Artificial Intelligence Department
Faculty of Computers and Artificial Intelligence
Cairo University

Graduation Project

SumGenQ

Name	ID
Abdelkhalek Mohamed Ali	20210221
Mohamed Ahmed Mohamed	20210325
Mohamed Mahmoud Hussein	20210355
Ahmed Osama Fawzy	20210008
Noor Alain Kahled Almatari	20210808

Supervised by:

Dr. Asmaa Ahmed E. Osman

July 2025

Table of Contents

List of Figures

List of Tables

List of Abbreviations

- **AI**: Artificial Intelligence
- **NLP**: Natural Language Processing
- **TTS**: Text-to-Speech
- **QA**: Question Answering
- **RAG**: Retrieval-Augmented Generation
- **BART**: Bidirectional and Auto-Regressive Transformer
- **T5**: Text-to-Text Transfer Transformer
- **mT5**: Multilingual Text-to-Text Transfer Transformer
- **BERT**: Bidirectional Encoder Representations from Transformers
- **LLM**: Large Language Model
- **PEFT**: Parameter-Efficient Fine-Tuning
- **LoRA**: Low-Rank Adaptation
- **ROUGE**: Recall-Oriented Understudy for Gisting Evaluation
- **BLEU**: Bilingual Evaluation Understudy
- **CNN/DM**: CNN/DailyMail Dataset
- **ACLSUM**: Aspect-based dataset for scientific summarization
- **QReCC**: Question Reformulation for Conversational QA
- **QG-NET**: Neural Network for Automatic Quiz Generation
- **TF-IDF**: Term Frequency-Inverse Document Frequency
- **LSTM**: Long Short-Term Memory
- **RNN**: Recurrent Neural Network
- **JWT**: JSON Web Token
- **OTP**: One-Time Password
- **API**: Application Programming Interface
- **EDR**: Entity Data Relationship
- **FP16**: 16-bit Floating Point (Mixed Precision Training)
- **DPR**: Dense Passage Retriever
- **BM25**: Best Matching 25 (keyword-based retriever)
- **FiD**: Fusion-in-Decoder
- **SpaCy**: Python library for Natural Language Processing
- **CUDA**: Compute Unified Device Architecture (for GPU acceleration)

Chapter 1. Introduction

1.1 Problem Definition

In the modern digital age, users are frequently overwhelmed by the sheer volume of textual information available online and in academic or professional settings. Manually reading, analyzing, and extracting relevant content from long texts can be time-consuming and mentally demanding. This project presents an intelligent solution to this problem by combining **automatic text, documents and URLs summarization**, interactive **chatbot question answering assistance**, **quiz generation**, and **text-to-speech (TTS)** technology into a unified platform.

The proposed system aims to enhance the way users interact with textual content by allowing them to quickly generate meaningful summaries, ask questions through a chatbot, assess their understanding via automatically generated quizzes, and listen to summaries in natural-sounding speech. This not only aids in better comprehension and retention of information but also makes the system highly accessible, particularly for users with visual impairments or reading challenges.

By integrating natural language processing (NLP) techniques and machine learning tools, the system provides a multi-functional educational and assistive solution. The summarization module condenses lengthy documents into concise formats, the chatbot enables natural question- answer interaction, the quiz module offers a self-evaluation mechanism, and the TTS feature makes the platform inclusive and accessible.

This project contributes to the fields of **educational technology**, **assistive tools**, and **artificial intelligence**, and demonstrates how multiple NLP capabilities can be integrated to serve a wide range of users with varying needs and preferences.

1.2 Objective

The primary objective of this project is to design and develop an intelligent system that automates the process of understanding and interacting with textual content.

The system aims to:

- **Automatically summarize text , documents and URLs** to provide users with concise and meaningful information.
- **Enable interactive question-and-answer communication** through a chatbot that responds to user queries about the summarized content.

- **Generate multiple-choice, T&F and essay quiz questions** to help users evaluate their comprehension of the material.
- **Convert the summary into natural-sounding speech** using Text-to-Speech (TTS) technology to improve accessibility and ease of use.

This project focuses on combining natural language processing (NLP), machine learning, and speech synthesis to create a unified platform that is educational, accessible, and user-friendly

1.3 Motivation

In today's information-driven world, individuals are constantly exposed to large volumes of text in academic, professional, and personal contexts. However, not everyone has the time, focus, or ability to read and comprehend long documents. This challenge is even more significant for people with visual impairments, learning difficulties, or limited literacy skills.

Furthermore, students and self-learners often struggle with retaining knowledge after reading. They require tools that not only summarize content but also engage them interactively and allow them to test their understanding.

These needs inspired the development of a system that goes beyond basic summarization by integrating a chatbot for interactive learning, a quiz generator for self-assessment, and a text-to-speech module to ensure accessibility and inclusiveness.

By combining multiple technologies into a single platform, this project aims to provide a holistic and intelligent learning experience that can benefit a wide range of users.

The motivation behind this project is to build a smart, educational tool that:

- Saves users time by condensing long texts into digestible summaries.
- Enhances comprehension through natural interaction.
- Encourages active recall through quizzes.
- Makes content accessible to all users, regardless of reading ability.

By combining multiple technologies into a single platform, this project aims to provide

a holistic and intelligent learning experience that can benefit a wide range of users.

During the development of this project, several challenges and limitations were encountered, both technical and conceptual. Understanding these constraints is essential to evaluating the current system and identifying opportunities for future improvement.

1.4 Challenges & Limitation

1.4.1 challenges

- Summarization Accuracy:

  Achieving high-quality, meaningful summaries was challenging, especially when dealing with unstructured or poorly formatted input text. Selecting the right model and fine-tuning it for various content types required extensive testing and adjustments.

- Chatbot Question Answering:

  Ensuring that the chatbot provided accurate and contextually appropriate answers from the summary posed difficulties, especially with ambiguous or complex questions.

- Quiz Generation Logic:

  Automatically generating meaningful and grammatically correct multiple-choice questions from summarized content required sophisticated NLP techniques and often led to issues with distractor choices and clarity.

- Text-to-Speech (TTS) Integration:

  Generating natural and clear audio from text was another challenge, particularly when using open-source models that required fine-tuning for pronunciation, speed, and tone.

- System Integration and Flow:

  Combining all modules—summamodules andatbot, quiz, and TTS—into a smooth and consistent user experience involved handling dependencies,

managing data flow between modules and maintaining performance.

1.4.2 Limitations

- Language Support:

  The system currently supports only English. Multilingual support is limited and would require additional model training and adaptation.

- Dependence on Pre-trained Models:

  The system heavily relies on pre-trained NLP models, which may not always produce ideal results for highly domain-specific or informal texts.

Chapter 2. Related Works

2.1 Related Summarization Papers

2.1.1 Long Text & Multiple Table Summarization

Text summarization is the process of automatically generating a concise and meaningful summary from a large body of text while preserving key information. It plays a crucial role in handling large volumes of information by enabling quick comprehension, improving search efficiency, and reducing the time required to process extensive documents. Summarization techniques can be categorized into traditional methods (TF-IDF, Latent Semantic Analysis) and deep learning-based techniques (Transformers like BERT, T5, BART, as well as RNNs and LSTMs). While deep learning models achieve state-of-the-art performance, they require large datasets and high computational resources.
To address challenges in long- text and multi-table summarization, the FINDSum dataset has been introduced as a large-scale dataset designed for summarizing complex financial documents. Built on 21,125 annual reports from 3,794 companies, it includes subsets for summarizing each company's results of operations and liquidity. This dataset supports both textual and non-textual content, making it highly suitable for real-world summarization tasks. By leveraging advanced summarization models and comprehensive datasets like FINDSum, this research aims to enhance the efficiency and accuracy of summarizing long and complex financial documents**.**

- **Used Models:**
  - **BART** (Bidirectional-and-Auto-Regressive Transformers)
  - **T5** (Text-to-Text-Transfer Transformer)
  - **MT5** (Multilingual-Text-to-Text Transfer Transformer
  - **Longformer**
  - **BigBird**
- **Used Datasets**:
  - **Source**: The dataset is derived from annual reports of companies, which include both textual and tabular data.
  - **Size**: It contains 21,125 annual reports from 3,794 companies

- **FINDSum-ROO:** Focuses on summarizing each company's results of operations
- **Content Each example in the dataset includes:** Tens of thousands of words from the textual content of the reports, Dozens of tables containing numerical data.

2.1.2 Aclsum

ACLSUM represents a significant advancement in scientific summarization by providing a high- quality, aspect-based dataset. The paper demonstrates the effectiveness of structured summarization, highlighting challenges in capturing abstract research problems. While ACLSUM is a step forward, future research should focus on scalability and broader domain coverage. The dataset's structured approach paves the way for improved scientific text summarization models, benefiting researchers and automated systems alike.

- **Used Models:**
  - **Extractive Models**
    - **Sentence-T5 (BASE, LARGE, XL):** A strong text encoder used for selecting relevant sentences.
    - **TextRank:** A graph-based ranking model for sentence extraction.
  - **Abstractive Models**
    - **BART (BASE, LARGE)**: A transformer-based seq2seq model designed for text generation tasks.
    - **T5 (BASE, LARGE)**: A text-to-text transfer transformer model that generates summaries from input text.
    - **Llama 2 (7B)**: A large language model fine-tuned for summarization using two strategies: End-to-End (E2E) and Extract-then-Abstract Chain-of-Thought (EtA-CoT).

2.1.3 Long Documents Summarization Using Top-Down & Bottom-Up Interfaces

Long document summarization is a critical challenge in natural language processing, requiring efficient techniques to extract key information from extensive texts. This paper explores a novel approach to summarizing long documents using top-down and bottom-

up interfaces, enabling more structured and adaptable summarization. The study leverages multiple large-scale datasets, including **PubMed, arXiv, TVMegaSite, ForeverDreaming, BookSum-Chapter-Level, and BookSum-Book-Level,** covering a diverse range of domains such as scientific articles, TV show transcripts, and book summaries. Traditional summarization methods, such as TF-IDF and Latent Semantic Analysis, struggle with lengthy documents due to limited contextual understanding, while machine learning and deep learning-based approaches, including Transformer models like BERT, T5, and BART, provide more effective solutions. However, deep learning models often require large-scale datasets and substantial computational resources. By integrating top-down and bottom-up interfaces, this research aims to enhance summarization efficiency by structuring long- form content in a hierarchical and context-aware manner. The proposed approach is evaluated using benchmark datasets, demonstrating improved coherence, relevance, and readability of generated summaries compared to existing techniques.

- **Used Models:**
  - **TDT (OracleAdaPool)** Getting best Accuracy with PubMed, TVMegaSite, CNN Daily Mails & arXiv.
  - **Extractive Oracle** Getting best Accuracy with BookSum-Book-Level & BookSum- Chapter-Level.
  - **Hybrid (BART + Orcale Content Selection)** Getting best Accuracy with Forever Dreaming.
- **Used Datasets:**

Dataset Name	Type	Number of Documents	Input Size	Output Size
PubMed	Scientific	133K	3.224	214
arXiv	Scientific	215K	6.913	292
TVMegaSite	Conversational	22.5K	6.420	380
ForeverDreaming	Conversational	4.3K	7.605	113
CNN-DM	Conversational	311K	906	63

BookSum-Chapter-Level	Conversational	12K	5.102	505
BookSum-Book-Level	Conversational	436K	112.885	1167

Table: 1

2.2 Related Chatbot Question Answering Papers

2.2.1 QReCC

QReCC (Question Reformulation for Conversational QA) is a benchmark dataset tailored for training and evaluating open-domain chatbot QA systems. It emphasizes the challenge of answering follow-up questions in multi-turn conversations, where context understanding and question rewriting are critical. The dataset contains over 80k question-answer pairs aligned with real user queries from search logs. QReCC pushes the development of conversational agents capable of handling contextual ambiguity and dialogue continuity, making it a valuable resource for both academic and practical chatbot systems.

The paper underlines the importance of question reformulation and retrieval-based QA models in improving chatbot interactions, especially in real-world information-seeking scenarios

- **Used Models:**
    - **Retrieval-Augmented QA Models**
        - **FiD (Fusion-in-Decoder):** Uses a generative decoder over multiple retrieved passages to answer questions with context.
        - **RAG (Retrieval-Augmented Generation):** Combines dense retrievers and seq2seq models to retrieve and generate answers.
    - **Context-Aware Reformulation Models**
        - **T5 (BASE, LARGE):** Fine-tuned for question rewriting and answer generation.
        - **BART (BASE, LARGE):** Applied for both rewriting and abstractive answering.
        - **GPT-3:** Used in zero-shot and few-shot setups for conversational QA tasks.
    - **Retrievers**
        - **BM25:** Classic keyword-based retriever used as a baseline.
        - **DPR (Dense Passage Retriever):** Neural retriever trained with contrastive learning for better semantic matching.

2.3 Related Quiz Generator Papers

2.3.1 G-NET: A Neural Network for Automatic Quiz Generation

QG-NET presents a neural architecture for automatic quiz generation from text, focusing on educational use cases. The model integrates contextual understanding and answer prediction

to generate relevant, diverse, and grammatically correct questions. QG-NET is trained on QA pairs extracted from academic and Wikipedia datasets, making it suitable for a wide range of learning domains. The paper highlights the importance of question quality, coverage of key concepts, and answer relevance.

This work supports the development of intelligent tutoring systems and self-assessment tools by automating quiz creation, especially valuable in low-resource educational settings.

- **Used Models:**
    - **Encoder-Decoder QA Models**
        - **T5 (BASE, LARGE):** Trained to generate questions conditioned on answers and context.
        - **BART:** Used for generating natural-sounding questions with contextual fluency.
        - **GPT-2/3:** Applied in prompt-based question generation, leveraging pre-trained knowledge.
    - **Answer Extraction and Ranking**
        - **BERT:** Used to extract and rank answer spans from source text.
        - **BiDAF:** An attention-based model used to identify precise answer locations.
    - **Evaluation Techniques**
        - **BLEU/ROUGE:** Used to evaluate linguistic quality and semantic overlap with reference questions.
        - **Human Evaluation:** Performed to assess answer relevance, grammar, and usefulness

2.4 Competitor Websites

2.4.1 QuillBot

Figure: 1

- **About QuillBot website:**

QuillBot is an online platform focused on writing and language enhancement. It provides AI-powered tools such as paraphrasing, grammar checking, and summarization to help users improve their writing. The platform is designed to assist students, professionals, and writers by making content creation more efficient, accurate and polished. QuillBot's summarizer helps users condense large

texts into concise summaries, supporting quick understanding and improved productivity

- **Learner User flow (Flowchart):**

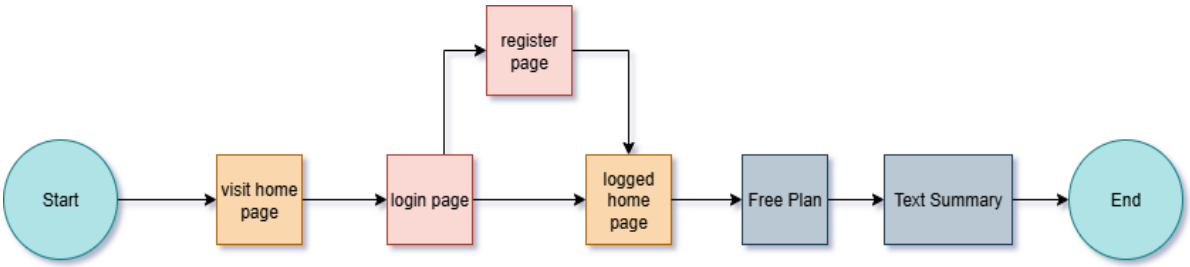


Figure: 2

- **Strength Points:**

  QuillBot offers two paths of learning

  - **QuillBot Summarizer:**

    The summarizer condenses long passages of text into concise summaries, making it easier for users to extract key points and understand complex materials quickly. It's useful for students, researchers, and professionals alike.

  - **QuillBot Paraphraser:**

    The paraphrasing tool helps users reword sentences while maintaining meaning, enhancing writing clarity and originality. It supports multiple modes (Standard, Fluency, Formal, etc.) for different writing needs**.**

- **weakness Points:**
  - Does not allow saving or organizing summaries in history
  - Cannot generate quizzes or interact with uploaded documents
  - Does not support voice output for summaries
  - Text input limit in the free plan (up to 1,200 words)
  - Lacks interactive or AI-driven feedback
  - File/book summarization not supported

2.4.2 ChatPDF

Figure: 3

- **About ChatPDF website:**

ChatPDF is an online tool that enables users to upload PDF documents and interact with them through a conversational interface. It allows users to ask questions about the content, receive instant answers, and explore summaries of books or academic texts. The platform enhances document comprehension by turning static files into interactive learning experiences through chat-based engagement.

- **Learner User flow (Flowchart):**

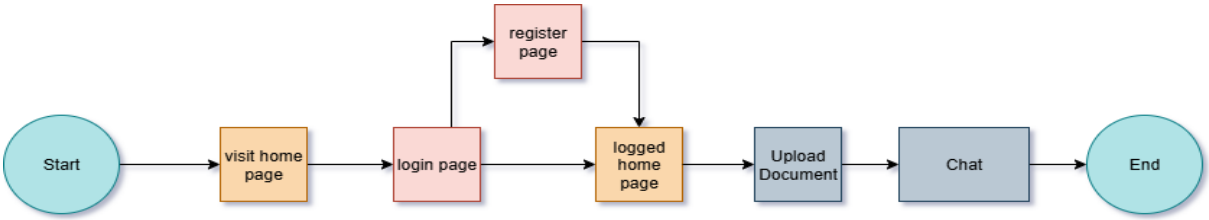


Figure: 4

- **Strength Points:**
  - Allows users to upload documents and chat with the content directly
  - Provides accurate answers based on the content of the uploaded PDF
  - Useful for summarizing and understanding books, research papers, and academic documents
  - Enables reopening previous chats and exporting conversations
- **weakness Points**
  - Cannot summarize plain text directly
  - Limited free plan: only 2 chats and 20 messages for free
  - No support for converting summaries to voice
  - No AI-generated quizzes or interactive feedback
  - Limited after deleting summaries or chats

2.4.3 TLDR this

Figure: 5

- **About TLDR this website:**

TLDR This is an online tool designed to automatically summarize long-form content such as articles, web pages, documents, and text. It extracts key points and presents concise summaries to help users quickly understand complex material. With support for both text input and URL-based summarization, TLDR This streamlines content consumption and is useful for students, professionals, and readers looking to save time.

- **Learner User flow (Flowchart):**

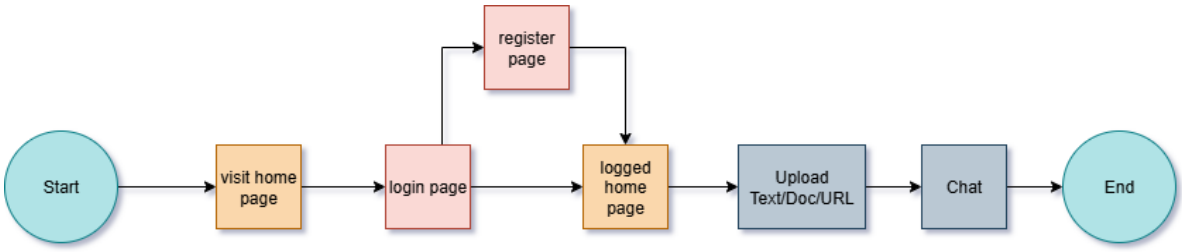


Figure: 6

- **Strength Points:**
  - Supports summarization of **text, documents, and URLs**
  - Provides concise summaries ideal for quick understanding
  - Can handle academic articles, blog posts, and web content
  - Easy-to-use interface suitable for fast summarization needs
- **weakness Points:**
  - Limited interactive features (no chat-based interface)
  - Does not store summary history or allow revisiting previous sessions
  - No quiz generation or deeper educational engagement
  - Lacks voice output for summaries
  - Free plan limitations on number of uses per day

2.4.4 SumGenQ

Figure: 7

- **About SunGenQ this website:**

Summrizier is an educational application designed for children aged 6 to 12 years old. The platform helps young learners explore and understand topics like Egyptian civilization through engaging features such as AI-generated summaries, interactive quizzes, and storytelling. It combines education with technology to make learning more accessible, interactive, and fun.

- **Learner User flow (Flowchart):**

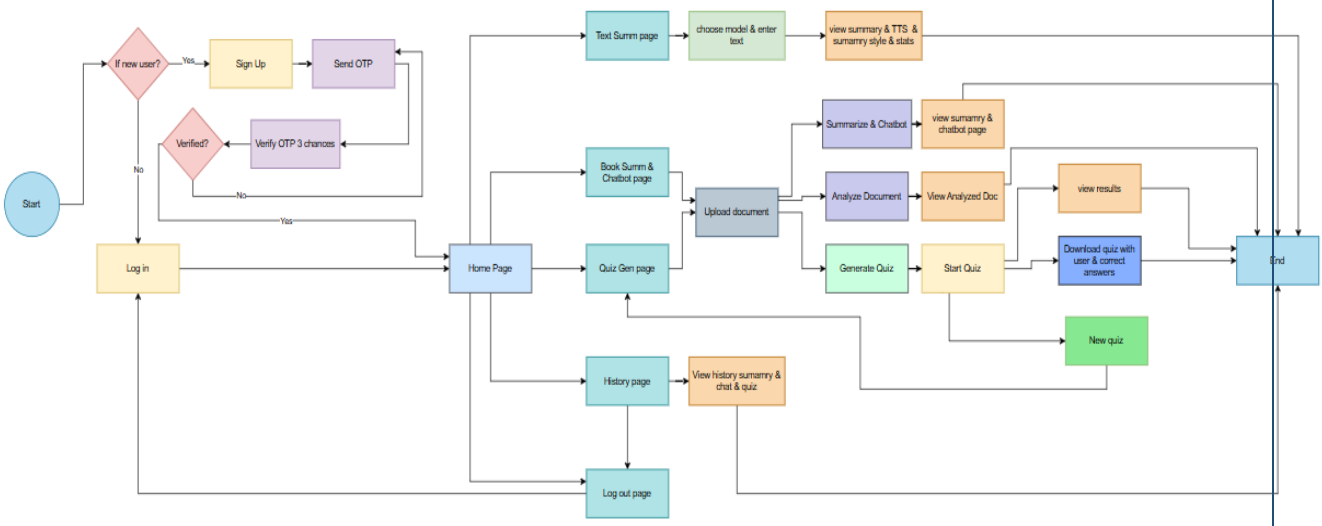


Figure: 8

- **Strength Points:**
  - 🧠 AI-powered summarization for both text and documents.
  - 🌐 support URL-based content summarization and interactive chatbot.
  - 🔊 Voice output that converts summaries into audio for accessibility.
  - 💬 Interactive chatbot to ask questions about any documents.
  - 🧪 Quiz generation based on uploaded documents.
  - 📥 Downloadable summaries as word files.

