Jarvis AI Desktop Assistant Powered by Google Gemini - Project Documentation

Developed by: Mohd Saif

Language: Python

AI Engine: Google Gemini API

Version: 1.0

# Introduction

Jarvis AI Desktop Assistant is a Python-based voice-controlled system that allows users to interact with their computer using natural speech. The assistant performs tasks such as opening websites, telling jokes, playing music, reporting time and date, taking screenshots, and even chatting with Google Gemini AI for smart answers. The purpose of this project is to help users interact with their computer more easily using natural voice commands.

# 2. Features

* Listens to your voice commands and understands them.
* Speaks back using a robotic voice.
* Searches on Wikipedia and gives short summaries.
* Opens popular websites like Google, YouTube, and Stack Overflow.
* Plays songs from your music folder.
* Answers smart questions using Google Gemini AI.
* System operations: screenshot, shutdown, restart, sleep, volume control.
* Tells jokes and makes the interaction fun.
* Announces the current time and date.

# 3. System Requirements

* Windows 10 or above
* Python 3.8 or higher
* Internet connection
* Working microphone
* Visual Studio Code installed

# 4. Installing Visual Studio Code

Follow these simple steps to install Visual Studio Code:

1. Go to the official website: https://code.visualstudio.com
2. Click on 'Download for Windows'.
3. Run the downloaded setup file and follow the installation instructions.
4. Once installed, open Visual Studio Code.
5. Click on 'File → Open Folder' and select your Jarvis project folder.
6. Open the 'jarvis.py' file.

Click on the 'Run ▶️' button to start your program.

# 5. Setting Up Jarvis

Steps to set up and run Jarvis:

1. Download or clone the project from GitHub.
2. Install all required libraries using the command: pip install -r requirements.txt
3. Create a .env file in your project folder and add your Gemini API key.
4. Run the program using the command: python jarvis.py
5. Speak your command when Jarvis says it is listening.

# 6. Explanation of Each Library

speech\_recognition: Allows Jarvis to listen and understand your voice.

pyttsx3: Enables Jarvis to speak responses.

webbrowser: Used to open websites like Google or YouTube.

wikipedia: Helps Jarvis search for summaries on topics.

pyautogui: Takes screenshots and controls system functions.

psutil: Checks system battery and performance information.

pyjokes: Generates random jokes for fun.

dotenv: Keeps API keys private using environment variables.

google.generativeai: Connects Jarvis to Google Gemini AI for smart answers.

# 7. Folder Structure

Jarvis/  
│  
├── jarvis.py ← Main program file  
├── .env ← Contains your Gemini API key  
├── .gitignore ← Keeps your .env private  
├── requirements.txt ← List of Python libraries  
└── README.md ← Project guide

# 8. How Jarvis Works

1. 1. Jarvis listens to your voice through the microphone.
2. 2. It converts your speech into text.
3. 3. Understands what command you said.
4. 4. Executes the task like opening a website or searching something.
5. 5. Speaks back the result to you.

# 9. Example Interactions

"Open YouTube" → Jarvis opens YouTube in your browser.

"What is the time?" → Jarvis tells you the current system time.

"Tell me a joke" → Jarvis responds with a random joke.

"Take a screenshot" → Jarvis captures and saves a screenshot.

"Search for Python programming" → Jarvis opens a Google search page.

# Future Enhancements

• Add weather updates and news headlines.

• Integrate email and messaging features.

• Add personalized reminders and calendar integration.

• Create a graphical interface for better user interaction.

• Support for multiple languages and voices.

# Conclusion

The Jarvis AI Desktop Assistant is a successful implementation of combining speech recognition, artificial intelligence, and automation. It simplifies daily computer operations using natural voice commands and showcases how AI can enhance human productivity . By building Jarvis, students learn the use of Python libraries, APIs, and AI models in real-world applications. This assistant is a simple step toward creating interactive and intelligent desktop systems.