FiveFold

TECHNICAL LIBRARY / GUIDE 02 / OCTOBER 2026

# MORIAH

## Technical Architecture & Capabilities Guide

Architecture, capabilities, connections and operating controls

by FiveFold Ventures

Built for substantial work: not a separate foundation model, not a single-purpose bot, and not simply another place to type prompts.

A practical technical orientation for software architects, engineering leaders, CIOs, CTOs, operators and integration teams assessing an existing AI operating foundation.

THE CENTRAL DESIGN CHOICE

The model reasons. Moriah manages the authorized environment in which that reasoning becomes work.

### What this guide covers

| FOUNDATION | OPERATING SURFACES | EXTENSION |
|---|---|---|
| **Model and provider abstraction** | Ask, sources, documents, outputs | Capabilities, permissioned tools |
| **Identity, scope and authority** | Active Threads, saved work, preferences | Microsoft 365 and other connectors |
| **Source and evidence governance** | Organization and admin controls | Configuration, support, validation |

A working technical foundation with documented acceptance in specific workflows. Customer tenant setup and regulated or high-impact actions are evaluated separately.

01 / EXECUTIVE ORIENTATION

# Why Moriah exists

The hard part of business AI is often not obtaining a capable answer. It is carrying the surrounding work forward without losing its authority, sources, identity or history.

Moriah began as a practical personal need. Its founder was using AI for inventions, research, business opportunities and ongoing projects. Strong model output still left context fragmented among conversations, documents, tools and follow-up tasks. The original ambition was a dependable second brain; the engineering expanded into a working environment with project boundaries, permissions, source handling, durable work and operating controls.

That origin matters technically. Instead of placing every responsibility inside a large prompt or assuming the model owns the state, Moriah separates the parts of a working system that must remain answerable to the user or organization.

### Two distinct responsibilities

| REASONING MODEL | MORIAH APPLICATION ENVIRONMENT |
|---|---|
| **Interprets intent; reasons, compares, analyzes and synthesizes.** | Determines permitted context, identity, work scope, source eligibility and available capabilities. |
| **Drafts useful prose, structured results and model-generated outputs.** | Handles source/use status, explicit saving, artifact identity and work continuity. |
| **May recommend or request an action.** | Checks policy, credentials, required consent and supported executor before an action can occur. |

### The product, at a glance

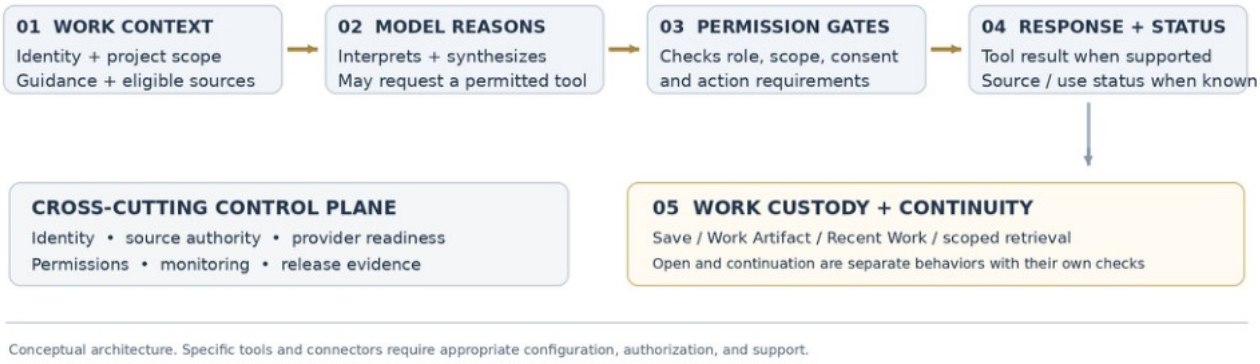
| DOMAIN | WHAT EXISTS IN THE FOUNDATION |
|---|---|
| **Work cockpit** | Ask, response, Active Thread, source and recent-work surfaces. |
| **Information** | Implemented multi-file intake and extraction, bounded scanned-PDF OCR and URL-to-Ask lineage, selected-source controls and source cards. |
| **Documents and outputs** | Model-owned Quick Outputs, saved responses and supported document/artifact workflows. |
| **Governance** | Authenticated user/project access controls, scoped memory and admin surfaces; formal cross-org isolation separately gated. |
| **Integration** | Read-only governed tool execution, accepted Microsoft 365 connected-source workflows, provider adapters and permissioned action contracts. |
| **Operations** | Support/diagnostic surfaces, documentation, release discipline and bounded acceptance evidence. |

02 / ARCHITECTURE

# How the system is assembled

Reasoning, operating authority, data, execution and retained work have distinct responsibilities.