- 📚 history tracking for text, documents, chatbot QA and quiz entries.
- 🔓 Completely free with generous access.
- 🎯 Designed for education, productivity, accessibility and helpful in general.

- **weakness Points :**
  - 🚧 Currently does not support uploading images or scanned handwritten notes.

- **Our Competitive Advantage**
  - Combines **text + document + URL summarization + chatbot question answering on the uploaded documents + quizzes + TTS** in one platform.
  - Offers a **full education-focused workflow**, not just summarization
  - Allows users to **chat with documents**, **test their understanding**, and **learn interactively**
  - **All features are free**, making it accessible to students, researchers, and lifelong learners
  - Tailored for **real use cases**: study help, research support, reading comprehension, and more

	SumGenQ	ChatPDF	TLDR this	Quill bot	Note GPT	BOIZ
Website	✓	✓	✓	✓	✓	✓
Mobile app	✗	✓	✗	✓	✗	✗
Text Summ.	✓	✓	✗	✓	✗	✗
URL/File Summ.	✓	✗	✗	✗	✓	✓
Chatbot QA	✓	✗	✓	✗	✗	✗
Quiz Gen.	✓	✗	✗	✗	✗	✗
TTS	✓	✗	✗	✗	✗	✗
History	✓	✗	✗	✗	✗	✗

Table: 2

Chapter 3. Proposed Model "System Specification"

The proposed system is an intelligent assistant designed to simplify and enhance the process of understanding lengthy textual content. It integrates four core features into a single platform: **automatic text summarization**, an **interactive chatbot**, **quiz generation**, and **text-to-speech (TTS)** for audio output of summaries. This system is aimed at students, educators, and users with visual or reading difficulties who require quick access to the essence of documents in both written and spoken form.

3.1 Dataset

3.1.1 CNN/DailyMail Dataset

A widely used dataset for summarization tasks, containing news articles paired with summaries. summaries are abstractive and concise, The CNN / DailyMail Dataset is an English-language dataset containing just over 300k unique news articles as written by journalists at CNN and the Daily Mail. The current version supports both extractive and abstractive summarization, though the original version was created for machine reading and comprehension and abstractive question answering

- **Dataset Description:**
  - **Source:** https://huggingface.co/datasets/abisee/cnn_dailymail
  - **Size:** 935,913 article-summary pairs
  - **Use Case:** While not book-specific, Suitable for training models on short to medium-length text summarization

3.1.2 BookSum Dataset

For this project, we use the BookSum Complete Cleaned dataset, hosted on Hugging Face. This dataset is specifically designed for text summarization tasks and contains a collection of books and chapters paired with summaries. It is well-suited for our project as it provides a diverse range of literary works and high-quality summaries.

- **Dataset Description:**
  - **Source:** https://huggingface.co/datasets/ubaada/booksum-complete-cleaned
  - **Purpose:** Some mismatched summaries have been corrected. Unaccessary columns have been discarded. Contains minimal text-to-summary rows. As there are multiple summaries for a given text, each row contains an array of summaries
- **Dataset Structure:**
  - **Chapters Part**

Id	Content
0	(book id)
1	book_title
2	chapter_id
3	text (raw chapter text)
4	summary (list of summaries from different sources) {source, text (summary), analysis}
5	is_aggregate (bool) (if true, then the text contains more than one chapter)

Table: 2

- **Chapters Part**

Id	Content
0	bid (book id)
1	title
2	text (raw text)
3	summary (list of summaries from different sources) {source, text (summary), analysis}

Table: 3

- **Dataset Distribution:**
  - **Chapters Part**

Split	Total	Missing Sum	Successfully Processed	Chapters
Train	9712	178	9534(98.17%)	5653

Test	1432	0	1432(100%)	950
Validate	1485	0	1485(100%)	854

Table: 4

- **Books Part**

Split	Total	Missing Sum	Successfully Processed	Books
Train	314	0	314(100%)	151
Test	46	0	46(100%)	17
Validate	45	0	45(100%)	19

Table: 5

3.1.3 WikiSum Dataset

The WikiSum Dataset is a large-scale dataset designed for summarization tasks, leveraging Wikipedia articles as source documents paired with their corresponding summaries. This dataset is particularly useful for training and evaluating models on long-form summarization tasks, as it contains diverse topics and high-quality summaries. It is well-suited for our project as it provides a rich source of structured text and summaries.

- **Dataset Description:**
  - **Source**: https://huggingface.co/datasets/d0rj/wikisum
  - **Purpose**: The dataset is designed for training and evaluating summarization models, particularly for long-form content like Wikipedia articles.
  - **Size**: The dataset contains 39,775 article-summary pairs
  - **Dataset Structure**
  -

ID	Title
0	url
1	Title
2	Summary
3	article

4	step_headers

Table: 6

3.1.4 XSum Dataset

Focuses on generating very short, one-sentence summaries, Complements CNN/DailyMail by offering an option for users who want quick, concise summaries.

- **Dataset Description:**
  - **Source:** https://huggingface.co/datasets/EdinburghNLP/xsum
  - **Use Case:** Add a feature to website for users who want extreme summarization

3.1.5 Project Gutenberg

A collection of over 75,000 free eBooks.

- **Dataset Description:**
  - **Source**: Project Gutenberg (https://www.gutenberg.org/)
  - **Features:**
    - Large and diverse collection of books.
    - No summaries, but we can generate summaries by using external tools.
    - Consider combining with other datasets (e.g., BookSum) to enhance the diversity.

3.1.6 Used Dataset

We used CNN/DailyMail Dataset

- **Initial Data Analysis:**
  - Calculate and visualize statistics (mean, std, min, max) for article and summary lengths.
  - Plot distributions of article and summary lengths for each split.
- **Text Cleaning and Preprocessing Steps:**
  - **Remove HTML tags** using BeautifulSoup.
  - **Normalize text**: lowercase, remove extra whitespace, normalize unicode.
  - **Expand contractions** (e.g., "don't" → "do not").
  - **Remove special characters** (keep alphanumeric and basic punctuation).
  - **Remove URLs and emails.**
  - **Remove numbers** (optionally preserve named entities).
  - **Remove punctuation** (optionally keep basic punctuation).
  - **Remove boilerplate/news template text** (e.g., "By staff", "Read more:").

- **Extract named entities** using SpaCy.
- **Lemmatize** words using WordNetLemmatizer.
- **Remove stopwords** (with a custom list to keep some negations).
- **Language detection**: keep only English texts.

- **Content Quality Filtering:**
  - Filter out samples that:
  - Are too short or too long (article: 100–2000 words, summary: 15–156 words).
  - Have an unreasonable article-to-summary length ratio.
  - Have too few contents words & Check if are not in English.

3.2 System Design

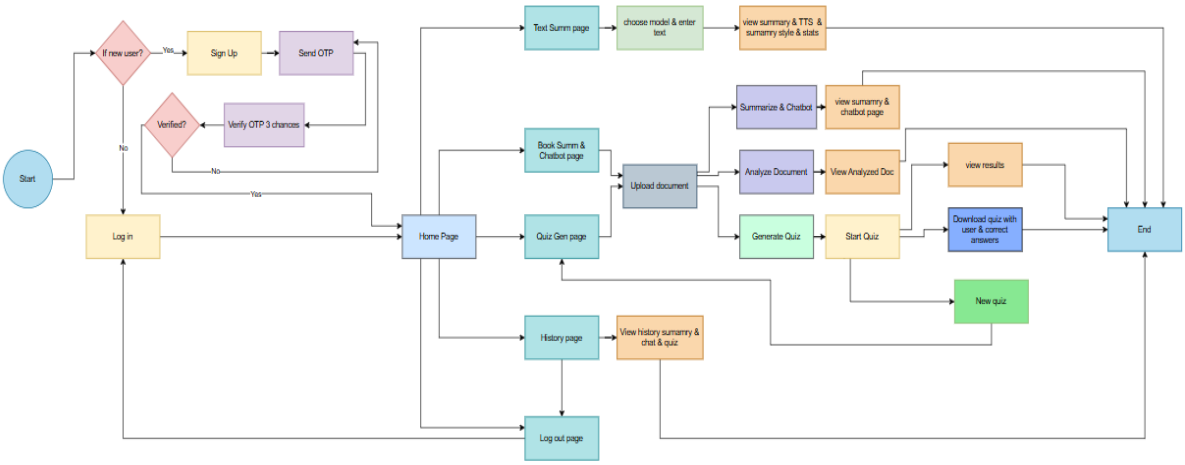


Figure: 9

The proposed system is a web-based platform that integrates advanced NLP and AI models to enable text summarization, book/document summarization with chatbot interaction, and automatic quiz generation from uploaded content. The system is designed to provide a seamless, intuitive experience for users with secure access and robust processing pipelines.

The system architecture follows a modular design that separates concerns into authentication, document processing, summarization, quiz generation, and interaction modules. The primary components of the system design include:

3.2.1 User Authentication & Security

- The process initiates with a decision point determining if the user is new.
- New users are directed to a" Sign Up" page, followed by an OTP (One-Time Password) verification with up to three attempts.
- Successful verification grants access to the" Home Page," serving as the central hub for subsequent interactions.

3.2.2 Home Page Navigation

The home page acts as a central hub providing access to:

- Text Summarization Module.
- Document and URL Summarization & Chatbot Module.
- Text to Speech Module.
- Quiz Generation Module
- History Module
- Log Out

Each module is encapsulated to allow independent scaling and maintenance.

3.2.3 Text Summarization Module

- **Input:** Users can enter custom text and select the summarization model and desired summary style.
- **Processing:** The backend processes the text, generating summaries along with text-to-speech (TTS) outputs and summary statistics.
- **Output:** Users can view the generated summary, listen via TTS, and analyze summary metrics.

3.2.4 Document & URL Summarization Module

- **Upload:** Users upload documents in various formats (PDF, DOCX, TXT) and URLs.
- **Summarize & Chat:** The system analyzes the document and generates a detailed summary. Users can interact with the content through a chatbot interface to ask questions about the uploaded document.
- **Analytics:** The analyzed document can be viewed with highlights and contextual insights.

3.2.5 Quiz Generation Module

- **Generate Quiz:** Users can generate quizzes based on the uploaded documents. The system ensures closely related multiple-choice with four choices, T&F and essay Questions.
- **Take Quiz:** Users start the quiz, view questions sequentially, and submit answers.
- **Results & Retry:** Upon completion, users see their results and can either download the quiz with correct answers and the user answers or attempt a new quiz.

3.2.6 History Module

- Maintains a log of previous summaries, chatbot interactions, and quizzes, allowing users to revisit past sessions.

3.2.7 Scalability & Extensibility

- The modular design allows for future enhancements, such as additional content types or advanced AI models. The use of distinct pages for each function facilitates maintenance and updates without disrupting the overall system.

3.2.8 Logout & End Section

- Users can securely log out, terminating their session and ensuring data integrity.

3.3 Used Models

3.3.1 Text Summarization

The text summarization module leverages transformer-based models to generate concise and meaningful summaries from raw text input. Users can choose between raw paragraph summaries or bullet key points. The following models are utilized:

- **Gemini API:**
    - **Description**: The Gemini API, a proprietary large language model, is used for flexible text summarization, producing either raw or bulleted summaries based on user preference.
    - **Role**: Processes user-provided text to generate summaries in the desired format, handling diverse text lengths and styles
    - **Architecture**: Gemini employs a transformer-based encoder-decoder architecture optimized for natural language tasks. The proprietary model uses multi-layer attention mechanisms to capture long-range dependencies.
    - **Implementation**: Integrated via API calls for seamless, cloud-based summarization

- **BART-Large-CNN (Pre-trained):**
    - **Description**: BART (Bidirectional and Auto-Regressive Transformer) is a sequence-to-sequence model pre-trained on the CNN/DailyMail dataset. The facebook/bart-large-cnn model is used without fine-tuning.
    - **Role**: Generates abstractive summaries for short to medium-length texts, leveraging pre-trained weights optimized for news articles.
    - **Architecture**: BART combines a bidirectional encoder (BERT-like) with an autoregressive decoder (GPT-like), with 406M parameters, 12 encoder and decoder layers, and a hidden size of 1024. It uses a denoising pre-training objective for robustness.
    - **Model Link**: https://huggingface.co/facebook/bart-large-cnn

- **BART-Large-CNN with PEFT LoRA Fine-Tuning:**

  - **Description**: A fine-tuned version of BART-Large-CNN using Low-Rank Adaptation (LoRA) on the preprocessed CNN/DailyMail dataset (mzizo4110/cnn-dailymail-final-preprocessed). Fine-tuning enhances performance on diverse inputs.
  - **Role**: Produces high-quality abstractive summaries with improved coherence and relevance, tailored to the project's dataset.
  - **Model Link**: https://huggingface.co/mzizo4110/BART-SUMMARIZATION-5
  - **Dataset Link**: https://huggingface.co/datasets/mzizo4110/cnn-dailymail-final-preprocessed

  - **Architecture:** Identical to pre-trained BART, with LoRA applied to attention projections (q_proj, v_proj, out_proj, k_proj). LoRA reduces trainable parameters to 1.1481% (rank=16, LoRA alpha=32).

```
lora_config = LoraConfig(
    r=16,
    lora_alpha=32,
    target_modules=["q_proj", "v_proj", "out_proj", "k_proj"],
    lora_dropout=0.1,
    bias="none",
    task_type=TaskType.SEQ_2_SEQ_LM
)

# Apply LoRA to the model
model = get_peft_model(base_model, lora_config)
model.print_trainable_parameters()
```

```
trainable params: 4,718,592 || all params: 411,009,024 || trainable%: 1.1481
```

Figure: 10

- **T5-Small with PEFT LoRA Fine-Tuning:**

  - **Description**: T5 (Text-to-Text Transfer Transformer) frames NLP tasks as text-to-text tasks. The t5-small model (60M parameters) is fine-tuned with LoRA on the CNN/DailyMail dataset.

  - **Role**: Generates concise summaries for short to medium-length texts, offering a lightweight alternative to larger models.

  - **Model Link**: https://huggingface.co/mzizo4110/Summarization_Continue

- **Dataset Link**: https://huggingface.co/datasets/mzizo4110/cnn-dailymail-final-preprocessed
- **Architecture**: T5-Small has 6 encoder and 6 decoder layers, with a hidden size of 512. LoRA is applied to query and value projections (q, v). LoRA reduces trainable parameters to 0.4850% (rank=8, LoRA alpha=32).

```
# Load the T5 model
base_model = AutoModelForSeq2SeqLM.from_pretrained(model_name)

# Define LoRA configuration for T5
lora_config = LoraConfig(
    r=8,
    lora_alpha=32,
    target_modules=["q", "v"],  # T5 attention query and value projections
    lora_dropout=0.1,
    bias="none",
    task_type=TaskType.SEQ_2_SEQ_LM
)

# Apply LoRA to the model
model = get_peft_model(base_model, lora_config)
model.print_trainable_parameters()
```

```
trainable params: 294,912 || all params: 60,801,536 || trainable%: 0.4850
```

Figure: 11

- Pegasus-CNN (Pre-trained).
  - **Description**: Pegasus is pre-trained on the CNN/DailyMail dataset with a gap-sentence objective for abstractive summarization. The google/pegasus-cnn_dailymail model is used without fine-tuning.
  - **Role**: Produces highly compressed summaries with strong content coverage, ideal for users needing concise outputs.
  - **Architecture**: Pegasus-CNN has 568M parameters, 12 encoder and decoder layers, and a hidden size of 1024. Its pre-training focuses on generating missing sentences.
  - **Model Link**: https://huggingface.co/google/pegasus-cnn_dailymail

3.3.2 Document Summarization & URL & Chatbot QA

The book summarization and QA module uses a Retrieval-Augmented Generation (RAG) framework with the Gemini API. Gemini API with RAG (LangChain STUFF, Chroma Vector DB):

- **Description:** The Gemini API is combined with a RAG pipeline using LangChain's STUFF chain and Chroma vector database for book summarization and contextual question answering. The RAG framework retrieves relevant book content and generates summaries or answers based on user queries.
- **Role:** Summarizes book content and enables interactive QA by retrieving relevant passages and generating contextually accurate responses. Th
- STUFF chain concatenates retrieved documents for processing, while Chroma stores vectorized book content for efficient retrieval.

- **Architecture**: Gemini's transformer-based model serves as the generative component, with the RAG pipeline using Chroma for vector search (based on embeddings) and LangChain's STUFF chain to combine retrieved documents into a single context for generation. The system leverages cosine similarity for document retrieval.
- **Implementation**: Integrated via LangChain for document processing and Chroma for vector storage, with API calls to Gemini for generation.
- **Process Diagram:**
    - Parsing Process Diagram: This diagram illustrates the initial parsing process in the RAG pipeline, where various input sources (e.g., documents like PDF, Word, text files, web pages, databases/dataframes, and video transcriptions) are broken down into smaller chunks (Chunk 1, Chunk 2, ..., Chunk n). The parsing step ensures that large documents are segmented into manageable pieces for further processing by the LLM, facilitating efficient summarization and QA.

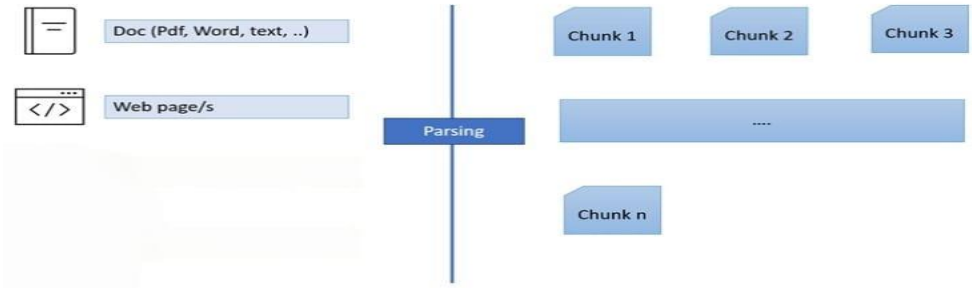


Figure: 12

- LLM Embedding Generation: This diagram depicts the LLM generating embeddings for each parsed chunk (Chunk 1, Chunk 2, ..., Chunk n). The LLM processes the chunks and produces corresponding embeddings (Embed 1, Embed 2, ..., Embed n), which are vector representations used for semantic understanding and retrieval in the RAG system, enhancing the accuracy of book summarization and question answering.

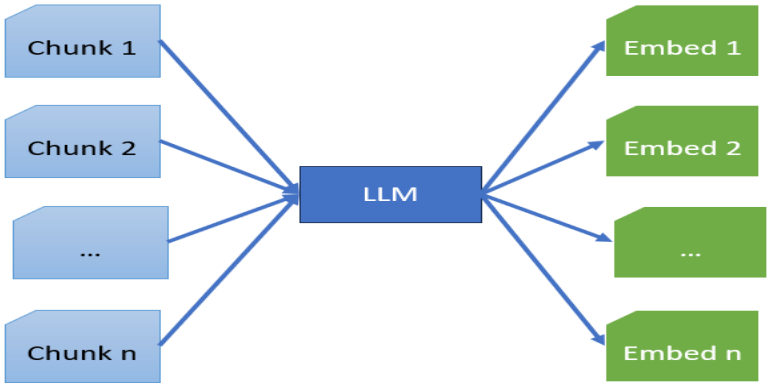


Figure: 13

- Indexing into Vector Database: This diagram illustrates the indexing process where the LLM generates embeddings for chunks (Chunk 1, Chunk 2, ..., Chunk n) and stores them in a vector-based database alongside the original chunks. This indexing enables efficient retrieval of relevant content, supporting the RAG pipeline's ability to handle large book datasets.

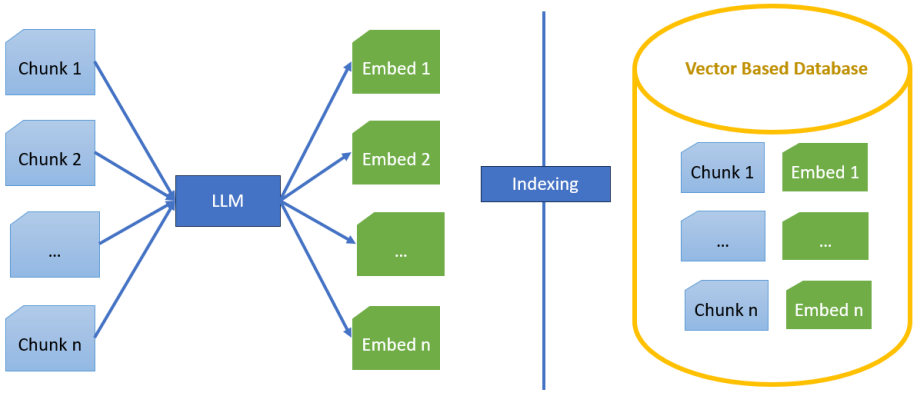


Figure: 14

- Vector-Based Database with Similarity: This diagram shows a vector-based database storing chunks and their embeddings (e.g., Chunk 1 with Embed 1, Chunk 71 with Embed 71, ..., Chunk n with Embed n). The LLM uses similarity matching (e.g., 78% similarity) to retrieve the most relevant chunk's

embedding based on a question, enabling precise context retrieval for summarization and QA

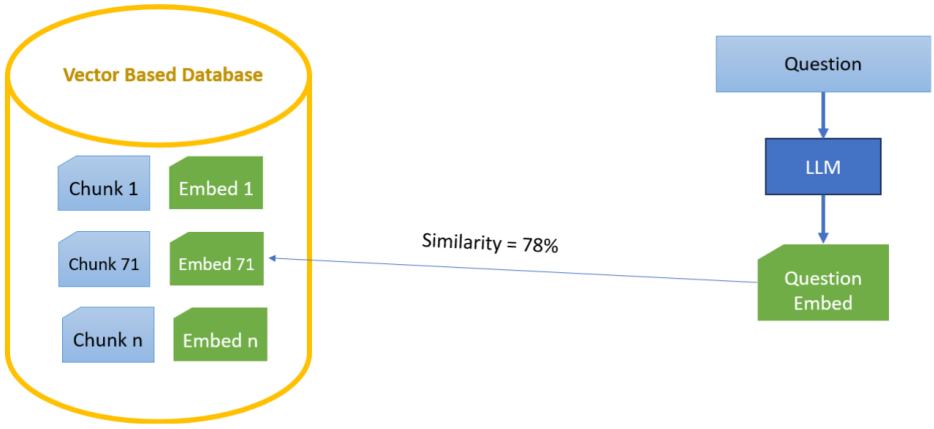


Figure: 15

- Most Similar Chunk Retrieval: This diagram depicts the retrieval of the most similar chunk (Chunk n with Embed n) based on a user question. The prompt combines the question with the retrieved text, which is then processed by the LLM to generate an answer, ensuring contextually accurate responses for book summarization and QA tasks.

