AdvanDEB System Architecture

AdvanDEB Project

December 17, 2025

1 Introduction

This document provides a printable overview of the AdvanDEB system architecture. It summarizes the main components, external services, integration patterns, and core development environment.

The project consists of the following repositories:

- `advandeb-modeling-assistant` – Main platform GUI (FastAPI + Vue) providing chat, knowledge exploration, and modeling features for all users.
- `advandeb-knowledge-builder` – Toolkit/package for knowledge operations: fact extraction, document ingestion, graph building. No UI, imported by Modeling Assistant.
- `advandeb-MCP` – Model Context Protocol server (Rust) exposing platform tools for LLM agent workflows.
- `advandeb-architecture` – System-level documentation, diagrams, development log, and planning documents.

2 Architecture Overview

advandeb-modeling-assistant serves as the **main platform GUI** and single entry point for all users. It hosts authentication (Google OAuth), implements role-based access control, and provides the complete user interface including chat, knowledge exploration, document ingestion, and modeling scenarios.

advandeb-knowledge-builder is a **Python package/toolkit** (not a standalone application) providing robust operations for knowledge processing. It contains the logic for fact extraction, stylized facts generation, document ingestion pipelines, knowledge graph building, and review workflows. The Modeling Assistant imports this package and calls its functions as needed.

advandeb-MCP is a **tool server** that exposes platform operations as MCP tools for LLM agents. It wraps Knowledge Builder operations and provides LLM inference via Ollama. Used by Modeling Assistant for agent-powered features like chat and intelligent extraction.

3 System Context

Figure 1 shows the high-level system context with the revised architecture model.

4 Modeling Assistant (Main Platform GUI)

The AdvanDEB Modeling Assistant is the **main platform interface** and single entry point for all users. It is a FastAPI + Vue application that provides:

- **Authentication**: Google OAuth integration and user management
- **Role-based Access**: Features shown/hidden based on user roles and capabilities
- **Chat Interface**: AI-powered chat using MCP server for LLM inference
- **Knowledge Exploration**: Browse facts, stylized facts, and knowledge graphs
- **Document Ingestion**: Interactive UI for uploading and processing documents
- **Modeling Features**: Scenario creation and model development

The Modeling Assistant backend imports the Knowledge Builder package to access knowledge operations, and calls the MCP server for agent-powered features.

Figure 2 shows the main containers and their relationships.

5 Knowledge Builder (Toolkit/Package)

The Knowledge Builder is a **Python package** (not a standalone application) that provides robust knowledge processing operations. It is imported by the Modeling Assistant backend.

Key modules:

- `ingestion` – Document processing pipelines
- `extraction` – Fact extraction algorithms
- `stylization` – Stylized facts generation
- `graph` – Knowledge graph building and querying
- `agents` – Agent workflows with Ollama
- `review` – Knowledge review and validation logic

The Knowledge Builder can also be used independently for batch processing operations outside of the web interface.

Figure 3 shows the conceptual data model for knowledge entities.

5.1 Document Ingestion Flow

When a user uploads a document through the Modeling Assistant GUI, the MA backend calls Knowledge Builder package functions to process it.

The document ingestion and fact extraction sequence is illustrated in Figure 4.

6 MCP Server (Tool Server)

The advandeb-MCP server is an internal service that exposes platform operations as MCP tools for LLM agent workflows.

Key Features:

- Wraps Knowledge Builder operations as MCP tools
- Provides LLM inference via Ollama
- Internal service (no authentication required)
- Used by Modeling Assistant for chat and agent features
- Rust-based for high performance

Tool Categories:

- Knowledge Building Tools: fact extraction, document ingestion, stylized facts

- Knowledge Query Tools: search facts, graph traversal
- Modeling Tools: scenario analysis, parameter suggestions
- Chat Tools: conversational LLM interface with knowledge context

7 Development Environment

All services use a single Conda environment named `advandeb`. The canonical `environment.yml` is stored in the `advandeb-architecture` repository.

To create and activate the environment:

```
conda env create -f environment.yml
conda activate advandeb
```

Core external services:

- MongoDB (default: `mongodb://localhost:27017`)
- Ollama (default: `http://localhost:11434`)

8 Integration Architecture

The platform uses a simple integration model:

- **Modeling Assistant** imports **Knowledge Builder** as a Python package
- **Modeling Assistant** calls **MCP Server** via internal HTTP/WebSocket
- **MCP Server** wraps **Knowledge Builder** operations as MCP tools
- All components share the same **MongoDB** database

8.1 Data Flow Examples

Example 1: Document Upload

1. User logs into Modeling Assistant GUI
2. User uploads PDF via web interface
3. MA backend imports KB package: `from advandeb_kb import ingest_document`
4. KB package processes document and stores in MongoDB
5. MA returns success response to frontend

Example 2: AI Chat

1. User opens chat interface in MA
2. User types question
3. MA backend sends request to MCP server
4. MCP invokes tools (using KB functions) and calls Ollama
5. MCP returns formatted response
6. MA displays chat response to user

8.2 Authentication and Authorization

All authentication is handled by the Modeling Assistant:

- Google OAuth 2.0 integration
- JWT tokens for API authentication
- Role-based access control (Administrator, Curator, Explorer)
- Capability-based permissions (Agent Access, Analytics, Reviewer)

The Knowledge Builder package and MCP server operate as internal services without authentication. User attribution is maintained by MA before calling these services.

