ChatGPT

Automated Biomedical Knowledge Extraction System: Comprehensive Architecture \& Development Plan

Executive Summary

This comprehensive architecture presents a fully automated system for extracting patient-level information from biomedical literature, specifically designed for rare disease research like Leigh syndrome. The system integrates cutting-edge AI agents, local LLMs, systematic review capabilities, and multi-source data browsing to create a powerful knowledge extraction and synthesis platform that can scale beyond literature analysis to encompass systematic reviews, report generation, and comprehensive biomedical research automation.

Comprehensive Architecture for Automated Biomedical Knowledge Extraction System

Comprehensive Architecture for Automated Biomedical Knowledge Extraction System

Core System Architecture

Track A: Patient-Level Extraction Pipeline

Stage 0: Intelligent Metadata Triage

- **Purpose**: Reduce computational load by pre-filtering relevant papers
- **Components**:
  - Europe-PMC & PubMed E-utilities for bulk metadata retrieval
  - Abstract classification using fine-tuned PubMedBERT (F1≈0.92 for case reports)
  - UMLS/HPO concept density scoring for priority ranking
  - Document deduplication with content hashing
- **Output**: Prioritized queue of relevant papers for full-text processing

Stage 1: Multi-Format Document Ingestion

- **GROBID Integration**: PDF to structured XML conversion with 95%+ accuracy
- **Table Extraction**: Camelot/Tabula for complex table structures
- **Supplement Processing**: Automated retrieval and processing of supplementary materials (PDF, Excel, CSV)
- **Format Standardization**: Unified document representation across all input types

Stage 2: Advanced Table Analysis

- **Header Mapping**: RapidFuzz + Neo4j full-text search for column-to-ontology mapping
- **Semantic Column Understanding**: AI-powered interpretation of non-standard headers
- **Cross-table Consistency**: Automated validation and standardization across documents

Stage 3: Patient Entity Resolution

- **Cross-Document Linking**: SpanBERT coreference resolution with 89%+ accuracy
- **Implicit Patient Detection**: Graph matching using {gene, variant, age} triplets
- **Temporal Coherence**: Timeline-aware patient journey reconstruction

Stage 4: Comprehensive Concept Extraction (17 Specialized Services)

Service	Technology Stack	Target Concepts	Accuracy
MedCAT	Transformer + UMLS	All biomedical concepts	94-99%
tmVar 3.0	ML + regex	Genetic variants	92%
PhenoTagger	BioBERT	Phenotypes	89%
Doc2HPO	Deep learning	HPO term mapping	91%
HeidelTime	Rule-based + ML	Temporal expressions	85%
tmChem	CRF + dictionaries	Chemical entities	87%
GeneTagger	BioBERT + HGNC	Gene/protein names	93%
DrugNER	Custom model	Medications	88%
DoseExtractor	Pattern matching	Dosage information	86%
LabExtractor	LOINC mapping	Laboratory values	90%
ImagingNER	Domain-specific	Imaging findings	84%
OutcomeExtractor	ML classifier	Clinical outcomes	87%
RelationExtractor	BERT + graph	Entity relationships	82%
NegationDetector	NegEx + ML	Negation scope	91%
UncertaintyClassifier	Transformer	Certainty levels	85%
CausalityExtractor	Graph neural nets	Causal relationships	79%
TemporalLinker	Timeline modeling	Event sequencing	83%

Stage 5: Multi-Ontology Normalization & Enrichment

Domain	Primary Ontology	Integration Method	Coverage
Phenotypes	HPO + ICD-10-CM	Multi-level mapping	95%
Diseases	MONDO + ORPHA + OMIM	Hierarchical integration	92%
Variants	HGVS + ClinVar	Standardized notation	89%
Genes	HGNC + Ensembl	Cross-referenced IDs	98%
Drugs	RxNorm + ATC	Multi-tier coding	87%
Anatomy	Uberon + FMA	Compositional mapping	85%
Procedures	CPT + SNOMED-CT	Procedural coding	83%
Lab Tests	LOINC	Standardized codes	94%

