Design and Implementation of a Voice identifying System

Draft Report

By

NAP Mbeje

21002606

Submitted in the Faculty of Engineering: Department of Electronic Engineering in Partial Fulfilment of the Requirements for the Bachelor of Technology in Electronic Engineering at the Durban University of Technology

May 2020

Plagiarism Declaration

1. I know and understand that plagiarism is using another person's work and pretending it is one's own, which is wrong.
2. This essay/report/project is my own work
3. I have appropriately referenced the work of other people I have used
4. I have not allowed, **and will not allow**, anyone to copy my work with the intention of passing it off as his/her own work.

Surname and Initials: Mbeje NAP Student number: 21002606

Table of Contents

Table of Figures

Abstract

Voice recognition enabled systems have been gaining popularity since multinational technology companies have released their developed voice systems that can even interact conversationally with a speaker. These systems are usually well developed and can come with a price if someone would be interested in implementing a voice recognition system in their own small projects.

This project aims to construct a successful voice recognition system using available open source hardware and software and aims to provide an easy to implement method for anyone to follow in assembling a voice recognition system for themselves to use. A successful voice system should be one that can interpret the spoken word in the form of a command and translate that command into an executable function that does meaningful work.

This project will provide a technical evaluation of all the steps in how to implement a successful open source voice recognition system.

The project makes use of the Raspberry Pi 3 model B.

Introduction

This document is a report of the design and implementation of a voice identifying system. This project attempts to create a voice identifying system, using open source tools, materials along with the Raspberry Pi, that can identify a command spoken by a user, that can also be trained to recognize commands.

The use of this sort of system can be realized in systems that integrate voice controls as in home automation.

With the increasing presence of personal voice assistance from multinational technology corporations in the technology space, there is interest for the knowledge on how to construct such a system for small scale personal projects.

The intention is to demonstrate the workability of open source technology.

Aims and Objectives of the Project

The aim of this project is to create a voice identifying system, using a RPi, that can be useful in deploying and implementing in an integrated system.

Overview of Report

This report is separated into five sections

1. Introduction
2. Research
3. Implementation
4. Result
5. Conclusion

In addition, there is a reference list and appendices which contain:

- Appendix A – Python Server Code
- Appendix B – System Command Map

Research

Overview of Digital Voice Detection

Sampling

Sampling is the process in which a continuous analogue signal is transformed into a discrete digital signal. The continuous signal is observed at set intervals using an analogue to digital converter which quantizes the observed signal (Ambardar, 2007).

Quantization is the assigning of a voltage or current measurement at each point of observation of the signal based on a set of known voltage or current levels. Each level is described by a binary number which is then stored in memory. If the signal should fall between levels, the closest level is selected, rounding the signal measurement to the nearest level. This means that the sampling process produces an approximation of the sampled analogue signal (Ambardar, 2007).

The collection of voltage or current measurements will be the set of values that will represent the original signal within the digital context. The stored values also represent the original analogue signal in space and time, plotting out the progression of the signal as time moves forward or backwards. Because digital signals can be operated on as a set of plotted points, performing digital signal processing with mathematical models becomes possible (Ambardar, 2007).

Because sampling happens at intervals, there is a potential for loss of information between the intervals. This will cause the problem of aliasing which describes the distortion of a signal when it is reconstructed so that it does not produce the same signal compared to the originally sampled signal. For this problem, the signal must be sampled in such a way that it prioritises the minimal loss of information. To achieve this, the Nyquist Rate is used as the lowest possible rate to sample a signal (Ambardar, 2007).

The Nyquist Rate is set at twice the analogue frequency. It is the lowest rate of sampling that an analogue signal can be sampled at without aliasing (Ambardar, 2007).

Digitization Process

The digitization of a sampled analogue signal is the converting of a sampled analogue signal into a digital form which are bits (Arul, 2015). These bits capture the quantized value observed by the sampling instrument. The number of bits used to represent the digital form of a sampled signal is called the bit resolution. It is important to recognise that while it is true that the greater the resolution the more information can be captured, even a system with a resolution of a single bit can still deliver information that can be useful.

Speech Identification

Speech is an analogue signal produces by a speech production system. In the case of human speech, our bodies have complex organs that work together to produce the sounds of language (Van Lanckert, 1987).

Voiced speech, which the project will be dealing, is nearly periodic in waveform, which means that its pattern of signal propagation looks to behave in a sinusoidal repeating way, this therefore also means that there would be a frequency and harmonics associated with voice produced speech (Arul, 2015).

