**Research Literature Review Agent**

**Problem Statement:**

Researchers struggle to keep up with the growing volume of academic literature. Manually finding, reading, and synthesizing relevant papers for reviews is time-consuming and inefficient.

**Goal:**

Develop an intelligent research assistant agent that can:

* Take a research topic as input
* Search academic databases (e.g., Semantic Scholar, arXiv)
* Retrieve and summarize key papers
* Present concise literature reviews or annotated bibliographies

**Scope:**

* Search and retrieval of academic abstracts via APIs
* Summarization of abstracts or conclusions using LLMs
* Literature organization and annotation
* Simple web/chat interface

**Further Improvement:**

* + Full paper PDF scraping (if restricted)
  + Citation graph visualization
  + Topic-specific fine-tuning of LLMs

**Key Lean Metrics:**

* Time saved compared to manual reviews
* Relevance of summaries (measured via feedback)
* Number of papers retrieved and summarized per query
* Accuracy of topic filtering

**Sprint Plan with User Stories:**

**Sprint Duration:** 1 week

**Team:**

AI Engineer, Backend Developer/Data Engineer, Test Engineer, Frontend/UI Developer

**Sprint 1: Topic Input & Data Retrieval**

**Goal:** Allow users to input a research topic and retrieve relevant papers.

**User Stories:**

* A user should be able to input a research topic and retrieve relevant academic content.
* AI Engineer should integrate the Semantic Scholar or arXiv API to fetch paper titles and abstracts.
* Product owner should define key fields (e.g., topic, publication year) for paper filtering.

**Deliverables:**

* Query input UI
* Backend integration with at least one academic search API
* Paper metadata (titles, abstracts) stored or cached

**Sprint 2: LLM-Based Summarization**

**Goal:** Generate summaries of retrieved papers using LLMs.

**User Stories:**

* AI engineer should apply a transformer-based model (e.g., BART, Pegasus) to summarize abstracts so users can review content quickly.
* AI Engineer should automate the pipeline from retrieval to summary, so the user gets output in one step.
* Tester validates summary quality for relevance and conciseness.

**Deliverables:**

* Summarization module using pre-trained models
* Auto-generated summary output for each abstract
* Quality test set and baseline metrics

**Sprint 3: Organize and Annotate Literature**

**Goal:** Structure summaries into a usable literature review format.

**User Stories:**

* A user would want the agent to organize papers by subtopic or relevance so that trends or clusters can be understood
* AI Engineer should group papers and generate an outline of the review for easier consumption.
* Frontend developer should generate a clear, readable interface showing organized outputs.

**Deliverables:**

* Topic-based grouping of summarized papers
* Annotated bibliography or literature review format
* Display interface with filtering options

**Sprint 4: Testing, Refinement, and Review**

**Goal:** Ensure performance, gather feedback, and finalize MVP.

**User Stories:**

* Test Engineer should test the system with 3–5 different research topics to ensure consistency.
* From a researcher/test user receive feedback on summary quality and organization.
* AI Developer should address bugs or errors and optimize summary pipelines

**Deliverables:**

* Functional feedback form and user testing results
* Polished user experience and summary formatting
* Final MVP(Minimum Viable Product) demo with 1-click topic-to-review flow

**Continuous Improvement Plan**

* Integrate citation graphs and trend analysis
* Add PDF parsing for full paper summaries (where allowed)
* Enable real-time monitoring of new publications in selected fields
* Explore LLM-based hypothesis generation or keyword extraction