### MODEL REASONS / APPLICATION GOVERNS




Accessible diagram summary: Authorized project/work context and eligible sources → model reasoning → permission gates → response and status → explicit work custody and continuity. Identity, source authority, provider readiness, permissions, monitoring and release evidence govern the flow across those stages.

Figure 1. Model reasoning and application-owned authority remain separate.

The cockpit is only the entry point. Application controls resolve identity and work scope, source eligibility, model routing and permissioned tools; results are not silently promoted into verified evidence or durable memory.

## Seven cooperating planes

| PLANE | SYSTEM RESPONSIBILITY |
|---|---|
| **Cockpit / experience** | Ask, Response, Active Thread, document and source interaction, operator and admin surfaces. |
| **Identity / scope** | Users, memberships, organizations, projects, roles and privileged boundaries. |
| **Source / evidence** | Admission, readability, selection/binding, provenance and source-use meaning. |
| **Reasoning / providers** | Invocation abstractions, configured model endpoints and task-appropriate reasoning. |
| **Capability / action** | Available tool contracts, auth and policy checks, execution adapters and result handling. |
| **Continuity / artifacts** | Durable Active Thread Save and Resume, work identity, discovery, governed reuse, and distinct saved-response readback. |
| **Operations / proof** | Health, diagnostics, provenance, controlled deployment, reproducible verification and support. |

03 / WORKING STATE

# From a question to a durable work product

Moriah is organized around the lifecycle of the work, not merely the transcript of a conversation.

## The working sequence

1 Ask. An authorized user begins in a scoped workspace or thread and asks a question or requests a deliverable.

2 Establish context. The application evaluates the acting identity, available project guidance, source state, appropriate model/provider and permitted tools.

3 Reason and use evidence. A model reasons using the authorized information provided. Where an eligible tool is involved, the application governs the invocation and handles results.

4 Deliver a result. Response may contain analysis, a document draft, a decision brief, next actions, or another useful output. Source/use status is kept distinct from the prose.

5 Save the work. Moriah supports durable Active Thread saving and explicit response capture. Active Threads, saved responses, and uploaded documents have distinct identities and life cycles, rather than becoming an undifferentiated transcript.

6 Resume and continue. Accepted Active Thread journeys have demonstrated Save, Recent Work, Resume and subsequent work. Moriah makes retained work a usable part of the operating experience. Opening a separately saved response in a particular viewer, sharing across users, and restoring changed source access remain distinct acceptance contracts.

### What is preserved—and what is not confused

| STATE TYPE | WHY IT IS SEPARATE |
|---|---|
| **Active Thread** | Current conversational work and immediate bindings; a New Thread is a meaningful boundary. |
| **Source record** | A document, selected URL or connected item with independent identity, availability and access status. |
| **Saved artifact** | An explicitly persisted response or output with its own work identity and lifecycle. |
| **Preference / guidance** | Scoped instruction about how work should be handled; not evidence that an external fact is true. |
| **Organization / project scope** | A permission and operating context; not automatic access to every colleague's material. |
| **Evidence record** | What the system has actually sourced, executed or demonstrated; not merely a plausible model assertion. |

A strong continuity architecture treats the work record as a governed object. It is more than pasting yesterday's chat history into today's prompt: identity, source relationships, permission and an actual permitted retrieval path all matter.

04 / REASONING AND PROVIDERS

# Models are engines, not the workplace

Moriah was designed to use model capability while retaining independent responsibility for work context and governance.

Moriah does not train or replace a foundation model. A configured reasoning service supplies semantic intelligence. The Moriah application supplies legitimate context, invokes the supported provider path, manages capability access and receives the result.