This characteristic of voiced speech would make it possible to discriminate between un-voiced speech, which lacks the nearly periodic wave form characteristic, and actual speech.

Speech Processing

After the analogue voice signal is sampled and digitised, speech can be identified by searching for patterns of periodicity within the signal. The signal must be analysed for the important properties for speech recognition (Arul, 2015).

The speech recognition process can be split into three distinct phases

Listening: The audio signal is analysed for any presence of speech. If speech is detected the information is moved through to the second phase which works to describe the information being analysed. Digital filters may be included to reduce the signal-to-noise ratio for better processing (Arul, 2015).

Processing: One common method of processing speech information is by first splitting the signal into small segments, usually into segments of 10ms to 30ms each and to analyse each segment using mathematical models to determine the segments information (Arul, 2015). In this short period of time, the signal segment can be treated as a statistically stochastic process, which is a process that is defined by a collection of random variables. This therefore makes it possible to analyse each segment in isolation without affecting the rest of the segments by the use of a mathematical model (Arul, 2015)..

Matching: This phase happens after processing has been done. It compares the results attained from the processing phase to information existing in a pre-defined database (Arul, 2015).

Overview of Voice Detection Technology

A voice system requires a reference template that can be used to compare input audio to perform voice identification. To construct the template a speaker must train the system by speaking set phrases several times so the system can build and keep a template. Voice recognition systems use variables in the construction of the template including the speakers pitch, vocal dynamics and audio waveform pattern (Gbadamosi, 2013).

There are two main types of voice identification systems

1. Text dependent

   These are systems that remember a word or phrase and is used as the reference upon which a speaker's voice input is compared.
2. Text independent

   These are systems that are trained to recognize spoken words without a set phrase. These types of systems need more input data from a speaker to compare vocal characteristics like a speaker pitch and other individual vocal variations

The most developed accessible voice identification technology is made by corporations that have spent many years in developing and researching the technology. Apple's 'Siri', Microsoft's 'Cortana', Amazon's 'Alexa' and the Google Assistant have all made a name for themselves as leading voice technologies. The Google Assistant is installed in millions of Android devices and other Google sponsored machines. Apple's 'Siri' also is a standard on Apple devices.

Benefits of voice identification systems

- provides hands-free control
- flexibility
- saves time on data input
- removes the possibility of spelling errors

Overview of the Raspberry Pi

This project will make use of the Raspberry Pi 3 Model B, which is a new model following the Raspberry Pi 2 Model B. It is an affordable microprocessor unit specially made for learners and makers, which makes it easier to learn about the integration of software and hardware without considering a potential high cost.

The Raspberry Pi has a large online community and various online resources which can help in learning.

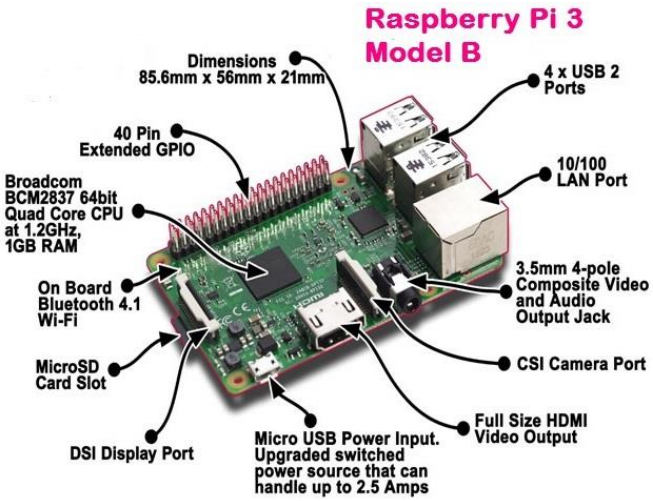

Figure 1 - Raspberry Pi Board (https://binaryupdates.com/introduction-of-raspberry-pi-3-model-b/)

Feature set from official website (https://www.raspberrypi.org/products/raspberry-pi-3-model-b/):

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
- 100 Base Ethernet
- 40-pin extended GPIO
- 4 USB 2 ports
- 4 Pole stereo output and composite video port
- Full size HDMI
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A

Benefits of the Raspberry Pi

Since the Raspberry Pi was first distributed, there have been several ways that people have used the board. Making use of the size, portability, cost, programmability and connectability of the board people have made a large variety of interesting and useful projects. From people making games and constructing robots from the Raspberry Pi to making mini-weather stations in their backyard, The Raspberry Pi has demonstrated its use as a reliable base to build a project while learning how computers work.

