

Functional Requirements Specification: MCP-Fess Bridge Server

1 Introduction

This document defines the functional requirements for a **Model Context Protocol (MCP) bridge server** that wraps a running **Fess** search server. The bridge exposes Fess functionality to agentic AI systems through the Model Context Protocol, enabling Large Language Models (LLMs) to query and retrieve domain-specific information using structured tools and resources. It supports two transport layers (stdio and HTTP) and is configurable via a JSON file in the user's home directory. Requirements are aligned with the Fess REST API documentation and the MCP 2025-03-26 specification.

1.1 Scope

- **MCP-Fess Bridge Server (the server)** provides an MCP server implementation that proxies requests to a Fess server via its REST API. It implements MCP server features (tools and resources) and supports progress, cancellation and chunked payloads.
- **Fess server:** a running Fess instance exposed via HTTP. The bridge does not manage Fess indexing or authentication beyond proxy configuration.
- **Users:** agentic AI hosts and LLMs consume the bridge via MCP. The host may forward metadata to the LLM; therefore, domain information must be present in tool/resource descriptions.

1.2 Definitions

Term	Description
Fess	Open-source enterprise search server. Its Search API endpoint <code>/api/v1/documents</code> accepts query parameters (e.g., <code>q</code> , <code>start</code> , <code>num</code> , <code>sort</code> , <code>fields.label</code> , <code>facets</code> , <code>geo</code> , <code>lang</code>) and returns JSON results ¹ . <code>/api/v1/documents/all</code> returns scrolling search results when <code>api.search.scroll=true</code> ² . Label API <code>/api/v1/labels</code> returns available labels ³ . Popular Words API <code>/api/v1/popular-words</code> returns trending words and accepts <code>seed</code> , <code>label</code> and <code>field</code> parameters ⁴ . Suggest API <code>/api/v1/suggest-words</code> provides suggestions based on <code>q</code> , <code>num</code> , <code>label</code> , <code>fields</code> , and <code>lang</code> ⁵ . Health API <code>/api/v1/health</code> returns system status (green/yellow/red) ⁶ .
MCP	Model Context Protocol. Provides JSON-RPC-based lifecycle, transports and utilities (progress, cancellation) for exposing tools and resources. Progress notifications allow servers to report progress via a <code>progressToken</code> ⁷ . Cancellation notifications allow clients to cancel in-flight requests ⁸ . The resources feature defines list and read operations with pagination ⁹ , and the tools feature defines listing and invocation semantics ¹⁰ .
Domain	A knowledge domain served by the bridge. Each domain corresponds to a Fess label; the bridge filters search requests using <code>fields.label=<label></code> ¹¹ . Multiple bridge instances can share one Fess server but operate on different domains.

Term	Description
Tool	An MCP tool exposed by the bridge. Tools provide Fess functions (search, suggest, popular words, label listing, health) to the host/LLM. Tool descriptions include a Knowledge Domain block (see §7.3) so the LLM can select the appropriate domain.
Resource	An MCP resource representing Fess documents. Resources are identified by custom URIs (<code>fess://<domainId>/doc/<docId></code>). Resource listing and reading support pagination and chunked responses ¹² .

2 References

1. Fess API Guide, Search API – request parameters and response structure ¹ .
2. Fess API Guide, Searching All Documents – scroll search requires enabling `api.search.scroll` ² .
3. Fess API Guide, Label API – endpoint and response format ³ .
4. Fess API Guide, Popular Words API – endpoint and parameters ⁴ .
5. Fess API Guide, Suggest API – endpoint and parameters ⁵ .
6. Fess API Guide, Health API – endpoint and status values ⁶ .
7. MCP Specification (2025-03-26), progress notifications ⁷ .
8. MCP Specification (2025-03-26), cancellation notifications ⁸ .
9. MCP Specification, resources capability and pagination ⁹ ¹² .
10. MCP Specification, tools capability and listing semantics ¹⁰ ¹³ .

3 System Overview

The bridge sits between an MCP client (agentic AI) and a Fess server. It translates MCP tool calls into HTTP requests to the Fess REST API and translates Fess responses back into MCP tool results or resource contents. The bridge supports both **stdio** and **HTTP** transports for the MCP JSON-RPC layer. All configuration is read from a JSON file located at `<USER_HOME_DIR>/mcp-feiss/config.json`. Logging is written to `<USER_HOME_DIR>/mcp-feiss/log/`.

