**Software Requirements Specification**

**for**

**AI - Driven Knowledge Synthesis Platform**

**Version 1.0 approved**

**Prepared by**

**Umang Goswami (22100BTCSAII11062)**

**Shreyansh Gupta (22100BTCSAII11052)**

**Naman Mathur (22100BTCSAII11027)**

**Under the Guidance of**

**Prof. Neeraj Mehta**

**Department of Computer Science & Engineering**

**Shri Vaishnav Institute of Information Technology**

**Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore**

**March 6th, 2025**

Table of Contents

Revision History ii

1. Introduction 1

1.1. Purpose 1

1.2. Document Conventions 1

1.3. Intended Audience and Reading Suggestions 1

1.4. Product Scope 2

1.5. References 2

2. Overall Description 4

2.1. Product Perspective 4

2.2. Product Functions 4

2.3. User Classes and Characteristics 4

2.4. Operating Environment 4

2.5. Design and Implementation Constraints 5

2.6. User Documentation 5

2.7. Assumptions and Dependencies 5

3. External Interface Requirements 6

3.1. User Interfaces 6

3.2. Hardware Interfaces 6

3.3. Software Interfaces 6

3.4. Communications Interfaces 6

4. System Features 7

4.1. Multi-Format Document Ingestion 7

4.2. Cross-Document Knowledge Synthesis 7

4.3. Multimodal Output Generation 7

5. Other Non-functional Requirements 7

5.1. Performance Requirements 7

5.2. Safety Requirements 7

5.3. Security Requirements 7

5.4. Software Quality Attributes 7

5.5. Business Rules 7

Appendix A: Glossary 8

Appendix B: Analysis Models 9

Appendix C: To Be Determined List 10

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

This SRS defines the requirements for the AI-Driven Knowledge Synthesis Platform, an advanced system that automates the conversion of unstructured data (e.g., PDFs, research papers, spreadsheets, web articles) into structured, multimodal outputs (summaries, quizzes, flashcards, podcasts) using Retrieval-Augmented Generation (RAG) and NLP models.

**Key Objectives**

* Eliminate manual content curation inefficiencies in education and corporate training.
* Provide cross-document knowledge synthesis to generate cohesive insights.
* Deliver personalized outputs (e.g., summaries for educators, quizzes for students).
* Support scalable deployment on Microsoft Azure for institutional use.

## Document Conventions

* **Priority Levels:**
* **High (H):** Core functionality (e.g., document ingestion, RAG pipelines).
* **Medium (M):** Enhanced features (e.g., multimodal outputs).
* **Low (L):** Nice-to-have features (e.g., advanced analytics).
* **Requirement IDs:** REQ-XX (e.g., REQ-1 for document parsing).
* **TBD:** Placeholder for pending decisions (e.g., third-party API integrations).

## Intended Audience and Reading Suggestions

|  |  |
| --- | --- |
| **Audience** | **Focus Areas** |
| Developers | Sections 2–4 (System Architecture, Functional Requirements) |
| Testers | Section 4 (System Features) for test case derivation |
| Project Managers | Sections 1–2 (Scope, Dependencies) |
| Educators/Users | Section 5 (Nonfunctional Requirements) for usability and security |

## Product Scope

* **Problem Statement:**
* Current tools (e.g., Quizlet, Grammarly) address niche tasks but lack cross-document synthesis and multimodal output generation.
* Manual curation of training materials/reports is time-consuming (70% reduction goal).
* **Solution:**
  + Unified AI Pipeline: Combines RAG, NLP, and multimodal generation.
* **Use Cases:**
* **Education:** Auto-generate lecture slides, quizzes, and audio summaries.
* **Corporate Training:** Synthesize reports from spreadsheets, PDFs, and articles.