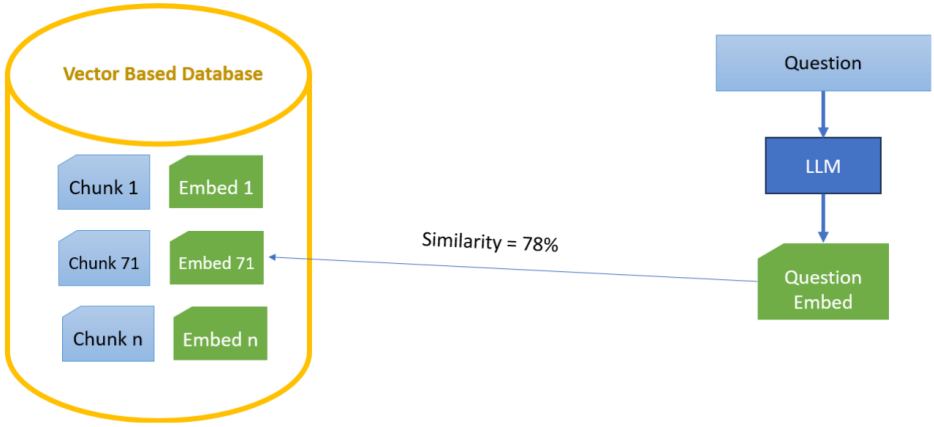


Figure: 16

- Stuff Method with Prompt: This diagram outlines the STUFF method in the LangChain RAG pipeline, where a user question is combined with multiple similar document contexts (Context 1, Context 2, Context 3) into a single prompt. The combined context is fed into the LLM, which generates an answer, ensuring comprehensive and context-aware responses for book-related queries

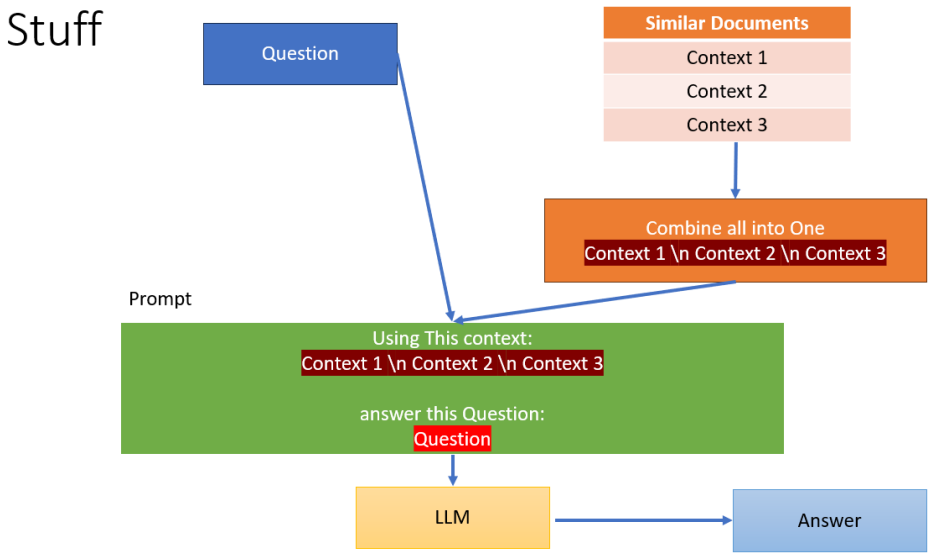


Figure: 17

3.3.3 Quiz Generation

The quiz generation module creates a comprehensive set of questions based on the content of uploaded documents, supporting multiple formats to enhance user learning and assessment. The Gemini API is utilized to generate questions tailored to user preferences.

- **Description**: The Gemini API generates a variety of question types, including 5 multiple-choice questions (MCQs) with 4 choices each, 3 true/false (T&F) questions, and 2 essay questions. Users can select the difficulty level (easy, medium, hard), which influences the complexity and depth of the questions.
- **Role**: Processes the uploaded document to generate questions with correct answers. For essay questions, it evaluates user-submitted answers post-quiz, providing scores (1-100) and feedback based on comparison with the correct answer. The system then calculates and displays the overall quiz result (0-100%) on the website.

- **Architecture:** Leverages Gemini's transformer-based architecture, fine-tuned for natural language generation and evaluation tasks. The model adapts question generation and scoring based on the document's content and user-selected difficulty
- **Implementation:** Integrated via API calls with tailored prompts. The process includes:
    - **Question Generation:** Extracts key concepts from the document and generates 5 MCQs (4 choices each), 3 T&F questions, and 2 essay questions, including correct answers.

- **Essay Evaluation:** After user submission, the Gemini API compares the user's essay answers against the correct answers, assigning a score (1-100) and generating feedback text highlighting strengths, weaknesses, and suggestions for improvement.
- **Result Display:** Computes the total score as a percentage (0-100%) based on MCQ, T&F, and essay performance, displayed on the website interface.

- **Workflow:**
  - The uploaded document is parsed and processed to identify relevant content.
  - The Gemini API generates questions tailored to the chosen difficulty level.
  - For essays, post-submission evaluation uses the API to assess responses and provide feedback.
  - The website updates with the final score and feedback for user review

Model	Pros	Cons
Gemini API	• Exceptional flexibility and high-quality output for summarization, QA, and quiz generation. • Seamless API integration enables fast, real-time use. • Handles complex books and essays; provides feedback and scoring.	• Reliance on cloud API may cause latency under heavy load. • Limited transparency in proprietary model.
BART-Large-CNN (Pre-trained & LoRA)	• Pre-trained: reliable out-of-the-box summaries (406M params). • LoRA fine-tuned: exceptional coherence with only 1.15% trainable params.	• Fine-tuning adds some computational cost. • May require adaptation for niche topics.

	• Efficient fine-tuning lowers resource needs.	
T5-Small (LoRA Fine-Tuned)	• Lightweight (60M params), fast inference. • LoRA fine-tuning improves diverse text handling. • Balances speed and quality well, ideal for real-time summarization.	• Less suited for very long documents. • Slightly lower scores on complex tasks.
Pegasus-CNN (Pre-trained)	• Gap-sentence pre-training gives highly compressed, content-rich summaries. • Excellent for news-like datasets. • High ROUGE-1 (44.16) validates strong performance.	

Table: 7

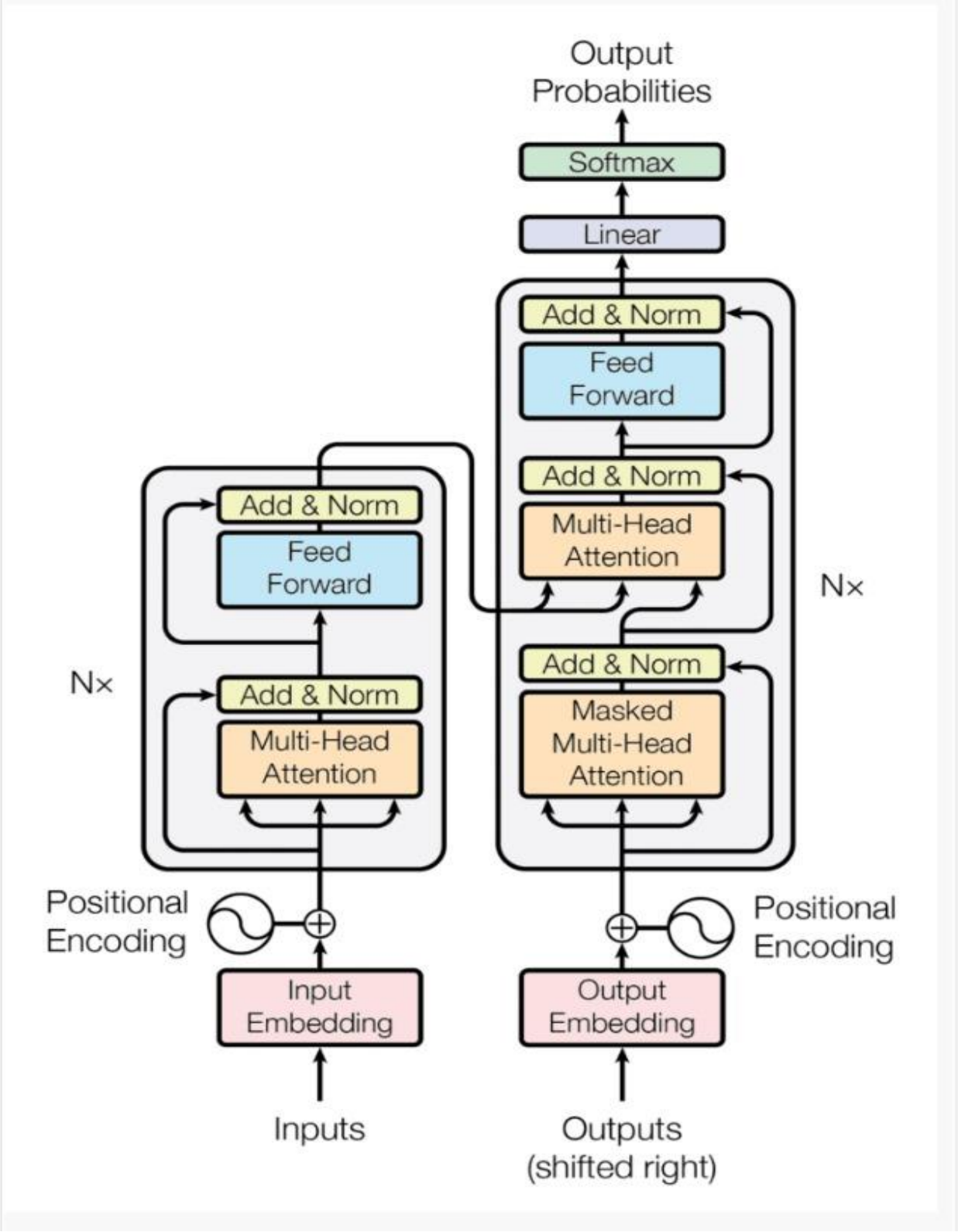

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

Figure: 18

3.4 Used Technologies

3.4.1 Model Technologies

The SumGenQ system integrates a suite of technologies to deliver its core functionalities: text summarization, book summarization, chatbot QA, quiz generation, and text-to-speech (TTS). Below, we outline the technologies, their roles, and their contributions to the system's performance and user experience.

- **Python**
  - **Role:** Primary programming language for backend development and model integration.**Purpose:** Implements core logic, data pipelines, and API integrations. Libraries like NumPy and Pandas support efficient data preprocessing.
- **Hugging Face Transformers**
  - **Role:** Primary programming language for backend development and model integration.
  - **Purpose:** Implements core logic, data pipelines, and API integrations. Libraries like NumPy and Pandas support efficient data preprocessing.
- **PEFT**
  - **Role:** Implements LoRA for fine-tuning BART and T5 models.
  - **Purpose:** Reduces computational cost by updating a small subset of parameters, enabling scalability on resource-constrained devices.
- **LangChain**
  - **Role:** Manages the RAG pipeline for book summarization and QA.
  - **Purpose:** Handles document retrieval and context aggregation using the STUFF chain, integrating with Chroma and Gemini API
- **Chrome Vector Database**
  - **Role**: Stores and retrieves vectorized book content for RAG.
  - **Purpose:** Enables fast vector search using embeddings and cosine similarity for efficient document retrieval.

- **Gemini API**
  - **Role:** Powers text summarization, book summarization, QA, and quiz generation.
  - **Purpose:** Provides a high-performance, cloud-based solution for NLP tasks, reducing local computational requirements.
  - **Implementation:** Integrated via secure API calls.

- **pyttsx3**

- **Role:** Converts text summaries to audio for TTS functionality.
- **Purpose:** Enhances accessibility for users with visual impairments or reading difficulties, providing natural-sounding speech output

- **FastAPI**
  - **Role:** Primary backend web framework.
  - **Purpose:** Hosts the SumGenQ API, enabling asynchronous handling of requests for text summarization, file uploads, and chatbot interactions.

- **BeautifulSoup & Spacy**
  - **Role:** Supports text preprocessing and cleaning.
  - **Purpose:** BeautifulSoup removes HTML tags and boilerplate, while SpaCy performs named entity recognition, lemmatization, and language detection for dataset preprocessing

- **Hugging Face Datasets**
  - **Role:** Provides access to the CNN/DailyMail dataset.
  - **Purpose:** Facilitates data loading and preprocessing for model training and evaluation.

- **Hardware Acceleration**
  - **Role:** Optimizes model inference and fine-tuning.
  - **Purpose:** Uses GPU acceleration with PyTorch and CUDA for faster training and inference.

3.4.2 Front-End Technologies

In this project, we designed and developed a responsive and user-friendly web application using Angular as the frontend framework and Bootstrap for styling and layout. The main goal was to create an accessible and aesthetically pleasing interface that connects seamlessly with the backend and provides a smooth user experience for summarization, chatbot interaction, quiz generation, and user authentication.

- **Technologies & Tools**
  - **Framework:** Angular
  - **Styling:** Bootstrap 5
  - **Languages:** TypeScript, HTML, CSS
  - **Other Tools:**
    - Angular CLI
    - RxJS for reactive programming

- ngx-toastr for alerts and messages
- Angular Router for navigation
- GitHub version control
- Vercel deployment

- **Application Pages**
  The Angular frontend consists of several interconnected pages and components, each tailored to a specific function:
    - **Landing Page:** Introduction to the app with navigation to login or signup.
    - **Login Page:** Authenticates users via email and password, handles validation, and stores the JWT token securely.
    - **Signup Page:** Registers new users with name, email and password with validation and feedback.
    - **Text Summarization Page:** Allows users to input text for summarization using the connected AI backend models and can use our TTS to listen to the summary.
    - **Book Summarization Page**: Users can upload documents; the model extracts and summarizes content.
    - **Quiz Generation Page:** Generates quiz questions from uploaded documents with choose the difficulty level and can download the quiz with answers.
    - **Book Summary View Page:** Displays summaries of uploaded books in a structured and readable format and a chatbot conversational interface connected to a generative AI model; supports both input and response
    - **History Page:** Allows users to track previous interactions, including summaries, quizzes, and chatbot questions.

- **Features & Functionality**
    - **JWT Authentication:** Secured login and signup using JSON Web Tokens.
    - Dynamic Routing: Angular Router is used to manage navigation and protect routes based on user authentication.
    - **Form Validation:** All forms include real-time validation with helpful error messages.
    - Responsive Design: Built with Bootstrap to ensure responsiveness across devices.
    - **API Integration**: All pages interact with backend APIs via Angular's HttpClient service to retrieve or send data.
    - **State Management:** Managed using services and localStorage/sessionStorage where appropriate.

- **Feedback Mechanisms:** Toast notifications used to inform users of success, error, or processing states.

- **Deployment**
  The Angular frontend was deployed using Vercel, ensuring fast performance, easy CI/CD integration, and seamless updates from GitHub. The environment variables and base URLs were configured securely to ensure proper communication with the backend.

3.4.3 Back-End Technologies

- **Data Management & Storage**

The backend handles structured data storage using **PostgreSQL**, a powerful relational database system known for its reliability and performance. **SQLAlchemy** is used as the ORM (Object-Relational Mapping) tool to manage and interact with the database efficiently. **SQLAlchemy** provides robust solutions for maintaining data consistency through features like foreign key constraints and relational modeling. By normalizing the data into related tables, we ensure flexibility, scalability, and efficient querying, especially as the dataset grows in size and complexity.

- Our database is designed to accommodate the following key collections:
  - **Users:** Stores user accounts and authentication credentials, including email and password. Each user is linked to their own summaries and chat history, enabling personalized access and interactions.
  - **TextSummary:** Stores user-submitted text and its corresponding summarized version. Each summary is linked to the user who submitted it and optionally categorized by topic.
  - **BookSummary:** Similar to TextSummary, this table stores summaries of entire books submitted by users. It includes the original book content, the summary, and an optional topic.
  - **ChatStorage:** Holds the interactive Q&A sessions related to summarized books. Each chat entry is linked to both the user and the specific book summary it references, storing the user's question and the system's answer for future access
  - **System EDR:**

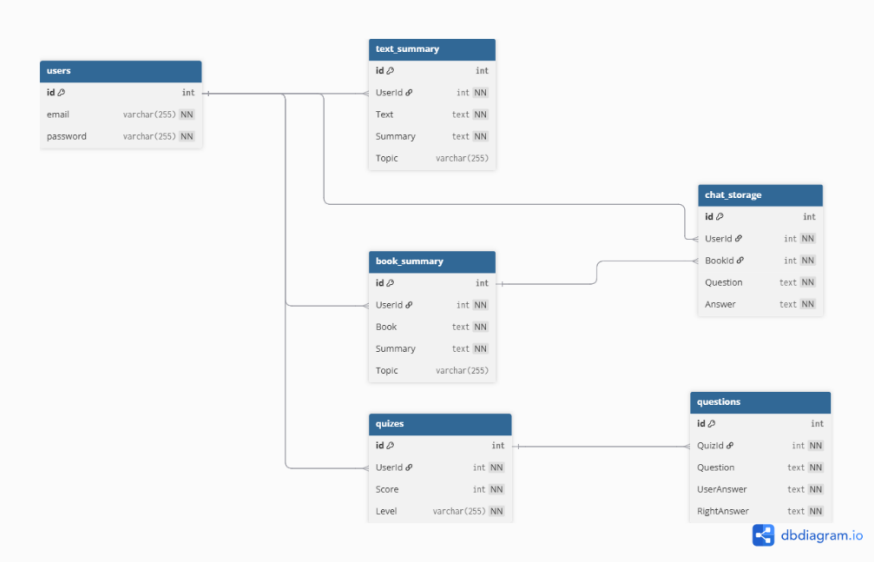


Figure: 19

- **User Authentication & Authorization:**

Secure user authentication is implemented using **JWT (JSON Web Tokens),** ensuring stateless and efficient session management. The authentication system includes:

- **User Registration & Login:** Users register and log in using their email and password. Upon successful authentication, a JWT token is issued to manage access securely.
    - **Token-Based Authentication:** JWT tokens include user information and expiration data, and are required for accessing protected routes. Tokens are verified on each request using a custom decorator to ensure session validity.
    - **Token Expiration & Renewal:** Tokens are set to expire after 7 days. If needed, new tokens can be generated—for example, during re-login or upon successful OTP verification.
    - **Account Deletion & Token Invalidation:** When a user deletes their account, their data is permanently removed from the database. Since JWT tokens are stateless, they automatically become invalid when associated user records no longer exist.
    - **Email Verification with OTP:** During registration, a 6-digit OTP is sent to the user's email to verify their identity before creating the account. OTPs are time-limited and securely stored in memory before account creation

- **Content Delivery & API Management**

The backend provides structured RESTful APIs to serve content dynamically. This includes:

  - **Summary Management:** Users can add, retrieve, and delete both text and book summaries. Each summary is linked to the authenticated user and categorized by topic for easier access.
  - **Interactive Chat Storage:** Users can store and retrieve question-answer pairs related to book summaries, enabling a conversational interface for reviewing content.
  - **User Progress & Personalization:** Each user's summaries and chat sessions are stored under their account, allowing them to resume, review, or delete their data at any time.

All API responses follow a standardized **JSON format** to ensure consistency, clear status reporting, and ease of integration with the frontend

- **User Progress Tracking**

  Progress tracking is implemented using user-linked records stored in relational tables:
  - **Summary Records:** Both text and book summaries are associated with each user. These serve as progress indicators by capturing what the user has read, summarized, and reviewed.
  - **Chat Interactions:** ChatStorage maintains a history of question-answer pairs linked to specific book summaries, allowing users to revisit their previous interactions and continue learning seamlessly.
  - **User-Centric Design:** While there is no explicit 'storyProgress' or 'quizProgress' table, each user's content and interactions are stored relationally via foreign keys. This supports personalized experiences and lays the groundwork for future features like rankings, achievements, or recently viewed items

This enables personalized learning experiences, allowing users to track their achievements.

- **Interactive Features & AI Integration**

  To enhance engagement, the backend integrates AI-powered voice and text-based Q&A using Google Generative AI. Key features include:

- **Chatbot Assistance**: Provides real-time responses to historical questions based on data entered by educators.
- **Voice Commands**: Enables users to interact with the app using voice input, with responses delivered through voice output.

This AI-driven approach improves accessibility and learning retention, ensuring that interactions are tailored to the educational content provided.

This part of Application is built using **FastAPI**, a high-performance web framework for Python, ensuring efficient request handling and real-time responses. The FastAPI backend powers the model's ability to process educator-uploaded data files, improving the accuracy of responses. It provides a robust API for configuring and interacting with the AI model, making it easily accessible for frontend integration.

