

HOME AUTOMATION PROJECT

(Using Arduino uno)

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT FOR THE DEGREE OF

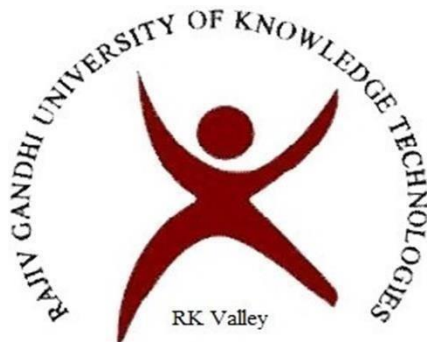
**BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE ENGINEERING**

By

MORAY HARINI- (R171011)

Under the Esteem Guidance of

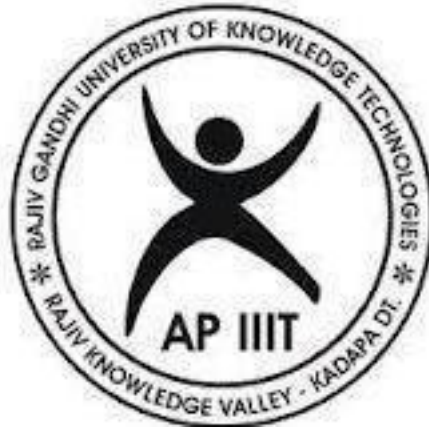
Ms.C.Suneetha



DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Rajiv Gandhi University of Knowledge and Technologies - R.K.Valley
Kadapa, Andra Pradesh-516330

Certificate of University



This is to certify that the **home automation project** is a record of the bonafide work done by **MORAY HARINI(R171011)** submitted in partial fulfilment of the requirement for the award of the Degree of Bachelor of Technology (B.Tech) in Computer Science of Rajiv Gandhi University of Knowledge and Technologies Rk valley, during the academic year 2022-23.

MS.C.Suneetha,
Project Co-ordinator,
Computer Science & Engineering,
Rgukt-Rk valley.

MR.SATYANANDARAM
Head of the department, CSE
Rgukt-Rk valley.

Declaration

I am, MORAY HARINI hereby declares that this report entitled “ Intern Home automation ” submitted by me under the guidance and supervision of Ms.Suneetha is a bonafide work. I also declare that it has not been submitted previously in part or in full to this university or other university or institution for the award of any degree or diploma in university.

Date: 20-04-23

MORAY HARINI(R171011)

Place: RK Valley

Acknowledgment

I would like to express our sincere gratitude to Ms.Suneetha , our project internal guide for valuable suggestions and keen interest throughout the progress of our course of internship.

I am grateful to Mr.Satyanandaram HOD CSE , for providing excellent computing facilities and a congenial atmosphere for progressing with our project.

I would like to express our special thanks of gratitude to **Mr.Munawar** (Trainee Teacher)for their able guidance and support in completing our Project.

At the outset, I would like to thank Rajiv Gandhi University of Knowledge Technologies, RK Valley for providing all the necessary resources for the successful completion of my course work. At last, but not the least We thank our teammates, our classmates and other students for their physical and moral support.

ABSTRACT

We live in an exciting time where more and more everyday items “things” are becoming smart! “Things” have sensors and can communicate with other “things” and can provide control to more “things”. The Internet of Things, IoT, is upon us in a huge way and people are rapidly inventing new gadgets that enhance our lives. The price of microcontrollers with the ability to talk over a network keeps dropping and developers can now think and build things inexpensively.

This IoT based home automation project is done using low-cost custom build ESP8266 WiFi Module, and Raspberry pi zero. A system that uses mobile app to control basic home functions and features automatically through internet from anywhere around the world, an automated home is sometimes called a smart home.

This project provides significant electric power saving solution at homes and at the offices with help of dynamic control of electric supply over internet using smart switch, and another is a lowcost wireless plug and use surveillance system with live streaming over internet connection and in-build video storage capacity.

Project contains both hardware and software development where hardware program is built using C and python language on Arduino IDE, whereas mobile application is built using java language Android studio.

Table of Contents

Contents			
			Page No.
Chapter 1		INTRODUCTION	7-8
	1.1	Introduction	7
	1.2	Overview of the Project	7
	1.3	Problem Statement	7
	1.4	Components Required	7
Chapter 2		BACKGROUND MATERIAL	8
	2.1	Conceptual Overview	8-10
	2.2	Technologies Involved	10-16
Chapter 3		IMPLEMENTATION	17
	3.1	Modules	17
	3.2	Source code	18-20
	3.3	Results and discussion	20-23
Last Chapter		CONCLUSION & REFERENCES	23
	4.1	Conclusion Future scope	23-24
REFERENCES			24

INTRODUCTION

Nowadays, we have remote controls television sets and other electronic systems which have made our lives real easy. We have come up with a new system called Arduino based home automation using Bluetooth. This system is super cost effective and can give the user the ability to control any electronic device without even spending a remote control. This project helps the user to control all the electronic devices using smart phone.