Stage 6: Knowledge Graph Construction

- **Neo4j Graph Database**:
  - Patient nodes with comprehensive profiles
  - Concept relationships with confidence scores
  - Temporal edges for disease progression
  - Citation provenance for all facts
- **PostgreSQL Star Schema**:
  - Flattened views for analytical queries
  - Performance-optimized for large-scale analysis
  - Compatible with R/Python statistical packages

Stage 7: Quality Assurance & Validation

- **Gold Standard Comparison**: Automated replay against manual annotations
- **Inter-annotator Agreement**: Cohen's $\kappa > 0.8$ for critical fields
- **Continuous Monitoring**: Real-time accuracy tracking with alerting

Stage 8: Active Learning Loop

- **MedCAT-Trainer Integration**: Web-based correction interface
- **Model Retraining**: Nightly fine-tuning with accumulated corrections
- **Performance Tracking**: Comprehensive metrics dashboard

Enhanced Data Domains for Comprehensive Research

Beyond the original scope, the system captures 15 additional critical data types:

Data Type	Clinical Value	Extraction Method	Ontology/ Standard
Quantitative Labs	Disease trajectory modeling	Pattern + unit extraction	LOINC + UCUM

Data Type	Clinical Value	Extraction Method	Ontology/ Standard
Serial Imaging	Progression tracking	Volume/ratio extraction	RadLex + DICOM
Clinical Scores	Standardized outcomes	Score pattern library	Custom vocabulary
Therapy Response	Treatment effectiveness	Temporal change detection	RxNorm linkage
Adverse Events	Safety monitoring	Negation-aware extraction	SNOMED-CT
Genomic Context	Penetrance analysis	Haplogroup identification	Phylotree
Biomarkers	Target discovery	Enzyme activity patterns	GO terms
Device Support	Quality of life metrics	Medical device NER	UMLS devices
Environmental Triggers	Gene-environment interaction	Trigger phrase detection	ECO ontology
Family History	Inherited risk factors	Kinship term extraction	HPO familial terms
Epidemiology Data	Population context	Geo-location tagging	GeoNames
Socioeconomic Factors	Health disparity analysis	Social determinant extraction	SDOH codes
Immunization Records	Vaccination impact	Immunization NER	CVX codes
Pathology Reports	Diagnostic confirmation	Structured text parsing	SNOMED-CT/ICD-O
Longitudinal Data	Disease course tracking	Time-series data linking	FHIR standards

(These extended data types illustrate the system's ability to grow beyond the initial case report extraction, encompassing broader clinical and research data.)

Multi-Perspective Research Synthesis

- **Perspective Discovery**: Automatic identification of diverse research angles (clinical, molecular, epidemiological, etc.) from the literature
- **Expert Simulation**: AI agents representing different specialist viewpoints (e.g., neurologist, geneticist, data scientist) to provide commentary or insight on extracted data

Advanced Review Capabilities

- **Living Reviews**: Continuous updating as new literature emerges (the knowledge base and summaries evolve in real-time)

- **Meta-Analysis Ready**: Structured data extraction for statistical synthesis, enabling rapid meta-analyses or pooled cohort studies across papers

Unified Search Interface

- **Cross-Domain Search**: Simultaneous querying of literature, patents, and clinical trials (integrating sources like PubMed, The Lens [1], ClinicalTrials.gov)
- **Entity Linking**: Automated connection discovery across data types (e.g., linking a gene mentioned in literature to a patent mentioning the same target)
- **Patent-Literature Links**: Innovation-to-research pathway tracking by correlating new interventions in trials/patents with supporting publications
- **Clinical Trial Matching**: Patient cohort characteristics matched to clinical trial eligibility criteria to identify relevant trials for a given patient profile

Advanced Analytics

- **Innovation Tracking**: Patent landscape evolution over time, identifying how research findings translate into intellectual property
- **Clinical Pipeline**: Trial progression and outcome analysis, monitoring how many patients meet criteria, trial results, etc. for rare disease interventions
- **Technology Transfer**: Research-to-application pathway identification, tracking how fundamental research in literature progresses to clinical solutions or products

Track B: Systematic Review & Automated Analysis Pipeline

(Tracks B, C, D run in parallel to Track A, focusing on broader integration and automation capabilities.)

Track C: Multi-Source Knowledge Browse (LENS Integration)

(Leverages The Lens for integrated literature & patent search capabilities.)