- **API Endpoints**
  - **Authentication APIs**
    - **User Registration – Send OTP**
      - **URL:** / Register
      - **Method:** POST
      - **Description:** Starts the registration process by sending a 6-digit OTP to the user's email.
      - **Request Body:**

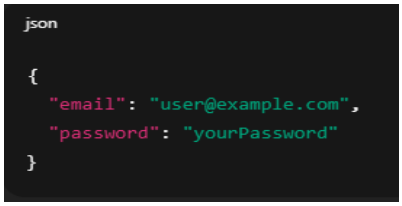


Figure: 20

      - **Status Codes:**
        - **201** OK: OTP sent to your email.
        - **400** Bad Request: Missing email or password / user already exists.
        - **500** Internal Server Error: Failed to send OTP or server error.

    - **Verify OTP – Complete Registration**
      - **URL:** / VerifyOTP
      - **Method:** POST

- **Description:** Verifies the OTP and creates a new user account. Also returns a JWT token.
- **Request Body:**

Figure: 21

- **Status Codes:**
  - **200** *OK*: User registered successfully, token returned.
  - **401** *Unauthorized*: Invalid OTP.
  - **404** *Not Found*: No OTP request found.
  - **410** *Gone*: OTP expired.
  - **500** *Internal Server Error*: Server error during registration.

- **Login to account**

  - **User Login**
  - **URL:** /Login
  - **Method:** POST
  - **Description:** Authenticates the user using email and password, and returns a JWT token.
  - **Request Body:**

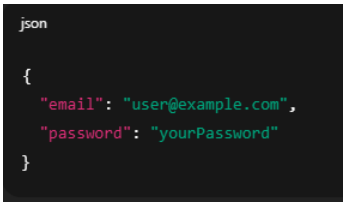


Figure: 22

  - **Status Codes:**
    - **200** *OK*: Login successful, token and user data returned.
    - **401** *Unauthorized*: Invalid email or password.
    - **500** *Internal Server Error*: Server error during login.

- **Verify JWT Token**

  - **URL:** /VerifyToken

- **Method:** POST
- **Description:** Verifies if a JWT token is valid and returns basic user info if it is.
- **Request Body:**

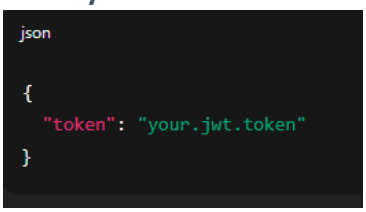


Figure: 23

- **Status Codes:**
  - **200** *OK*: Token is valid, user data returned**.**
  - **401** *Unauthorized*: Token is invalid or expired / user not found.
  - **500** *Internal Server Error*: Server error.

➢ **Text Summary APIs**

- **Store Text Summary**

- **URL:** / AddTextSummary
- **Method:** POST
- **Description:** Adds a new text summary for the authenticated user.
- **Request Body:**

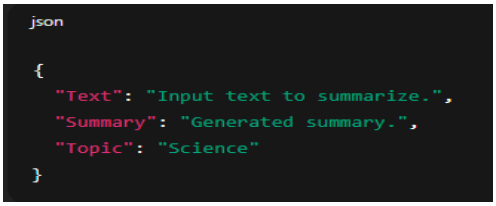


Figure: 24

- **Status Codes:**
  - **200** *OK*: Summary Added**.**
  - **401** *Unauthorized*: Missing Fields.
  - **500** *Internal Server Error*: DB Error.

- **Get All Text Summaries**

- **URL:** / TextSummarize
- **Method:** GET
- **Auth:** Bearer token
- **Description:** Retrieves all text summaries for the logged-in user.

- **Status Codes:**
  - **200** *OK*: Summaries returned**.**
  - **500** *Internal Server Error*: Fetching Failed.

- **Delete Sigle Summary**

  - **URL:** / DeleteTextSummary
  - **Method:** POST
  - **Auth:** Bearer token
  - **Description:** Deletes a specific text summary by ID.
  - **Request Body:**

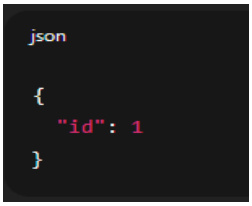


Figure: 25

  - **Status Codes:**
    - **200** *OK*: Deleted**.**
    - **400** *Not Found***:** Summary not Found.
    - **500** *Internal Server Error*: Fetching Failed.

- **Delete All Summaries**
  - **URL:** / DeleteAllTextSummaries
  - **Method:** POST
  - **Auth:** Bearer token
  - **Description:** Deletes all text summaries for the current user.
  - **Status Codes:**
    - **200** *OK*: All Deleted**.**
    - **500** *Internal Server Error*: Process Failed.

➢ **Book Summary APIs**

- **Store Book Summary**

  - **URL:** / AddBookSummary
  - **Method:** POST
  - **Description:** Adds a new book summary for the authenticated user.
  - **Request Body:**

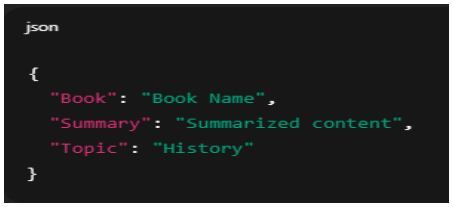


Figure: 26

- **Status Codes:**
  - **200** *OK*: Summary Added**.**
  - **401** *Unauthorized*: Missing Fields.
  - **500** *Internal Server Error*: DB Error.

- **Get All Book Summaries**

  - **URL:** / BookSummarizes
  - **Method:** GET
  - **Auth:** Bearer token
  - **Description:** Retrieves all Book summaries for the logged-in user.
  - **Status Codes:**
    - **200** *OK*: Book Summaries returned**.**
    - **500** *Internal Server Error*: Fetching Failed.
    -

- **Delete Single Book Summary**
  - **URL:** / DeleteBookSummary
  - **Method:** POST
  - **Auth:** Bearer token
  - **Description:** Deletes a specific book summary and its chat history.
  - **Request Body:**

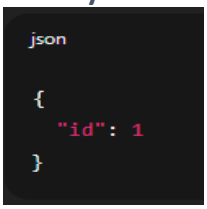


Figure: 27

  - **Status Codes:**
    - **200** *OK*: Deleted**.**
    - **400** *Not Found***:** Summary not Found.
    - **500** *Internal Server Error*: Fetching Failed.

- **Delete All Text Summaries**

  - **URL:** / DeleteAllBookSummaries
  - **Method:** POST
  - **Auth:** Bearer token
  - **Description:** Deletes all book summaries and related chat history.
  - **Status Codes:**
    - **200** *OK*: All Deleted**.**
    - **500** *Internal Server Error*: Process Failed.

➢ **Summary Chat APIs**

- **Add Chat Message**

  - **URL:** / AddChat
  - **Method:** POST
  - **Auth:** Break Token
  - **Description:** Saves a new question-answer chat entry linked to a book summary.
  - **Request Body:**

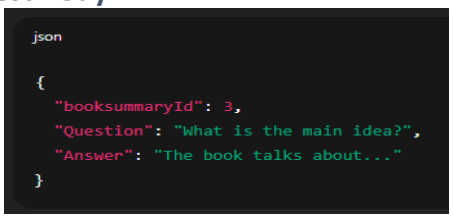


Figure: 28

  - **Status Codes:**
    - **200** *OK*: Chat Message Added**.**
    - **401** *Unauthorized*: Missing Fields.
    - **500** *Internal Server Error*: DB Error.

- **Get All Chat**

  - **URL:** / GetAllChat
  - **Method:** POST
  - **Auth:** Bearer token
  - **Description:** Retrieves all Q&A pairs linked to a given book summary.
  - **Request Body:**

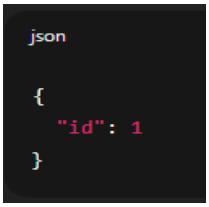


Figure: 29

- **Status Codes:**
  - **200** *OK*: Chat retrieved**.**
  - **400** *Not Found***:** Book ID Missing.
  - **500** *Internal Server Error*: Fetching Failed.

➢ **Quiz APIs**

- **Add Quiz**

  - **URL:** / AddQuiz
  - **Method:** POST
  - **Auth:** Break Token
  - **Description:** Saves a new quiz entry for a user along with a list of answered questions.
  - **Request Body:**

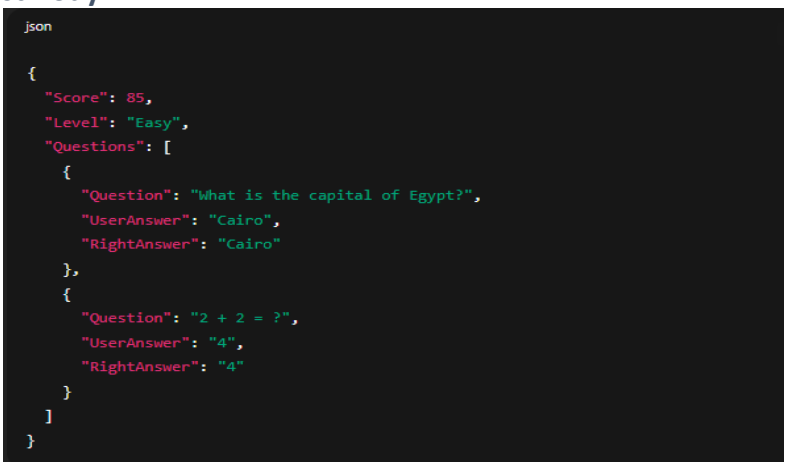


Figure: 30

- **Status Codes:**
  - **200** OK: Quiz Added Successfully**.**
  - **400** Bad Request: Missing fields (e.g., Score or Questions not provided)**.**
  - **401** Unauthorized: Invalid or missing token**.**
  - **500** Internal Server Error: Database error or server exception

- **Delete Quiz**

  - **URL:** /DeleteQuiz/<quiz_id>
  - **Method:** DELETE
  - **Auth:** Bearer token
  - **Description:** Deletes a specific quiz entry belonging to the authenticated user.
  - **Status Codes:**
    - **200** *OK*: Quiz deleted successfully**.**
    - **404** *Not Found***:** Quiz not found or user not authorized to delete it.
    - **401** *Unauthorized*: Invalid or missing token
    - **500** *Internal Server Error*: Database error or server exception.

- **Get Quiz**

  - **URL:** /GetUserQuizzes
  - **Method:** GET
  - **Auth:** Bearer token
  - **Description:** Retrieves all quizzes submitted by the authenticated user, including their questions and answers.
  - **Status Codes:**
    - **200** *OK*: Quizzes retrieved successfully**.**
    - **401** *Unauthorized*: Invalid or missing token

- **Deployment & Infrastructure**
  - **Database: PostgreSQL**

    The database for this application is deployed on **PostgreSQL**, a powerful open-source relational database system known for its robustness, security, and scalability. It is hosted in the cloud and managed using **SQLAlchemy**, a flexible and expressive Object-Relational Mapping (ORM) library for Python. The primary database schema includes structured tables for users, text summaries, book summaries, and chat history.

    Database connections are securely managed using environment variables to protect sensitive credentials. Authentication and access control are handled through secure **JWT**-based tokens, ensuring that only authorized users can access their data. The relational schema design enforces data integrity through foreign key relationships while supporting efficient queries and future scalability

- **Hosting Server: Vercel**

  To ensure seamless integration between the backend and frontend teams, the project is deployed on Vercel — a cloud platform offering fast, serverless hosting with automatic scaling. The backend is divided into two distinct deployments for modularity and performance optimization.

  - The **first deployment**, named gpFirst, is responsible for handling the machine learning model, which manages learning tasks and provides intelligent answers to user queries.
  - The **second deployment**, named summrizier, manages all core backend functionalities, including user authentication, data handling (text and book summaries), OTP verification, and chat storage via well-defined RESTful API routes.

  These structured deployments allow the frontend to interact with the backend using clean, consistent API endpoints, improving developer collaboration and testing. All APIs are accessible via Vercel-generated URLs, enabling real-time updates and quick iteration.

  Sensitive credentials, such as PostgreSQL connection strings and email service credentials, are securely stored as environment variables within Vercel. In addition, GitHub integration ensures that any changes pushed to the repository automatically trigger a new deployment — keeping the backend continuously updated and aligned with frontend development

- **Remote Repo Storage: GitHub**

  The project's source code is managed on **GitHub**, which serves as the version control system to track changes, collaborate effectively, and uphold code quality. A structured Git workflow is followed to ensure stability and scalability. Each new feature or fix is developed in a dedicated feature branch, isolating changes and preventing disruptions to the main codebase, Once development and local testing are completed, changes are submitted via a pull request (PR) to the main branch. This process ensures that all updates are reviewed, tested, and approved before being merged into production. The workflow promotes clean collaboration between the backend and frontend teams, enabling parallel development and clear code ownership.

  GitHub is also integrated with Vercel, allowing automatic deployments every time changes are merged into the main branch. This CI/CD setup ensures the backend remains up to date with no manual intervention, streamlining development and delivery.

  By following this branching strategy and automated deployment pipeline, the project maintains a clean, organized, and continuously evolving codebase

Chapter 4. Experimental Work and Results

4.1 Hyperparameters

This section outlines the hyperparameters utilized in the fine-tuning process of the BART (Bidirectional and Auto-Regressive Transformer) model, enhanced with Low-Rank Adaptation (LoRA), on the preprocessed CNN/DailyMail dataset for text summarization. The fine-tuning leverages the Hugging Face Transformers library and the PEFT (Parameter-Efficient Fine-Tuning) framework to optimize the model efficiently while maintaining high performance. The hyperparameters are carefully selected to balance training efficiency, model performance, and resource constraints, ensuring effective adaptation to the summarization task. Below, we detail the key hyperparameters for the LoRA configuration and the training arguments.

These hyperparameters ensure efficient fine-tuning of the BART model with LoRA, optimizing performance for text summarization while minimizing computational overhead.

4.1.1 Fine Tuning BART With LoRA: Training Results of epechs

Figure: 31

4.1.2 Fine Tuning T5 with LoRA: Training Results of 4 epochs

Figure: 32

4.1.3 Comparison between Them

- **Fine-Tuning BART with LoRA VS Fine-Tuning T5 with LoRA:**

comparative analysis between the fine-tuning processes of **BART** and **T5** using **LoRA (Low-Rank Adaptation)** for the summarization task. This breakdown focuses on differences and similarities in model configuration, LoRA settings, training strategies, and additional setups:

- **LoRA (Low-Rank Adaptation) Configuration Table**

Parameter	BART Configuration	T5 Configuration
Rank (r)	16 Controls the size of the low-rank matrices; more expressive with more trainable params.	8 Fewer trainable parameters; lightweight and efficient adaptation.
LoRA Alpha	32 Scaling factor for low-rank updates; adjusts impact of LoRA on the model.	32 Same function; balances LoRA adaptation strength.
Target Modules	["q_proj", "v_proj", "out_proj", "k_proj"]Focuses on key BART attention projections.	["q", "v"]Applies LoRA to query and value projections in T5 attention blocks.
LoRA Dropout	0.1 Prevents overfitting by randomly disabling parts of the LoRA layers.	0.1 Same regularization technique used during training.
Bias	"none" Bias terms are not updated in LoRA; focuses only on weight adaptation.	"none" Same setup to reduce unnecessary parameter updates.
Task Type	TaskType.SEQ_2_SEQ_LMSequence-to-sequence language modeling (e.g., summarization).	TaskType.SEQ_2_SEQ_LMSame task definition, suited for text generation tasks.

Table: 8

- **Models and Datasets Table**

Feature	BART Fine-Tuning	T5 Fine-Tuning
Base Model	facebook/bart-large-cnn	t5-small
Dataset	mzizo4110/cnn-dailymail-final-preprocessed	abisee/cnn_dailymail, version 3.0.0

Tokenizer	BART Tokenizer (same as base model)	T5 Tokenizer (same as base model)
Max Input Length	512 tokens	1024 tokens
Max Target Length	128 tokens	128 tokens

Table: 9

- **Training Arguments Table**

Parameter	BART Configuration	T5 Configuration
Output Directory	"BART-SUMMARIZATION-5"Directory to save checkpoints.	"Summarization-Continue"Directory to save checkpoints.
Evaluation Strategy	"steps"Evaluate at set step intervals.	"steps"Same strategy for periodic evaluation.
Save Strategy	"steps"Save checkpoints periodically.	"steps"Same.
Evaluation Steps	250Evaluate every 250 steps.	500Evaluate every 500 steps.
Save Steps	1000Checkpoint every 1000 steps.	1000Same.
Train Batch Size (per device)	16Samples per batch per GPU/CPU.	16Same.
Eval Batch Size (per device)	16Samples per batch for evaluation.	16Same.
Gradient Accumulation Steps	4Accumulate gradients before backpropagation (16×4=64).	4Same effective batch size of 64.
Learning Rate	1e-4Initial LR for AdamW optimizer.	1e-4Same.

Weight Decay	0.01Regularization to avoid overfitting.	0.01Same.
Save Total Limit	2Keep only the last 2 checkpoints.	2Same.
Epochs (num_train_epochs)	5Total full passes over dataset.	4One epoch fewer.
Logging Directory	"./logs"Folder for training logs.	"./logs"Same.
Logging Steps	500Log metrics every 500 steps.	500Same.
Load Best Model at End	TrueRestores best checkpoint at end.	TrueSame.
Report To	"none"Disable Weights & Biases etc.	"none"Same.
Hub Model ID	"mzizo4110/BART-SUMMARIZATION-5"HF Hub model identifier.	"mzizo4110/Summarization_Continue"HF Hub model identifier.
Push to Hub	TrueUpload model after training.	TrueSame.
FP16	TrueUse mixed-precision to save memory and speed up.	TrueSame.

Table: 10

- **Optimizer, Scheduler, Data Collator, and Training Output Table**

Category	BART Configuration	T5 Configuration
Optimizer	AdamWEfficient optimizer with weight decay (0.01) to prevent overfitting.	AdamWSame configuration used for stable and regularized updates.
Learning Rate Scheduler	CosineScheduleWithWarmup Gradually increases LR, then decays using cosine curve. Warmup = 10%.	Linear (default for Seq2SeqTrainer)Increases LR linearly then decays steadily.

Early Stopping	patience=3Stops training after 3 evaluations without improvement.	patience=5More tolerant, waits for 5 bad evaluations before stopping.
Data Collator	label_pad_token_id = -100Ensures padding tokens are ignored during loss calculation.	DataCollatorForSeq2Seq (default)Also uses label_pad_token_id = -100 internally.
Training Duration	5 epochsTrained over 5 full passes of the dataset.	4 epochsTrained over 4 epochs (1 less than BART).
Evaluation Frequency	250 stepsMore frequent evaluation.	500 stepsLess frequent evaluation, which may speed up training.
Effective Batch Size	64 (16 × 4)Due to gradient accumulation steps.	64 (16 × 4)Same effective batch size as BART.
FP16 Mixed Precision	EnabledReduces memory use and speeds up training on compatible GPUs.	EnabledSame.
Model Output	mzizo4110/BART-SUMMARIZATION-5Uploaded to Hugging Face Hub.	mzizo4110/Summarization_ContinueUploaded to Hugging Face Hub.

Table: 11

4.2 Dataset Samples

- **Input**

  “Natural language processing (NLP) is a subfield of computer science and artificial intelligence (AI) that uses machine learning to enable computers to understand and communicate with human language.
  NLP enables computers and digital devices to recognize, understand and generate text and speech by combining computational linguistics, the rule-based modeling of

human language together with statistical modeling, machine learning and deep learning.
NLP research has helped enable the era of generative AI, from the communication skills of large language models (LLMs) to the ability of image generation models to understand requests. NLP is already part of everyday life for many, powering search engines, prompting chatbots for customer service with spoken commands, voice-operated GPS systems and question-answering digital assistants on smartphones such as Amazon's Alexa, Apple's Siri and Microsoft's Cortana. Benefits of NLP: NLP makes it easier for humans to communicate and collaborate with machines by allowing them to do so in the natural human language they use every day. This offers benefits across many industries and applications.
Automation of repetitive tasks
Improved data analysis and insights
Enhanced search
Content generation
Automation of repetitive tasks
NLP is especially useful in fully or partially automating tasks like customer support, data entry and document handling. For example, NLP-powered chatbots can handle routine customer queries, freeing up human agents for more complex issues. In document processing, NLP tools can automatically classify, extract key information and summarize content, reducing the time and errors associated with manual data handling. NLP facilitates language translation, converting text from one language to another while preserving meaning, context and nuances. NLP also plays a growing role in enterprise solutions that help streamline and automate business operations, increase employee productivity and simplify business processes."

4.3 Output Samples

- **Text Summary Models**
  - **Gemini Output**

    "*Natural Language Processing (NLP) uses machine learning to allow computers to understand and generate*

human language. This technology powers applications like search engines, chatbots, and voice assistants, automating tasks, improving data analysis, and enhancing communication between humans and machines across various industries."

- **Bulleted Gemini Output**

  "• NLP enables computers to understand and generate human language using machine learning.
  • It automates tasks (e.g., customer service, data entry), improves data analysis, and enhances search capabilities.
  • NLP powers everyday applications like search engines, chatbots, and voice assistants."

- **BART Output**

  "*Natural language processing (NLP) is a subfield of computer science. It uses machine learning to understand and communicate with human language. NLP is already part of everyday life, powering search engines and question-answering systems."*

- **T5-Small Output**

  "*converting text from one language to another while preserving meaning, context and nuances. NLP is a subfield of computer science and artificial intelligence (AI) that uses machine learning to enable computers to understand and communicate with human language. NLP enables computers and digital devices to recognize, understand and generate text and speech by combining computational linguistics, the rulebased modeling of human language together with statistical modeling, machine learning and deep learning.*"

- **BART Fune-Tuned Output**

  "natural language processing (NLP) is a subfield of computer science and artificial intelligence (AI) that uses machine learning to enable computers to understand

and communicate with human language. NLP enables computers and digital devices to recognize, understand and generate text and speech by combining computational linguistics, statistical modeling, machine learning and deep learning."

