ISSN 2347 - 3983
Volume 8. No. 7, July 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter98872020.pdf
https://doi.org/10.30534/ijeter/2020/98872020

Design and Development of IoT based Intelligent Home Automation System

Shubham Ghanghas, Sonal Dahiya, Manoj Kumar Pandey, Sweta Tripathi
UG Scholar, Amity University Haryana, Gurugram, India, shubh2099@hotmail.com
Assistant Professor, Amity University Haryana, Gurugram, India, mkpandey@ggn.amity.edu
Assistant Professor, Amity University Haryana, Gurugram, India, sdahiya@ggn.amity.edu
Associate Professor, Kanpur Institute of Technology, Kanpur, India, swetatripathi16@gmail.com

ABSTRACT

Home automation has made its way from science fiction movies to forefront of consumer market. It is not just the new buzzword; it is making life simpler and less demanding. This paper presents the design of Home Automation System which uses conventional development board with sensors and also include concepts like deep learning techniques via cloud for provision of idiosyncratic features. This low cost and reliable control and monitoring system runs over internet and is therefore free from physical restrictions. This Smart Home Controller could be fitted inside home without infrastructure changes. The proposed system consists of a Raspberry Pi, a camera, android mobile, Relays, multiple sensors. The feed from the camera will take pictures and match them with households to make sure only authorized people get house access. The computation will be done on cloud using AWS. The user can also control the system using android based application. This System also responds to voice commands to Google Assistant. The system is very flexible and user friendly.

Key words : Automation, Android, Raspberry Pi, Internet of Things (IoT), Voice Recognition, Cloud.

1. INTRODUCTION

Home automation is a technology that allows home appliances to perform different sets of tasks automatically. Home automation not only provides peace of mind but also the ease and expediency. It liberates the user from the hassle of remembering to perform other tasks as watering plants, feeding fish and other household work. The idea of smart homes has captured people’s imagination for a long time from the science fiction movies, now it only takes few bucks to achieve the same [1]. More and more smart home systems will be controlled by smartphones and microcontrollers as the technology becomes simple, affordable and readily available [2]. An Application has been developed that will run on smartphone which will help user in controlling and monitoring home functions over internet. This project provides a composition of components to build a robust approach of an advanced smart home concept and implementation.

In the past few years IoT technology has advanced significantly and came from uncommonness to limelight. The number of devices connected is assumed to keep increasing at an ever-increasing speed and the number of things has already surpassed the number of humans.

Automation makes sense since appliances only require turning on/off based on generally only one or sometimes two factors but rarely more than that and such decisions could be easily taken by a computer which in some cases may require basic learning but nothing that a cheap computer could not handle. Automation reduces human judgment although it is not wise to completely eliminate it. The Home Automation system allows the user to view the status of devices at home and also allows to control them while being free of physical limitations. The user upon returning to home will find a very contented and pleasant home.

For the people who are constantly on the move, not only do they need to monitor security of house but also stuff as watering the plants, feeding the fish. A home automation system can not only do such works, but it also does the work of CCTV [3]. Such a system not only is more convenient, secure but is also eco-friendly since it reduces electricity wastage by switching off components when not in use. It does not need to run camera all the time like a CCTV does rather it uses sensors which sense movements and turn on the camera which also saves on storage.

An automated system not only reduces the burden to attend each device installed and the stress induced with it but also saves on the resources as time, power wastage and money. A remote-control system liberates user from physical restriction of being at home while likewise mechanizing most assignments hence being increasingly practical and thus expedient.

IoT can greatly benefit from cloud services [4]. IoT devices generate a tremendous amount of data. Not only can cloud store that data efficiently but it can use cloud computing to decrease the hardware and software demand form the user. This allows for a system with weak processor of Raspberry Pi to perform deep learning tasks as facial recognition[11][12].

Speech recognition is also one of the main components of this project. As speech is one of the most natural means of communication so it seems only natural to have it as point of interaction between user and system. The system uses Google's speech recognition. Every Android smartphone has Google Assistant built-in [5]. So, the system can be used on any Android smartphone. The system could be controlled either using voice or text of via app.

2. RELATED WORK

An excellent point made by Li et al. that even though it might appear that products such as home automation are just another rich man sycophancy; The reality is far from speculation. Using Google Trends, they found that places leading the world in interest of home automation are the growing economies of the world notably India and South Africa. As country grows so does the expectation of people, they want more than shelter, water, and electricity, they want luxury which is conveniently and cheaply offered by Home Automation System. In second part of their research they focus on sustainable development in terms of smart home systems. A home automation system not only is more convenient, secure but is also eco-friendly since it reduces electricity wastage by switching off components when not in use and uses right amount of water for plants reducing water wastage [6].