Domains are configured via `config.json`. Each domain corresponds to a Fess label; search tools automatically apply `fields.label=<label>` ¹¹ . Multiple bridge instances may run simultaneously, each with a different domain configuration, enabling fine-grained knowledge domains on a shared Fess server.

4 Configuration & Runtime

4.1 Config File

The bridge **MUST** read configuration from `<USER_HOME_DIR>/mcp-feiss/config.json`. Missing or invalid configuration causes a startup failure with a clear error message. On startup, the bridge **MUST** create `<USER_HOME_DIR>/mcp-feiss/log/` if it does not exist. The configuration schema includes:

Field	Type	Description
<code>fessBaseUrl</code>	string (required)	Base URL of the Fess server, e.g., <code>http://localhost:8080</code> .

Field	Type	Description
<code>domain</code>	object (required)	Defines the knowledge domain. Contains: <code>id</code> (string, unique identifier), <code>name</code> (string, human-readable), <code>description</code> (string, optional), <code>labelFilter</code> (string, Fess label value). Searches automatically include <code>fields.label=<labelFilter></code> ¹¹ .
<code>httpTransport</code>	object (optional)	HTTP transport configuration: <code>bindAddress</code> (default <code>127.0.0.1</code>), <code>port</code> (default <code>0</code> meaning auto), <code>path</code> (default <code>/mcp</code>), <code>enableSse</code> (boolean, default <code>true</code>). A separate <code>security.allowNonLocalhostBind</code> flag (default <code>false</code>) controls whether non-loopback addresses may be bound.
<code>timeouts</code>	object (optional)	<code>fessRequestTimeoutMs</code> (default 30000), <code>longRunningThresholdMs</code> (default 2000). If a Fess request exceeds <code>longRunningThresholdMs</code> , it is treated as long-running (see §9).
<code>limits</code>	object (optional)	<code>maxPageSize</code> (default 100 – Fess permits up to 100 results per page ¹⁴), <code>maxChunkBytes</code> (default 262144), <code>maxInFlightRequests</code> (default 32).
<code>logging</code>	object (optional)	<code>level</code> (<code>error</code> / <code>warn</code> / <code>info</code> / <code>debug</code>), <code>retainDays</code> (default 7).
<code>security</code>	object (optional)	<code>httpAuthToken</code> (string; if set, HTTP requests require <code>Authorization: Bearer <token></code>), <code>allowNonLocalhostBind</code> (boolean; if false and <code>bindAddress</code> is not loopback, startup fails).
<code>contentFetch</code>	object (optional)	Controls full-content fetching: <code>enabled</code> (default <code>true</code>), <code>maxBytes</code> (default 5 MiB), <code>timeoutMs</code> (default 20000), <code>allowedSchemes</code> (default <code>["http", "https"]</code>), <code>allowPrivateNetworkTargets</code> (default <code>false</code>), <code>allowedHostAllowlist</code> (array, optional), <code>userAgent</code> (string), <code>enablePdf</code> (boolean, default <code>false</code>).

4.2 CLI Flags

- `--transport <stdio|http>` – optional; selects the MCP transport. Default is `stdio`. In HTTP mode, the server binds to `bindAddress:port` and serves MCP over HTTP at `path`.
- `--debug` – enables verbose logging. When set, logs are written to `<Date and Time>_server.log` in the log directory. Each line MUST be prefixed with `[HH:MM:SS]` representing elapsed time since server start.
- `--cody` – forces the server to implement MCP revision **2024-11-05** instead of the default **2025-03-26**.

4.3 Logging

- **Default logging:** logs written to a stable file (e.g. `server.log`) in `<USER_HOME_DIR>/mcp-feiss/log/`. Log rotation may be based on size or time.

- **Debug logging:** when `--debug` is provided, verbose logs go to `<Date and Time>_server.log` with each line prefaced by `[HH:MM:SS]` (elapsed). The directory and file are created on startup.

4.4 Startup Errors

- If `config.json` is missing or invalid, the server MUST exit with a non-zero status and an error describing the issue.
- If `fessBaseUrl` is unreachable during startup, the server MAY continue (requests will fail individually) but SHOULD log a warning.
- If `httpTransport.bindAddress` is not loopback (`127.0.0.1` or `::1`) and `security.allowNonLocalhostBind` is false, the server MUST fail at startup.

