AI-VOLUTION

2025

TEAM: DATA SQUID

Table of Contents

0 LINKS

GITHUB LINK: https://github.com/tanush-128/ai-volution

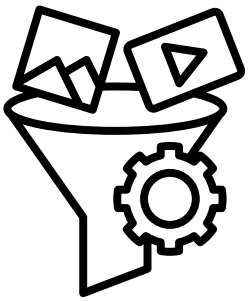
APP LINK: https://drive.google.com/drive/folders/1gSFXVsQVn-nZ65PsbJTbwlc3PGxn2kOK

VIDEO LINK: https://drive.google.com/drive/folders/1gKqtpKMw84Sk7DQH9yzHWOqQQPD4BDpc:

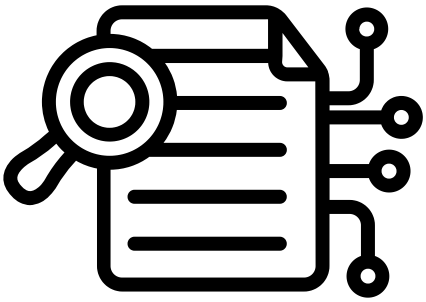
1 PROBLEM STATEMENT

Senior bureaucrats handle vast amounts of information daily, often struggling with document organization, content extraction, and staying updated with relevant news. The overwhelming volume of unstructured data slows down decision-making and increases inefficiencies. To address this, the goal is to develop an AI-powered information management system that enhances file organization, context extraction, and news aggregation. The system should automatically categorize and organize uploaded files (such as PDFs, DOCX, and PPTs) based on their type, content, or relevance— for example, a Budget Report PDF might be classified under Finance, while a Team Meeting PPT is sorted into Presentations. Additionally, it should extract key insights from documents, summarizing critical information for faster processing, and provide personalized AI-driven news recommendations, filtering out irrelevant content while prioritizing governance, policy, and decision-making updates. By leveraging AI, the solution should streamline information handling, reduce cognitive overload, and enable senior bureaucrats to make faster, more informed decisions.

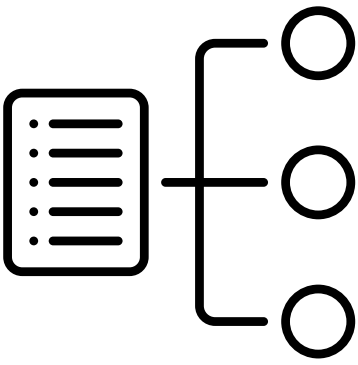
News Recommendation



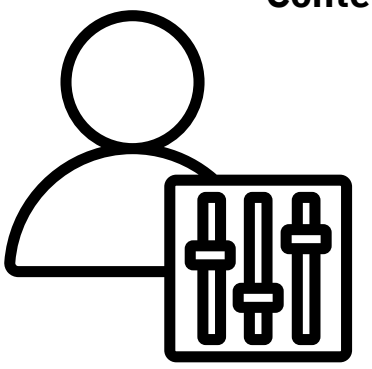
Filtering News



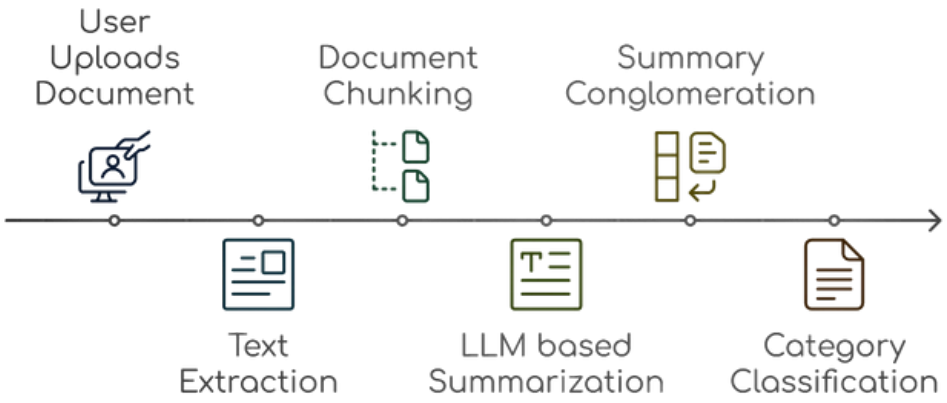
Highlighting Content



Autocategorization Of Files



Personalization



In this project, we present an advanced File Organizer and Context Extraction system that leverages AI-driven content analysis and hierarchical text processing to automate document categorization and summarization.he system is designed to ingest various document formats (PDF, DOCX, PPT), decompose large documents into semantically meaningful segments, generate multi-level structured summaries through a recursive chunk-based summarization pipeline, and classify documents into predefined categories for improved organization and retrieval.y implementing multi-stage text refinement, Chain of Thought (CoT) reasoning, and dynamic document chunking, our system ensures high-fidelity content extraction and context-aware summarization.nitially, we experimented with local packages like PyPDF for parsing; however, we found that the 'unstructured' library is more effective for parsing, particularly in handling tables and images.he 'unstructured' library provides open-source components for ingesting and pre-processing images and text documents, such as PDFs, HTML, and Word docs, making it a versatile tool for our needs.