Overview of the Project

The objective of this project is to design and build new IoT hardware which can be control using mobile application from anywhere around the world and that can installed at every home and office at lowest possible cost

Problem Statement

Today people are looking at ways and means to better their life-style using the latest technologies that area available. Any new facility or hope appliance that promises to enhance their life-style is grabbed by the consumers. The more such facilities and appliances are added, it becomes inevitable to have easy and convenient methods and means to control and operate these appliances. Conventional wall switches are located in different parts of a house and thus necessitates manual operations like to switch on or off these switches to control various appliances. It gets virtually impossible to keep track of appliances that are running and also to monitor their performance.

Components Required

- 1.Power supply
- 2.Load(bulb 220V)
- 3.Connecting wires & Sensors
- 4.Rasberry pie board
- 5.smart phone(Bluetooth enabled)
- 6.Bluetooth module RN-42

BACKGROUND MATERIAL

Configuration of Raspberry Pi 4 on a Laptop

There are different ways of configuring Raspberry Pi either on a laptop or using a desktop (Monitor) which is suitable for HDMI ports.

Here we are configuring Raspberry Pi 4 on a Laptop by using the below-listed components: - 1) Raspberry Pi 4

2) SD-Card

3) Card-Reader

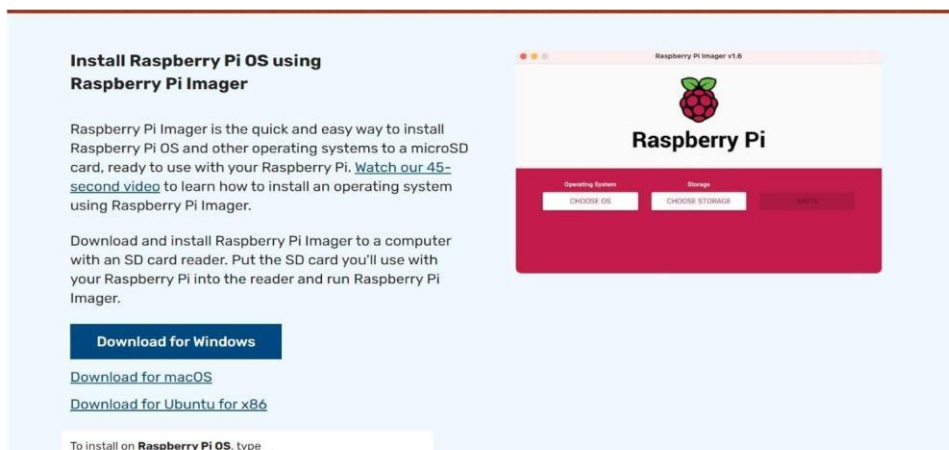
4) Raspberry Pi 4 official USB-C power supply

5) Laptop or Desktop

Step 1: - Install Raspberry Pi Image Software using the below link

<https://www.raspberrypi.com/software/>

After opening the link it looks as seen in the below image



Step 2: After completion of the installation insert the SD card and open the Raspberry Pi imager.

Install Raspberry Pi OS using Raspberry Pi Imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi. [Watch our 45-second video](#) to learn how to install an operating system using Raspberry Pi Imager.

Download and install Raspberry Pi Imager to a computer with an SD card reader. Put the SD card you'll use with your Raspberry Pi into the reader and run Raspberry Pi Imager.

[Download for Windows](#)

[Download for macOS](#)

[Download for Ubuntu for x86](#)

To install on **Raspberry Pi OS**, type



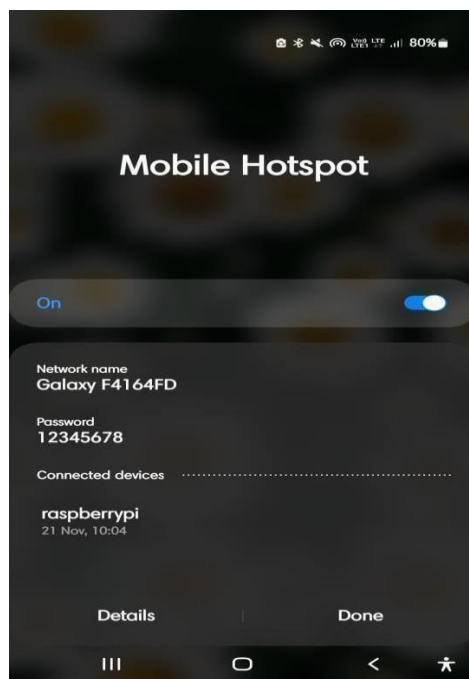
Step 2: - After completion of the installation insert the SD card and open the Raspberry Pi imager.

Choose the Operating System that you want to install on SD Card that you can use when inserted into the Raspberry Pi board. While installing the operating follow the procedure in reference no 4.

Then choose the storage that is your SD Card.

Step 3: - After installation, the size of the SD Card will be reduced need not worry about it. You can remove the SD Card from the PC. Then insert it again and open the SD Card on the PC. You can see some files installed. Create a “.txt” file and then rename it as “SSH”. Then create a “wpa_supplicant.conf” file. To create the file follow the procedure in the below link from Step 2 to Step4

Step 4: - Now after the completion of step 3 insert the SD Card into the Raspberry Pi board and power it using a power supply cable. Make sure that the pc that you are using and the Wi-Fi details provided in “wpa_supplicant.conf” are the same otherwise, you cannot connect to the Raspberry. You can also the after the Raspberry Pi is connected to the Wi-Fi.



