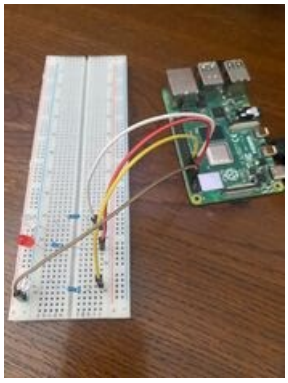
Burbulea Ioana-Georgiana

Title : SiriControl

Elevator pitch: Using SiriControl module with a Raspberry Pi 4 to turn on and off 3 LEDs.

Cover image:



Story: It is an interesting project because the LEDs are controlled by Siri voice commands. This functionality can be extended, in the sense that a multitude of voice commands can be added with which we can control the raspberry Pi. This is a basic project that introduce us in this technology.

The voice commands used in this project are:

-turn on: all 3 LEDs will turn on;

-turn off: all 3 LEDs will turn off;

-red led on :the red LED is on;

-red led off:the red LED is off;

-yellow led on: the yellow LED is on;

-yellow led off:the yellow LED is off;

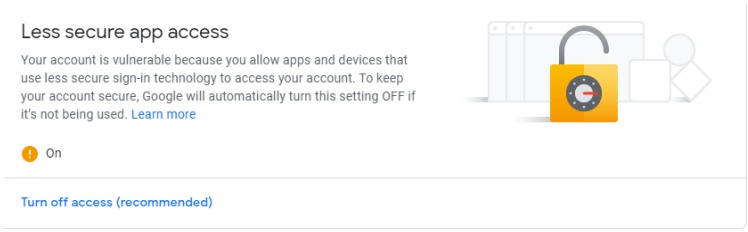
-white led on: the white LED is on;

-white led off:the white LED is off;

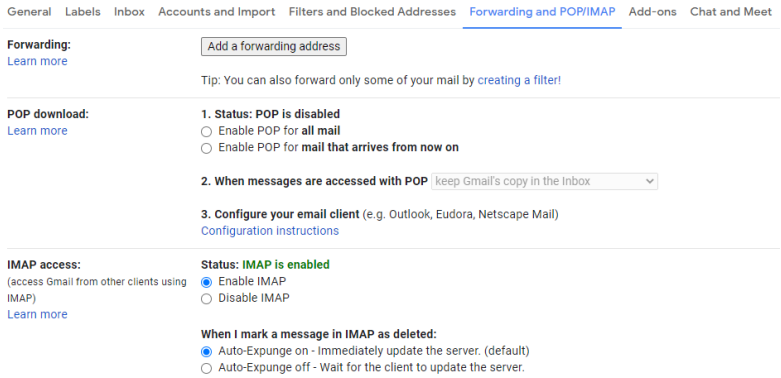
Steps

A. Create a Gmail account.

The SiriControl module requires a Gmail account to work. I set up a new Gmail account just for use with SiriControl. Change 2 settings:

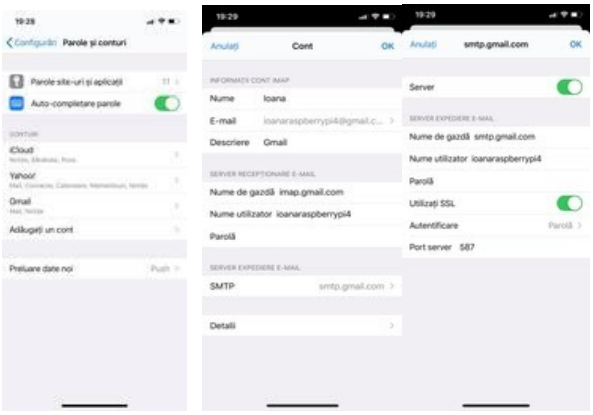


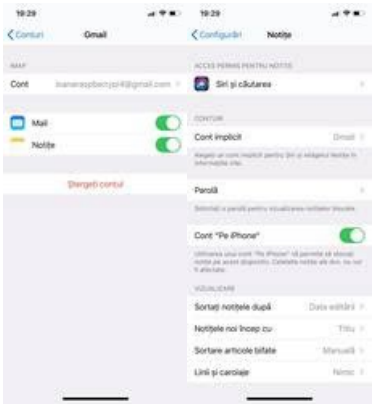
IMAP must be enable for accessing the Gmail address by other devices.



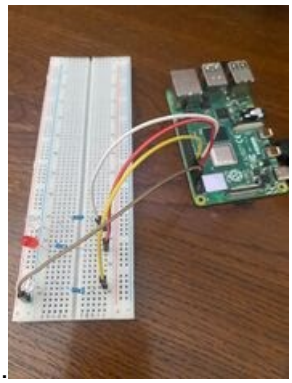
B.Setup the iOS device.

