

# **Home Automation System**

*A Project Submitted in Partial Fulfillment of the Requirements for the Degree of  
Bachelor in Computer Science & Engineering*

*by*

**Mohd. Ashraf Uz Zaman**

CSE 05006454

&

**Rafiqul Hasan Rony**

CSE 05006429

Supervised by: Zonayed Ahmed  
Lecturer



Department of Computer Science and Engineering  
STAMFORD UNIVERSITY BANGLADESH

July 2017

# Abstract

Smart homes are no longer design concepts of the future. They are being built now and they are having a direct impact on the lifestyles of people living in them. An improved home automation system that permits the user of the system to remotely control the operation of various devices. The aim of smart home systems is to create an environment that is aware of the activities taking place within it. Beside the healthy people, disabled people also need such systems to make their life easier. Because they encounter with a lot of difficulties in their everyday life especially when they are at home. In this project a smart home application are designed to control individual devices by using voice command or manual system. In order to maintain security a doorbell notification system and security camera was developed. A temperature sensor also added for observing room temperature and fire alert. Home automation system saves time, money and energy.

# Approval

The Project Report “Home Automation System” submitted by Mohd. Ashraf Uz Zaman, STUDENT ID: CSE 05006454, Rafiqul Hasan Rony, STUDENT ID: CSE 05006429, to the Department of Computer Science & Engineering, Stamford University Bangladesh, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science (Hons) in Computer Science & Engineering and approved as to its style and contents.

Board of Examiner’s Name, Signature and Date:

.....

**Adnan Ferdous Ashrafi**

Lecturer

Department of CSE

Date:

**Maliha Mahbub**

Lecturer

Department of CSE

Date:

**Samia Sultana**

Lecturer

Department of CSE

Date:

Supervisor’s Signature and Date:

.....

**Zonayed Ahmed**

Lecturer

Department of CSE

Date:

# Declaration

We, hereby, declare that the work presented in this Project is the outcome of the investigation performed by us under the supervision of Zonayed Ahmed, Lecturer, Department of Computer Science & Engineering, Stamford University Bangladesh after Dr. Mohammad Shaharia Bhuiyan, Assistant Professor of the Department of Computer Science & Engineering, Stamford University Bangladesh leaving the university. We also declare that no part of this Project and thereof has been or is being submitted elsewhere for the award of any degree or Diploma.

Signature and Date:

.....

**Mohd. Ashraf Uz Zaman**

Date:

.....

**Rafiqul Hasan Rony**

Date:

# Acknowledgments

First and foremost, we are grateful to Allah, the Almighty, the Merciful without whose blessing, this project would not have been successful. He gave us confidence, courage and determination to overcome the obstacles we faced during this journey.

At first we would like to thank our honorable supervisor Dr. Mohammad Shaharia Bhuiyan, Assistant Professor of the Department of Computer Science & Engineering, Stamford University Bangladesh, for his unconditional guidance, insights, immense patience and inspired us to take this project. But its a matter of sorrow that he left the university. Then we resumed with Zonayed Ahmed, Lecturer of the Department of Computer Science & Engineering, Stamford University Bangladesh. We would like to express our eternal gratitude to Zonayed Ahmed, Lecturer, our respected supervisor, for taking the time from his hectic schedule to guide us in this project. He is the one who inspires us to this whole project and supported us every step of the way in every possible way. Without him, this project would not have come to fruition. He provided with valuable technical advice and pointed us in the right directions which were much required for a project like this. We are truly grateful for having a teacher like him as our supervisor. We would also like to thank our parents and our best friends for pushing and encouraging us to do the best in our ability.

# Table of Contents

<b>List of Figures</b>	<b>1</b>
<b>1: Introduction</b>	<b>3</b>
1.1 Introduction . . . . .	3
1.2 Objective . . . . .	4
1.3 Scope . . . . .	4
<b>2: Literature Review</b>	<b>5</b>
2.1 The era of Home Automation System . . . . .	5
2.1.1 Definition of Home Automation System . . . . .	6
2.1.2 Advantage of Home Automation System . . . . .	6
2.1.3 Home Automation System Categories . . . . .	6
2.1.3.1 Power Line Systems . . . . .	7
2.1.3.2 Wired Systems . . . . .	7
2.1.3.3 Wireless Systems . . . . .	7
2.2 Required Hardware . . . . .	7
2.2.1 Raspberry Pi 3 Model B . . . . .	7
2.2.2 Raspberry Pi 3 Model B Required Accessories . . . . .	8
2.2.3 BCM2837 . . . . .	9
2.2.4 Power . . . . .	9
2.2.5 GPIO pins . . . . .	10

2.2.6	USB . . . . .	12
2.2.7	NoIR Camera V2 . . . . .	13
2.2.8	DIY Night Vision . . . . .	13
2.2.9	DS18B20 (Temperature Sensor) . . . . .	14
2.2.10	PIR Sensor . . . . .	14
2.2.11	MANHATTAN 703307 Case/Power Supply Fan . . . . .	15
	2.2.11.1 Features . . . . .	16
2.2.12	Piezo Buzzer . . . . .	16
	2.2.12.1 Applications . . . . .	17
2.3	Raspberry Pi 3 Model B Operating System . . . . .	18
	2.3.1 Raspbian . . . . .	18
	2.3.2 Android Things . . . . .	18
2.4	Network Server & Software . . . . .	18
	2.4.1 Access over Internet . . . . .	18
	2.4.2 VNC . . . . .	18
	2.4.3 SSH (Secure Shell) . . . . .	19
	2.4.4 Set Up YouTube Channel . . . . .	19
	2.4.5 Prepare the Raspberry Pi for Live YouTube Streaming . . . . .	20
2.5	Programming & Scripting Languages . . . . .	21
	2.5.1 XML . . . . .	21
	2.5.2 Material Design . . . . .	21
	2.5.3 Java . . . . .	22
2.6	Software Tools . . . . .	22
	2.6.1 Android Studio IDE . . . . .	22
	2.6.2 Android SDK . . . . .	23
	2.6.3 Gradle . . . . .	23
	2.6.4 FFmpeg . . . . .	24
2.7	Database . . . . .	24
	2.7.1 Firebase Realtime Database . . . . .	24
	2.7.1.1 How does it work? . . . . .	24

<b>3: System Design</b>	<b>26</b>
3.1 Hardware Design & Development . . . . .	26
3.1.1 Raspbian-Jessie Installing On Raspberry pi 3 . . . . .	26
3.1.2 Android Things Installing On Raspberry pi 3 . . . . .	27
3.1.3 Device Interfacing . . . . .	28
3.2 Software Design & Development . . . . .	29
3.2.1 Program Flow . . . . .	29
3.3 Chapter Summary . . . . .	30
<b>4: Implementation</b>	<b>31</b>
4.1 Frame . . . . .	31
4.2 Android Application . . . . .	31
4.3 Chapter Summary . . . . .	38
<b>5: Conclusion</b>	<b>39</b>
5.1 Limitations . . . . .	39
5.2 Future Works . . . . .	40
5.2.1 Computer Vision . . . . .	40
5.2.1.1 Powerful image Analysis . . . . .	40
5.2.2 Face recognition . . . . .	40
5.2.3 Biometric Door Lock . . . . .	40
<b>References</b>	<b>41</b>
<b>Appendix A: Appendix</b>	<b>44</b>
A.1 Logic & Code . . . . .	44
A.1.1 Light control . . . . .	44
A.1.2 Fan control . . . . .	46
A.1.3 Doorbell control . . . . .	48
A.1.4 Mobile Application Design Code . . . . .	50
A.1.5 Raspberry pi Camera Streaming Setup Code . . . . .	53



# List of Figures

2.1	Raspberry Pi 3 model B [1]. . . . .	8
2.2	Raspberry Pi 3 pin diagram [2]. . . . .	11
2.3	NoIR Camera V2 [3]. . . . .	13
2.4	DIY Night Vision [4]. . . . .	14
2.5	Temperature Sensor [5]. . . . .	14
2.6	PIR Sensor [6]. . . . .	15
2.7	MANHATTAN 703307 Case/Power Supply Fan [7]. . . . .	16
2.8	piezobuzzer [8]. . . . .	17
2.9	Live Streaming . . . . .	19
2.10	Live Streaming Key . . . . .	20
2.11	Raspberry pi Camera Enable . . . . .	20
3.1	Raspbian-Jessie installation . . . . .	26
3.2	Devices Interfacing . . . . .	28
3.3	Hardware Flow-chat . . . . .	29
3.4	Program Flow-chat . . . . .	30
4.1	House Design . . . . .	31
4.2	App Splash Screen . . . . .	32
4.3	App Sign up . . . . .	32
4.4	App Login . . . . .	33

4.5	App Home Screen . . . . .	33
4.6	System Check Message . . . . .	34
4.7	App Light Control Screen . . . . .	34
4.8	App Fan Control Screen . . . . .	35
4.9	Voice Command Screen . . . . .	35
4.10	App Big Picture Style Notification . . . . .	36
4.11	App Notification image . . . . .	36
4.12	App Climate Screen . . . . .	37
4.13	Security Camera Live Streaming . . . . .	37

# 1 Introduction

## *1.1 Introduction*

It is a fact that human beings have been creating everything to live more conveniently since the beginning of the world. Automating works and objects is one of the major topics which is helping to achieve this fact. Thanks to the technology, these days we are going toward automating everything to achieve comfort as much as we can. Since people spend a big portion of their life at home, automating a house would be an attractive subject for ease of life. With a home automation system, people are able to program many objects to work automatically, which is automation of the home, housework or household activities. In addition, there are new available features that let users control objects remotely. Nowadays most people have their smartphone nearby them; therefore, adding an interface on the smartphone to control an automated system is a big plus point.