It has been contemplated that huge amount of data being generated from sensors can be analyzed to make a true automated system which instead of actions as a result of triggers can makes predictions. They claim that such a system unlike one which does things simply because it was programmed to do so is unintuitive; However, a system which using learning algorithms can use resources to pin-point precision not only being more sustainable but also more flexible [7].

A voice-controlled automation system using Raspberry Pi has also been made. The voice command is received by Google Assistant which it forwards to IFTTT App. IFTTT server analyzes the voice commands and matches it to users store of commands, if a match is found it forwards the specific command to Blynk App which sends to data over server to Raspberry Pi which is using authentication key for connection. Raspberry Pi then turns the respective pin to High Output in turning the device on. The claim has been made that their system takes as much as more than 1 minute to show the result which is fascinating since a similar approach is used in this project which renders instant action. Also, the developed system does not use any sort of learning techniques [8].

A comparison between various flavors in implementation of home automation systems has also been done and includes systems connected via Bluetooth, Wi-fi, embedded server, and cloud server. Systems connected via Bluetooth have a short range but is very cost effective so is good for making something quick and dirty. A system using Wi-fi not only increase the range but also tweaked to reduce power consumption. A system using embedded server can be cost effective yet secured but it requires more technical knowledge. Using a cloud server can make a system truly universal and sometimes for usage involving a large number of users is the best option [9].

The issues that are associated with smart home control system and devices attached to it have also been discussed. It is suggested that because succumbing to security features might decrease performance of a system whose most important job is to do things instantly, security is overlooked. Also, there are money constraints, IoT systems are expected to be cheap. The embedded security framework as algorithms suitable for connected devices have not been developed. Dynamic and heterogenous nature of devices is the big factor responsible [10].

3. METHODOLOGY

The following contributions are made to achieve the objectives: -

- The proposed system uses a Raspberry Pi board which is a miniature computer with all the functionalities of a desktop computer. It has built in Wi-fi and Bluetooth. It can interact with vast array of sensors and can be programmed with multiple programming languages.
- The developed Home Automation System uses conventional development board with sensors but also deep learning techniques via cloud which provides idiosyncratic features.
- The major components used in the developed system consists of a Raspberry Pi, a camera, android mobile, Relays, multiple sensors.
- The feed from the camera take pictures and match them with households to make sure only authorized people get house access. The computation will be done on cloud using AWS.
- The developed system is controlled via Blynk app which can receive vocal commands from Google Assistant.
- The system uses an ultrasonic sensor to measure the distance of user from front door and when a threshold is reached, it turns on camera which saves on both power and storage.
- The system includes an LDR which automatically turns ON lights if it detects low-lit environment.
- The system also uses a Soil Moisture Sensor which turn on the sprinklers to water the plants if it detects dry conditions.

The circuit diagram of the developed project is shown in Figure 1.

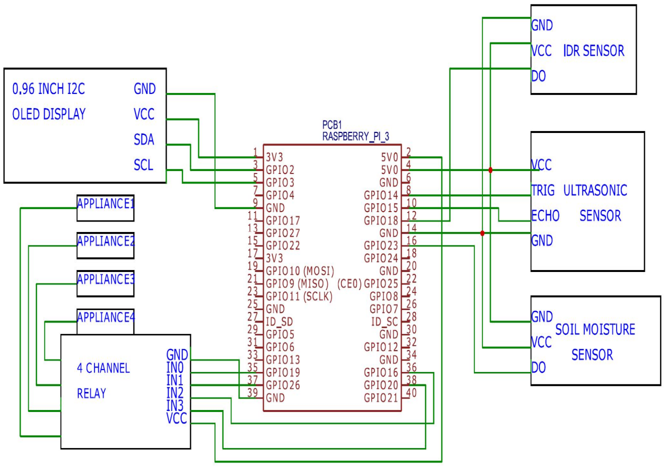


Figure 1: Circuit diagram of System

Functional description of various hardware components is given below:

Raspberry Pi

Raspberry Pi board as can be seen in Figure 2, is a miniature computer with all the functionalities of a desktop computer. It has built in Wi-fi and Bluetooth. It can interact with vast array of sensors. It has camera port and 40 pins. The specifications are given in Table 1.

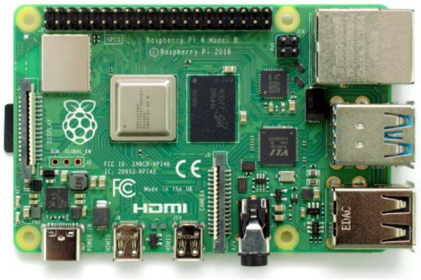


Figure 2: Raspberry Pi Model 4B

Table 1: Raspberry Pi Model 4B Specifications

Specification	Value
GPIO Pins	40
Memory	LPDDR4-3200 SDRAM
CPU	Quad Core 1.5 GHz
Output Video	4K @ 60fps

Raspberry Pi Camera module

Raspberry pi camera module as shown in Figure 3, is a tiny board with attached camera that can capture both images and videos. The camera is connected to Pi using CSI bus. The 5MP camera is capable of 2592 x 1944 static or up to 1080p @ 30fps video.