Step 5: - Now open the command prompt on the PC. And then type the command “ssh pi@username” and then type the “password”. Need to worry that the password you are trying might not be visible.

```
Command Prompt
ED25519 key fingerprint is SHA256:GDbWBhuZ55mJY6+xxvD9kpkhhvmLL51h16Z0NLb4JQho.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'raspberrypi' (ED25519) to the list of known hosts.
pi@raspberrypi's password:
Linux raspberrypi 5.15.61-v7l+ #1579 SMP Fri Aug 26 11:13:03 BST 2022 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~$ sudo apt update
Get:1 http://raspbian.raspberrypi.org/raspbian bullseye InRelease [15.0 kB]
Get:2 http://archive.raspberrypi.org/debian bullseye InRelease [23.6 kB]
Get:3 http://raspbian.raspberrypi.org/raspbian bullseye/main armhf Packages [13.2 MB]
Get:4 http://archive.raspberrypi.org/debian bullseye/main armhf Packages [311 kB]
Fetched 13.6 MB in 18s (755 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
46 packages can be upgraded. Run 'apt list --upgradable' to see them.
pi@raspberrypi:~$ client_loop: send disconnect: Connection reset

C:\Users\bhopa>
```

Technologies involved

- NodeMCU ESP8266 Wi-Fi module.
Type - 32-bit micro-controller
CPU - 160MHz
GPIO -16 pin
Memory - 4MB
- 4 two-way switches.
- 4 port electromagnet relay.
- 220 Volt to 5 Volt AC to DC power adapter.

Software used to program ESP8266 microcontroller

- Arduino IDE using C language.

Development of mobile application.

- Android Studio 3.1.1 - IntelliJ platform.

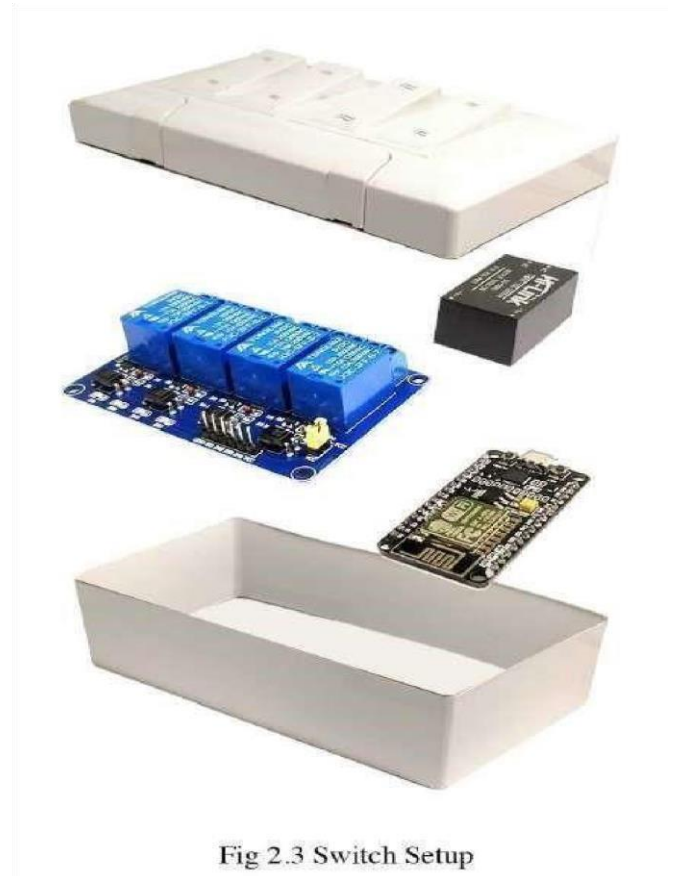


Fig 2.3 Switch Setup

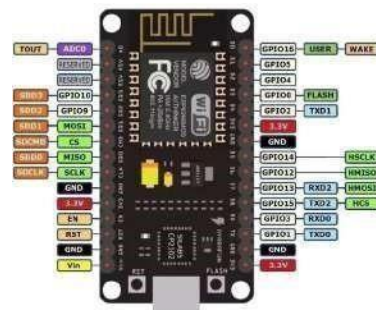
Device	Price
Smart-Switch	550 Rs (4 switch)
IP-Based Surveillance	3,000 Rs (per unit with 24 hours video storage)

NodeMCU ESP8266 Wi-Fi module ^[1]

The NodeMCU (Node MicroController Unit) is an open source software and hardware development environment that is built around a very inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains all crucial elements of the modern computer: CPU, RAM, networking (Wi-Fi), and even a modern operating system and SDK. When purchased at bulk, the ESP8266 chip costs only \$2 USD a piece. That makes it an excellent choice for this project.



NodeMCU



NodeMCU architecture

Raspberry Pi Zero

The Raspberry Pi is a popular Single Board Computer (SBC) in that it is a full computer packed into a single board. Many may already be familiar with the Raspberry Pi 3 and its predecessors, which comes in a form factor that has become as highly recognizable. The Raspberry Pi comes in an even smaller form factor. The introduction of the Raspberry Pi Zero allowed one to embed an entire computer in even smaller projects. This project will use the latest version of the Zero product line, the Raspberry Pi Zero - Wireless, which has an onboard Wi-Fi module.