Home automation systems could control lighting , fan and heating system, appliance and security door bell to bring more accessibility, convenience, safety, ease and energy efficiency. An automated home is an integrated system to gather security systems and appliances. This house system is connectable to a smart grid system over a network, which causes energy saving as well as decreasing the cost.

The new idea of Internet of Things (IoT) is growing fast and people are doing more and more work over the Internet. All controllable objects in a house could be connected into a local area network (LAN) and the main board of the system also could send and receive data over the internet for a remote user. Both of the local connection and the mainboard to the internet connection could be wired or wireless with different technologies such as WLAN, ZigBee (XBee), Bluetooth, etc.

Security is an important subject that could be added to a home automation system. Gathering all different security systems such as alarm, the fire alarm, access control (door bell notification), security camera etc. makes our life safer. All the security systems mentioned above could be integrated into a home automation system.

Energy saving is another benefit of the smart home system. It enables us to save energy and cost with having a smart control of the lighting system, water sprinkler system, solar system, etc.

## **1.2 Objective**

The objective of this project is to use Raspberry pi 3 model B to design and build a smart home automation system which provides the user with new features such as doorbell notification, smart electronic devices control system, fire alarm system and Security camera system plus remote access to control home appliance and objects wirelessly over Internet via any smartphone. The purpose of the project is to bring comfort, security and energy saving to our lives.

## **1.3 Scope**

This project can be play a vital role for make life easier. This project has been implemented with a low cost whereas anyone can use it, specially disabled person for make their every days life easier. This system can be used as controlling over the electro device like on/off light, fan or other electronics accessories via a mobile application.

In future, Face recognition, Biometric Door Lock will added for better security. When intruder tries to break the door, the vibration is sensed by sensor which makes an alarm. This will inform the neighbors or security about intruders and this will help to take further action to prevent intruder from entering. Or, if the face recognition system is unable to recognize the stranger face, it will notify the user.

However in this project Computer Vision also can be added for monitoring the entrance.

## 2 Literature Review

In this chapter we will discuss about all the hardware, software tools and languages which is required for our project.

### ***2.1 The era of Home Automation System***

Early home automation began with labor-saving machines. Self-contained electric or gas powered home appliances became viable in the 1900s with the introduction of electric power distribution and led to the introduction of washing machines (1904), water heaters (1889), refrigerators, sewing machines, dishwashers, and clothes dryers.

In 1975, the first general purpose home automation network technology, X10, was developed. It is a communication protocol for electronic devices. It primarily uses electric power transmission wiring for signaling and control, where the signals involve brief radio frequency bursts of digital data, and remains the most widely available. By 1978, X10 products included a 16 channel command console, a lamp module, and an appliance module. Soon after came the wall switch module and the first X10 timer.

By 2012, in the United States, according to ABI Research, 1.5 million home automation systems were installed.

According to Li et al. (2016) there are three generations of home automation [9]:

- First generation: wireless technology with proxy server, e.g. Zigbee automation.
- Second generation: artificial intelligence controls electrical devices, e.g. Amazon Echo.
- Third generation: robot buddy who interacts with humans, e.g. Robot Rovio, Roomba.

### *2.1.1 Definition of Home Automation System*

Home automation or smart home is building automation for the home. It involves the control and automation of lighting, heating (such as smart thermostats), ventilation, air conditioning (HVAC), and security, as well as home appliances such as washer/dryers, ovens or refrigerators/freezers. Wi-Fi is often used for remote monitoring and control. Home devices, when remotely monitored and controlled via the Internet, are an important constituent of the Internet of Things. Modern systems generally consist of switches and sensors connected to a central hub sometimes called a "gateway" from which the system is controlled with a user interface that is interacted either with a wall-mounted terminal, mobile phone software, tablet computer or a web interface, often but not always via Internet cloud services [10].

### *2.1.2 Advantage of Home Automation System*

Home automation has greatly increased in popularity over the past several years. One of the greatest advantages of an automated home is the ease with which functionality can be managed on an array of devices: desktop, laptop, tablet or smart-phone. Before determining which home automation package is right for you and your family, it is important to become better informed of the features and settings associated with home safety and security systems.

One of the greatest advantages of home automation systems is that users can protect against break-ins and fires, while enjoying automations for lights, temperature, and more. The automation of features in ones home helps to promote security, comfort, energy efficiency, and convenience. Another benefit of home automation systems is the amount of labor, time, energy and materials that is saved.

Home automation systems are becoming more and more affordable. Not only are prices decreasing, but operating systems are also become less complex so that users can readily master all the controls associated with their safety and security devices. Home automation commands can now be given through smartphones, tablets, and televisions, in addition to computers [11].

### *2.1.3 Home Automation System Categories*

There are three main types of home automation systems that can be installed in the average home as follow [12].

### *2.1.3.1 Power Line Systems*

This is the most affordable of all the home automation products. Power line systems rely on the existing power lines in your home to transfer things such as security camera feeds and lighting information to a common control interface. Often, these are X10 technology based systems.

### *2.1.3.2 Wired Systems*

Wired systems use Cat 5 cables to communicate information. They may connect into a proprietary control system or a more open control center. Wired systems can be implemented in both new and existing homes, however it is easier to install in new homes. Hardwiring a home automation system is a good choice as the wiring is reliable and it is often simpler to join all of the systems together as the cables run to a common junction point.

### *2.1.3.3 Wireless Systems*

These are great for existing homes as there is no need to run cabling behind the walls, which can be a very expensive endeavor. Wireless systems are also starting to integrate with WiFi networks, meaning that they are easily compatible with any existing home networks, such as the ones formed by computers. However, some wireless systems use a different radio frequency, making them incompatible with open networks. This is something that will need to be checked before installation.

## **2.2 Required Hardware**

### *2.2.1 Raspberry Pi 3 Model B*

The Raspberry Pi is a very small and low-cost computer which has the size of a credit card and can be plugged into any computer monitor or TV. It uses a standard keyboard and mouse. It has all the capability of a desktop like browsing the Internet, playing high definition video, making spreadsheets, word processing and playing games [13]. What is unique to Raspberry Pi is that it allows the user to interact with the outside world and is currently used in many digital projects. Users can learn to write programs by using languages like Scratch and Python [14].

The Multipurpose Surveillance Robot has utilized the Raspberry Pi 3 Model B which is

the second generation Raspberry Pi which replaced the original Raspberry Pi 3 Model B+. It has:

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5 mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot
- Video Core IV 3D graphics core



**Figure 2.1: Raspberry Pi 3 model B [1].**

### 2.2.2 *Raspberry Pi 3 Model B Required Accessories*

**Micro SD Card:** An 8 GB class 4 micro SD card with Raspbian jessie pre-installed and another 8 GB class 4 micro SD card with Android Things pre-installed, is recommended. The minimum recommended capacity of an SD card is 8 GB. 4 GB is recommended for image installation. Even smaller cards can be used for some distributions like OpenElec



and Arch. The SD card class determines the write speed a card can sustain. A class 4 SD card can achieve 4 MB/s write speed whereas for a class 10 card 10 MB/s is attainable. But this does not mean a class 10 card will perform better than a class 4 one. Because, in many cases, the higher write speed is achieved at the cost of read speed and increased seek time. The Raspberry Pi 3 Model B requires a micro-SD card.

**Display and connectivity cable:** Any HDMI/DVI monitor and any TV works as display for pi. But one with an HDMI input is recommended.

**Keyboard and mouse:** Raspberry pi works with any standard keyboard and mouse. Wireless keyboard and mouse will work if already paired with Pi. Keyboard layout can be configured through `raspberry -config`.

**Power supply:** The Pi is powered by a USB micro power supply like most standard mobile phone chargers. A good-quality power supply is required that can supply at least 700mA at 5V. Power supplies with less than 700 mA current should work for basic usage but the Pi might reboot if too much power is drawn .

**Ethernet (network) cable:** An Ethernet cable is necessary to connect Pi to a local network and the internet.

**USB wireless dongle:** Pi can also be connected to a wireless network through use of a USB wireless dongle which will need to be configured.

**Audio lead:** A standard 3.5mm audio jack is used to audio through speakers and headphones. An audio is required in the absence of an HDMI cable. If the Pi is connected to the monitor through an HDMI cable no separate audio lead is necessary as audio can be played directly from the display. But if playing audio through speakers is preferable, it will have to be configured.

### 2.2.3 *BCM2837*

The Raspberry Pi 3 Model B uses the Broadcom processor BCM2837. The underlying architecture in BCM2837 is identical to BCM2836 . It has an ARMv8 CPU.

### 2.2.4 *Power*

The device is powered by a 5V micro USB supply. Exactly how much current (mA) the Raspberry Pi requires is dependent on what is connected to it. A 1.2A (1200 mA) power supply from a reputable retailer will provide ample power to run a Pi. Typically, the model B uses between 700-1000mA depending on what peripherals are connected; the

model A can use as little as 500mA with no peripherals attached. The maximum power the Raspberry Pi can use is 1 Amp. If someone needs to connect a USB device that will take the power requirements above 1 Amp, then they must connect it to an externally-powered USB hub. The power requirements of the Raspberry Pi increase as it makes use of the various interfaces on the Raspberry Pi. The GPIO pins can draw 50mA safely ; distributed across all the pins; an individual GPIO pin can only safely draw 16mA. The HDMI port uses 50mA, the camera module requires 250 mA, and the keyboards and mice can take as little as 100mA or over 1000mA . Back powering occurs when USB hubs do not provide a diode to stop the hub from powering against the host computer. Some hubs will back feed the Raspberry Pi. This means that the hubs will power the Raspberry Pi through its USB input cable, without the need for a separate micro-USB power cable, and bypass the voltage protection. If the Raspberry Pi is connected to a hub that back feeds to it and the hub experiences power surge, the Raspberry Pi could potentially be damaged [15].