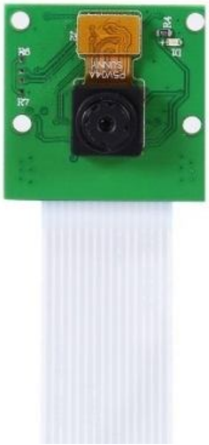


Figure 3: Raspberry Pi Camera Module

Light Sensor

Light Dependent Resistor (LDR) as shown in Figure 4, is a resistor whose resistance is proportional to that of incandescing light. Increase in light give rise to more and more electronics to jump to conduction band thereby conduction electricity.

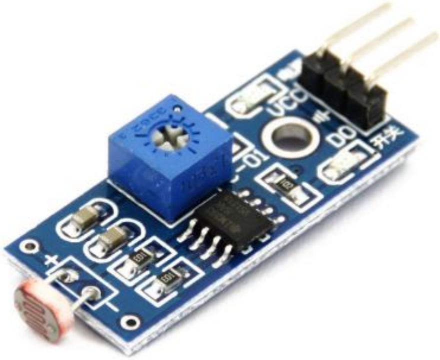


Figure 4: LDR

Ultrasonic sensor

Ultrasonic sensors as shown in Figure 5, as its name suggest use ultrasonic waves to measure distance of object from itself. These inaudible waves can be used to pin-point distances to objects across short distances.

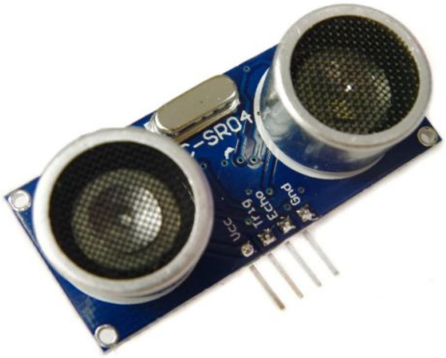


Figure 5: Ultrasonic Sensor

Soil Moisture Sensor

The two probes of moisture sensor act as variable resistors. More moisture will lead to more conductivity. It is pretty forthright in use and is shown in Figure 6.

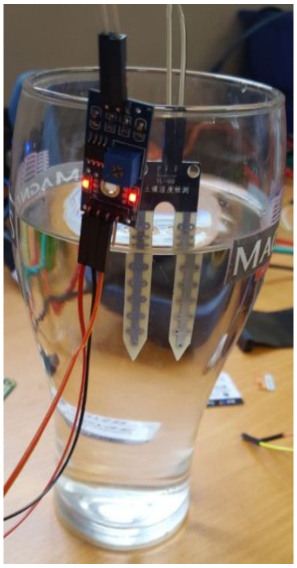


Figure 6: Soil Moisture Sensor

Relay Module

Relay as shown in Figure 7, is an electrically functioned switch. Relays are relatively simple devices which makes it easy for logic-level power to regulate very high-powered devices.

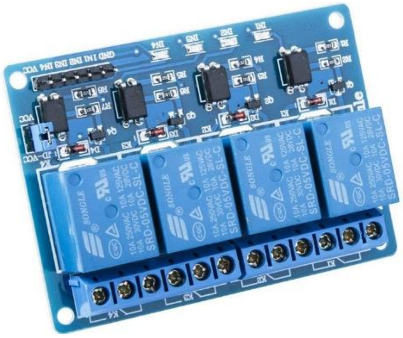
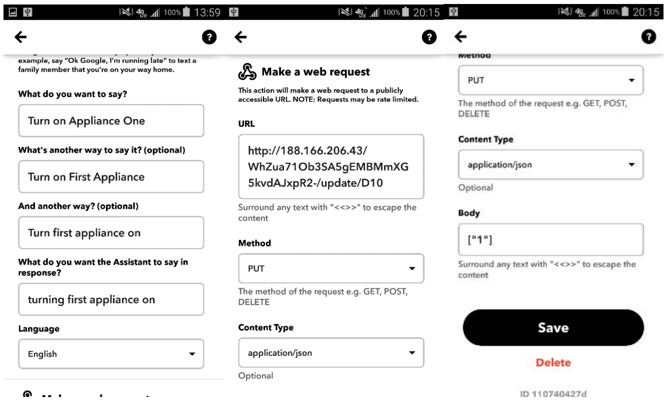


Figure 7: Relay Module

4. IMPLEMENTATION AND WORKING

4.1. Set up IFTTT

Connections are made as schematic diagram shown in Fig. 1. Applets are created to control various appliances in IFTTT App via Google Assistant voice recognition as shown in Figure8. URL must contain the connection key of Blynk to make interaction between apps possible.