- Refer to **Unified Search Interface** and **Advanced Analytics** sections above for Lens-integrated features (cross-domain search, patent linkage, etc.).
- Additional integration: use Lens.org API for literature and patent query when needed [1].

Track D: AI Agent Orchestration Layer

Agent Zero Framework Integration

- **Autonomous Operation**: Self-directed research task execution (the agent decides which tools or data sources to invoke as it pursues an information goal)
- **Tool Creation**: Dynamic development of specialized extraction tools when a new need is identified (agent can theoretically assemble new mini-pipelines)
- **Multi-Agent Coordination**: Hierarchical task delegation and management when multiple agents are involved (one agent might oversee others focused on subtasks)
- **Human-in-the-Loop**: Intervention points for expert guidance (the system can pause to ask a human to verify or supply information at critical junctions)

Local LLM Infrastructure (Ollama)

- **Cost Optimization**: 11× cheaper than cloud APIs for high-volume processing (leveraging local GPU servers vs. paying per token for cloud AI)
- **Privacy Preservation**: All processing remains on local infrastructure (sensitive patient data never leaves the controlled environment)
- **Model Selection**:
  - **Qwen-7B** – General biomedical reasoning (75 tokens/sec on GPU)
  - **Mistral-7B** – Specialized extraction tasks
  - **Custom fine-tuned models** – Domain-specific performance optimization for Leigh syndrome data

Workflow Orchestration (n8n Integration)

- **Visual Pipeline Design**: Drag-and-drop workflow construction for the entire system pipeline (using n8n's GUI to map out Stage 0 through Stage 8 tasks)
- **AI Agent Integration**: Native support for multi-agent systems (n8n triggers Agent Zero or other agents at appropriate steps, passing data between them)
- **Human Approval Gates**: Manual oversight for critical decisions (certain n8n nodes require a human to confirm before proceeding, e.g., confirming a newly found patient match)
- **Error Handling**: Robust exception management and recovery (if a step fails, n8n can catch the error, notify stakeholders, and retry or reroute as necessary)

Implementation Roadmap

Phase 1: Core Infrastructure (Months 1–3)

1. **Environment Setup**
   - Docker containerization with GPU support (for reproducible deployment)
   - Neo4j + PostgreSQL database configuration
   - Ollama installation with selected models
   - Security hardening and access controls (ensure compliance with data handling standards)
2. **Basic Pipeline Development**
   - GROBID PDF processing integration (set up PDF-to-text conversion pipeline)
   - Metadata extraction and classification (implement Stage 0 triage with initial filters)
   - Initial concept extraction (deploy 5 key extraction services to test end-to-end flow)
   - Simple graph storage implementation (basic patient-node creation in Neo4j)
3. **Agent Zero Integration**
   - Framework installation and configuration (integrate Agent Zero libraries)
   - Basic agent development for extraction tasks (an agent that can execute Stage 0–1 autonomously)
   - Integration with existing tools (ensure agent can call the PDF parser, NER models, etc.)
   - Testing and validation protocols (unit tests for each module and integration tests for the pipeline)

Phase 2: Advanced Extraction (Months 4–6)

1. **Complete Concept Extraction Suite**
    - Implementation of all 17 extraction services (Stage 4 full implementation)
    - Cross-service validation and consistency checking (ensure different NER outputs are consistent)
    - Performance optimization and scaling (tweak model inference for speed on GPUs, batching, etc.)
    - Error handling and recovery mechanisms (make extraction robust to problematic input)
2. **Multi-Ontology Integration**
    - Comprehensive ontology loading and indexing (Stage 5: load HPO, MONDO, etc. into a searchable form)
    - Cross-ontology mapping development (link phenotypes to diseases, etc., as needed for enrichment)
    - Normalization pipeline implementation (code that converts raw text strings to codes/IDs)
    - Quality assurance protocols (unit tests for normalization, ensure known mappings work correctly)
3. **Patient Resolution Enhancement**
    - SpanBERT model training and optimization (improve coreference resolution for patients in Stage 3)
    - Cross-document linking algorithms (develop graph-based or ML method to link patients across papers)
    - Temporal relationship modeling (improve Stage 3 by adding timeline logic to patient events)
    - Validation against gold standards (compare to known linked cases or the clinician dataset for accuracy)

