SOFTWARE ENGINEERING PROJECT REPORT

**A GENAI POWERED SMART RESEARCH ASSISTANT FOR DELIVERING ALL-PURPOSE END TO END RESEARCH PAPERS**

**TEAM :** Research Minds

**PROJECT NAME :** SMART RESEARCH ASSISTANT

**Team Members:**

Arkapravo Mandal (42) - 12023052002043

Atreyee Das (44) - 12023052002045

Ayan Chattopadhyay (46) - 12023052002047

Deboparna Das (48) - 12023052002049

# Abstract

The rapid expansion of academic research has created a pressing need for automated, intelligent systems capable of assisting researchers in synthesizing, validating, and generating scholarly content. The developed MVP — Smart All-Purpose Research Assistant (R-Agents) — is a unified, GenAI-driven platform designed to automate the end-to-end process of academic paper creation. Built with a FastAPI backend, React + Tailwind CSS frontend, and PostgreSQL database, the system integrates twelve AI-powered modules that perform tasks such as literature survey generation, research gap detection, methodology building, citation validation, translation, and LaTeX-ready report creation. The platform features a switchable LLM orchestration layer, allowing seamless toggling between OpenAI GPT-4o and Gemini 1.5-Pro, ensuring adaptability, cost-efficiency, and high-quality output. By combining modular AI workflows with user-friendly design, R-Agents enables students, scholars, and educators to complete research documentation efficiently and ethically within a single digital ecosystem.

# Literature Review

The Smart All-Purpose Research Assistant builds upon developments in Generative AI, automated text understanding, and academic content retrieval. This section reviews the major research directions and technologies that influenced the architecture and design of the developed MVP.

*1. AI-Powered Academic Information Retrieval*

Recent advances in academic information retrieval have focused on combining scholarly APIs and machine learning to enhance literature accessibility. Systems such as Semantic Scholar and OpenAlex provide structured metadata and abstracts, while CrossRef offers DOI-based linking for citation validation. Unlike earlier retrieval-augmented generation frameworks, the proposed system adopts a direct retrieval and synthesis approach — dynamically querying these databases through REST APIs and summarizing results via LLMs. This allows real-time access to open-access papers without relying on complex vector search or embedding-based retrieval pipelines.

*2. Modular and Scalable Backend Design*

Traditional research automation tools often rely on monolithic AI frameworks or external orchestration engines. In contrast, the developed MVP adopts a modular FastAPI-based architecture, where each research feature — such as literature summarization, research gap identification, or methodology generation — is implemented as an independent microservice module. This design enhances maintainability, enables concurrent task execution, and ensures scalability without introducing the computational overhead of agent-based reasoning.

*3. Integrated Citation Validation and Reference Management*

Accurate citation handling remains a challenge in AI-generated academic text. Prior systems like Zotero or Scite.ai manage references separately, without integration into content generation. The proposed system embeds citation verification directly into the generation process, cross-referencing DOIs and metadata from CrossRef and OpenAlex. This ensures that all generated references are verifiable and conform to academic formatting standards such as APA or IEEE, enhancing both credibility and usability.

*4. Multilingual and Cross-Domain Research Support*

Globalization of research requires cross-lingual accessibility and interdisciplinary synthesis. The implemented MVP integrates multilingual translation and summarization using GPT-4o and Gemini 1.5-Pro APIs. This enables users to upload papers in foreign languages and obtain translated and contextually accurate summaries. Furthermore, the Cross-Domain Synthesizer module applies these models to discover links between unrelated domains — for example, connecting medical and AI research — encouraging interdisciplinary collaboration.

*5. Persona-Based Summarization and Contextual Adaptation*

Contemporary research communication increasingly values adaptive summarization. Instead of generic content generation, the system’s Persona-Based Summarizer tailors summaries to specific audiences (students, reviewers, educators, or policymakers). By leveraging the instruction-following capability of LLMs, the module modifies language complexity, focus, and length to match the reader’s role, improving comprehension and engagement.

*6. Web Integration and Asynchronous Processing*

Many early academic automation platforms were limited by synchronous, single-threaded designs or required frameworks like Streamlit. The present MVP integrates a React + Tailwind CSS frontend with a FastAPI asynchronous backend, ensuring smooth real-time interaction. Each API call runs independently, allowing multiple modules to operate concurrently (e.g., translation and citation validation). Environment management through .env and database integration via PostgreSQL ORM guarantee stability, security, and ease of deployment across environments.

