Project Report: GeminiDecode: Multi Language Document Extraction by Gemini Pro

1. Introduction

1.1 Project Title: *GeminiDecode: Multi Language Document Extraction by Gemini Pro*

Made By: Shubham Godse
Mentor :

2. Project Overview

GeminiDecode: Multilanguage Document Extraction by Gemini Pro is a cutting-edge generative AI project designed to extract and process data from documents in multiple languages.
Leveraging advanced **Natural Language Processing (NLP)** and **Deep Learning** models, this tool allows global businesses and organizations to efficiently extract, process, and categorize information from documents in over 50 different languages.

This solution caters to multiple industries, including the legal, financial, and healthcare sectors, ensuring efficiency, accuracy, and ease of document handling. The project automates the extraction of key information and supports various document formats, enabling streamlined workflows, improved productivity, and enhanced decision-making.

3. Technical Skills Required

- **Python**: For implementing the back-end logic and interfacing with APIs.

- **Deep Learning**: For model handling and document extraction.

- **Streamlit**: For building the user interface (UI) and deploying the application.

4. Project Scenarios and Use Cases

Scenario 1: Legal Sector In the legal sector, **GeminiDecode** can swiftly extract and organize multilingual legal documents, including contracts and affidavits. This helps law firms reduce

manual processing time, ensure compliance with various legal standards, and focus on client interaction and strategy.

Scenario 2: Financial Institutions Banks and financial institutions can use **GeminiDecode** to process loan applications and financial statements in multiple languages. Automating data extraction helps streamline loan approvals and ensure compliance with global financial regulations, improving customer service.

Scenario 3: Healthcare Industry Hospitals and clinics benefit from **GeminiDecode** by extracting and processing multilingual patient records, enabling healthcare providers to offer timely and accurate care. The system enhances the management of medical records and treatment plans, improving patient outcomes.

5. Technical Architecture

Below is the **technical architecture** used in this project:
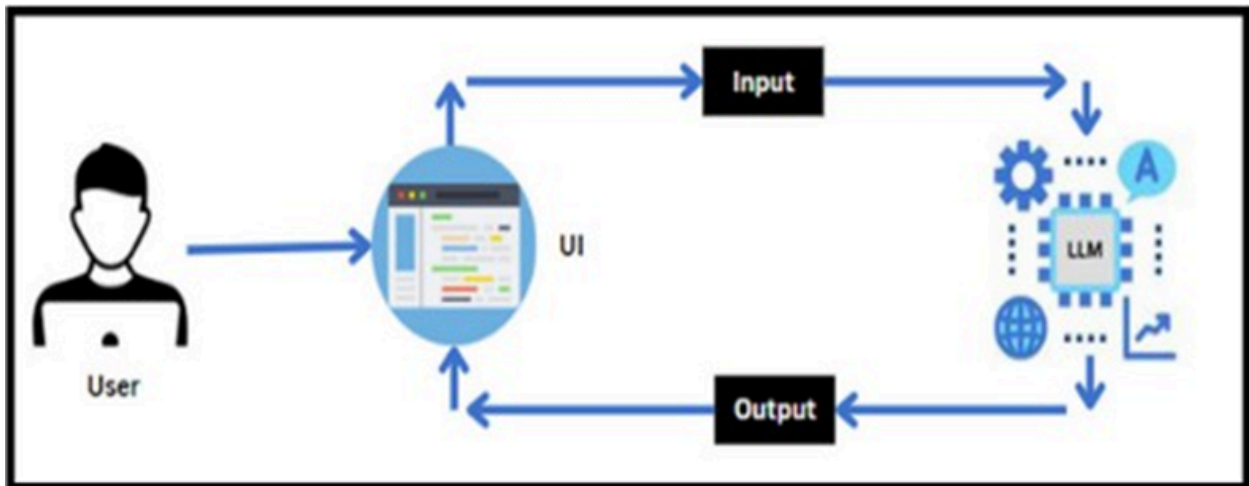
The architecture consists of the following components:

User Interface (UI): Users input data through a Streamlit-based front-end interface.

