CALIFORNIA STATE UNIVERSITY, BAKERSFIELD

SMART HOME USING VOICE CONTROL

BY: KHOA T TRUONG
ROLAND TEREZON
ROBERTO CALDERON
TUONG TRIEU
OMAR MARTINEZ
TAI PHAN

OUTLINE

- Introduction
- Problem Definition
- Problem Statement and Formulation
- Design Requirement
  - Server Hardware Selection
  - Software Selection
  - Speech Recognition Hardware
  - Amazon Echo Interaction Model
- Demonstration

INTRODUCTION

- Technological advancements have made day to day life more comfortable.
- Technology is getting more and more comprehensive.
- Everything now a days has either an application or can be controlled remotely.
- Our project should be controlled by the mere sound of your voice.

PROBLEM DEFINITION

- We will try and address some of the problems with existing smart home models
- Non recognizable commands.
- More interactive software.
- Easier installation if not installed already in a home.

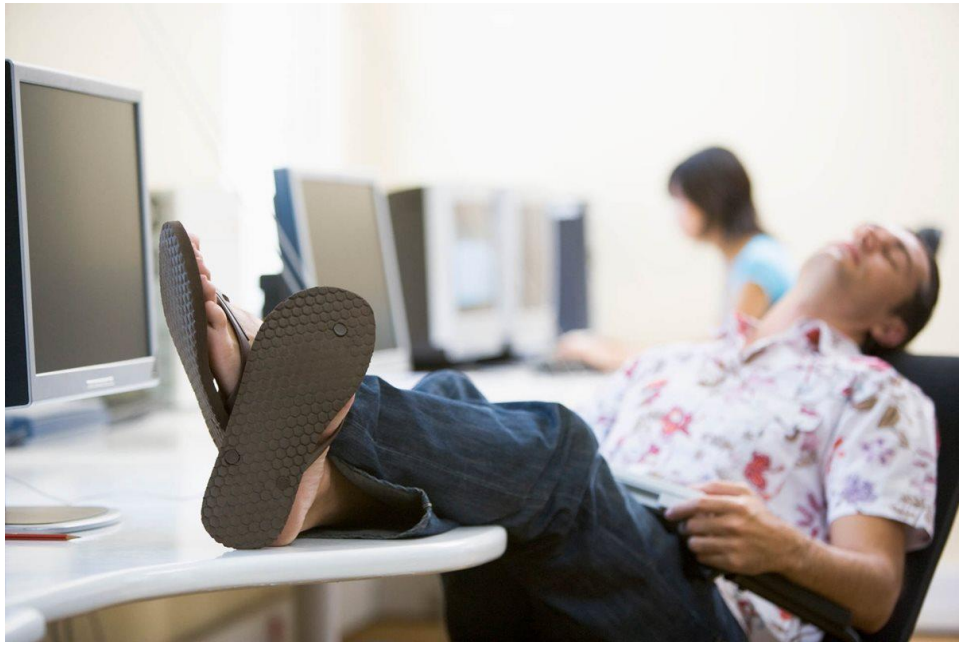
PROBLEM STATEMENT AND FORMULATION

Smart Home System is:

- Interactive

PROBLEM STATEMENT AND FORMULATION

- Convenient, …+



- Energy Saving

PROBLEM STATEMENT AND FORMULATION

- Cost Effective

COST ESTIMATE

Components	Cost
1. Raspberry Pi 4 Ultimate Starter Kit	$35-$55
2. ESP 8266	$10
3. Amazon Echo	$35
4. Mini Breadboard	$5
5. RGBW LED 5 meters (16ft)	$40
6. Mini Model Home	$200 (Provided)
7. Power Supply for Model Home	Provided?
…	
Total	$135~

ORIGINAL MODEL HOME IDEA

MATERIALS:

- Wood
- Carboard
- Nails
- Glue
- Etc.

NEW MODEL HOME

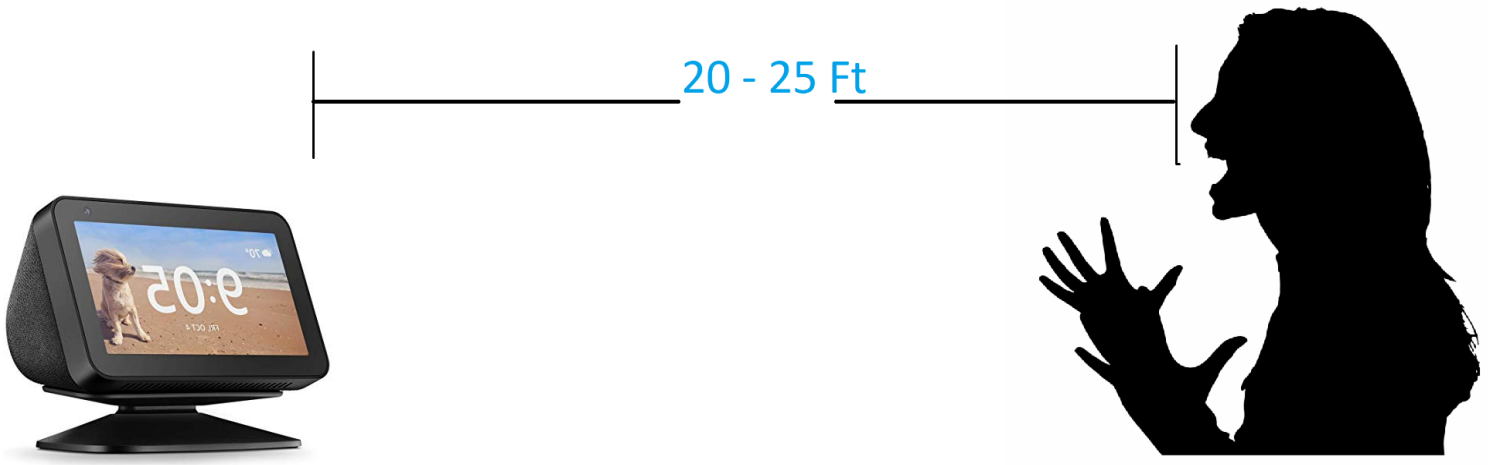
MATERIALS:

- Premade Dollhouse

PROS:

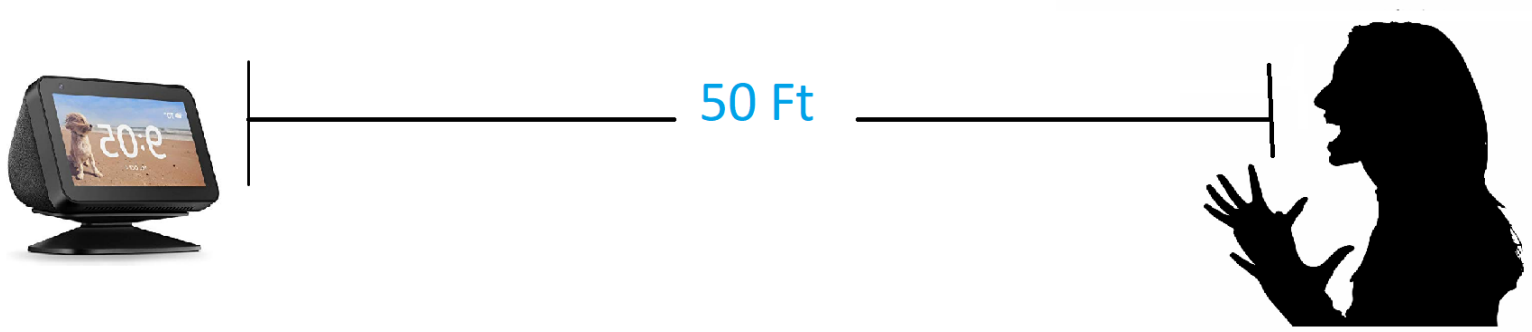
- Opens up
- Easy access for lighting set-up
- Easy access for wiring
- Portable