This project will be making use of the Raspberry Pi to make a voice identification system as said before.

Implementation

It is the purpose of this project to create a voice identification system from open source tools.

There were a few speech identification platforms that were discovered. Most open source voice identification platforms provide a speech-to-text service that transforms sounds into text that can then be used to compare to a list of 'known' words. Once the words are successfully compared, the platform would then follow with the response that was programmed into the system that is triggered by the word.

It was decided to select a voice recognition platform as a base for the project because of the highly complex mathematical algorithms used in audio manipulation through code. It was judged that coding a platform would require more time that provided for the duration of the project. Using an open source platform that has already been tested and performance verified was seen to be a suitable option for the purposes of the project. Only the commands and the functions the commands call would be programmed from scratch, thus the platform chosen must be able to allow the user to add commands and program the functions that the commands would trigger relatively easily.

Voice Identification Platforms

These are the following platforms tested

- Jasper
  Jasper is an open source platform for developing 'always-on' voice-controlled applications released under the MIT license

- DeepSpeech
  DeepSpeech is an open source Speech-To-Text engine made by Mozzila, using a model trained by machine learning techniques

- OpenJarvis
  Jarvis is an ultra-light and multilingual voice assistant. Designed for home automation, it is easy to use. It lets the user choose the speech recognition and speech synthesis engine.

These are the platforms chosen to be the back bone of the voice identification system. They were each installed and tested for ease of use and functionality. Though all were demonstrated to function appropriately, OpenJarvis was chosen for its expanded scope on adaptability as it could be directly trained by its onboard selection of speech engines and synthesizers and adding commands to the system was made accessible without having to access the code, which makes it a dynamic option.

Materials

Raspberry Pi Model B with Raspbian OS

OpenJarvis

USB microphone

Breadboard

Jumper Cables

LED

220 ohm Resistor

Raspberry Pi Setup

The Raspberry Pi microSD card was loaded with the Raspbian operating system, which is a Linux based operating system made specially to work on Raspberry Pi. The size of the micoSD was 8GB. This size was enough to hold the operating system and all the applications needed to put the system together.

To get into the Raspberry Pi and start importing all the needed software packages and writing the needed software, the Raspberry Pi needed to be set up to a display. But the Raspberry Pi circuit board came by itself so it was needed to set up the Raspberry Pi for remote access.

Setup took a while to complete though in following the instruction on (web3) for a headless Raspberry Pi using SSH commands with 'PuTTY', which is a terminal emulator, setup was completed.

Loading OpenJarvis

After it was possible to enter the Raspberry Pi Raspbian system, A voice identification platform was needed to be installed into the Raspberry Pi.

OpenJarvis was settled on because of its setup procedure that was experienced as simple and much easier to understand. Jarvis also had accessible voice training options that suited the project.

The installation steps can be found on the OpenJarvis GitHub page (https://github.com/alexylem/jarvis)

After installation of OpenJarvis, it had to be configured to identify key words that would direct the voice identification system to perform certain actions. This required a routing mechanism to enable the voice command, when recognized, to route to some other system that would respond to the received command and run software for the Raspberry Pi to perform actions.

API Server Setup

The API server was written in Python and uses the lightweight library called Flask. Python was chosen as the language because it is the coding language the RaspbianOS is most suited for. Using any other system could have cause compatibility issues that have not yet been foreseen. Python tutorials were heavily relied on to get my python skills to where I could write the code to construct the server.

Voice Training

Jarvis has an internal method for voice training, this has been very useful in programming 'words' that the voice recognition system will respond to as opposed to directly interacting with the code itself.

'Start', 'Stop', 'On', 'Off' and 'Hello' were the list of words chosen to for the system to be able to recognize. Training the platform to recognize the words was simple. Adding new words to the system is made possible by the accessible command line interface that OpenJarvis utilizes.

Response Mechanism

A single LED was used a test output device. The LED was connected to the GND and GPIO18 pins, GPIO18 was set up as an output pi. Please refer to Appendix A for the code used to set up the RPi pin as an output pin. The LED will be turned on by a server function after the call to the server is made by the voice system.

