SAVITRIBAI PHULE PUNE UNIVERSITY
A MINI-PROJECT REPORT ON

# Voice Controlled Appliances

**SUBMITTED TOWARDS THE**

**PARTIAL FULFILLMENT OF THE REQUIREMENTS OF**

**THIRD YEAR SEMESTER II OF ENGINEERING**
**(Computer Engineering)**
**BY**

**AJIT HANDE 31422**
**YASH HIRAN 31423**
**ATHARV JAGTAP 31425**
**NIKHIL JAIN 31427**
**KESHAV KADAM 31428**

**Under the guidance of**

**Prof. B.P. MASRAM**

**DEPARTMENT OF COMPUTER ENGINEERING**

**Pune Institute of Computer Technology Dhankawadi,**
**Pune-411043**
**Academic Year 2019-20**

**Pune Institute of Computer Technology**

**DEPARTMENT OF COMPUTER ENGINEERING**
**CERTIFICATE**

**This is to certify that the Mini-Project Entitled**

# Voice Controlled Appliances

**Submitted by**

**AJIT HANDE 31422**
**YASH HIRAN 31423**
**ATHARV JAGTAP 31425**
**NIKHIL JAIN 31427**
**KESHAV KADAM 31428**

**It is a bonafide work carried out by Students under the guidance of Prof. Bhumesh P. Masram and it is submitted towards the partial fulfillment of the requirement of Third Year Computer Engineering Semester II of Savitribai Phule Pune University.**

**Prof. Bhumesh Masram**
**Internal Guide**
**Dept. of Computer Engg.**

**Date :**

**Place**

# Contents

# VOICE CONTROLLED APPLIANCES

## Abstract:

The Internet of Things is emerging technology as the third wave in the development of the internet. The Internet of Things (IoT) is expected to have a massive impact on consumer products, business, wider culture, but these are still early days. It has potential for very wide applicability to almost all verticals and aspects of business, industries, manufacturing, consumer goods, supply chains, etc. IoT as The whole is a very broad area.

This project focuses specifically to design a voice controlled home automation system which makes operating electrical appliances in home through android application and set up the controlling actions in the mobile. The Bluetooth technology devices are used for home automation in a cost effective manner .

**Keywords**— Android application, Arduino UNO, Bluetooth Module, IoT.

# 1 INTRODUCTION

Internet of Things (IoT) heralds a vision of the future Internet through a network where connecting physical things, will let them take an active part in the Internet, exchanging information about themselves and their surroundings. This will give immediate access to information about the objects and the physical world in it leading to innovative services and increase in efficiency and productivity.

Home automation is housework or household activity. Home automation may include centralized control of lighting, HVAC (heating, ventilation and air conditioning), appliances, and other systems, to provide improved convenience, comfort, energy efficiency and security.

## 1.1 Motivation

Home automation for the elderly and disabled can provide increased quality of life for persons who might otherwise require institutional care.As per survey currently there exists no system at cheaper rates. Various systems are hard to install, difficult to use and maintain. Current systems are generally proprietary and closed, not very customizable by the end user. This served as the motivation for making a mini project of voice controlled appliances for home automation.

## 1.2 Purpose

Introducing a automatic voice command based home automation that can control various household appliances as well as some other tasks that constitute the home automation system .So that we can decrease the effort for doing the daily routine work and make life more easier and comfortable.

# 2 PROBLEM STATEMENT AND SCOPE

## 2.1 Problem Statement

To develop an automatic voice command based home automation that can control various household appliances.

## 2.2 Scope

The scope of the project is: The basic appliances like tube light and fan can be controlled by the user through his mobile phone .User can switch on fan, tubelight and switch it off.

# 3. Design and Analysis of System

## 3.1 Existing systems

Appliances in the household are provided power using switches. These switches are manual. Thus, these switches need to be operated manually. There are different types of switches according to the power rating of the appliance. Usually the appliances are equipped with an adapter to stabilise the source. When it comes to switching off the appliances, you need to be there physically to turn it on and off. There are various different types of appliances which are located in different parts of  the house.

## 3.2 Problems in existing systems

- Be physically present
- Cannot be operated remotely
- Zero automation
- Tedious operation

## 3.3 Design of newer system

Each system will have a central controller. The controller is connected to I/O devices. The controller uses the data from the input device and performs a specific action according to the analysis of the data. The microphone will convert the audio into text form and extract keywords. After the analysis of these words, the specific action will be performed on the output device. The component diagram can be represented as in figure below.