### Provider and capability are different things

A capability describes the work requested—reasoning, research, document processing, connected- source retrieval or artifact creation. A provider is an engine or service that may satisfy it. Separating those concepts helps maintain one product rather than multiple vendor-specific assistants.

- Provider adapters. Common invocation contracts and OpenAI, Anthropic and Gemini adapters are implemented. Selected provider-policy and organization-owned OpenAI-key paths passed bounded runtime acceptance; active routes depend on configuration.
- Reasoning fit. Not every task needs the same model or reasoning effort. Model choice, response length, availability, provider policy and usage cost are legitimate operational considerations.
- Credentials and organizational control. Authenticated organization-owned provider-key use and selected-source policy binding have passed acceptance. Admin grants, per-provider readiness, billing terms and permitted use rights remain separately configured.
- Provider-independent work state. Project context, source identity and saved artifacts belong to the Moriah operating system design rather than to a model vendor's chat transcript.

### A provider change is not magic

The architecture anticipates changes in model availability and approved configurations. It should not be interpreted as automatic cross-provider failover, universal bring-your-own-key support, equal behavior from all vendors or seamless switching in every deployed workflow. Those behaviors have their own implementation and acceptance requirements.

The value proposition is not simply access to several models—an increasingly common feature. It is governing model use inside a coherent work, authority, source and artifact environment.

05 / INFORMATION AND EVIDENCE

# Documents, sources and the meaning of "grounded"

Moriah keeps source identity, admission, selection and supported source-use status distinct rather than burying every document inside a prompt.

### Document and source families

- Uploads and Inbox. Implemented multi-file upload, admission, extraction and follow-up source-use pathways have passed bounded tests. Supported file types and parser setup remain relevant.
- Document formats. Word/PDF extraction, selected spreadsheet reading, scanned-PDF OCR and bounded document/presentation exports have accepted evidence. Arbitrary workbook editing and perfect Office-format fidelity are separate capabilities.
- Current and public research. URL extraction and Firecrawl-to-Ask source lineage passed bounded authenticated tests. Current search/retrieval operation depends on the configured provider and access rights.
- Connected business sources. Read-only Microsoft 365 workflows have retrieved Outlook messages and OneDrive documents, bound selected content as a source and used it in model reasoning in accepted scenarios.
- Saved work as source. A prior artifact may become context through an approved workflow, but its status as previous model output remains distinct from independent documentary evidence.

### A source has multiple states

| SOURCE STATE | WHAT IT MEANS |
|---|---|
| **Available / candidate** | The system knows about an item, but may not have permission or readable content. |
| **Admitted / readable** | The supported content is eligible to be read within approved scope. |
| **Selected / bound** | A source has been explicitly attached or made available to the current work. |

| SOURCE STATE | WHAT IT MEANS |
|---|---|
| **Actually used** | The particular response is supported by recorded or verifiable source use; selection alone does not prove causality. |
| **Saved reference / later reread** | Retained identity can persist even when fresh retrieval updates the external content. |

These distinctions matter for engineering teams, researchers, operations managers and executives: a convincing answer must not be mistaken for a verified source conclusion. Moriah is built around preserving the difference between available information, the evidence that was actually used, and conclusions that still require human validation.

06 / INTEGRATIONS

# How tools and external systems fit together

The foundation is meant to absorb additional capabilities without each integration inventing its own identity, permission and work-state model.

## The governed tool pattern

Moriah has an implemented provider-neutral read-only tool registry with policy gates, authorized execution and structured results, demonstrated in bounded runtime paths. Each capability has a defined purpose, request shape, access policy and executor. The model may propose use; Moriah decides whether the action is permitted.

| STEP | ENGINEERING QUESTION |
|---|---|
| **1 Declare capability** | What action or read operation exists? What arguments and result types are valid? |
| **2 Resolve authority** | Which organization, user, role, source and resource permit this request? |
| **3 Check provider/connection** | Does the selected connector have valid consent, credentials and supported API availability? |
| **4 Confirm consequential actions** | Does this operation change external state, send a message or move money? Is explicit approval required? |
| **5 Execute through an adapter** | Can a configured executor perform the bounded operation and report its result or error? |
| **6 Return usable status** | What succeeded, failed or needs retry? What source was read or record changed? |
| **7 Retain only as authorized** | Should this result become a work artifact, and under whose scope and retention rules? |

## What is already reusable

Moriah has already implemented the shared identity, Ask, source, artifact and read-only capability-execution foundation. An additional integration can reuse those operating controls rather than recreating a separate assistant, while its specific adapter and permissions remain explicit engineering work.