Technical Implementation

1. Hierarchical Chunk-Based Summarization

The document summarization pipeline follows a multi-stage process, where text is progressively segmented, analyzed, and reassembled to form a structured and coherent summary.

1.1. Multi-Scale Recursive Chunking

- The RecursiveCharacterTextSplitter dynamically partitions large documents into contextually aware chunks with a size of 3000 characters and an overlap of 300 characters.
- Unlike naïve segmentation, our chunking method ensures logical coherence across segments by maintaining an overlap region.
- This hierarchical chunking structure prevents loss of information and enhances summarization fidelity.

1.2. Hierarchical Summarization with CoT Reasoning

Each chunk undergoes multi-step reasoning-based summarization using a stepwise extraction pipeline powered by LLMs:

1. **Topic Segmentation**
   - The model first identifies latent topics within the chunk.
   - Extracts thematic keywords that define the overall context.
2. **Salient Point Extraction**
   - A logical breakdown of the most critical key insights and subtopics is performed.
   - A relevance-based filtering mechanism prioritizes semantically dense statements.
3. **Context-Aware Compression**
   - The extracted content undergoes a compression phase where redundant information is eliminated.
   - A refined hierarchical summary tree is constructed for better readability.
4. **Adaptive Summarization Refinement**
   - To ensure coherence, individual chunk summaries are iteratively merged using an interleaved ranking approach.
   - This phase ensures that cross-references and contextual dependencies across chunks are preserved.

2. Document Classification

Beyond summarization, the system also classifies documents using deep semantic reasoning:
A category-matching model aligns extracted summaries with predefined document types:

- Technical Documentation
- Business Strategy
- Research Paper
- Educational Material
- Project Planning

The classification follows a four-stage reasoning framework:

1. Feature Extraction: Identifies dominant structural and linguistic features.
2. Pattern Recognition: Compares extracted features with category archetypes.
3. Confidence Scoring: Assigns a category based on weighted similarity.
4. Final Category Selection: Outputs the most probable classification with an explanatory rationale.

The core functionality is implemented in Python, leveraging advanced Natural Language Processing (NLP) techniques via LangChain and LLMs. The system is designed to:

- Extract content from uploaded documents.
- Segment and analyze the extracted text.
- Highlight key sections using color-coded categorization.
- Embed highlights back into the original document.

Key Features:

1. Document Parsing & Text Extraction

- Uses an UnstructuredParser to extract text and metadata from documents asynchronously.
- Supports PDF, DOCX, and PPT/PPTX file formats.

2. Chunk-Based Analysis

- Implements RecursiveCharacterTextSplitter to divide large documents into manageable text chunks.
- Each chunk undergoes AI-powered context analysis to determine important segments.

3. AI-Powered Context Extraction

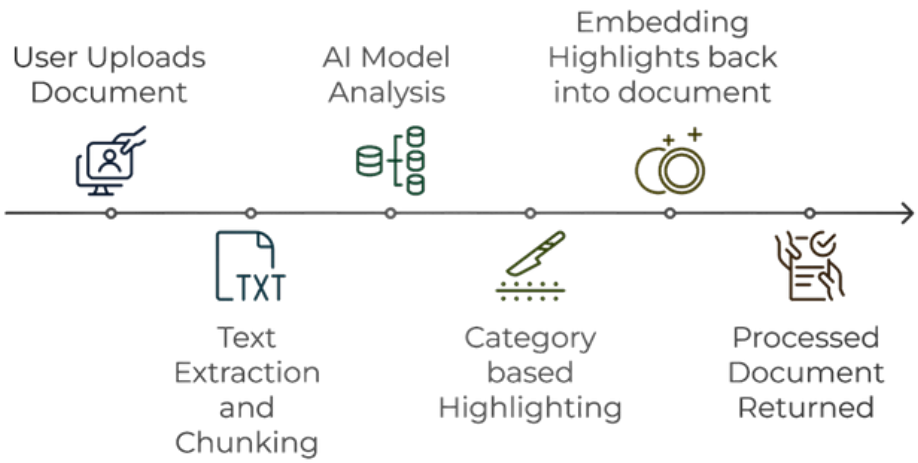
- Used an LLM and implemented Chain of Thought prompting to analyze each chunk and categorize highlights:
  GREEN: Main ideas and big takeaways.
  YELLOW: Important vocabulary and definitions.
  PINK: Questions or areas needing clarification.
  BLUE: Supporting details and sub-ideas.