Input Processing: The input is transmitted to the backend using a **Google API key**

Gemini Pro Pre-trained Model: Input is processed by Gemini Pro, which identifies,extracts, and categorizes information from documents.

Output: The processed information is returned to the UI for presentation and analysis.

6. Project Flow

The project follows these key stages:

1. **User Input**

   ■ Users interact with the UI (Streamlit-based) to provide documents for extraction.

   ■ Input is transmitted to the backend via the **Google API key**.

2. **Interfacing with Pre-trained Model**

   ■ The **Gemini Pro Pre-trained Model** processes the user-provided documents.

   ■ The model extracts and categorizes information using NLP and machine learning

   techniques.

3. **Output**

   ■ The results of the document extraction are formatted and displayed on the

   front-end UI for review and further analysis.

7. Implementation Steps

To execute the **GeminiDecode** project, follow these steps:

7.1 Requirements Specification

- **Create a `requirements.txt`** file to list the required libraries.

- Install all necessary dependencies, including:

    - `streamlit`

    - `transformers`

    - `requests`

    - Other libraries required for NLP and API interfacing.

7.2 Google API Key Initialization

- **Generate a Google API key** to access cloud-based models.

- Add the Google API key to a `.env` file for secure storage.

7.3 Interfacing with the Pre-trained Model

- Load the **Gemini Pro Pre-trained Model**.

- Implement functions to:

    - Handle user input (read PDF documents, etc.).

    - Send input to the Gemini API for processing.

    - Receive and display the processed output.

7.4 Deployment and Hosting

- Integrate the application with **Streamlit** for easy deployment.

- Deploy the project to a platform like **Heroku** or **Streamlit Cloud** for real-time document

    extraction and processing.

8. File Organization

Organize your project files into a well-structured directory for ease of management and deployment. Below is an example of the project folder structure:

bash

Copy code

```
GeminiDecode/
|
├── assets/      # Stores images used in the UI.
├── outputs_screenshots/ #screenshots of execution and User Interface
├── .env         # Contains Google API key for secure access.
├── app.py       # Main application file for Streamlit and model logic.
├── requirements.txt # Lists all necessary libraries and dependencies.
```

```
└── README.md    #Provides an overview of the project and instructions.
```

9. Code Execution Screenshots

The following screenshots showcase various stages of the project execution, including code snippets, model interactions, and the user interface (UI) on Streamlit. These screenshots serve as proof of concept and successful execution of the GeminiDecode project.
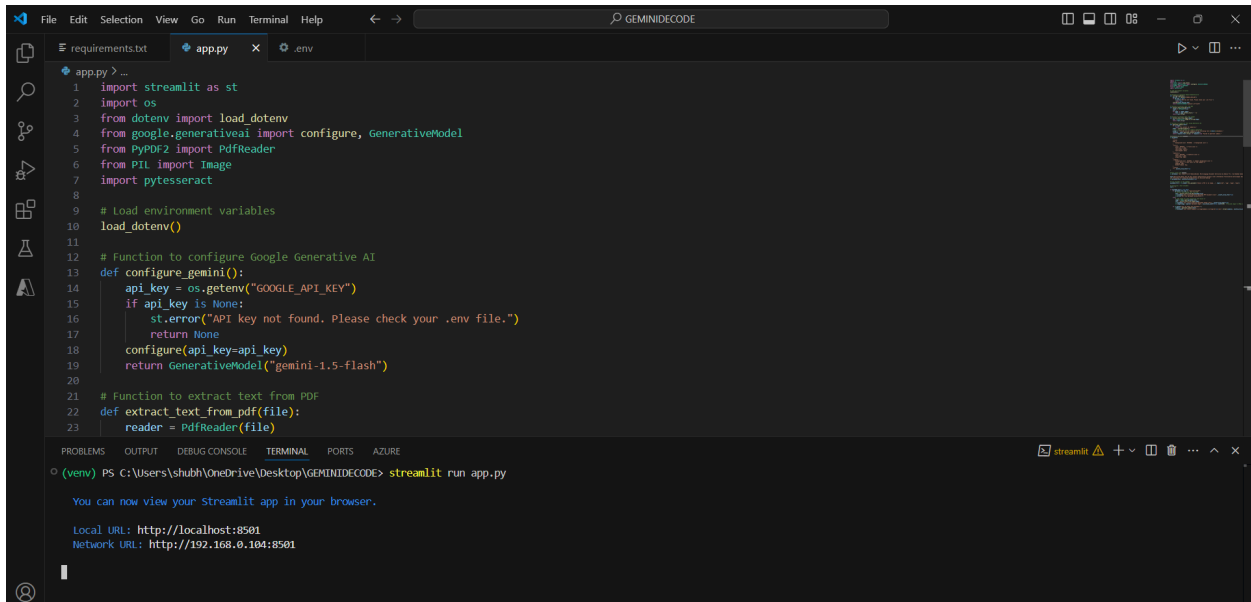
1. **Streamlit UI Interface**