## References

|  |  |
| --- | --- |
| [1] | “Azure AI Search Documentation,” 25 February 2025. [Online]. Available: https://learn.microsoft.com/en-us/azure/search/. [Accessed 10 March 2025]. |
| [2] | “FAISS GitHub Wiki,” Meta AI Research, 24 February 2025. [Online]. Available: https://github.com/facebookresearch/faiss/wiki. [Accessed 10 March 2025]. |
| [3] | “Flask Documentation (3.1.x),” Pallets, 5 January 2025. [Online]. Available: https://flask.palletsprojects.com. [Accessed 10 March 2025]. |
| [4] | “LangChain Documentation,” LangChain, Inc., 30 January 2025. [Online]. Available: https://python.langchain.com/docs/introduction/. [Accessed 10 March 2025]. |
| [5] | “LangChain HuggingFace Integrations Documentations,” LangChain, Inc., 16 October 2024. [Online]. Available: https://python.langchain.com/docs/integrations/providers/huggingface/. [Accessed 10 March 2025]. |
| [6] | “LangChain Python API Reference,” LangChain Inc., 2 March 2025. [Online]. Available: https://python.langchain.com/api\_reference/. [Accessed 10 March 2025]. |
| [7] | “Gradio Documentation,” Gradio, 9 March 2025. [Online]. Available: https://www.gradio.app/docs. [Accessed 10 March 2025]. |
| [8] | “Python 3.13.2 documentation,” 10 March 2025. [Online]. Available: https://docs.python.org/3/. [Accessed 10 March 2025]. |
| [9] | “GitHub Docs,” GitHub, Inc., [Online]. Available: https://docs.github.com/en. [Accessed 10 March 2025]. |

# Overall Description

## Product Perspective

* **System Context:**
* Inputs: PDFs, PPTs, spreadsheets, web articles.
* Processing: Document Chunking → Vector Embeddings → RAG-Based Retrieval → Output Generation.
* **Outputs:** Summaries, quizzes, flashcards, podcasts.
* **Integration:**
* Frontend: Gradio (Python) + JavaScript for interactivity.
* Backend: LangChain, Hugging Face Transformers, FAISS.
* Deployment: Docker containers on Microsoft Azure.

## Product Functions

|  |  |  |
| --- | --- | --- |
| **Function** | **Description** | **Priority** |
| Multi-Format Ingestion | Parse PDFs, CSVs, HTML into structured text. | H |
| Semantic Embedding | Generate embeddings using LangChain Transformers. | H |
| Cross-Document Synthesis | Retrieve related content across documents via FAISS. | H |
| Dynamic Output Generation | Create summaries, quizzes, podcasts tailored to user roles. | M |

## User Classes and Characteristics

|  |  |
| --- | --- |
| **User Class** | **Characteristics** |
| Educators | Create course materials; need batch processing. |
| Students | Generate study aids; prefer mobile-friendly outputs. |
| Professionals | Synthesize reports; require compliance with corporate templates. |
| Administrators | Manage user access, monitor system performance. |

## Operating Environment

* **Backend**: Python 3.13, LangChain, Hugging Face Transformers.
* **Frontend**: Gradio, Chart.js for visualizations.
* **Database**: FAISS (vector storage), Azure SQL (metadata).
* **Deployment**: Azure Web Application.

## Design and Implementation Constraints

* **Data Privacy**: GDPR/FERPA compliance for educational data.
* **Performance**: <2s response time for queries.

## User Documentation

* User Manual: PDF + interactive in-app guide.
* API Docs: Swagger/OpenAPI for developers.

## Assumptions and Dependencies

* Assumption: Users have basic technical literacy to upload documents.
* Dependency: Microsoft Azure for cloud hosting.

# External Interface Requirements

## User Interfaces

* **Dashboard:** Role-based views (educator/student/professional).
* **Upload Portal:** Drag-and-drop with real-time preview.
* **Output Customization:** Sliders for summary length, quiz difficulty.

## Hardware Interfaces

* **Minimum:** 4GB RAM, 2GHz CPU (local testing).
* **Cloud:** Azure VMs with GPU support (NLP tasks).

## Software Interfaces

|  |  |
| --- | --- |
| **Component** | **Integration Purpose** |
| LangChain | Document chunking, RAG pipelines. |
| Hugging Face | Text generation (e.g., GPT-4 for summaries). |
| Azure Cognitive Services | Text-to-speech for podcasts. |

## Communications Interfaces

* APIs: RESTful endpoints (Flask backend).
* Security: OAuth 2.0 (user auth), TLS 1.3 (encryption).

# System Features

## Multi-Format Document Ingestion

1. **REQ-1:** Support PDF, CSV, PPTX, HTML uploads.
2. **REQ-2:** Auto-detect file type and extract text.