4. Automated Highlighting & Document Enhancement

- Identified key segments are embedded back into the document:
  PDFs: Highlights are added using add_pdf_highlights.
  DOCX: Uses add_docx_highlights.
  PPT: Modifies slides with add_ppt_highlights.

5. Error Handling & Logging

- Implements structured logging for debugging and error reporting.
- Uses async-await for efficient execution and exception handling.



In the era of information overload, filtering out irrelevant content while ensuring users receive high-priority governance, policy, and decision-making updates is a complex challenge. Our hackathon project presents an AI-driven news recommendation system that personalizes news feeds based on semantic relevance, user preferences, and intelligent content categorization.
By integrating OpenAI's embedding models, cosine similarity-based topic classification, and user behavior modeling, the system delivers precisely curated news articles tailored to individual interests.

Technical Implementation

1. News Data Extraction & Processing

- The system processes a large dataset of news articles stored in JSON format.
- Titles and descriptions are extracted and concatenated into structured representations for downstream processing.
- A predefined taxonomy of 21 topics (e.g., Politics, Economy, Technology, Health) ensures effective categorization.

2. AI-Powered News Classification

The system employs a multi-stage topic classification pipeline:

- Embedding Generation

- Uses OpenAI's text-embedding-3-small model to convert news titles and predefined topic labels into numerical vector representations.
- The model generates high-dimensional embeddings capturing semantic relationships between different articles and topics.

- Cosine Similarity-Based Topic Matching

- Each article embedding is compared against predefined topic embeddings using cosine similarity.
- The top-N most relevant topics are assigned to each article based on similarity scores.

This process ensures precise categorization, preventing misclassification of news articles.

4. Implementation Flow

- News data is loaded from a structured JSON file.
- Title and description embeddings are generated using OpenAI models.
- Each article is classified into relevant topics using cosine similarity.
- User engagement data is analyzed to identify preferences.

A personalized news feed is generated, prioritizing governance, policy, and decision-making updates.

3 SOLUTION OVERVIEW TASK 1

The AI-Driven Personalized News Recommendation System is a sophisticated solution designed to provide real-time, high-relevance news recommendations while filtering out irrelevant content. The system ensures that users receive curated, governance, policy, and decision-making updates by leveraging AI-powered search, intelligent topic modeling, and real-time news aggregation.
Core Functionalities

1. **User-Driven Topic Selection**
   - Users define areas of interest (Politics, Economy, Science, Technology, etc.).
   - The system dynamically adjusts search queries to match user preferences.
2. **AI-Optimized Query Expansion & Search**
   - Uses Tavily Search API to analyze trending news developments.
   - GPT-4o-mini generates high-relevance search queries using keyword optimization techniques (+, "quotes", OR).
   - This ensures broad yet relevant coverage of current news topics.
3. **Asynchronous Multi-Source News Retrieval**
   - The system executes multiple news queries in parallel using AsyncIO.
   - Fetches real-time news from multiple sources via a custom news API integration.
4. **Intelligent News Categorization & Filtering**
   - Uses semantic similarity scoring to categorize and rank articles based on topic relevance.
   - Filters out redundant or low-quality articles to ensure precision.
5. **Structured Response & Personalization**
   - Outputs structured JSON data for integration into news dashboards or APIs.
   - Ensures that users receive personalized, high-priority news updates.

Key Benefits

- Personalized News Recommendations based on user-defined topics.
- AI-Powered Search Optimization for better query precision.
- Parallel Asynchronous Processing for faster real-time news retrieval.
- Intelligent Categorization & Relevance Filtering to remove noise.
- Seamless Integration with existing systems via structured JSON output.

This scalable, AI-enhanced approach ensures that users receive only the most relevant, policy-focused news, making it an essential tool for decision-makers and professionals.

4 SOLUTION OVERVIEW TASK 2

Our AI-Driven Personalized News Recommendation System is designed to filter out irrelevant content while prioritizing governance, policy, and decision-making updates. The solution integrates AI-powered search, intelligent topic modeling, and real-time news aggregation to deliver highly personalized and relevant news recommendations.

Key Functionalities:

- User-Centric Topic Selection
- Users specify topics of interest (Politics, Economy, Science, Technology, etc.).
- The system dynamically tailors news searches to match user preferences.

AI-Powered Search & Query Optimization

- Uses Tavily Search API to retrieve trending news insights.
- LLM generates optimized keyword-based search queries to enhance relevance.
- AI-driven search techniques ensure precision and diversity in retrieved articles.

Asynchronous Multi-Source News Retrieval

- Executes multiple real-time search queries across news sources.
- Parallel processing (AsyncIO) ensures fast and efficient aggregation of diverse news articles.