### 3.3.1 Module Information

Following modules have been included-

1. Microphone
2. Raspberry Pi 3
3. Jumper wires
4. Android application
5. LED’s, etc.

6. PC
7. Relays

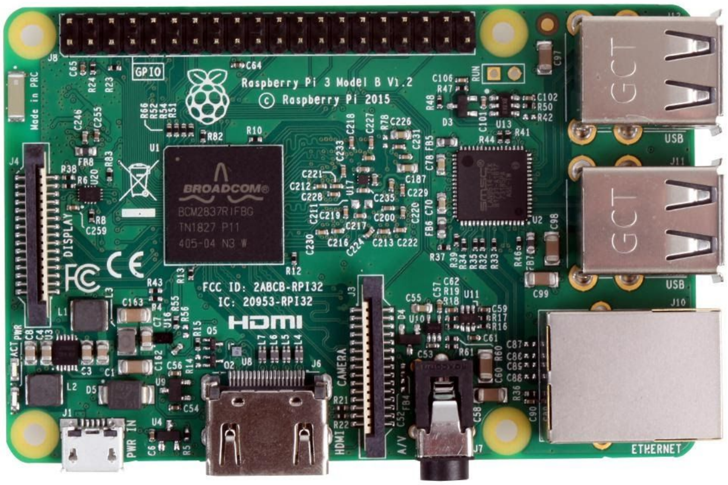
### 3.3.2 Module Description

**1. Microphone**

The microphone aka mic is used as an input device. The mic continuously listens for audio signals. The input from this input device is converted into text and this text is later used for data analysis to perform some action.

**2. Raspberry Pi 3**

Raspberry Pi 3 is the third generation of raspberry pi and is its earliest model. It has a 40 pin GPIO and is equipped with 4 USB 2 ports, full size HDMI, micro sd for loading operating system and storing data. It has micro usb source upto 2.5 A. The controller takes the input from the input device and performs the analysis of data and based on it triggers some action.



3. **Jumper Wires(Generic)**

Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of test equipment.

4. **Android application**

The android application will be used for remote access of the appliances.

5. **LED's, other devices**

LEDs are semiconductor light emitting sources which emit light when current flows through it. These devices will be used as appliances and act as output devices. They are connected using relays which act as switches.

6. **PC**

PC is used to dump code onto the controller and test the circuit.

7. **Relay**

Relays here are used as electrical switched to control the state of the appliances. These relays are controlled using the raspberry pi controller.

## 3.4 IoT Design Methodology Steps

### Step 1: Purpose and Requirement Specification

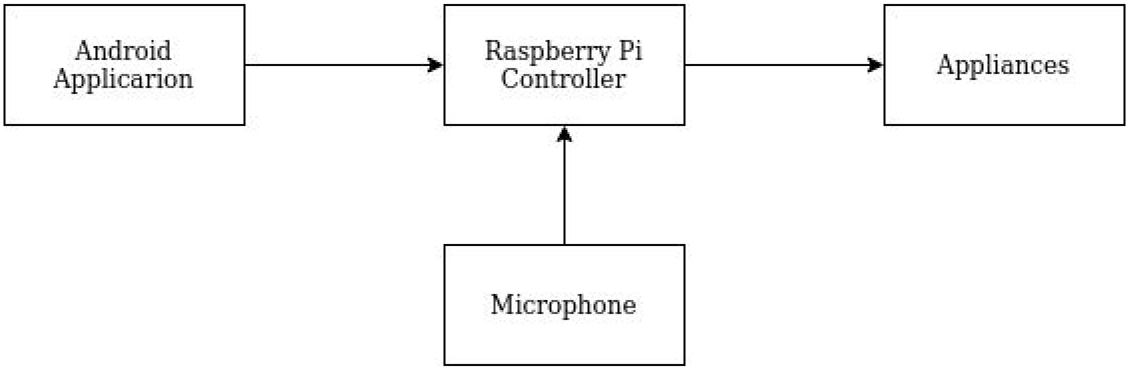
1. The purpose of the system is to automate the switching of the appliances.
2. The system should perform the following action as voiced by the user.

### Step 2: Process Specification

1. Every appliance is like an entity.
2. When the user speaks the name of the entity and its state, the controller has to analyse the input and trigger the action pronounced.
3. The output entity must change state according to input.

