GEMINI DECODE PROJECT REPORT

Gemini Decode : Multilanguage Document Extraction Using GeminiPro by Tejaswi Tripuraneni

INTERNSHIP by SmartInternz & Google on

“Google Cloud Generative AI”, conducted on a virtual platform.

INDEX

I. Abstract

With the exponential rise in digital documentation, organizations and individuals often handle documents in multiple languages and formats. Manual processing of these documents is both time-consuming and error-prone, particularly when they involve non-native languages. This project addresses this challenge by utilizing Gemini Pro, a state-of-the-art AI model capable of understanding text, images, and structured content across a wide spectrum of languages. By automating extraction, translation, and summarization, this ensures faster processing, reduced human effort, and higher accuracy in handling multilingual documents. This innovative approach leverages natural language processing (NLP), computer vision, and AI-powered language models, enabling it to work seamlessly across global business environments, academic research, cross-border legal processes, and more.

II. Workflow

- User interacts with the UI to enter the input.
- User input is collected from the UI and transmitted to the backend using the Google API key.
- The input is then forwarded to the Gemini Pro pre-trained model via an API call.
- The Gemini Pro pre-trained model processes the input and generates the output.
- The results are returned to the frontend for formatting and display.

To accomplish this, we have to complete all the activities listed below:

- *Requirements Specification*
    - Create a requirements.txt file to list the required libraries.
    - Install the required libraries
- *Initialization of Google API Key*
    - Generate Google API Key
    - Initialize Google API Key

- *Interfacing with Pre-trained Model*
    - Load the Gemini Pro pre-trained model
    - Implement a function to get Gemini response
    - Implement a function to read PDF content
    - Write a prompt for Gemini model
- *Model Deployment*
    - Integrate with Web Framework
    - Host the Application

III. Code Implementation

Create a file named “chatpdf_app.py”.

```
from dotenv import load_dotenv
import streamlit as st
import os
import google.generativeai as genai
from PIL import Image

load_dotenv()##load all the environment variables
genai.configure(api_key="my-api-key")

model=genai.GenerativeModel('gemini-1.5-flash')
def get_gemini_response(input, image):
   if input!="":
      response=model.generate_content([input, image])
   else:
```

```
    response=model.generatie_content(image)
  return response.text


st.set_page_config(page_title="GeminiDecode:Multilanguage Document Extraction using GeminiPro by Tejaswi")


st.header("GeminiDecode: Multilanguage Document Text Extraction")
input = st.text_input("Input : ", key = "input")


uploaded_file=st.file_uploader("choose an image of the document:", type=["jpg","jpeg","png"])
image = ""
if uploaded_file is not None:
  image=Image.open(uploaded_file)
  st.image(image,caption="Uploaded Image",use_container_width=True)
submit=st.button("SUBMIT")


input_prompt="""
You are expert in understanding invoices.
We will upload a image as invoice and you will have to answer any questions based on the uploaded invoice image.
"""


#initialize streamlit app
st.header("GeminiDecode: Multilanguage Document Extraction Project")
text="Utilizing Gemini Pro AI,this project effortlessly extracts vital information + \
```

```
from diverse multilingual documents, transcending language barriers with \nprecision and + \

efficiency for enchanced  productivity and decision making."

styled_text = f"<span style='font-family:serif;'>{text}</span>"

st.markdown(styled_text, unsafe_allow_html=True)


## If submit button is clicked

if submit:

    response = get_gemini_response(input, image)


    # Check if response is valid before displaying

    if response:

        st.subheader("The Output Is:")

        st.write(response)

    else:

        st.error("No response received from the API. Please check your input or try again.")
```

IV. Software Requirements

- **Libraries Required:**

•**Streamlit:** Streamlit is a powerful framework for building interactive web applications with Python.

•**Streamlit_extras:** Additional utilities and enhancements for Streamlit applications.

•**Google-generativeai:** Python client library for accessing the GenerativeAI API, facilitating interactions with pre-trained language models like Gemini Pro.

•**Python-dotenv:** Python-dotenv allows you to manage environment variables stored in a .env file for your Python projects.

•**PyPDF2:** It is a Python library for extracting text and manipulating PDF documents.

•**Pillow:** Pillow is a Python Imaging Library (PIL) fork that adds support for opening, manipulating, and saving many different image file formats.