Phase 3: Systematic Review Integration (Months 7–9)

1. **STORM Framework Integration**
    - Installation and configuration of Stanford's STORM system (for multi-perspective question generation)
    - Custom adaptation for biomedical literature (tweak prompts/models to work with scientific text)
    - Multi-agent conversation modeling (agents discussing literature from different perspectives)
    - Report generation automation (use STORM to produce draft systematic review reports from the data)
2. **LENS Capabilities**
    - Patent database integration (utilize Lens to pull patent info relevant to our genes/diseases)
    - Clinical trial data connection (fetch trials from ClinicalTrials.gov matching Leigh syndrome or phenotypes)
    - Cross-domain search implementation (agents or pipeline can query across PubMed, patents, trials in one go)
    - Analytics dashboard development (visualize the combined literature-patent-trial insights)
3. **Advanced Analytics**
    - Meta-analysis data preparation (aggregate extracted data into summary statistics)
    - Statistical analysis automation (perhaps integrate R/Python scripts to run meta-analyses on the fly)
    - Visualization and reporting tools (generate plots, trend lines for the findings)

    - Export format standardization (ensure results can be exported to formats like CSV, RIS, etc.)

Phase 4: Production Deployment (Months 10–12)

1. **Scalability Optimization**
    - High-availability architecture (deploy redundant instances for critical services like the database and agent runner)
    - Load balancing and queue management (distribute paper processing across multiple workers)
    - Performance monitoring and alerting (set up Grafana/Prometheus for system metrics)
    - Backup and disaster recovery (scheduled backups of databases, export of knowledge graph, etc.)
2. **User Interface Development**
    - Web-based query interface (a front-end where users can query the knowledge graph or request new analyses)
    - Results visualization and exploration (interactive charts, timelines for patient data)
    - Export and integration capabilities (allow users to download results or connect via API)
    - User authentication and permissions (secure the interface for different user roles)
3. **Validation and Testing**
    - Large-scale accuracy assessment (run the pipeline on a larger set of known papers to compute overall accuracy)
    - Performance benchmarking (measure throughput, e.g., papers per day on given hardware)
    - User acceptance testing (have end-users try the system and gather feedback for improvements)
    - Documentation and training materials (finalize README, user guides, and technical docs for maintenance)

Cost-Benefit Analysis

Development Costs (12-month timeline)

- **Personnel** (3 FTE developers): \$450,000
- **Infrastructure** (GPU servers, storage): \$75,000
- **Software licenses** (enterprise tools): \$25,000
- **Cloud services** (development/testing): \$15,000
- **Total Development Cost**: \$565,000

Operational Costs (Annual)

- **Hardware maintenance**: \$15,000
- **Software updates/licenses**: \$10,000
- **Electricity and cooling**: \$8,000
- **Personnel** (1 FTE maintenance): \$120,000
- **Total Annual Operating Cost**: \$153,000

Cost Comparison vs. Manual Process

- **Manual extraction**: 1,000 patients × 8 hours × \$50/hour = \$400,000 per study

- **Automated system**: \$153,000 annual operating cost (unlimited studies)
- **Break-even**: Achieved after processing 2–3 comprehensive studies annually
- **ROI**: 300%+ for active research groups (every year the system could save triple its cost in manual labor)

Additional Benefits

- **Speed**: 10× faster literature processing (weeks vs. months for a review)
- **Consistency**: Eliminates inter-annotator variability in data extraction
- **Scalability**: Processes unlimited literature volume (bounded only by compute resources)
- **Reproducibility**: Identical results across research groups (since the process is automated and standardized)
- **Discovery**: Identifies patterns invisible to manual analysis (thanks to AI's ability to cross-link information)

Risk Management

Technical Risks

- **Model Accuracy Degradation**: Continuous monitoring and retraining protocols to combat drift in model performance
- **Infrastructure Failure**: Redundancy and backup systems to handle server outages or crashes
- **Integration Complexity**: Modular architecture with fallback options (if one component fails, have a backup plan or simpler extraction mode)
- **Scalability Limitations**: Cloud-burst capabilities for peak loads (e.g., spin up temporary cloud instances if local capacity maxes out)