13:59

example, say "Ok Google, I'm running late" to text a family member that you're on your way home.

What do you want to say?

Turn on Appliance One

What's another way to say it? (optional)

Turn on First Appliance

And another way? (optional)

Turn first appliance on

What do you want the Assistant to say in response?

turning first appliance on

Language

English

20:15

Make a web request

This action will make a web request to a publicly accessible URL. NOTE: Requests may be rate limited.

URL

http://188.166.206.43/WhZua71Ob3SA5gEMBMmXG5kvdAJxpR2-/update/D10

Surround any text with "<<>>" to escape the content

Method

PUT

The method of the request e.g. GET, POST, DELETE

Content Type

application/json

Optional

20:15

Method

PUT

The method of the request e.g. GET, POST, DELETE

Content Type

application/json

Optional

Body

["1"]

Surround any text with "<<>>" to escape the content

Figure 8: Making an Applet in FTTT App

4.2. Set up Blynk

Various components are set up in the Blynk app as shown in the Figure 9. The switches control the appliances whereas Gauge Indicator and graph indicates LDR brightness levels.

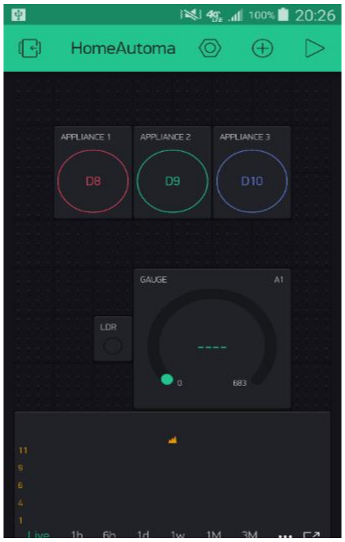


Figure 9: Blynk App Setup

4.3. Set up AWS User

AWS is used to provide cloud services, so an AWS account is required. First a user is set up in AWS IAM. Then the user is provided with access to Amazon Recognition and Amazon S3 services. Creating the user will generate Access key ID and secret keys which will be required in coding of Raspberry Pi. The snapshots for code ca be seen in Figure 10 and Figure 11.

4.4. Set up S3

A Bucket is created in AWS S3 which must have a unique name. Inside the bucket various directories are created; one for each user. Each directory has the name of a user and contains a few of their pictures. These pictures will be used of authentication purpose. The details can be seen in Figure 12, Figure 13, Figure 14 and Figure 15.

4.5. Code

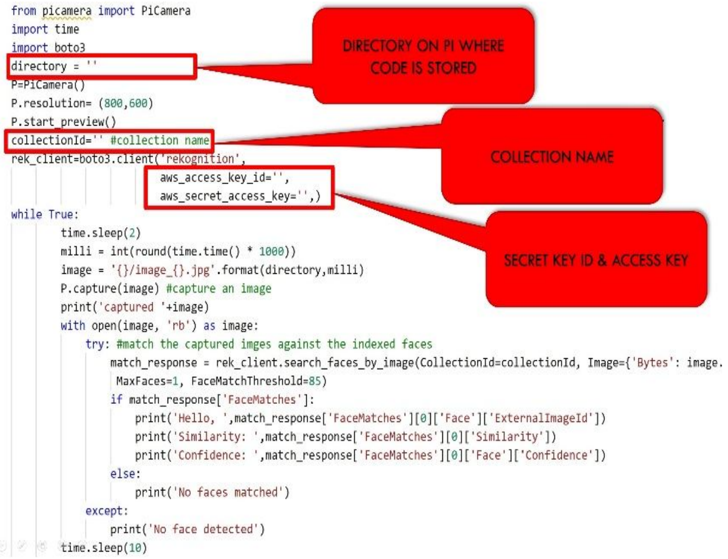
```
import boto3
s3_client = boto3.client(
   's3',
   aws_access_key_id=' ',
   aws_secret_access_key=' ',
)
collectionId=' '
rek_client=boto3.client('rekognition',
                      aws_access_key_id=' ',
                      aws_secret_access_key=' ',)
bucket = ' '
all_objects = s3_client.list_objects(Bucket =bucket )
list_response=rek_client.list_collections(MaxResults=2)
if collectionId in list_response['CollectionIds']:
   rek_client.delete_collection(CollectionId=collectionId)
rek_client.create_collection(CollectionId=collectionId)
for content in all_objects['Contents']:
   collection_name,collection_image =content['Key'].split('/')
   if collection_image:
      label = collection_name
      print('indexing: ',label)
      image = content['Key']
      index_response=rek_client.index_faces(CollectionId=collectionId,
                           Image={'S3Object':{'Bucket':bucket,'Name':image}},
                           ExternalImageId=label,
                           MaxFaces=1,
                           QualityFilter="AUTO",
                           DetectionAttributes=['ALL'])
      print('FaceId: ',index_response['FaceRecords'][0]['Face']['FaceId'])
```