A screenshot demonstrating the front-end interface for document input and output display.
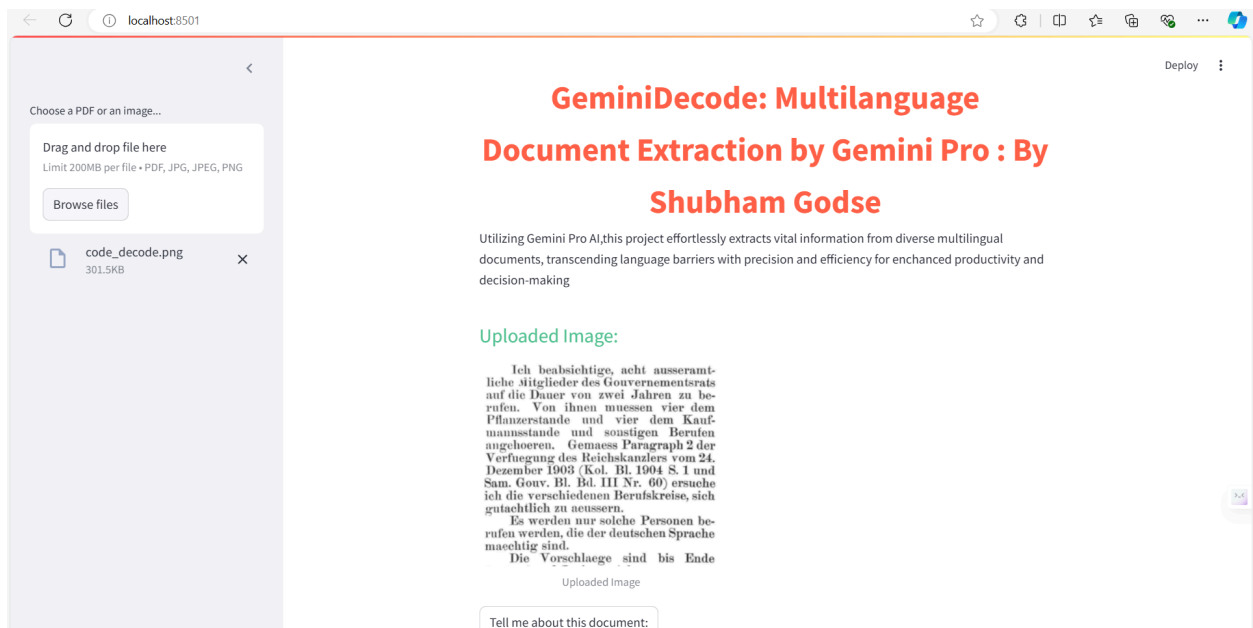
1. **Code Snippets**