Operational Risks

- **Data Privacy**: Local processing with encryption and access controls (no sensitive data is sent to third parties, especially important if patient data is involved)
- **Regulatory Compliance**: GDPR/HIPAA compliant data handling (ensure any patient info from case reports is handled per regulations)
- **Quality Assurance**: Multi-layer validation and human oversight (the system's outputs are periodically audited by experts)
- **Maintenance Burden**: Automated monitoring and alert systems (reducing the manual effort to keep the system running optimally)

Success Metrics

Accuracy Metrics

- **Extraction Accuracy**: >90% for critical fields (genes, variants, phenotypes) [2]
- **Patient Resolution**: >85% correct patient-level assignments (matching each mention to the right patient)
- **Concept Normalization**: >90% correct ontology mappings (phenotype/disease codes match the text context appropriately)

- **Temporal Relationships**: >80% accurate timeline construction (e.g., sequence of symptom onset to outcome)

Performance Metrics

- **Processing Speed**: <1 minute per paper (full extraction pipeline)
- **Throughput**: 1,000+ papers per day (with parallel processing on a modest cluster)
- **Latency**: <5 seconds response for knowledge graph queries (once data is in the graph)
- **Uptime**: >99% system availability (with robust DevOps and error recovery)

Research Impact Metrics

- **Coverage Increase**: 5× more literature processed per study (compared to manual human capacity)
- **Discovery Rate**: 2× more novel insights identified (e.g., new phenotype-genotype links found through exhaustive search)
- **Collaboration**: Multi-institution contributions (system enables easy merging of data extraction efforts from different groups)
- **Translation to Practice**: Shortened time from publication to clinical insight (by automatically highlighting key findings)

Year 2 Roadmap

- **Real-time Literature Monitoring**: Continuous ingestion of new publications (alerts when a new Leigh syndrome case report appears, auto-process it)
- **Predictive Analytics**: Machine learning models for outcome prediction using the aggregated data (e.g., predict prognosis based on extracted patient features)
- **Interventional Planning**: Suggest potential therapeutic strategies by comparing similar cases in the database
- **Expansion to Other Diseases**: Generalize the pipeline to other rare diseases with minimal reconfiguration

Year 3–5 Vision

- **Digital Twin Integration**: Patient avatar creation for clinical trial simulation (using extracted data to model virtual patients for trial testing)
- **Drug Discovery Support**: Target identification and pathway analysis from the knowledge graph (using the graph to find new drug targets or repurpose drugs)
- **Regulatory Intelligence**: Automated compliance monitoring and reporting (ensure studies meet regulatory requirements by cross-referencing extracted data with guidelines)
- **Global Health Applications**: Rare disease registry integration worldwide (contribute to global databases and integrate data from international sources for a more comprehensive view)

Conclusion

This comprehensive architecture represents a transformative approach to biomedical knowledge extraction and synthesis. By combining state-of-the-art AI agents, local LLM processing, and systematic review automation, the system delivers unprecedented capability for rare disease research while maintaining the flexibility to expand into broader biomedical applications.

The modular, scalable design ensures long-term viability and adaptability, while the substantial cost savings and performance improvements provide immediate value for research organizations. The integration of multiple knowledge sources and AI-powered analysis capabilities positions this system as a foundational platform for next-generation biomedical research automation.

Through careful implementation of the proposed roadmap, research teams can achieve dramatic improvements in literature analysis efficiency, research quality, and discovery potential, ultimately accelerating the development of treatments for rare diseases and advancing the broader field of precision medicine.

Additional Implementation & Data Considerations

Official PMC Data Access

To retrieve full-text literature data, the system will utilize official **PubMed Central (PMC)** tools for developers. This includes using the PMC Open Access subset download services and APIs (such as the OAI-PMH service or FTP archives) to programmatically obtain articles [3] [4]. This approach ensures compliance with PMC's policies on bulk retrieval. For initial development, targeted queries and on-demand downloads will be used; at full scale, the entire PMC Open Access corpus can be downloaded via the provided AWS Open Access endpoint for baseline processing [5].