9 Future Work

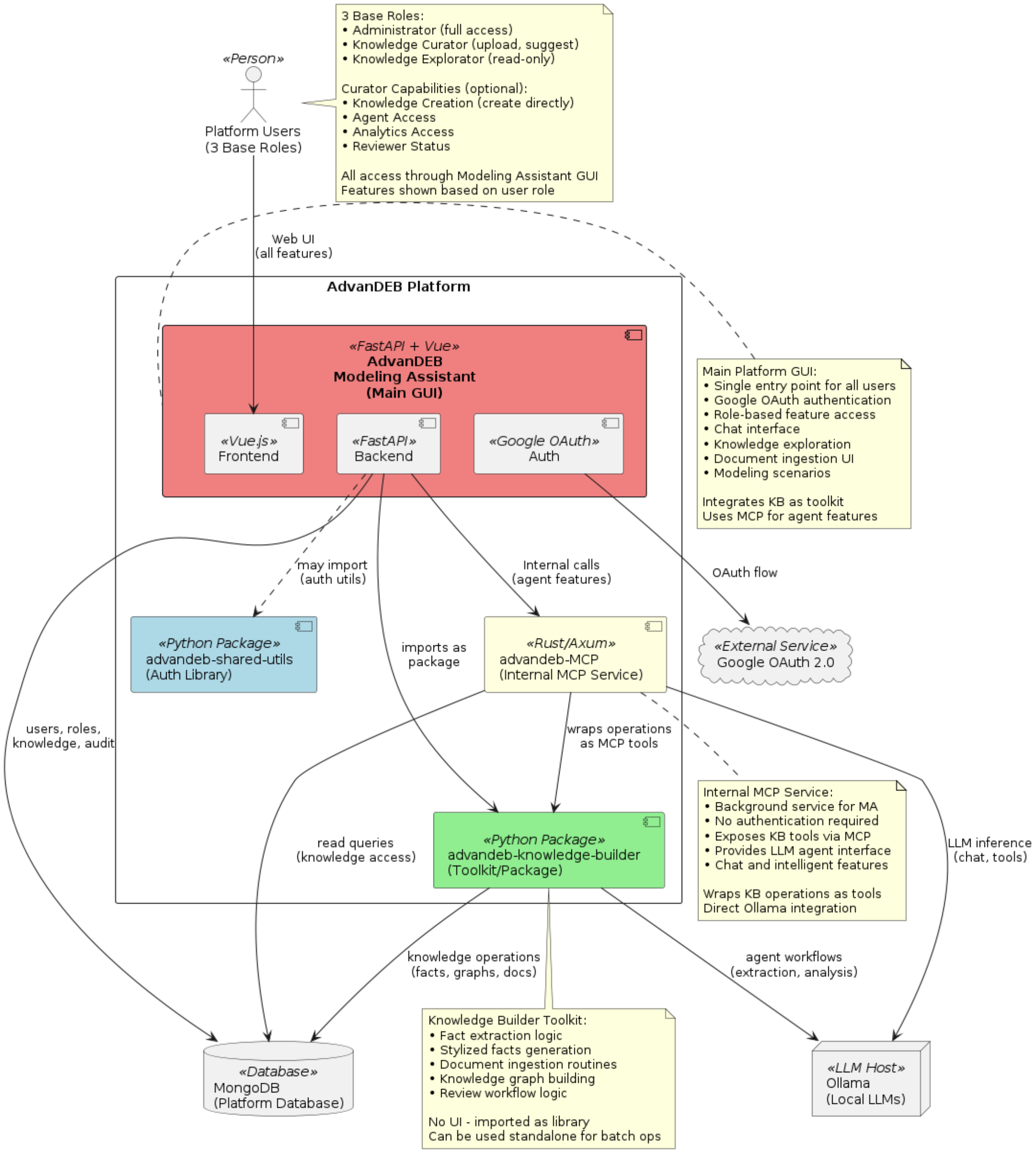
Planned extensions to this architecture and documentation include:

- Complete implementation of Modeling Assistant as main GUI platform
- Refactoring Knowledge Builder into Python package structure
- MCP server tool implementations for KB and MA operations
- Enhanced role-based access control and capability management
- Additional visualization tools for knowledge graphs
- Batch processing workflows using KB package independently
- Performance optimization for large-scale knowledge bases

10 References

For more detailed information, consult:

- `SYSTEM-OVERVIEW.md` – Comprehensive system architecture
- `ARCHITECTURE-REVISION.md` – Detailed explanation of architectural model
- `MCP-PLAN.md` – MCP server implementation plan
- `ROADMAP.md` – Development timeline and phases
- `USER-MANAGEMENT-PLAN.md` – Authentication and authorization details



3 Base Roles:
• Administrator (full access)
• Knowledge Curator (upload, suggest)
• Knowledge Explorator (read-only)

Curator Capabilities (optional):
• Knowledge Creation (create directly)
• Agent Access
• Analytics Access
• Reviewer Status

All access through Modeling Assistant GUI
Features shown based on user role

Main Platform GUI:
• Single entry point for all users
• Google OAuth authentication
• Role-based feature access
• Chat interface
• Knowledge exploration
• Document ingestion UI
• Modeling scenarios

Integrates KB as toolkit
Uses MCP for agent features

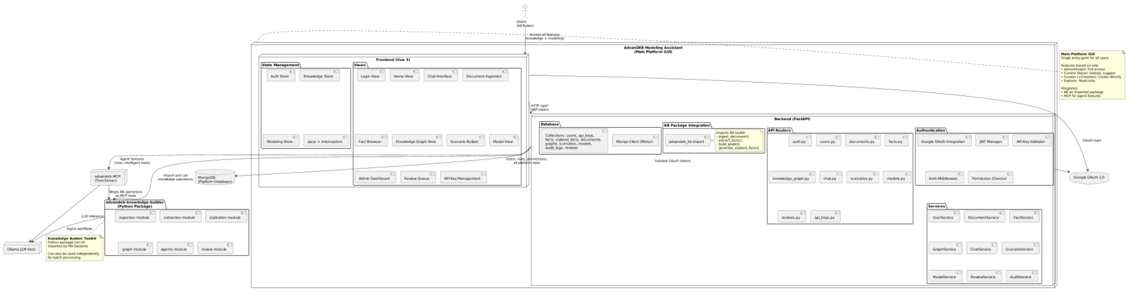
Internal MCP Service:
• Background service for MA
• No authentication required
• Exposes KB tools via MCP
• Provides LLM agent interface
• Chat and intelligent features

Wraps KB operations as tools
Direct Ollama integration

Knowledge Builder Toolkit:
• Fact extraction logic
• Stylized facts generation
• Document ingestion routines
• Knowledge graph building
• Review workflow logic

No UI - imported as library
Can be used standalone for batch ops

Figure 1: AdvanDEB System Context - Modeling Assistant as Main GUI



Imports KB toolkit:
- ingest_document()
- extract_facts()
- build_graph()
- generate_stylized_facts()

Main Platform GUI
Single entry point for all users

Features based on role:
• Administrator: Full access
• Curator (base): Upload, suggest
• Curator (+Creation): Create directly
• Explorer: Read-only