**Feasibility Study**

*1. Technical Feasibility*

The MVP leverages mature, publicly available technologies.

* Backend: FastAPI with modular routes and async handling.
* Frontend: React + Tailwind CSS ensuring fast UI rendering.
* Database: PostgreSQL for relational data with secure ORM-based access.
* AI Layer: Dual-LLM configuration (GPT-4o / Gemini 1.5-Pro) selected via a central utility.  
  All these technologies are lightweight, open-source, and proven for research and enterprise deployment, making the system technically robust and scalable.

*2. Operational Feasibility*

The platform supports multiple user roles — Student, Researcher, Reviewer, and Faculty — each accessing a tailored interface. REST endpoints ensure interoperability with other academic systems. All modules execute autonomously yet communicate through a shared user session, enabling full workflow automation from problem definition to formatted output.

*3. Economic Feasibility*

The entire solution is cost-efficient:

* All frameworks are open-source.
* Model APIs (OpenAI or Gemini) operate on pay-per-use, manageable for academic deployment.
* PostgreSQL and FastAPI run effectively on low-spec local or cloud systems.  
  Hence, institutions and individuals can deploy this MVP without expensive infrastructure.

*4. Legal and Ethical Feasibility*

The system restricts data retrieval to open-access repositories. User content remains private in the database, and no sensitive information leaves the local environment. Each generated text includes verifiable citations, adhering to ethical AI and responsible research principles.

# Requirement Analysis

The following requirements were defined for developing the MVP version of R-Agents.

*Functional Requirements*

| **Functionality** | **Description** |
| --- | --- |
| Literature Survey Generator | Retrieves, filters, and summarizes papers using Semantic Scholar, OpenAlex, and CrossRef APIs. |
| Research Gap Finder | Identifies underexplored themes and limitations within gathered papers. |
| Methodology Builder | Generates structured procedural descriptions from problem statements. |
| Citation Validator | Validates citations in real time against CrossRef metadata. |
| Persona Summarizer | Produces summaries customized for different audience roles. |
| Translator | Provides multilingual paper translation using the LLM layer. |
| Contradiction Analyzer | Detects inconsistencies between paper claims and retrieved evidence. |
| Cross-Domain Synthesizer | Suggests interdisciplinary extensions of current work. |
| Benchmark Explorer | Recommends relevant datasets, metrics, and evaluation trends. |
| LaTeX Generator | Converts plain or markdown text into publication-ready .tex files. |
| Experiment Replicator | Suggests reproducible experiment setups based on existing works. |
| Voice/Text I/O | Allows speech input and TTS output for accessibility. |

*Hardware Requirements*

* CPU: 4+ cores (Intel i5/Ryzen 5 or better).
* RAM: Minimum 8 GB (16 GB recommended).
* Storage: ≥ 20 GB free space for caching and exports.
* Network: Stable internet connection for live retrieval and LLM calls.
* Optional GPU: NVIDIA GPU with CUDA for accelerated inference.

*Non-Functional Requirements*

| Property | Description |
| --- | --- |
| Scalability | Horizontal scaling via modular architecture. |
| Portability | Deployable on local, cloud, or Docker containers. |
| Security | JWT-based authentication with hashed passwords. |
| Usability | Minimal-click UI with React-based design. |
| Extensibility | Easily add new feature modules. |

*User Requirements*

Each role experiences customized access:

* Students: Simplified summaries and tutorials.
* Researchers: Full access to literature, methodology, and citation tools.
* Reviewers: Tools for contradiction detection and validation.
* Faculty/Admin: Oversight features and project dashboards.

*System Constraints*

| **Aspect** | **Description** |
| --- | --- |
| Framework | FastAPI, React, PostgreSQL |
| Input Support | Text, PDF, and speech input |
| Processing Time | ≤ 5 s for text queries under 1 000 words |
| Model Hosting | Local or API-based |
| Output Format | JSON responses, downloadable .tex and .pdf |

# System Architecture

The system architecture retains a layered modular structure.