Figure 10: Code for indexing faces index_faces.py

```
from picamera import PiCamera
import time
import boto3
directory = ' '
P=PiCamera()
P.resolution= (800,600)
P.start_preview()
collectionId=' '
rek_client=boto3.client('rekognition',
                    aws_access_key_id='',
                    aws_secret_access_key='',)
while True:
      time.sleep(2)
      milli = int(round(time.time() * 1000))
      image = '{}/image_{}.jpg'.format(directory,milli)
      P.capture(image)
      print('captured '+image)
      with open(image, 'rb') as image:
         try:
            match_response =
rek_client.search_faces_by_image(CollectionId=collectionId,
Image={'Bytes': image.read()}, MaxFaces=1, FaceMatchThreshold=85)
            if match_response['FaceMatches']:
               print('Hello,
',match_response['FaceMatches'][0]['Face']['ExternalImageId'])
               print('Similarity:
',match_response['FaceMatches'][0]['Similarity'])
               print('Confidence:
',match_response['FaceMatches'][0]['Face']['Confidence'])
            else:
               print('No faces matched')
         except:
```

Figure 11: Code for matching faces match_faces.py

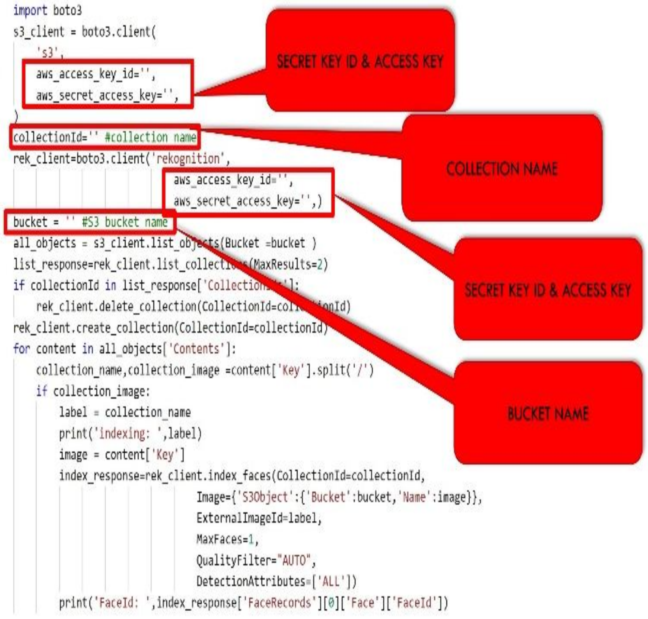
Code must be modified to add values of Bucket Name, Collection ID, Secret ID, and Secret Key.



```
from picamera import PiCamera
import time
import boto3
directory = ''
P=PiCamera()
P.resolution= (800,600)
P.start_preview()
collectionId='' #collection name
rek_client=boto3.client('rekognition',
                        aws_access_key_id='',
                        aws_secret_access_key='',)
while True:
        time.sleep(2)
        milli = int(round(time.time() * 1000))
        image = '{}/image_{}.jpg'.format(directory,milli)
        P.capture(image) #capture an image
        print('captured '+image)
        with open(image, 'rb') as image:
            try: #match the captured imges against the indexed faces
                match_response = rek_client.search_faces_by_image(CollectionId=collectionId, Image={'Bytes': image.
                 MaxFaces=1, FaceMatchThreshold=85)
                if match_response['FaceMatches']:
                    print('Hello, ',match_response['FaceMatches'][0]['Face']['ExternalImageId'])
                    print('Similarity: ',match_response['FaceMatches'][0]['Similarity'])
                    print('Confidence: ',match_response['FaceMatches'][0]['Face']['Confidence'])
                else:
                    print('No faces matched')
            except:
                print('No face detected')
        time.sleep(10)
```

DIRECTORY ON PI WHERE CODE IS STORED
COLLECTION NAME
SECRET KEY ID & ACCESS KEY

Figure 12: Placing values in index_faces.py



```
import boto3
s3_client = boto3.client(
    's3',
    aws_access_key_id='',
    aws_secret_access_key='',
)
collectionId='' #collection name
rek_client=boto3.client('rekognition',
                        aws_access_key_id='',
                        aws_secret_access_key='',)
bucket = '' #S3 bucket name
all_objects = s3_client.list_objects(Bucket =bucket )
list_response=rek_client.list_collections(MaxResults=2)
if collectionId in list_response['CollectionIds']:
    rek_client.delete_collection(CollectionId=collectionId)
rek_client.create_collection(CollectionId=collectionId)
for content in all_objects['Contents']:
    collection_name,collection_image =content['Key'].split('/')
    if collection_image:
        label = collection_name
        print('indexing: ',label)
        image = content['Key']
        index_response=rek_client.index_faces(CollectionId=collectionId,
                                Image={'S3Object':{'Bucket':bucket,'Name':image}},
                                ExternalImageId=label,
                                MaxFaces=1,
                                QualityFilter="AUTO",
                                DetectionAttributes=['ALL'])
        print('FaceId: ',index_response['FaceRecords'][0]['Face']['FaceId'])
```