LLM Model Deployment via OpenRouter

In place of a local Ollama-based LLM deployment, the design will leverage **OpenRouter** to access free large language models over the cloud. OpenRouter provides access to various models at no cost [6], eliminating the need to host models on local hardware. The environment will require an OpenRouter API key, after which the system can utilize available free models (e.g. DeepSeek, Llama2 variants, etc.) through OpenRouter's unified API. This change significantly reduces infrastructure complexity and cost, though it relies on an internet connection for LLM queries. *(If needed, the system can be configured to fall back to local models in the future, but the primary plan is to use OpenRouter's cloud models.)*

PDF Retrieval and Manual Import

For each candidate paper, the system's ingestion process will handle PDF retrieval in an interactive manner. The user should be prompted to confirm whether to download the article PDF automatically (using the PMC API or direct download links if available), or to supply a PDF manually. This confirmation step ensures that paywalled or non-open-access papers can be handled by manual upload if required, and avoids unnecessary downloads. By giving the user control over PDF acquisition, we maintain flexibility and compliance with any access restrictions. Once obtained (either via download or user upload), PDFs are processed through the pipeline (Stage 1 with GROBID and table extractors as described).

Processing of CSV Data Inputs

Multiple CSV files are involved as input/output in the development workflow, and the system will include methods to process each: - **Clinician Input Template**: This CSV (provided to the clinician) contains a list of papers and predefined columns for information to extract. The system can use this as a reference for what data points (columns) are of interest. Each row in this template corresponds to a paper. - **Clinician-Provided**

Output: This is the completed CSV filled by the clinician with extracted data (considered the ground truth). The pipeline will use this for validation (Stage 7) by comparing automated extraction results to the clinician's data. Notably, in this ground truth file each row currently corresponds to a paper (which might describe multiple patients). The system will include a preprocessing step to split or normalize entries so that each patient case is represented individually, aligning with the patient-level focus of the extraction. All columns in this file are treated as ground truth fields for evaluation purposes. - **Lens.org Metadata Export**: An example export from Lens.org is available, listing the relevant papers with structured metadata (e.g. titles, authors, DOIs, etc.). This serves as a reference for how literature metadata can be organized and what key identifiers to capture. The system can ingest this file to obtain a list of target paper identifiers (PMIDs/DOIs) and cross-check metadata. It ensures that when retrieving articles and linking data, each paper is correctly identified (via DOI or PMID) and helps automate matching the clinician's records with the retrieved literature. The Lens export is primarily a guide and validation resource; core metadata retrieval will ultimately be handled via PMC/PubMed APIs, but having this file helps verify completeness and consistency of the literature set.

Required External Files & Resources

Several external resources must be obtained and prepared in advance for the system to function: - **Human Phenotype Ontology (HPO)** – Download the latest HPO ontology file (hp.obo format) [7] for phenotype normalization. This will be used in Stage 5 for mapping extracted phenotype terms to standardized HPO codes. - **Mondo Disease Ontology** – Download the Mondo ontology file (mondo.obo or OWL format) [8] for disease normalization (Stage 5). MONDO provides a unified disease identifier space integrating OMIM, Orphanet, etc. - **SNOMED CT** – Obtain the SNOMED CT clinical terminology dataset for comprehensive clinical concept coverage. SNOMED CT is not openly downloadable; a UMLS license or a SNOMED International affiliate license is required to access it [9]. The user will need to download the SNOMED CT release files (e.g., the International Edition) via official channels (such as through the UMLS Terminology Services or SNOMED member portal). If an automated fetch is desired, it must be done through authorized APIs or by prompting the user to provide the files, due to licensing constraints. - **UMLS Metathesaurus (optional)** – If using tools like MedCAT that rely on a comprehensive concept database, the UMLS dataset (which includes SNOMED CT and other terminologies) should be prepared. This also requires a UMLS account/license. - **Initial Literature Dataset** – The list of target publications (e.g., from the Lens export or a curated list of PMIDs/DOIs) should be assembled. This can be stored as a CSV or text file that the system will use in Stage 0 to seed the search/triage process (in addition to automated search queries). - **Clinician-curated Data CSV** – The ground truth dataset provided by the clinician (described above) should be available to the system for use in evaluation and for any initial training or rule-setting (for example, to see examples of how data is structured).