Connect “Notes” on iOS device to the Gmail account set up to be used with SiriControl. Connect “Notes” on your iOS device to the Gmail account set up to be used with SiriControl. Go to **Settings->Accounts & Passwords->Add Account** and add the Gmail account . After adding that account, select it and enable notes. Next, I go to **Settings->Notes** and enable “**On My iPhone” Account**. I then change my **Default Account** to the Gmail Account.





C. Wiring the circuit**.**



D. Write and run the code.

To run the code execute **python siri.py** command**.**

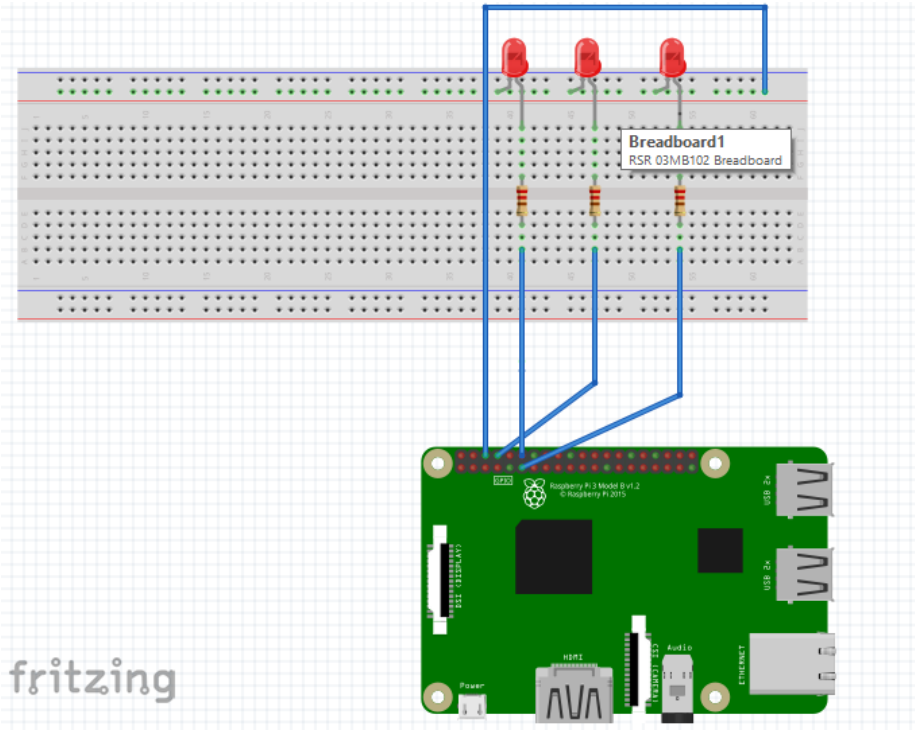
5.Hardware components:

- Raspberry Pi 4 model B/4GB;
- USB type C;
- jumpers;
- 3 x resistor(470Ω);
-3 LEDs;
-wireless mouse and keyboard, microHDMI-HDMI cable and a display (to setup the Raspberry Pi 4 and use it);
- 830p breadboard.

Software:

-Apple Siri;
-Python;
-Raspbian OS.

6.Schematics:



Directory schema:

Project

siri.py

modules

LED_on.py

LED_off.py

LED_red_on.py

LED_red_off.py

LED_yellow_on.py

LED_yellow_off.py

LED_white_on.py

LED_white_off.py

7.Code:

siri.py

```
 import time
import imaplib
import email
import os
import pkgutil
```

```
username = "ioanaraspberrypi4@gmail.com"
password = "raspberrypi4"

class ControlException(Exception):
    pass

class Control():
    def __init__(self, username, password):
        try:
            self.last_checked = -1
            self.mail = imaplib.IMAP4_SSL("imap.gmail.com", 993)
            self.mail.login(username, password)
            self.mail.list()
            self.mail.select("Notes")

            result, uidlist = self.mail.search(None, "ALL")
            try:
                self.last_checked = uidlist[0].split()[-1]
            except IndexError:
                pass

            self.load()
            self.handle()
  except imaplib.IMAP4.error:
            print("Your username and password is incorrect")
            print("Or IMAP is not enabled.")

    def load(self):
        """Try to load all modules found in the modules folder"""
        print("\n")
        print("Loading modules...")
        self.modules = []
        path = os.path.join(os.path.dirname(__file__), "modules")
        directory = pkgutil.iter_modules(path=[path])
        for finder, name, ispkg in directory:
            try:
```

```
            loader = finder.find_module(name)
            module = loader.load_module(name)
            if hasattr(module, "commandWords") \
                  and hasattr(module, "moduleName") \
                  and hasattr(module, "execute"):
self.modules.append(module)
               print("The module '{0}' has been loaded, "
                   "successfully.".format(name))
            else:
               print("[ERROR] The module '{0}' is not in the "
                   "correct format.".format(name))
         except:
            print("[ERROR] The module '" + name + "' has some errors.")
      print("\n")

   def fetch_command(self):
      """Retrieve the last Note created if new id found"""
      self.mail.list()
      self.mail.select("Notes")

      result, uidlist = self.mail.search(None, "ALL")
      try:
         latest_email_id = uidlist[0].split()[-1]
except IndexError:
         return

      if latest_email_id == self.last_checked:
         return

      self.last_checked = latest_email_id
      result, data = self.mail.fetch(latest_email_id, "(RFC822)")
      voice_command = email.message_from_string(data[0][1].decode('utf-8'))
      return str(voice_command.get_payload()).lower().strip()

   def handle(self):
      """Handle new commands

      Poll continuously every second and check for new commands.
```

```
        """
        print("Fetching commands...")
        print("\n")

        while True:
            try:
                command = self.fetch_command()
                if not command:
                    raise ControlException("No command found.")

                print("The word(s) '" + command + "' have been said")
                for module in self.modules:
                    foundWords = []
                    for word in module.commandWords:
                        if str(word) in command:
                            foundWords.append(str(word))
                    if len(foundWords) == len(module.commandWords):
                        try:
                            module.execute(command)
                            print("The module {0} has been executed "
                                  "successfully.".format(module.moduleName))

                        except:
                            print("[ERROR] There has been an error "
                                  "when running the {0} module".format(
                                      module.moduleName))
                    else:
                        print("\n")
            except (TypeError, ControlException):
                pass
            except Exception as exc:
                print("Received an exception while running: {exc}".format(
                    **locals()))
                print("Restarting...")
            time.sleep(1)

if __name__ == '__main__':
    Control(username, password)
```

LED_on.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName = "LED_on"
commandWords = ["turn", "on"]

def execute(command):
 GPIO.setwarnings(False)
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(8, GPIO.OUT)
 GPIO.setup(11,GPIO.OUT)
 GPIO.setup(12,GPIO.OUT)
 GPIO.output(8, GPIO.HIGH)
 GPIO.output(11, GPIO.HIGH)
 GPIO.output(12, GPIO.HIGH)
```

LED_off.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName= "LED_off"
commandWords = ["turn", "off"]
def execute(command):
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(8, GPIO.OUT, initial=GPIO.HIGH)
 GPIO.output(8, GPIO.LOW)
 GPIO.setup(11, GPIO.OUT, initial=GPIO.HIGH)
 GPIO.output(11, GPIO.LOW)
 GPIO.setup(12, GPIO.OUT, initial=GPIO.HIGH)
 GPIO.output(12, GPIO.LOW)
```

LED_red_on.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName = "LED_on"
commandWords = ["red", "led", "on"]

def execute(command):
 GPIO.setwarnings(False)
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(8, GPIO.OUT)
 GPIO.output(8, GPIO.HIGH)
```

LED_red_off.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName = "LED_on"
commandWords = ["red", "led", "on"]

def execute(command):
 GPIO.setwarnings(False)
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(8, GPIO.OUT)
 GPIO.output(8, GPIO.HIGH)
```

LED_yellow_on.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName = "LED_on"
commandWords = ["yellow", "led", "on"]

def execute(command):
 GPIO.setwarnings(False)
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(12, GPIO.OUT)
```

```
 GPIO.output(12, GPIO.HIGH)
```

LED_yellow_off.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName= "LED_off"
commandWords = ["yellow", "led", "off"]
def execute(command):
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(12, GPIO.OUT, initial=GPIO.HIGH)
 GPIO.output(12, GPIO.LOW)
```

LED_white_on.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName = "LED_on"
commandWords = ["white", "led", "on"]

def execute(command):
 GPIO.setwarnings(False)
 GPIO.setmode(GPIO.BOARD)
 GPIO.setup(11, GPIO.OUT)
 GPIO.output(11, GPIO.HIGH)
```

LED_white_off.py

```
import RPi.GPIO as GPIO
from time import sleep

moduleName = "LED_on"
commandWords = ["white", "led", "on"]

def execute(command):
```

```
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)
GPIO.output(11, GPIO.HIGH)
```

https://www.hackster.io/ioana-burbulea/siricontrol-aab3ef

SiriControl-youtube