

Architecture of Agents

List (12):

1. User Intent / Parser
2. Canonicalizer (Name → Structure)
3. Literature Retrieval
4. Reaction Retrieval
5. Retrosynthesis Planner
6. Forward Reaction Predictor / Feasibility
7. Route Scorer / Ranker
8. Safety & Compliance Checker
9. Protocol Generator
10. Procurement & Cost Estimator
11. Data Curation & Provenance Logger
12. Feedback / Active Learning Agent

1) User Intent / Parser

What it does: extract user goal, constraints, delivery format (bench vs automation), priorities (cost/safety/time), and optional files.

Inputs: raw user text + optional attachments (SMILES/Image).

Outputs: structured request JSON: {target (name/SMILES), constraints, mode, user_id, priority}.

Internal components:

- Lightweight NLP pipeline (intent classification + slot filling).
- Validator (ensures required fields present).
- Preprocessor for images (OCR or image→SMILES if needed).

Recommended tools/models:

- Small LLM or intent classifier (DistilBERT / open LLM prompts) for slot filling.
- Tesseract / image2mol for structure images (optional).

Persistence / APIs:

- Stateless; store parsed request in requests table (DB) and return request_id.

2) Canonicalizer (Name → Structure)

What it does: normalize names to canonical identifiers (SMILES, InChI, PubChem CID).

Inputs: chemical name or uploaded structure.

Outputs: canonical SMILES, InChI, synonyms, molecular metadata (MW, formula).

Internal components:

- Name-resolution service calling PubChem / ChemSpider / OPSIN.

- RDKit for validation, sanitization, tautomer handling, standardization.

Recommended tools:

- PubChem REST, OPSIN, RDKit.

Persistence / APIs:

- Cache canonicalized results in molecules table; API returns canonical IDs and confidence.

3) Literature Retrieval

What it does: find papers, patents, protocols, and extraction of reaction snippets + experimental conditions.

Inputs: SMILES/InChI + keywords + constraints.

Outputs: ranked list of documents + extracted snippets (reaction SMILES, conditions, yields) + embeddings.

Internal components:

- Vector retrieval (semantic search) + keyword search.
- Document parser & extractor (PDF → text → structured reaction snippets).
- Deduplication and relevance scoring.

Recommended tools/models:

- Vector DB (Qdrant/Pinecone/Weaviate), sentence-transformers for embeddings, PDF parsing (pdfminer, Grobid), SciBERT-based extractor.
- Optional external APIs: PubMed, USPTO dataset, OpenReactionDatabase.

Persistence / APIs:

- Store results in search_results table and store PDFs in object storage. Return list with provenance, snippet, and embedding id.

4) Reaction Retrieval

What it does: query reaction databases for precedent transformations matching target or substructures.

Inputs: target SMILES, substructure pattern, similarity threshold.

Outputs: list of recorded reactions with stepwise reagents, yields, conditions, reaction IDs.

Internal components:

- Exact/template match engine + similarity search (fingerprint/RDKit).
- Template expansion to map reagents → steps.

Recommended tools:

- RDKit fingerprints, reaction DB (USPTO, Reaxys if licensed, OpenReactionDatabase).
- Local reaction index with Postgres + JSONB or vector DB for reaction embeddings.

Persistence / APIs:

- reactions table with reaction SMILES, conditions, citations.

5) Retrosynthesis Planner

What it does: propose one or more synthetic routes (tree/linear) from starting materials to target.

Inputs: target SMILES + constraints (max steps, avoid reagents, prefer reagents).

Outputs: N candidate routes (each route = ordered steps with reagents, conditions, confidence).

Internal components:

- Multi-strategy planner: template-based + ML generative model (transformer seq2seq / Graph2SMILES).
- Constraint filter and policy (prune forbidden reagents, step limits).
- Generate candidate subgoals and expand search tree.

Recommended models/tools:

- Open-source retrosynthesis models (e.g., trained transformer models) + AiZynthFinder-like template planner + RDKit for validation.
- Beam-search or MCTS for route exploration.

Persistence / APIs:

- Save routes in routes table with per-step metadata and model-version tags.

6) Forward Reaction Predictor / Feasibility

What it does: predict product yield, selectivity, potential side-products, and reaction feasibility for proposed steps.

Inputs: reaction SMILES (reactants + reagent + conditions).

Outputs: predicted yield, selectivity/confidence, risk flags for low feasibility.

Internal components:

- Predictive model (graph neural network or reaction transformer).
- Heuristic checks (mass balance, impossible transforms).

Recommended tools/models:

- Pretrained forward prediction models (transformer-based) or a simpler ML regressor trained on reaction data. RDKit-based sanity checks.

Persistence / APIs:

- Store predictions in route step metadata; return scores to Ranker.