In summary, before running the pipeline, the ontologies (HPO, Mondo, etc.) must be loaded into the system, and the necessary input datasets (list of papers and ground truth extractions) should be placed in the designated input directory.

Virtual Environment and Requirements

A dedicated Python virtual environment will be used to manage dependencies. Key libraries and tools will be installed as specified in a `requirements.txt`. Below is an indicative list of requirements (versions can be adjusted as needed):

```
numpy
pandas
requests
beautifulsoup4                # for any HTML parsing if needed
lxml                          # for XML parsing (e.g., PMC XML or GROBID
output)
PyMuPDF                       # for PDF text extraction if needed as backup
camelot-py[cv]                # for table extraction from PDFs (requires
OpenCV)
spacy
scispacy
medcat                        # medical concept annotation tool (UMLS-based)
transformers                  # Hugging Face Transformers for BERT models
torch                         # PyTorch for model inference
neo4j                         # Neo4j Python driver
psycopg2-binary               # PostgreSQL driver
matplotlib                    # (optional, for plotting metrics)
seaborn                       # (optional, for plotting/reporting)
```

(The above list is not exhaustive; additional specialized packages or model files might be required for the full extraction suite – for example, a time expression tagger for temporal data, or libraries for agent frameworks. These will be added as development progresses.)

Other non-Python prerequisites include: - **GROBID** – set up as a separate service (Java-based). This can be run via Docker or as a standalone server for PDF to XML conversion. - **n8n** – the workflow orchestrator, which can be run via Docker or installed directly, to coordinate the multi-step pipeline and agent triggers. - **Neo4j and PostgreSQL** – database systems which should be installed or containerized and running, with credentials configured for the pipeline to connect.

Additionally, an example environment file (`.env.example`) will be provided to illustrate required environment variables. For instance:

```
OPENROUTER_API_KEY="<YOUR_OPENROUTER_API_KEY>"
NEO4J_URI="bolt://localhost:7687"
NEO4J_USER="neo4j"
NEO4J_PASSWORD="<YOUR_NEO4J_PASSWORD>"
POSTGRES_URI="postgresql://user:pass@localhost:5432/yourdb"
```

The user should create their own `.env` file (or equivalent configuration) with the actual secrets and connection details, and the code will load these (for example, to authenticate to OpenRouter or to connect to databases).

Architecture Schema Diagram

The overall system architecture is illustrated in the figure above, which outlines the flow from data sources to extracted knowledge. To summarize, the process begins with a **Metadata Triage** phase (Stage 0) where a list of candidate papers is compiled (from queries or an input list) and filtered down by relevance (using abstracts and concept density). These papers then move into **Document Ingestion** (Stage 1), where full-text PDFs are acquired (via PMC downloads or user-provided PDFs) and converted to structured formats (using GROBID for text and layout, and tools like Camelot for tables).

Next, the pipeline performs **Advanced Table Analysis** (Stage 2) to interpret and standardize any tabular data in the papers, followed by **Patient Entity Resolution** (Stage 3) to ensure that each patient case is uniquely identified even across multiple mentions or documents. The core of the system is the **Comprehensive Concept Extraction** stage (Stage 4), where a suite of NLP models and algorithms extract entities such as phenotypes, genes, variants, clinical measurements, treatments, outcomes, temporal information, and more from the text. These raw extractions are then passed through **Multi-Ontology Normalization & Enrichment** (Stage 5), leveraging resources like HPO and MONDO to normalize terms to standard codes and enrich the data with additional context (e.g., mapping a colloquial disease name to a MONDO ID that ties together OMIM and Orphanet references).

After extraction and normalization, the information is integrated into a structured **Knowledge Graph and Database** (Stage 6). Here, each patient is a node in Neo4j with connections to various extracted concepts (phenotypes, genotypes, diagnoses, treatments, etc.), and simultaneously the data is stored in relational form in PostgreSQL for flexible querying. Every data point is linked back to its source paper (and even specific section or table, if needed) to maintain provenance.