SMART HOME COMMUNICATION DISTANCE



- Using normal tone (no yelling, shouting, etc.)
- Normal indoor setting

SMART HOME COMMUNICATION DISTANCE

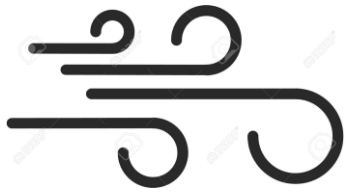


- Using louder tone (some shouting, maybe yelling)
- Normal indoor setting
- May need to shout louder outdoors

SMART HOME COMMUNICATION DISTANCE

- Communication device can be taken anywhere—As long as the Wifi is connected to it

SMART HOME COMMUNICATION DISTANCE



- More distance = more ambient noise = more interruption = more likely to fail

SMART HOME COMMUNICATION DISTANCE

- Enclosed spaces with less noise will give better distance results

SMART HOME COMMUNICATION ADDS

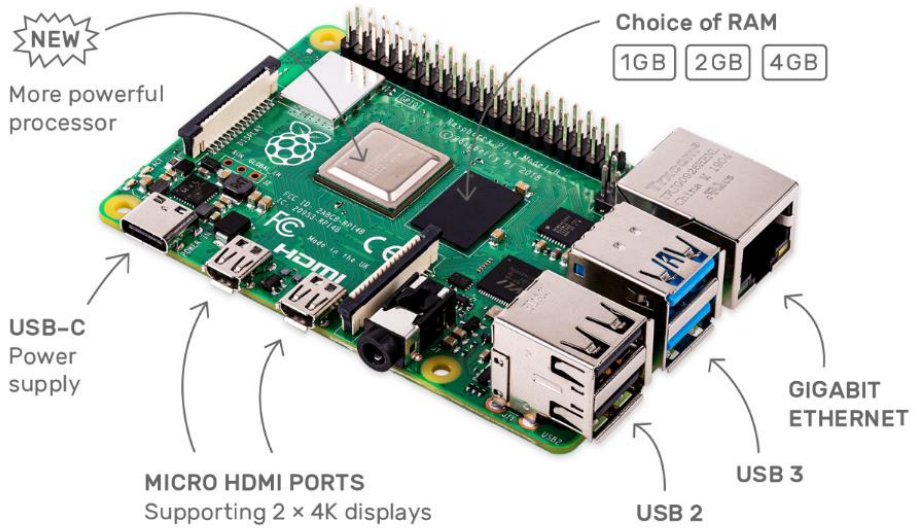
- There is a way to have multiple communication devices, thus moving one device around the house will not be necessary

Pro: Command the Smart Home from virtually anywhere in the house
Con: Will have to buy more communication devices = more spending

DESIGN REQUIREMENT - SERVER HARDWARE SELECTION

RASPBERRY PI 3

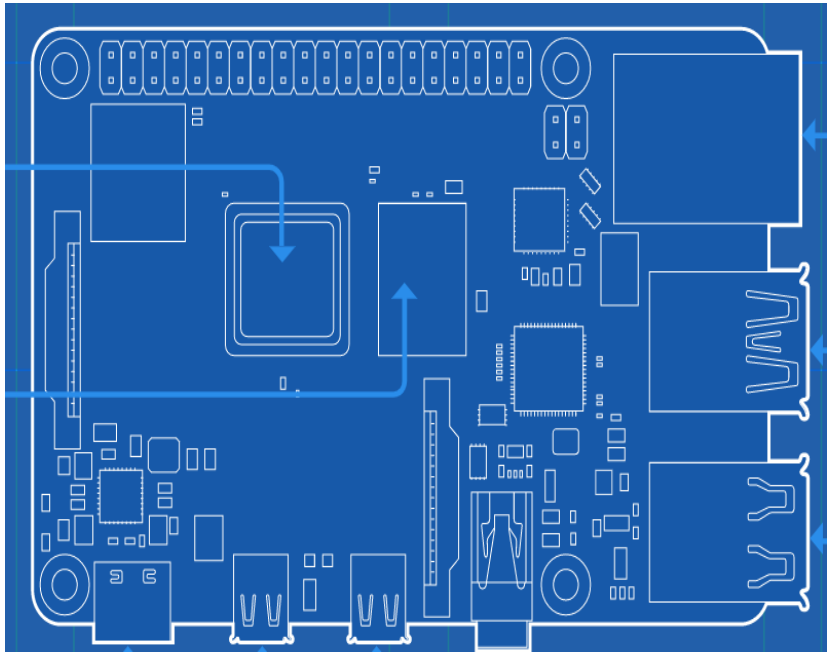
- This will be our main server.
- 1.2 GHz 64-bit quad core ARM Cortex-A71 processor.
- Bluetooth 5.0 and Wi-Fi hardware
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO connector.
- 2 × micro-HDMI ports (up to 4kp60 supported).
- Has 1 GB of onboard Ram.



DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

RASPBERRY PI 3 (CONT'D)

- Can run a host of operating systems:
  - Raspbian
  - Android
  - Windows 10
- Collects, analyzes, and acts on data collected.
- Can connect multiple sensors to GPIO pins.
- Can install openHABian that will allow the use of openHAB.
- Trouble with USB cables such as those used on MacBooks.
- Ideal candidate for IoT projects.



DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

5V RELAY

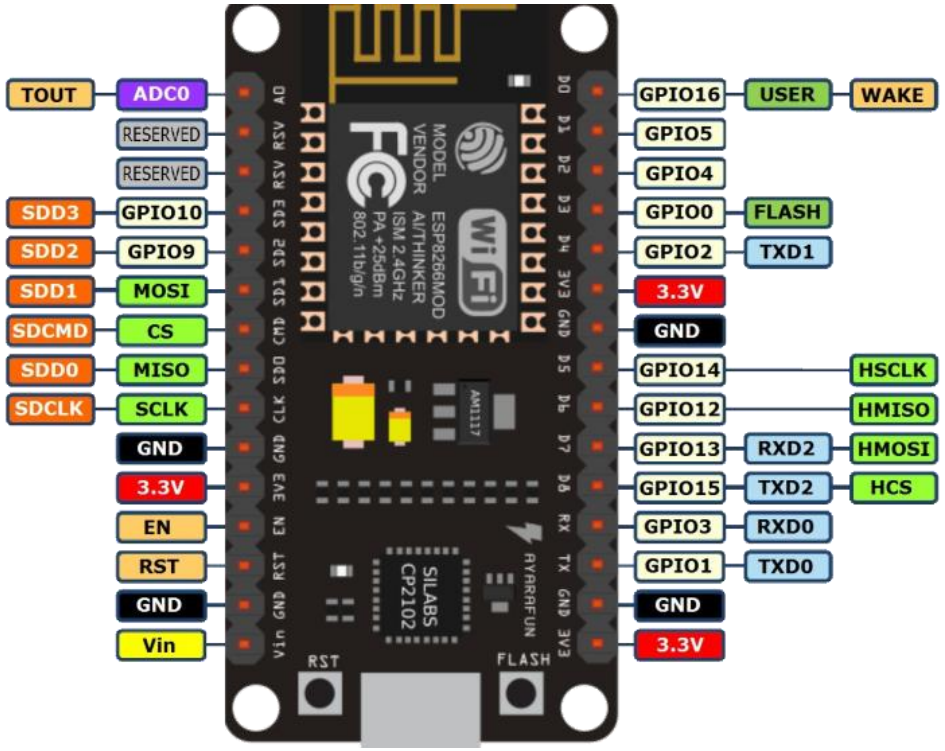
- The Raspberry Pi nor ESP8266 can control high voltage devices.
- It is an electrically operated switch or component used to break or interrupt a circuit.
- Can be turned on or off.
- It is controlled by low voltages that are provided by the ESP8266.
- Signal carries the trigger signal (HIGH) from the ESP8266 that activates the relay.
- Common is where the 120-2540V supply current enters the relay.

