIF, SBOM, secret‑scan summary, monitoring snapshot) and are retained ≥ 1 year.

**7) Performance & Operational Considerations**

* **Isolation level:** READ COMMITTED (default). No explicit transactions required for read‑only SPs.
* **Timeouts:** Application commands default to **30s**; long operations are out of scope for these SPs.
* **Indexes:** Rely on table PKs. Consider additional nonclustered indexes only after profiling.
* **Caching:** App may cache AppConfig values in memory with short TTL to reduce round‑trips; always revalidate on cache expiry.
* **Concurrency:** SPs are read‑optimized; write frequency is low (migrations/admin tools).
* **Logging:** No secrets or PII; log only key names when necessary.

**8) Migrations & Change Governance**

**Add‑only**: never drop/alter destructively. New columns must be nullable or have safe defaults. New SPs are allowed; existing SP signatures remain stable.

**File naming:** /db/migrations/VYYYYMMDDHHMM\_\_<slug>.sql (24‑hour clock, UTC).  
**Ordering:** Apply in lexical order; each migration must be **idempotent** (use existence checks).  
**Example — Initial Schema**

-- /db/migrations/V202509230900\_\_init\_schema.sql

IF NOT EXISTS (SELECT 1 FROM sys.tables WHERE name = 'AppConfig' AND schema\_id = SCHEMA\_ID('dbo'))

BEGIN

CREATE TABLE dbo.AppConfig(

[Key] NVARCHAR(200) NOT NULL PRIMARY KEY,

[Value] NVARCHAR(MAX) NULL,

[UpdatedAt] DATETIME2 NOT NULL CONSTRAINT DF\_AppConfig\_UpdatedAt DEFAULT (SYSUTCDATETIME())

);

END;

IF NOT EXISTS (SELECT 1 FROM sys.tables WHERE name = 'FeatureFlag' AND schema\_id = SCHEMA\_ID('dbo'))

BEGIN

CREATE TABLE dbo.FeatureFlag(

[Name] NVARCHAR(200) NOT NULL PRIMARY KEY,

[Enabled] BIT NOT NULL,

[UpdatedAt] DATETIME2 NOT NULL CONSTRAINT DF\_FeatureFlag\_UpdatedAt DEFAULT (SYSUTCDATETIME())

);

END;

**Example — Seed Values (Non‑Secret)**

-- /db/migrations/V202509230905\_\_seed\_appconfig\_featureflag.sql

MERGE dbo.AppConfig AS T

USING (VALUES

(N'Mcp:ChildCommand', N'npx'),

(N'Mcp:ChildArgs', N'-y @progress/kendo-react-mcp@latest'),

(N'Mcp:ChildCwd', N''),

(N'Security:AllowedOrigins', N'https://chat.openai.com,https://platform.openai.com'),

(N'Network:SseKeepAliveSeconds', N'15'),

(N'Network:RequestTimeoutSeconds', N'120')

) AS S([Key],[Value])

ON T.[Key] = S.[Key]

WHEN NOT MATCHED THEN INSERT([Key],[Value]) VALUES(S.[Key],S.[Value])

WHEN MATCHED AND ISNULL(T.[Value],N'') <> S.[Value] THEN

UPDATE SET T.[Value] = S.[Value], T.[UpdatedAt] = SYSUTCDATETIME();

MERGE dbo.FeatureFlag AS T

USING (VALUES

(N'EnableLegacyHttpSse', 0)

) AS S([Name],[Enabled])

ON T.[Name] = S.[Name]

WHEN NOT MATCHED THEN INSERT([Name],[Enabled]) VALUES(S.[Name],S.[Enabled])

WHEN MATCHED AND T.[Enabled] <> S.[Enabled] THEN

UPDATE SET T.[Enabled] = S.[Enabled], T.[UpdatedAt] = SYSUTCDATETIME();

**Stored Procedures (Idempotent Creates)**

-- /db/stored\_procedures/sp\_Config\_GetValue.sql

CREATE OR ALTER PROCEDURE dbo.sp\_Config\_GetValue

@Key NVARCHAR(200)

AS

BEGIN

SET NOCOUNT ON;

SELECT [Value] FROM dbo.AppConfig WHERE [Key] = @Key;

END;

GO

-- /db/stored\_procedures/sp\_Config\_GetAll.sql

CREATE OR ALTER PROCEDURE dbo.sp\_Config\_GetAll

AS

BEGIN

SET NOCOUNT ON;

SELECT [Key],[Value],[UpdatedAt] FROM dbo.AppConfig ORDER BY [Key];

END;

GO

-- /db/stored\_procedures/sp\_Feature\_IsEnabled.sql

CREATE OR ALTER PROCEDURE dbo.sp\_Feature\_IsEnabled

@Name NVARCHAR(200)

AS

BEGIN

SET NOCOUNT ON;

SELECT CAST(COALESCE((SELECT [Enabled] FROM dbo.FeatureFlag WHERE [Name] = @Name), 0) AS BIT) AS [Enabled];