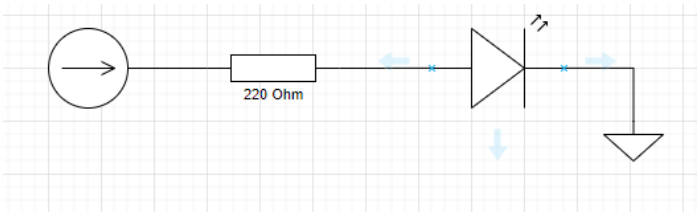

Figure 2 - Output Circuit

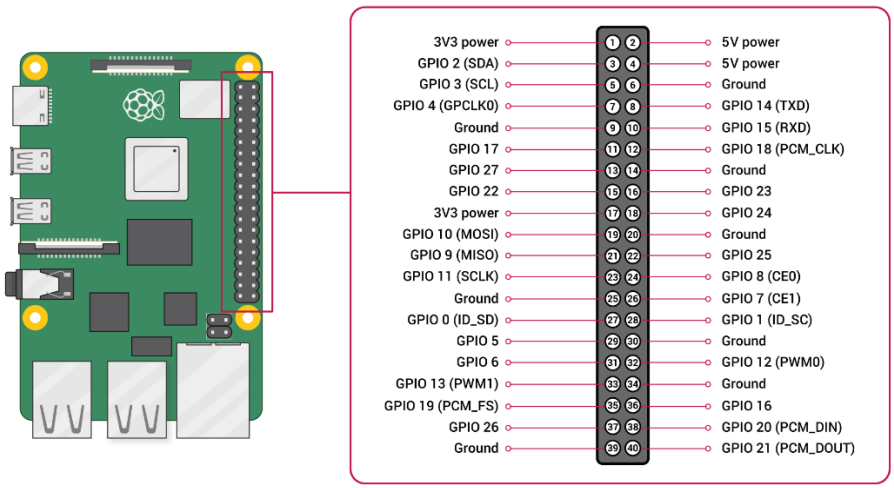

Figure 3 - GPIO-Pinout-Diagram-2 (https://www.raspberrypi.org/documentation/usage/gpio/)

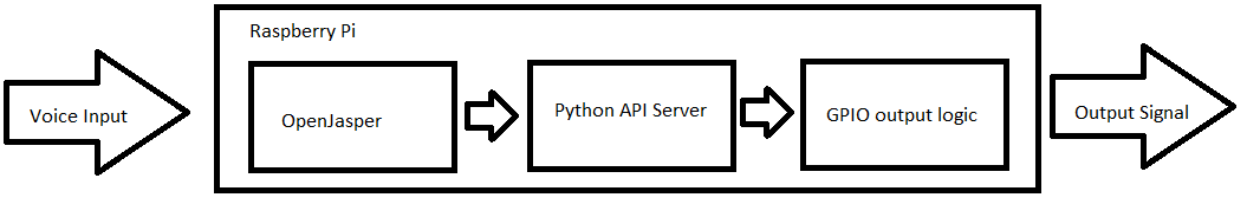

Figure 4 - Overall System Diagram

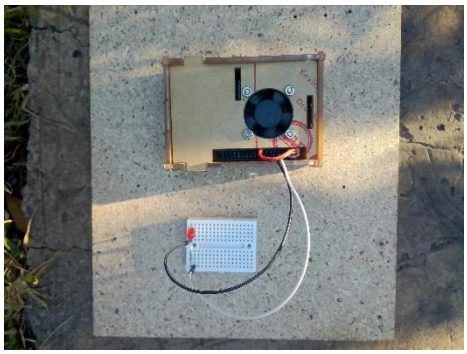

Figure 5 - RPi with LED circuit

Results

The voice system worked as expected. When the command to turn on the LED was detected, the call to the API server was made and the function to put GPIO18 to a digital high was run and the LED lit up. The same for when the command to turn off the LED was detected.

```
Jarvis: Waiting to hear 'Jarvis'
Philani: Jarvis
Jarvis: Hi
Philani: start
Jarvis: Operation: null
Philani: off
Jarvis: Turning off
Philani: (timeout)
Jarvis: Waiting to hear 'Jarvis'
Philani:
```

Figure 6 - OpenJarvis command line interface test responses

```
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
Turned On
127.0.0.1 - - [16/May/2020 20:01:42] "GET /turnon HTTP/1.1" 200 -
Turned Off
127.0.0.1 - - [16/May/2020 20:01:52] "GET /turnoff HTTP/1.1" 200 -
```

