

Wikipedia Article Summary Generator

Project Title: Wikipedia Summary Tool

Developer: Gedda Muthya Harika

Date: 17 July 2025

Version: 1.0

Language & Tools: Python, Tkinter, pyttsx3, Wikipedia API, Threading

Purpose: To create a desktop tool that fetches summarized Wikipedia articles and reads them aloud with a graphical interface.

Abstract:

The Wikipedia Summary Tool is designed to provide quick, readable, and audible summaries of any Wikipedia topic. It combines three core components:

- Wikipedia API for retrieving content.
- pyttsx3 for text-to-speech conversion.
- Tkinter for the GUI interface.

The aim is to increase accessibility for users who prefer or require auditory learning, and to streamline content retrieval for students, researchers, and casual learners. The application presents information both visually and through voice, allowing simultaneous reading and listening experiences.

Introduction:

Problem Statement

In a world of information overload, summarization tools are essential for focused learning. Many users require not just visual but auditory means to access information—whether due to visual impairments, multitasking needs, or learning preferences.

Solution:

This tool provides a compact, responsive application that:

- Fetches content from Wikipedia
- Summarizes it in clean bullet-style format

- Displays results in a GUI
 - Speaks the content aloud concurrently using text-to-speech
-

Technology Stack:

Component	Role
-----------	------

Python	Programming language
--------	----------------------

Wikipedia API	Retrieve article content and summaries
---------------	----------------------------------------

pyttsx3	Offline text-to-speech engine
---------	-------------------------------

Tkinter	GUI framework for building the interface
---------	------------------------------------------

Threading	Enables parallel execution of voice + text
-----------	--------------------------------------------

All libraries are open-source, lightweight, and platform-independent, making this tool easy to set up and distribute.

System Architecture:

□ Workflow Overview

[User Input] → [Wikipedia API] → [Summary Text]



[Display in GUI] + [Speak with pyttsx3]

⚙️ Component Roles

- **Entry Field:** Accepts user topic query.
 - **Wikipedia API:** Queries and fetches summarized results.
 - **Text Box:** Displays the summary.
 - **pyttsx3 Voice Engine:** Speaks the text aloud using a separate thread.
 - **Threading:** Prevents GUI blocking during audio playback.
-

Implementation Details:

📄 Wikipedia Summary Retrieval

```
def simple_summary(topic, bullet_count=5):
```

```
    return wikipedia.summary(topic, sentences=bullet_count)
```

- Fetches limited summary sentences from the Wikipedia API.
- Handles disambiguation and fetch errors gracefully.

🗣️ Text-to-Speech with Threading

```
def speak_async(text):
```

```
    def run():
```

```
        engine = pyttsx3.init()
```

```
        engine.say(text)
```

```
        engine.runAndWait()
```

```
    threading.Thread(target=run).start()
```

- Runs the voice engine on a separate thread.
- Prevents blocking of the GUI and ensures fluid user interaction.

🖥️ GUI Structure



```
def create_gui():
```

```
    # GUI logic with entry, button, and output box
```

- Interactive GUI created using tkinter.
- Button triggers on_search() for both voice and summary display.

Features & Functionality:

Feature	Description
🔍 Wikipedia summary	Fetches summaries using topic search
🗣️ Voice output	Reads summary aloud in real-time
🖥️ Graphical interface	Clean, responsive GUI built with Tkinter

Feature	Description
 Thread-safe operation	Uses threading to speak without freezing the GUI
 Error handling	Displays meaningful messages if the topic is ambiguous

Optional enhancements could include:

- Highlighting spoken text word-by-word
- Saving voice output as a file
- Advanced NLP summarization

User Guide:

How to Run

1. Ensure Python and required modules are installed.
2. Run the script via command line or IDE.
3. Enter a Wikipedia topic in the input field.
4. Click “Summarize”.
5. Read and listen to the generated summary.

Requirements

`pip install wikipedia pyttsx3`

Add screenshots to demonstrate each phase:

- Startup screen
 - Input example
 - Output with voice
-

Challenges & Solutions:

Challenge	Solution
✗ GUI froze during speech	Introduced threading to decouple pyttsx3 from main loop
✗ Wikipedia DisambiguationError	Added error handling to guide users with suggestions
✗ Lag in response	Used parallel execution to maintain UI responsiveness

Other minor issues such as speech rate tuning or non-standard topics were handled by refining input queries and adjusting pyttsx3 settings.

