Smart Home Security Application Enabled by IoT: Using Arduino, Raspberry Pi, NodeJS, and MongoDB

Chad Davidson[1], Tahsin Rezwana[2], and Mohammad A. Hoque[2(✉)]

[1] Department of Electrical Engineering and Computer Science, University of Tennessee Knoxville, Knoxville, TN 37996, USA
cdavid11@vols.utk.edu

[2] Department of Computing, East Tennessee State University, Johnson City, TN 37614, USA
{rezwana,hoquem}@etsu.edu

Abstract. Recent advances in smartphones and affordable open-source hardware platforms have enabled the development of low-cost architectures for IoT-enabled home automation and security systems. These systems usually consist of a sensing and actuating layer that is made up of sensors such as PIR (Passive Infra-red) sensors, also known as motion sensors; temperature sensors; smoke sensors, and web cameras for security surveillance. These sensors, smart electrical appliances and other IoT devices connect to the Internet through a home gateway. This paper lays out architecture for a cost effective "smart" door sensor that will inform a user through an Android application, of door open events in a house or office environment. The proposed architecture uses an Arduino-compatible Elegoo Mega 2560 microcontroller (MCU) board along with the Raspberry Pi 2 board for communicating with a web server that implements a RESTful API. Several programming languages are used in the implementation and further applications of the door sensor are discussed as well as some of its shortcomings such as possible interference from other RF (Radio Frequency) devices.

Keywords: Smart home · Security · IoT · Arduino · Raspberry Pi
NodeJS · MongDB · RF transmission

1 Introduction

Smart home is a section of the Internet of Things (IoT) paradigm that aims to integrate home automation and security. Enabling objects in a typical household to be connected to the Internet allows home-owners to remotely monitor and control them. From lamps that are set on timers to turn off at a specific time of the day, to smart thermostats that will regulate the temperatures in a house and generate detailed reports about energy usage, smart homes have found it's

A.-S. K. Pathan et al. (Eds.): SGIoT 2018, LNICST 256, pp. 46–56, 2019.
https://doi.org/10.1007/978-3-030-05928-6_5

niche in the consumer market. The availability of affordable smartphones, microcontrollers and other open-source hardware along with the increasing use of cloud services, has made it possible to develop low-cost smart home security systems. With families having busier lives than ever, smart home automation and security systems can also cater to household members with limited mobility such as the handicapped and the old.

The purpose of this paper is to present a low-cost architecture using RF based communication in a household to create an IoT-enabled smart home security system. Smart home devices that typically consume low power such as smart bulbs and door or window sensors use RF transceivers to communicate with each other. In this paper, an inexpensive architecture is proposed for a smart door sensor that will utilize an Elegoo Mega 2560 microcontroller board, Raspberry Pi 2, a web server, and an Android application.

2 Related Work

Prior work in IoT-enabled home security system has proposed architectures that focused on the use of low-cost open-source hardware components like the Arduino and Raspberry Pi MCU boards and a combination of sensors. PIR (Passive Infrared) sensors are used to detect motion and can work in sync with a webcam that captures images to alert users of trespassing.

Kodali et al. describe a cost-effective wireless home security and automation system based on the TI-CC3200 LaunchPad: a battery-powered microcontroller unit (MCU) with built-in WiFi connectivity [1]. PIR motion sensors are placed at the entrances to a building and connect to a digital in-out pin of the MCU. The MCU is programmed using Energia IDE (Integrated Development Environment) and Wi-Fi enabled. Kodali et al.'s configuration allows mobile phones without Internet connectivity to receive security alerts and control IoT devices connected to the microcontroller. Tanwar et al. describe an inexpensive home security system that implements a real-time email alert system [2]. The system uses a PIR module and a Raspberry Pi MCU. Security cameras and PIR sensors are connected to the Raspberry Pi via USB ports and GPIO (General Purpose Input/Output) pins respectively. The system assumes that homes have Internet access; it uses the Internet to send e-mails to the resident in real-time. The system's intrusion detection logic identifies motion by comparing signal inputs from the PIR sensors with their previous values. When current and previous signals differ, the security camera captures an image that is stored temporarily on the Raspberry Pi and then automatically e-mailed to the resident.