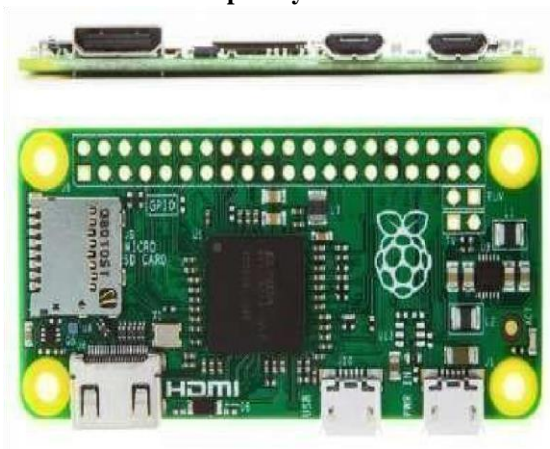
"IMX219", Type 1/4.0, approximately 8.08M effective pixels, back-illuminated CMOS image sensor for the growing mobile market. The IMX219 satisfies the fulfillment performance of "high sensitivity" and "high frame rate imaging" to meet the demands for high-quality camera applications, and can be easily put on camera fronts of stylish and slim bezel devices achieved by the reduced size.

Not only the normal rate of all-8M pixels at full field of view and 30 frame/s, 4 times faster imaging is also possible by 2×2 analog binning mode. Also, a LSC function*1 corrects optical unevenness to adjust during the module fabrication process, and contributes to suppressing of system cost. Additionally, combining with a rear-end ISP supporting of BME-HDR*2 enables future scalability of the high-dynamic-range video imaging. Image sensor is perfect for video surveillance with low power consumption and high quality video. This makes image sensor perfect fit for this project.

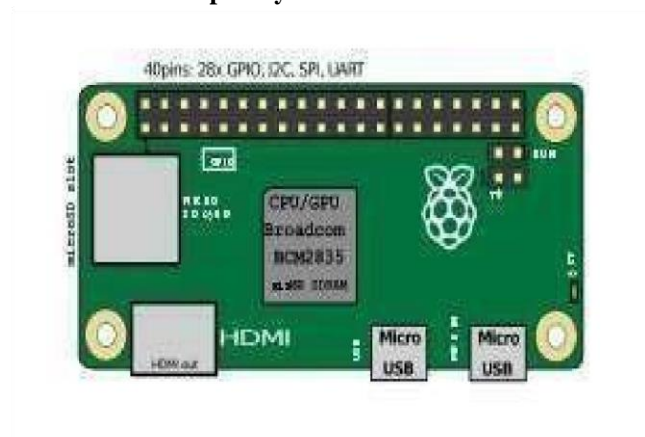


Image Sensor

Raspberry Pi Zero



Raspberry Pi Zero Architecture



Android Mobile Application

About half of the population around the world prefers Android over another operating systems or devices. IoT is one of the biggest areas where Android app development technology is continuously contributing with its enormous benefits. Internet of Things is generally recognized as the interconnectedness of different smart devices over the Internet. The devices make use of sensors and internet connectivity, which helps them receive, collect and transmit information. So development on android platform is the right option for this project.

Smart-Switch application named Smart-Fox is developed on android platform with API 19 compatible which covers 90.1% android device of the total devices.

List of support used on android device to run mobile application.

- Wi-Fi.
- Internet GSM.
- Location.

Hardware used to build Surveillance Camera

- Raspberry Pi Zero W. o 1GHz, single-core CPU o 512MB RAM o Mini HDMI and USB On-The-Go ports o Micro USB power o HAT-compatible 40-pin header o Composite video and reset headers o CSI camera connector
- Sony IMX219 image sensor.

Software used to build Surveillance Cam

- 2.7 version python platform.
- OpenCV.

Applications

- Can be used for making supercomputers
- Solar Raspberry Pi Power Pack.
- It can make your old TV into a smart TV. (You can play videos, 3D Games, Music, Browse Internet, and much more).
- Raspberry Pi can Act as a Full HD 1080p Media Player.
- You can connect a Monitor, Keyboard, and Mouse and use it as a normal computer.



Fig 2.7 Cam Setup

Disadvantages

- It does not have a Hard Disk associated with it for permanent storage of files, we have to connect one externally or have to use an SD card for the purpose.
- The RAM is a POP package on top of the SoC, so it's not removable or swappable.
 - There is no Real Time Clock associated with the board. Adding an RTC is expensive. You can add one yourself using the GPIO pins.