END;

GO

-- /db/stored\_procedures/sp\_Lookup\_Get.sql

CREATE OR ALTER PROCEDURE dbo.sp\_Lookup\_Get

@Type NVARCHAR(100),

@Key NVARCHAR(200)

AS

BEGIN

SET NOCOUNT ON;

SELECT CAST(NULL AS NVARCHAR(MAX)) AS [Value];

END;

GO

**9) SP‑Only DAL Usage Patterns (C#)**

**.NET 8**; SqlClient; CommandType.StoredProcedure; async + cancellation token; CommandTimeout = 30. **No inline SQL.**

**Example: Get one config value**

using System.Data;

using System.Data.SqlClient;

public static async Task<string?> GetConfigAsync(string key, string cs, CancellationToken ct)

{

await using var conn = new SqlConnection(cs);

await using var cmd = new SqlCommand("dbo.sp\_Config\_GetValue", conn)

{

CommandType = CommandType.StoredProcedure,

CommandTimeout = 30

};

cmd.Parameters.Add(new SqlParameter("@Key", SqlDbType.NVarChar, 200) { Value = key });

await conn.OpenAsync(ct);

var result = await cmd.ExecuteScalarAsync(ct);

return result == DBNull.Value ? null : (string?)result;

}

**Example: Feature flag check**

public static async Task<bool> IsEnabledAsync(string name, string cs, CancellationToken ct)

{

await using var conn = new SqlConnection(cs);

await using var cmd = new SqlCommand("dbo.sp\_Feature\_IsEnabled", conn)

{

CommandType = CommandType.StoredProcedure,

CommandTimeout = 30

};

cmd.Parameters.Add(new SqlParameter("@Name", SqlDbType.NVarChar, 200) { Value = name });

await conn.OpenAsync(ct);

using var rdr = await cmd.ExecuteReaderAsync(ct);

return await rdr.ReadAsync(ct) && rdr.GetBoolean(0);

}

**Timeouts & Types**

* Keep CommandTimeout = 30.
* Map NVARCHAR(200) via SqlDbType.NVarChar, size: 200.
* Handle NULL from sp\_Config\_GetValue by applying application defaults.

**10) Validation & Test Plan**

**SQL Validation (post‑migration)**

1. Tables exist with expected columns & defaults.
2. SPs exist and execute.
3. Seed values present and correct.

**Sample checks**

-- Existence checks

SELECT 1 FROM sys.tables WHERE name='AppConfig' AND schema\_id = SCHEMA\_ID('dbo');

SELECT 1 FROM sys.procedures WHERE name='sp\_Config\_GetValue' AND schema\_id = SCHEMA\_ID('dbo');

-- Seed checks

SELECT [Value] FROM dbo.AppConfig WHERE [Key] = N'Mcp:ChildCommand';

SELECT [Enabled] FROM dbo.FeatureFlag WHERE [Name] = N'EnableLegacyHttpSse';

**Integration Tests (CI)**

* **Config provider:** Fetch known keys; verify types and fallbacks.
* **Flag provider:** Toggle EnableLegacyHttpSse via migration; confirm endpoint behavior changes.
* **Effective config endpoint:** Ensure only **non‑secret** values are returned; confirm redaction policy.

**Performance Sanity**

* SP round‑trip p50/p95 under nominal load (local VPC) meets NFR budgets.

**11) Examples & Quick Verification Queries**

**List all config**

EXEC dbo.sp\_Config\_GetAll;

**Get one value**

DECLARE @v NVARCHAR(MAX);

EXEC dbo.sp\_Config\_GetValue @Key = N'Network:SseKeepAliveSeconds';

**Is legacy enabled?**

EXEC dbo.sp\_Feature\_IsEnabled @Name = N'EnableLegacyHttpSse';

**Ad‑hoc inspection (DBA only)**

SELECT TOP 50 \* FROM dbo.AppConfig ORDER BY [UpdatedAt] DESC;

SELECT TOP 50 \* FROM dbo.FeatureFlag ORDER BY [UpdatedAt] DESC;

**12) Assumptions & Next Steps**

**Assumptions**

1. The service connects with a DB principal granted **EXECUTE** on listed SPs only.
2. Environments (Alpha → Beta → RTM → Prod) may override non‑secret values via controlled migrations. RTM validates against **Prod DB** for parity.
3. Ingress supports SSE; the app caches config with short TTL but treats SPs as source of truth.

**Next Steps**

1. Apply the initial schema & seed migrations; verify SPs.
2. Wire the application config provider to sp\_Config\_GetValue/sp\_Config\_GetAll; add memory cache w/ TTL.
3. Enforce role‑based grants (EXEC on SPs only); verify app principal lacks table SELECT.
4. Add release evidence (migration logs, test results) to the Evidence Pack and retain ≥ 1 year.

**Footer (optional for Word header/footer):**  
*MCPX‑KendoBridge • Data & DB Contracts • Version 1.0.0 (Draft) • 2025‑09‑23 • Confidential — Technijian Internal*