Gupta and Chhabra describe a cost-effective Ethernet-based smart home system for monitoring energy consumption, smoke and temperature levels and detecting trespassing [3]. This system uses the Arduino-certified Intel Galileo 2nd generation microcontroller board. Temperature, smoke and PIR sensors are connected directly to the microcontroller, while four 220 V devices are connected via a relay module. An android based mobile app that connects to the Intel Galileo based server over the Internet allows users to toggle switching devices by tap-to-touch or voice commands through Google API speech recognition tools.

Piyare et al. present a Bluetooth-based home automation system where an Android cell phone running a Python script communicates with an Arduino BT board with digital and analog input/output ports to which sensors and appliances are connected [4]. The smartphone application has a toggle on and off feature for each device. However, Bluetooth connectivity between the smartphone and the Arduino BT board required a range of 50 meters or less within a concrete building and mobile platforms other than Symbian do not support the Pyhton application.

Behera et al. designed and implemented a real-time smart home automation system using an Arduino Uno board along with an Arduino Wi-Fi Shield and a PC home server [5]. A PIR or motion sensor, an LDR (Light Dependent Resistor) and an LM35 temperature sensor were used to collect data which was made available on the PC server that also implemented a MATLAB-GUI platform to control the temperature, lights, and fans. The PIR sensor also acted as a security component by detecting possible intrusions and setting off a buzzer to alert the residents.

Howedi et al. proposed a low-cost smart home system built upon a similar architecture using the Arduino Uno board, PIR sensors, DHT11 temperature sensors, INA219 high side DC current sensor and servo motors that control doors and windows [6]. The Arduino IDE is used to implement the control and monitoring module of the system while the MIT App Inventor is used to develop a simple Android application.

Panwar et al. implemented the Eyrie smart home automation system using the Raspberry Pi 3 MCU as the central hub [7]. Their proposed architecture connected several Arduino Nano boards located around the house to various types of sensors and NRF24L trans-receivers that eliminated the need for Ethernet or Wi-Fi connectivity. Mosquito Broker, an open-source message broker used for relaying messages to the Raspberry Pi 3, operates using the Debian OS. Eclipse SmartHome framework was used to implement a web interface and a smartphone app for end-users.

In [8], a home automation system with smart task scheduling is developed making use of wireless ZigBee to connect appliances and wired X10 technology to connect light and switch modules to an Arduino microcontroller. An Ethernet shield mounted on the Arduino MCU allows communication between Arduino and a Web-based Android application which is then used to remotely add and manage devices and view recommended scheduling.

ShariqSuhail et al. implemented a prototype for smart home security system that uses PIR sensors for intrusion detection, MQ2 sensors for detecting smoke and gas leaks, LM35 temperature sensors as input to an Arduino Mega 2560 [9]. A buzzer, LCD, LED strip and a GSM module are outputs to this MCU board while a Raspberry Pi 2 board is used to include a webcam that captures images upon motion detection. GSM (Global System for Mobile communication), a wireless technology interfaces with the Arduino Mega to send SMS notifications and calls to the user's cell phone whenever potential intrusion, smoke or gas leak is detected.

In [10], a similar architecture utilizes an Arduino Mega 2560 board with a Wi-Fi module to implement a voice-controlled smart home system. The Elechouse V3 voice recognition module allows users to send voice commands to adjust lighting, open or close windows and control a folding bed.

Vineeth's et al. voice-controlled secure eHome also make use of the V3 voice recognition module but use an RF module instead of Wi-Fi for wireless communication between an Arduino UNO and Raspberry Pi MCUs [11]. The Raspberry Pi supports sensor connectivity to the Internet so all sensory data can be logged onto a Google spreadsheet. There is no implementation of a mobile or web app thus confining the controlling of this system to the location of the mic.

