A Project Report On

**Text Summarizer**

Submitted in partial fulfillment of the requirement for the award of the degree

Bachelor of Computer Application (BCA)/ Bachelor of Science (IT)

Academic Year 2023 – 24

**Student-1 – Rohit Luni**

**Enr. No - 92100588045**

| **Internal Guide** |
| --- |
| Vinod Pal |



Rajkot-Morbi Road, At & PO : Gauridad, Rajkot 360 003. Gujarat. India.



**Faculty of Computer Applications (FCA)**

****

**This is to certify that the project work entitled**

**Text Summarizer**

**submitted in partial fulfillment of the requirement for**

**the award of the degree of**

**Bachelor of Computer Application/ Bachelor of Science (IT)**

**of the**

**Marwadi University**

**is a result of the bonafide work carried out by**

**Rohit Luni - 92100588045**

**during the academic year 2023-24.**

| **Faculty Guide** |  | **HOD** |  | **Dean** |
| --- | --- | --- | --- | --- |

**DECLARATION**

I/Wehereby declare that this project work entitled **Text Summarizer** is a record done by me.

I also declare that the matter embodied in this project is genuine work done by me and has not been submitted whether to this University or to any other University / Institute for the fulfillment of the requirement of any course of study.

Place :

Date :

**Rohit Luni – 92100588045 Signature :**

**CONTENTS**

| **Chapters** | **Particulars** | **Page No.** |
| --- | --- | --- |
| **1** | **SYNOPSIS** |  |
| **2**  2.1  2.2 | **PREAMBLE**  General Introduction  Module description |  |
| **3** | **REVIEW OF LITERATURE** |  |
| **4**  4.1  4.2 | **TECHNICAL DESCRIPTION**  Hardware Requirement  Software Requirement |  |
| **5**  5.1  5.2  5.2.1  5.3  5.3.1  5.3.2  5.3.3  5.4  5.4.1  5.4.2 | **SYSTEM DESIGN AND DEVELOPMENT**  **(Only applicable diagrams)**  **Architectural Design**   * Class Diagram   **Dynamic Modeling**   * Use Case Diagram * Sequence Diagram * Activity Diagram * Any other applicable diagram (applicable)   Database Design (If applicable)  Relationship Diagram (ER)  Menu Design  Screen Design |  |
| **6** | **TESTING** |  |
| **7** | **CONCLUSION** |  |
| **8** | **LEARNING DURING PROJECT WORK** |  |
| **9** | **BIBLIOGRAPHY** |  |

**1.SYNOPSIS**

A text summarizer is a software tool or system that condenses a given piece of text, whether it's a document, article, or any written content, into a shorter version while preserving its essential information and meaning. The purpose of text summarization is to make content more concise and accessible, making it easier for readers to grasp the main points without reading the entire text.

There are generally two types of text summarization:

1. **Abstractive Summarization**: In abstractive summarization, the system generates a summary that may not be present in the original text. It interprets and paraphrases the content to create a concise and coherent summary. This method requires a deeper understanding of language and context.

2. **Extractive Summarization**: Extractive summarization, on the other hand, selects sentences or phrases directly from the original text to construct the summary. It doesn't create new sentences but extracts the most important parts based on predefined criteria like sentence importance or relevance.

Text summarizers find applications in various fields, including journalism, content curation, and data analysis. They help users quickly understand the core message of lengthy texts, making it easier to process large volumes of information.

Text summarization is often implemented using natural language processing (NLP) techniques and machine learning algorithms. These algorithms analyze the content, identify key sentences or concepts, and arrange them into a coherent and concise summary. The choice between abstractive and extractive summarization depends on the specific use case and the desired level of detail in the summary.