- **Programming Language:**
  Python– for AI integration, data processing, and automation, which is implemented using VS Code.

- **Cloud Services:**
  Google Cloud Platform (GCP) – to access Gemini Pro and Vision APIs, and to obtain Google API Key.

- **Operating System:**
  Compatible with **Windows / Linux / macOS**.

V. Testing

The testing process starts with uploading a document image in **JPG, JPEG, or PNG** format using **Streamlit**'s file uploader. The uploaded image is displayed to confirm successful upload. The user can also enter an optional text **input prompt** for context.

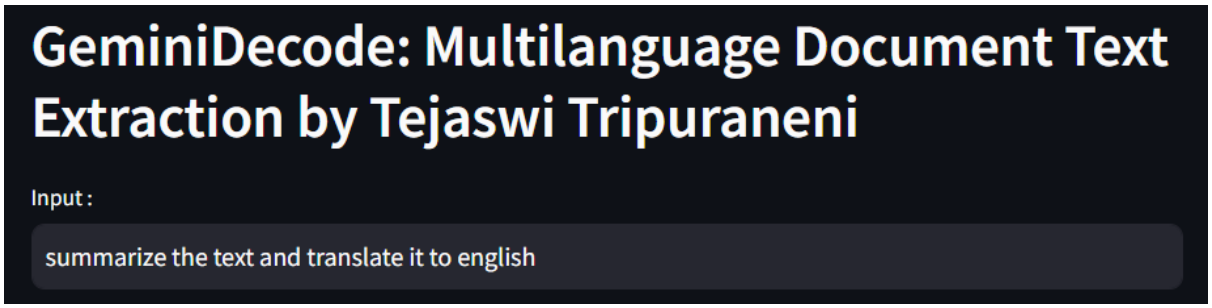
When the **Submit** button is clicked, the image and input are sent to the **Gemini Pro API** using the configured API key. The API processes the content and returns the extracted text, which is displayed.

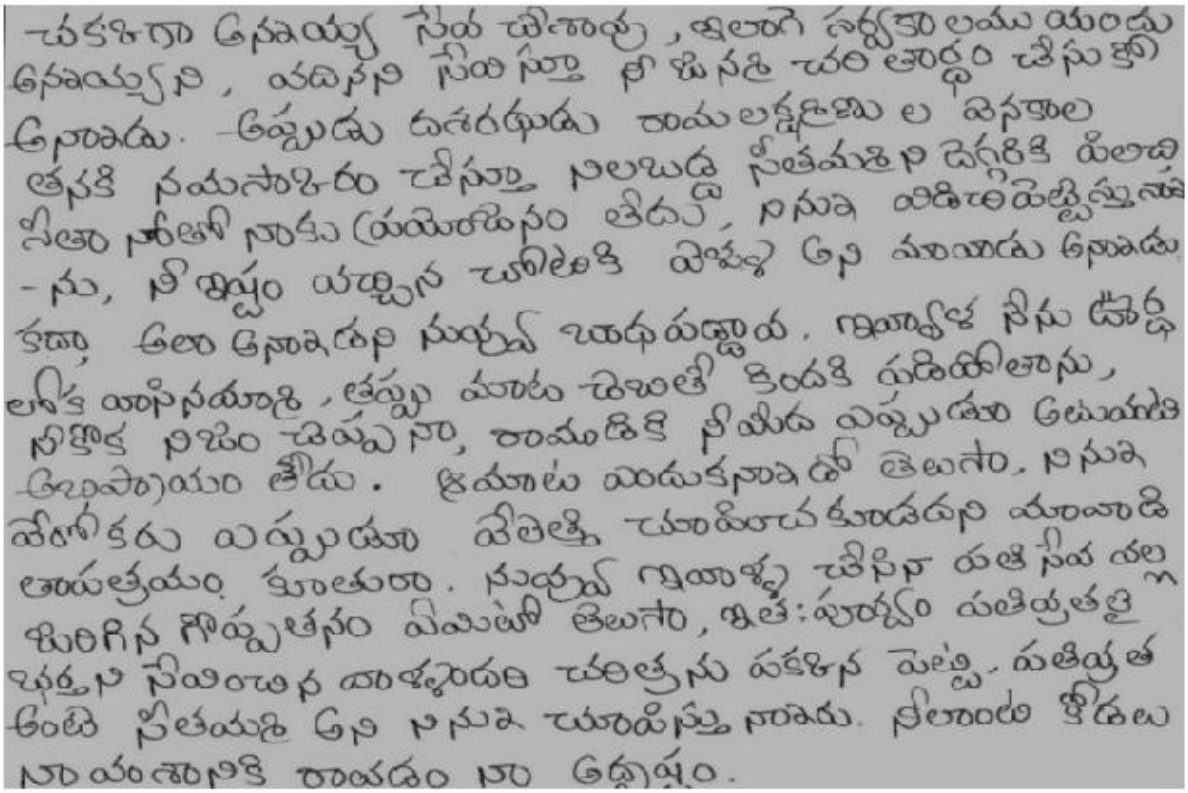
Testing covers image compatibility, accurate text extraction, response handling, and proper error messages for empty responses or invalid inputs, ensuring smooth performance.