- **Pegasus**

  "Natural language processing (NLP) is a subfield of computer science and artificial intelligence (AI) that uses machine learning to enable computers to understand"

- **Models Results Compression Table**

	Input Text	Gemini	Bullet-Gem.	BART	FT-BART	T5-small	Pegasus
W-count	282	42	41	34	53	72	23
S-count	11	2	5	3	2	3	1
Chars	1998	334	98	238	392	511	161
Red. %	0	85	85	88	81	74	92
Time/s	0	2.25	7.50	15.70	19.8	10.60	22

Table: 12

Pegasus achieved the **highest compression (92%)** but was the **slowest** (22s) and the **largest model (~568M params)**. **Gemini** models were the **fastest** (as low as 2.25s) with strong reduction (85%). **BART** provided a balanced performance (**88% reduction**, ~406M params), while **Fine-tuned BART** achieved similar results using only **1.15% trainable**

parameters. **T5** was smaller (~60M) with moderate effectiveness.
Overall, there's a clear **trade-off between compression quality, speed, and model size.**

- **Book Summary Models**
  - **File name: genetic algorithm**
  - **File Summary:**

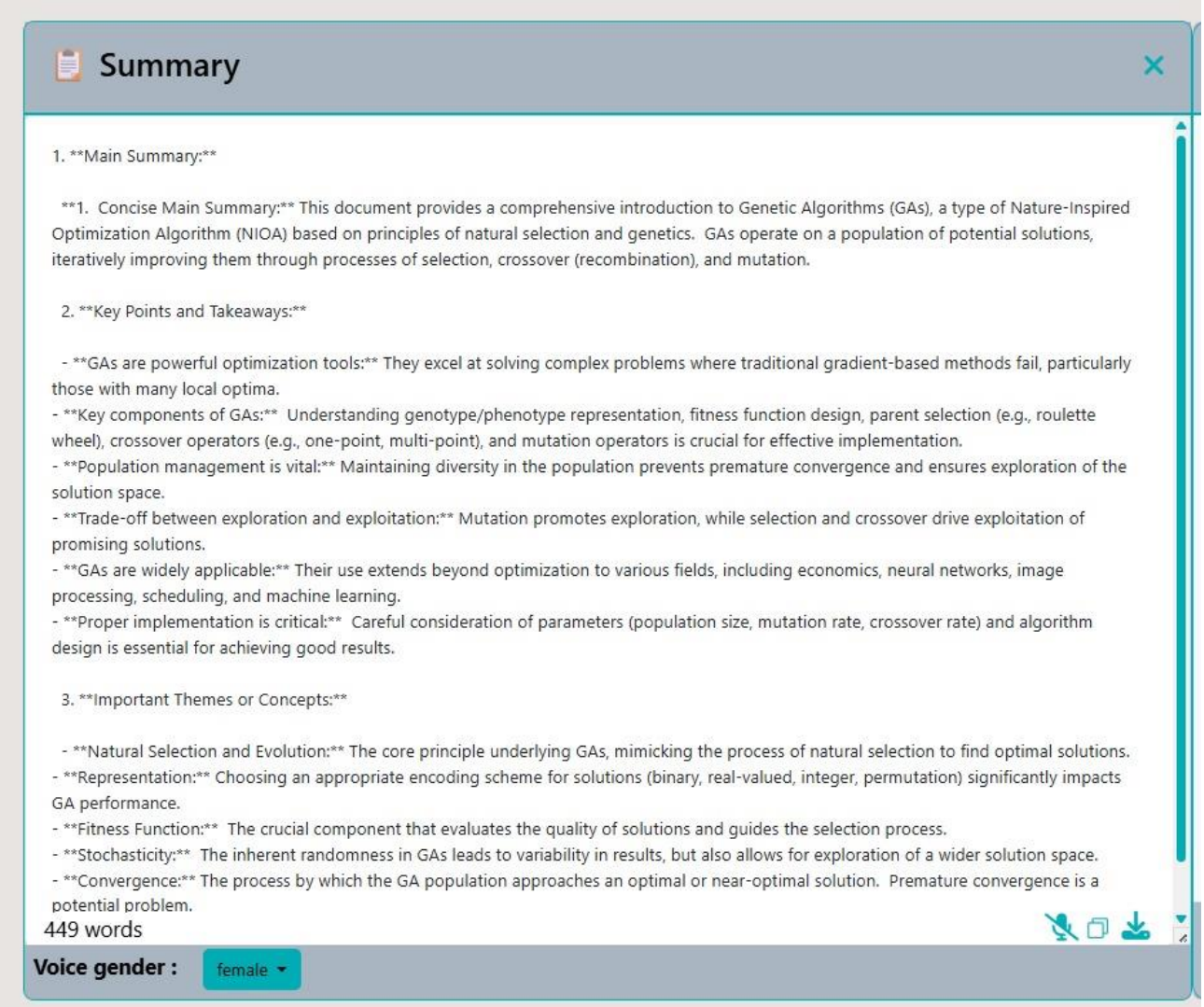


Figure: 33

- Summary Generating Time is**: 12.36s**
- **Analysis Uploaded File:**

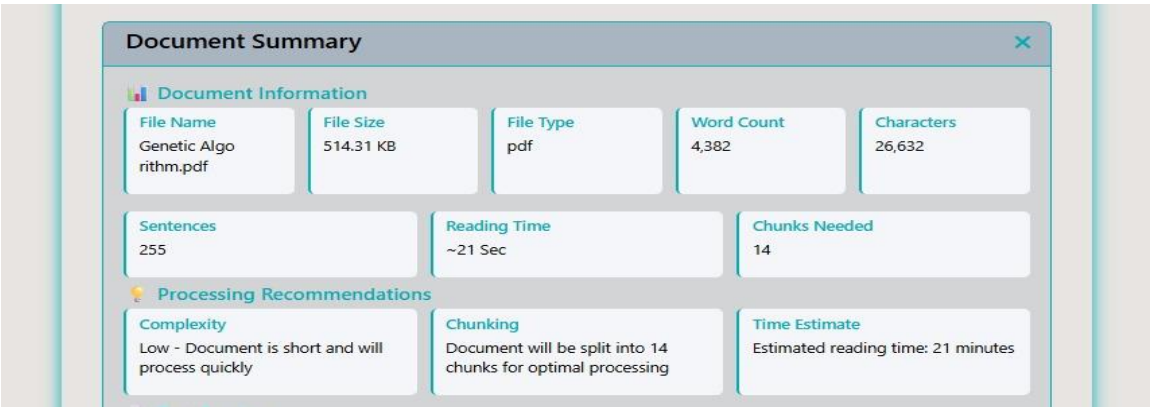


Figure: 34

➢ Chatbot QA about file:

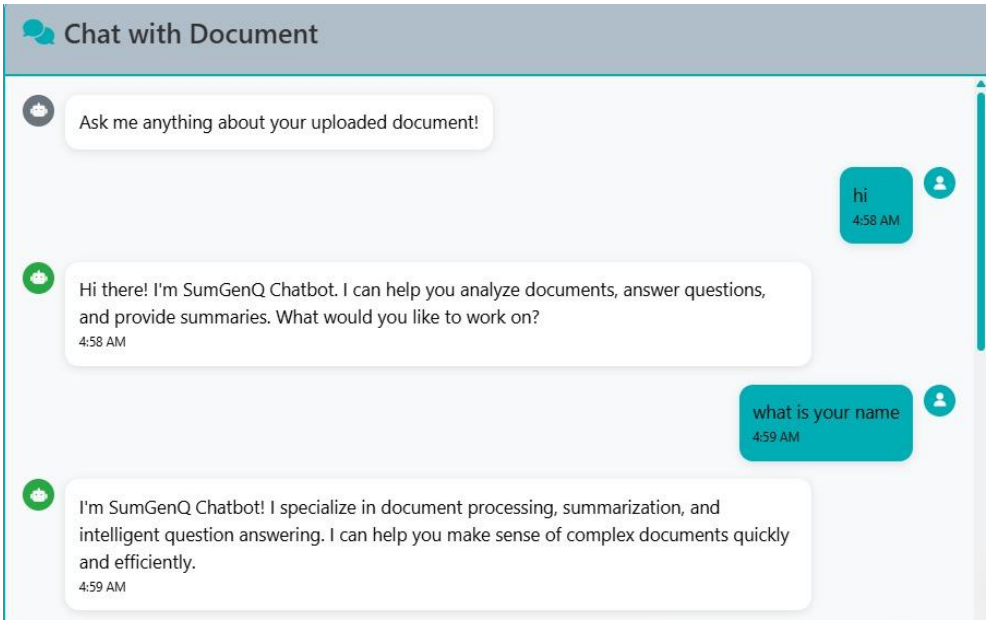


Figure: 35

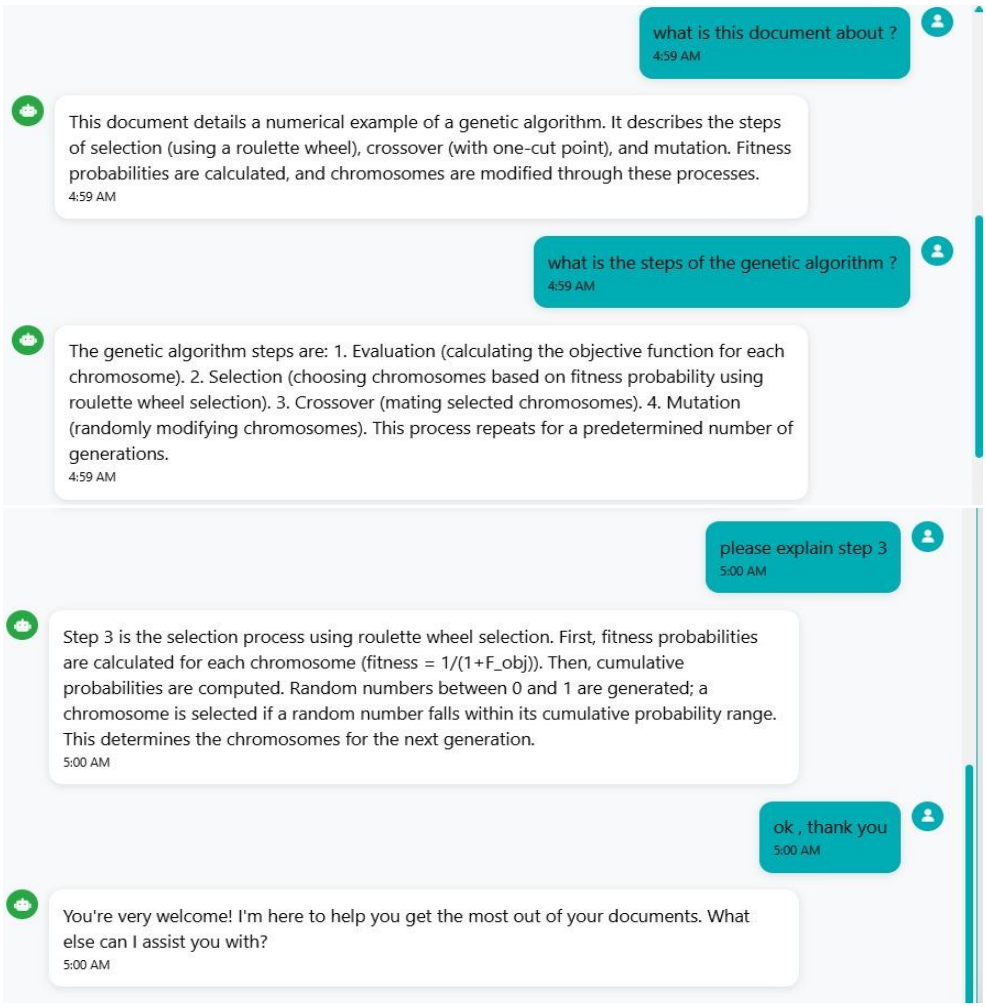


Figure: 36

➢ Quiz Generator

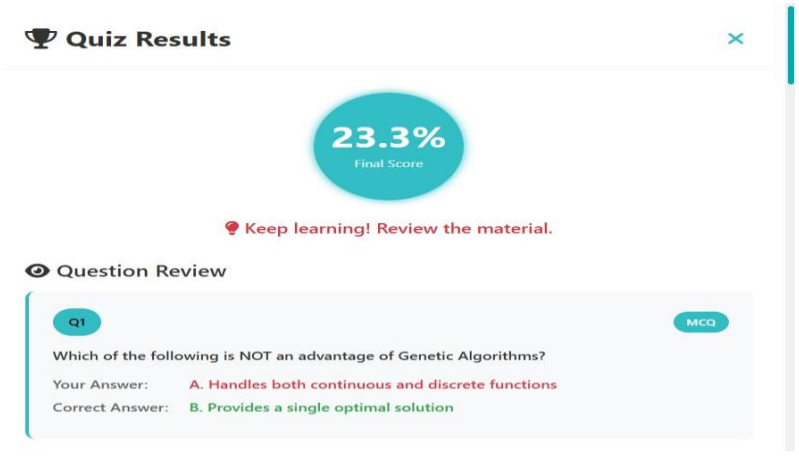


Figure: 37

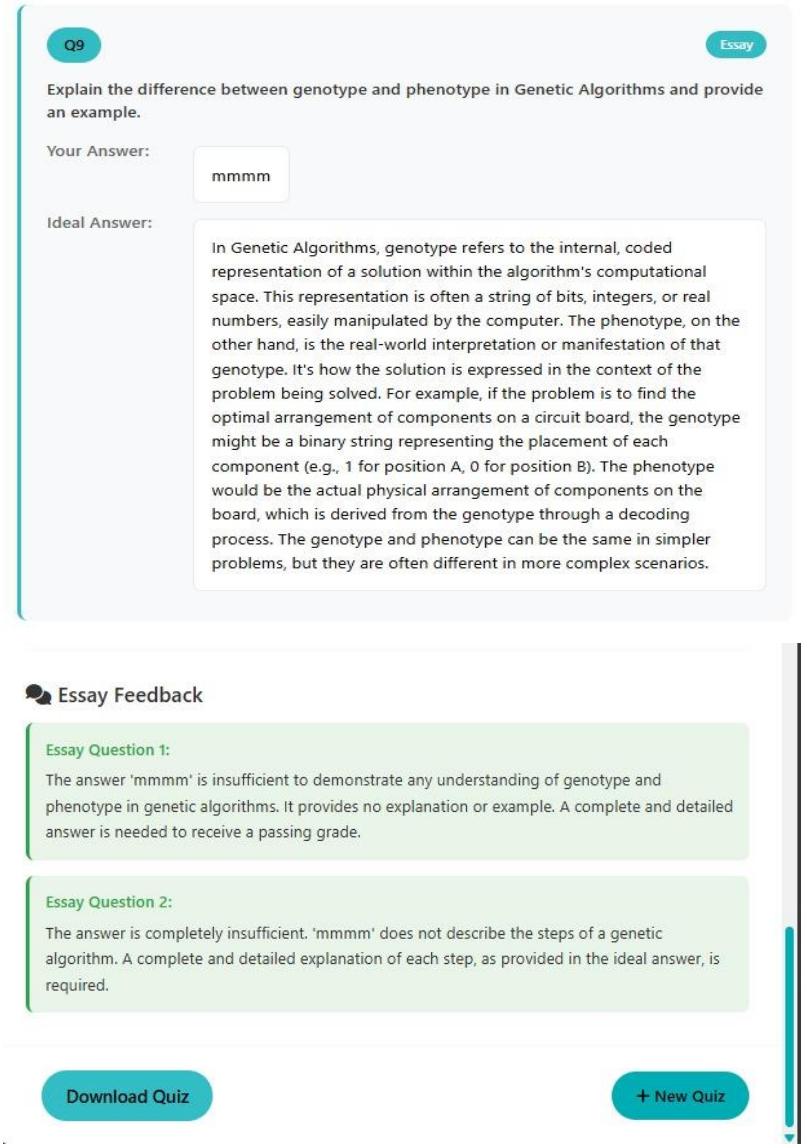


Figure: 38

4.4 Evaluation Metrics for Summarization Models on CNN/DailyMail Dataset

Overview of Evaluation Metrics

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) measures the overlap of n-grams, word sequences, and longest common subsequences between generated and reference summaries, with higher scores indicating better similarity. BLEU (Bilingual Evaluation Understudy) evaluates the precision of n-grams in generated summaries against

references, often used to assess fluency and correctness. Both metrics are standard for evaluating text summarization performance.

Evaluation Metrics Summary

- **ROUGE-1**: Measures unigram (single word) overlaps between generated and reference summaries, focusing on content coverage and recall.
- **ROUGE-2**: Evaluates bigram (two-word sequence) overlap, emphasizing fluency and phrase-level similarity.
- **ROUGE-L**: Assesses the longest common subsequence, capturing structural similarity and longer matching sequences.
- **ROUGE-Lsum**: Extends ROUGE-L to evaluate sentence-level coherence by considering the union of longest common subsequences across multiple reference sentences.
- **BLEU**: Measures n-gram precision (typically up to 4-grams) of generated summaries against references, adjusted by a brevity penalty, focusing on fluency and correctness.

Higher scores for all metrics indicate better alignment with reference summaries, with ROUGE emphasizing recall and BLEU focusing on precision.

➢ **Model Performance Metrics**

- **T5-Small (Fine-Tuned)**

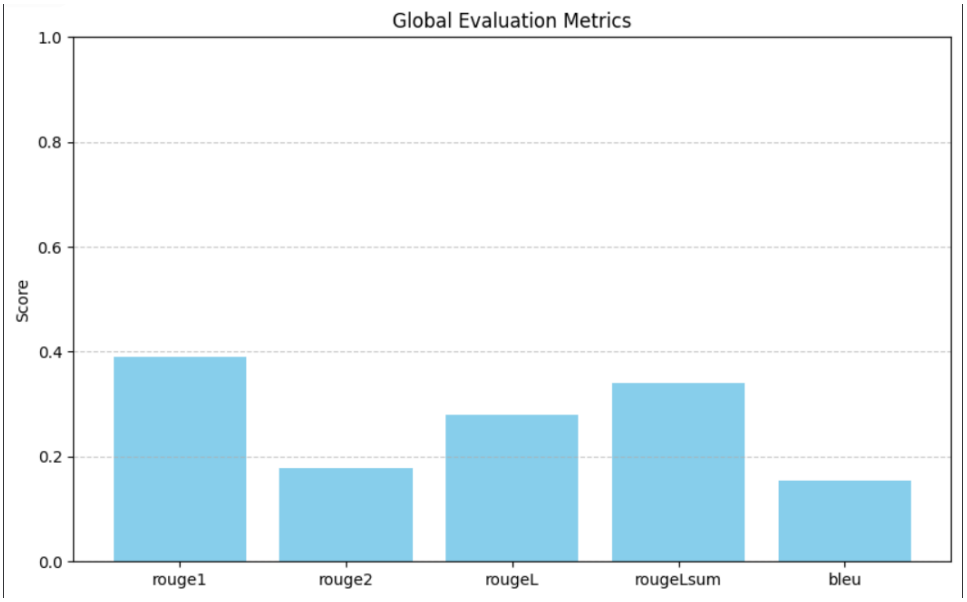


Figure: 39

- **BART-Large-CNN (Pre-Trained)**

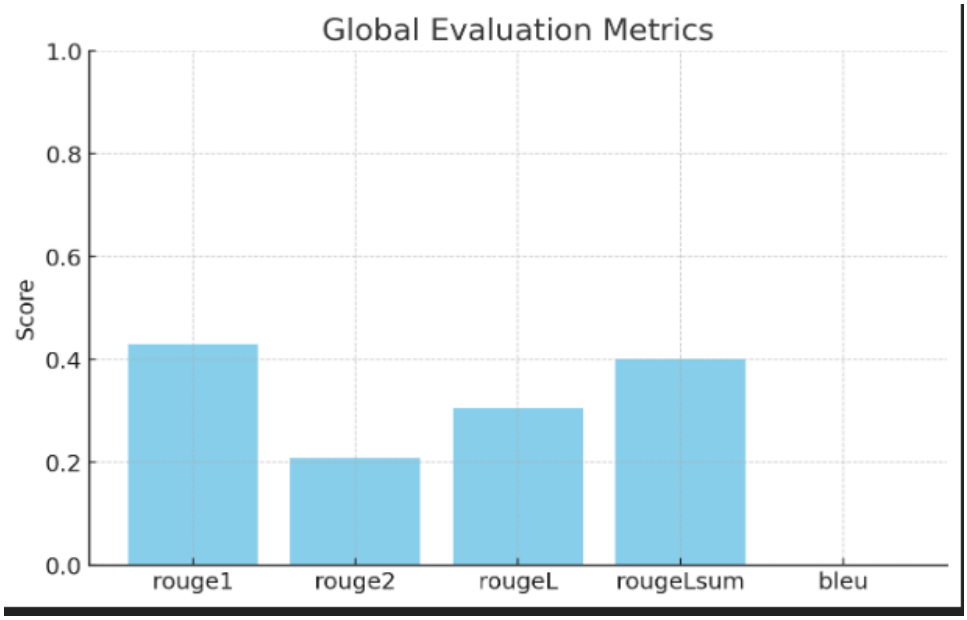


Figure: 40

- **Pegasus-CNN (Pre-Trained)**

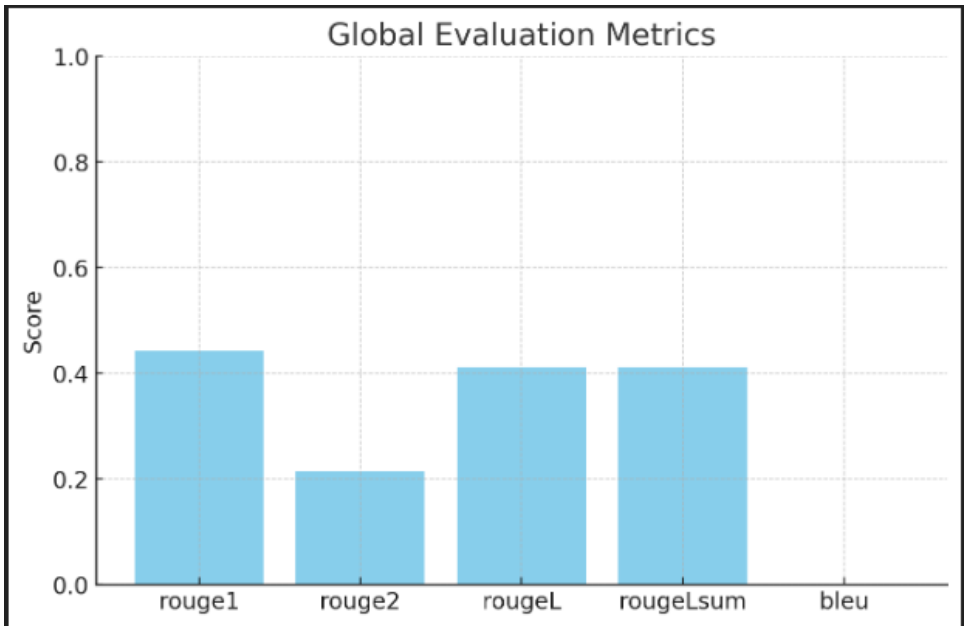


Figure: 41

- **Comparison Evaluation Results**

Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum	BLEU
T5-Small (Fine-Tuned)	0.3907	0.1784	0.2795	0.3400	0.1545
BART-Large-CNN (Fine-Tuned)	0.3906	0.1844	0.2912	0.2912	0.1402
BART-Large-CNN (Pre-Trained)	0.4295	0.2082	0.3062	0.4004	N/A
Pegasus-CNN (Pre-Trained)	0.4417	0.2147	0.4111	0.4111	N/A

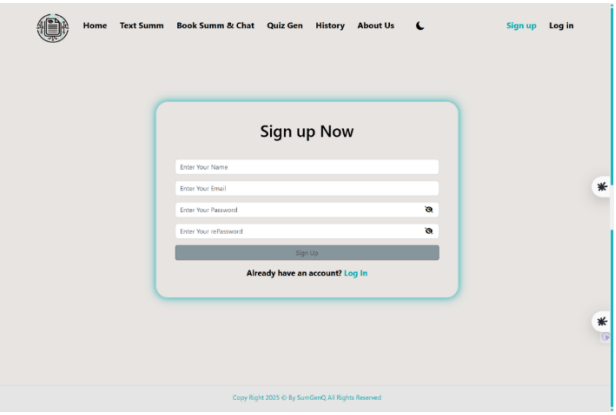
Table: 13

- **Observations**
  - **Pegasus-CNN (Pre-Trained)** achieves the highest ROUGE scores across all variants, indicating superior performance in capturing reference summary content.
  - **T5-Small (Fine-Tuned)** outperforms **BART-Large-CNN (Fine-Tuned)** in BLEU and ROUGE-Lsum, suggesting better fluency and summary coherence despite being a smaller model.
  - **BART-Large-CNN (Pre-Trained)** shows strong performance, particularly in ROUGE-Lsum, compared to its fine-tuned counterpart.
  - **BLEU** scores are only available for fine-tuned models, limiting direct comparisons with pre-trained models.