*1. Layered Architecture*

* Frontend (UI Layer): Built using React and Tailwind CSS for responsive, role-specific interaction.
* Backend (Application Layer): FastAPI orchestrates routes, processes inputs, and calls feature modules.
* AI/LLM Layer: Manages GPT-4o and Gemini 1.5-Pro via centralized orchestration in llm\_utils.py.
* Database Layer: PostgreSQL stores users, projects, and AI-generated outputs.
* File Management Layer: Handles .tex exports, media uploads, and retrieval.

*2. Component Details*

*Retrieval & Processing*

| **Tool / Module** | **Purpose** |
| --- | --- |
| Semantic Scholar / OpenAlex / CrossRef / Unpaywall APIs | Retrieve open-access research papers, abstracts, and citation metadata directly through REST endpoints. |
| Requests & AsyncIO (FastAPI) | Handle concurrent API calls efficiently for large-scale literature fetching and aggregation. |
| Text Pre-Processor | Cleans, tokenizes, and structures retrieved text for LLM input without external embedding or vector indexing. |
| Citation Parser | Extracts DOI, title, author, and publication details from metadata to ensure traceable references. |

*Generation & Reasoning*

| **Tool / Model** | **Purpose** |
| --- | --- |
| GPT-4o (OpenAI) | Primary large-language model for summarization, literature synthesis, and citation-aware drafting. |
| Gemini 1.5-Pro (Google GenAI) | Alternative model providing multilingual translation, summarization, and contextual reasoning. |
| LLM Utils Module (llm\_utils.py) | Central orchestration layer that switches between GPT-4o and Gemini based on configuration for cost- or accuracy-optimized processing. |
| Prompt Templates & System Roles | Predefined structured prompts for literature review, gap analysis, and methodology generation tasks. |

*Data Sources*

| Corpus / API | Use Case |
| --- | --- |
| Semantic Scholar API | Abstracts, keywords, and citation links for general research topics. |
| OpenAlex API | Metadata-rich scholarly graph for author, journal, and institution mapping. |
| CrossRef API | DOI verification and reference formatting. |
| Unpaywall | Identification of freely accessible full-text papers. |
| User Uploads (PDF / Text) | Local research papers provided by users for summarization or translation. |

*Tooling & Middleware*

| **Tool / Platform** | **Functionality** |
| --- | --- |
| Python 3.10 + FastAPI | Core backend for feature-specific routes and asynchronous processing. |
| React + Tailwind CSS | Responsive and interactive frontend for real-time visualization and editor access. |
| PostgreSQL + SQLAlchemy ORM | Persistent storage for user profiles, project data, and generated outputs. |
| Whisper (OpenAI) | Speech-to-text input for accessibility and verbal command entry. |
| TTS (Edge / Coqui) | Converts generated summaries or explanations into audio feedback. |
| Docker (Planned) | Portable containerization for cloud or on-premise deployment. |
| .env Configuration | Secure storage of API keys and database credentials for OpenAI, Gemini, and external scholarly APIs. |

## UML Class Diagram

A diagram of a company

AI-generated content may be incorrect.

Figure 1: UML Class Diagram

## Decision Tree

A diagram of a user flow

AI-generated content may be incorrect.

Figure 2: Decision Tree

## Decision Table

A blue and white checklist

AI-generated content may be incorrect.

Figure 3: Decision Table

## Use Case Diagram

Several rows of circles

AI-generated content may be incorrect.

Figure 4: Use Case Diagram

## Architecture DiagramA diagram of a smart all purpose research system AI-generated content may be incorrect.

Figure 5: Architecture (Working Diagram)

## Data Flow Diagram (Level 0)

A diagram of a system

AI-generated content may be incorrect.

Figure 6: Data Flow Diagram (Level 0)

## Data Flow Diagram (Level 1)

A diagram of data flow

AI-generated content may be incorrect.

Figure 7: Data Flow Diagram (Level 1)

## Activity Diagram

A diagram of a diagram

AI-generated content may be incorrect.

Figure 8: Activity Diagram

## Sequence Diagram

A diagram of a sequence diagram

AI-generated content may be incorrect.

Figure 9: Sequence Diagram

## ER Diagram

A diagram of a company structure

AI-generated content may be incorrect.

Figure 10: ER Diagram

# Testing Reports

A screenshot of a computer

AI-generated content may be incorrect.