VI. Results/Output

This project successfully extracts text from JPG, JPEG, PNG images using Gemini Pro, handles multilingual content well, and gives quick, accurate results. Errors like invalid files are handled properly, ensuring smooth and user-friendly performance.

❖ *Input 1: TELUGU TEXT*





Corresponding Output 1:

The Output Is:

Here's a summary and translation of the provided Telugu text:

Summary:

The text narrates a conversation between Dasharatha (Rama's father) and Seetha. Dasharatha praises Seetha's devotion and service to Rama, emphasizing that her exemplary behavior elevates her above all other devoted wives in history. He expresses his happiness and fortune in having such a daughter-in-law in his family.

English Translation:

"-You have served your brother-in-law (Rama) very well. Continue serving your brother-in-law and sister-in-law (Rama and Lakshmana) like this throughout your life and fulfill your purpose of birth," he said. Then Dasharatha, seeing Seeta standing respectfully behind Rama and Lakshmana, called her near and said, "Seeta, I have no complaints against you; you can go wherever you wish," he said. You were upset that he had said something like that. Today I am going to the upper world. If I tell a lie, I will fall down. Can I tell you the truth? Rama never had any such negative opinion about you. Do you know why he said that? He said it so that no one else would ever point a finger at you. It's my concern, daughter. Do you know the greatness of the service you performed to your husband today? From now on, when people talk about devoted wives, they will only talk about you. You have become the epitome of a devoted wife. To have a daughter-in-law like you is my good fortune."

❖ *Input 2: GERMAN TEXT*

Die Frösche in der Milch

Frei nacherzählt nach einer Fabel des Aesop

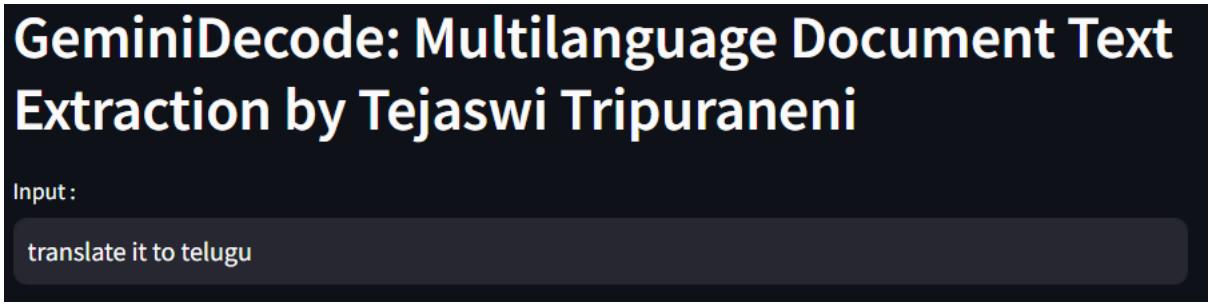
gestaltet von Janine Kahl

In einem heißen Sommer hatte die Sonne den Teich ausgetrocknet und zwei Frösche mussten sich auf die Wanderschaft machen. Im benachbarten Bauernhaus fanden sie die Küche und die kühle Speisekammer und dort einen Topf mit frischer fetter Milch. Schwupps, sprangen sie hinein und tranken, dass es schmatzte.