**2.PREAMBLE**

**2.1.General Introduction**

**Project Overview:**

The "Text Summarizer Using Chat-GPT Turbo 3.5" project is a robust and efficient text summarization application that leverages a combination of powerful technologies. At its core, this project utilizes Django, a Python-based web framework, to create a versatile backend API. This API, enriched by the Chat-GPT Turbo 3.5 model from OpenAI, offers the capability to summarize extensive textual content with ease. The frontend, developed using Next.js and TypeScript, adds an intuitive layer to the application, enabling users to interact seamlessly with the summarization feature.

**Backend API:**

The backend is constructed using Django and Python, which facilitates the development of robust web applications. Python is instrumental in connecting the application to OpenAI's Chat-GPT Turbo 3.5 model, responsible for the remarkable text summarization capabilities. The API exposes a single endpoint, 'api/summarize/', designed to receive JSON payloads containing text that requires summarization. Python, in combination with the Django framework, efficiently processes these requests and responds with a JSON-formatted summary of the input text. This backend is not only powerful but also highly extensible, making it an ideal choice for this project.

**Frontend Interface:**

The user-facing frontend is built using Next.js, a popular and versatile React framework, and TypeScript, which enhances code maintainability and development speed. This interface is designed to be user-friendly, allowing users to input text and view summarized content in a user-friendly format. The frontend seamlessly communicates with the Django backend, providing a complete end-user experience. It is the bridge between the user and the powerful text summarization capabilities of Chat-GPT Turbo 3.5, making it accessible to a wide range of users.

**Libraries and Tools:**

This project relies on several key libraries and tools, including Django, Python, OpenAI's Chat-GPT Turbo 3.5 model, Next.js, and TypeScript. Django and Python are the foundation of the backend API, while Chat-GPT Turbo 3.5 is the driving force behind the text summarization process. Next.js and TypeScript enhance the frontend, offering a responsive and user-friendly interface. These libraries and tools, carefully selected for their strengths, ensure that the project delivers a robust and efficient text summarization solution with a seamless user experience.

**Impact and Utility:**

The "Text Summarizer Using Chat-GPT Turbo 3.5" project represents a significant advancement in the field of natural language processing and content summarization. By harnessing the power of Chat-GPT Turbo 3.5, it offers a versatile tool for extracting key insights from extensive textual content. Whether applied in research, content curation, or information digestion, this project has the potential to improve productivity and knowledge retrieval across various domains, thanks to its carefully chosen libraries and tools.

**2.2.Module description**

1. **Django Backend (Backend):**

- **Description**: The Django Backend module serves as the core of the project, responsible for handling HTTP requests, processing text inputs, and communicating with the Chat-GPT Turbo 3.5 model for text summarization.

- **Key Components**:

-**API Endpoints**: The primary endpoint, `/api/summarize/`, receives JSON payloads containing text for summarization.

-**Views and Controllers:** Utilizes Django views and controllers to handle incoming requests and send responses.

- **Integration with Chat-GPT Turbo 3.5:** Integrates with the Chat-GPT Turbo 3.5 model to perform text summarization.

- **JSON Responses:** Provides JSON-formatted summaries in response to incoming requests.

2. **Chat-GPT Turbo 3.5 Integration (Text Summarizer):**

- **Description:** This module is responsible for integrating the Chat-GPT Turbo 3.5 model from OpenAI, which performs the actual text summarization.

- **Key Components:**

- **Chat-GPT Turbo 3.5 Model:** Utilizes OpenAI's powerful language model to generate concise summaries from input text.

- **Text Preprocessing**: Preprocesses input text to ensure optimal summarization results.

- **Text Summarization Logic:** Contains the logic for interacting with the model and generating summaries.

3. **Next.js Frontend (Frontend):**

- **Description:** The Next.js Frontend module focuses on providing a user-friendly interface for inputting text and displaying summarization results.

- **Key Components:**

- **User Interface:** Presents a web interface for users to interact with the application.

- **Input Forms:** Offers forms for users to input text for summarization.

- **Communication with Backend:** Communicates with the Django backend through HTTP requests to send text for summarization and display the resulting summaries.

- **Display of Summaries:** Renders the summarization results in a readable format for users.