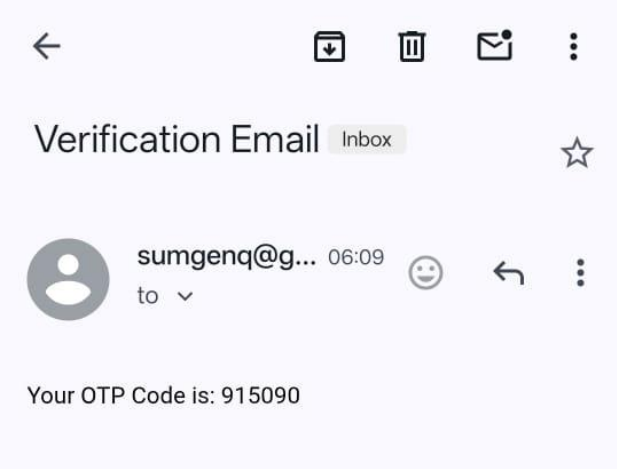
Chapter 5. User Interface

5.1 Authentication

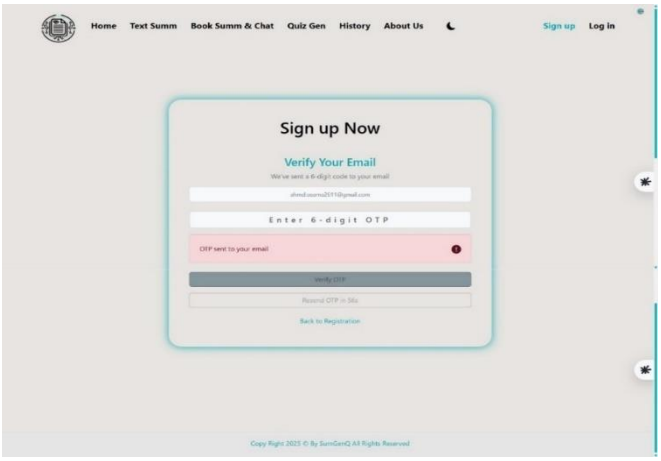
- **Create a New Account**



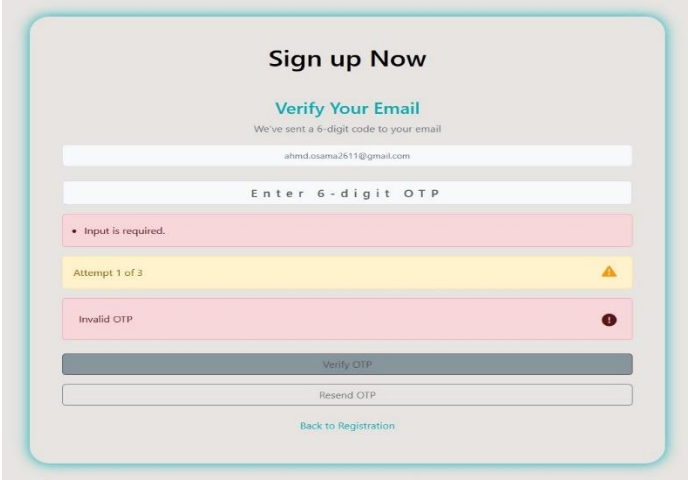
- **Receive OTP via Email**



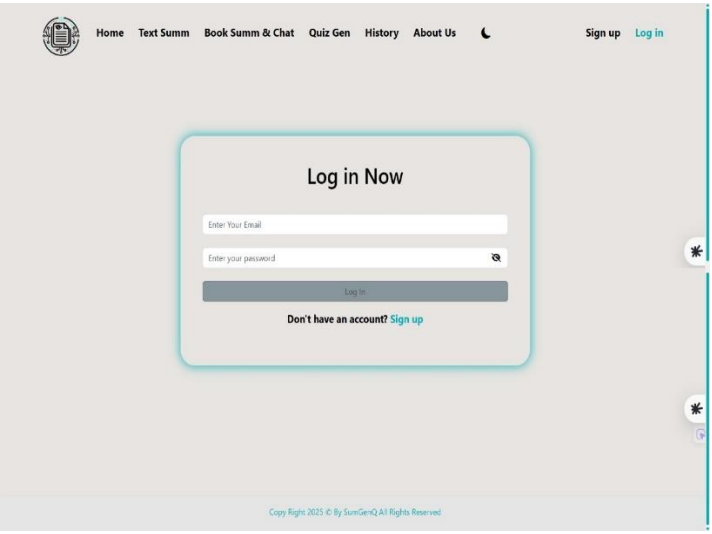
- **Verify email account**



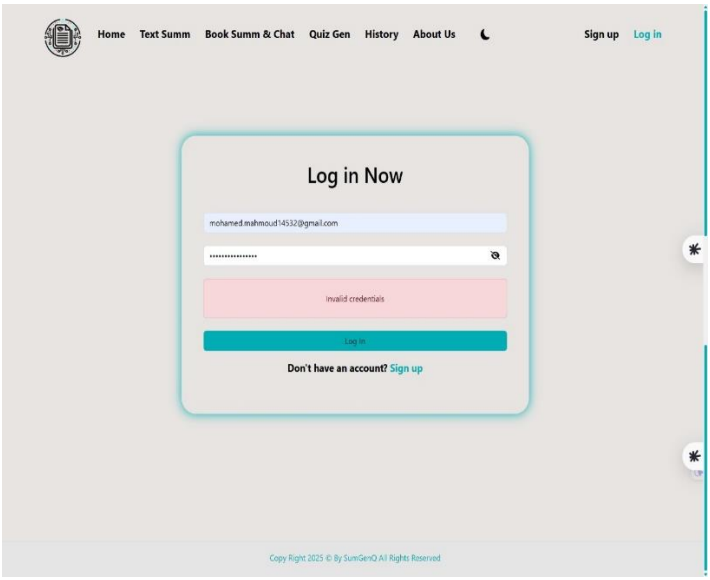
- **Check OTP Verification (3 Chances)**



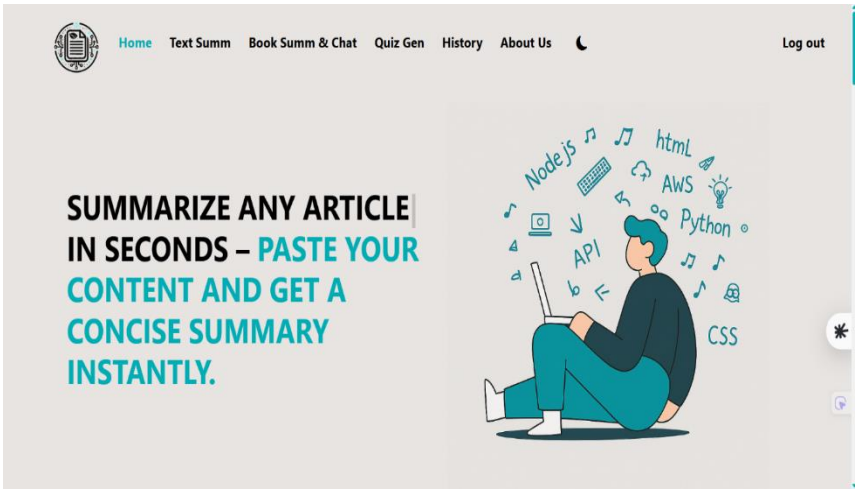
- **Login to your account**



- **Wrong account data**

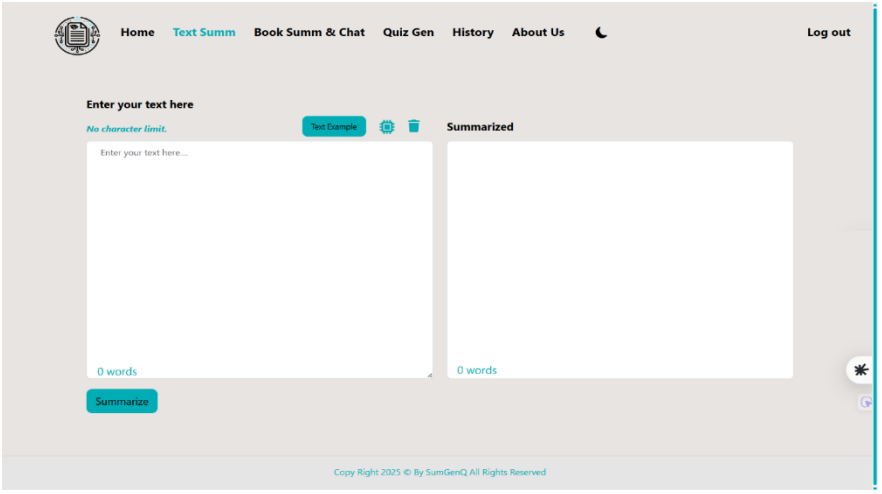


- **Right account data (Logged User Home)**

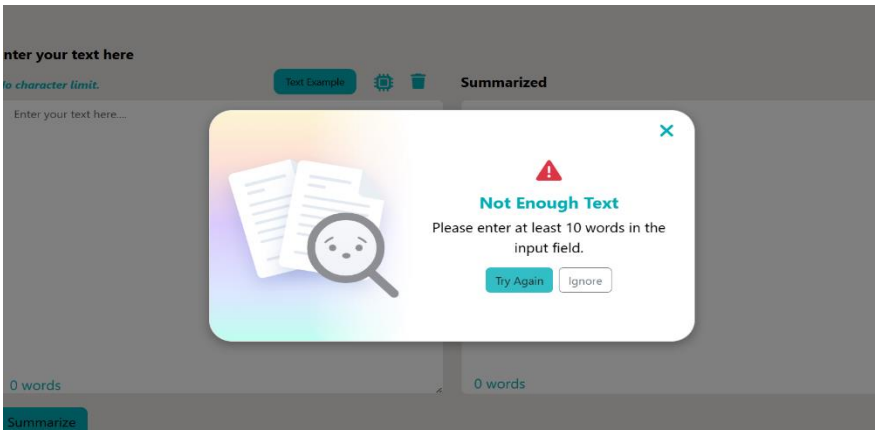


5.2 Text Summarization

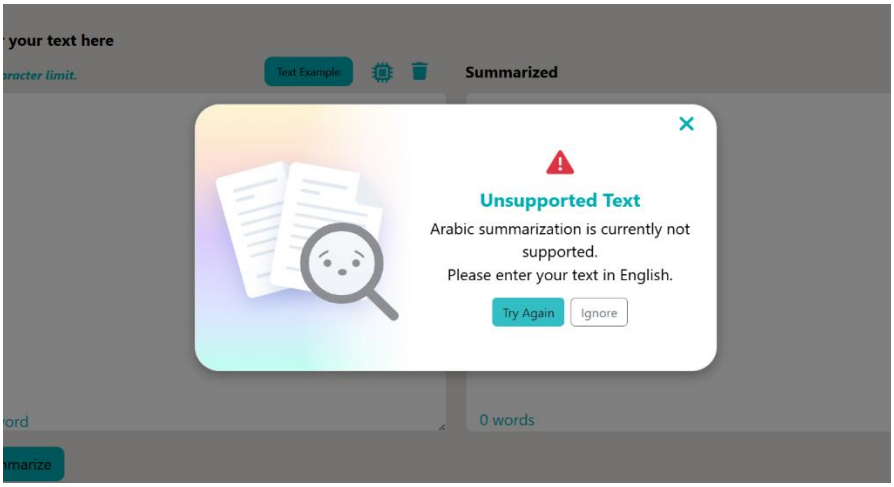
- **Main Summarizer**



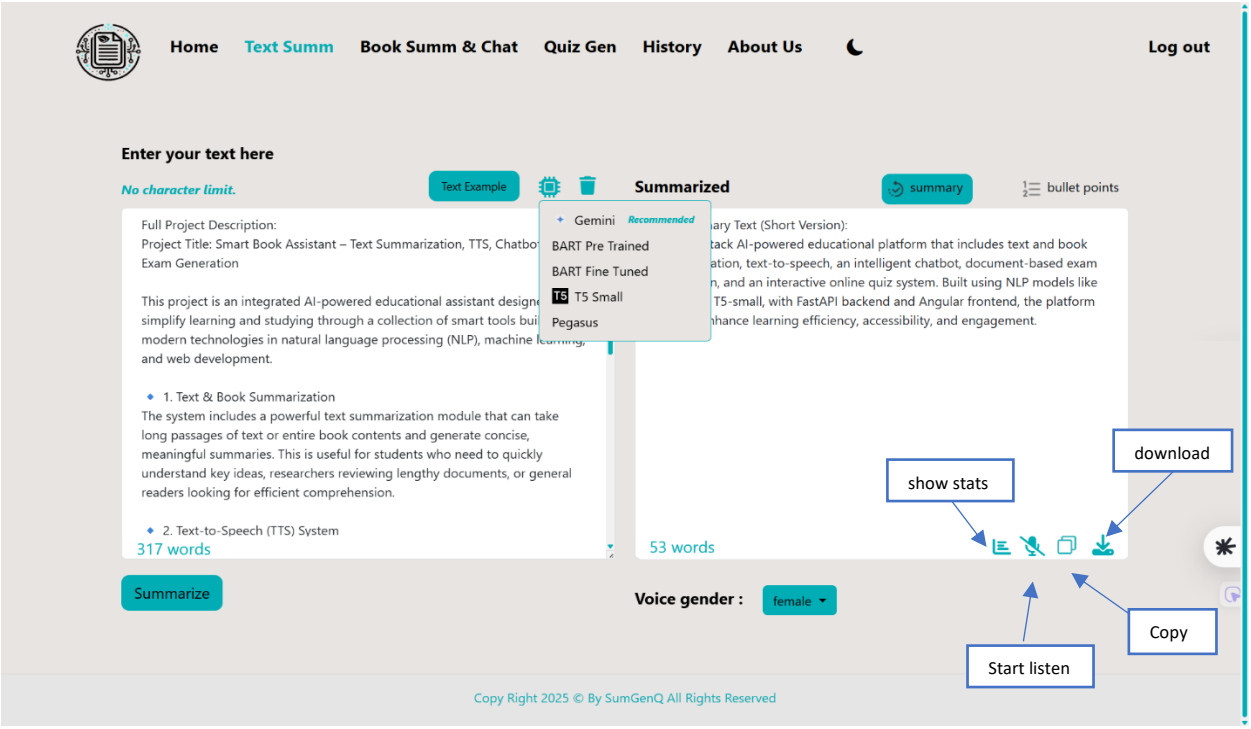
- **Short Text Error (Text Should > 10 Words)**



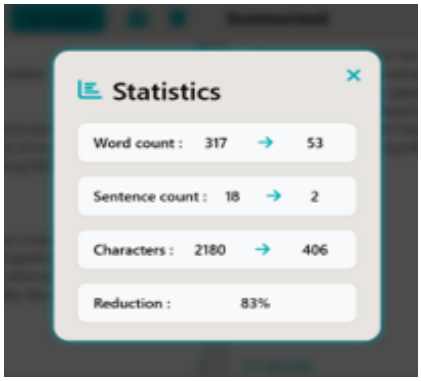
- **Unsupported Arabic Text**



- **Text Summarization**

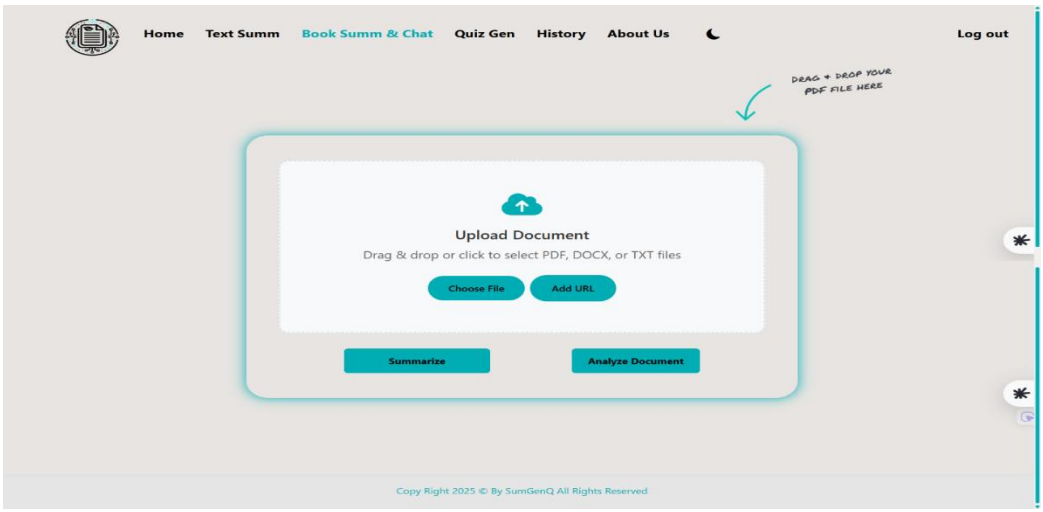


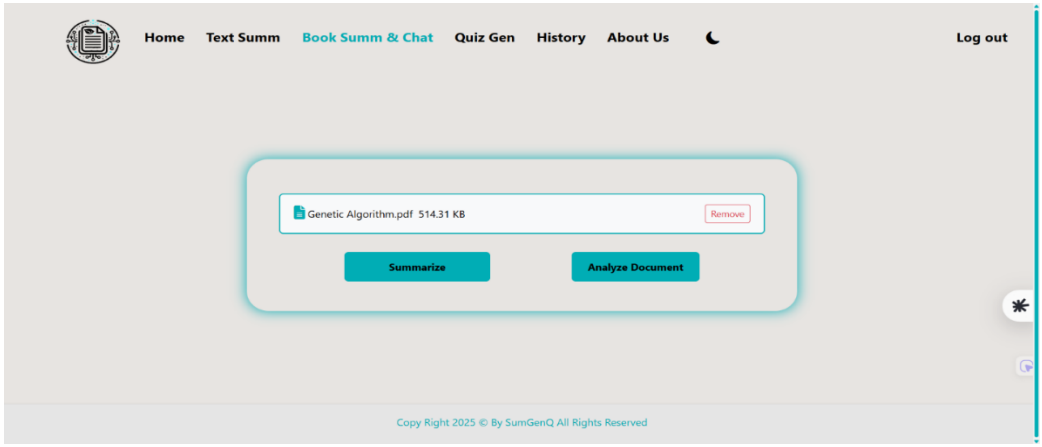
- **Text Stats**



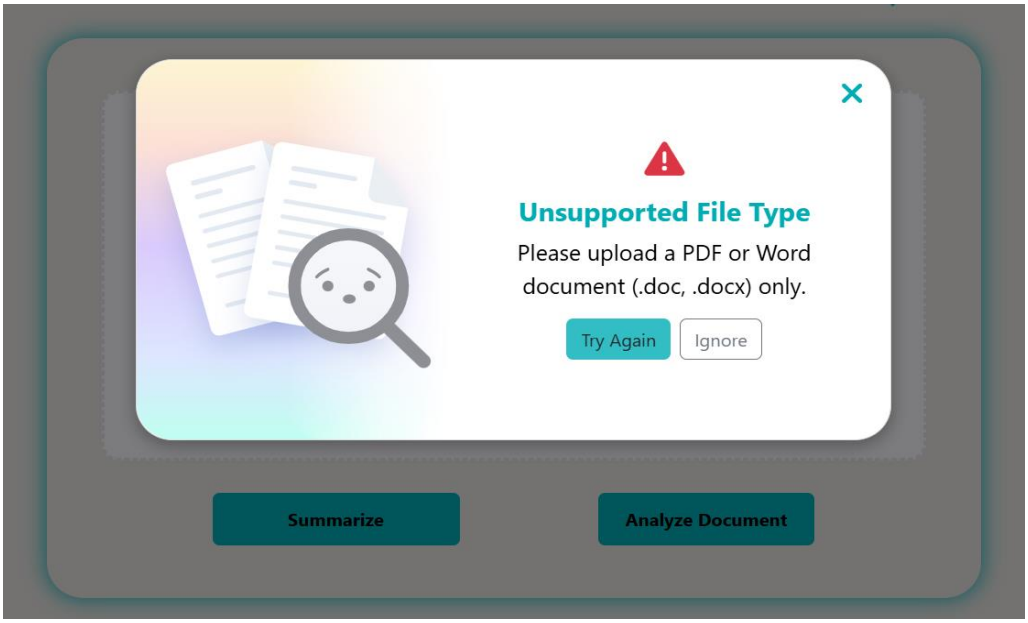
5.3 Document, URL Summarization & Chatbot