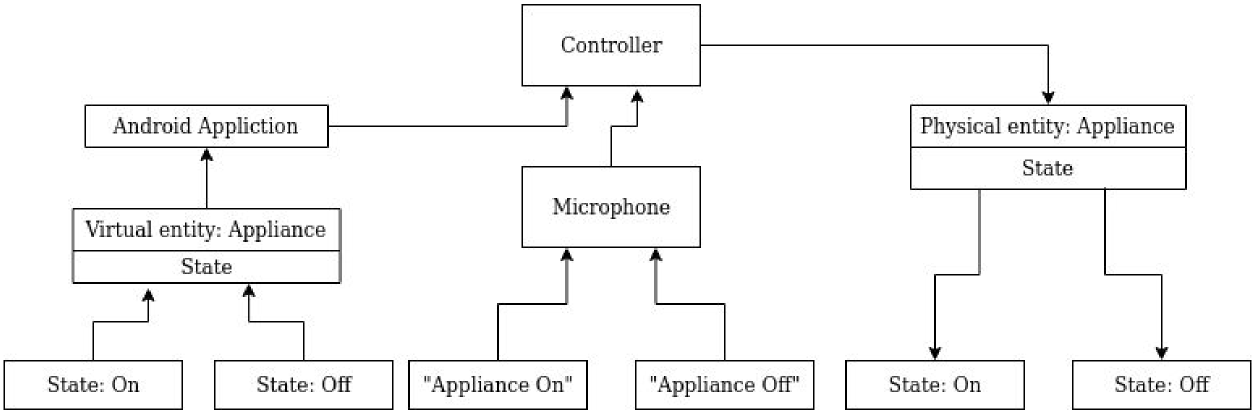
### Step 3: Domain Model Specification

1. There are 4 different entities in the system.
2. The different entities are an android application, the central controller, microphone, and appliances.



### Step 4: Information Model Specification

1. The data is generated by the input device-microphone and this data is sent to the controller.
2. The controller sends required signals to the appliances.
3. The data generated by the android application is also sent to the controller. Here the data generated is already processed, the controller performs the action required.



## Step 5: Service Specification

1. The microphone acts as the input device and listens for audio signals. The signals are then sent to the controller.
2. The application also generates data according to the selection and sends the data to the controller.
3. The controller then processes this data and generates required signals to change the state of the appliance.

## Step 6: IoT Level Specification

1. IoT level specification defines the deployment level of the system.
2. The system implemented is IoT Level 1 as it is confined to be used remotely.

## Step 7: Functional View Specification

1. The microphone listens for audio signals. The user speaks the name of the entity and its state.
2. The signal is sent to the controller. The signal is converted into text and is refined to generate keywords.
3. The controller then generates the appropriate signals to control the state of the appliance.

## Step 8: Operational View Specification

1. The microphone generates signals based on the user input.
2. The system does not store this data. The system only stores the metadata required for the functioning of the system.

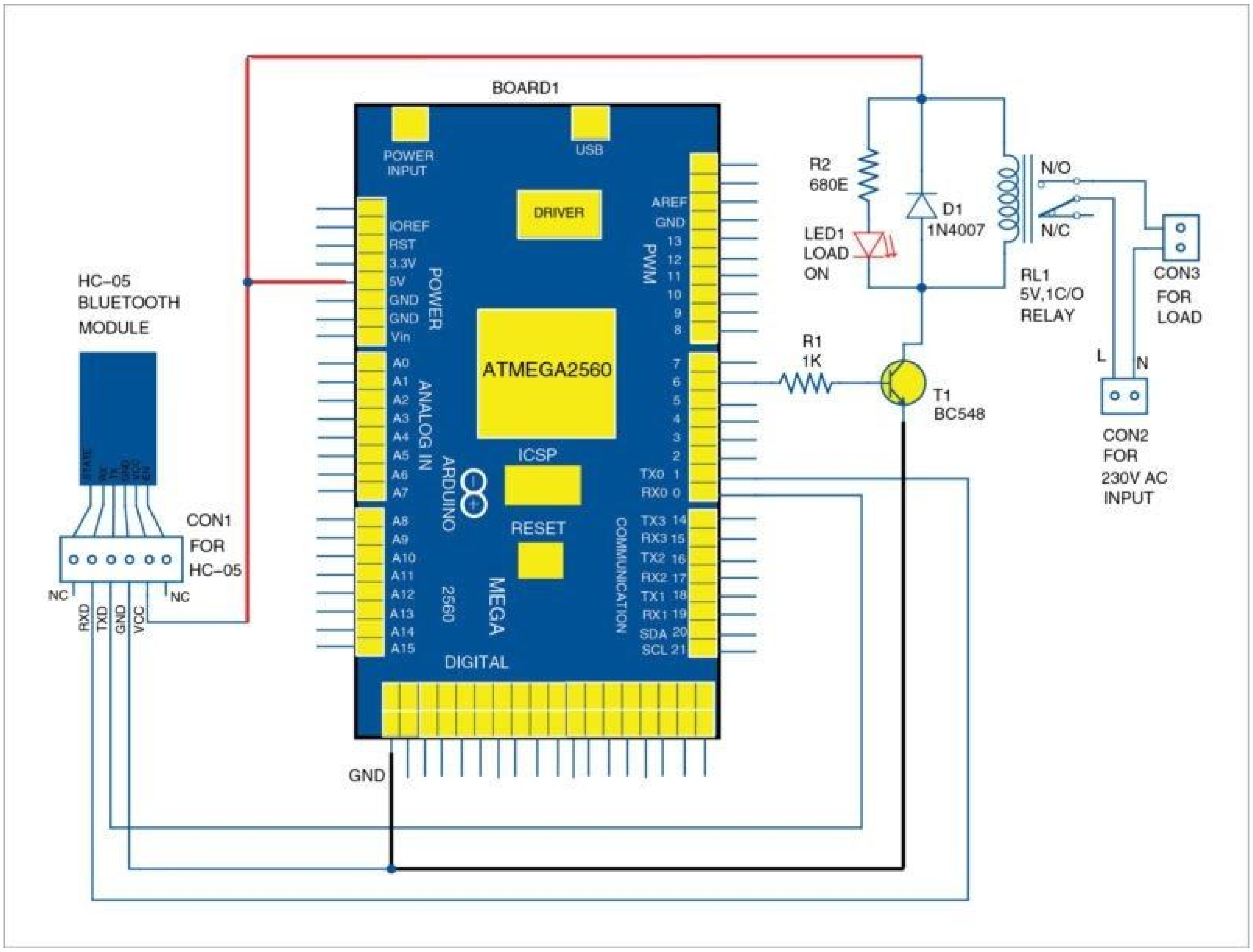
## Step 9: Device and Component Integration