Automated News Categorization & Personalization

- Uses semantic similarity and relevance scoring to classify articles.
- Ensures topic coherence while avoiding redundancy.
- Formats results into a structured JSON output, making it easy to integrate into user dashboards.

Key Advantages:

- Highly Personalized News Recommendations
- Real-Time, AI-Enhanced Search Queries
- Efficient Asynchronous Processing for Fast Results
- Multi-Source Aggregation for Comprehensive Coverage

This scalable, AI-powered solution ensures that users receive only the most relevant news, tailored to their interests and decision-making needs.

5 TECHNICAL ARCHITECTURE TASK 1

The AI-Driven Personalized News Recommendation System is a modular, scalable architecture designed for real-time, high-relevance news retrieval and categorization. It integrates LLMs, search APIs, asynchronous processing, and intelligent filtering to ensure precision-driven recommendations.

System Components & Workflow

1. **User Input & Topic Selection**
- Users define topics (Politics, Economy, Technology, etc.).
- The system dynamically adjusts search queries.

2. AI-Powered Query Expansion
- Tavily Search API gathers trending insights.
- GPT-4o-mini generates optimized search queries using:
  - Must-have terms (+)
  - Exact phrase matching ("")
  - Alternative keywords (OR)

3. Asynchronous News Retrieval
- Parallel API calls (AsyncIO) reduce latency.
- Custom News API fetches real-time news articles.

4. Categorization & Filtering
- Semantic similarity scoring classifies articles.
- Duplicate filtering ensures relevance.

5. Structured Response & Delivery
- Outputs JSON-formatted results for seamless integration into dashboards/APIs.

Key Benefits

- Real-Time, AI-Powered Search Optimization
- Scalable Asynchronous Processing for multiple users
- Intelligent News Categorization & Filtering
- Seamless API Integration with structured JSON output

This AI-driven system ensures highly personalized, real-time news recommendations, making it an essential tool for decision-makers and policy analysts.

6 TECHNICAL ARCHITECTURE TASK 2

System Components

1. User Input Layer

- Users specify topics of interest (e.g., Politics, Economy, Technology, etc.).
- Input is structured using the UserTopics model.

2. AI-Powered Search & Query Generation

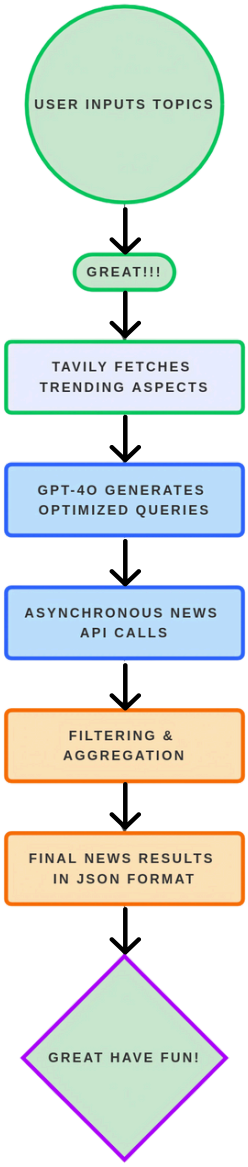
- Uses Tavily Search API to find trending aspects of each topic.
- GPT-4o-mini generates optimized search queries based on Tavily results.
- Queries are structured using:
  - "+" for must-include terms
  - Quotes for exact phrases
  - "OR" between alternatives

3. Asynchronous News Retrieval

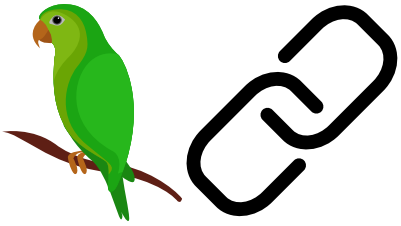
- AsyncIO is used for concurrent processing of multiple topics.
- Calls a custom get_news API to retrieve real-time news.
- Fetches up to 5 articles per query to ensure diverse coverage.

4. AI-Based Article Aggregation & Filtering

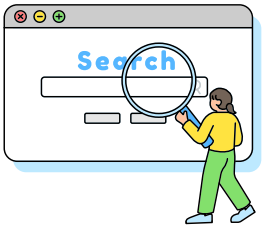
- Combines results from multiple searches.
- Filters irrelevant or duplicate articles.
- Outputs structured JSON data with the most relevant articles.



7 Technical Stack



LangChain (ChatOpenAI, AgentExecutors)
for LLM processing



Travily Search Engine

Pydantic adds
Type safety

FastAPI

FastAPI for creating APIs

PostgreSQL

PostgresSQL for storing the file uploaded and updating the data

LLM of OoenAI used

Pathlib for File Handling

Asyncio for doing concurrent tasks

{.json}

For Storing structured data