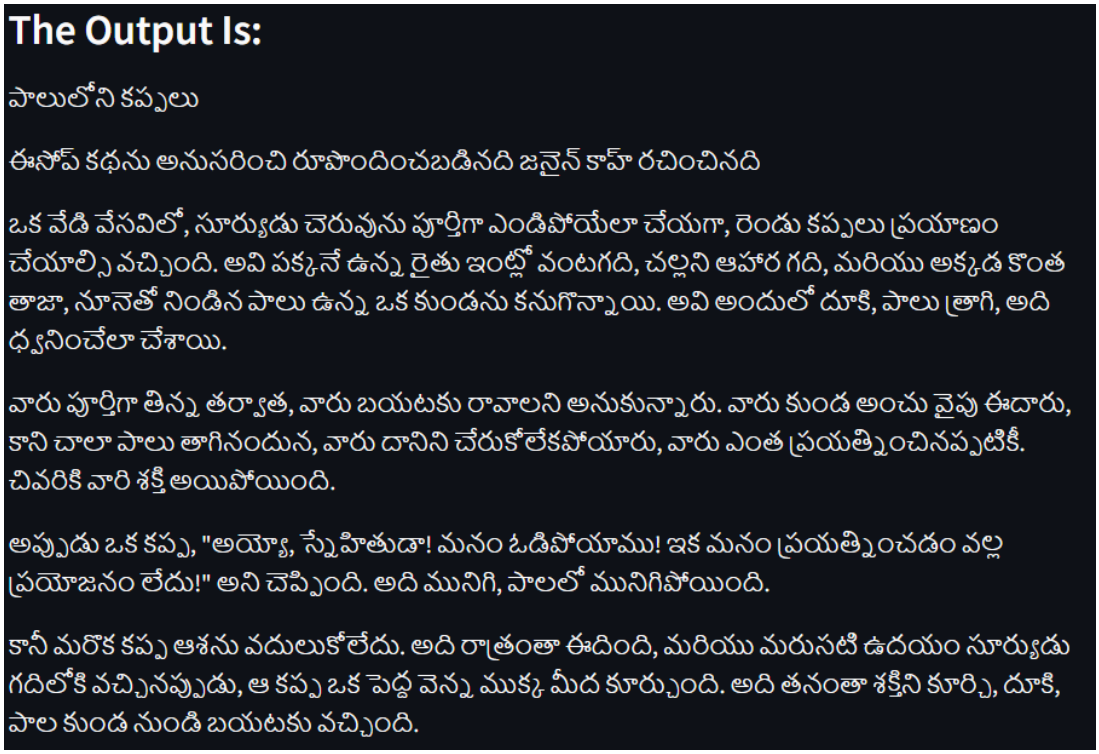
Als sie nun satt waren, wollten sie wieder heraus. Sie schwammen zum Rand des Kruges, doch weil sie so viel gesoffen hatten, kamen sie nicht mehr an ihn heran, sosehr sie auch hampelten und strampelten. Allmählich ließen auch ihre Kräfte nach.

Da sagte der eine Frosch: "Es ist aus, Kamerad! Wir sind verloren! Es hat keinen Sinn mehr, dass wir uns weiter abmühen!" Damit ließ er sich sinken und ertrank in der Milch.

Der andere Frosch aber gab die Hoffnung nicht auf. Er schwamm und strampelte die ganze Nacht, und als am nächsten Morgen die Sonne in die Kammer schien, saß der Frosch auf einem großen Butterklumpen. Er nahm all seine Kraft zusammen und hupps, sprang er aus dem Milchkrug und davon.