#### 2.2.5 *GPIO pins*

One powerful feature of the Raspberry Pi is the row of GPIO (General Purpose Input/Output) pins. These pins are a physical interface between the Pi and the outside world. The GPIO pins can be programmed to interact in amazing ways with the real world. Inputs don't have to come from a physical switch; it could be input from a sensor or a signal from another computer or device, for example. The output can also do anything. In addition to the familiar USB, Ethernet and HDMI ports, the Raspberry Pi offers the ability to connect directly to a variety of electronic devices. These include:

**Digital outputs:** turn lights, motors, or other devices on or off.

**Digital inputs:** read an on or off state from a button, switch, or other sensor.

**Communication with chips or modules using low-level protocols:** SPI, IC, or serial UART. Connections are made using GPIO ("General Purpose Input/Output") pins. Unlike USB, etc., these interfaces are not "plug and play" and require care to avoid miswiring. The Raspberry PI GPIOs use 3.3V logic levels, and can be damaged if connected directly to 5V levels (as found in many older digital systems) without level-conversion circuitry. Note that no analogue input or output is available. However, add-on boards such as the Rpi Gert board provide this capability. The Raspberry Pi Model A+ and B+ boards, and the Pi 3 Model B, have a 40-pin header marked J8, arranged as 2x20 pins. The first 26 pins are the same as P1 on the A/B boards, with the remaining 14 pins providing additional GPIO and ground pins, and an EEPROM ID feature for auto-configuration with add-on "HAT"

boards. GPIO voltage levels are 3.3 V and are not 5 V tolerant. There is no over-voltage protection on the board - the intention is that people interested in serious interfacing will use an external board with buffers, level conversion and analog I/O rather than soldering directly onto the main board.

All the GPIO pins can be reconfigured to provide alternate functions, SPI, PWM, IC and so. At reset only pins GPIO 14 15 are assigned to the alternate function UART, these two can be switched back to GPIO to provide a total of 17 GPIO pins. Each of their functions and full details of how to access are detailed in the chipset datasheet. Each GPIO can interrupt, high/low/rise/fall/change. There is currently no support for GPIO interrupts in the official kernel, however a patch exists, and requiring compilation of modified source tree. The 'Raspbian "wheezy"' version that is currently recommended for starters already includes GPIO interrupts. GPIO input hysteresis (Schmitt trigger) can be on or off, output slew rate can be fast or limited, and source and sink current is configurable from 2 mA up to 16 mA. Note that chipset GPIO pins 0-27 are in the same block and these properties are set per block, not per pin. See GPIO Datasheet Addendum - GPIO Pads Control. Particular attention should be applied to the note regarding SSO (Simultaneous Switching Outputs): to avoid interference, driving currents should be kept as low as possible. The available alternative functions and their corresponding pins are detailed below. These numbers are in reference to the chipset documentation and may not match the numbers exposed in Linux. Only fully usable functions are detailed, for some alternative functions not all the necessary pins are available for the functionality to be actually used [16]. Raspberry Pi 3 pin diagram is given below:

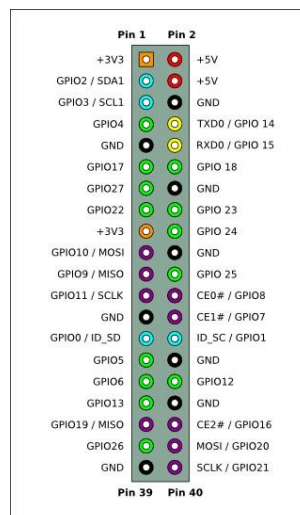


Figure 2.2: Raspberry Pi 3 pin diagram [2].

### 2.2.6 USB

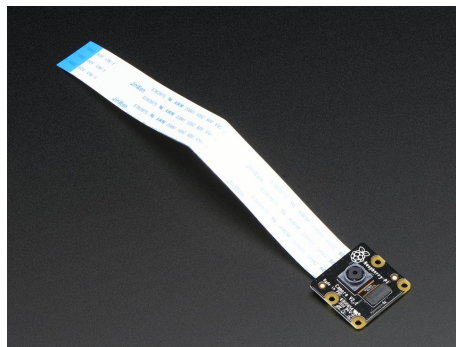
The Raspberry Pi 3 Model B is equipped with four USB 2.0 ports. These are connected to LAN9512 combo hub/Ethernet chip IC3, which is itself a USB device connected to the single upstream USB port on BCM2837. The USB ports enable the attachment of peripherals such as keyboard, mice, webcams that provide the Pi with additional functionality. There are some differences between the USB hardware on the Raspberry Pi and the USB hardware on desktop computers or laptop/tablet devices. The USB host port inside the Pi is an On-The-Go (OTG) host as the application processor powering the Pi, BCM2836, was originally intended to be used in the mobile market: i.e. as the single USB port on a phone for connection to a PC, or to a single device. In essence, the OTG hardware is simpler than the equivalent hardware on PC. OTG in general supports communication to all types of USB device, but to provide an adequate level of functionality for the most of the USB devices that one might plug into a Pi, the system software has to do more work. In general, every device supported by Linux is possible to use with the Pi, subject to a few caveats detailed further down. Linux has probably the most comprehensive driver database for legacy hardware of any operating system. The OTG hardware on Raspberry Pi has a similar level of support for certain devices, which may present a higher software processing overhead. The Raspberry Pi also has only one root USB port: all traffic from all connected devices is funneled down this bus, which operates at a maximum speed of 480 mbps. The OTG hardware on Raspberry Pi has a simpler level of support for certain devices, which may present a higher software processing overhead. The Raspberry Pi also has only one root USB port: all traffic from all connected devices is funneled down this bus, which operates at a maximum speed of 480mbps. The USB specification defines three device speeds - Low, Full and High. Most mice and keyboards are Low-speed, most USB sound devices are Full-speed and most video devices (webcams or video capture) are High-speed. Generally, there are no issues with connecting multiple High-speed USB devices to a Pi. The software overhead incurred when talking to Low- and Full-speed devices means that there are soft limitations on the number of simultaneously active Low- and Full-speed devices. Small numbers of these types of devices connected to a Pi will cause no issues. USB devices have defined power requirements, in units of 100mA from 100mA to 500mA. The device advertises its own power requirements to the USB host when it is first connected. In theory, the actual power consumed by the device should not exceed its stated requirement. The USB ports on a Raspberry Pi have a design loading of 100mA each - sufficient to drive "low-power" devices such as mice and keyboards. Devices such as WiFi adapters, USB hard drives, USB pen drives all consume much more current and should be powered from an external hub with its own power supply. While it

is possible to plug a 500mA device into a Pi and have it work with a sufficiently powerful supply, reliable operation is not guaranteed. In addition, hot plugging high-power devices into the Pi's USB ports may cause a brownout which can cause the Pi to reset.

### 2.2.7 *NoIR Camera V2*

The Raspberry Pi NoIR Camera Board v2 is a high quality 8 megapixel Sony IMX219 image sensor custom designed add-on board for Raspberry Pi, featuring a fixed focus lens. It's capable of 3280 x 2464 pixel static images, and also support 1080p30, 720p60, and 640x480p60/90 video.

It attaches to the Pi by way of one of the small sockets on the board's upper surface and uses the dedicated CSI interface, designed especially for interfacing to cameras. The NoIR Camera has No InfraRed (NoIR) filter on the lens which makes it perfect for doing Infrared photography and taking pictures in low light (twilight) environments [3].



**Figure 2.3: NoIR Camera V2 [3].**

### 2.2.8 *DIY Night Vision*

Raspberry Pi Night Vision Camera plugs directly into the CSI connector on the Raspberry Pi, and features two high intensity Infrared LED spotlights for night time recording! The IR LED's are powered directly from the CSI port, and are capable of lighting an area at a distance of up to 8m! In testing, the best images were captured at a distance of 3m to 5m. The camera also features an adjustable 3.6mm focal length lens and 75.7 degree viewing angle.

This Raspberry Pi night vision camera uses the same OV5647 as the standard Raspberry Pi camera, and is therefore able to deliver a crystal clear 5MP resolution image, or 1080p HD video recording at 30fps [4].



**Figure 2.4: DIY Night Vision [4].**

### 2.2.9 DS18B20 (Temperature Sensor)

The DS18B20 "1-wire" sensors can be connected in parallel - unlike nearly any other sensor sold! All sensors should share the same pins, but only need one 4.7K resistor for all of them.

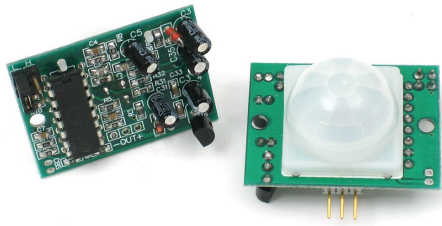
The resistor is used as a 'pullup' for the data-line, and is required to keep the data transfer stable and happy [5].



**Figure 2.5: Temperature Sensor [5].**

### 2.2.10 PIR Sensor

PIR sensors allow to sense motion, almost always used to detect whether a human has moved in or out of the sensors range. They are small, inexpensive, low-power, easy to use and don't wear out. For that reason they are commonly found in appliances and gadgets used in homes or businesses. They are often referred to as PIR, "Passive Infrared", "Pyroelectric", or "IR motion" sensors.