Sunehra et al. propose two schemes for a speech-based home automation system [12]. The first scheme uses HC-05 Bluetooth module along with Arduino Bluetooth controller mobile application to control appliances when inside the house. GSM/GPRS technology is used to remotely control appliances and receive SMS alerts for possible intrusion detections. The ARM11 Raspberry Pi board acts as the central hub for receiving voice commands though the HC-05 Bluetooth module and connects to a PIR sensor, relays, a Wi-Fi router and a webcam.

3 System Architecture and Design

3.1 Devices and Sensors

- Elegoo Mega 2560 Board (or Arduino)
- RF Receiver-Transmitter Pair (433 Hz)
- Magnetic Reed Switch
- Raspberry Pi 2
- Female to Female-Female to Male Jumper Cables
- Android phone or emulator

3.2 Schematics

Figures 1 and 2 describe the schematics for wiring the Elegoo Mega and Raspberry Pi boards. In Fig. 1, a magnetic reed switch and RF 433 Hz transmitter is attached to the Mega 2560 board. The ordering of the wires do not matter for the reed switch. One wire leads to ground and for the code located in the repository, pin 2 on the Mega 2560 board is where the other wire will lead. The RF transmitter has leads to ground, a 5 V power supply, and a lead to pin 10 on our board. The antenna used for this project is a simple rolled piece of aluminum foil.

Figure 2 presents a diagram of the pinout of the Raspberry Pi 2 attached to an RF Receiver. The RF Receiver has leads to ground, a 5 V power supply, and to pin 13 on the Raspberry Pi (used in OpenDoor.cpp in repository). Note that there is a second Data pin that is not to be utilized in this project and an antenna (not pictured) was formed out of rolled aluminum foil.

