Research Paper Summarization System

-Software Requirements Specification Document

Revision history

Version	Date	Author	Description of changes
1.0	Oct 23, 2025	S V S Sai Sreenivas	Initial version of SRS document

1. Introduction

1.1 Purpose

The purpose of the Research Paper Summarization System is to provide an intelligent, automated platform that assists users in understanding and analyzing research papers quickly and accurately.

The software enables users to:

- Upload or discover research papers related to a chosen topic.
- Automatically generate extractive, abstractive, and factually refined summaries using advanced natural language processing (NLP) models.
- Choose between Project Research (technical and concise) or Study Research (simplified and conceptual) summary modes.
- Organize, pin, and manage summaries through a personalized workspace.

By combining machine learning with web automation and text processing, the system minimizes the time and effort required for literature review and helps users focus on critical insights rather than manual reading.

1.2 Scope

The Research Paper Summarization System aims to simplify and accelerate the research process by integrating automated summarization with intelligent paper discovery.

Key objectives include:

- Enabling users to upload files or enter a topic of interest to find research papers.
- Using a multi-phase ML pipeline to create structured, factually accurate summaries:
    - Extractive summarization (sentence selection using Graph Attention Networks).
    - Abstractive summarization (rewriting via PEGASUS).
    - Fact-based refinement (FactEditor validation loop).
- Allowing users to store, view, and manage summaries through an intuitive web dashboard.

The software operates as a standalone web-based system deployable on cloud infrastructure

1.3 Overview

The system combines deep learning, web automation, and natural language processing to summarize long research texts efficiently. It is designed as a modular web platform with a backend API, machine learning layer, and user-facing frontend.

Key technologies include FastAPI, PyTorch, Transformers, React.js, and PostgreSQL, ensuring scalability, performance, and security.

.

2. General description

2.1 Product Perspective

The system operates as a standalone web application composed of interconnected modules:

- **Frontend:** Web-based interface for user interactions (upload/search/summarize/view).
- **Backend API:** Manages authentication, job creation, summarization requests, and database communication.

- **Machine Learning Pipeline:** Handles extractive, abstractive, and fact-based summarization.
- **Database Layer:** Stores users, papers, summaries, and metadata.
- **External APIs:** Fetch research papers from arXiv, OpenAlex, or Semantic Scholar.
- **Storage Layer:** Manages uploaded PDFs and summary files via cloud storage (e.g., AWS S3).

2.2 Product Functions

Main capabilities include:

- **User Management:** Register, log in, or use guest access.
- **Document Upload:** Accept PDF, DOCX, and TXT formats.
- **Topic-Based Search:** Retrieve papers based on topic queries using research APIs.
- **Summarization:** Execute multi-stage summarization pipeline (extractive → abstractive → fact-based).
- **Evaluation:** Compute factual accuracy and ROUGE scores.
- **Folder Management:** Create folders and pin important summaries.
- **Asynchronous Processing:** Handle long tasks via Celery job queues.

Dependencies:

- Internet connectivity for API access and model inference.
- GPU-accelerated environment for transformer-based summarization.

2.3 Design Constraints

- File size limit: 50 MB per upload.
- Dependence on external APIs and their availability.
- Model inference latency: up to 60 seconds for large documents.
- Must comply with data privacy and open-access policies.
- Deployment in Dockerized GPU environment (AWS/GCP).

3. System features

Feature ID	Name of feature	Description
F1	User Registration & Login	Enables secure access and supports guest sessions.
F2	Upload Document	Uploads PDF/DOCX/TXT files for summarization.
F3	Topic-Based Search	Fetches related papers from APIs based on a user-provided topic.
F4	Extractive Summarization	Selects key sentences using Graph Attention Networks.
F5	Abstractive Summarization	Generates human-like summaries using the PEGASUS model.
F6	Fact-Based Refinement	Ensures factual accuracy using FactEditor.
F7	Summary Evaluation	Calculates ROUGE and factual consistency scores.
F8	Folder Management	Lets users save and organize summaries.
F9	Job Queue System	Manages concurrent summarization jobs efficiently.

4. Interface requirements

4.1 Functional requirements

- Users shall register, log in, or use guest mode.
- The system shall allow document uploads in PDF/DOCX/TXT.
- The system shall allow users to input topics to search for papers.
- The system shall fetch metadata from APIs (title, authors, abstract, year).
- The system shall fetch or download open-access PDFs.
- The system shall execute a three-phase summarization pipeline.
- The system shall evaluate summaries using ROUGE and factual scores.
- The system shall allow users to save and organize summaries in folders.
- The system shall display job status and progress for active summaries.
- The system shall log operations and failures.

4.2 Non-functional requirements

- Summaries generated in ≤ 60 seconds for standard papers; handles concurrent users through asynchronous task queues.
- Encrypted storage, HTTPS communication, JWT-based authentication.
- $\geq 99\%$ uptime; automatic retries for failed jobs.
- Horizontal scaling via container orchestration.
- Modular microservice architecture for easy updates.
- Automatic deletion of temporary files and user-controlled data removal.

4.3 External interface requirements

- **User Interfaces:**
- Responsive web UI for upload, topic search, summary view, and folder management.
- **Hardware and Software Interfaces:**
    - GPU-enabled servers for ML inference.
    - PostgreSQL and Redis databases.
    - REST APIs between frontend and backend.
- **External Systems:**
- Integration with research APIs (arXiv, OpenAlex, Semantic Scholar).

5. Other requirements

5.1 Legal requirements	● Must comply with copyright and data-sharing terms of external APIs. ● Must respect open-access and institutional data-use regulations. ● Handle user data according to privacy policies (GDPR-like standards).
5.2 Safety requirements	● Daily automated database backups. ● Auto-deletion of temporary guest uploads after 24 hours. ● Error handling and alerts for failed inference or long processing times.
5.3 Add other requirements	● Future support for multi-document summarization. ● Multilingual summarization capability. ● Export summaries in JSON or PDF format.

6. Appendices

Term/Acronym	**Definition**
GAT (Graph Attention Network)	Neural network used for ranking sentences by importance.
PEGASUS	Transformer model trained for abstractive summarization tasks.
FactEditor	Model used for fact-based summary refinement.
ROUGE	Metric evaluating summary similarity to reference text.
JSON	Lightweight data format for storing summaries.
API	Application Programming Interface, used to access external data.