**Figure 2.6: PIR Sensor [6].**

PIRs are basically made of a pyroelectric sensor, which can detect levels of infrared radiation. Everything emits some low level radiation, and the hotter something is, the more radiation is emitted. The sensor in a motion detector is actually split in two halves. The reason for that is that we are looking to detect motion (change) not average IR levels. The two halves are wired up so that they cancel each other out. If one half sees more or less IR radiation than the other, the output will swing high or low [6].

#### *2.2.11 MANHATTAN 703307 Case/Power Supply Fan*

MANHATTAN Case/Power Supply Fans are equipped with high-quality sleeve or ball bearings to help deliver efficient, reliable cooling and quiet performance. Easily installed and offered in range of sizes, MANHATTAN Case/Power Supply Fans help circulate cool air through the case to dissipate heat away from processors and components to reduce heat damage [7].



**Figure 2.7: MANHATTAN 703307 Case/Power Supply Fan [7].**

#### *2.2.11.1 Features*

- Three-pin power connector for easy installation
- Ideal for new systems, upgrades and replacements
- Ball bearing delivers quiet, reliable operation
- Rated voltage: 12 V DC
- Input current: 0.25 A (maximum)
- Air flow: 66 cfm (maximum)
- Fan speed: 1,800 rpm
- Noise level: 30 dBA (maximum)
- Dimensions: 120 x 120 x 25 mm

#### *2.2.12 Piezo Buzzer*

Piezo buzzers (see figure 2.8 ) are used for making beeps, tones and alerts [8].





**Figure 2.8: piezobuzzer [8].**

This one is petite but loud! Drive it with 3-30V peak-to-peak square wave. To use, connect one pin to ground (either one) and the other pin to a square wave out from a timer or micro controller. For the loudest tones, stay around 4 KHz, but works quite well from 2KHz to 10KHz. For extra loudness, you can connect both pins to a micro controller and swap which pin is high or low (differential drive) for double the volume.

#### *2.2.12.1 Applications*

- Judging panels
- Educational purposes
- Annunciator panels
- Electronic metronomes
- Game show lock-out device
- Microwave ovens and other household appliances
- Sporting events such as basketball games
- Electrical alarms
- Klaxon

## **2.3 Raspberry Pi 3 Model B Operating System**

There are different ways to install operating system into raspberry pi. The main two systems are discussed below :

### *2.3.1 Raspbian*

Raspbian jessie is the default operating system for regular use on Raspberry Pi. Raspbian jessie is a free operating system based on Debian and optimized for Raspberry Pi hardware. Raspbian jessie comes with over 35000 packages; precompiled software bundled in a nice format for easy installation on Raspberry Pi. Raspbian jessie is a community project under active development, with the emphasis on developing stability and performance of as many Debian packages as possible [17].

### *2.3.2 Android Things*

Android Things (codenamed Brillo) is an Android-based embedded operating system platform by Google, announced at Google I/O 2015. It is aimed to be used with low-power and memory constrained Internet of Things (IoT) devices, which are usually built from different MCU platforms [18].

## **2.4 Network Server & Software**

### *2.4.1 Access over Internet*

Raspberry Pi can be connected to another computer or a mobile device. One method is to set-up port forwarding. To set-up port forwarding one must change the configuration of their router to forward all inbound traffic from the Internet to a specific port to the local IP address of their Raspberry Pi. One disadvantage of doing this is that it exposes a network port on a private LAN to the public network. This is security vulnerability and must be managed carefully. One secure alternative to port forwarding is the Weaved service. Weaved is software that needs to be install on the Raspberry Pi and it will allow Pi to connect to any device over the Internet.

### *2.4.2 VNC*

VNC is a graphical desktop sharing system that allows someone to remotely control the desktop interface of one computer from another. It transmits the keyboard and mouse

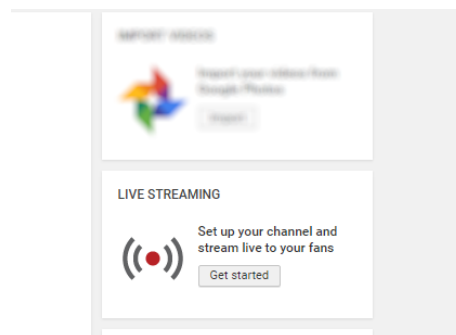
events from the controller, and receive updates to the screen over the network from the remote host. This enables a user to use the desktop of the Pi inside a window on their computer. They will be able to control it as though they were working on Raspberry Pi itself. VNC is used widely across every industry sector by individuals and organizations for different use cases which include providing IT desktop support to colleagues and friends, and also accessing system and services on the move [19].

#### 2.4.3 SSH (Secure Shell)

Secure Shell, or SSH is a network protocol which operates at Application layer of OSI model to allow encrypted remote login and other services for secure operations over an unsecured network. SSH provides a secure channel in a unsecure network through a client-server architecture. SSH is capable of securing any network service but typical applications include remote command-line login and remote command execution. The command line of Raspberry Pi can be accessed from another computer in the same network using SSH. SSH server is enabled on Raspberry Pi by default. It can be disabled as well. SSH is built into Linux distributions and Mac OS, and a third-party SSH client is available for Windows.

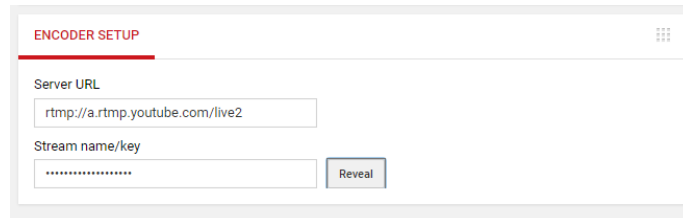
#### 2.4.4 Set Up YouTube Channel

Here we need a youtube account with personal Channel and We need a special URL from here that we can use to direct the footage captured by the Raspberry Pis camera to YouTube, thus streaming it. This is called an RTMP address and is basically a specific media URL [20].



**Figure 2.9: Live Streaming**

To find this, head to YouTube, sign in, and look for the Upload button. This is what you would normally use in YouTube to add a video. On this occasion, however, were going to ignore this and click Get started button under Live Streaming.



**Figure 2.10: Live Streaming Key**

the subsequent screen, we need to fill in the details for the live feed. This will be information about the subject of the feed, and a title, which we should add under Basic Info. In the next tab, Stream Options, look for Encoder Setup and copy the Server URL and Stream name/key. Note that the Stream key needs to be kept private anyone with this information can stream to the YouTube channel!

#### 2.4.5 Prepare the Raspberry Pi for Live YouTube Streaming

Now, its time to set up your Raspberry Pi for streaming.

Begin by running an upgrade. This ensures youre running the most recent version of Raspbian, with all of the necessary system and software updates, including raspivid [20].

```
Raspberry Pi Software Configuration Tool (raspi-config)
1 Expand Filesystem           Ensures that all of the SD card
2 Change User Password        Change password for the default
3 Enable Boot to Desktop/Scratch Choose whether to boot into a
4 Internationalisation Options Set up language and regional s
5 Enable Camera                Enable this Pi to work with th
6 Add to Rastrack             Add this Pi to the online Rasp
7 Overclock                   Configure overclocking for you
8 Advanced Options            Configure advanced settings
9 About raspi-config          Information about this configu

<Select>                                <Finish>
```

**Figure 2.11: Raspberry pi Camera Enable**

Next, we connect camera and boot up. If you dont have a monitor attached, use VNC to establish a remote desktop connection to the Pi and test the camera.

## **2.5 Programming & Scripting Languages**

### *2.5.1 XML*

In addition to Java code, Android projects (and their developers) have the ability to utilize XML to perform many standard tasks. Some XML usage is required, such as the definition of the projects and its components. Much of XML's usage is optional, making many common tasks easier. The Android XML schema is highly flexible and may be used in combination with code, exclusively, or not at all. Below is a list of common usage of XML [21]:

- Manifest - definition of the project
- Layout - creation of partial or complete layouts for Activities, Dialogs, and Widgets
- Colors-Constant color values used throughout the project
- Style & Themes - application of custom standardized looks of Views
- Animations - Standardized animations that may be applied to Views
- Drawables - Some specialized icons and graphics that may not be created wholly with an image editor. (StateDrawables, TranstionDrawables, Shapes, and Vector-Graphics)
- Menu - A resource used to aid in standardized menus for an Activity Integers, Strings, and Arrays - Constants used by the Application as resources.

### *2.5.2 Material Design*

Material design is a comprehensive guide for visual, motion, and interaction design across platforms and devices. Material Design makes more liberal use of grid-based layouts, responsive animations and transitions, padding, and depth effects such as lighting and shadows. Google announced Material Design on June 25, 2014, at the 2014 Google I/O conference. Android 5.0 Lollipop includes support for material design application. Polymer and Angular Material projects also provide official implementations [22].

### 2.5.3 *Java*

Java is a general-purpose computer programming language that is concurrent, class based, object oriented and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "Write Once, Run Anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to byte code that can run on any Java Virtual Machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (byte code compiler), GNU Classpath (standard libraries), and Iced Tea-Web (browser plug in for applets). The latest version is Java 8, which is the only version currently supported for free by Oracle, although earlier versions are supported both by Oracle and other companies on a commercial basis [23].

## **2.6 Software Tools**

### 2.6.1 *Android Studio IDE*

Android Studio was on May 16, 2013. Google's product manager Ellie Powers on the developer conference Google I/O announced. Shortly after this time, Google periodically new test versions. After a development time of two years, Google published on 8 December 2014 Version 1.0 for Windows, OS X and Linux. Since the alpha version 1.2 Preview 1, which was published on 10 March 2015 is based on Android Studio IntelliJ IDEA 14. With the preview version 1.3 of 28 May 2015 the SDK Manager has been fully integrated into Android Studio, further is now support for the Android NDK ( Native Development Kit ) available. In the final version 1.3 also the Android Memory Viewer and Allocation Tracker was integrated. From this version, it is also possible to in line annotations for the

new application permissions of Android M Data Binding to use as well feature is Instant Run available, which allows developers to modify the amended code and Since the Android Studio preview version 2.0, the resources directly on the device within the current app [24].