1. The android application is connected wirelessly to the controller.
2. The microphone is connected to the controller using USB connection.
3. The appliance is connected in series through the relay using GPIO pins.
4. The relay acts as the electrical switch.

## Step 10: Application Development

1. The operating system raspbian is installed onto the controller.
2. The circuit is implemented according to the diagram.
3. Required libraries and APIs are included and the code is dumped onto the raspberry pi.

# Circuit Diagram



# Conclusion and Future Works

## 5.1 Conclusion

This project covers the most important feature, in which it could provide the complete smart home environment. The voice controlled home automation using Raspberry Pi is projected for the easy use and control of electronic devices by old age and disabled people. This project provides a basic system of home automation which can be easily implemented and used effectively. This system allows users to take decisions and to regulate the home appliances with the help of an android application , thus making one’s life comfortable and at the same time remotely accessible through portable devices like android phones.

## 5.2 Future Works

The future scope of this project is:

1. Authentication: In future use, we can give voice authentication to provide security. In this only authenticated persons' voices can access secured devices (like lockers).

2. Sensor: By using sensors we reduce the effort of declaring each and every device a particular name.
Example: If a person gives a command “lights on” the sensor will sense the person's location and only that light will get on.

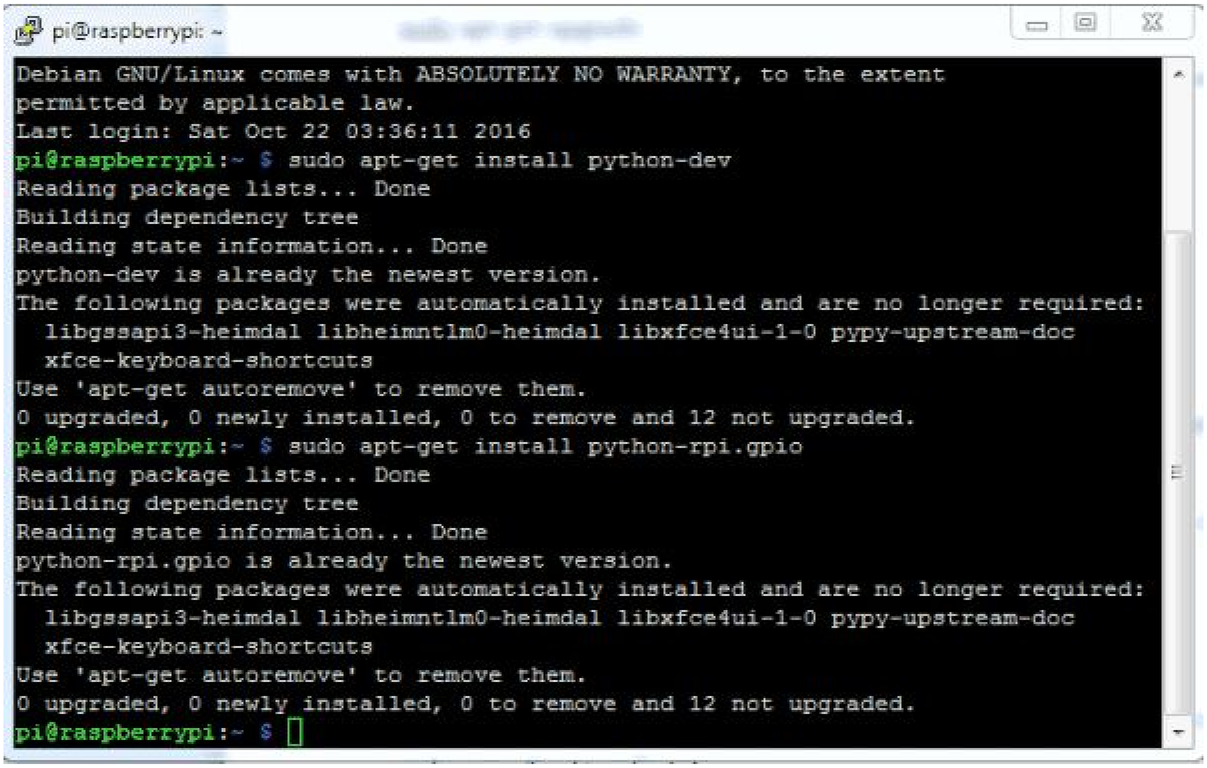
# REFERENCES

1. Bhavik Pandya, Mihir Mehta, and Nilesh Jain, “Android Based Home Automation System Using Bluetooth & Voice Command

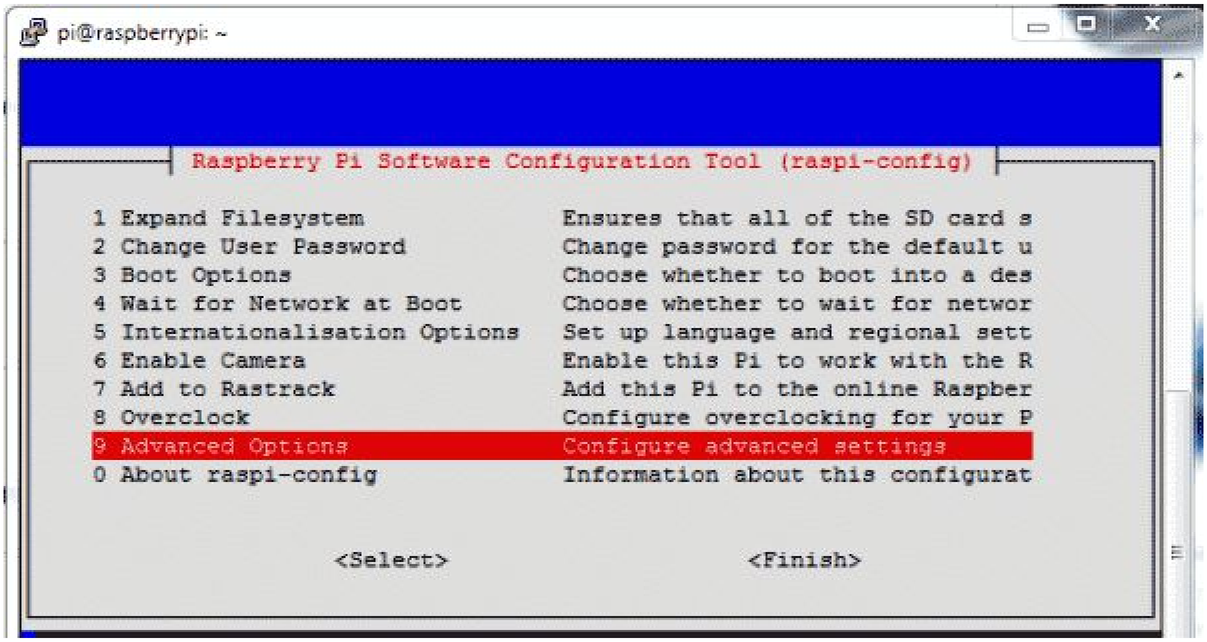
2. Faisal Baig, Saira Beg, and Muhammad Fahad KhanInternational, Controlling Home Appliances Remotely through Voice Command

3. Arduino and Matlab interfacing via Bluetooth module - http://crackeconcept.blogspot.in/2014/03/arduino-and-matlab-interfacing-via.html