5V Relay Terminals and Pins

DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

ESP8266

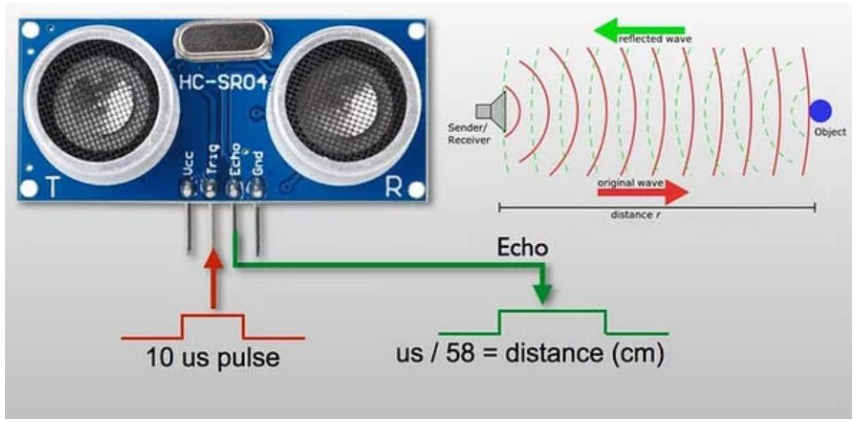
- It is a 3V WiFi module.
- VCC and GND are powering pins.
- Transmission (from server) and Receive (to server).
- It's I/O pins run at 3.3V



DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

HC-SR04ULTRASONIC SENSOR

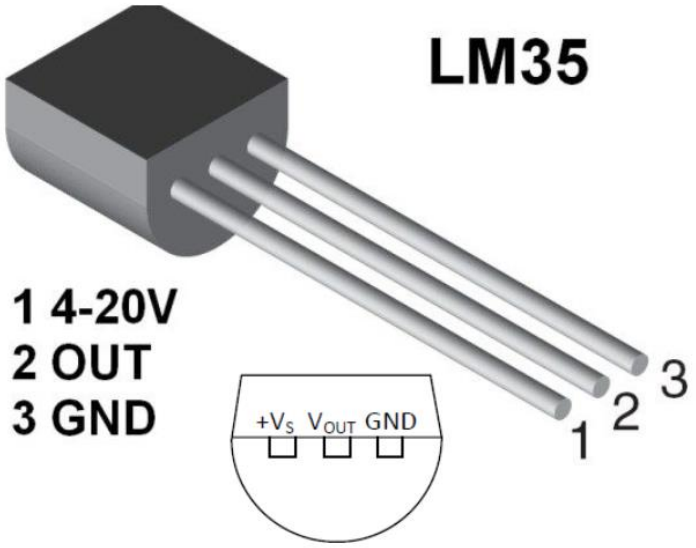
- Emits sound waves at frequency too high for humans to hear.
- It waits for sound to be reflected back and calculates the distance based on the time.
- Not affected by color of the material but can have difficulty if material is made from something that absorbs sound waves or reflects sound waves from the receiver.
- It has 4 pins:
  - VCC- needs 5V to be active.
  - Trig- it is triggered by the ESP8266 to emit the soundwave
  - Echo- Informs the ESP8266 when the receiver received the bounced back wave.
  - Ground- needs to be grounded using the ESP8266



DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

LM35 TEMPERATURE SENSOR

- Outputs an analog signal that is proportional to the instantaneous temperature.
- Output voltage can be interpreted to obtain a temperature reading in Celsius.
- Can measure from -55 degrees to 150 degrees Celsius with very high accuracy levels.
- It is a +10 mills volt per degree centigrade, meaning that with an increase in output of 10 mills volt by the sensor Vout pin, the temperature value increases by one.
- Has 3 pins:
  - Vs- Voltage from the ESP8266 needed to activate the sensor.
  - Vout- Informs the ESP8266 of the reading
  - Ground- It is connected to the Ground of the ESP8266.



LM35 Pin out

DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

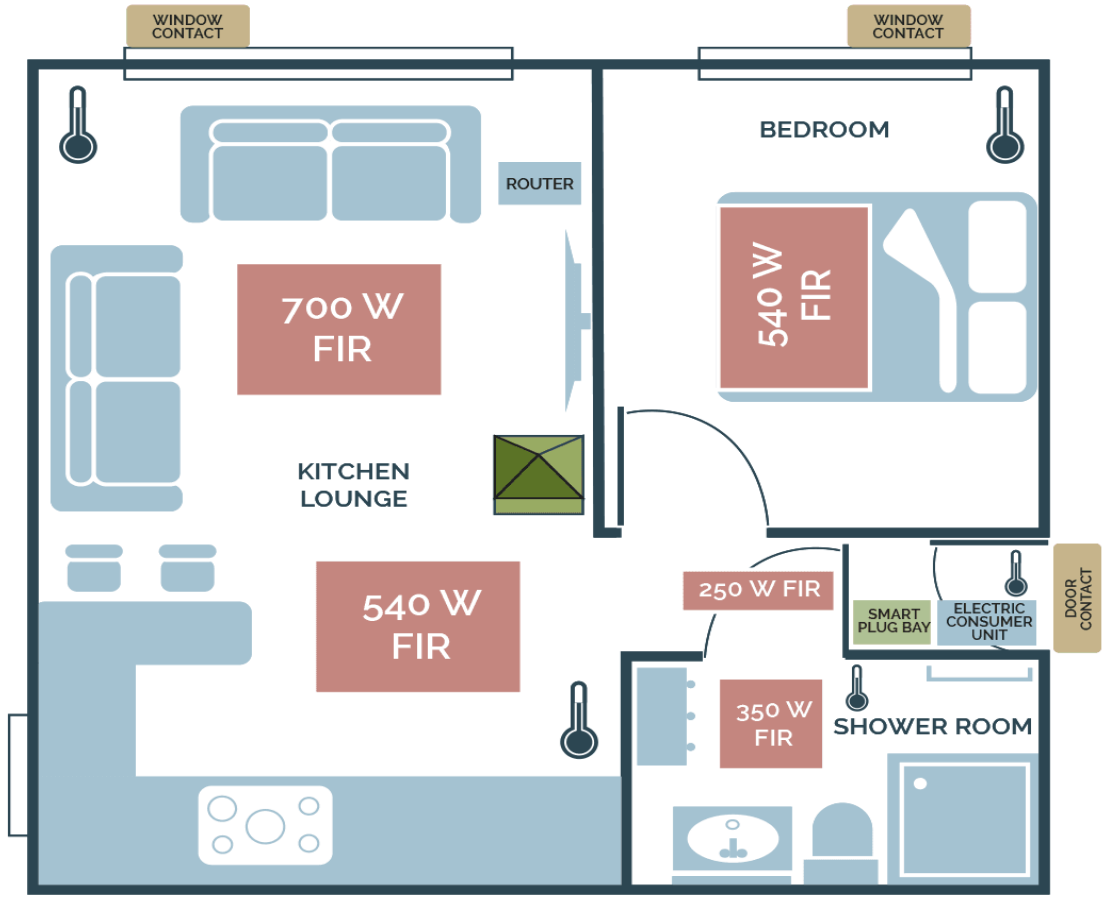
MQ-2 GAS SENSOR

- Gas sensor suitable for sensing LPG, Smoke, Alcohol, Propane, Hydrogen, Methane, and Carbon Monoxide concentrations in the air.
- It is a metal oxide semiconductor type gas sensor that is based upon the change of resistance of the sensing material when gas contacts the material.
- A voltage divide network is used to detect the concentrations of gas.
- Specifications:
  - Operates on 5V
  - 20 kOhms of load resistance
  - 10 kOhms – 60 kOhms of sensing resistance
  - Concentration scope of 200 – 10000 ppm
- It has 4 pins:
  - Vcc- Connected to the ESP8266 to be active
  - Ground- Also connected to the ESP8266
  - A0- Provides analog output voltage in proportional to the concentration of smoke/gas
  - D0- Provides digital representation of the presence of combustible gases.

DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

SAMPLE IMAGE

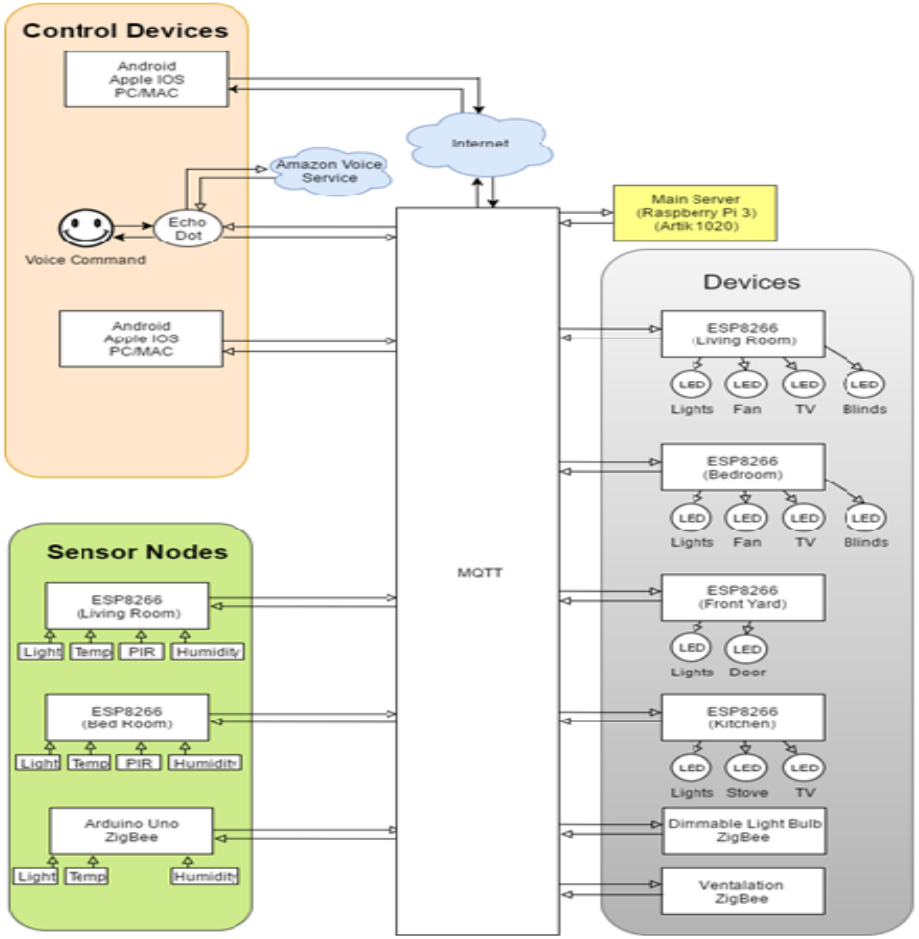
- It will require wiring for:
- High Voltage (AC) – 120 to 240 VAC
- Low Voltage(DC) – 3.3 to 5 VDC
- Data (Ethernet & communication) – cat5e and Wireless
- Protection devices – Breaker and Fuses
- Control Devices – Switches, outlets and relays



DESIGN REQUIREMENT - SERVER HARDWARE SELECTION (CONT'D)

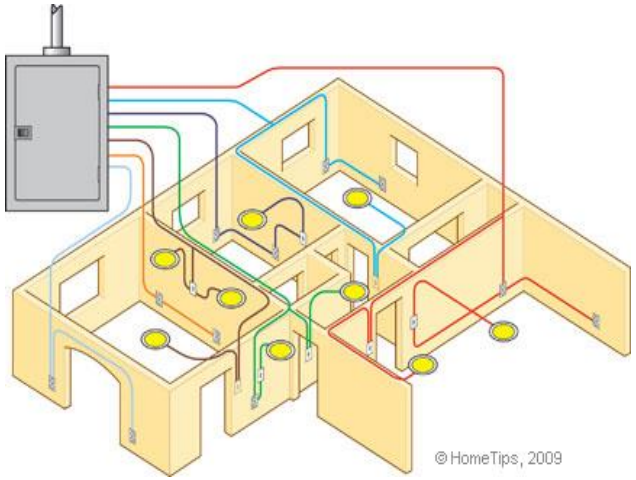
SET-UP

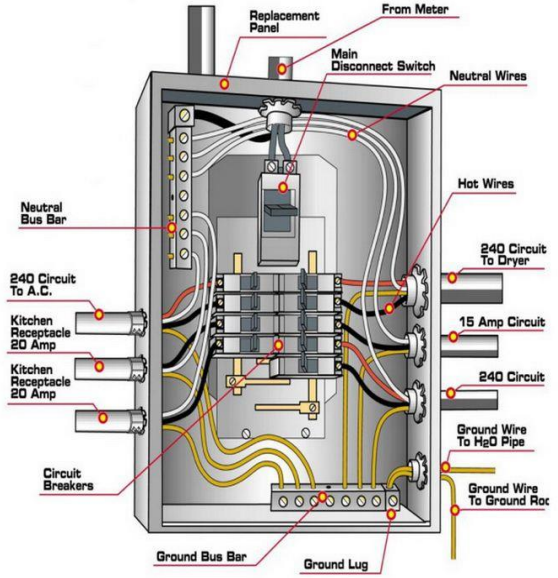
- Raspberry Pi will be our main server.
- It will connect to appliances throughout the house.
- Will connect to sensors, lighting, and other devices.
- We will most likely be using smart appliances.
- For those that do not have smart home ready we can hardwire it with the help of an ESP8266.



HOME WIRING

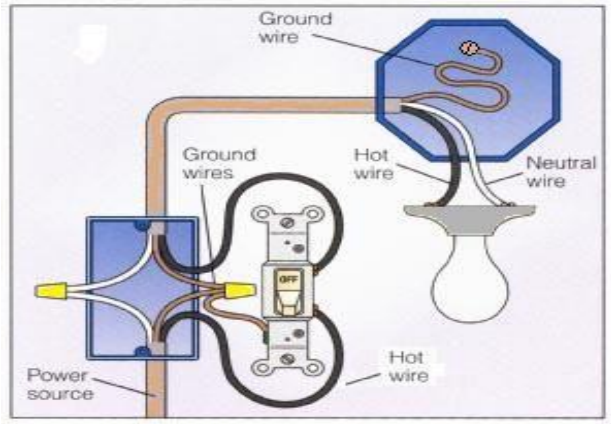
- Main power feed is located at the exterior of the dwell with 120-240 VAC service lines.
- Circuits branches inside the home breaker box will route the dedicated electrical service to devices(outlets, switches and fixtures)
- Control Box will interrupt the circuits to add a smart control device that can be manipulated via software.