5 Protocol Versioning and Lifecycle

5.1 Supported Revisions

- **Default:** The server implements MCP revision **2025-03-26**. The `initialize` response MUST advertise this revision and the server MUST abide by all required behaviors (tools/resources, progress, cancellation, etc.).
- **Cody mode:** When started with `--cody`, the server implements MCP revision **2024-11-05**; `initialize` MUST advertise that revision.

5.2 Initialize Flow

The server MUST follow the MCP lifecycle. It waits for an `initialize` request, responds with server capabilities and metadata, and then waits for `notifications/initialized` before accepting ordinary requests. The server MUST expose capabilities for **tools** and **resources** (see §7–§8) using the structures defined in the MCP specification ⁹ ¹⁰.

5.3 Capabilities Metadata

Capabilities MUST declare support for tools and resources with `listChanged` optional flags. Even though the host may not forward arbitrary metadata to the LLM, the server MUST include domain information in capabilities for completeness.

6 Transports

6.1 Stdio Transport

When `--transport` is `stdio` (default), the server reads MCP JSON-RPC messages from standard input and writes responses to standard output. The server supports exactly one connected client at a time.

6.2 HTTP Transport

When `--transport` is `http`, the server exposes an MCP endpoint at `http://<bindAddress>:<port><path>`. The server MUST support concurrent sessions and optionally Server-Sent Events (SSE) if `enableSse=true`. If `security.httpAuthToken` is set, the server MUST require a bearer token in the `Authorization` header; unauthenticated requests MUST return an HTTP 401 error.

The default binding MUST be loopback only (`127.0.0.1` or `::1`). Binding to other addresses is only allowed when `security.allowNonLocalhostBind=true` .

7 Domain Behaviour and Tools

7.1 Automatic Domain Filtering

For all Fess search-related API calls, the bridge MUST automatically add `fields.label=<domain.labelFilter>` to the request ¹¹ . For popular words and suggest APIs, the bridge MUST set the `label` parameter to the domain's label when present ⁴ ⁵ . This ensures each bridge instance operates exclusively on its configured knowledge domain.

7.2 Tool Naming Convention

Tool names MUST incorporate the domain ID to avoid collisions across multiple bridge servers. For example, if the domain ID is `finance` , the search tool name could be `fess_finance_search` . All tool names MUST be lower-case, use underscores to separate words, and be unique within the server.

7.3 Knowledge Domain Block

Because agentic hosts may pass only tool descriptions to the LLM, the server MUST include domain information in every tool description and in resource descriptions. The description MUST contain a **Knowledge Domain block** as follows:

```
[Knowledge Domain]
id: <domain.id>
name: <domain.name>
description: <domain.description>
fessLabel: <domain.labelFilter>
```

This block MUST appear verbatim in the description of every tool and resource and MUST be identical across all tools/resources in the same server. It enables the LLM to associate tools with the appropriate knowledge domain.

7.4 Exposed Tools

At minimum, the server MUST expose the following tools. Each tool's description MUST include the Knowledge Domain block and clearly describe its purpose. Tools MUST support pagination where appropriate by accepting `pageSize` (\leq `limits.maxPageSize`) and `start` parameters.

1. **Search Tool** (`fess_<domainId>_search`)
2. **Description:** "Search the knowledge domain for documents matching a query." Includes the Knowledge Domain block.
3. **Input Schema:** An object with the following fields:
 - `query` (string, required): search term; maps to Fess `q` ¹ .
 - `pageSize` (integer, optional): number of results (default 20, \leq 100) ¹⁴ .
 - `start` (integer, optional): starting index (default 0) ¹⁵ .
 - `sort` (string, optional): sort order ¹⁶ .
 - `lang` (string, optional): search language ¹⁷ .