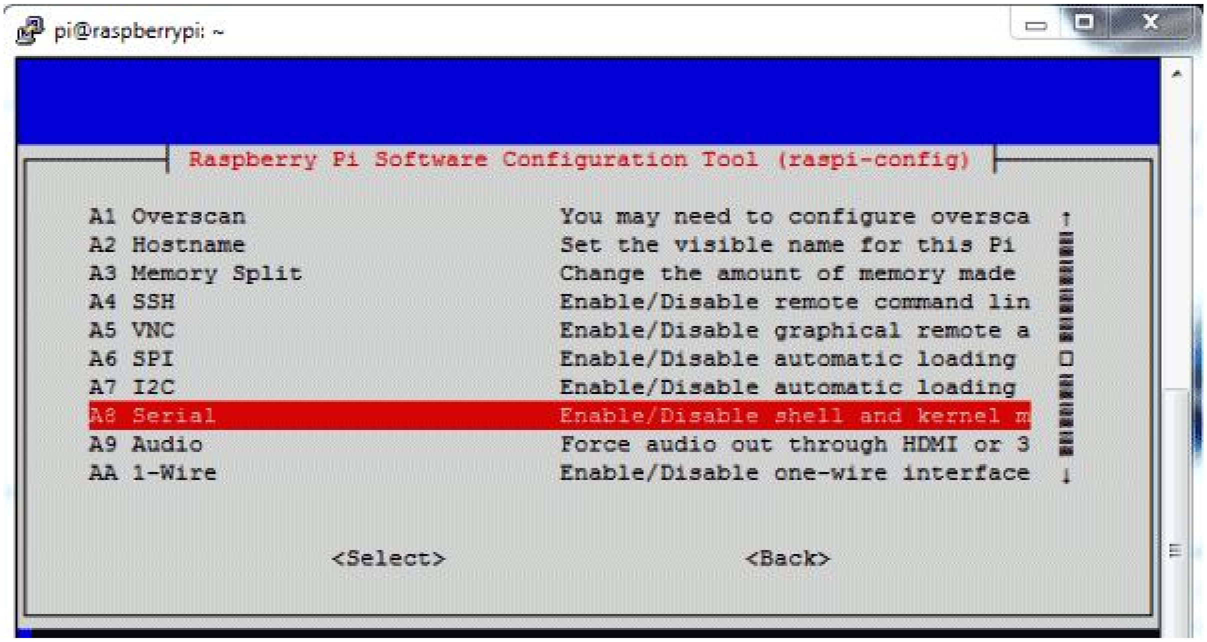
## Screenshot & Photos



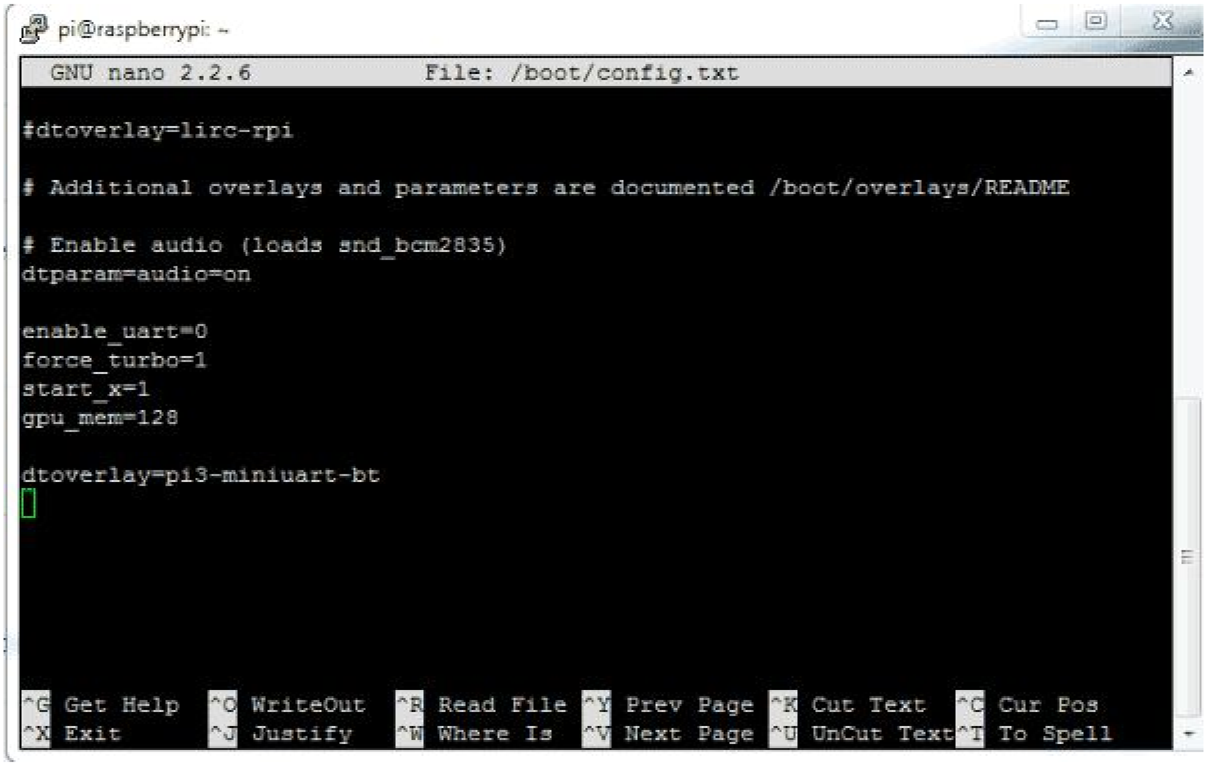
```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Oct 22 03:36:11 2016
pi@raspberrypi:~ $ sudo apt-get install python-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
python-dev is already the newest version.
The following packages were automatically installed and are no longer required:
  libgssapi3-heimdal libheimntlm0-heimdal libxfce4ui-1-0 pypy-upstream-doc
  xfce-keyboard-shortcuts
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
pi@raspberrypi:~ $ sudo apt-get install python-rpi.gpio
Reading package lists... Done
Building dependency tree
Reading state information... Done
python-rpi.gpio is already the newest version.
The following packages were automatically installed and are no longer required:
  libgssapi3-heimdal libheimntlm0-heimdal libxfce4ui-1-0 pypy-upstream-doc
  xfce-keyboard-shortcuts
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
pi@raspberrypi:~ $
```