- **Upload Document or URL**

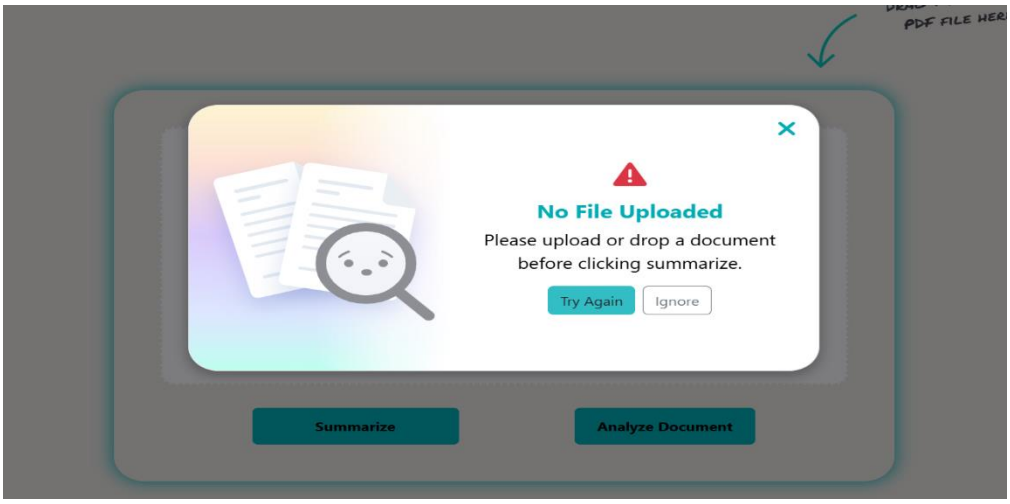




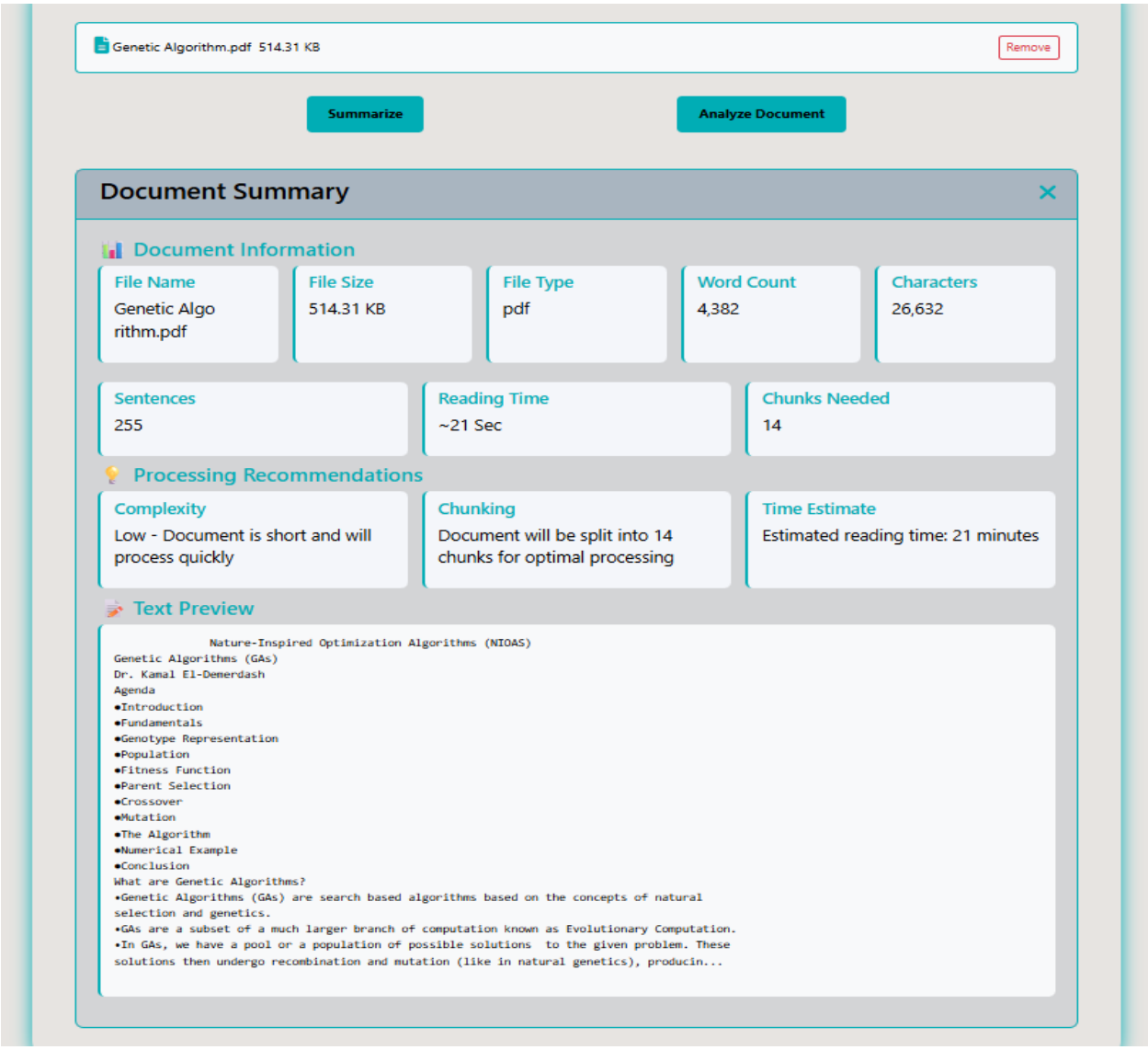
- **Failed file (file's type unsupported)**



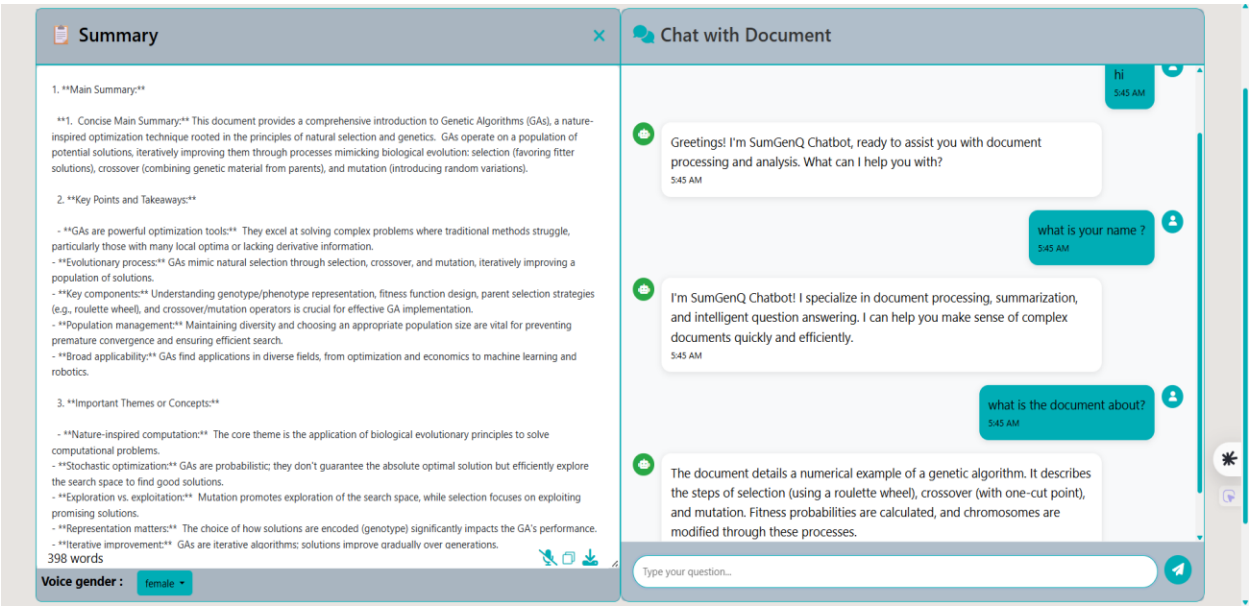
- **No File Uploaded**



- **Document Analysis**

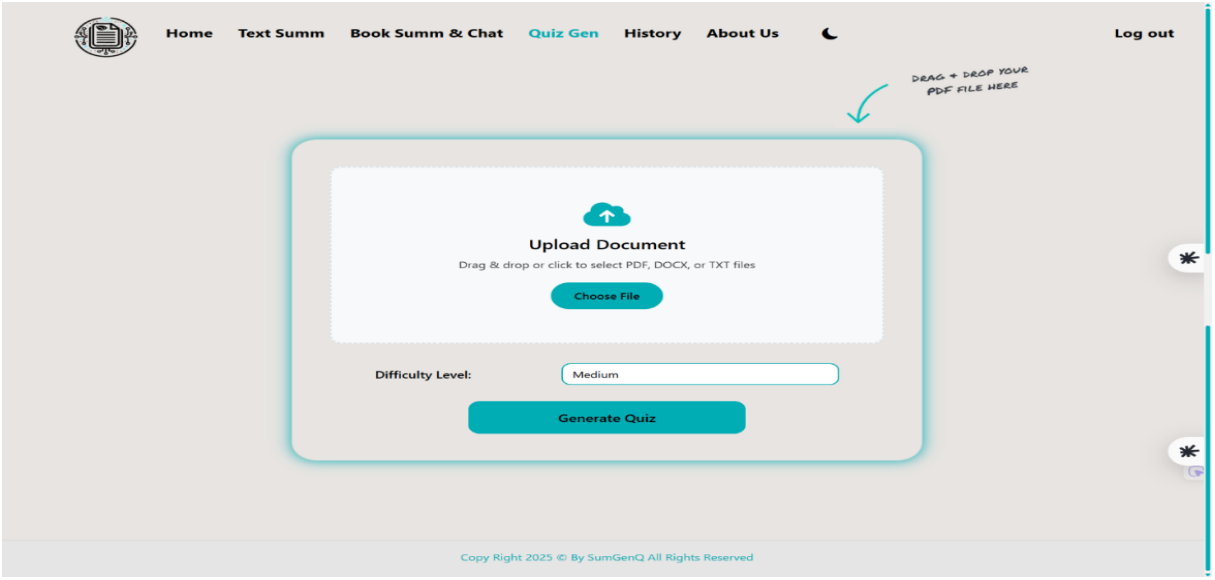


- **Summary & Chatbot**

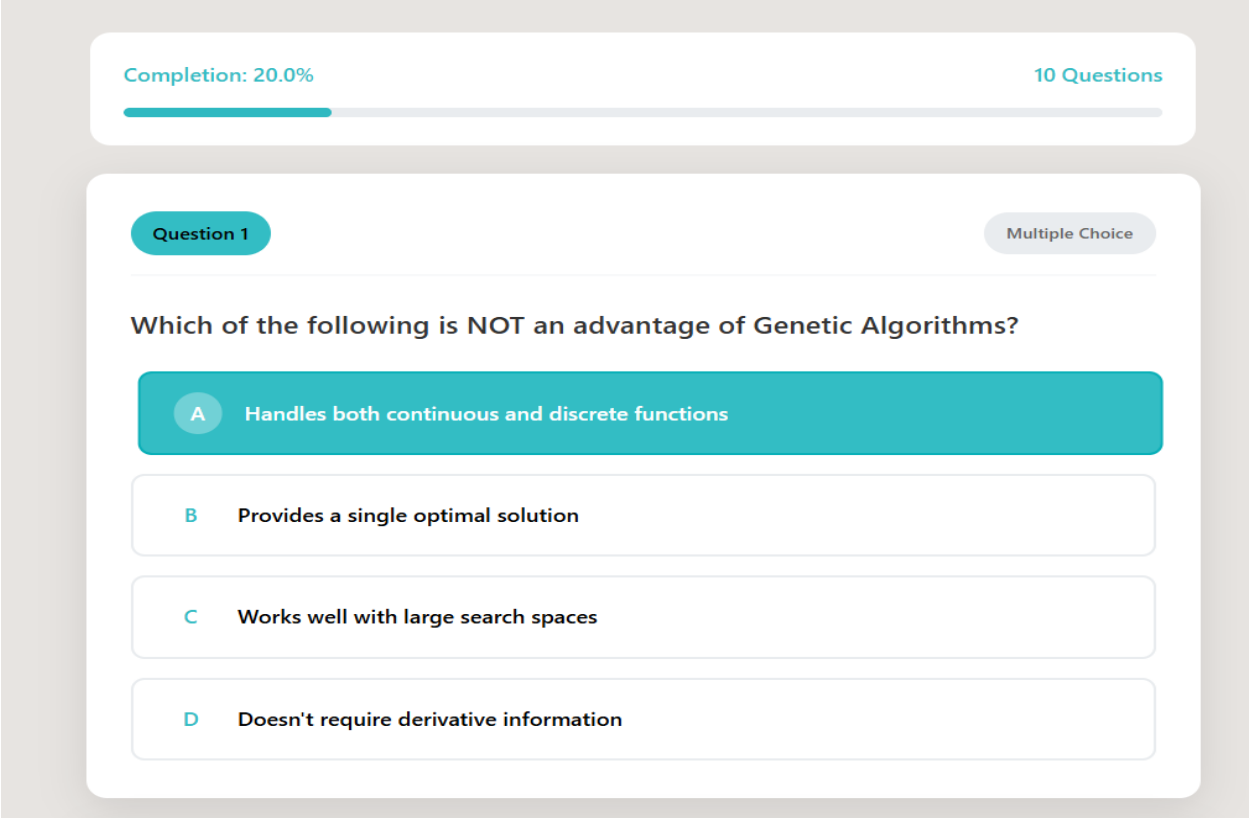


5.4 Quiz Generation

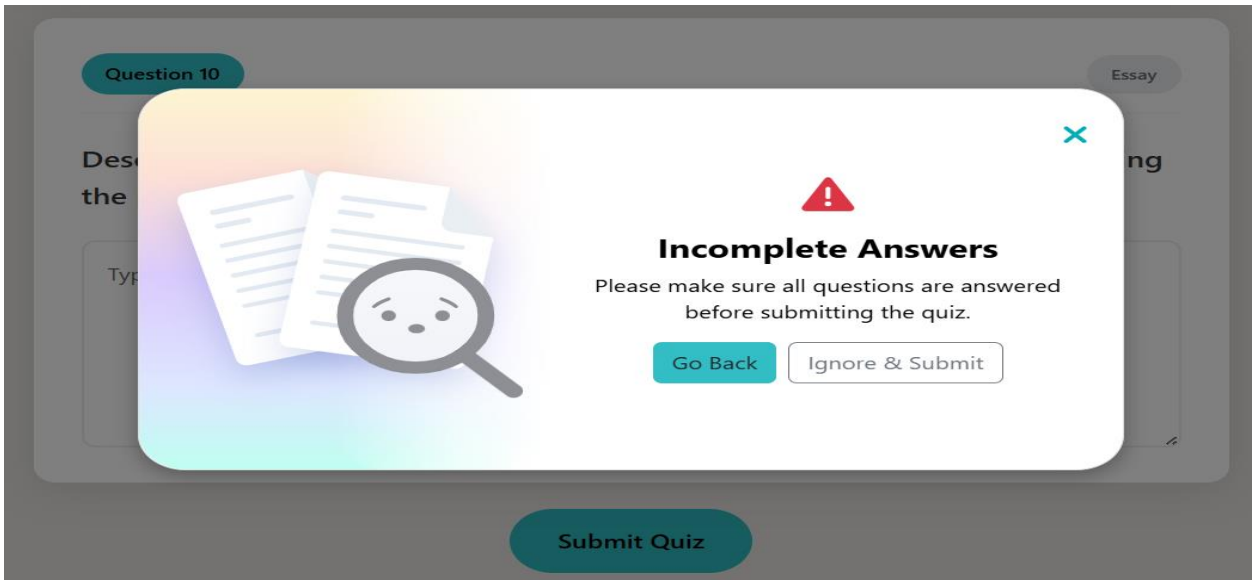
- **Upload File to Generate Quiz**



- **Start Quiz**



- **Incomplete Answers**



- **Quiz Results**

Quiz Results

23.3%
Final Score

Keep learning! Review the material.

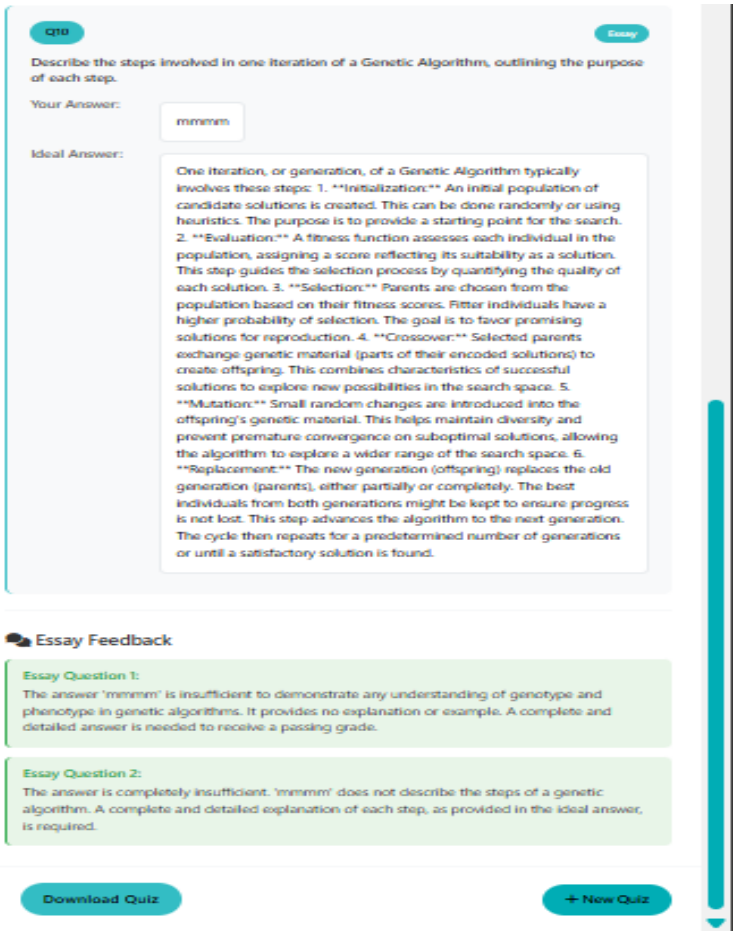
Question Review

Q1 MCQ

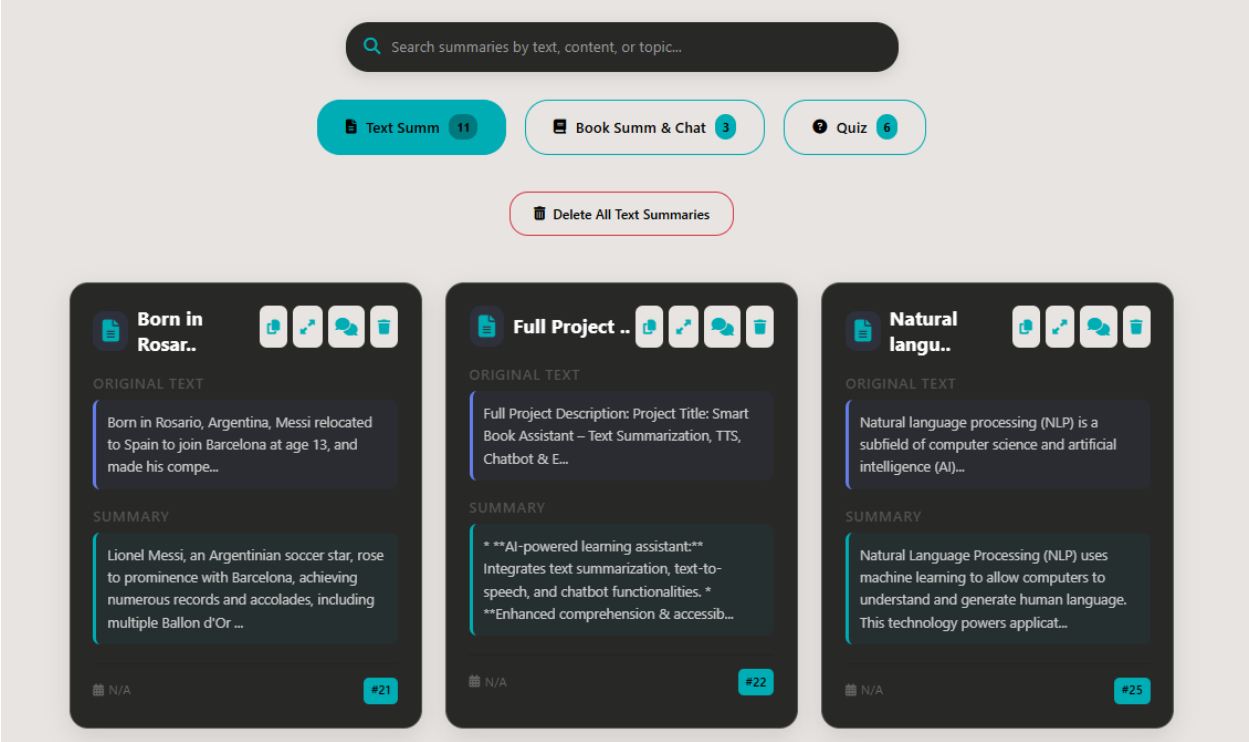
Which of the following is NOT an advantage of Genetic Algorithms?