- `facets` (object, optional): maps to Fess `facet.field`, `facet.query`, `facet.size`, `facet.minDocCount` ¹⁸.
 - `geo` (object, optional): `point` (lat,long) and `distance` (e.g., "10km") ¹⁹.
 - `includeFields` (array of strings, optional): list of result fields to include (title, url, digest, doc_id, content_length, etc.).
4. **Behavior:** The bridge sends a GET request to `/api/v1/documents` with the above parameters and `fields.label=<domain.labelFilter>` ¹¹. Results are returned as an array of summaries (title, url, digest, doc_id, score if available). Fess returns up to 100 results per request ¹⁴.
5. **Progress/Jobs:** If the underlying call exceeds `longRunningThresholdMs`, the server MUST treat it as a job (see §9) and emit progress notifications.
6. **Suggest Tool** (`fess_<domainId>_suggest`)
7. **Description:** "Suggest related terms for a query in the knowledge domain." Includes the Knowledge Domain block.
8. **Input Schema:**
- `prefix` (string, required): maps to Fess `q` ⁵.
 - `num` (integer, optional): number of suggestions (default 10) ²⁰.
 - `fields` (array of strings, optional): `fields` parameter ²¹.
 - `lang` (string, optional): language `【677245133564858†L118-L117】`.
9. **Behavior:** GET `/api/v1/suggest-words` with `q=prefix`, `label=<domain.labelFilter>`, plus other parameters ⁵. Returns suggestions.
10. **Popular Words Tool** (`fess_<domainId>_popular_words`)
11. **Description:** "Retrieve popular words in the knowledge domain." Includes the Knowledge Domain block.
12. **Input Schema:**
- `seed` (integer, optional) ⁴.
 - `field` (string, optional) ²².
13. **Behavior:** GET `/api/v1/popular-words` with `seed`, `field`, and `label=<domain.labelFilter>` ⁴. Returns list of words.
14. **Labels Tool** (`fess_<domainId>_list_labels`)
15. **Description:** "List all labels configured in the underlying Fess server." Includes the Knowledge Domain block. Useful for diagnostics.
16. **Input Schema:** none.
17. **Behavior:** GET `/api/v1/labels` ³. Returns all labels (not restricted to the domain). The domain filter is not applied here.
18. **Health Tool** (`fess_<domainId>_health`)
19. **Description:** "Check the health status of the underlying Fess server." Includes the Knowledge Domain block.
20. **Input Schema:** none.

21. **Behavior:** GET `/api/v1/health` and return the `status` and `timed_out` fields ⁶.
22. **Job Progress Tool** (`fess_<domainId>_job_get`)
23. **Description:** "Retrieve progress information for a long-running operation." Includes the Knowledge Domain block.
24. **Input Schema:** `jobId` (string, required).
25. **Behavior:** Returns `{ state, progress, total?, message?, startedAt, updatedAt }`. See §9.

7.5 Input Validation

The server MUST validate tool parameters and return MCP errors with clear messages when invalid (e.g., `pageSize` exceeds `maxPageSize`, missing required fields).

7.6 Domain Metadata in Server Info

In addition to tool descriptions, the `initialize` response's `serverInfo` MUST include `domain.id`, `domain.name`, `domain.description` and `domain.labelFilter` fields. This metadata may not reach the LLM but aids debugging.

8 Resources

8.1 Resource URIs

The bridge MUST expose Fess documents as resources using a custom URI scheme:

- `fess://<domainId>/doc/<docId>` – represents a document with `docId` returned by Fess.
- `fess://<domainId>/doc/<docId>/content` – represents the full content (text) of the document.

8.2 Resource Listing

Clients can list available resources via `resources/list`. The server MUST implement pagination using cursors as per the MCP specification ¹². Each resource in the list MUST include:

- `uri` – unique identifier (see above).
- `name` – usually the document title.
- `title` – same as name.
- `description` – MUST include the Knowledge Domain block and a short summary.
- `mimeType` – `text/plain` for content resources.

8.3 Resource Reading

Clients read resources via `resources/read`. For document metadata, the server returns the Fess summary (title, url, digest, doc_id, timestamps). For `.../content`, the server returns the document's text content. The response MUST follow the MCP schema for resources (contents array with `uri`, `mimeType` and `text`) ²³.

8.4 Chunked Content

If the content length exceeds `limits.maxChunkBytes`, the server MUST split the response into chunks. The first `resources/read` call returns the first chunk plus a `nextCursor`. Subsequent calls with that cursor return the next chunk until `isLast=true`. Chunks MUST be sequential and non-overlapping.

8.5 Content Acquisition

The server MUST provide full document text according to the following strategy:

1. **Index content:** If Fess returns the document's body or digest fields with sufficient length, the server SHOULD use them as the content.
2. **Fetch:** Otherwise, the server MUST fetch the document from its `url` using HTTP GET. The server MUST respect `contentFetch.maxBytes`, `contentFetch.timeoutMs`, `allowedSchemes`, and host restrictions. Private network targets (RFC1918, localhost) MUST be blocked unless `allowPrivateNetworkTargets=true` and/or the host is in `allowedHostAllowlist`.
3. **Conversion:** If the retrieved content is HTML, the server MUST strip tags and preserve paragraph breaks to produce plain text. If the content is plain text, return it directly. If the content is PDF and `contentFetch.enablePdf=true`, the server MUST convert it to text; otherwise, the server MUST return an error stating PDF conversion is disabled.
4. **Hash:** The server MUST compute a stable hash of the final content and include it in resource metadata. This aids in verifying chunk consistency.