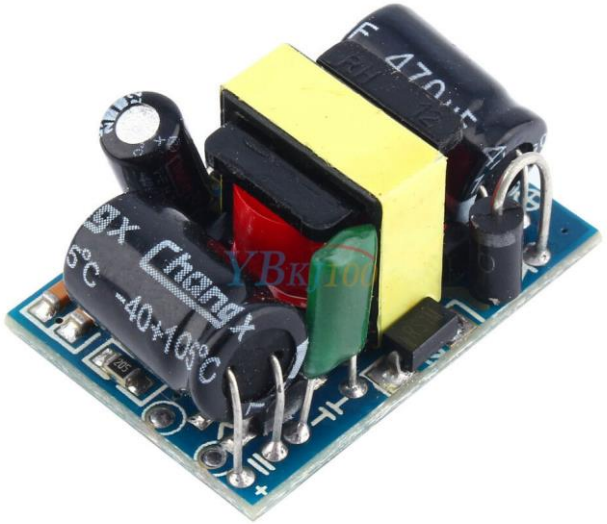
CONTROL MODULE

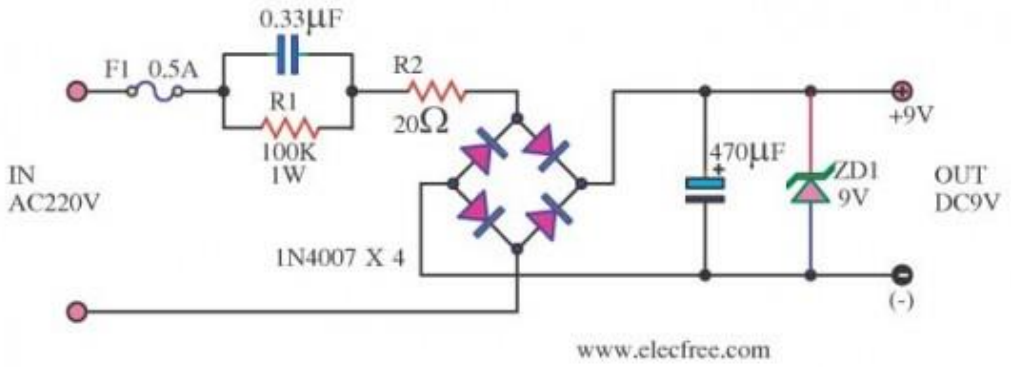
- Voltage regulator: capable of providing the VDC for electronic components.
- ESP8266 board: allows wireless communication with main server to transmit data commands.
- Relay box: it acts as a switch to allow the VAC source on/off.
- Hardwire device: usually an outlet, or wall switch where devices are control or connected.



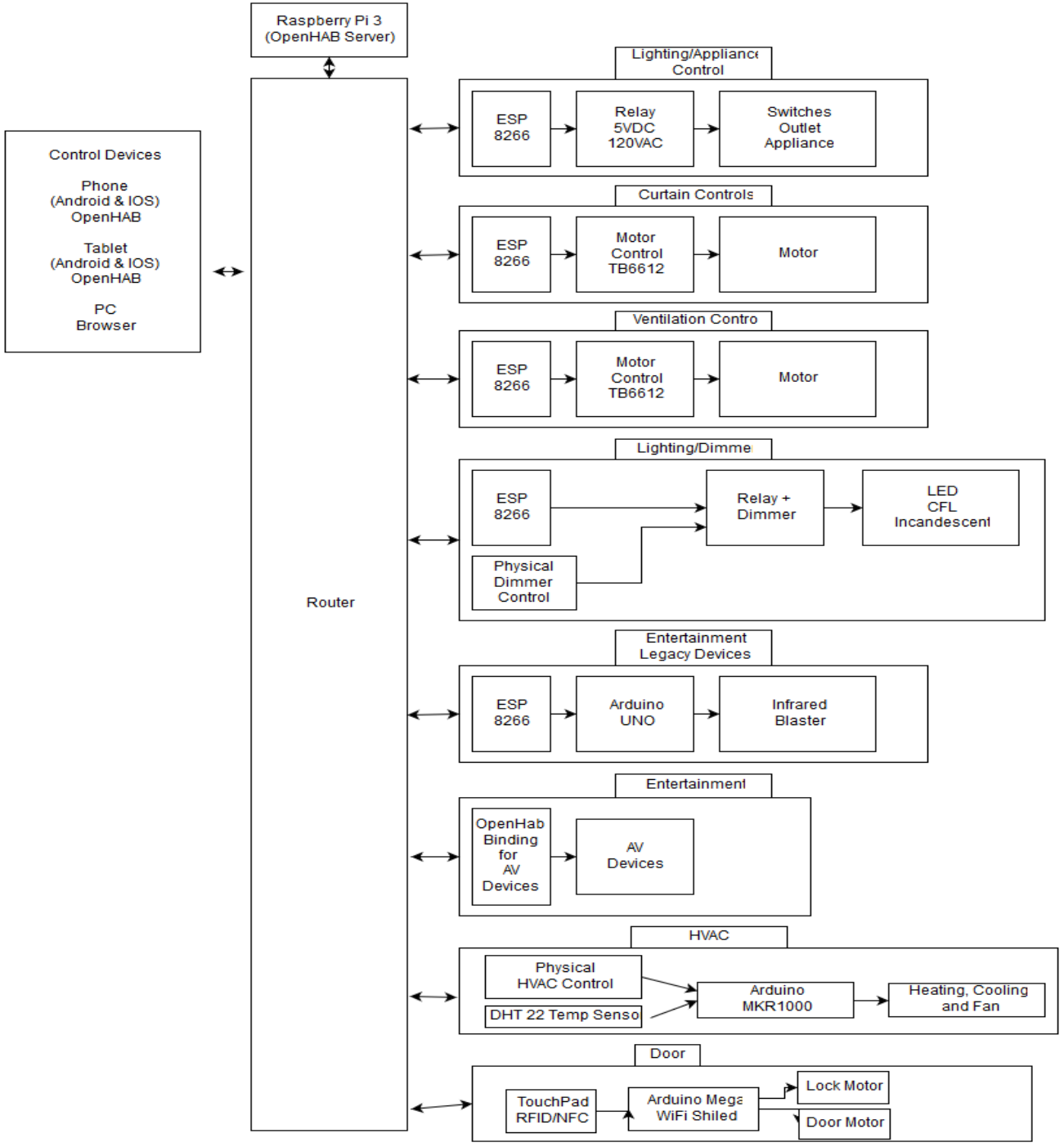
VOLTAGE A.C – VOLTAGE D.C

- Voltage regulator
- Transformer
- Transformer less
- Size efficient





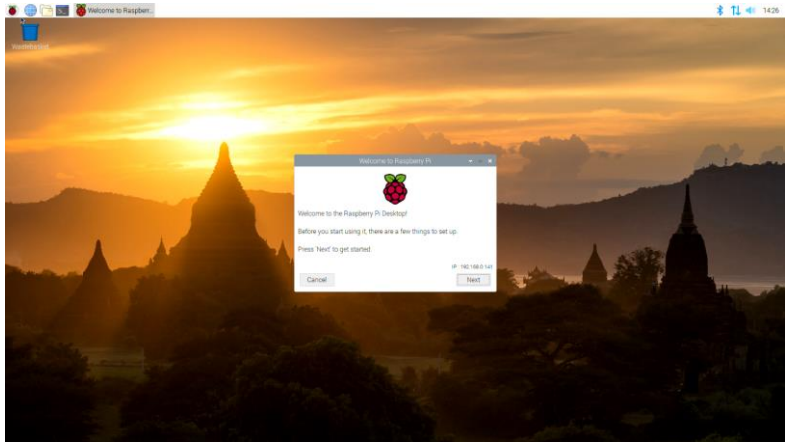
SYSTEM DIAGRAM



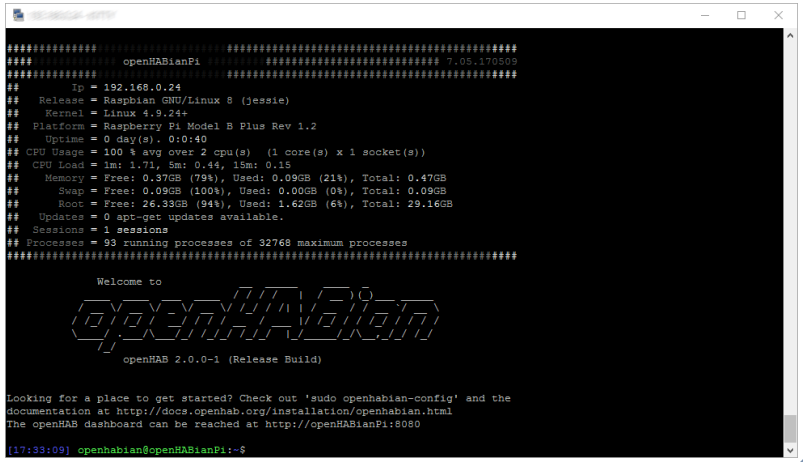
DESIGN REQUIREMENT - SOFTWARE SELECTION

RUN ON LINUX

- Setup Rasperian OS on Rasperry Pi 3
- Using Python for coding application on Server side
- Seting openHAB Server on Rasperian OS