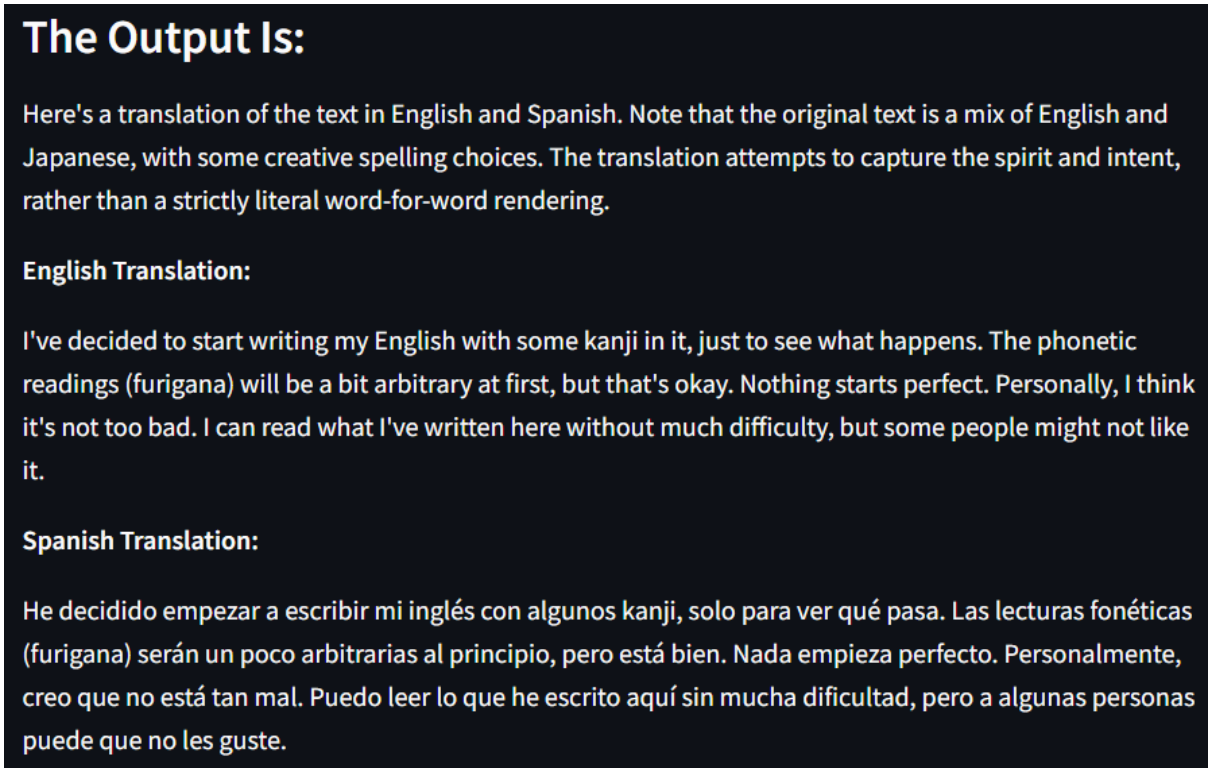


Corresponding Output 2:



- ***Input 3: MIXED LANGUAGE TEXT [JAPANESE, CHINESE & ENGLISH LANGUAGES]***

Corresponding Output 3:



VII. Challenges & Solutions

CHALLENGES	SOLUTIONS
1. Mixed-language documents	1. Used **Gemini Pro**'s multilingual capabilities.
2. Unformatted text display	2.Used **Streamlit markdown** for clean output.
3. API errors or timeouts	3.Error handling was added.

4. Low quality images with blur or noise.	4.Used **Pillow** Library for resizing & enhancing contrast of images.
5. Unsupported file formats	5. Restricted uploads to **JPG, JPEG, PNG** using file_uploader type filter.

VIII. Future Scope

- Improve **language detection** for rare/regional languages.
- Enhance **speed and accuracy** using future **Gemini Pro versions**.
- Expand input capability to handle **PDFs, & scanned documents.**
- Add **voice input support** to describe document context.
- Enable **real-time translation** for live document feeds.

IX. Conclusion

Gemini Decode: Multilingual Document Extraction using GeminiPro, leverages the power of Gemini Pro to efficiently extract and process text from multilingual documents across formats like JPG, JPEG, PNG. With its user-friendly interface built on Streamlit, it bridges language barriers and enhances productivity by providing accurate, fast, and structured data extraction. Combined with robust error handling and future-ready scalability, this project demonstrates the potential of AI-powered document understanding, making it highly valuable for businesses, researchers, and professionals working with global, multilingual content.

X. Acknowledgement

With this, I conclude my project *GeminiDecode: Multilingual Document Extraction using Gemini Pro*, and I sincerely thank SmartInternz and Google Cloud for their invaluable support and platform. This journey has been a great learning experience, allowing me to explore AI, Cloud Integration, and Document Processing. Looking forward to applying these skills in future projects.