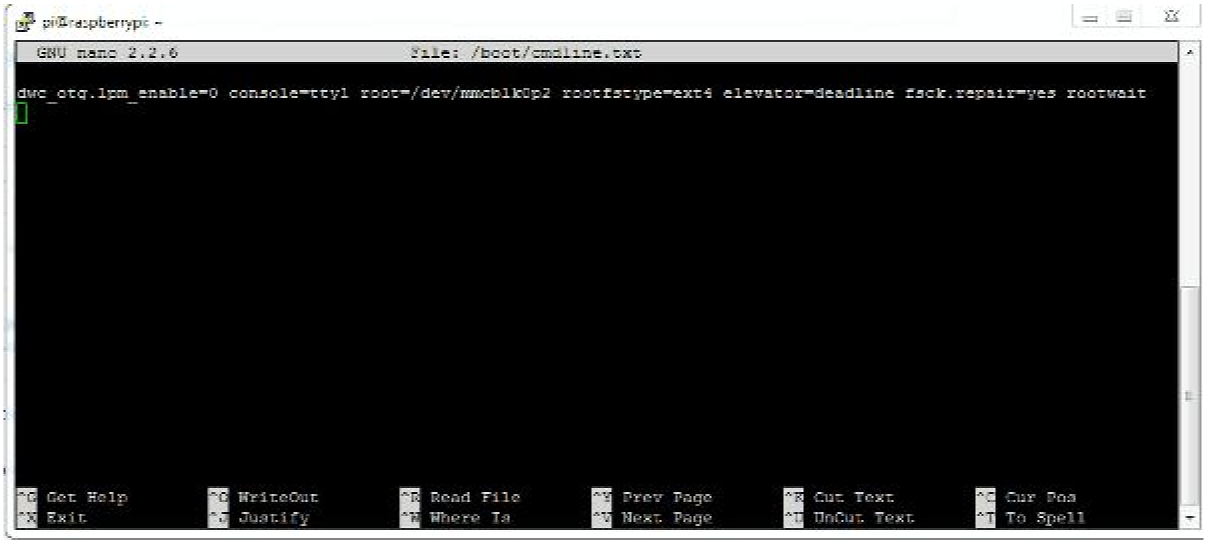
```
Raspberry Pi Software Configuration Tool (raspi-config)

1 Expand Filesystem              Ensures that all of the SD card s
2 Change User Password           Change password for the default u
3 Boot Options                   Choose whether to boot into a des
4 Wait for Network at Boot       Choose whether to wait for networ
5 Internationalisation Options   Set up language and regional sett
6 Enable Camera                  Enable this Pi to work with the R
7 Add to Rastrack                Add this Pi to the online Raspber
8 Overclock                      Configure overclocking for your P
9 Advanced Options               Configure advanced settings
0 About raspi-config             Information about this configurat

            <Select>                         <Finish>
```



```
pi@raspberrypi: ~

        Raspberry Pi Software Configuration Tool (raspi-config)

   A1 Overscan                  You may need to configure oversca
   A2 Hostname                  Set the visible name for this Pi
   A3 Memory Split              Change the amount of memory made
   A4 SSH                       Enable/Disable remote command lin
   A5 VNC                       Enable/Disable graphical remote a
   A6 SPI                       Enable/Disable automatic loading
   A7 I2C                       Enable/Disable automatic loading
   A8 Serial                    Enable/Disable shell and kernel m
   A9 Audio                     Force audio out through HDMI or 3
   AA 1-Wire                    Enable/Disable one-wire interface

              <Select>                   <Back>
```



```
pi@raspberrypi: ~
  GNU nano 2.2.6            File: /boot/config.txt

#dtoverlay=lirc-rpi

# Additional overlays and parameters are documented /boot/overlays/README

# Enable audio (loads snd_bcm2835)
dtparam=audio=on

enable_uart=0
force_turbo=1
start_x=1
gpu_mem=128

dtoverlay=pi3-miniuart-bt

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```



