Text Summarization

---------------------------------------------------------------------------------------------------------------------------------------------------

This Project aim to develop an automated test summarization system. It is a process which consumes large amount of data from articles, meeting notes, documents and websites and provide summery while maintaining the key meaning of original text.

# Objectives

* To design and implement a REST API for text summarization
* To provide fast, scalable and secure API endpoints using FASTAPI.
* To integrate the Gemini model for high-quality test summarization

# Technology Stack

|  |  |
| --- | --- |
| Programming Language | Python 3.11+ |
| Framework | FastAPI |
| Model | Google Gemini (via Vertex AI API / Gemini API) |
| Web Server | Uvicorn (ASGI server) |

# Features

• Accepts any large block of text and generates a concise summary.

• RESTful API design for easy integration with web or mobile apps.

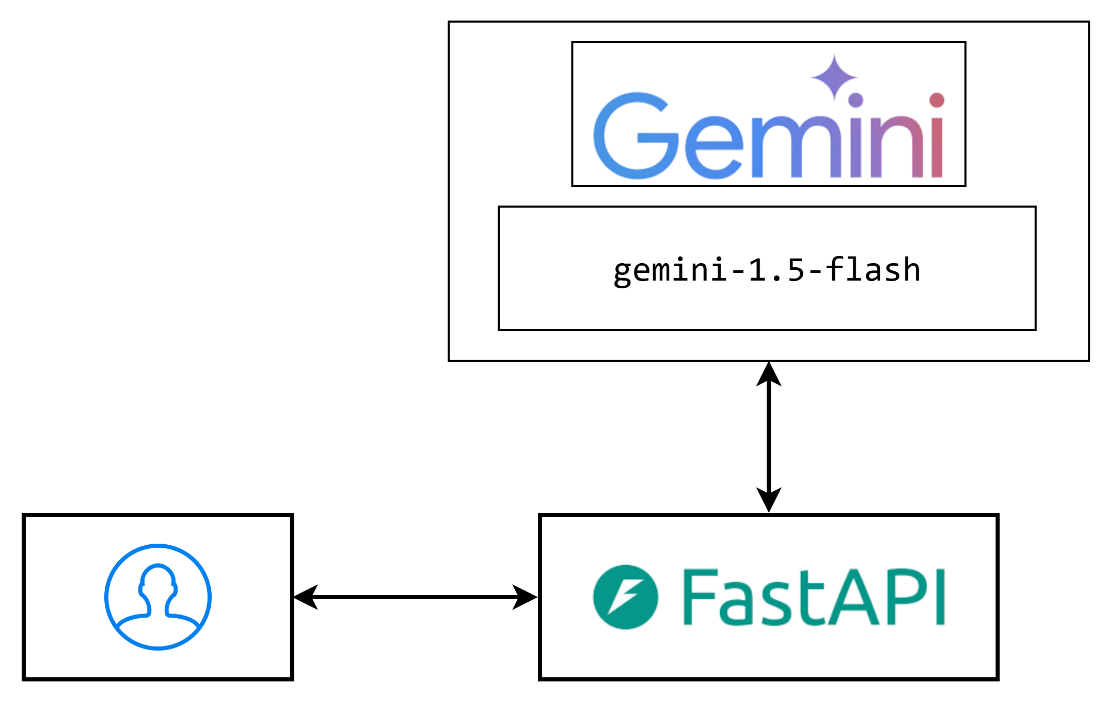
• Secure (API key integration with Gemini).

• High accuracy due to Gemini’s state-of-the-art language understanding.

# System Architecture

Workflow:

1. User sends a POST request with the input text to the FastAPI endpoint.  
2. The FastAPI server validates the request and forwards the text to the Gemini Model API.  
3. The Gemini model processes the text and generates a summary.  
4. FastAPI returns the summarized text as JSON response to the client.  
  
Diagram (conceptual):  
Client → FastAPI Endpoint → Gemini Model API → Summary → FastAPI Response → Client



# Conclusion

This project demonstrates the power of combining FastAPI and Gemini Model to deliver an efficient and scalable text summarization service. By leveraging state-of-the-art AI and modern API frameworks, we created a microservice that can be easily integrated into any application requiring automated text summarization.

# Future Improvements

• Add multi-language summarization support.

• Enable summarization of URLs, PDFs, or images (Gemini’s multimodal capability).

• Provide extractive vs. abstractive summarization options.

• Add user authentication and rate limiting.

• Integrate a front-end UI using React or Vue.