Integrates:
• KB as imported package
• MCP for agent features

Knowledge Builder Toolkit
Python package (no UI)
Imported by MA backend

Can also be used independently
for batch processing

Figure 2: Modeling Assistant as Main GUI

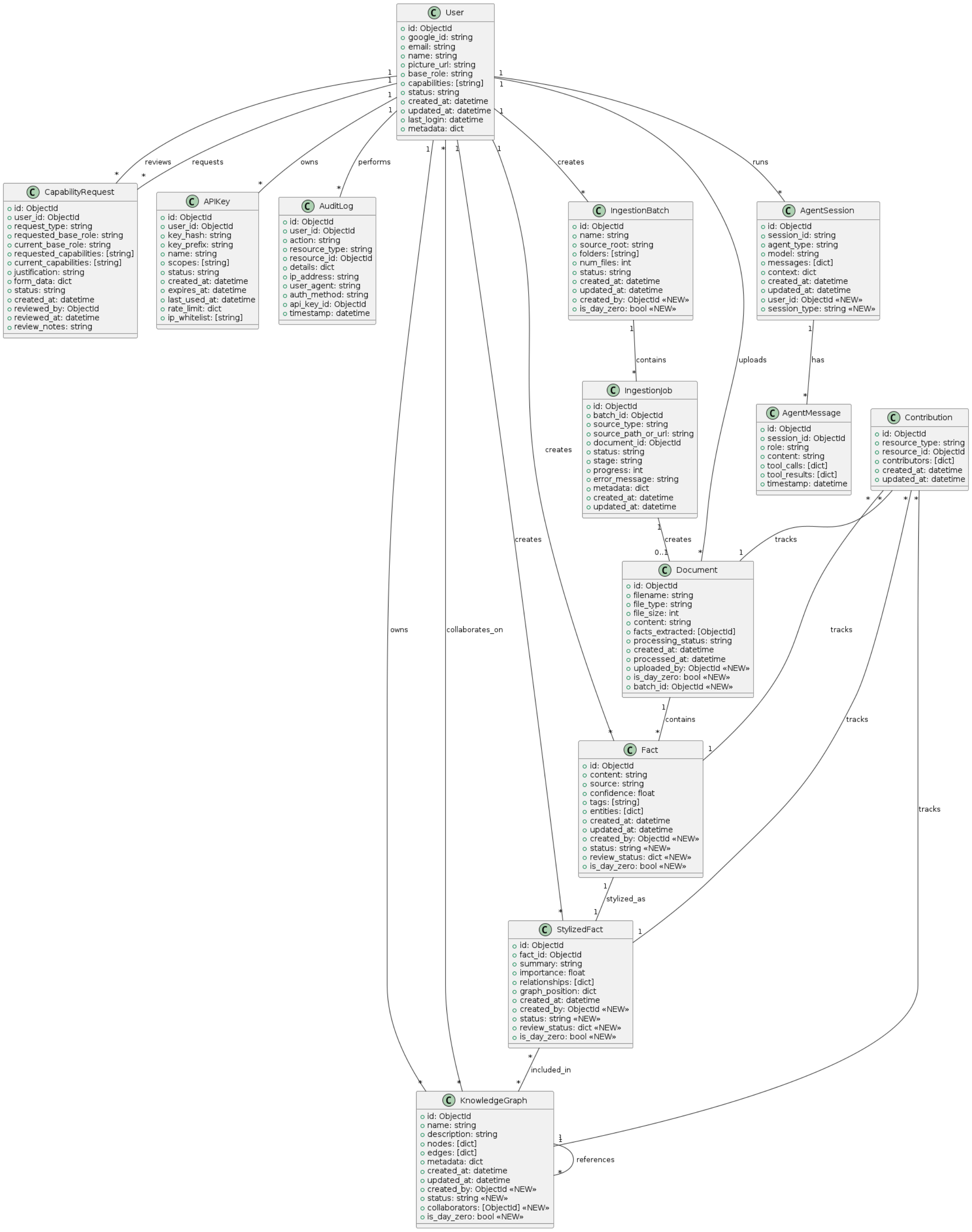


Figure 3: Knowledge Builder Conceptual Data Model

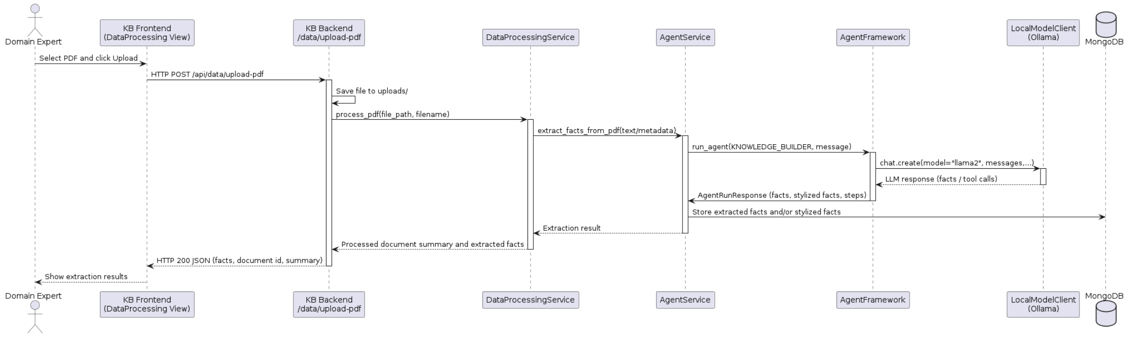

Domain Expert
KB Frontend (DataProcessing View)
KB Backend /data/upload-pdf
DataProcessingService
AgentService
AgentFramework
LocalModelClient (Ollama)
MongoDB
Select PDF and click Upload
HTTP POST /api/data/upload-pdf
Save file to uploads/
process_pdf(file_path, filename)
extract_facts_from_pdf(text/metadata)
run_agent(KNOWLEDGE_BUILDER, message)
chat.create(model="llama2", messages,...)
LLM response (facts / tool calls)
AgentRunResponse (facts, stylized facts, steps)
Store extracted facts and/or stylized facts
Extraction result
Processed document summary and extracted facts
HTTP 200 JSON (facts, document id, summary)
Show extraction results

Figure 4: Document Ingestion and Fact Extraction