Project Structure (Single-File Version)

You can save this entire code in a file named:

wikipedia_gui_summary.py

And run it using:

```
python wikipedia_gui_summary.py
```

Source Code

```
import wikipedia
import pyttsx3
import tkinter as tk from tkinter
import messagebox import threading # For simultaneous voice + GUI output
# 🔊 Voice output using pyttsx3 (runs in a separate thread)
def speak_async(text):
    def run(): try: engine = pyttsx3.init() engine.setProperty('rate', 150)
# Adjust speed as needed engine.say(text)
engine.runAndWait()
except Exception as e:
    print(f"Voice error: {e}") threading.Thread(target=run, daemon=True).start()
```

```

# daemon=True ensures thread exits with GUI
# 📄 Fetch Wikipedia summary
def simple_summary(topic, bullet_count=5):
    try: wikipedia.set_lang("en")
    return wikipedia.summary(topic, sentences=bullet_count)

    except wikipedia.exceptions.DisambiguationError as e: return "
    ⚠ Disambiguation error. Try one of these:\n- " + "\n- ".join(e.options[:5])
    except wikipedia.exceptions.PageError: return "
    ✖ No page found for that topic." except Exception as e: return f"
    ✖ Error: {str(e)}"

# 😊 GUI setup def create_gui():
def on_search():
    topic = entry.get().strip()
    if not topic:
        messagebox.showwarning("Input Error", "Please enter a topic.")
    return summary = simple_summary(topic)
    text_box.delete("1.0", tk.END)
    text_box.insert(tk.END, summary)
    speak_async(summary) window = tk.Tk()
    window.title("📄 Wikipedia Summary Tool")
    window.geometry("650x450")
    tk.Label(window, text="Enter a topic:", font=("Arial", 12)).pack(pady=10)
    entry = tk.Entry(window, width=50, font=("Arial", 12))
    entry.pack(pady=5)
    search_btn = tk.Button(window, text="Summarize", command=on_search,
    font=("Arial", 12))
    search_btn.pack(pady=10)

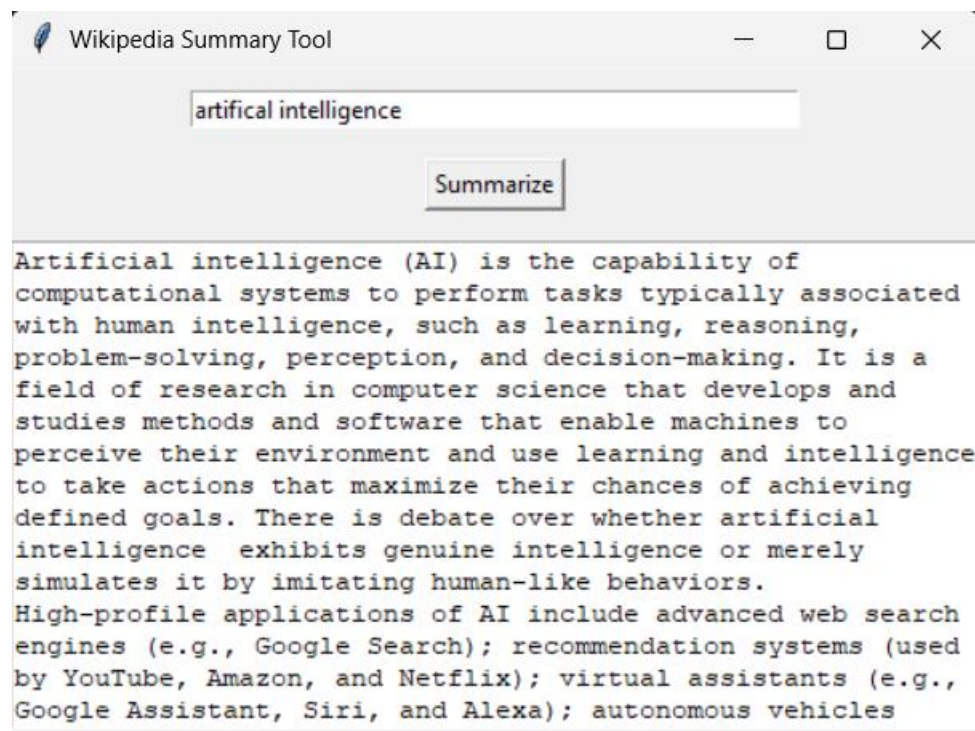
```

```
text_box = tk.Text(window, wrap=tk.WORD,  
width=70, height=15, font=("Arial", 11))  
text_box.pack(pady=10)  
window.mainloop()  
  
# 🌀 Run the app if __name__ == "__main__":  
create_gui()
```

□ How It Works

- The user inputs a topic in the GUI.
- The tool fetches the summary using the wikipedia API.
- It displays the summary in the Text box.
- Simultaneously, it reads the summary aloud using pyttsx3.

Output:



Conclusion & Future Scope:

The Wikipedia Summary Tool successfully integrates summary extraction and spoken output into a compact GUI application. It streamlines learning, supports auditory comprehension, and provides meaningful accessibility.

Future Enhancements:

- Add **NLP summarization** for deeper article understanding.
 - Include **highlighting of spoken text** in sync with audio.
 - Deploy as a **Flask web application** with REST API capabilities.
 - Store **user history** and generate quiz questions.
 - Package as a Windows .exe using pyinstaller.
-