Hardware

- 1) 10/100 BaseT Ethernet socket
- 2) HDMI socket
- 3) USB 2.0 socket
- 4) RCA video
- 5) SD card socket
- 6) Powered micro-USB socket
- 7) 3.5mm audio out jack
- 8) Header footprint for camera connection



Operating System

- Linux on a bootable SD card
Fedora

Raspbian
Debian
ArchLinux ARM

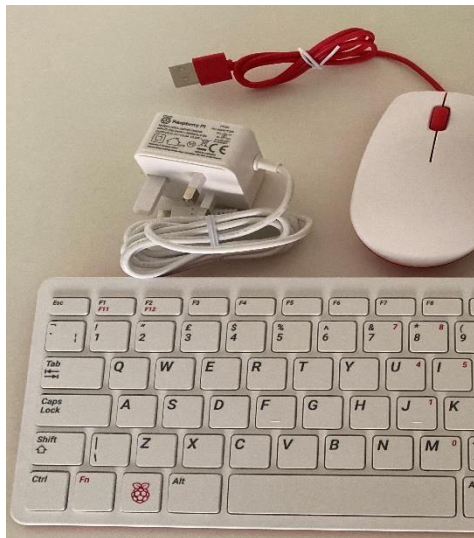
Features

- ✚ Ultra-low-cost (Model A \$25, Model B \$35)
- ✚ Ultra-low-power ~ 1W
- ✚ Credit-card sized, fanless, instant start-up
- ✚ Complete easy-to-program computer

Technology

- ✚ The Raspberry Pi has a Broadcom BCM2835 system on a chip (SoC), which includes an ARMI 176JZF-S 700 MHz processor.
- ✚ Video core IV GPU
- ✚ Originally shipped with 256 megabytes of RAM, later upgraded to 512 MB.

It does not include a built-in hard disk but uses an SD card for booting and long-term storage.



The Raspberry Pi
400 complete kit

The Model 2B boards incorporate four USB ports for connecting peripherals

Raspberry Pi 1 Model A, with an HDMI port and a standard RCA composite video port for older displays.

Programming Language used during project

Project has used three different programming languages for our project. For the development of the application on android, we have used Java Platform. Android Software Development kit incorporates IntelliJ software where Java programming is performed. IntelliJ software is used to write the codes for the application under Java Platform. Raspbian OS is used at the raspberry pi.

Java: Java is a set of several computer software products and specifications from Oracle Corporation that provides a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones on the low end, to enterprise servers and supercomputers on the high end.

Python: Python is an interpreter, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. Python combines remarkable power with very clear syntax. It has interfaces too many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. Python is a high-level general-purpose programming language that can be applied to many different classes.

C: For embedded programming of NodeMCU for Smart Switch using Arduino IDE. It is a non object-oriented high-level programming language.

Raspbian OS: Raspbian is a free Operating System based on Debian optimized for the raspberry pi hardware. Raspbian comes with more than 35000 packages; pre-combined software bundled in a nice format for easy installation on Raspberry pi.

Implementation

Modules

Bluetooth module RN-42 :

This module from Roving Networks is powerful, small, and very easy to use. The end user just sees serial characters being transmitted back and forth. RN-42 is a Class 2 device meaning its range is about 50 to 60 feet and correspondingly the power consumption is also low. The RN-42 is perfect for short range, battery powered applications. The RN-42 uses only 26uA in sleep mode while still being discoverable and connectable. Supporting multiple Bluetooth profiles such as SPP and HID and simple UART hardware interface, it is simple to integrate into an embedded system or simply connect to an existing device.

Arduino:

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. It is loaded with the huge collection of in-built libraries. It is very much suitable for multi-tasking application as well as for the function programming and works very well with Master-slave environment.

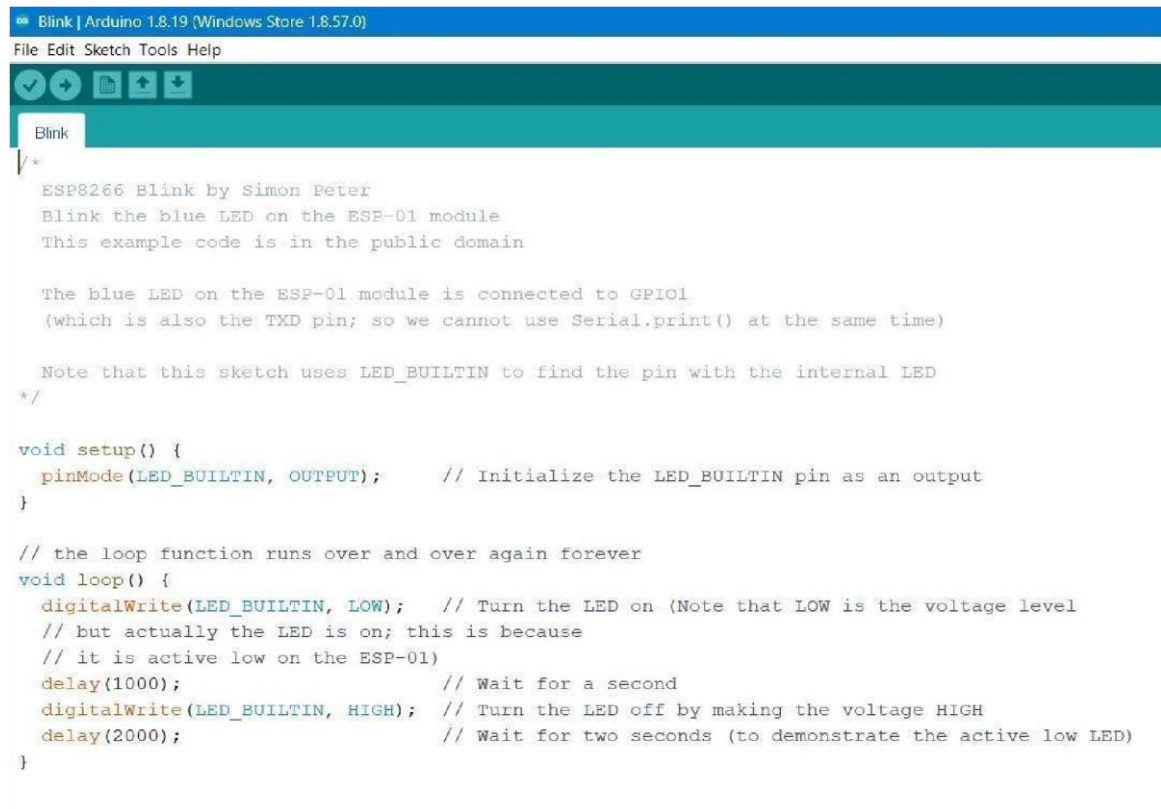
Android:

It is a Linux-based operating system designed primarily for touch screen mobile devices such as smart phones and tablet computers. Google releases the Android code as open source, under the Apache License. The Android Open

Source Project (AOSP), led by Google, is tasked with the maintenance.

Source Code

Experiment 1:- Blink Led



```

Blink | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

Blink

/*
  ESP8266 Blink by Simon Peter
  Blink the blue LED on the ESP-01 module
  This example code is in the public domain

  The blue LED on the ESP-01 module is connected to GPIO1
  (which is also the TXD pin; so we cannot use Serial.print() at the same time)

  Note that this sketch uses LED_BUILTIN to find the pin with the internal LED
  */

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);    // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, LOW);  // Turn the LED on (Note that LOW is the voltage level
  // but actually the LED is on; this is because
  // it is active low on the ESP-01)
  delay(1000);                     // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH); // Turn the LED off by making the voltage HIGH
  delay(2000);                     // Wait for two seconds (to demonstrate the active low LED)
}

```

Experiment 2: - Wi-Fi Scan

In this experiment, we are going to see all the Wi-Fi networks available in nearby surrounding to us using ESP8266.

```
WiFiScan | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

WiFiScan

/*
 * This sketch demonstrates how to scan WiFi networks.
 * The API is almost the same as with the WiFi Shield library,
 * the most obvious difference being the different file you need to include:
 */

#include <ESP8266WiFi.h>

void setup() {
  Serial.begin(115200);
  Serial.println(F("\nESP8266 WiFi scan example"));

  // Set WiFi to station mode
  WiFi.mode(WIFI_STA);

  // Disconnect from an AP if it was previously connected
  WiFi.disconnect();
  delay(100);
}

void loop() {
  String ssid;
  int32_t rssi;
  uint8_t encryptionType;
  uint8_t* bssid;
  int32_t channel;
  bool hidden;
  int scanResult;

  Serial.println(F("Starting WiFi scan..."));
  scanResult = WiFi.scanNetworks(/*async=*/false, /*hidden=*/true);
  if (scanResult == 0) {
    Serial.println(F("No networks found"));
  } else if (scanResult > 0) {
    Serial.printf(PSTR("%d networks found:\n"), scanResult);

    // Print unsorted scan results
    for (int8_t i = 0; i < scanResult; i++) {
      WiFi.getNetworkInfo(i, ssid, encryptionType, rssi, bssid, channel, hidden);

      Serial.printf(PSTR("  %02d: [CH %02d] [%02X:%02X:%02X:%02X:%02X:%02X] %ddBm %c %c %s\n"),
        i,
        channel,
        bssid[0], bssid[1], bssid[2],
        bssid[3], bssid[4], bssid[5],
        rssi,
        (encryptionType == ENC_TYPE_NONE) ? ' ' : '**',
        hidden ? 'H' : 'V',
        ssid.c_str());

      delay(0);
    }
  } else {
    Serial.printf(PSTR("WiFi scan error %d"), scanResult);
  }
  // Wait a bit before scanning again
  delay(5000);
}
```

Experiment 3: - Access Point

The access point is a device that creates a wireless local area network or WLAN, usually in the office or large building. An access point connects to a wired router, switch, or hub via an Ethernet cable and projects a WiFi signal to a designated area

```
WiFiAccess_Point | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

WiFiAccess_Point
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>

/* Set these to your desired credentials: */
const char *ssid = "ESPap"; // You can change it according to your ease
const char *password = "thereisnospoon"; // You can change it according to your ease

ESP8266WebServer server(80); // establishing server at port 80 (HTTP protocol's default port)
// Writing a simple HTML page.
char HTML[] = "<html><body><h1>For ESP8266 tutorials visit</h1><h2>www.techiesms.com</h2><a href='\"http://192.168.4.1/1\"'>Go to Page 1 </a> <button><a href='\"toggle\"'>togg
// This function will be called whenever anyone requests 192.168.4.1 within the local area connection of this ESP module.
void handleRoot()
{
  server.send(200, "text/html", HTML);
}
// This function will be called whenever anyone requests 192.168.4.1/1 within the local area connection of this ESP module.
void Page1()
{
  server.send(200, "text/html", "<h1>techiesms</h1><h2>explore | learn | share </h2>");
}
// This function will be called whenever anyone requests 192.168.4.1/toggle within the local area connection of this ESP module.
void toggle()
{
  digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
  server.send(200, "text/html", HTML);
}

void setup() {
  delay(1000);
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(115200);
  Serial.println();
  Serial.print("Configuring access point...");
  /* You can remove the password parameter if you want the AP to be open. */
  WiFi.softAP(ssid, password); // --> This line will create a WiFi hotspot.

  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
  server.on("/", handleRoot);
  server.on("/1", Page1);
  server.on("/toggle", toggle);
  server.begin();
  Serial.println("HTTP server started");
}

void loop() {
  server.handleClient();
}
```