### 2.6.2 *Android SDK*

The Android Software Development Kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.5.8 or later, and Windows 7 (previously XP) or later. As of March 2015, the SDK is not available on Android itself, but the software development is possible by using specialized Android applications. Until around the end of 2014, the officially supported Integrated Development Environment (IDE) was Eclipse using the Android Development Tools (ADT) Plug-in, though IntelliJ IDEA IDE (all editions) fully supports Android out of the box and NetBeans IDE also supports Android development via a plug-in. As of 2015, Android Studio, made by Google and powered by IntelliJ, is the official IDE; however, developers are free to use others. Additionally, developers may use any text editor to edit Java and XML files, then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g. triggering a reboot, installing software package(s) remotely). Enhancements to Android's SDK go hand in hand with the overall Android platform development. The SDK also supports older versions of the Android platform in case developers wish to target their applications at older devices. Development tools are download able components, so after one has downloaded the latest version and platform, older platforms and tools can also be downloaded for compatibility testing. Android applications are packaged in .apk format and stored under folder on the Android OS (the folder is accessible only to the root user for security reasons). APK package contains dex files (compiled byte code files called Dalvik executables), resource files etc [25].

### 2.6.3 *Gradle*

Gradle is an open source build automation system that builds upon the concepts of Apache Ant and Apache Maven and introduces a Groovy-based Domain-Specific Language (DSL) instead of the XML form used by Apache Maven of declaring the project configuration. Gradle uses a Directed Acyclic Graph ("DAG") to determine the order in which tasks can

be run. Gradle was designed for multi-project builds which can grow to be quite large and supports incremental builds by intelligently determining which parts of the build tree are up-to-date, so that any task dependent upon those parts will not need to be re-executed. The initial plug-in are primarily focused around Java, Groovy and Scala development and deployment, but more languages and project work flows are on the road map [26].

#### 2.6.4 *FFmpeg*

FFmpeg is a free software project that produces libraries and programs for handling multimedia data. FFmpeg includes libavcodec, an audio/video codec library used by several other projects, libavformat (Lavf), an audio/video container mux and demux library, and the ffmpeg command line program for transcoding multimedia files. FFmpeg is published under the GNU Lesser General Public License 2.1+ or GNU General Public License 2+ (depending on which options are enabled) [27].

## 2.7 **Database**

### 2.7.1 *Firestore Realtime Database*

The Firestore Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data [28].

#### 2.7.1.1 *How does it work?*

The Firestore Realtime Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, realtime events continue to fire, giving the end user a responsive experience. When the device regains connection, the Realtime Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Realtime Database provides a flexible, expression-based rules language, called Firestore Realtime Database Security Rules, to define how your data should be structured and when data can be read from or written to. When integrated with Firestore Authentication, developers can define who has access to what data, and how they can access it.

The Realtime Database is a NoSQL database and as such has different optimizations and



functionality compared to a relational database. The Realtime Database API is designed to only allow operations that can be executed quickly. This enables you to build a great realtime experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.

# 3 System Design

This System design is consist of two part. One of the part Hardware Design and another one is Software Design.

## 3.1 Hardware Design & Development

### 3.1.1 Raspbian-Jessie Installing On Raspberry pi 3

Raspbian-Jessie Software is an easy operating system install manager for the Raspberry Pi. The most convenient way to get Raspbian-Jessie is to purchase a micro SD card with Raspbian-Jessie pre-installed. It can also be downloaded from the Raspberry Pi website. After downloading the Raspbian-Jessie zip file, the contents of the file will have to be copied to a formatted SD card on a computer.

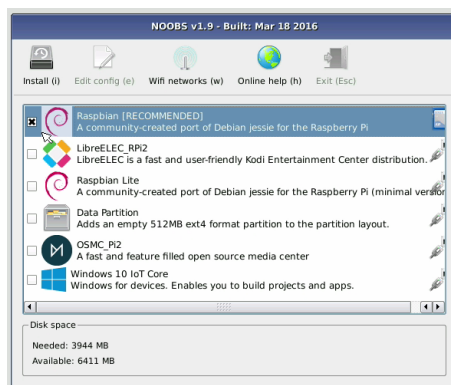


Figure 3.1: Raspbian-Jessie installation

**Installing Raspbian-Jessie on an SD card:** Format an SD that is 4 GB or larger as FAT. Download and extract files from the Raspbian-Jessie zip file. Copy the extracted file onto

the SD card immediately after formatting so that the files are in the root directory of the SD card. The files might be copied into a single folder. In that case, the files inside the folder must be copied across rather than the folder itself. The RECOVERY fat partition will be automatically resized to a minimum and a list of available to install OS will be displayed.

**Windows:** The SD Associations Formatting Tool is recommended for Windows users which can be downloaded from [sdcard.org](http://sdcard.org). It is necessary to set FORMAT SIZE ADJUSTMENT option ON in the Options menu so that the entire SD card volume is formatted instead of a single partition.

**Mac OS:** The SD Associations Formatting Tool is also available for Mac users. The default OSX Disc utility can also be used to format the disc. In order to do that, the SD card has to be selected and choose Erase with MS-DOS format.

**Linux:** For Linux users parted (or the command line version parted) is recommended. Included in Raspbian-Jessie. Only Raspbian is installed in Raspbian-Jessie by default. The others can be installed with a network connection. Raspbian-Jessie and Raspbian-Jessie Lite.

### *3.1.2 Android Things Installing On Raspberry pi 3*

The Android Things platform is an operating system from Google intended to be used on IoT devices. It is in essence a version of Android that can run on a variety of platforms (such as Raspberry Pi 3 or Intel Edison).

The way to get Android Things is to purchase a micro SD card with Android Things pre-installed. It can also be downloaded from the Android website. After downloading the Android Things zip file, the contents of the file will have to be copied to a formatted SD card on a computer.

**Windows:** Etcher is typically the easiest option for most users to write images to SD cards, so it is a good place to start.

If you're looking for an alternative on Windows, you can use Win32DiskImager:

- Insert the SD card into your SD card reader. You can use the SD card slot if you have one, or an SD adapter in a USB port. Note the drive letter assigned to the SD card. You can see the drive letter in the left hand column of Windows Explorer, for example G:
- Download the Win32DiskImager utility from the Sourceforge Project page as an installer file, and run it to install the software.

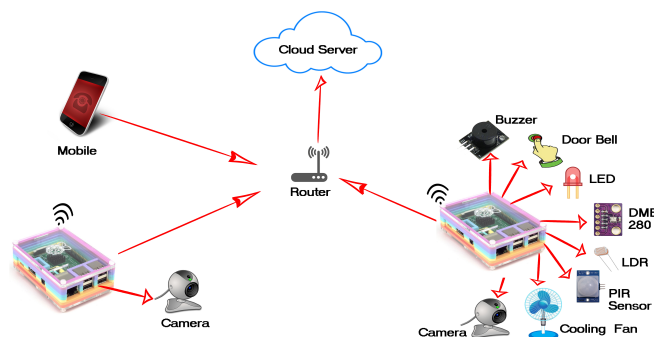
- Run the Win32DiskImager utility from your desktop or menu.
- Select the image file you extracted earlier.
- In the device box, select the drive letter of the SD card. Be careful to select the correct drive: if you choose the wrong drive you could destroy the data on your computer's hard disk! If you are using an SD card slot in your computer, and can't see the drive in the Win32DiskImager window, try using an external SD adapter.
- Click 'Write' and wait for the write to complete.
- Exit the imager and eject the SD card.

**Mac:** Etcher is typically the easiest option for most users to write images to SD cards, so it is a good place to start. If you're looking for more advanced options on Mac OS, you can use the built-in graphical and command line tools below.

**Linux:** For Linux users parted (or the command line version parted) is recommended. Included in Raspbian-Jessie. Only Raspbian is installed in Raspbian-Jessie by default. The others can be installed with a network connection. Raspbian-Jessie and Raspbian-Jessie Lite.

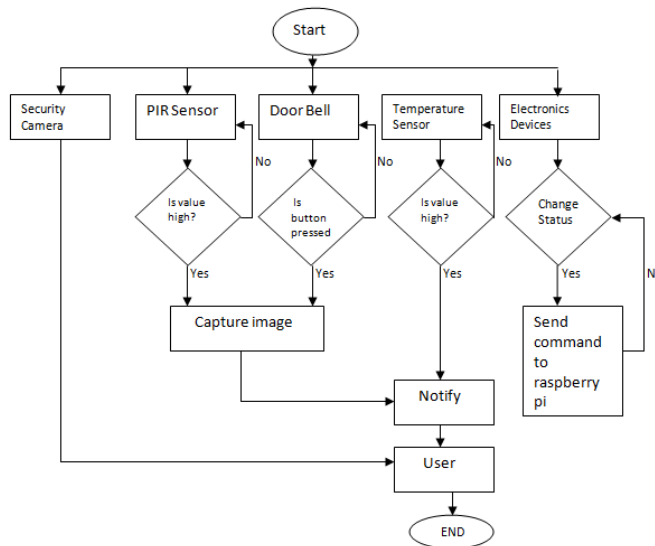
### 3.1.3 Device Interfacing

The Smart Home System consists of two Raspberry pi 3 model B, two PIR sensor, a LDR sensor, four LED, four Fan, a Temperature Sensor, a NOIR Camera, Night vision camera, a Piezo Buzzer, a Router, an Android application and Android phone.



