In [14]:

```python
import speech_recognition as sr
import os
import sys
from reportlab.platypus import SimpleDocTemplate, Table, TableStyle, Paragraph
from reportlab.lib.styles import getSampleStyleSheet
from reportlab.lib.pagesizes import letter
from reportlab.lib import colors
from reportlab.lib.units import inch

def calculate_wer(reference, hypothesis):
    """
    Calculates the Word Error Rate (WER) between a reference and hypothesis st
    """
    ref_words = reference.lower().split()
    hyp_words = hypothesis.lower().split()

    d = [[0] * (len(hyp_words) + 1) for _ in range(len(ref_words) + 1)]

    for i in range(len(ref_words) + 1):
        d[i][0] = i
    for j in range(len(hyp_words) + 1):
        d[0][j] = j

    for i in range(1, len(ref_words) + 1):
        for j in range(1, len(hyp_words) + 1):
            cost = 0 if ref_words[i - 1] == hyp_words[j - 1] else 1
            d[i][j] = min(d[i - 1][j] + 1, d[i][j - 1] + 1, d[i - 1][j - 1] +

    errors = d[len(ref_words)][len(hyp_words)]
    wer = errors / len(ref_words) if len(ref_words) > 0 else float('inf')
    return wer

def save_results_as_pdf(reference_text, results, filename="comparison_results.
    """
    Saves the comparative analysis results to a PDF file in a table format.
    """
    try:
        doc = SimpleDocTemplate(filename, pagesize=letter)
        styles = getSampleStyleSheet()
        elements = []

        # Title
        title = Paragraph("Comparative Speech Recognition Analysis", styles['h
        elements.append(title)
        elements.append(Spacer(1, 0.2*inch))

        # Reference Text
        ref_style = styles['Normal']
        ref_paragraph = Paragraph(f"<b>Reference Text:</b> {reference_text}",
        elements.append(ref_paragraph)
        elements.append(Spacer(1, 0.3*inch))

        # Prepare data for the table
        google_wer_str = f"{results['google']['wer']:.2%}" if results['google'
        whisper_wer_str = f"{results['whisper']['wer']:.2%}" if results['whisp
        google_acc_str = f"{results['google']['accuracy']:.2%}" if results['go
        whisper_acc_str = f"{results['whisper']['accuracy']:.2%}" if results['

        # Wrap text for transcription cells
        google_transcription_p = Paragraph(results['google']['transcription'],
        whisper_transcription_p = Paragraph(results['whisper']['transcription'

        data = [
            ['Metric', 'Google Speech API', 'Whisper'],
            ['Recognized Text', google_transcription_p, whisper_transcription_
            ['Word Error Rate', google_wer_str, whisper_wer_str],
            ['Accuracy', google_acc_str, whisper_acc_str]
        ]

        # Create and style the table
        table = Table(data, colWidths=[1.5*inch, 3*inch, 3*inch])
        style = TableStyle([
            ('BACKGROUND', (0,0), (-1,0), colors.grey),
            ('TEXTCOLOR', (0,0), (-1,0), colors.whitesmoke),
            ('ALIGN', (0,0), (-1,-1), 'CENTER'),
            ('VALIGN', (0,0), (-1,-1), 'MIDDLE'),
            ('FONTNAME', (0,0), (-1,0), 'Helvetica-Bold'),
            ('BOTTOMPADDING', (0,0), (-1,0), 12),
            ('BACKGROUND', (0,1), (-1,-1), colors.beige),
            ('GRID', (0,0), (-1,-1), 1, colors.black)
        ])
        table.setStyle(style)

        elements.append(table)
        doc.build(elements)
        print(f"\nResults successfully saved to '{filename}'")
    except Exception as e:
        print(f"\nError creating PDF: {e}")


if __name__ == "__main__":
    r = sr.Recognizer()
    mic = sr.Microphone()

    print("Choose an audio source:")
    print("1: Microphone | 2: Audio File (.wav) | 3: Exit")
    source_choice = input("Enter choice (1/2/3): ")

    if source_choice == '3': sys.exit("Exiting program.")
    if source_choice not in ['1', '2']: sys.exit("Invalid choice. Exiting.")

    reference_text = input("\nEnter the exact reference text: ").strip()
    if not reference_text: sys.exit("Reference text cannot be empty. Exiting."