## Cross-Document Knowledge Synthesis

1. REQ-3: Hybrid search (keyword + vector similarity) via FAISS.
2. **REQ-4:** Metadata tagging for source attribution.

## Multimodal Output Generation

1. **REQ-5:** Role-based summarization (e.g., "student" vs. "professor" mode).
2. **REQ-6:** Auto-generate MCQs with distractor options.

# Other Non-functional Requirements

## Performance Requirements

1. Process 100+ documents in parallel.
2. 99.9% uptime for Azure-hosted instances.

## Safety Requirements

1. Plagiarism checks via source embedding traceability.

## Security Requirements

1. AES-256 encryption for data at rest.

## Software Quality Attributes

1. **Scalability**: Modular design for new output types.
2. **Usability:** 90% success rate in user testing.

## Business Rules

1. Only admins can deploy NLP model updates.

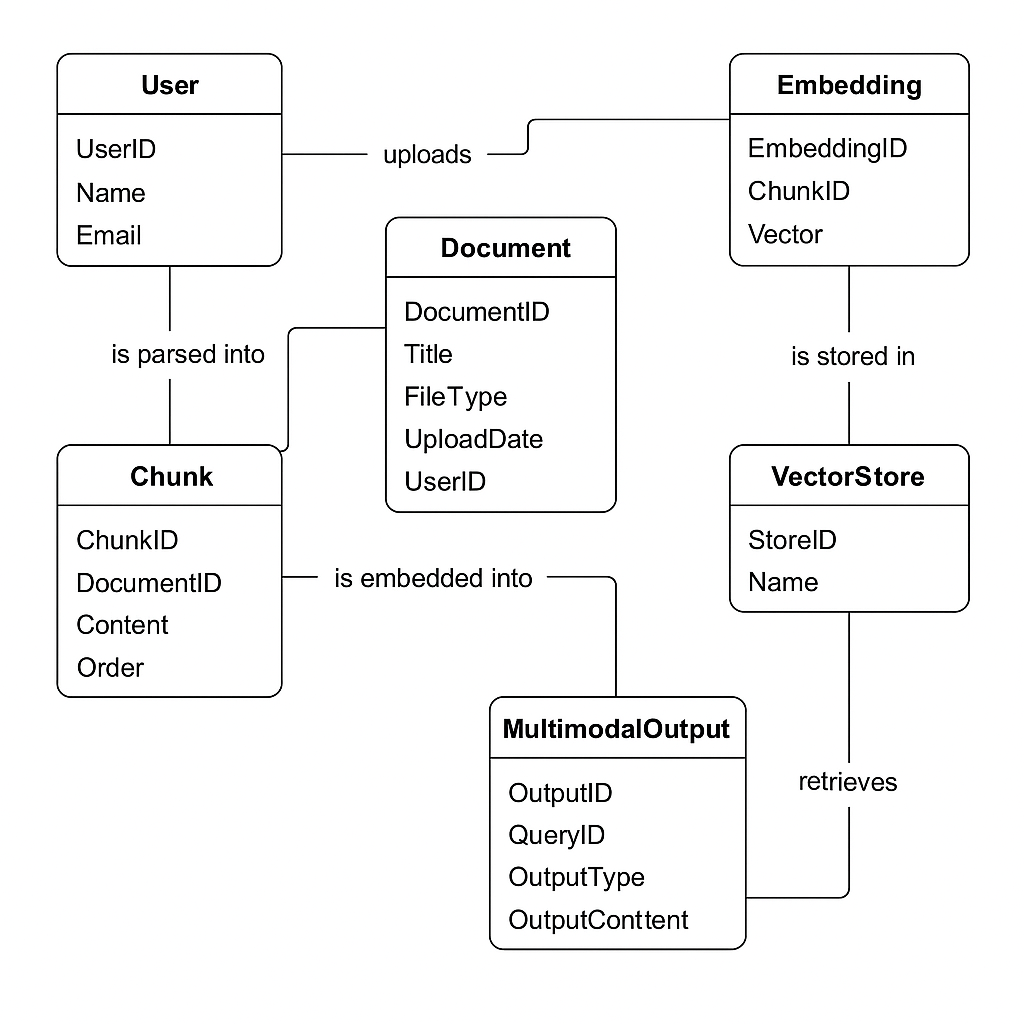
# 

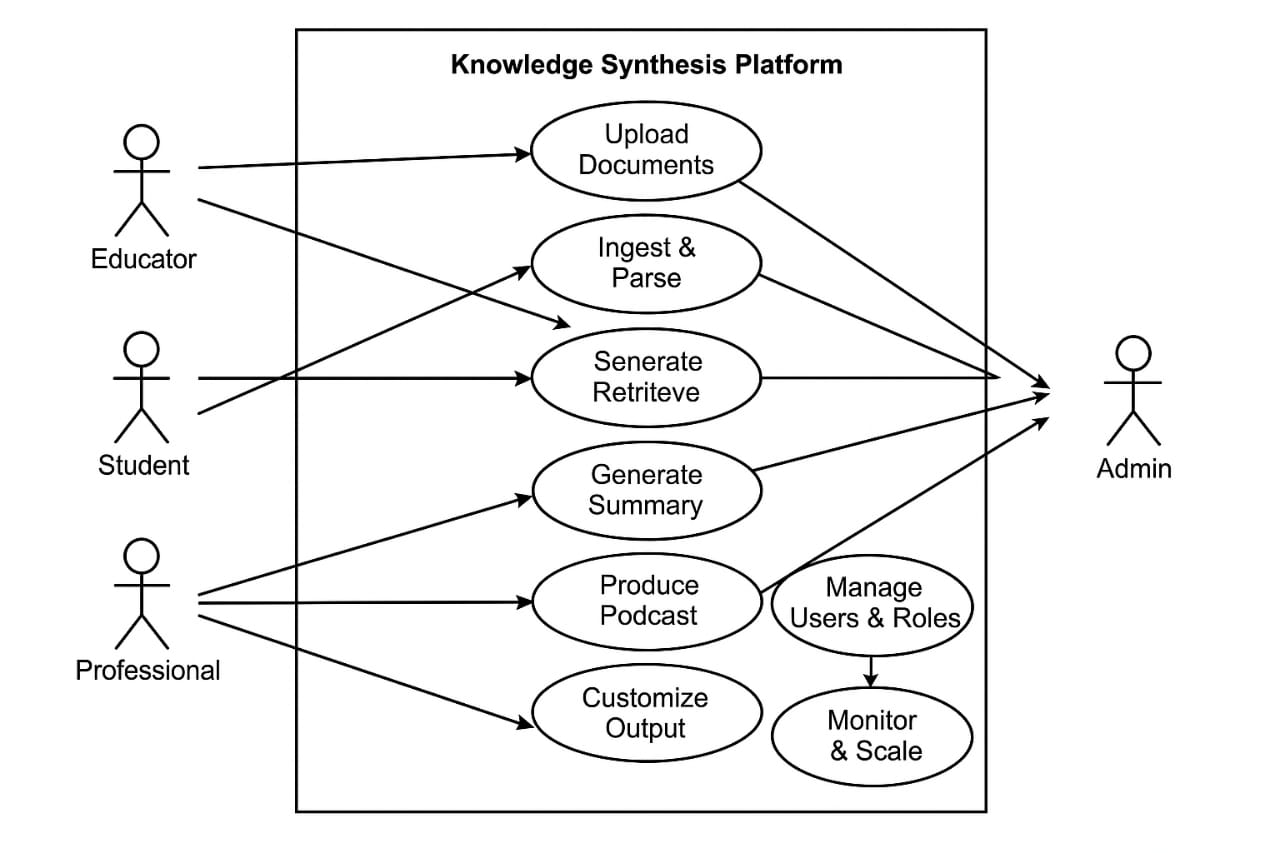
# Appendix A: Glossary

* **RAG Pipeline:** A retrieval-augmented generation system that enhances response relevance.
* **FAISS**: Facebook’s vector similarity search library for efficient retrieval of semantically relevant document chunks during RAG pipeline execution.
* **Embedding:** A numerical representation of text for semantic similarity.
* **Docker:** A platform for containerizing applications

# Appendix B: Analysis Models

**ENTITY-RELATIONSHIP DIAGRAM (ERD)**



**THE USE CASE DIAGRAM**

# Appendix C: To Be Determined List

1. **TBD-1:** Finalize TTS API (Azure vs. Google WaveNet).
2. **TBD-2:** Implement RBAC for enterprise clients.