Results and Discussion

Experiment 1:



Experiment 2:

Now you are ready to verify and upload the code. Then you can see the results on the serial monitor.

```
COM5
Starting WiFi scan...
9 networks found:
00: [CH 01] [38:17:C3:33:56:C0] -67dBm * V srmmap-univfi
01: [CH 01] [38:17:C3:33:56:C1] -66dBm * V srmmap-byod
02: [CH 01] [38:17:C3:33:56:C2] -66dBm * V SRMAP-DIC
03: [CH 01] [38:17:C3:33:56:C3] -68dBm * H
04: [CH 11] [38:17:C3:33:64:A0] -76dBm * V srmmap-univfi
05: [CH 11] [38:17:C3:33:59:22] -86dBm * V SRMAP-DIC
06: [CH 11] [38:17:C3:33:71:E0] -85dBm * V SRMAP-DIC
07: [CH 11] [38:17:C3:33:71:E2] -82dBm * V srmmap-byod
08: [CH 11] [38:17:C3:33:64:A2] -74dBm * V SRMAP-DIC
Starting WiFi scan...
9 networks found:
00: [CH 01] [38:17:C3:33:56:C0] -68dBm * V srmmap-univfi
01: [CH 01] [38:17:C3:33:56:C1] -66dBm * V srmmap-byod
02: [CH 01] [38:17:C3:33:56:C2] -67dBm * V SRMAP-DIC
03: [CH 01] [38:17:C3:33:56:C3] -68dBm * H
04: [CH 11] [38:17:C3:33:64:A1] -74dBm * V srmmap-byod
05: [CH 11] [38:17:C3:33:64:A2] -74dBm * V SRMAP-DIC
06: [CH 11] [24:F2:7F:DB:F1:67] -94dBm * V SRMAP-DIC
07: [CH 11] [24:F2:7F:DB:F1:63] -92dBm * V SRMAP-DIC
08: [CH 11] [38:17:C3:33:59:21] -89dBm * V srmmap-byod
Starting WiFi scan...
9 networks found:
00: [CH 01] [38:17:C3:33:56:C0] -69dBm * V srmmap-univfi
01: [CH 01] [38:17:C3:33:56:C2] -68dBm * V SRMAP-DIC
02: [CH 01] [38:17:C3:33:56:C1] -69dBm * V srmmap-byod
03: [CH 01] [38:17:C3:33:56:C3] -68dBm * H
04: [CH 11] [38:17:C3:33:64:A2] -76dBm * V SRMAP-DIC
05: [CH 11] [38:17:C3:33:59:22] -82dBm * V SRMAP-DIC
06: [CH 11] [38:17:C3:33:71:E0] -88dBm * V SRMAP-DIC
07: [CH 11] [38:17:C3:33:71:E3] -86dBm * H
08: [CH 01] [38:17:C3:33:54:02] -87dBm * V SRMAP-DIC
Starting WiFi scan...

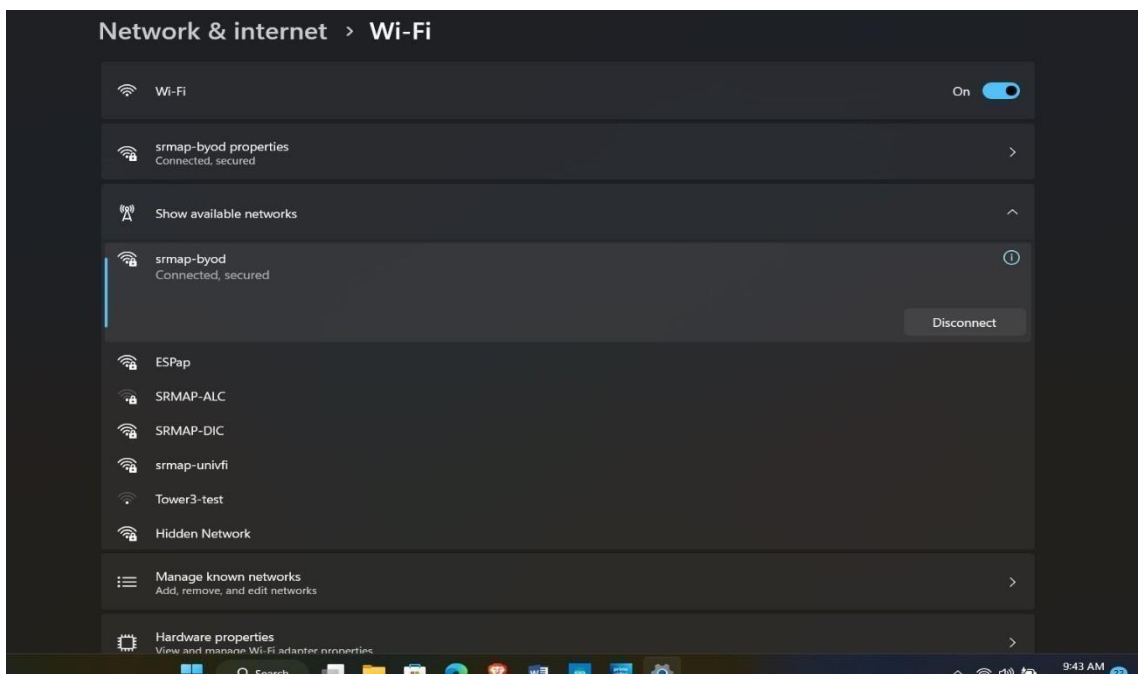
Autoscroll ☒ Show timestamp Newline 115200 baud Clear output
```