A new adapter may require engineering; an existing adapter may only need tenant consent, licensing or configuration. Neither is equivalent to building Moriah from scratch.

07 / CONNECTED ENTERPRISE SYSTEMS

# Microsoft 365: demonstrated read-only connectivity

Accepted Outlook and OneDrive read workflows demonstrate real connected-source operation, not merely a proposed connector. Moriah remains independent of Microsoft as its operating foundation.

Microsoft Graph connectivity uses organization-specific Entra configuration and delegated permissions. Connect/reconnect, encrypted durable refresh, guarded source cards and bounded read operations passed earlier authenticated tests. Each new tenant still needs its own consent and authorized account access.

### The connection responsibilities

| SURFACE / CAPABILITY | HOW IT FITS |
|---|---|
| **OneDrive and SharePoint** | OneDrive discovery of live Word/PDF/spreadsheet content and selected Word/PDF read-and-reason workflows passed browser proof. SharePoint belongs to the routed family; site/list coverage is separately assessed. |
| **Outlook / email sources** | An entitled Outlook read-only workflow found mail, read the authorized message body, bound it as a source and produced an AI summary. User consent and mailbox access still control availability. |
| **Email send / calendar updates** | Separate write operations, not an extension of read permission. Email sending and calendar changes require a specifically implemented, authorized and tested executor plus Graph write scopes. |
| **Teams and meetings** | Read-only calendar assistance and selected meeting/email artifact reasoning have historical acceptance. Live Teams transcript retrieval and outgoing Teams actions are not established by that evidence. |
| **Jira, CRM, tracking and other tools** | Integrate by supported APIs, approved imports or purpose-built adapters; records remain subject to system-of-record authority. |

### Proven reading is distinct from permission to write

Accepted read-only M365 workflows located and summarized authorized Outlook email and OneDrive documents. These functions do not permit the model to send email, alter records or access another user's files.

08 / PRODUCT SURFACES

# What users and teams actually operate

A substantial system also needs usable surfaces: the things a person can see, control, create, save and revisit.

### Everyday working surfaces

| SURFACE | PURPOSE |
|---|---|
| **Ask and Response** | Reasoning, grounded review where supported, refinement and follow-up. |
| **Active Thread** | Current working exchange, with its own distinct source and context boundaries. |
| **Sources / Connections** | Select admitted documents, inspect supported source state and manage connections. |
| **Upload / Inbox** | Bring supported files into the environment for review, handling and later use. |

| SURFACE | PURPOSE |
|---|---|
| **Recent Work / saved artifacts** | Capture useful outputs and locate eligible saved material through governed discovery. |
| **Customize / Preferences** | Review and manage supported personal, project or organization operating guidance. |
| **Admin / Capabilities** | Inspect supported organization membership, readiness, approved connections and available functions. |
| **Session / Support** | Collect bounded operational feedback, safe diagnostics and support context. |

### Five model-generated Quick Outputs

Moriah includes five model-owned output types for common work products: Decision Brief, Next Actions, Checklist, Meeting Recap, and Proposal/Scope of Work. The system uses the reasoning model to prepare content rather than presenting a fixed answer bank. Output density and additional guidance can shape a result within supported workflow boundaries.

Other historically accepted paths include document extraction, scanned-PDF OCR, selected Excel content handling and bounded Word/PDF/presentation outputs. Voice input and read-aloud also have historical implementation evidence. These capabilities do not imply unrestricted editing of arbitrary files.

09 / ADMINISTRATION AND SECURITY

# Where authority lives

Moriah is designed around application- and server-owned authority rather than assuming that a well-phrased prompt can grant access.

### Different roles, different controls

- Platform Owner / Admin. Operates platform-level capabilities and specific privileged responsibilities under controlled procedures.
- Organization Admin. Manages authorized organization-level memberships, selected connection/capability posture, and operating preferences within its granted scope.
- Organization member. Uses the permitted cockpit and sources; membership does not automatically confer administrative access or another member's document rights.
- User connections. Personal delegated credentials are separate from organization-wide configuration; a user's service consent is not a universal tenant permission.