Figure 7 - API server test response

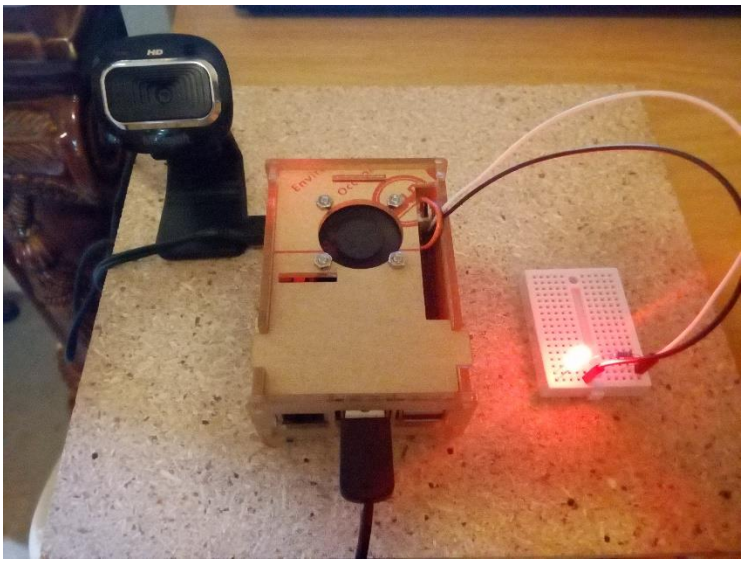

Figure 8 - LED Turned ON

Figure 9 - LED Tuned OFF

Conclusion

In conclusion, the voice Identification system was implemented successfully.

Applications and Further Development

Although the output device for this project was a simple LED, it serves as demonstration that the correct signal can be sent out by the RPi when the appropriate command is identified. The immediate application for this project can be a voice controlled light source but this project has the potential to be a simple implementation of a voice system for home automation. It is not advised to use this project as it is to serve as a security implementation as project doesn't include safety features that would be necessary to make it a secure system.

OpenJarvis can handle several different commands and actions, the developed Python API server that is hosted internally in the Raspberry Pi can be further developed and extended to include further functions.

References

Voice Identification. 2020. Voice Identification. [ONLINE] Available at: http://www.di-srv.unisa.it/~ads/corso-security/www/CORSO-9900/biometria/Voice.htm. [Accessed 12 May 2020].

Ambardar, A., 2007. *Digital Signal Processing: A Modern introduction*. India: Cengage Learning.

Van Lanckert, D., 1987. VOICE DISCRIMINATION AND RECOGNITION ARE SEPARATE ABILITIES. *Neuropsychologia,*, 25, 829 – 834

Omyonga, K., 2015. The Application of Real-Time Voice Recognition to Control Critical Mobile Device Operations. *International Journal of Research Studies in Science, Engineering and Technology*, 2 (7), 174-184

Arul, V.H., Marimuthu, R., 2015. A Study on Speech Recognition Technology. *Journal of Computing Technologies*, 3 (7), 2278 – 3814

D'Souza, M., Joshi, S., Kumari, A., Pooja, P., Sangaonkar, S., 2019. Voice Recognition System. *Journal for Research*, [Online]. 3 (1/002), 6-9. Available at: www.journal4research.org [Accessed 16 April 2020].

Gbadamosi, Luqman. (2013). Voice Recognition System using Template Matching. International Journal of Research in Computer Science. 3. 13-17. 10.7815/ijorcs.35.2013.070.

web1: Raspberrypi.org. 2020. [online] Available at: <https://www.raspberrypi.org/> [Accessed 01 April 2020].

web2: Index of /raspbian_images/raspberrypi/images/raspbian/2019-04-08-raspbian-stretch. 2020. Index of /raspbian_images/raspberrypi/images/raspbian/2019-04-08-raspbian-stretch. [ONLINE] Available

at: http://debian.rutgers.edu/raspbian_images/raspberrypi/images/raspbian/2019-04-08-raspbian-stretch/. [Accessed 13 April 2020].

web3: Circuit Basics. 2020. How to Access the Raspberry Pi Desktop with a Remote Desktop Connection. [ONLINE] Available at: https://www.circuitbasics.com/access-raspberry-pi-desktop-remote-connection/. [Accessed 13 April 2020].