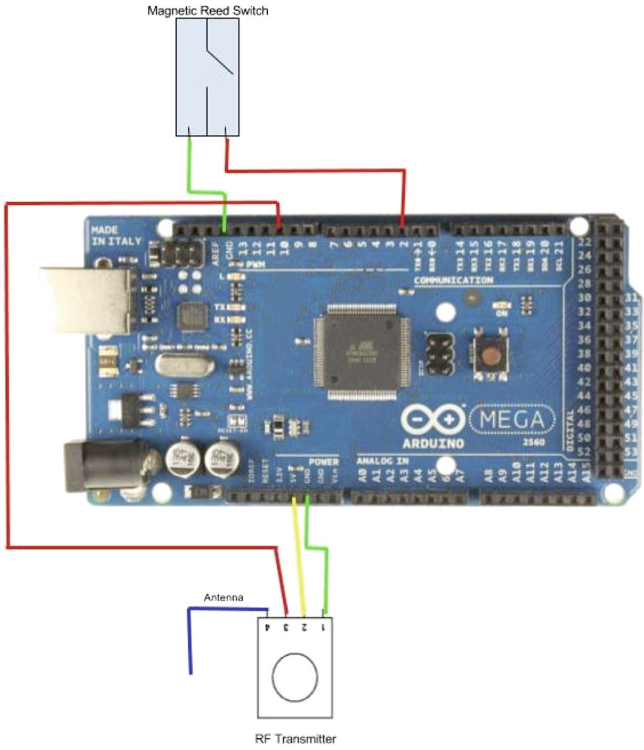

Fig. 1. Mega 2560 board with magnetic reed switch and RF transmitter diagram

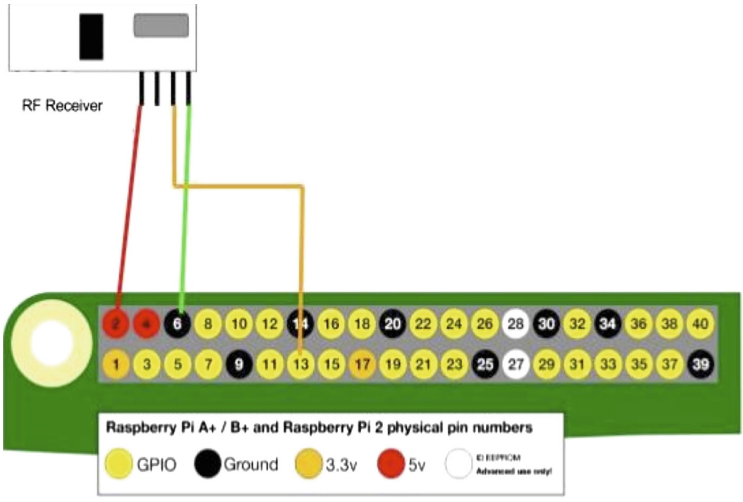

Fig. 2. Raspberry Pi-2 pinout diagram with RF Receiver Wiring

3.3 System Architecture

Figure 3 describes the overall architecture of the smart home security system. Figure 4 shows the flow diagram of data communication. Each of the following subsections provide a detailed explanation of how the devices communicate with each other. A simple overview of the system architecture will help understand why the sections have been laid out in the order they are. Initially, the reed switch is opened which causes the Elegoo board to send an RF signal from its transmitter to the RF receiver located on the Raspberry Pi 2. The Raspberry Pi then sends an HTTP POST request to a RESTful web server that is set up in the cloud. This web server then either pushes information, or receives GET requests from an Android application to allow the end user to view the door open events by date and time. All libraries mentioned in the following sections are open source.