Once the data is assembled, the system enters **Quality Assurance** (Stage 7). The extracted patient data can be automatically compared to the clinician-provided gold standard CSV. Metrics like precision, recall, and F1-score can be computed for each field to quantify performance, identifying where the AI extraction meets or falls short of human extraction [2]. Any discrepancies can be flagged for review. This stage may also include a manual review interface (e.g., using MedCAT Trainer) to allow a human expert to correct errors; these corrections can then feed into the **Active Learning Loop** (Stage 8) to retrain models and improve the system over time.

Throughout the process, the **Agent Orchestration Layer** (Track D) using Agent Zero and the **n8n workflow engine** ensures that each step flows into the next. The agent(s) handle logic such as deciding which extraction tools to apply, when to ask for user input (for example, confirming a PDF download or a ambiguous data point), and how to recover from errors. With the integration of OpenRouter for LLM capabilities, these agents can also call on powerful language models when complex reasoning or summarization is needed (for instance, summarizing a lengthy case report, or interpreting subtle implications in clinical text). The orchestrator manages parallelization (processing multiple papers concurrently) and monitors the overall pipeline, providing a high-level interface to start, stop, or inspect the process at any time.

In essence, the architecture involves multiple modules working in sequence and in tandem: ingesting data, extracting and normalizing knowledge, storing it, and constantly validating and improving. This modular design allows each component to be developed and tested independently, while the agent framework and workflow engine tie everything together into a cohesive automated system.

Project File Structure

For clarity and maintainability, the project will adhere to a clear directory structure. Below is a proposed structure for the repository:

```
project-root/
├── data/
│   ├── raw/                  # Raw input files (CSVs, ontology files, PDFs,
etc.)
│   ├── interim/              # Intermediate data outputs (e.g., GROBID XML,
parsed tables)
│   └── output/               # Final output files (extracted patient data,
reports)
├── models/                   # Pre-trained or fine-tuned model files (if stored
locally)
├── src/                      # Source code for the project
│   ├── triage/               # Stage 0: metadata retrieval & filtering scripts
│   ├── ingestion/            # Stage 1: PDF download/parsing scripts
│   ├── extraction/           # Stage 4: NLP extraction modules (NER, relation
extraction, etc.)
│   ├── normalization/        # Stage 5: ontology normalization and enrichment
code
│   ├── resolution/           # Stage 3: patient entity resolution code
│   ├── analysis/             # Stage 2 and other analytical scripts (table
analysis, metrics, etc.)
│   ├── agents/               # Agent Zero logic, definitions for autonomous
agents
│   ├── orchestration/        # n8n workflow files or orchestration scripts
│   ├── utils/                # Utility functions (e.g., API clients, file
handlers)
│   └── app.py                # Main script to run the entire pipeline (could
also be a CLI entry point)
├── notebooks/                # (Optional) Jupyter notebooks for prototyping or
data analysis
├── config/
│   ├── config.yaml           # Configuration file for pipeline settings (paths,
thresholds, etc.)
│   └── logging.conf          # Logging configuration
├── tests/                    # (Optional) Automated tests for various
components
├── requirements.txt          # Python dependencies for the project
├── .env.example              # Example environment variables for the project
└── README.md                 # Documentation and usage instructions
```

This structured layout ensures that data, models, and code are organized by function. For example, separating the extraction code from normalization code makes it easier for developers to focus on one

stage of the pipeline at a time. Configuration and documentation files are provided to simplify setup for new users or contributors. By preparing this file structure upfront, we facilitate a smooth development process and make the project scalable and collaborative.

⁂

---

[1] [2] Automated Biomedical Knowledge Extraction System_.md
file://file-Xask2knEWjkfLwMKcVLv2U

[3] [4] [5] For Developers - PMC
https://pmc.ncbi.nlm.nih.gov/tools/developers/

[6] OpenRouter Models
https://docs.typingmind.com/manage-and-connect-ai-models/openrouter-models

[7] GitHub - obophenotype/human-phenotype-ontology: Ontology for the description of human clinical features
https://github.com/obophenotype/human-phenotype-ontology

[8] Download -
http://mondo.monarchinitiative.org/pages/download/

[9] Knowledge Article · NLM Customer Support Center
https://site-bf31cb1c-ac53-4cc2-9ad4-db81f24f361d-gcv.azurewebsites.us/kbArticle/?pn=KA-04087