web4: balenaEtcher. 2020. balenaEtcher - Flash OS images to SD cards & USB drives. [ONLINE] Available at: https://www.balena.io/etcher/. [Accessed 13 April 2020].

web5: Download PuTTY - a free SSH and telnet client for Windows. 2020. Download PuTTY - a free SSH and telnet client for Windows. [ONLINE] Available at: https://www.putty.org/. [Accessed 13 April 2020].

web6: Présentation | Jarvis. 2020. Présentation | Jarvis. [ONLINE] Available at: http://openjarvis.com/. [Accessed 13 April 2020].

web7: Voice recognition using Raspberry Pi 3. 2020. Voice recognition using Raspberry Pi 3. [ONLINE] Available at: https://www.rhydolabz.com/wiki/?p=16234. [Accessed 13 April 2020].

web8: GitHub. 2020. GitHub - StevenHickson/PiAUISuite: Raspberry PI AUI Suite. [ONLINE] Available at: https://github.com/StevenHickson/PiAUISuite. [Accessed 13 April 2020].

web9: GitHub. 2020. GitHub - mozilla/DeepSpeech: A TensorFlow implementation of Baidu's DeepSpeech architecture. [ONLINE] Available at: https://github.com/mozilla/DeepSpeech/. [Accessed 13 April 2020].

web10: Maker Pro. 2020. The Best Voice Recognition Software for Raspberry Pi | Raspberry Pi | Maker Pro. [ONLINE] Available at: https://maker.pro/raspberry-pi/tutorial/the-best-voice-recognition-software-for-raspberry-pi. [Accessed 13 April 2020].

web11: Steves Computer Vision Blog: Voice Control on the Raspberry Pi. 2020. Steves Computer Vision Blog: Voice Control on the Raspberry Pi. [ONLINE] Available

at: https://stevenhickson.blogspot.com/2013/04/voice-control-on-raspberry-pi.html. [Accessed 13 April 2020].

web12: Jasper | Documentation. 2020. Jasper | Documentation. [ONLINE] Available at: https://jasperproject.github.io/documentation/installation/. [Accessed 13 April 2020].

web13: Snowboy Hotword Detection. 2020. Snowboy Hotword Detection. [ONLINE] Available at: https://snowboy.kitt.ai/hotword/29. [Accessed 13 April 2020].

web14: GitHub. 2020. GitHub - alexylem/jarvis: Jarvis.sh is a simple configurable multi-lang assistant.. [ONLINE] Available at: https://github.com/alexylem/jarvis. [Accessed 13 April 2020].

web15: GPIO in Python - Raspberry Pi Documentation. 2020. GPIO in Python - Raspberry Pi Documentation. [ONLINE] Available at: https://www.raspberrypi.org/documentation/usage/gpio/python/README.md. [Accessed 13 April 2020].

web16: Opensource.com. 2020. What is a Raspberry Pi? | Opensource.com. [ONLINE] Available at: https://opensource.com/resources/raspberry-pi. [Accessed 24 May 2020].

Appendix

Appendix A: Python Server Code

```
from flask import Flask, jsonify
import RPi.GPIO as GPIO
import time

#RPi gpio setup
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(18, GPIO.OUT)

app = Flask(__name__)

@app.route('/')
def hello_world():
        return 'Hello World!'

@app.route('/turnon')
def turn_on():
        try:
                GPIO.output(18, GPIO.HIGH)
                print('Turned On')
                return jsonify(LED = 'Turned On')
        except:
                print('some error happened')
                return jsonify(ERROR= 'something happened')

@app.route('/turnoff')
def turn_off():
        try:
                GPIO.output(18, GPIO.LOW)
                print('Turned Off')
                return jsonify(LED = 'Turned Off')
        except:
                print('some error happened')
                return jsonify(ERROR= 'something happened')

app.run(port=5000)
```

Appendix B: System Command Map

```
*MENU*==jv_display_commands
*HELLO*|*HOWZIT*==say "Hey $username"
*ON*|*START*==say "Operation: $(curl -s "http://127.0.0.1:5000/turnon" | jq -r $
*OFF*|*STOP*==say "Turning off" && jv_curl "http://127.0.0.1:5000/turnoff"
*THANKS*|*THANK*YOU*==say "You're welcome"
*GOOD*BYE*|*BYE*==say "Goodbye, $username"; jv_exit
*REPEAT*==jv_repeat_last_command
*VERSION*==say "My version is $jv_version"
```