8.6 Resource Identity

The server MUST expose `doc_id`, `url`, and `hash` fields in resource metadata. `doc_id` is the Fess document ID; `url` is the canonical URL; `hash` is computed from the fetched content.

9 Long-Running Operations & Progress

9.1 Long-Running Threshold

If a Fess API call or content fetch is expected to exceed `timeouts.longRunningThresholdMs`, the bridge MUST treat it as a long-running operation. The server MUST immediately respond with a job handle containing `jobId` and start processing asynchronously.

9.2 Progress Notifications

If the client provided a `progressToken`, the server MUST send `notifications/progress` messages including the token, progress value, total (if known) and human-readable message ⁷. Progress values MUST monotonically increase. The phases MAY include: `queued`, `fetching`, `converting`, `chunking`, `done`.

9.3 Job Status Queries

The `job_get` tool MUST allow clients to poll for job status. It returns `state` (queued, running, done, failed, cancelled), `progress`, `total`, `message`, `startedAt`, and `updatedAt`.

9.4 Cancellation

The server MUST implement `notifications/cancelled` according to the MCP specification ⁸. Upon receiving a cancellation notification for a request, the server MUST attempt to cancel the underlying Fess call or content fetch. If cancellation succeeds, the server MUST free resources and MUST NOT send a final response for the cancelled request ²⁴. Cancellation of the `initialize` request is not permitted ²⁵.

10 Error Handling

- Errors from Fess (HTTP 4xx/5xx) MUST be translated into MCP JSON-RPC errors. The error object MUST include an internal code, the HTTP status, the Fess endpoint, and a sanitized message.
- Invalid tool parameters MUST result in an MCP error with a clear message.
- Timeouts MUST produce either an MCP error (for synchronous requests) or a job failure (for asynchronous operations) with a reason.
- If content fetch is disallowed by scheme or host restrictions, the server MUST return an error indicating the specific restriction.
- PDF conversion attempts when disabled MUST produce an error explaining that PDF support is turned off.

11 Security Considerations

- The bridge MUST not expose Fess credentials or internal URLs.
- When using HTTP transport, the default binding is loopback only. Non-localhost binding requires explicit configuration (`allowNonLocalhostBind=true`) to mitigate exposure.
- If `httpAuthToken` is configured, the server MUST require a bearer token for all HTTP requests and reject unauthorized requests.
- Content fetching MUST respect `allowPrivateNetworkTargets` and host allowlists to prevent server-side request forgery.

12 Performance and Limits

- The server MUST enforce `limits.maxInFlightRequests`. Additional requests should be queued or rejected with a retryable error.
- The server MUST enforce `maxPageSize` for search results and `maxChunkBytes` for resource chunks.
- The server SHOULD trim unused fields from Fess responses when `includeFields` is provided to reduce payload size.

13 Extensibility

The server design should allow additional Fess endpoints to be exposed as tools in the future (e.g., administrative APIs) without breaking existing clients. Tools and resources MUST remain versioned via the domain ID to avoid collisions. Additional features such as summarization or topic clustering may be implemented as higher-level tools built atop the basic search and content fetch functions.

3 Label API

<https://fess.codelibs.org/15.4/api/api-label.html>

4 22 Popular Words API

<https://fess.codelibs.org/15.4/api/api-popularword.html>

5 20 21 Suggest API

<https://fess.codelibs.org/15.4/api/api-suggest.html>

6 Health API

<https://fess.codelibs.org/15.4/api/api-health.html>

7 Progress - Model Context Protocol

<https://modelcontextprotocol.io/specification/draft/basic/utilities/progress>

8 24 25 Cancellation - Model Context Protocol

<https://modelcontextprotocol.io/specification/draft/basic/utilities/cancellation>

9 12 23 Resources - Model Context Protocol

<https://modelcontextprotocol.io/specification/draft/server/resources>

10 13 Tools - Model Context Protocol

<https://modelcontextprotocol.io/specification/draft/server/tools>