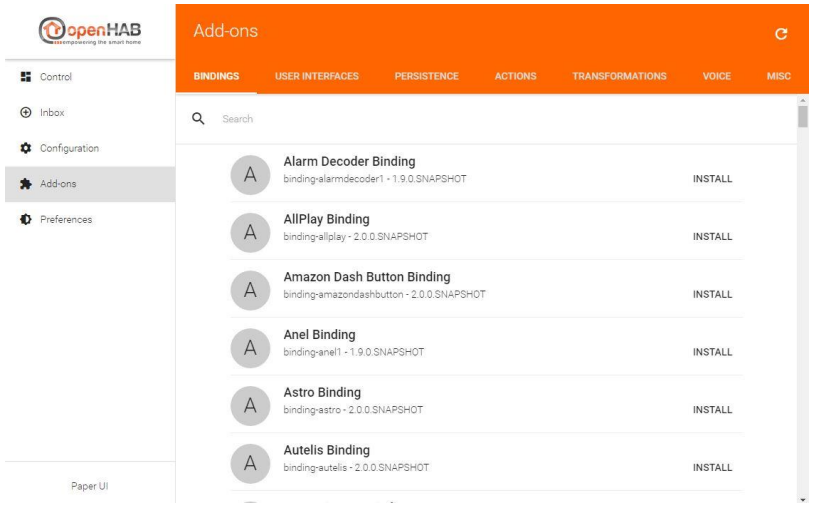
python™



DESIGN REQUIREMENT - SOFTWARE SELECTION (CONT'D)

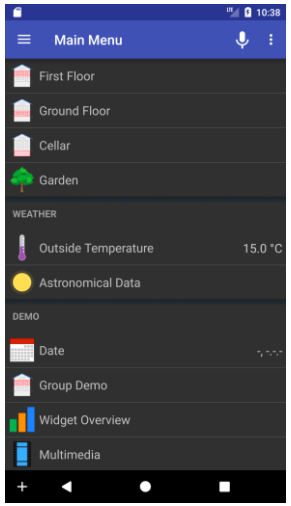
FRONT END – CONTROL DEVICES

- Our smart home system to be able to control devices with any phone or computer
- Smart home ecosystem called OpenHAB, provide
  - Web version
  - Mobile applications that run on Android and iOS



WEB VERSION

IOS VERSION

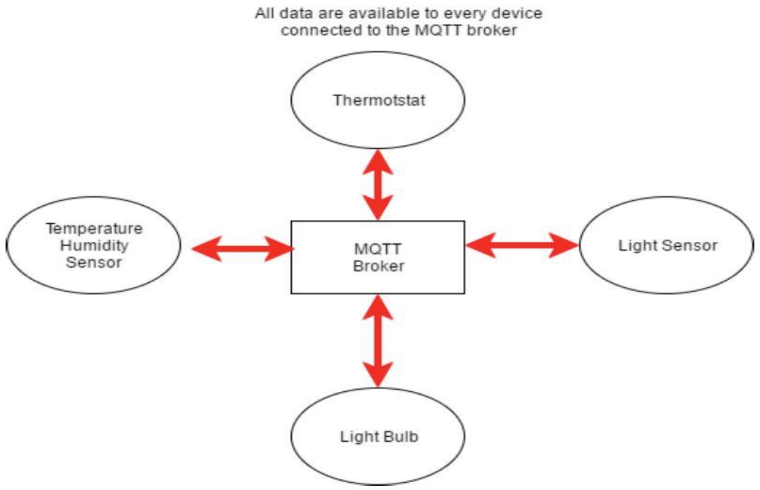


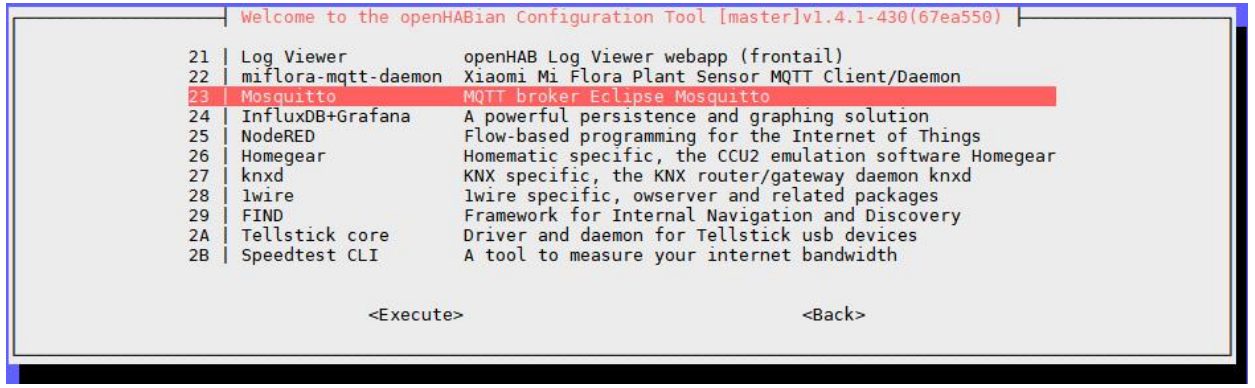
ANDROID VERSION

DESIGN REQUIREMENT - SOFTWARE SELECTION (CONT'D)

UNIFICATION PROTOCOL

- We seek is a protocol will translate different protocol and seamlessly integrate them together
- Message Queue Telemetry Transport (MQTT) messaging protocol also known as Mosquitto
  - MQTT is a lightweight messaging protocol that sites on top of the TCP/IP protocol and is an ISO standard (ISO/IEC PRF 20922)
  - MQTT is a very small protocol that is a publish and subscribe based messaging protocol



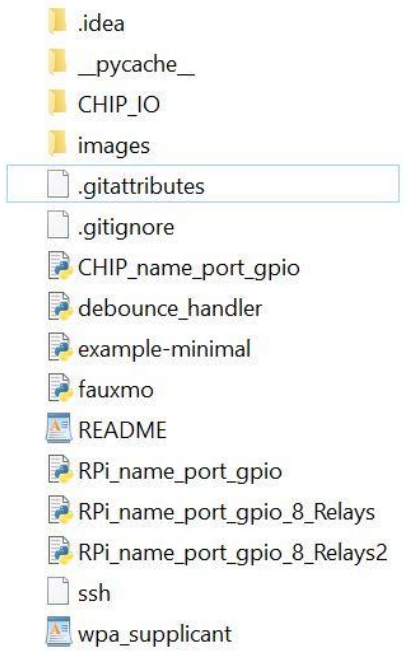


Enable MQTT broker on openHAB

DESIGN REQUIREMENT - SOFTWARE SELECTION (CONT'D)

SERVER SIDE

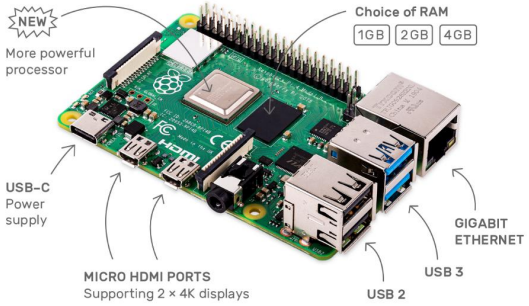
- Using Python to coding Server-Side application connect with Amazon Echo Show (use AWS)