**Figure 3.2: Devices Interfacing**

Raspberry pi connected with router via wifi to Internet and pass information to the real time fire-base database. When android application connect to the Internet the real time fire-base Database and you-tube Streaming server pass the information about the the current state of electronics accessories and security camera.

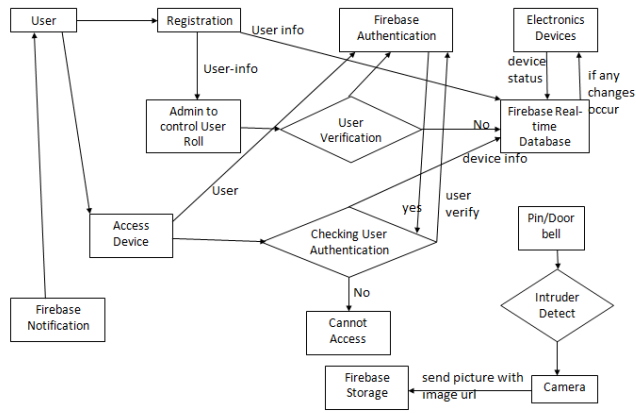


**Figure 3.3: Hardware Flow-chat**

## 3.2 Software Design & Development

### 3.2.1 Program Flow

The system architecture is as shown in Figure 3.3. First user need to registration with their name, email and password. After successful registration admin will notify by a notification. If admin approved the user, then user can login with their email and password. Then the user can control electronics devices and other usage services such as Smart light fan on/off system, Doorbell notification, Security camera and Home Temperature.



**Figure 3.4: Program Flow-chat**

### 3.3 Chapter Summary

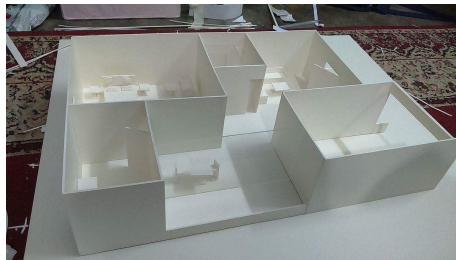
In this chapter we focused on hardware design & software design. Hardware design has been described in device interfacing section. Software design has been also described in program flow.

# 4 Implementation

In this chapter we will show our model home and how our smart home apps works.

## 4.1 *Frame*

At first we measured a space for our entire project that how much space will it to take.

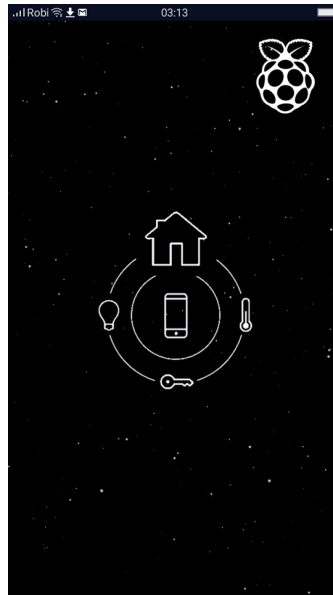


**Figure 4.1: House Design**

We shape the house and design it. Then We put sensor, fan, light, camera and raspberry pi in this demo home.

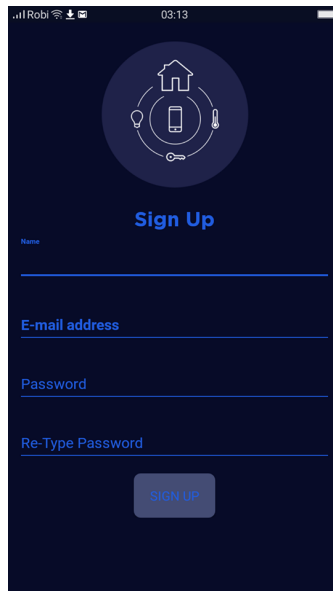
## 4.2 *Android Application*

This is the splash screen in our project app.



**Figure 4.2: App Splash Screen**

This is our project app's sign up screen for user. User need to sign up first.

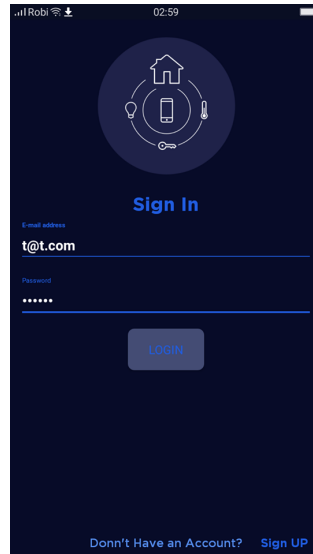


**Figure 4.3: App Sign up**

This is our app login activity for user. User will insert user email address & Password. If

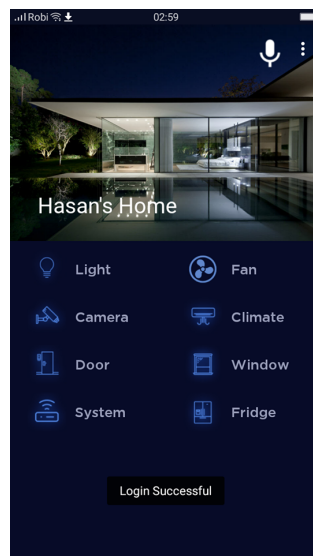


it is valid email address & Password then it will login successfully.



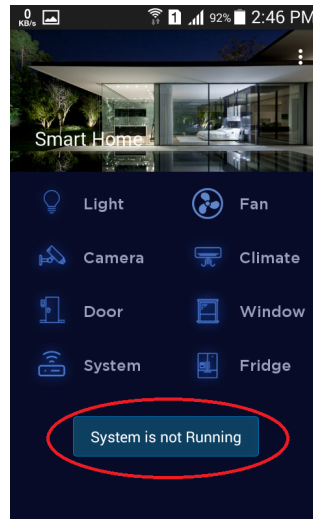
**Figure 4.4: App Login**

After login successfully another activity will be open. In this activity user can control electronic devices, get door bell notifications, observing room temperature and can see the security camera footage.



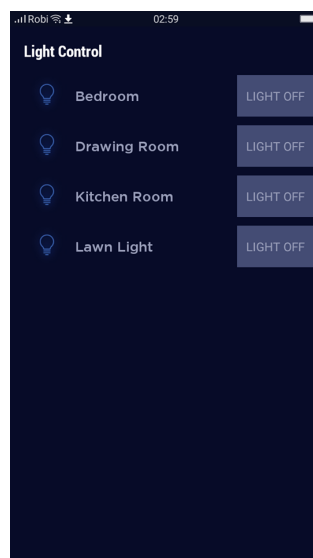
**Figure 4.5: App Home Screen**

This activity also confirmed user by a toast message whether system is running or not.



**Figure 4.6: System Check Message**

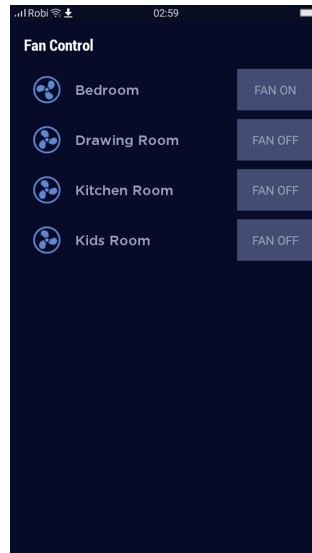
**light Control button:** It will open an activity for light on/off for user.



**Figure 4.7: App Light Control Screen**

User can change current status on any room light.

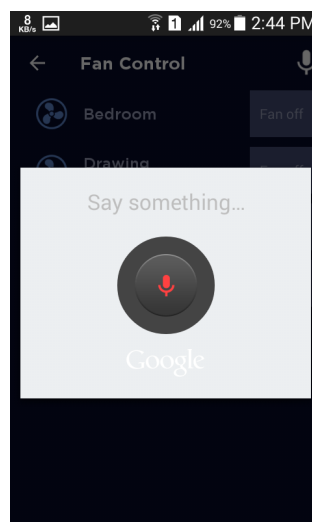
**Fan Control button:** It will open an activity for fan on/off for user.



**Figure 4.8: App Fan Control Screen**

User can change current status on any room fan.

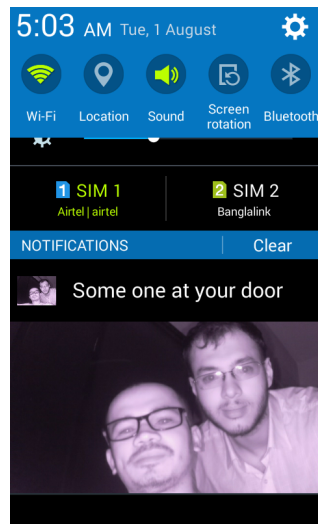
**Voice Command:** In this activity user can control various electronic device by voice command.



**Figure 4.9: Voice Command Screen**

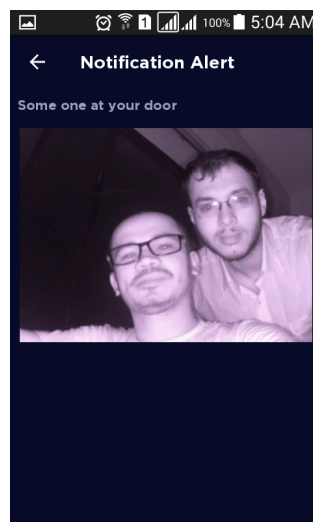
User also can turn on/off all light/fan by a single command.

**Door Bell notification:** When someone press the door bell button, a camera will take strangers image and send it to real time fire-base storage and fire-base storage sent the image to the user as big picture style notification.