    audio_data = None
    if source_choice == '1':
        with mic as source:
            print("\nAdjusting for ambient noise...")
            r.adjust_for_ambient_noise(source, duration=1)
            print("Speak the reference text now...")
            try:
                audio_data = r.listen(source)
                print("Audio captured.")
            except sr.WaitTimeoutError:
                sys.exit("Listening timed out. Exiting.")
    elif source_choice == '2':
        file_path = input("Enter the path to your .wav file: ")
        if not os.path.exists(file_path): sys.exit("Error: File not found. Exi
        with sr.AudioFile(file_path) as source:
            audio_data = r.record(source)

    if not audio_data: sys.exit("Could not capture audio. Exiting program.")

    results = {
        "google": {"transcription": "N/A", "wer": None, "accuracy": None},
        "whisper": {"transcription": "N/A", "wer": None, "accuracy": None}
    }

    print("\nProcessing... This may take a moment.")

    # --- Google Speech API Analysis ---
    try:
        google_transcription = r.recognize_google(audio_data)
        results["google"]["transcription"] = google_transcription
        wer = calculate_wer(reference_text, google_transcription)
        results["google"]["wer"] = wer
        results["google"]["accuracy"] = max(0, 1 - wer)
    except sr.RequestError as e: results["google"]["transcription"] = f"API Er
    except sr.UnknownValueError: results["google"]["transcription"] = "Could r

    # --- Whisper Analysis ---
    try:
        whisper_transcription = r.recognize_whisper(audio_data, model="base")
        results["whisper"]["transcription"] = whisper_transcription
        wer = calculate_wer(reference_text, whisper_transcription)
        results["whisper"]["wer"] = wer
        results["whisper"]["accuracy"] = max(0, 1 - wer)
    except sr.RequestError as e: results["whisper"]["transcription"] = f"API E
    except sr.UnknownValueError: results["whisper"]["transcription"] = "Could

    # --- Display Results Table in Console ---
    print("\n" + "="*80)
    print("                    COMPARATIVE ANALYSIS RESULTS")
    print("="*80)
    print(f"REFERENCE TEXT: '{reference_text}'\n")
    print(f"{'Metric':<20} | {'Google Speech API':<35} | {'Whisper'}")
    print("-"*80)
    print(f"{'Recognized Text':<20} | {results['google']['transcription']:<35}
    google_wer_str = f"{results['google']['wer']:.2%}" if results['google']['w
    whisper_wer_str = f"{results['whisper']['wer']:.2%}" if results['whisper']
    print(f"{'Word Error Rate':<20} | {google_wer_str:<35} | {whisper_wer_str}
    google_acc_str = f"{results['google']['accuracy']:.2%}" if results['google
    whisper_acc_str = f"{results['whisper']['accuracy']:.2%}" if results['whis
    print(f"{'Accuracy':<20} | {google_acc_str:<35} | {whisper_acc_str}")
    print("="*80)

    # --- Save to PDF ---
    save_pdf_choice = input("Save these results to a PDF? (y/n): ").lower()
    if save_pdf_choice == 'y':
        pdf_filename = input("Enter a filename for the PDF (e.g., results.pdf)
        if not pdf_filename.lower().endswith('.pdf'):
            pdf_filename += '.pdf'
        save_results_as_pdf(reference_text, results, pdf_filename)

    print("\nAnalysis Complete.\n")
```

```
Choose an audio source:
1: Microphone | 2: Audio File (.wav) | 3: Exit
Adjusting for ambient noise...
Speak the reference text now...
Audio captured.

Processing... This may take a moment.

===============================================================================
=
                    COMPARATIVE ANALYSIS RESULTS
===============================================================================
=
REFERENCE TEXT: 'Max Verstappen is the worst driver in the world.'

Metric               | Google Speech API                    | Whisper
-------------------------------------------------------------------------------
-
Recognized Text      | Max verstappen is the worst driver in the world |  Mats
u's tapen is the worst driver in the world.
Word Error Rate      | 11.11%                               | 22.22%
Accuracy             | 88.89%                               | 77.78%
===============================================================================
=

Results successfully saved to 'CA3.pdf'

Analysis Complete.
```

In [ ]:
```
'''
The sun had begun its descent, casting long shadows across the valley and pair
'''
```