AdvandEB System Architecture

AdvandEB Project

December 8, 2025

1 Introduction

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

The project consists of the following repositories:

- `advandeb-knowledge-builder` – knowledge ingestion, extraction, and exploration.
- `advandeb-modeling-assistant` – modeling-oriented retrieval and reasoning (planned).
- `advandeb-architecture` – system-level documentation, diagrams, and shared environment definition.

2 System Overview

Figure **??** shows the high-level system context, including primary users, the AdvandEB platform, and external services.

3 Knowledge Builder

The AdvandEB Knowledge Builder is a FastAPI + Vue application used to construct and explore a biological knowledge base.

Figure **??** shows the main containers.

3.1 Data Model (Conceptual)

Conceptually, the Knowledge Builder organises information into documents, facts, stylised facts, knowledge graph elements, and agent sessions. A high-level class diagram is shown in Figure **??**.

3.2 Document Ingestion Flow

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

4 Modeling Assistant

The AdvandEB Modeling Assistant is a planned component that will consume knowledge from the Knowledge Builder to support individual-based modeling.

Figure **??** shows its planned containers and dependencies.

Figure 1: AdvandEB System Context

5 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`)

Figure 2: Knowledge Builder Container Diagram

6 Integration APIs

The Modeling Assistant interacts with the Knowledge Builder primarily via HTTP APIs. This section summarises the main contracts; full details are maintained in the Markdown documentation (e.g., `docs/markdown/INTEGRATION-APIS.md`).

6.1 Knowledge Search

The Knowledge Builder exposes an endpoint for free-text search over facts, stylised facts, and graph elements.

- Endpoint: `GET /api/knowledge/search`
- Typical parameters: query string `q`, optional type filter (e.g., `fact`, `stylized_fact`, or `graph_node`), and a result limit.

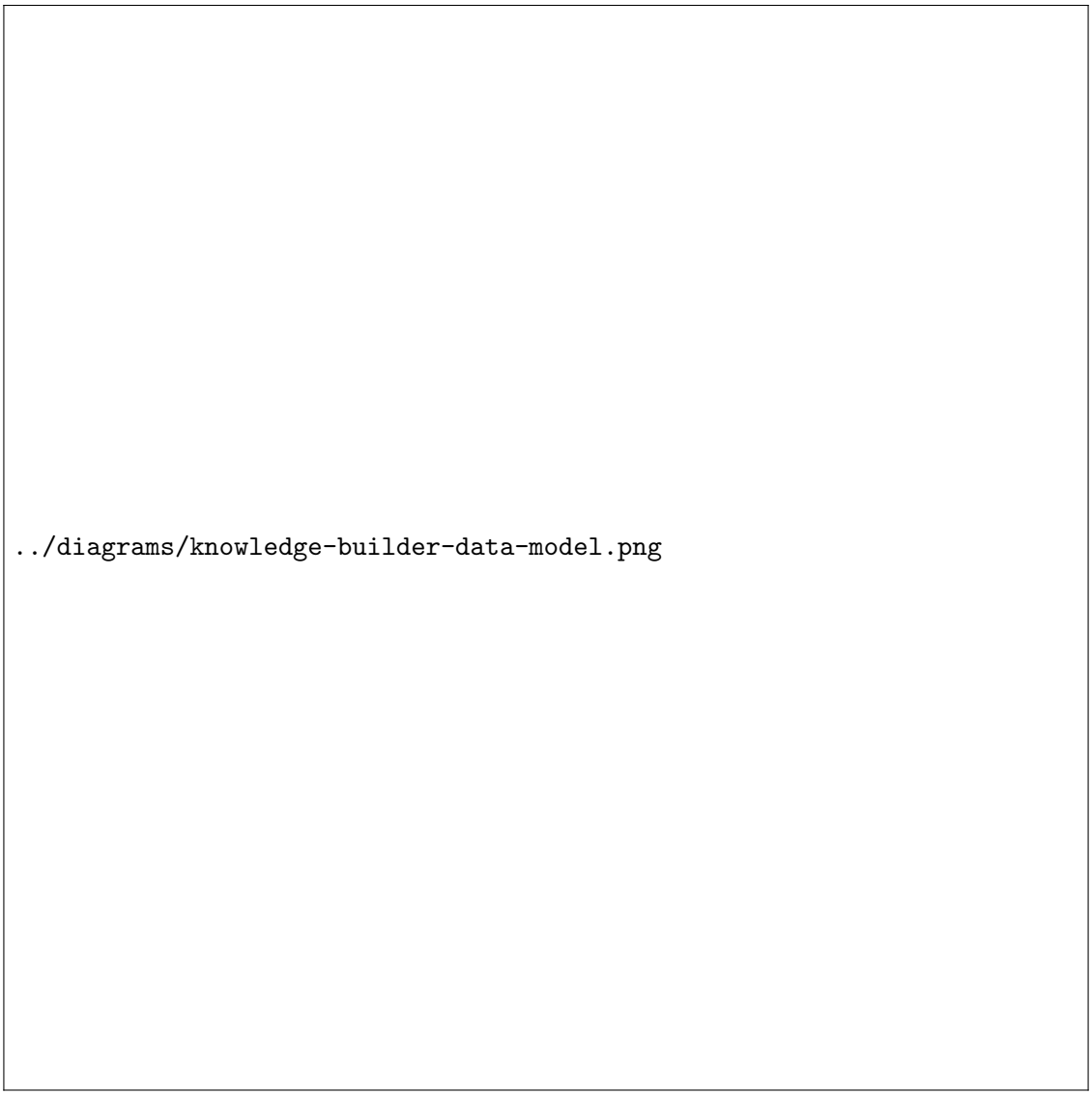

Figure 3: Knowledge Builder Conceptual Data Model

- Response: a list of scored results with identifiers, text, source references, and tags.

6.2 Facts and Stylised Facts

Individual facts and stylised facts can be retrieved by identifier.

- Endpoint: `GET /api/knowledge/facts/{fact_id}`
- Endpoint: `GET /api/knowledge/stylized-facts/{stylized_fact_id}`
- Responses include text, source, timestamps, tags, and optional metadata.

6.3 Graph Exploration

To understand local structure around entities, the Modeling Assistant can request graph neighbourhoods.

Figure 4: Document Ingestion and Fact Extraction

- Endpoint: `GET /api/knowledge/graph/neighborhood`
- Typical parameters: `node_id`, a depth parameter, and a maximum number of nodes.
- Response: sets of nodes and edges that can be visualised or analysed.

6.4 Agent Invocation

The Knowledge Builder also provides higher-level reasoning via agents.

- Endpoint: `POST /api/agents/run`
- Request: an agent identifier, natural-language input, and an optional context referencing facts, stylised facts, or graph nodes.
- Response: structured or textual output from the agent, tool call traces, and a session identifier for follow-up interactions.

Figure 5: Modeling Assistant Container Diagram (Planned)

6.5 Modeling Assistant APIs

The Modeling Assistant exposes its own APIs for clients, while internally consuming the Knowledge Builder endpoints.

- Scenario management (e.g., `POST /api/scenarios`) to define modelling problems and objectives.
- Model proposal endpoints (e.g., `POST /api/scenarios/{scenario_id}/propose-model`) that assemble candidate model structures and provide justifications referencing Knowledge Builder items.

7 Future Work

Planned extensions to this printable document include:

- Detailed data model diagrams for the Knowledge Builder.
- Additional sequence diagrams for Modeling Assistant workflows.
- Short design decision records for major architectural choices.