**Figure 4.10: App Big Picture Style Notification**

User also can see the image notification inside the app.



**Figure 4.11: App Notification image**

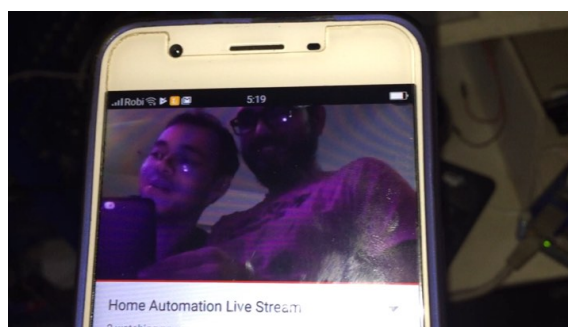
**Climate:** Here user can observe the current temperature and pressure inside the house.



**Figure 4.12: App Climate Screen**

If the temperature is above 50 Degree, raspberry pi will send a warning message via notification to the user.

**Security Camera:** When user clicks on "Camera" option then it will be open a youtube page where security camera continuously streaming the live view of inside or outside of the house.



**Figure 4.13: Security Camera Live Streaming**

### **4.3 Chapter Summary**

In this chapter we focused on our real time application that how are they worked and how was like the output and showing some sample code.

# 5 Conclusion

Completing an IoT project requires a great deal of knowledge and commitment. During this endeavor, it was necessary to gain a considerable knowledge on Java and Android Application Development, know how a connection works, acquire a more comprehensive understanding of electronics and of course, obtain the skill to utilize all of these together to assemble a fully functional system.

This project was started to help people to make their daily life safe and easier.

This project was also inspired from the problems that disabled people encounter in their everyday life while most of other people do not aware of their difficulties. One of the biggest needs required for disabled people is to continue their daily life activities when they are alone at home and there is nobody to help them.. There are many studies about smart houses but we observed that there is not enough smart home system aims to help disabled people. We added a new aspect to smart home systems by aiming to help disabled people.

In this project we suggested a new perspective to use of wireless sensor network to build smart home systems. We do not claim that the wireless protocol we used is the best solution for smart home systems. But it provides us easy implementation, installation and portability. After designing and implementation works finished, whole system is tested and we obtained successful results. Even though we implemented this system for disabled people, it can be used by healthy people. Because this system aims to make easy of peoples daily life at home. Actually disabled people encounter with more problems than mentioned in this project. For the future work, the actuation scenarios for disabled people can be increased and improved.

## 5.1 Limitations

Overall the system developed has limitation as follows :

- Security camera can not stream and record footage more than 6 hours.

- If the power cut off. Security camera need to start manually.

## **5.2 Future Works**

Time and other resources is always factor in a better product. There is always room for improvement and this project has a lot of areas that needs work.

### *5.2.1 Computer Vision*

Cloud Vision API will be added to our System.

#### *5.2.1.1 Powerful image Analysis*

Google Cloud Vision API enables developers to understand the content of an image by encapsulating powerful machine learning models in an easy to use REST API. It quickly classifies images into thousands of categories (e.g., "sailboat", "lion", "Eiffel Tower"), detects individual objects and faces within images, and finds and reads printed words contained within images. You can build metadata on your image catalog, moderate offensive content, or enable new marketing scenarios through image sentiment analysis. Analyze images uploaded in the request or integrate with your image storage on Google Cloud Storage.

#### *5.2.2 Face recognition*

Facial recognition (or face recognition) is a biometric method of identifying an individual by comparing live capture or digital image data with the stored record for that person. Facial recognition systems are commonly used for security purposes but are increasingly being used in a variety of other applications.

Face Recognition System will be added to this project.

#### *5.2.3 Biometric Door Lock*

Most electronic door locks come with one or more means of entry, which include RFID, keypad, biometric (fingerprint), or Bluetooth or internet. All you do is program your fingerprint, or those you want to have access to your home, and the system knows this is an acceptable person to unlock for.

Fingerprint door lock system will also be added in future.



## References

- [1] amazon, “Raspberry pi 3 model b+,” [Online]. Available: <https://www.amazon.com/Raspberry-Pi-Model-512MB-Computer/dp/B00LPESRUK>.
- [2] HACKLE, “Raspberry pi 3 gpio pin diagram,” [Online]. Available: <https://hackle.com/44771052-2/>.
- [3] adafruit.com, “Pi noir camera v2,” [Online]. Available: <https://www.adafruit.com/product/3100>.
- [4] modmypi.com, “Diy night vision camera,” [Online]. Available: <https://www.modmypi.com/raspberry-pi/camera/camera-boards/raspberry-pi-night-vision-camera>.
- [5] maximintegrated.com, “Ds18b20 temperature sensor,” [Online]. Available: <https://www.maximintegrated.com/en/products/analog/sensors-and-sensor-interface/DS18B20.html>.
- [6] adafruit.com, “Pir sensor,” [Online]. Available: <https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor?view=all>.
- [7] M. Products, “Case/power supply fan,” [Online]. Available: <http://www.manhattan-products.com/case-power-supply-fan21>.
- [8] adafruit.com, “Piezo buzzer,” [Online]. Available: <https://www.adafruit.com/product/160>.
- [9] wikipedia.org, “Home automation,” [Online]. Available: <https://en.wikipedia.org/wiki/Home-automation>.

- [10] Nitin, “smart home automation,” [Online]. Available: <http://www.smarthomeautomation2020.com/>.
- [11] xfinity, “Home automation benefits,” [Online]. Available: <https://www.xfinity.com/hub/smart-home/home-automation>.
- [12] C. Gibson, “Home automation type,” [Online]. Available: <https://www.homeimprovementpages.com.au/article/typeshomeautomation>.
- [13] R. Pi, “Raspberry pi,” *Raspberry Pi*, vol. 1, p. 1, 2013.
- [14] weworkweplay.com, “Automatically connect raspberry pi to a wi-fi network,” [Online]. Available: <http://weworkweplay.com/play/automatically-connect-a-raspberry-pi-to-a-wifi-network/>, Accessed on 2017-03-01.
- [15] Raspberrypi.org, “Raspberrypi power,” [Online]. Available: <https://www.raspberrypi.org/documentation/hardware/raspberrypi/usb/>.
- [16] raspberrypi.org, “An introduction to gpio and physical computing on raspberry pi,” [Online]. Available: <https://www.raspberrypi.org/documentation/usage/gpio/>.
- [17] Raspberrypi.org, “Raspbian jessie with desktop,” [Online]. Available: <https://www.raspberrypi.org/downloads/raspbian/>.
- [18] android.com, “Android things,” [Online]. Available: <https://developer.android.com/things/preview/download.html>.
- [19] raspberrypi.org, “Virtual network computing,” [Online]. Available: <https://www.raspberrypi.org/documentation/remote-access/vnc/README.md>.
- [20] C. Cawley, “Raspberry pi 3 youtube live streaming,” [Online]. Available: <http://www.makeuseof.com/tag/live-stream-youtube-raspberry-pi/>.
- [21] stackoverflow, “Xml,” [Online]. Available: <https://stackoverflow.com/tags/android-xml/info>.
- [22] R. Nurik, “Material design,” [Online]. Available: <https://android-developers.googleblog.com/2014/08/material-design-in-2014-google-io-app.html>.

- [23] Tutorialspoint, “Java (programming language),” [Online]. Available: [https://www.tutorialspoint.com/java/java\\_overview.htm](https://www.tutorialspoint.com/java/java_overview.htm), Accessed on 2017-01-01.
- [24] R. Rogers, J. Lombardo, Z. Mednieks, and B. Meike, *Android application development: Programming with the Google SDK*. O’Reilly Media, Inc., 2009.
- [25] Wikipedia.org, “Android sdk,” [Online]. Available: <https://en.wikipedia.org/wiki/Android-software-development>.
- [26] wikipedia.org, “Gradle,” [Online]. Available: <https://en.wikipedia.org/wiki/Gradle>.
- [27] F. team, “Ffmpeg,” [Online]. Available: <https://en.wikipedia.org/wiki/FFmpeg>.
- [28] google.com, “Fire-base,” [Online]. Available: <https://firebase.google.com/products/>.

# A Appendix

## A.1 Logic & Code

For this system we use Raspberry pi 3 model b as a micro controller. We implemented our logic and code in Raspberry pi 3. So all the code are given below.