### Trust boundaries reflected in the architecture

| BOUNDARY | ENGINEERING PRINCIPLE |
|---|---|
| **Authentication and sessions** | Governed sign-in and server-issued authority; passkey/WebAuthn foundations, with privileged step-up and recovery paths where supported. |
| **Work and source scope** | Sources, saved work, documents and preferences remain scoped to approved identities and contexts. |
| **Tool execution** | A model suggestion is not approval; actions pass through application policy, role and consent checks. |

| BOUNDARY | ENGINEERING PRINCIPLE |
|---|---|
| **Untrusted information** | Uploaded or retrieved content is data, not authority to override system rules or execute arbitrary instructions. |
| **Support and diagnostics** | Safe, bounded evidence and operating status rather than unrestricted access to a person's device or production secrets. |
| **Assurance** | Internal engineering controls and release proof are meaningful, but not an independent audit, pentest certification or universal regulatory attestation. |

Moriah has implemented internal access controls and personal/project scoped memory with accepted lifecycle tests. High-impact actions require separately implemented, authorized and confirmed execution paths; accepted read-only operations do not establish writeback. Formal cross-organization isolation and target-specific Security gates remain distinct.

10 / WORK EXAMPLES

# What the architecture enables in practice

Examples show the interaction of built-in capability families. They are patterns to evaluate, not claims that a particular customer's systems are already connected.

## Example A — recurring operational decision brief

A team has an issue or change request with meeting notes, a tracker record, specifications and several open decisions. Approved records enter through document upload or authorized connected-source reads. The user asks Moriah to summarize relevant facts, compare dependencies, identify unknowns and prepare a Decision Brief or Next Actions. The output can be explicitly retained in Recent Work. Any handoff or update to the original tracker is a separately authorized and validated operation.

## Example B — document-driven research and proposal work

An operator gathers approved technical, contractual and public sources to evaluate options or draft a scope. The model performs the reasoning; Moriah tracks source status and workflow limits. The operator can retain useful work, return through Recent Work, and resume the Active Thread in supported, accepted journeys. Full-content viewing of a separately saved response and authorized teammate access require their own matching proof.

## Example C — organizational planning and assistant operating rules

A responsible administrator establishes the team's operating context, available capabilities and approved connectors. Organization guidance and project-specific preferences help shape appropriate behavior without taking priority over permission, evidence or Security rules. Different model/provider setups can be evaluated without assuming the model owns the organization's work state.

## What to measure in an evaluation

| MEASURE | WHY IT MATTERS |
|---|---|
| **Less reconstruction** | Time spent recovering decisions, assumptions and source history. |
| **Better source discipline** | Ability to distinguish read information from unsupported or outdated claims. |
| **Fewer handoff errors** | Whether authorized teammates can retrieve the right approved record. |

| MEASURE | WHY IT MATTERS |
|---|---|
| **Controlled operations** | How action permissions, confirmation, failure and revocation actually work. |
| **Operating economics** | Configuration effort, provider/API costs and ongoing ownership, not just model-token price. |

11 / IMPLEMENTATION APPROACH

# How an engineering team would extend Moriah

Deployments should begin with a clearly scoped work problem, using the operating foundation and adding only the interfaces actually required.

## A practical implementation path

| WORKSTREAM | WHAT AN ENGINEER DEFINES |
|---|---|
| **Workflow** | User role, trigger, sources, desired output, decisions, human approvals and acceptance criteria. |
| **Data / custody** | System of record, metadata, retention, tenant rights and permitted data classes. |
| **Connection** | Existing adapter versus new adapter, API support, licensing, rate limits and error model. |
| **Authorization** | Organization and user grants; delegated versus app-level access; read/write scopes; confirmation policy. |
| **Execution** | Schema, request validation, bounded executor, result envelope, monitoring and revoke handling. |
| **Continuity** | Work/artifact identity, saved source relationships, Open/readback, separate Resume and permitted handoff. |
| **Validation** | Functional behavior, permission failures, source-use status, regression, browser experience and recovery expectations. |
| **Economics** | One-time discovery/setup/integration; usage and third-party consumption; recurring hosting/support. |

## Deployment models and commercial shape

Moriah can be discussed as organizational software with setup, a configured workflow foundation, a co-developed operating layer, or a broader defined technology/IP arrangement. The right commercial structure depends on rights, deployment, support obligations, operating complexity and the intended customer use—not a one-size-fits-all seat price.