Figure 13: Placing values in Match_faces.py

```
import BlynkLib
import time
import RPi.GPIO as gpio
led1 = 8
led2 = 9
led3 = 10
ldr = 12
soil = 16
appliance1 = 35
appliance2 = 36
appliance3 = 37
appliance4 = 38
auth = "WhZua71Ob3SA5gEMBMmXG5kvdAJxpR2-"
gpio.setmode(gpio.BOARD)
#to control 3 led's withblynk
gpio.setup(led1, gpio.OUT)
gpio.setup(led2, gpio.OUT)
gpio.setup(led3, gpio.OUT)
#to control the 4-channel relay
gpio.setup(appliance1, gpio.OUT)
gpio.setup(appliance2, gpio.OUT)
gpio.setup(appliance3, gpio.OUT)
gpio.setup(appliance4, gpio.OUT)
#to read the state of ldr sensor and soil sensor
gpio.setup(ldr, gpio.IN)
gpio.setup(soil, gpio.IN)
#start a connection between python code and blynk
blynk = BlynkLib.Blynk(auth)
#sending data of ldr sensor to blynk at virtual pin v1
blynk.virtualWrite(V1, gpio.input(ldr));
#sending data of soil sensor to blynk at virtual pin v2
blynk.virtualWrite(V1, gpio.input(ldr));
```

Figure 14: Code for running appliances and LDR

```
import time
import RPi.GPIO as gpio
GPIO.setmode(GPIO.BCM)
TRIG = 13
ECHO = 16

GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)

def ultra():
   GPIO.output(TRIG,False)
   GPIO.setwarnings(False)
   time.sleep(0.1)
   pulse_end = 0
   pulse_start = 0
   distance = 0
   GPIO.output(TRIG,True)
   time.sleep(0.00001)
   GPIO.output(TRIG,False)
   while GPIO.input(ECHO)==0:
      pulse_start = time.time()

   while GPIO.input(ECHO)==1:
      pulse_end = time.time()

   pulse_duration = pulse_end - pulse_start

   distance=pulse_duration*17150
   distance=round(distance,2)
   print(distance)
   if distance>2 and distance<400:
      return distance
```

Figure 15: Code for running Ultrasonic sensor

The voice command is received by Google Assistant which it forwards to IFTTT App. IFTTT server analyzes the voice commands and matches it to users store of commands, if a match is found it forwards the specific command to Blynk App which sends to data over server to Raspberry Pi which is using authentication key for connection. Raspberry Pi then turns the respective pin to High Output in turning the device on.