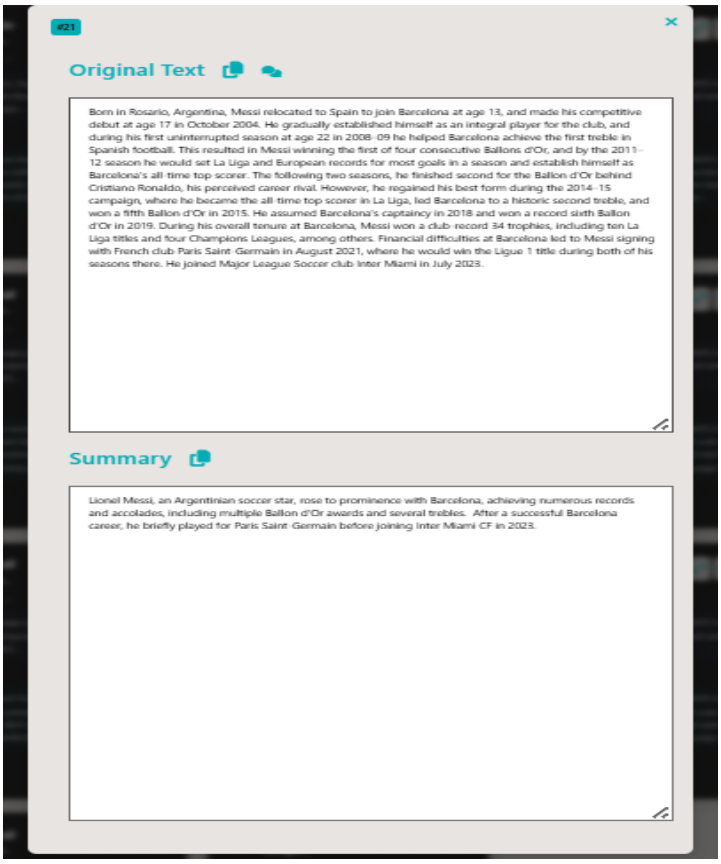
Your Answer: A. Handles both continuous and discrete functions

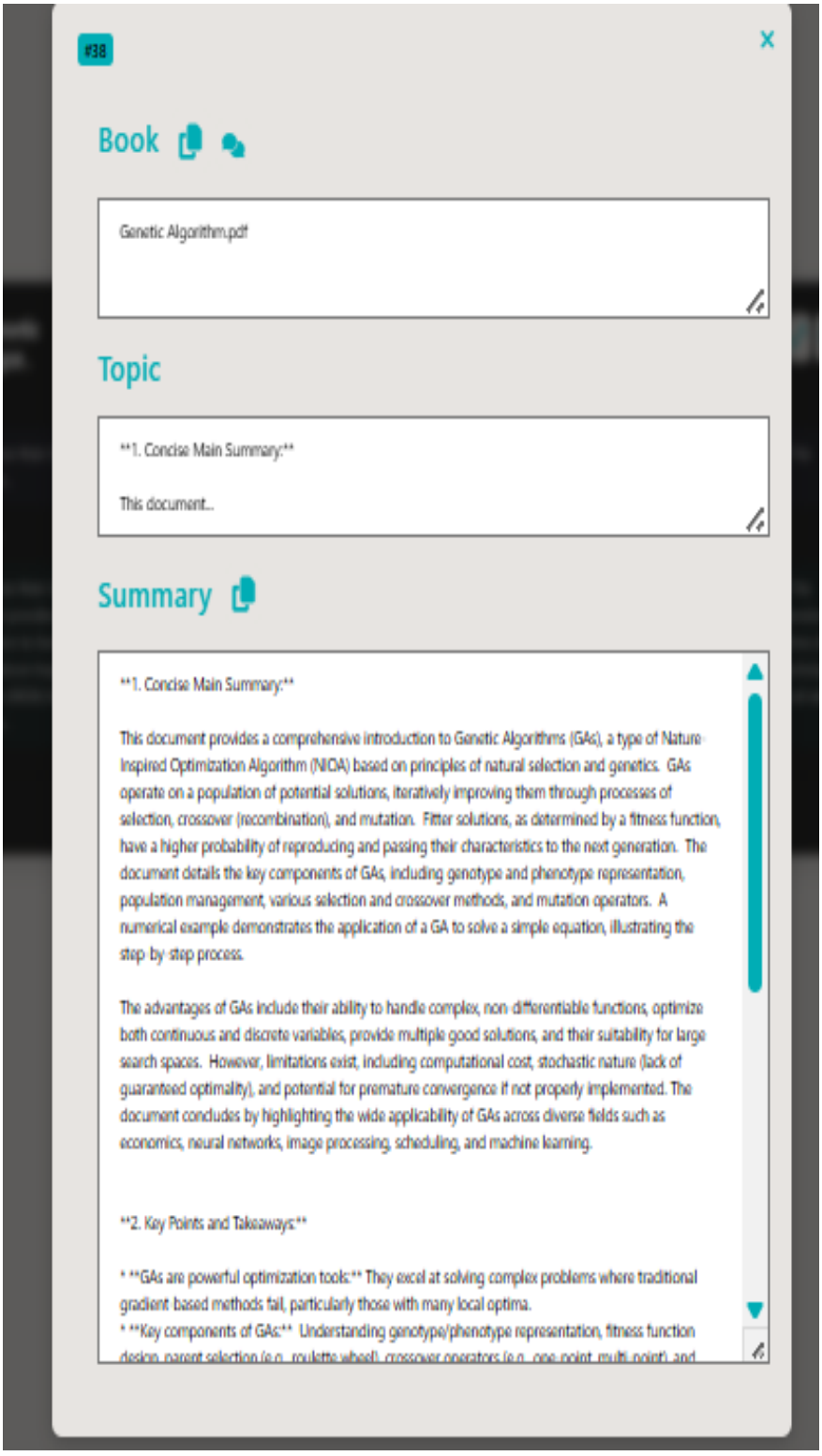
Correct Answer: B. Provides a single optimal solution

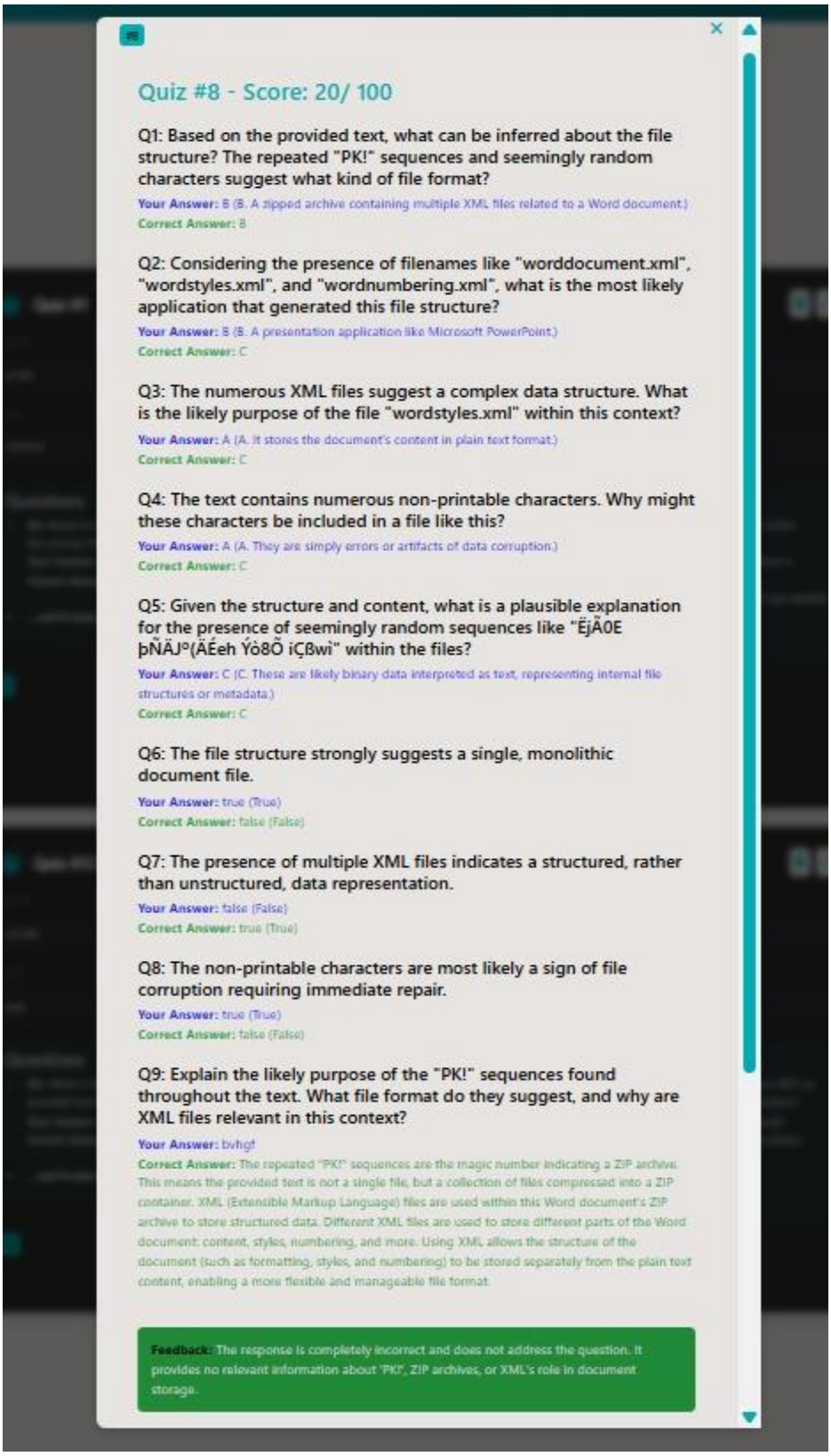


5.5 History


Search summaries by text, content, or topic...
Text Summ 11
Book Summ & Chat 3
Quiz 6
Delete All Text Summaries
Born in Rosar..
ORIGINAL TEXT
Born in Rosario, Argentina, Messi relocated to Spain to join Barcelona at age 13, and made his compe...
SUMMARY
Lionel Messi, an Argentinian soccer star, rose to prominence with Barcelona, achieving numerous records and accolades, including multiple Ballon d'Or ...
N/A
#21
Full Project ..
ORIGINAL TEXT
Full Project Description: Project Title: Smart Book Assistant – Text Summarization, TTS, Chatbot & E...
SUMMARY
* **AI-powered learning assistant:** Integrates text summarization, text-to-speech, and chatbot functionalities. * **Enhanced comprehension & accessib...
N/A
#22
Natural langu..
ORIGINAL TEXT
Natural language processing (NLP) is a subfield of computer science and artificial intelligence (AI)...
SUMMARY
Natural Language Processing (NLP) uses machine learning to allow computers to understand and generate human language. This technology powers applicat...
N/A
#25


#21
Original Text
Born in Rosario, Argentina, Messi relocated to Spain to join Barcelona at age 13, and made his competitive debut at age 17 in October 2004. He gradually established himself as an integral player for the club, and during his first uninterrupted season at age 22 in 2008–09 he helped Barcelona achieve the first treble in Spanish football. This resulted in Messi winning the first of four consecutive Ballons d'Or, and by the 2011–12 season he would set La Liga and European records for most goals in a season and establish himself as Barcelona's all-time top scorer. The following two seasons, he finished second for the Ballon d'Or behind Cristiano Ronaldo, his perceived career rival. However, he regained his best form during the 2014–15 campaign, where he became the all-time top scorer in La Liga, led Barcelona to a historic second treble, and won a fifth Ballon d'Or in 2015. He assumed Barcelona's captaincy in 2018 and won a record sixth Ballon d'Or in 2019. During his overall tenure at Barcelona, Messi won a club-record 34 trophies, including ten La Liga titles and four Champions Leagues, among others. Financial difficulties at Barcelona led to Messi signing with French club Paris Saint-Germain in August 2021, where he would win the Ligue 1 title during both of his seasons there. He joined Major League Soccer club Inter Miami in July 2023.
Summary
Lionel Messi, an Argentinian soccer star, rose to prominence with Barcelona, achieving numerous records and accolades, including multiple Ballon d'Or awards and several trebles. After a successful Barcelona career, he briefly played for Paris Saint-Germain before joining Inter Miami CF in 2023.


#38
Book
Genetic Algorithm.pdf
Topic
1. Concise Main Summary:
This document...
Summary
1. Concise Main Summary:
This document provides a comprehensive introduction to Genetic Algorithms (GAs), a type of Nature-Inspired Optimization Algorithm (NIOA) based on principles of natural selection and genetics. GAs operate on a population of potential solutions, iteratively improving them through processes of selection, crossover (recombination), and mutation. Fitter solutions, as determined by a fitness function, have a higher probability of reproducing and passing their characteristics to the next generation. The document details the key components of GAs, including genotype and phenotype representation, population management, various selection and crossover methods, and mutation operators. A numerical example demonstrates the application of a GA to solve a simple equation, illustrating the step-by-step process.
The advantages of GAs include their ability to handle complex, non-differentiable functions, optimize both continuous and discrete variables, provide multiple good solutions, and their suitability for large search spaces. However, limitations exist, including computational cost, stochastic nature (lack of guaranteed optimality), and potential for premature convergence if not properly implemented. The document concludes by highlighting the wide applicability of GAs across diverse fields such as economics, neural networks, image processing, scheduling, and machine learning.
2. Key Points and Takeaways:
* **GAs are powerful optimization tools:** They excel at solving complex problems where traditional gradient-based methods fail, particularly those with many local optima.
* **Key components of GAs:** Understanding genotype/phenotype representation, fitness function



Quiz #8 - Score: 20/ 100

Q1: Based on the provided text, what can be inferred about the file structure? The repeated "PK!" sequences and seemingly random characters suggest what kind of file format?

Your Answer: B (B. A zipped archive containing multiple XML files related to a Word document.)
Correct Answer: B

Q2: Considering the presence of filenames like "worddocument.xml", "wordstyles.xml", and "wordnumbering.xml", what is the most likely application that generated this file structure?

Your Answer: B (B. A presentation application like Microsoft PowerPoint.)
Correct Answer: C

Q3: The numerous XML files suggest a complex data structure. What is the likely purpose of the file "wordstyles.xml" within this context?

Your Answer: A (A. It stores the document's content in plain text format.)
Correct Answer: C

Q4: The text contains numerous non-printable characters. Why might these characters be included in a file like this?

Your Answer: A (A. They are simply errors or artifacts of data corruption.)
Correct Answer: C

Q5: Given the structure and content, what is a plausible explanation for the presence of seemingly random sequences like "ËjÃ0E þÑÄJº(ÁÉeh Ýò8Õ iÇßwi" within the files?

Your Answer: C (C. These are likely binary data interpreted as text, representing internal file structures or metadata.)
Correct Answer: C

Q6: The file structure strongly suggests a single, monolithic document file.

Your Answer: true (True)
Correct Answer: false (False)

Q7: The presence of multiple XML files indicates a structured, rather than unstructured, data representation.

Your Answer: false (False)
Correct Answer: true (True)

Q8: The non-printable characters are most likely a sign of file corruption requiring immediate repair.

Your Answer: true (True)
Correct Answer: false (False)

Q9: Explain the likely purpose of the "PK!" sequences found throughout the text. What file format do they suggest, and why are XML files relevant in this context?

Your Answer: bvhgf
Correct Answer: The repeated "PK!" sequences are the magic number indicating a ZIP archive. This means the provided text is not a single file, but a collection of files compressed into a ZIP container. XML (Extensible Markup Language) files are used within this Word document's ZIP archive to store structured data. Different XML files are used to store different parts of the Word document: content, styles, numbering, and more. Using XML allows the structure of the document (such as formatting, styles, and numbering) to be stored separately from the plain text content, enabling a more flexible and manageable file format.

Feedback: The response is completely incorrect and does not address the question. It provides no relevant information about 'PK!', ZIP archives, or XML's role in document storage.

6. Future Work

This To extend the capabilities of our graduation project and enhance its usability, we propose several advanced features for future implementation. These enhancements aim to make the system more interactive, inclusive, and capable of handling diverse input modalities.

- Interactive Voice Chatbot

  We plan to integrate a **voice-based conversational agent** that enables users to communicate with the system through natural spoken language. This interactive voice chatbot will be powered by **Automatic Speech Recognition (ASR)** and **Text-to-Speech (TTS)** technologies, creating a fluid, hands-free interaction environment. By leveraging models such as **OpenAI Whisper for ASR** and **Google TTS or similar neural TTS models**, the system will convert user speech into text, process it using the chatbot's NLP engine, and respond with synthesized speech.

  This addition is particularly useful in accessibility contexts and provides a more human-like experience, especially for users with reading or motor challenges. The chatbot will be tightly integrated with the existing text-based backend, ensuring consistent behavior and reliable response generation across both voice and text inputs.

- Optical Character Recognition (OCR)

  To support document-based and image-based inputs, we also aim to implement **Optical Character Recognition (OCR)** functionality. This feature will allow users to upload scanned documents or photos containing text, which the system will process using OCR engines such as **Tesseract OCR** or more advanced deep learning-based models. The extracted English text will then be fed into the summarization or question-answering modules, making the system capable of handling a broader range of input formats.

  This is particularly valuable in educational and professional domains where users often deal with printed documents or handwritten notes.

- Language Support Scope

  To ensure optimal performance and reduce computational complexity in early development stages, the extended features will **support English language only**. This constraint allows us to optimize voice recognition, NLP understanding, and OCR pipelines without the overhead of multilingual data processing. In the future, support for additional languages, especially Arabic, can be considered as a natural progression.

7. Conclusion

This project successfully designed and implemented an intelligent system aimed at simplifying user interaction with extensive textual content. By seamlessly integrating automatic text summarization, an interactive chatbot, quiz generation, and text-to-speech (TTS) capabilities, the platform emerges as a comprehensive, educational, and accessible tool tailored for learners, educators, and individuals with special needs. The solution effectively mitigates the challenge of information overload by distilling lengthy texts into concise, impactful summaries, enabling users to quickly assimilate key concepts. The chatbot fosters engagement through dynamic Q&A sessions, while the quiz generation feature strengthens knowledge retention. Additionally, the audio output functionality enhances inclusivity, catering to users with visual impairments or reading difficulties.

Throughout the development process, several challenges were adeptly addressed, including the fine-tuning of natural language processing (NLP) models, ensuring the contextual accuracy of chatbot responses, and generating relevant, high-quality quiz questions. These hurdles were overcome by employing cutting-edge machine learning techniques and harnessing the power of robust open-source tools and datasets. In summary, this system exemplifies the transformative potential of integrating diverse NLP capabilities into a cohesive application, significantly enhancing learning experiences and accessibility. Beyond its immediate educational applications, this project establishes a solid foundation for future advancements in intelligent content delivery.

8. References

- **Datasets**:
  - **CNN/DailyMail Dataset**: Available at https://huggingface.co/datasets/abisee/cnn_dailymail
  - **BookSum Complete Cleaned Dataset**: Available at https://huggingface.co/datasets/ubaada/booksum-complete-cleaned
  - **WikiSum Dataset**: Available at https://huggingface.co/datasets/d0rj/wikisum
  - **XSum Dataset**: Available at https://huggingface.co/datasets/EdinburghNLP/xsum
  - **Project Gutenberg Dataset**: Available at https://www.gutenberg.org/
  - **FINDSum Dataset**: Referenced for financial document summarization, no specific URL provided.
  - **Preprocessed CNN/DailyMail Dataset**: Available at https://huggingface.co/datasets/mzizo4110/cnn-dailymail-final-preprocessed
- **Models**:
  - **BART-Large-CNN**: Available at https://huggingface.co/facebook/bart-large-cnn
  - **BART-Large-CNN with LoRA Fine-Tuning**: Available at https://huggingface.co/mzizo4110/BART-SUMMARIZATION-5
  - **T5-Small with LoRA Fine-Tuning**: Available at https://huggingface.co/mzizo4110/Summarization_Continue
  - **Pegasus-CNN**: Available at https://huggingface.co/google/pegasus-cnn_dailymail
  - **Gemini API**: Proprietary model, no specific public URL provided.
  - **Longformer, BigBird, Sentence-T5, TextRank, Llama 2, FiD, RAG, GPT-3, GPT-2, BiDAF**: Referenced in related works, no specific URLs provided.
- **Tools and Technologies**:
  - **Hugging Face Transformers**: https://huggingface.co/
  - **LangChain**: Framework for RAG pipeline, no specific URL provided.
  - **Chroma Vector Database**: Used for vector storage, no specific URL provided.

- **pyttsx3**: Python library for text-to-speech, no specific URL provided.
- **FastAPI**: https://fastapi.tiangolo.com/
- **BeautifulSoup**: https://www.crummy.com/software/BeautifulSoup/
- **SpaCy**: https://spacy.io/
- **PostgreSQL**: https://www.postgresql.org/
- **SQLAlchemy**: https://www.sqlalchemy.org/
- **Angular**: https://angular.io/
- **Bootstrap 5**: https://getbootstrap.com/
- **Vercel**: https://vercel.com/
- **GitHub**: https://github.com/

- **Related Works**:
  - **ACLSUM**: Referenced for scientific summarization, no specific paper or URL provided.
  - **QReCC**: Benchmark dataset for conversational QA, no specific URL provided.
  - **QG-NET**: Neural network for quiz generation, no specific URL provided.
  - **Competitor Websites**:
    - **QuillBot**: https://quillbot.com/
    - **ChatPDF**: https://www.chatpdf.com/
    - **TLDR This**: https://tldrthis.com/