Figure 11: Login Page and Dashboard

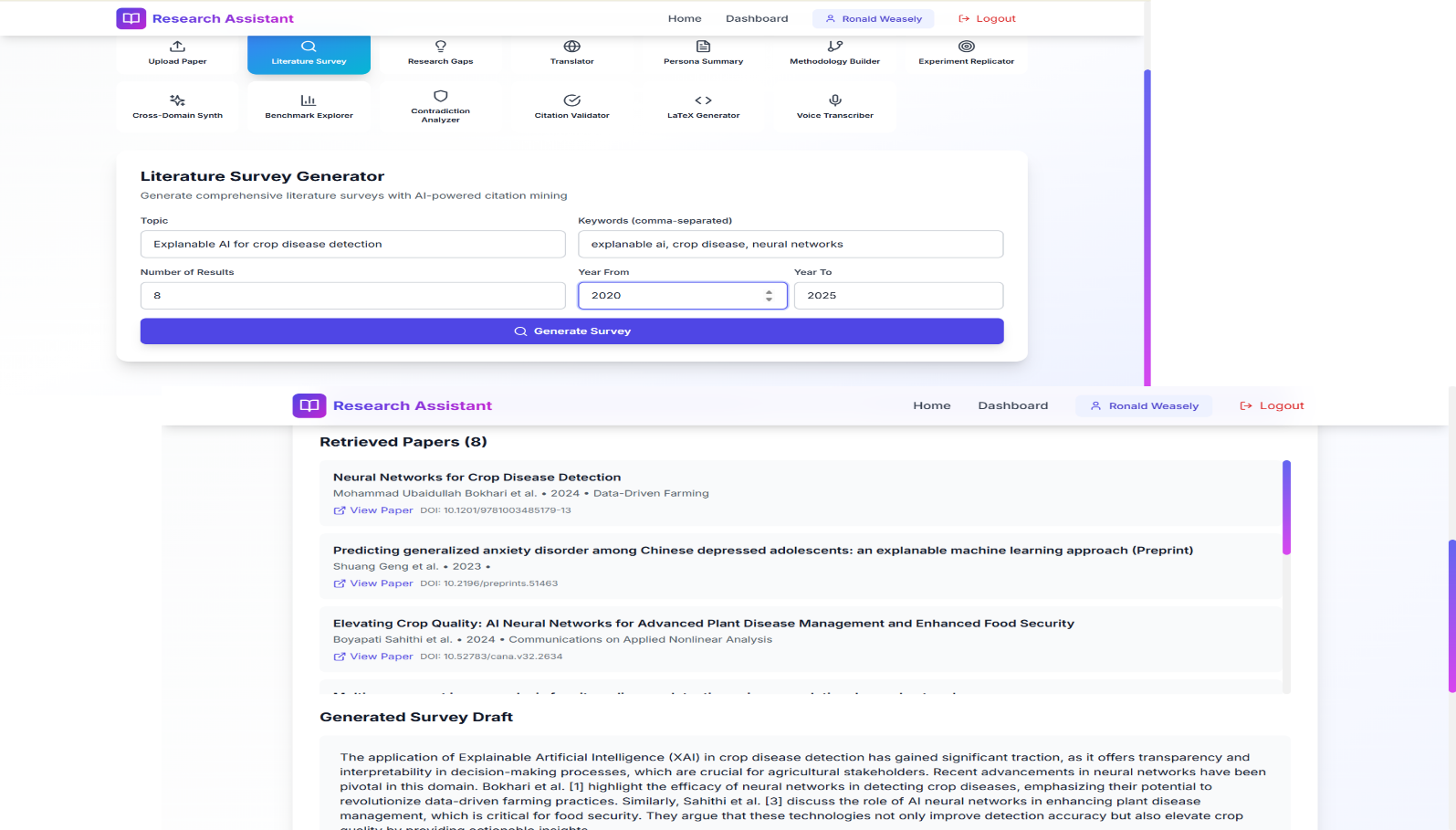


Figure 12: Literature Survey Generator Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 13: Research Gap Finder Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 14: Persona Summarizer Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 15: Multi Language Translator Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 16: Cross Domain Synthesizer Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 17: Contradiction Analyzer Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 18: Citation Validator Page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 19: Latex Generator Page