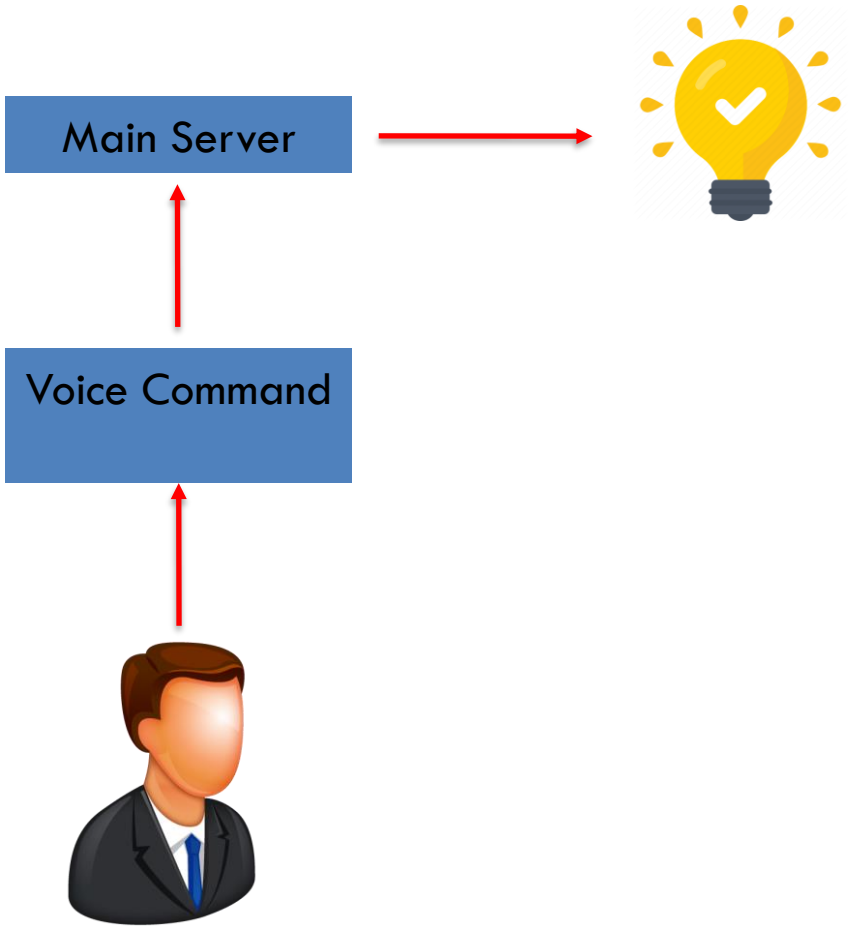



```python
import fauxmo
import logging
import time
import sys
import RPi.GPIO as GPIO ## Import GPIO library

from debounce_handler import debounce_handler

logging.basicConfig(level=logging.DEBUG)

class device_handler(debounce_handler):
    """Publishes the on/off state requested,
       and the IP address of the Echo making the request.
    """
    #TRIGGERS = {str(sys.argv[1]): int(sys.argv[2])}
    TRIGGERS = {"FAN": 52000,"LED":51000}

    def act(self, client_address, state, name):
        print("State", state, "from client @", client_address)
        # GPIO.setmode(GPIO.BOARD) ## Use board pin numbering
        # GPIO.setup(int(7), GPIO.OUT)   ## Setup GPIO Pin to OUTPUT
        # GPIO.output(int(7), state) ## State is true/false
        if name=="kitchen":
            GPIO.setmode(GPIO.BOARD) ## Use board pin numbering
            GPIO.setup(int(7), GPIO.OUT)   ## Setup GPIO Pin to OUTPUT
            GPIO.output(int(7), state) ## State is true/false
        elif name =="living room":
            GPIO.setmode(GPIO.BOARD) ## Use board pin numbering
            GPIO.setup(int(11), GPIO.OUT)   ## Setup GPIO Pin to OUTPUT
            GPIO.output(int(11), state) ## State is true/false
        else:
            print("Device not found!")
```

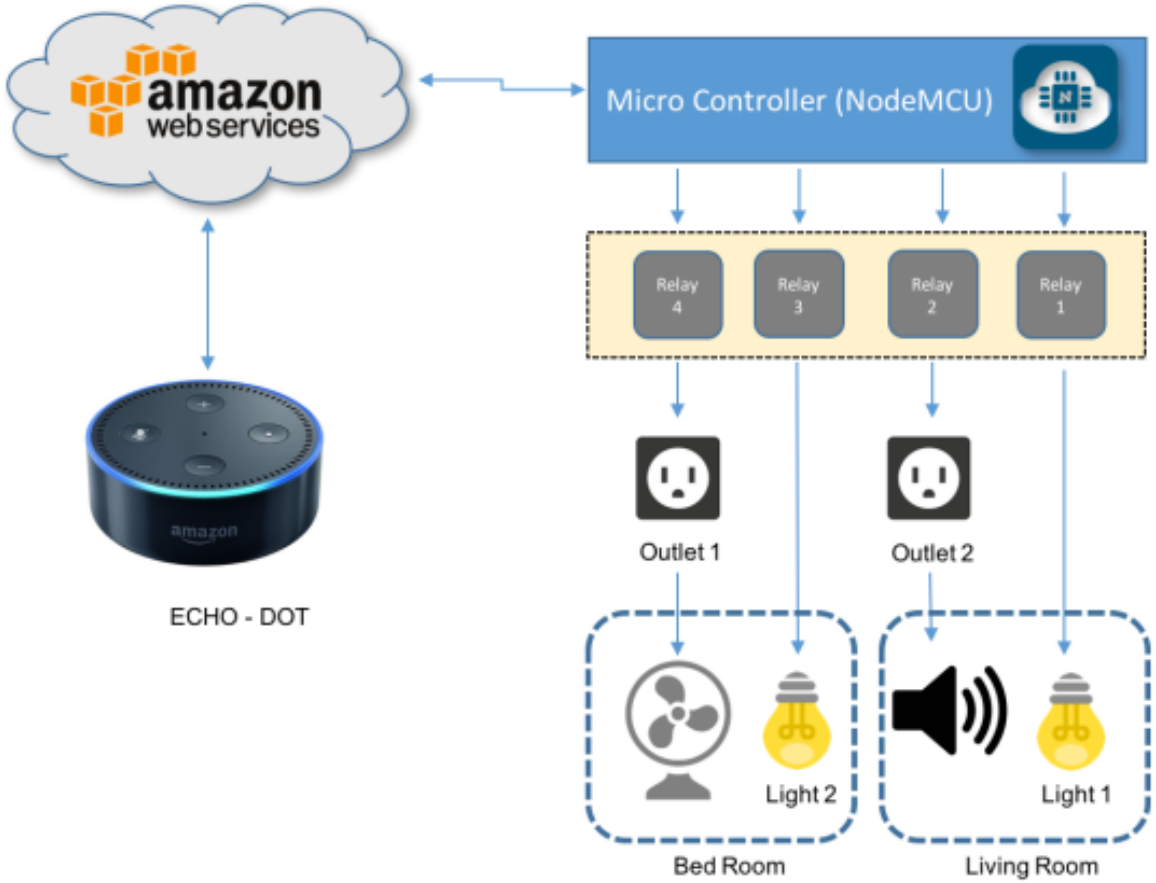
VOICE CONTROL

- Raspberry Pi 3
- Amazon Echo Show 5





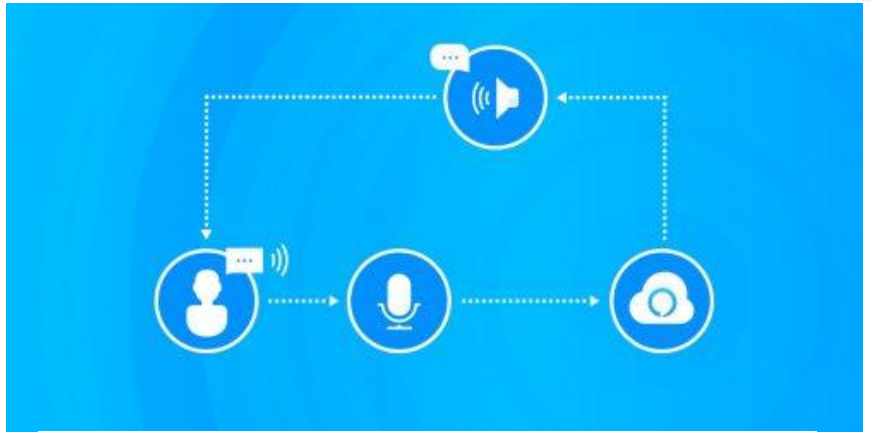
AMAZON ECHO SHOW 5



DESIGN REQUIREMENT - SOFTWARE SELECTION (CONT'D)

SPEECH RECOGNITION HARDWARE – AMAZON ECHO SHOW

- Pick-up sound from any direction.
- Echo Show voice processing is done in the cloud through Amazon Voice Service.
- Being improved to better recognize spoken words.