```
  GNU nano 2.2.6                File: /boot/cmdline.txt

dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait










^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is      ^V Next Page     ^U UnCut Text    ^T To Spell
```

# Source Code

## Steps for setting up environment :

We are using Python language here for the Program. Before coding, the user needs to configure Raspberry Pi..

After that you need to run following commands to run latest updates on Raspbian Jessie:

```
sudo apt-get update
sudo apt-get upgrade
```

After it we need to install Raspberry Pi GPIO development tool, it can be installed by following commands:

```
sudo apt-get install python-dev
sudo apt-get install python-rpi.gpio
```

Then the user needs to configure the serial port of Raspberry Pi. Here we have used Raspberry Pi 3 for this project. So the user needs to configure the serial port according to their Raspberry Pi version. For Raspberry Pi 3, the first user needs to disable console login via serial port, through RPi Software Configuration Tool. Open it by using below command:

```
sudo raspi-config
```

Then go to ‘Advanced Options’, select ‘Serial’ and ‘Disable’ it.

After this we need to **disable inbuilt Bluetooth of Raspberry Pi 3** by adding *dtoverlay=pi3-miniuart-bt* at the end of */boot/config.txt* file:

```
sudo nano /boot/config.txt
```

After adding the line **reboot Raspberry Pi** by issuing *sudo reboot* command.

Finally login in Raspberry Pi again and configure /boot/comline.txt file:

```
sudo nano /boot/comline.txt
```

And edit the file as below:

```
dwc_otg.lpm_enable=0 console=tty1 console=serial0,115200 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait
```

Now you can run the Python program given below in Raspberry Pi and you are done! Program is easy and can be easily understandable.

```
import serial

import RPi.GPIO as GPIO

import os, time

led1=17

led2=27

led3=22

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(led1, GPIO.OUT)

GPIO.setup(led2, GPIO.OUT)

GPIO.setup(led3, GPIO.OUT)

GPIO.output(led1 , 0)

GPIO.output(led2 , 0)

GPIO.output(led3 , 0)

Serial = serial.Serial("/dev/ttyS0", baudrate=9600, timeout=2)

data1=""

data=''

while 1:

  while data != '#':

    data = Serial.read(1)
    data1+=data

  print data1

  if data1.find("blue light on")>0:

      GPIO.output(led1 , 1)

      print "Blue Light on"

  if data1.find("blue light off")>0:

      GPIO.output(led1 , 0)

      print "Blue Light Off"

  if data1.find("red light on")>0:

      GPIO.output(led2 , 1)

      print "Red Light on"

  if data1.find("red light off")>0:

      GPIO.output(led2 , 0)

      print "red Light Off"

  if data1.find("green light on")>0:

      GPIO.output(led3 , 1)

      print "Green Light on"

  if data1.find("green light off")>0:

      GPIO.output(led3 , 0)

      print "Green Light Off"

  if data1.find("all lights on")>0:

      GPIO.output(led1 , 1)

      GPIO.output(led2 , 1)

      GPIO.output(led3 , 1)

      print "All Lights on"

  if data1.find("all lights off")>0:
      GPIO.output(led1 , 0)
      GPIO.output(led2 , 0)
      GPIO.output(led3 , 0)
      print "All Light Off"
  if data1.find("blink")>0:
      for k in range (100):
        GPIO.output(led1 , 1)
        GPIO.output(led2 , 1)
        GPIO.output(led3 , 1)
        time.sleep(0.1)
        GPIO.output(led1 , 0)
        GPIO.output(led2 , 0)
        GPIO.output(led3 , 0)
        time.sleep(0.1)

  Serial.flush();
  data="";
  data1="";
```