

AI Research Paper Review & Summarization System

15 Days Internship Project Report Submitted to Infosys Springboard

Submitted by:

Name: Tanuj Kumar

Email: tanujkumar9995@gmail.com

Under the guidance of

Mentor Name: Likhita

Designation / Infosys Springboard

AI Research Paper Review & Summarization System

1. Introduction / Objective

The objective of this project is to design and implement an **AI-based system** that automatically reviews, compares, and summarizes academic research papers.

The system reduces the manual effort required for literature review by automating paper retrieval, content analysis, section-wise comparison, and structured summarization.

The system supports **two modes of input**:

1. **Automatic mode** – fetches research papers using the Semantic Scholar API
2. **Manual mode** – allows users to upload or place their own research PDFs for analysis

Large Language Models (LLMs) powered by **Google Gemini API** are used to perform semantic analysis, comparison, and generation of structured summaries.

2. System Overview

The system follows a **modular and pipeline-based architecture**, where each stage of the research workflow is handled by a dedicated module.

It processes up to **three research papers** at a time and generates a final reviewed output consisting of:

- Abstract (≤ 100 words)
- Methodology comparison
- Results synthesis
- Key insights
- APA-formatted references

The final output is displayed in a **Gradio web interface** and also stored locally for future use.

3. Methodology / Workflow

Phase 1: Research Phase

- Topic planning and scoping using the Planner module
- Automated paper search via **Semantic Scholar API**
- Smart paper selection (up to 3 open-access papers)

- Automatic PDF download

Phase 2: Analysis Phase

- PDF text extraction using **PyMuPDF4LLM**
- Section-wise text processing
- Identification of key objectives, methods, and results
- Cross-paper comparison

Phase 3: Writing Phase

- AI-generated structured content:
 - Abstract (100 words)
 - Methods comparison
 - Results synthesis
 - Key insights
- Reference formatting using APA style

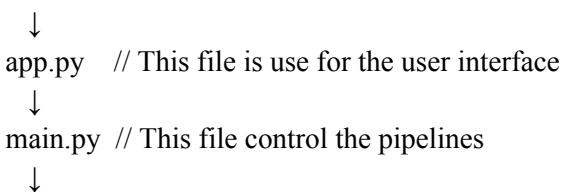
Phase 4: Review Phase

- Quality assessment of generated content
- Language refinement and coherence checks
- Final draft preparation

4. System Architecture

The project follows a layered modular architecture at this phase of project:

User (Gradio UI)



Planner → Paper Retrieval → PDF Downloader → Text Extractor → Analyzer → Draft Generator → Reviewer



The architecture ensures:

- Clear separation of responsibilities
 - Easy debugging and extension
 - Maintainable and scalable codebase
-

5. Module-wise Description

1. Planner Module (`planner.py`)

- Refines and scopes the research topic
- Prepares structured prompts for downstream modules

2. Paper Retrieval Module (`paper_retrieval.py`)

- Uses Semantic Scholar API
- Fetches metadata and open-access PDF links
- Limits results to the top relevant papers

3. PDF Downloader Module (`pdf_downloader.py`)

- Downloads PDFs securely
- Validates file type and content

4. Text Extraction Module (`text_extractor.py`)

- Extracts raw text from PDFs using PyMuPDF4LLM
- Handles multi-page documents

5. Analyzer Module (`analyzer.py`)

- Performs section-wise comparison across papers
- Identifies similarities, differences, and key findings
- Uses Gemini LLM for semantic reasoning

6. Draft Generator Module (`draft_generator.py`)

- Generates structured sections:
 - Abstract
 - Methods comparison
 - Results synthesis
 - Key insights

7. Reviewer Module (`reviewer.py`)

- Refines the generated draft
- Improves clarity, academic tone, and coherence

8. APA Formatter (`utils/apa_formatter.py`)

- Formats references in APA style using paper metadata
-

6. Technology Stack

Programming Language

- Python 3.x

Libraries / Frameworks

- Gradio (UI)
 - Semantic Scholar API
 - Google Gemini API
 - LangChain (prompt orchestration)
 - LangGraph / Grandalf (workflow modeling – conceptual)
 - PyMuPDF4LLM (PDF text extraction)
 - Pydantic (data validation)
-

7. Features Implemented

- Automatic and manual paper input modes
- Section-wise comparison of multiple research papers

- AI-generated structured summaries
 - APA-formatted references
 - Local storage of final summaries
 - Clean GitHub-ready project structure
-

8. Output Storage

All generated summaries are stored in:

data/outputs/

This ensures:

- Reproducibility
 - Offline access to results
 - Easy documentation and submission
-

9. Challenges Faced

- API rate limits (Gemini & Semantic Scholar)
- Handling missing or invalid PDFs
- Managing token limits for large research papers
- Ensuring consistent section-wise comparison

These challenges were handled through:

- Input validation
 - Text truncation strategies
 - Robust error handling
-

10. Conclusion

This project successfully demonstrates how **AI** can automate complex academic tasks such as research paper review and summarization.

The system is modular, scalable, and suitable for real-world academic and industry use cases.

It aligns well with the **Infosys Springboard internship objectives** by showcasing skills in:

- AI/LLMs
 - API integration
 - Modular system design
 - End-to-end application development
-

11. Future Enhancements

- Support for more than three papers
- Graph-based workflow visualization
- Citation cross-validation
- Export summaries as PDF/Word documents
- Cloud deployment