7) Route Scorer / Ranker

What it does: aggregate multiple metrics to score and rank candidate routes (cost, safety, yield, steps, patent risk).

Inputs: list of routes + forward predictions + procurement cost + safety flags.

Outputs: ranked routes with composite score and per-metric breakdown.

Internal components:

- Scoring function that weights normalized metrics (configurable).
- Explainability module to show why route ranked higher (top contributing metrics).

Recommended tools:

- Simple microservice with pluggable scoring formula; can use LightGBM/linear model for learned ranking later.

Persistence / APIs:

- Save ranked order and scores in route_rankings table; provide API for UI.

8) Safety & Compliance Checker

What it does: evaluate reagents/steps for hazards, regulatory control, thermal risks, environmental impact, and generate mitigation suggestions.

Inputs: route steps (reagents, solvents, conditions) + jurisdiction/regulatory context.

Outputs: hazard score, flags (explosive, toxic, controlled), mitigation recommendations, human-review required flag.

Internal components:

- Rule-based engine (dangerous reagent list, GHS hazard mapping).
- ML classifier for less obvious risks (e.g., peroxide-forming reagents).
- Links to SDS (safety data sheets) and regulatory databases.

Recommended tools:

- Local rules engine (Drools/simple logic), GHS hazard datasets, PubChem safety endpoints.

Persistence / APIs:

- Save flags to safety_logs; trigger human-review workflow if high-risk.

9) Protocol Generator

What it does: convert selected route into a clear, actionable protocol (bench-level or machine-run format).

Inputs: chosen route + chosen scale (mg/g), user preferences, equipment availability.

Outputs: step-by-step protocol (volumes, concentrations, times, quench steps) and optional machine instructions (CSV/JSON for autosynth).

Internal components:

- Template engine (protocol templates per reaction class).
- Unit converter & stoichiometry calculator.
- Safety inserts and QA checks (check for hazardous quenches, exotherms).

Recommended tools:

- Jinja-style templates, RDKit for stoichiometry, internal protocol templates curated by chemists.

Persistence / APIs:

- Store protocol versions in protocols table and offer ELN export (CSV/JSON).

10) Procurement & Cost Estimator

What it does: query vendor catalogs and estimate reagent availability, lead time and cost.

Inputs: list of reagents + preferred vendors / region.

Outputs: availability, cost estimates, alternative suppliers, lead times.

Internal components:

- Vendor API connectors (Sigma-Aldrich, Fisher, TCI, or generic vendor scraping).
- Alternative substitution suggestions (from reaction DB or substitution rules).

Recommended tools:

- REST connectors, caching layer for price updates, fallback to catalog CSVs.

Persistence / APIs:

- Save procurement info to procurement table linked to route steps.

11) Data Curation & Provenance Logger

What it does: record full provenance — documents, model versions, timestamps, agent outputs and user interactions.

Inputs: outputs from all agents, user approvals, experimental results.

Outputs: auditable records, ELN-compatible logs, versioned dataset for retraining.

Internal components:

- Immutable logging store (append-only), metadata management, DB schema for provenance.
- Exporters for ELN / LIMS.

Recommended tools:

- Postgres with append-only tables + object store for files; store model-version and commit hash.

Persistence / APIs:

- Central provenance DB and API for audits and for feeding the Feedback agent.

12) Feedback / Active Learning Agent

What it does: ingest experimental outcomes or user feedback to update rankings, retrain models, and improve heuristics.

Inputs: experimental results (yields, times, notes), user ratings/edits.

Outputs: updated route preferences, training records, flagged model drift.

Internal components:

- Data validation & cleaning pipeline, training job launcher, monitoring dashboard.
- Small active-learning selection strategy (triage which experiments to use for retraining).

Recommended tools:

- MLOps stack (MLflow or similar), scheduled retraining jobs, metrics tracking.

Persistence / APIs:

- Stores datasets and model artifacts; logs retraining metadata.

Orchestration & Communication

- Use a **central orchestrator** (FastAPI gateway + task queue) to accept requests and coordinate agents.
- Communication: agents expose REST/gRPC endpoints; long jobs are queued (Celery/RQ) and workers pick up tasks.
- Shared state: relational DB (Postgres) for structured records; object store (S3) for files; vector DB for embeddings.
- Pub/Sub for events (Kafka/RabbitMQ) to trigger background workflows (e.g., when user approves a route → Protocol Generator runs).

Data Flow (compact)

1. UI → User Intent Agent → save request.
2. Canonicalizer resolves molecule.
3. Parallel: Literature Retrieval + Reaction Retrieval.
4. Retrosynthesis Planner proposes routes.
5. Forward Predictor + Safety Checker evaluate steps.
6. Route Scorer ranks routes (Procurement optionally enriches costs).
7. User selects route → Protocol Generator produces lab-ready protocol.
8. Data Curation logs provenance and Feedback agent waits for results.