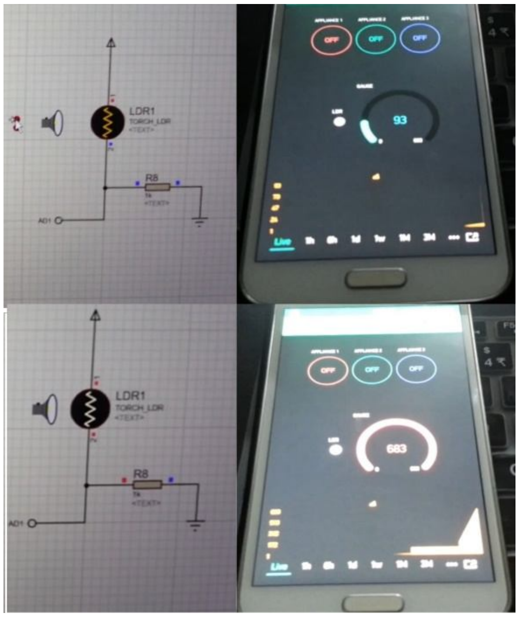


Figure 16: Simulating LDR and its reflection on Blynk App

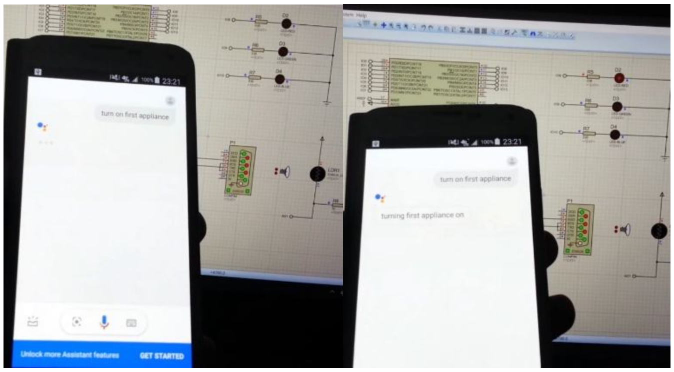
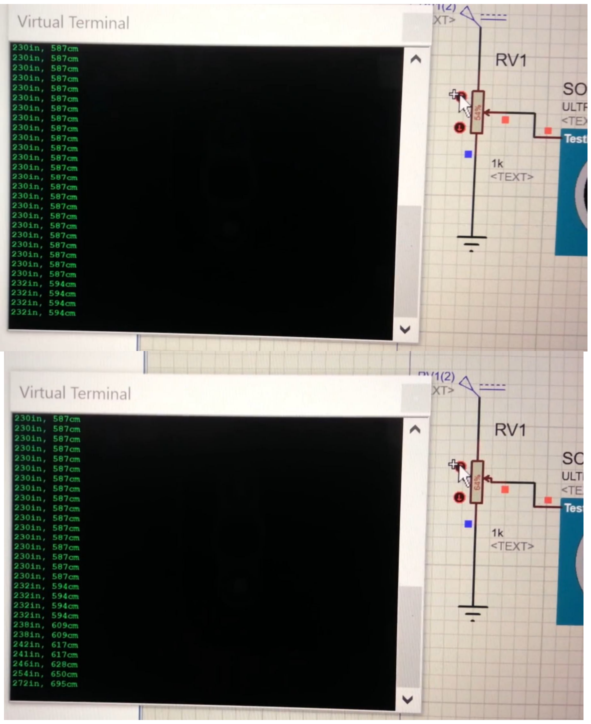


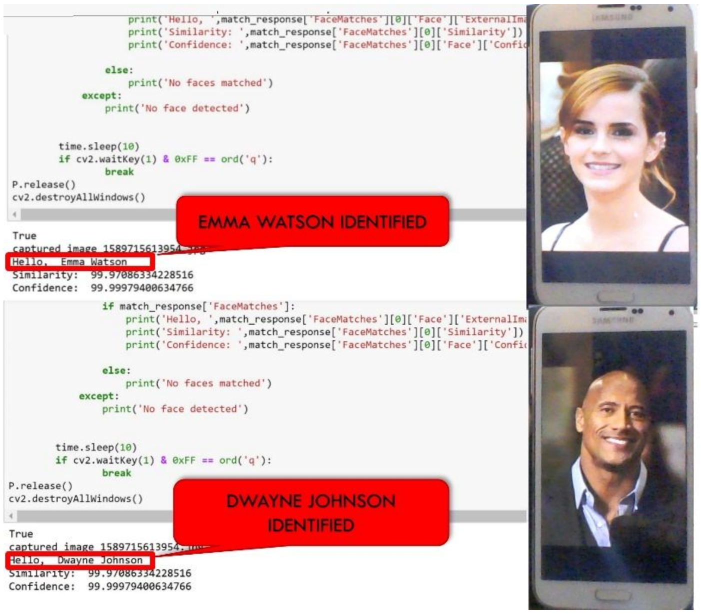
Figure 17: Simulating Appliance control via Vocal Commands using Google Assistant



```
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
232in, 594cm
232in, 594cm
232in, 594cm
232in, 594cm
```

```
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
230in, 587cm
232in, 594cm
232in, 594cm
232in, 594cm
232in, 594cm
238in, 609cm
238in, 609cm
242in, 617cm
241in, 617cm
246in, 628cm
254in, 650cm
272in, 695cm
```

Figure 18: Simulating working of Ultrasonic sensor



```
                print('Hello, ',match_response['FaceMatches'][0]['Face']['ExternalIma
                print('Similarity: ',match_response['FaceMatches'][0]['Similarity'])
                print('Confidence: ',match_response['FaceMatches'][0]['Face']['Confi

            else:
                print('No faces matched')
        except:
            print('No face detected')


    time.sleep(10)
    if cv2.waitKey(1) & 0xFF == ord('q'):
            break
P.release()
cv2.destroyAllWindows()
```

```
True
captured image 1589715613954.jpg
Hello,  Emma Watson
Similarity:  99.97086334228516
Confidence:  99.99979400634766
```

```
            if match_response['FaceMatches']:
                print('Hello, ',match_response['FaceMatches'][0]['Face']['ExternalIma
                print('Similarity: ',match_response['FaceMatches'][0]['Similarity'])
                print('Confidence: ',match_response['FaceMatches'][0]['Face']['Confi

            else:
                print('No faces matched')
        except:
            print('No face detected')


    time.sleep(10)
    if cv2.waitKey(1) & 0xFF == ord('q'):
            break
P.release()
cv2.destroyAllWindows()
```

```
True
captured image 1589715613954.jpg
Hello,  Dwayne Johnson
Similarity:  99.97086334228516
Confidence:  99.99979400634766
```

Figure 19: Recognizing people using AWS [Inset: what camera sees]

Ultrasonic sensor measures the distance at front door. If it reaches below a threshold implies that someone's at the door which in turn turns on the camera to verify the identity of the person. The snapshots for simulation can be seen in Figure 16, Figure 17 and Figure 18.