A first evaluation can be deliberately narrow: one authorized data pathway, one useful model- generated output, explicit saving, independent retrieval, and a measured comparison with how the same work happens today. Further integrations or enterprise controls follow the evidence and business need.

12 / ENGINEERING AND MAINTAINABILITY

# Why the foundation was built deliberately

An external reviewer should be able to understand not just the interfaces, but how the platform can be operated, changed and maintained responsibly.

Moriah was developed independently with extensive AI assistance under founder-directed requirements, architecture, decisions, source review, testing and release control. The founder's background in cross-functional infrastructure operations influenced its insistence on clear ownership, accountability, documented decisions and system-wide rather than siloed thinking.

The engineering system separates application source from a built runtime, technical release from authenticated behavioral acceptance, and a visible screen from proof of the system state behind it. Canonical contracts, subsystem ownership maps, operating handbooks, and proof/acceptance procedures support controlled change and technical transfer.

## What a technical reviewer can reasonably examine

- Architectural boundaries among reasoning, source authority, application execution and saved work.
- Implemented source, artifact, admin and capability surfaces, assessed against a specified release and matching acceptance evidence.
- Permissions and credential behavior for a chosen connector and specific organizational scope.
- Error handling, audit/diagnostics, release/recovery process and independent operation responsibilities.
- A bounded end-to-end scenario proving the claimed work path rather than a one-off generated answer.

## Built foundation, distinct deployment obligations

Core reasoning, document, connected-source, saved-work, memory, administration and capability surfaces have passed bounded tests. Enabling a new tenant, adding a specific writeback adapter, proving complete artifact viewer readback or obtaining independent security certification are separate obligations—not evidence that the operating platform is only conceptual.

13 / QUICK REFERENCE

# Capabilities, configuration and next technical discussion

A technical review should distinguish the platform's existing capabilities from the additional work required to use them in a particular environment.

| CAPABILITY FAMILY | EXISTING FOUNDATION | FOR A PARTICULAR ORGANIZATION |
|---|---|---|
| **Reasoning and work cockpit** | Working Ask and Active Thread; five model-generated Quick Outputs with targeted acceptance. | Enabled provider route, configuration and workflow acceptance. |
| **Documents and sources** | Accepted upload/selected-source workflows, URL extraction, Word/PDF parsing, scanned-PDF OCR and supported outputs. | Permitted inputs, supported formats, extraction and source policy. |
| **Retained work** | Durable Active Thread Save → Resume → Continue journeys; separate saved-response capture and Recent Work discovery. | Saved-response viewer readback, changed-source restoration, and cross-user sharing need independent acceptance. |
| **Preferences and org controls** | Accepted personal/project scoped memory, org preference primitives, Guided Initiation and Admin UI. | Role-specific lifecycle and organizational boundary checks. |
| **Model/provider plane** | Implemented common model invocation and provider-policy paths; OpenAI org-key use accepted; Anthropic/Gemini adapters exist. | Choose and configure supported routes/credentials; test actual behavior. |
| **Tools and connectors** | Governed read-only tool registry and execution; source-card actions, policy gates and defined adapters. | Customer consent, app scopes, executor implementation/configuration and proof. |
| **Microsoft 365** | Historically proven Outlook read/search/bind/summarize, OneDrive discovery and selected document reasoning; Entra connect/refresh. | New tenant consent and authorized scopes; mail/calendar writes and live Teams transcript paths require separate acceptance. |
| **Security/operations** | Implemented session, ACL/CSRF, selected upload and cloud detection controls; guarded release and rollback. | Risk review, policies, support levels, independent assurance as required. |

## Learn more

FiveFold Ventures · fivefoldv.com | Product and public gallery: fivefoldv.com/moriah/

Official references: FiveFold Ventures | Moriah at Work