### A.1.1 Light control

**Listing A.1: Light Control Code**

```
1
2 package com.heisenberg.smarthome.model;
3
4 /**
5  * Created by IOT on 7/9/2017.
6  */
7
8 public class LightControlModel {
9
10     int bedRoomLightStatus , drawingRoomLightStatus ,
11         kitchenLightStatus , lawnLightStatus ;
12
13     public int getBedRoomLightStatus () {
14         return bedRoomLightStatus ;
15     }
16
17     public int getDrawingRoomLightStatus () {
18         return drawingRoomLightStatus ;
19     }
```

```

19
20     public int getKitchenLightStatus () {
21         return kitchenLightStatus;
22     }
23
24     public int getLawnLightStatus () {
25         return lawnLightStatus;
26     }
27
28     @Override
29     public String toString () {
30         return "LightControlModel{" +
31             .....
32             .....
33     }
34
35     public void setBedRoomLightStatus(int
36         bedRoomLightStatus) {
37         this.bedRoomLightStatus = bedRoomLightStatus;
38     }
39
40     public void setDrawingRoomLightStatus(int
41         drawingRoomLightStatus) {
42         this.drawingRoomLightStatus =
43         drawingRoomLightStatus;
44     }
45
46     public void setKitchenLightStatus(int
47         kitchenLightStatus) {
48         this.kitchenLightStatus = kitchenLightStatus;
49     }
50
51     public void setLawnLightStatus(int lawnLightStatus)
52     {
53         this.lawnLightStatus = lawnLightStatus;
54     }

```

```

49     }
50
51     public LightControlModel() {
52
53     }
54
55     public LightControlModel(int bedRoomLightStatus ,
56                               int drawingRoomLightStatus , int
57                               kitchenLightStatus , int lawnLightStatus) {
56
57         this.bedRoomLightStatus = bedRoomLightStatus;
58         .....
59         .....
60     }
61 }

```

#### A.1.2 Fan control

**Listing A.2: Fan Control Code**

```

1  package com.heisenberg.smarthome.model;
2
3  /**
4   * Created by IOT on 7/9/2017.
5   */
6
7  public class FanControlModel {
8
9      int bedRoomFanStatus , drawingRoomFanStatus ,
10     kitchenFanStatus , kidsRoomFanStatus ;
11
12     public FanControlModel() {
13
14     }
15

```

```

16     public FanControlModel(int bedRoomFanStatus , int
        drawingRoomFanStatus , int kitchenFanStatus , int
        kidsRoomFanStatus) {
17         .....
18         .....
19         .....
20     }
21
22     public void setBedRoomFanStatus(int
        bedRoomFanStatus) {
23         this.bedRoomFanStatus = bedRoomFanStatus ;
24     }
25
26     public void setDrawingRoomFanStatus(int
        drawingRoomFanStatus) {
27         this.drawingRoomFanStatus =
        drawingRoomFanStatus ;
28     }
29
30     public void setKitchenFanStatus(int
        kitchenFanStatus) {
31         this.kitchenFanStatus = kitchenFanStatus ;
32     }
33
34     public void setKidsRoomFanStatus(int
        kidsRoomFanStatus) {
35         this.kidsRoomFanStatus = kidsRoomFanStatus ;
36     }
37
38     public int getBedRoomFanStatus() {
39         return bedRoomFanStatus ;
40     }
41
42     public int getDrawingRoomFanStatus() {
43         return drawingRoomFanStatus ;
44     }
45

```

```

46     public int getKitchenFanStatus () {
47         return kitchenFanStatus;
48     }
49
50     public int getKidsRoomFanStatus () {
51         return kidsRoomFanStatus;
52     }
53
54     @Override
55     public String toString () {
56         return "FanControlModel{" +
57             "bedRoomFanStatus=" + bedRoomFanStatus
58             .....
59             .....
60             .....
61             '}' ;
62     }
63 }

```

### A.1.3 Doorbell control

**Listing A.3: Doorbell Control Code**

```

1
2 package com.heisenberg.smarthome.doorbell;
3
4 import android.os.Handler;
5 import android.util.Log;
6
7 import com.google.android.things.pio.Gpio;
8 import com.google.android.things.pio.GpioCallback;
9 import com.google.android.things.pio.
    PeripheralManagerService;
10 import com.heisenberg.smarthome.pinprovider.
    RaspberryPiPinProvider;
11 import com.heisenberg.smarthome.systemcontroller.
    DoorBellListener;

```



```

12
13 import java.io.IOException;
14
15 /**
16  * Created by IOT on 7/24/2017.
17  */
18
19 public class DoorBellController {
20     boolean btnPresses = false;
21     private Gpio mButtonGpio;
22     DoorBellListener mDoorBellListener;
23
24     public DoorBellController(PeripheralManagerService
        service , RaspberryPiPinProvider
        raspberryPiPinProvider , DoorBellListener
        doorBellListener){
25         try {
26
27             mDoorBellListener=doorBellListener;
28             mButtonGpio = service.openGpio(
                raspberryPiPinProvider.getDoorBell());
29             mButtonGpio.setDirection(Gpio.DIRECTION_IN)
                ;
30             mButtonGpio.setEdgeTriggerType(Gpio.
                EDGE_FALLING);
31             mButtonGpio.registerGpioCallback(new
                GpioCallback() {
32                 @Override
33                 public boolean onGpioEdge(Gpio gpio) {
34
35
36                     if (!btnPresses) {
37                         //takePicture();
38                         mDoorBellListener.
                            onDoorBellPressed();
39                         btnPresses = true;

```

```

40         new Handler().postDelayed(new
41             Runnable() {
42                 @Override
43                 public void run() {
44                     .....
45                     .....
46                 }
47             }, 10000);
48         try {
49             Log.e("Button", gpio.
50                 getValue() + "");
51         } catch (IOException e) {
52             e.printStackTrace();
53         }
54         // Return true to continue
55         // listening to events
56         return true;
57     }
58 });
59 } catch (IOException e) {
60     Log.e("button", "button driver error " + e.
61         toString());
62 }
63
64
65 }

```

#### A.1.4 Mobile Application Design Code

Key code of mobile application design code using XML.

#### Listing A.4: Desing Mobile Application

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.
   com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical"
8     android:background="@color/colorPrimary"
9     tools:context="com.heisenberg.smarthome.ui.login.
   view.LoginActivity">
10
11     <ImageView
12         android:id="@+id/ivLogo"
13         android:layout_width="200dp"
14         android:layout_height="200dp"
15         android:src="@drawable/splash_logo"
16         android:layout_centerHorizontal="true"/>
17
18     <com.heisenberg.smarthome.ui.ui_components.
   TextViewGothamBold
19         android:id="@+id/tvSignIn"
20         android:layout_width="wrap_content"
21         .....
22         .....
23     />
24
25     <android.support.design.widget.TextInputLayout
26         android:id="@+id/tiUserEmail"
27         android:layout_below="@id/tvSignIn"
28         .....
29         .....
30         app:hintTextAppearance="@style/Widget.Design.
   TextInputLayout"
31         android:theme="@style/Widget.Design.
   TextInputLayout">
```

```

32     >
33
34     <EditText
35         android:id="@+id/etUserEmail"
36         android:layout_width="match_parent"
37         .....
38         .....
39     />
40
41 </android.support.design.widget.TextInputLayout>
42
43 <android.support.design.widget.TextInputLayout
44     android:id="@+id/tiUserPassword"
45     android:layout_below="@id/tiUserEmail"
46     .....
47     .....
48     app:hintTextAppearance="@style/Widget.Design.
49         TextInputLayout"
49     android:textColorHint="@color/colorAccent"
50     android:theme="@style/Widget.Design.
51         TextInputLayout">
52
53     <EditText
54         android:id="@+id/etUserPassword"
55         android:layout_width="match_parent"
56         .....
57         .....
58     />
59 </android.support.design.widget.TextInputLayout>
60
61 <Button
62     android:id="@+id/btnLogin"
63     android:layout_width="wrap_content"
64     .....
65     .....
66 />

```

```

67     <com.heisenberg.smarthome.ui.ui_components.
        TextViewGothamRegular
68         android:id="@+id/tvText"
69         android:layout_alignParentBottom="true"
70         .....
71         ..... / >
72     <com.heisenberg.smarthome.ui.ui_components.
        TextViewGothamBold
73         android:id="@+id/tvSignUp"
74         .....
75         ..... / >
76
77     <com.wang.avi.AVLoadingIndicatorView
78         android:id="@+id/pdLoadingView"
79         .....
80         .....
81
82     />
83
84 </RelativeLayout>

```

#### A.1.5 Raspberry pi Camera Streaming Setup Code

Key code of setting Camera.

**Listing A.5: libavcodec**

```

1  if (!av_strcasecmp(tagname, "font")) {
2      if (tag_close && sptr > 0) {
3          struct font_tag *cur_tag = &stack[sptr--];
4      @@ -209,48 +207,46 @@ int ff_htmlmarkup_to_ass(void *
          log_ctx, AVBPrint *dst, const char *in)
5
6      *new_tag = stack[sptr++];
7      while (param) {
8          if (!av_strncasecmp(param, "size=", 5)) {
9              param += 5 + (param[5] == ' ');

```

```

10     if (sscanf(param, "%u", &new_tag->size) == 1)
11         av_bprintf(dst, "{\\fs%u}", new_tag->size);
12     }
13     else if (!av_strncasecmp(param, "color=", 6)) {
14         int color;
15         param += 6 + (param[6] == ' ');
16         color = html_color_parse(log_ctx, param);
17         if (color >= 0) {
18             new_tag->color = 0xff000000 | color;
19             av_bprintf(dst, "{\\c&H%X&}",
20                 new_tag->color & 0xffff);
21         }
22     }
23     else if (!av_strncasecmp(param, "face=", 5)) {
24         param += 5 + (param[5] == ' ');
25         _____
26         param[-1] == ' ' ? "\\ " : " ";
27         av_strlcpy(new_tag->face, param,
28
29             FFMIN(sizeof(new_tag->face), len+1));
30         param += len;
31         av_bprintf(dst, "{\\fn%s}", new_tag->
32             >face);
33 +         while (param) {
34 +             if (!av_strncasecmp(param, "size=", 5)) {
35 +                 param += 5 + (param[5] == ' ');
36 +                 if (sscanf(param, "%u", &new_tag->size)
37 == 1)
38 +
39                 av_bprintf(dst, "{\\fs%u}", new_tag->
40                 >size);
41 +
42             }
43         else if (!av_strncasecmp(param, "color=", 6)) {
44 +             int color;
45 +             param += 6 +
46 +             (param[6] == ' ');
47 +             color = html_color_parse(log_ctx,

```

```
47         param);
48 +         if (color >= 0) {
49 +             new_tag->color = 0xff000000 | color;
50 +
51 av_bprintf(dst, "\\c&H%X&}", new_tag->color &
52 0xffffffff);
53             }
```