4. **Libraries and Tools (Dependencies):**

- **Description:** This module lists the libraries and tools that are essential for the project's functionality.

- **Key Components:**

- **Django:** The Python web framework that powers the backend API.

- **Python:** The programming language used for backend development and integration.

- **Chat-GPT Turbo 3.5:** OpenAI's language model responsible for text summarization.

- **Next.js:** The React framework used to create the frontend interface.

- **TypeScript:** Enhances code maintainability in the frontend.

This module description provides an overview of the project's key components, their roles, and their interactions. It's a valuable resource for understanding the structure and functionality of your "Text Summarizer Using Chat-GPT Turbo 3.5" project.

**3.REVIEW OF LITERATURE**

The development of text summarization tools has been a significant area of interest in the field of natural language processing and artificial intelligence. Researchers and developers have long sought efficient ways to condense lengthy texts, making information retrieval and content curation more manageable. In this context, OpenAI's Chat-GPT Turbo 3.5 model has emerged as a game-changer. It represents a culmination of advancements in language models, combining the capabilities of GPT-3.5 with a chat-based interface. Chat-GPT Turbo 3.5's ability to comprehend and generate coherent human-like text makes it a natural fit for text summarization tasks. Its integration in this project reflects the state-of-the-art in AI-driven summarization technology.

Django, a popular Python web framework, forms the backbone of the backend in this project. Django's robust and flexible design simplifies the creation of APIs, enabling the efficient handling of HTTP requests. It serves as a testament to the power of Python in web development and demonstrates how versatile web frameworks can be leveraged for AI-driven applications. The backend's integration with the Chat-GPT Turbo 3.5 model showcases the potential of combining mature web development tools with cutting-edge AI technology to deliver practical solutions.

On the frontend, Next.js and TypeScript make for a compelling combination. Next.js simplifies React application development with server-side rendering, enhancing the user experience. TypeScript, known for its strong typing and code maintainability, ensures the frontend remains robust and reliable. Together, they form the foundation for a user-friendly interface that complements the backend's capabilities. This literature review underscores the convergence of mature and innovative technologies in the "Text Summarizer Using Chat-GPT Turbo 3.5" project, marking it as a noteworthy advancement in the field of text summarization and AI-driven applications.

The project's blend of Chat-GPT Turbo 3.5, Django, Python, Next.js, and TypeScript illustrates the evolution of AI-driven text summarization, where the synergy of powerful AI models and reliable web development tools empowers users with an efficient and accessible solution for content summarization. It encapsulates the ongoing efforts to bridge the gap between sophisticated AI technology and real-world applications, offering the potential to revolutionize information processing and knowledge retrieval across diverse domains.

**4.TECHNICAL DESCRIPTION**

**4.1 Hardware Requirement**

* Any Operating System
* At least 1 GB of RAM
* At least 100 MB of free disk space

**4.2 Software Requirement**

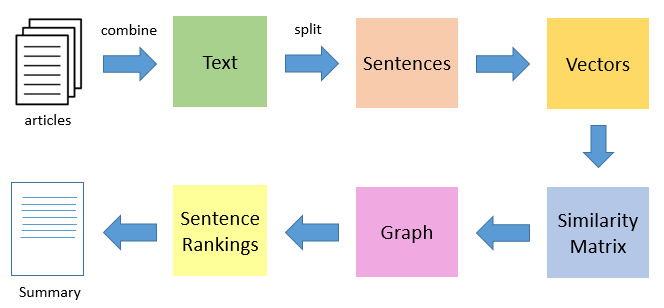
* Visual Studio Code (1.81)
* Python 3.11.4
* Django 4.2.5
* Import openai library
* os