Screenshots of the key sections of the Python code, including loading the Gemini Pro Pre-trained model, input processing, and API interfacing.


```python
import streamlit as st
import os
from dotenv import load_dotenv
from google.generativeai import configure, GenerativeModel
from PyPDF2 import PdfReader
from PIL import Image
import pytesseract

# Load environment variables
load_dotenv()

# Function to configure Google Generative AI
def configure_gemini():
    api_key = os.getenv("GOOGLE_API_KEY")
    if api_key is None:
        st.error("API key not found. Please check your .env file.")
        return None
    configure(api_key=api_key)
    return GenerativeModel("gemini-1.5-flash")

# Function to extract text from PDF
def extract_text_from_pdf(file):
    reader = PdfReader(file)
```

```
(venv) PS C:\Users\shubh\OneDrive\Desktop\GEMINIDECODE> streamlit run app.py

    You can now view your Streamlit app in your browser.

    Local URL: http://localhost:8501
    Network URL: http://192.168.0.104:8501
```

2. **Model Output Example**

A screenshot of the output displayed after processing a multilingual document.
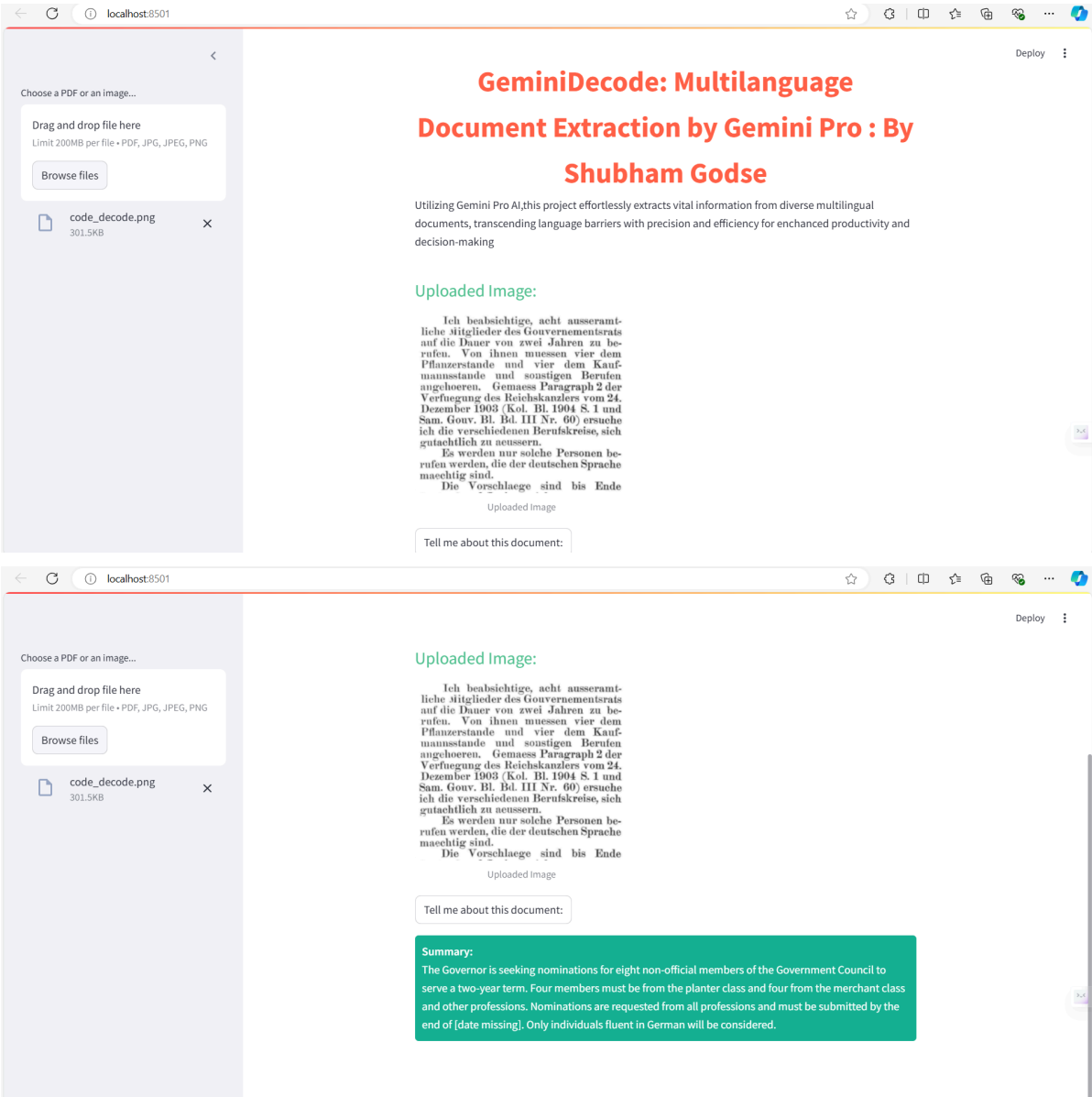
Deploy