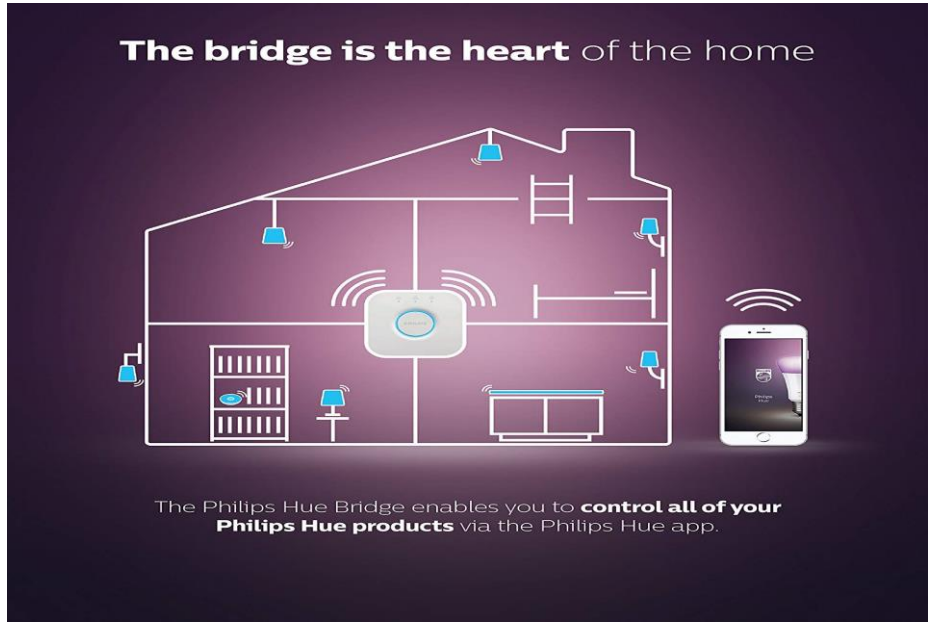


amazon alexa

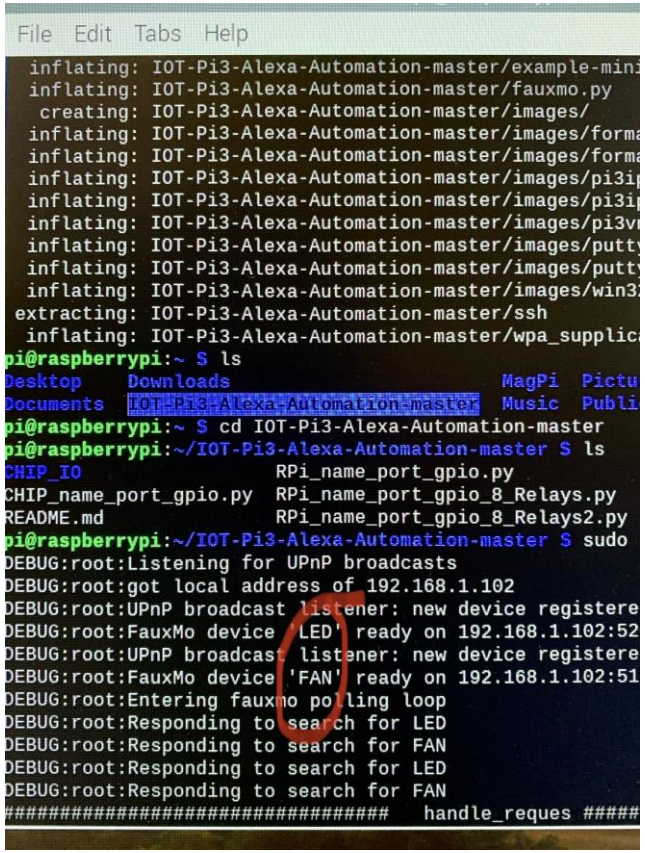
DESIGN REQUIREMENT - SOFTWARE SELECTION (CONT'D)

AMAZON ECHO INTERACTION MODEL - ECHO DOT PHILIPS HUE EMULATOR

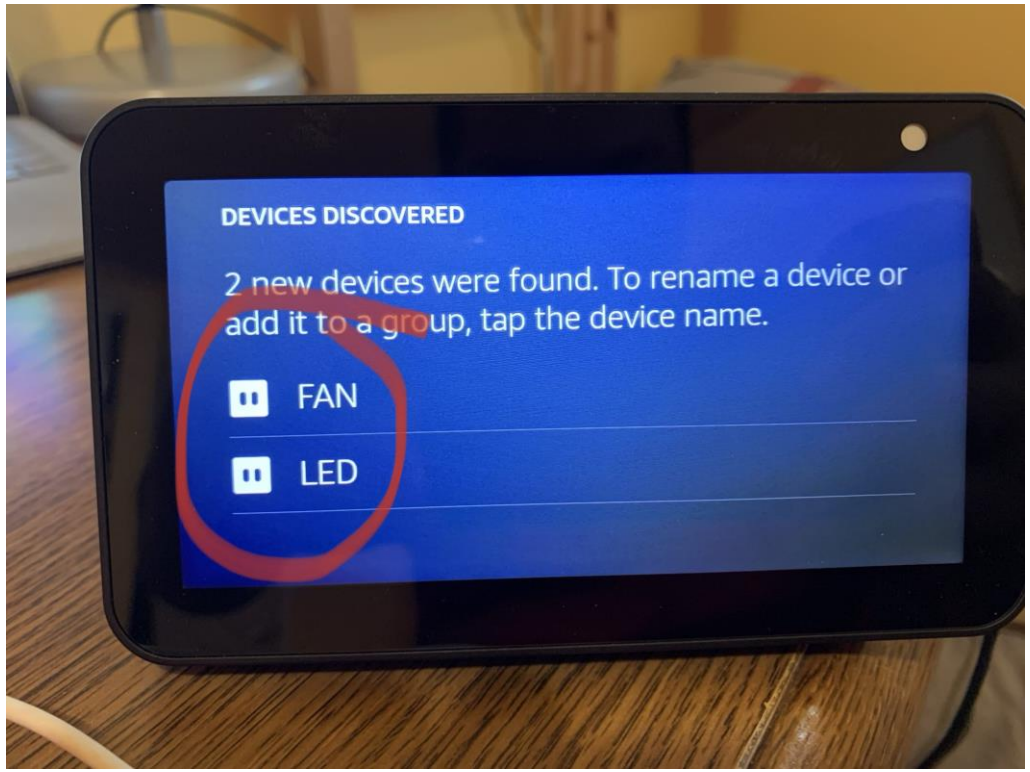
- Translate any device protocol to the Echo Dot.
- Turn any device into Philip Hues device then will be able to control it using voice command through the Echo Dot.



DEMONSTRATION



Server side



Echo Show recognize all devices

- Link demo: https://youtu.be/oP55Qt_gB0w