**5.SYSTEM DESIGN**

**AND DEVELOPMENT**

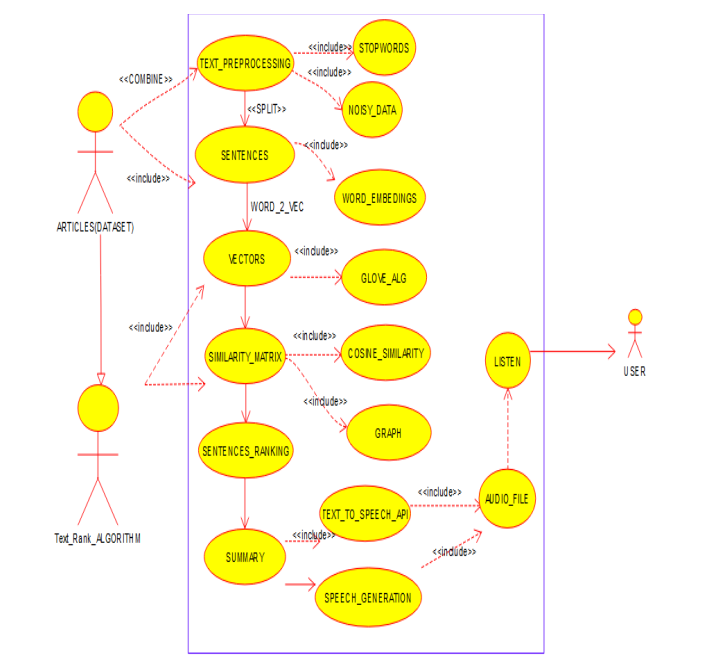
**5.1 Architectural Design**

* **Flow Chart**

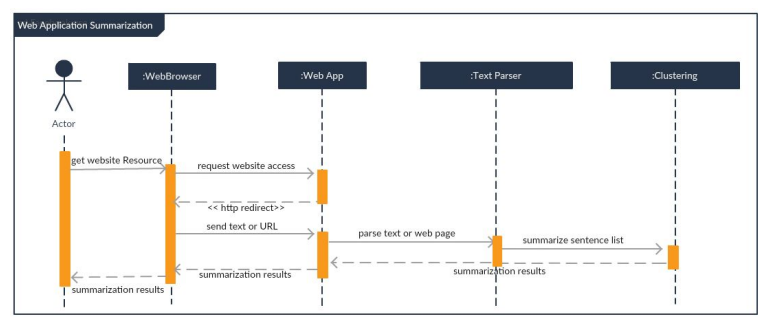


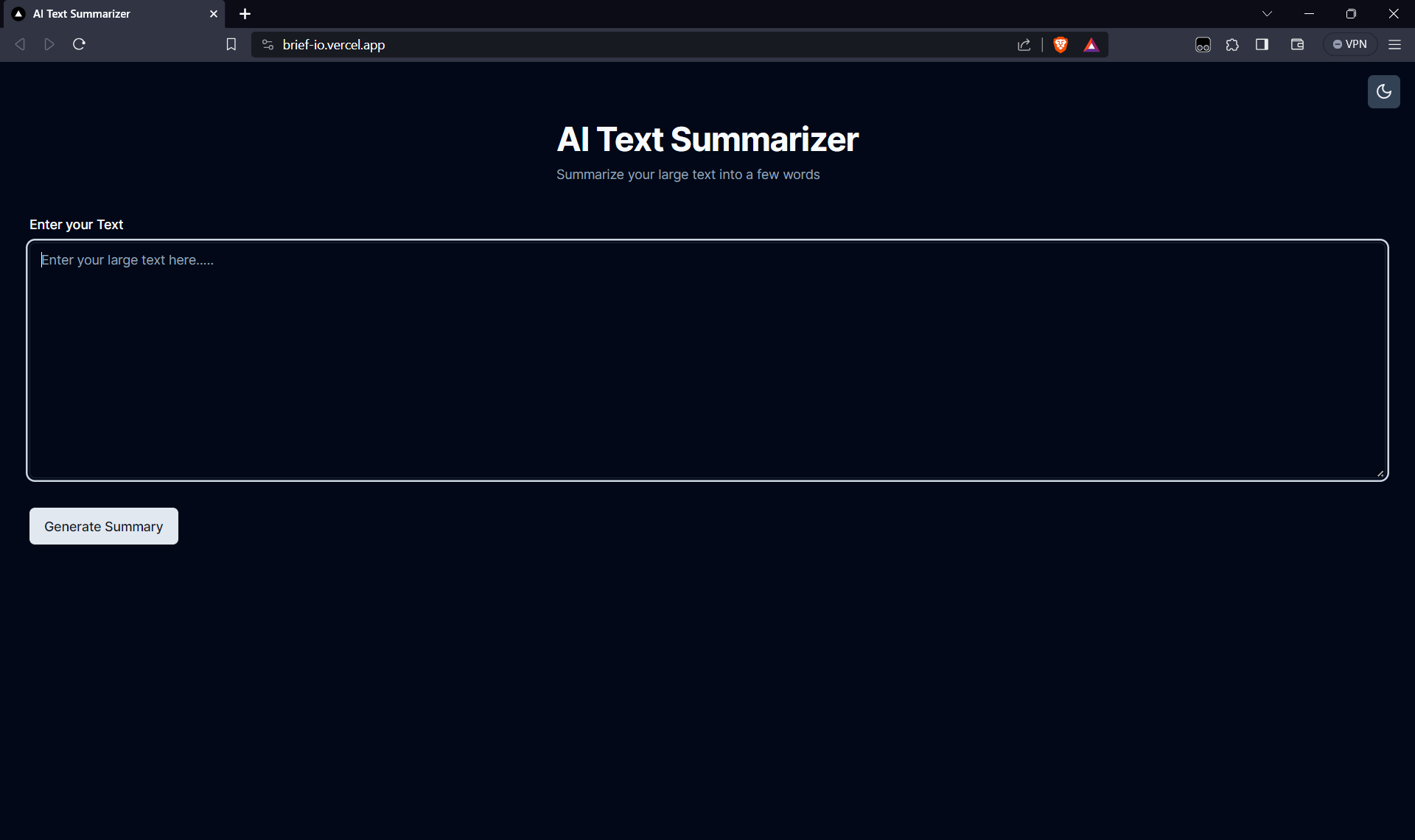
**5.2 Dynamic Modeling**

* **Use case Diagram**



* **Sequence Diagram**



**6. TESTING**

**7. CONCLUSION**

* This project has implemented a simple text summarizer in Python. The text summarizer uses the summarize our large text into the small and few lines of text. The project also shows how to use the ChatGPT turbo 3.5 Api to summarize our text into small text with Developed a front-end site for the project to make it easier to use.

**8.LEARNING DURING PROJECT WORK**

During this project we learn following topic

1. How to use the Python openai library to send requests and get back to the result.
2. How to send a prompt to ChatGPT for a batter result.
3. How to use Django for creating our backend apis.
4. How to use the front end react to create GUI and show the result.
5. How to use render to deploy backend code.
6. How to use vercel to deploy frontend code.
7. How to test and debug your code.

In General We learn following topic

1.**Text Summarization**: The core of your project revolves around text summarization, where you'll learn techniques to condense lengthy text into shorter, more concise versions.

2.**Python Programming**: This project provides hands-on experience in Python programming. You'll work with Python libraries and APIs for text processing and summarization.

3.**OpenAI's GPT-3.5 Turbo API**: Leveraging the power of OpenAI's GPT-3.5 Turbo API, you'll implement advanced text summarization techniques for efficient content condensation."

This summary highlights the central aspects of your project, emphasizing text summarization, Python programming, and the integration of the OpenAI GPT-3.5 Turbo API.

**9.BIBLIOGRAPHY**

* "ChatGPT turbo 3.5 Api Documentation”

<https://openai.com/blog/gpt-3-5-turbo-fine-tuning-and-api-updates>

* "NextJS Documentation"

<https://vercel.com/solutions/nextjs>

* "Django Documentation"

<https://www.djangoproject.com/>