GeminiDecode: Multilanguage Document Extraction by Gemini Pro : By Shubham Godse

Utilizing Gemini Pro AI,this project effortlessly extracts vital information from diverse multilingual documents, transcending language barriers with precision and efficiency for enchanced productivity and decision-making

Uploaded Image:

IO MANUEL

misere; la prospérité & les succès accompagnent l'homme industrieux.
Quel est celui qui a acquis des richesses, qui est devenu puissant, qui s'est couvert de gloire, dont l'éloge retentit par-tout, qui siege au conseil du Roi? C'est celui qui bannit la paresse de sa maison, & qui a dit à l'oisiveté : tu es mon ennemie.

Uploaded Image

Tell me about this document:

Deploy

Uploaded Image:

IO MANUEL

misere; la prospérité & les succès accompagnent l'homme industrieux.
Quel est celui qui a acquis des richesses, qui est devenu puissant, qui s'est couvert de gloire, dont l'éloge retentit par-tout, qui siege au conseil du Roi? C'est celui qui bannit la paresse de sa maison, & qui a dit à l'oisiveté : tu es mon ennemie.

Uploaded Image

Tell me about this document:

Summary:
The text, written in French, is a motivational piece emphasizing the connection between hard work and success. It highlights the characteristics of a successful individual, emphasizing diligence and the rejection of idleness. The text suggests that those who work hard will achieve wealth, power, glory, and recognition, while those who indulge in laziness will remain unsuccessful.

1. Challenges Faced

During the implementation of the **GeminiDecode** project, the following challenges were encountered:

● **Handling Multiple Languages**: Processing and extracting information from multiple languages required optimization in terms of NLP models and their deployment.

● **API Response Time**: Ensuring minimal delay in processing large document files.

● **Data Formatting**: Properly formatting the extracted information for presentation in the UI. These challenges were addressed through model fine-tuning and API optimization techniques.

2. Conclusion

The **GeminiDecode: Multilanguage Document Extraction by Gemini Pro** project was successfully executed and tested. It has proven to be a powerful tool for extracting, processing, and categorizing multilingual documents in industries such as legal, financial, and healthcare. With the integration of advanced machine learning and NLP techniques, the project automates document management tasks, reduces manual effort, and improves decision-making processes across global organizations.

By utilizing a user-friendly interface built with Streamlit, the project is easily accessible and can be deployed for real-time document extraction tasks.

3. Future Work

Further improvements could include:

● Expanding the language support to additional languages.
● Enhancing the accuracy of document classification and extraction.
● Adding more customizability to the extraction process to allow users to specify sections of documents to extract.