Database of all authorized faces is stored in AWS S3 (Simple Storage Service) with a directory corresponding to each person. Amazon Recognition is a service provided by AWS which uses deep learning technology to match images provided by Raspberry Pi camera to the database of images in S3. If a match is found, then a message welcoming the user is printed. This is used to control the door to open it to only authorized people as shown in Figure 19.

5. CONCLUSION

This project aims to design and develop an economical and efficient home automation system which is capable of controlling home appliances based on user's commands via smartphone. It also responses to vocal commands using Google Assistant.

The developed project has used AWS cloud service to store the database of allowed users and authenticates people by taking photos via camera and matching them with database using cloud computing. The system makes it easy and convenient to automate your house.

Applications:

- The main application to automate basic household tasks and control home appliances based on user commands.
- It uses internet so the physical distance between user and the system does not matter.
- It authenticates people entering the house by taking their pictures and matching them with already stored database using machine learning on cloud service.
- It is a flexible system that can be used to control and automate many household appliances by interacting with vast array of sensors and can be programmed with multiple programming languages.
- The system will be controlled via Blynk app which can receive vocal commands from Google Assistant
- The system is cheap, easy and convenient to use.

REFERENCES

1. M. E. Haque, M. R. Islam, M. T. Fazle Rabbi and J. I. Rafiq. **IoT Based Home Automation System with Customizable GUI and Low-Cost Embedded System**. *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), Dhaka, Bangladesh*, 2019, pp. 1-5, doi: 10.1109/STI47673.2019.9068035.
2. https://realty.economictimes.indiatimes.com/news/residential/smart-home-market-to-grow-at-30-to-double-revenue-every-3-years/50106530
3. E. Rohadi et al. **Internet of Things: CCTV Monitoring by Using Raspberry Pi**. *2018 International Conference on Applied Science and Technology (ICAST), Manado, Indonesia*, pp. 454-457,2018. https://doi.org/10.1109/iCAST1.2018.8751612
4. A. R. Biswas and R. Giaffreda. **IoT and cloud convergence: Opportunities and challenges**, *2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul*, pp. 375-376, 2014.
5. Bloomberg. (2013, Jun. 6). *Behind the 'Internet of Things' Is Android—and It's Everywhere* [Online].

https://www.bloomberg.com/news/articles/2013-05-30/behind-the-internet-of-things-is-android-and-its-everywhere

6. Li, Rita Yi Man and Li, Herru and Mak, Cho and Tang, Tony. **Sustainable Smart Home and Home Automation: Big Data Analytics Approach** *International Journal of Smart Home*, Vol. 10, No.8, pp. 177-187, 2016. https://doi.org/10.14257/ijsh.2016.10.8.18
7. Pandey, Sakshi and Jaiswal, Shanu and Yadav, Nitin and Sonawane, Jayashree. **IoT Based Home Automation and Analysis Using Machine Learning**. 2019.
8. Maharaja, Amrita and Ansari, Namrata. **Voice Controlled Automation Using Raspberry Pi.** *2nd International Conference on Advances in Science & Technology (ICAST)* 2019. https://doi.org/10.2139/ssrn.3366895
9. Jain, Nikhil and Desai, Mitali and P. Rana, Dipti. **Empirical Comparison of IoT Based Smart Home Models**. *International Journal of Computational Intelligence & IoT*, Vol. 2, No. 2, 2019.
10. Madhugundu, Dhanunjaya Kumar and Ahmed, Fatema and Roy, Bholanath. **A Survey on Security Issues and Challenges in IoT Based Smart Home**. *Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT), 2018 held at Malaviya National Institute of Technology, Jaipur* (India) 2018.
11. W N Hussein, L M Kamarudin, M R Hamzah, H N.Hussain, K J Jadaa. **A Methodology for Big Data Analytics and IoT-Oriented Transportation System for future implementation**. *International Journal of Emerging Trends in Engineering Research,* Vol. 7, No. 11, pp 449-453, 2019. https://doi.org/10.2139/ssrn.3167433
12. K. Ruth ramya, B. Manjula josephine, K. Durga praveen, M. Bala maruthi, Ch.sai kumar. **An Efficient and Secured Biometric Authentication for IoT**. *International Journal of Emerging Trends in Engineering Research,* vol. 7, No. 11, pp 604-609, 2019. https://doi.org/10.30534/ijeter/2019/327112019