INNOMATICS®
RESEARCH LABS

INNOVATION. AUTOMATION. ANALYTICS

PROJECT ON

AI-Powered Solution for Assisting Visually Impaired Individuals

Submitted By
Pramod Mahendarkar

About me

I currently work at Infosys as System engineer, where I'm gaining valuable experience in Mainframes Technology but right now, I'm enhancing my skills through an exciting internship opportunity. Currently, I'm taking an exciting Internship in data science, a field that's revolutionizing industries through the power of data-driven insights. Passionate about Generative AI and Machine Learning, Data Engineering. I'm always eager to learn and adapt in this ever- evolving industry. I'm passionate about harnessing the potential of data to solve real-world problems, and this hands-on experience is sharpening my skills in programming languages like Python, working with big data, and utilizing cutting-edge tools like TensorFlow and Hadoop.

https://www.linkedin.com/in/pramod-mahendarkar/

https://github.com/PramodMahendarkar

INNOMATICS RESEARCH LABS

Problem Statement

➢ Visually impaired individuals often face challenges in understanding their environment, reading visual content, and performing tasks that rely on sight.

➢ This project leverages Generative AI to assist visually impaired individuals by providing functionalities like real-time scene understanding, text-to-speech conversion, and object detection.

- There is a need for an intelligent, adaptable, and user-friendly solution that provides:
- Real-time scene understanding.
- Text-to-speech conversion for reading visual content.
- Object and obstacle detection for safe navigation.
- Personalized assistance for daily tasks.

Objectives:

- Provide real-time scene understanding.
- Extract and convert text from images to speech.
- Identify objects and obstacles for safe navigation.
- Offer personalized assistance for daily tasks.

Implementation Details:

Technologies Used

1. LangChain: Conversational AI capabilities.
2. Streamlit: User-friendly web application interface.
3. Google Generative AI (Gemini API): Generates detailed scene descriptions.
4. Pytesseract: Optical Character Recognition (OCR) to extract text from images.
5. pyttsx3: Text-to-Speech engine for audio output.

Functionalities Implemented:

1. Real-Time Scene Understanding

- Generates detailed scene descriptions using Google Generative AI.
- Output includes objects, overall scene description, and safety suggestions.

2. Text-to-Speech Conversion

- Extracts text from images using OCR.
- Converts the extracted text to audible speech for accessibility

User Interaction Flow:

1.**Image Upload**: Users upload an image through the Streamlit app.
2. **Feature Selection:** Choose between scene description, text extraction, or text-to-speech.
3. **Processing and Output: -**
   Scene Understanding: Detailed descriptions generated.
   OCR and Speech: Text extracted and read aloud.

Implementation Code:

```
import streamlit as st
from PIL import Image
import pytesseract
import pyttsx3
import os
import google.generativeai as genai

pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'  # Replace with your path
print(pytesseract)

# Initialize Google Generative AI with API Key
GEMINI_API_KEY = "AIzaSyAb9_zwqvdREhTSe4WMjgtRfmWMt5p63eI"  # Replace with your valid API key
os.environ["GOOGLE_API_KEY"] = GEMINI_API_KEY
genai.configure(api_key="AIzaSyAb9_zwqvdREhTSe4WMjgtRfmWMt5p63eI")

# Initialize Text-to-Speech engine
engine = pyttsx3.init()
```

```
# Set up Streamlit page
st.set_page_config(page_title="SightAssist", layout="wide")
st.title("🤖 AI Assistant for Visually Impaired🧠")
st.sidebar.title("🔧 Features")
st.sidebar.markdown("""
- Scene Understanding
- Text-to-Speech
- Object & Obstacle Detection
""")

def extract_text_from_image(image):
    """Extracts text from the given image using OCR."""
    text = pytesseract.image_to_string(image)
    return text

def text_to_speech(text):
    """Converts the given text to speech."""
    engine.say(text)
    engine.runAndWait()
```

```
def generate_scene_description(input_prompt, image_data):
    """Generates a scene description using Google Generative AI."""
    model = genai.GenerativeModel('gemini-1.5-pro')
    response = model.generate_content([input_prompt, image_data[0]])
    return response.text

def input_image_setup(uploaded_file):
    """Prepares the uploaded image for processing."""
    if uploaded_file is not None:
        bytes_data = uploaded_file.getvalue()
        image_parts = [
            {
                "mime_type": uploaded_file.type,
                "data": bytes_data
            }
        ]
        return image_parts
    else:
        raise FileNotFoundError("No file uploaded.")
```

INNOMATICS
RESEARCH LABS

```
# Main app functionality
uploaded_file = st.file_uploader("📤 Upload an image...", type=["jpg", "jpeg", "png"])
if uploaded_file:
    image = Image.open(uploaded_file)
    st.image(image, caption="Uploaded Image",  use_container_width=True)

# Buttons for functionalities
col1, col2, col3 = st.columns(3)
scene_button = col1.button("🔍 Describe Scene")
ocr_button = col2.button("📝 Extract Text")
tts_button = col3.button("🔊 Text-to-Speech")

# Input Prompt for AI Scene Understanding
input_prompt = """
You are an AI assistant helping visually impaired individuals by describing the scene in the image. Provide:
1. List of items detected in the image with their purpose.
2. Overall description of the image.
3. Suggestions for actions or precautions for the visually impaired.
"""
```

```
# Process based on user interaction
if uploaded_file:
    image_data = input_image_setup(uploaded_file)

    if scene_button:
        with st.spinner("Generating scene description..."):
            response = generate_scene_description(input_prompt, image_data)
            st.subheader("Scene Description")
            st.write(response)

    if ocr_button:
        with st.spinner("Extracting text from image..."):
            text = extract_text_from_image(image)
            st.subheader("Extracted Text")
            st.write(text)

    if tts_button:
        with st.spinner("Converting text to speech..."):
            text = extract_text_from_image(image)
            if text.strip():
```

```
    text_to_speech(text)
    st.success("Text-to-Speech Conversion Completed!")
else:
    st.warning("No text found in the image.")
```

INNOMATICS
RESEARCH LABS

Conclusion

This project demonstrates an effective application of AI to assist visually impaired individuals. It combines real-time scene understanding with OCR and text-to-speech capabilities, enabling enhanced accessibility and independence.

THANK YOU

INNOMATICS
RESEARCH LABS