You need to be careful after the uploading program into the ESP8266 board because if the baud rate given in the code and the baud rate seen on the serial monitor are not the same then we would not get proper results. And you can see the baud rate given in the code is matching to the baud rate seen on the serial monitor is same. And we are getting the result we want.

Experiment 3:

Now you are ready to verify and upload the code. To see the results you need to connect it to the Wi-Fi network provided in the code. The Wi-Fi name here we provided is **“ESPap”**.

After completion of verification of code you can see the Wi-Fi network **ESPap** in the network list.



You can see the Wi-Fi is shown in the available networks. Now you connect to the network. After that open the browser and request for local area connection by the giving ip address “192.168.4.1” in the search bar. After typing the ip address the HTML page gets as seen below.



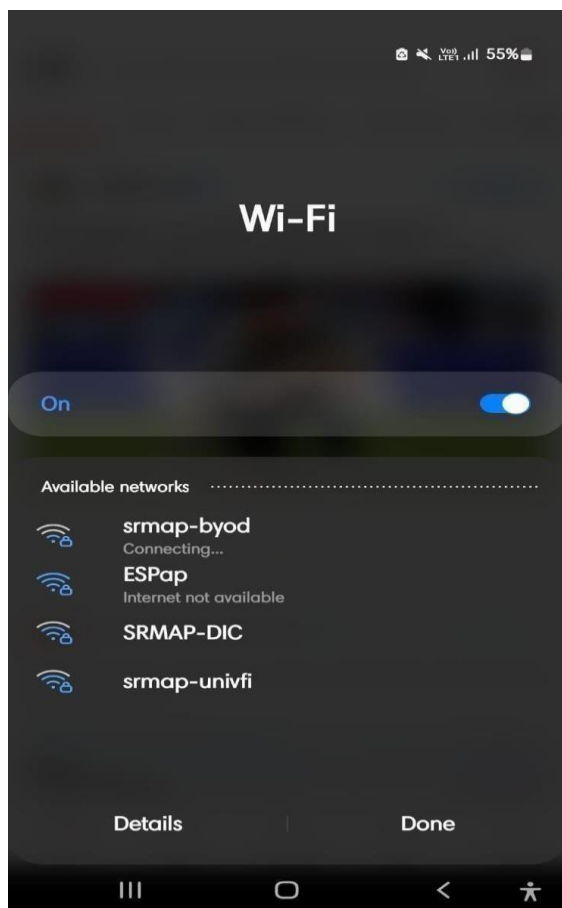
For ESP8266 tutorials visit

www.techiesms.com

[Go to Page 1](#) [Toggle LED](#)

Whenever you click on ToggleLED the light on the ESP8266 boards on and OFF. This can also be done in any mobile or pc by request the ip.address to create an local network.

You can the see below pictures seen working through by requesting the ip address.



You can see the above picture captured in the mobile. Controlling the ESP8266 board to turn ON and OFF the led.

CONCLUSIONS

The prime objective of our project is to use the Smartphone to control the home appliances effectively. The switch mode are used to control the home appliances. The video feedback is received in the android app which streams the video of IP- Camera. This project is based on the Raspberry pi, Android platform Java, NodeMCU and Python. These platforms are Free Open Source Software. So the overall implementation cost is low and can be easily configured. User can easily interact with the android phone/tablet. The user can send commands via the mobile application. The data are being analyzed by the application and are sent over a network. The Raspberry pi and NodeMCU acts as a hardware, analyses the data and activates the GPIO (General Purpose Input Output) Pins. The GPIO Pins are connected to the relays switch which activated the required home appliances. In this way, automation process is carried out

Future Scope of Work

The future of iot will assist in enhancing better control of medical parameters. With help of 5g, AI and sensors it will be easy for medical professionals to monitor patients activities and vitals. It is an evolution of a distributed control system which permits for a higher degree of automation using cloud to refine and optimize the process control.

REFERENCES

Web

- [1] About ESP8266, www.espressif.com
- [2] Raspberry Pi Zero W, www.raspberrypi.org
- [3] Camera Sensor, www.sony-semicon.co.jp
- [4] Software development cycle, www.upedu.org
- [5], NodeMCU, www.nodemcu.com
- [6], Raspberry Pi Zero W setup, www.developer.ibm.com
- [7], Implementation of OpenCV using python, docs.opencv.org
- [8], Motion detection in video using python, www.technicdynamic.com
- [9], Video quality enhancement using OpenCV, www.opencv-pythontutroals.readthedocs.io/ [10],
Video storage in H.264, www.learnopencv.com