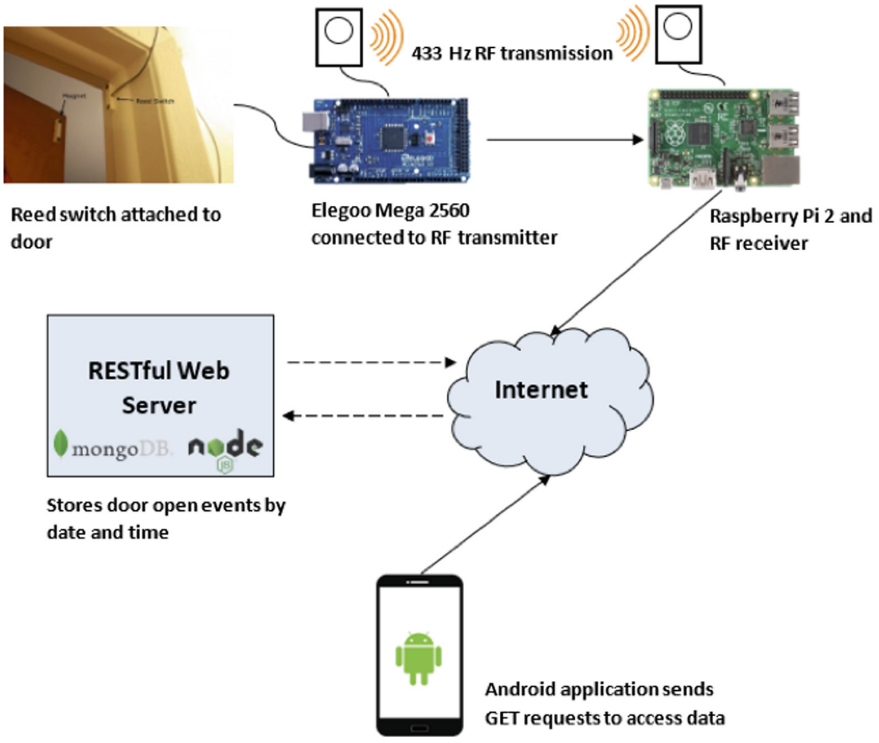

Fig. 3. System architecture for IoT-based smart home security

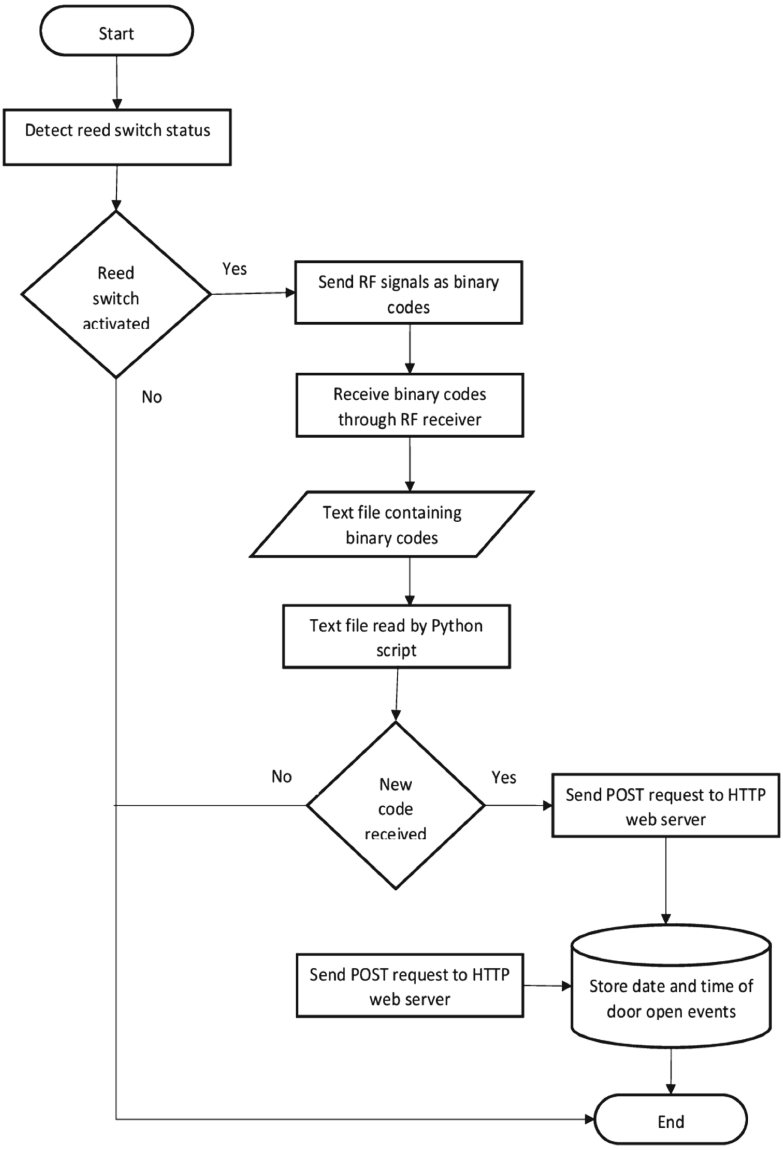

Fig. 4. Flowchart representing data flow through the system.

3.4 Communication Between Elegoo and Raspberry Pi-2

The connection between the Elegoo and the Raspberry Pi is the most important part as it is where the sensor in the IoT architecture communicates with another device. Once the door is open and the reed switch is activated, an RF transmission of 433 Hz is sent from the transmitter using the rc-switch library for the

Arduino. A binary code is sent to the receiver connected to the Raspberry Pi and it is incremented for every subsequent door open event. This enables the Raspberry Pi to keep track of individual door open events. The RF transmission range was tested in a 1200 sq. ft apartment. When the reed switch is activated, a constant stream of binary numbers is transmitted to the receiver attached to the Raspberry Pi. Using a simple piece of aluminum foil as antennae on the receiver and the transmitter, the entire 1200 sq. footage of the house is covered. The Raspberry Pi utilized the wiringPi library and the 433Utils library to receive the binary codes. These codes are then sent as a text file output that is read by a Python script that will perform the next step of the communication process.

3.5 Communication Between Raspberry Pi-2 and NodeJS Web Server

When the binary codes have been received and sent as output to a text file, a Python script utilizing the Requests library checks the text file every second for new updates. When a new code is placed in the text file, a POST request is then sent to an HTTP web server that is set up with NodeJS and a MongoDB database. This server implements a RESTful API that stores door open events by date and time. The data located in this database are accessible via GET requests that can be performed by the Android application.

3.6 Communication Between Web Server and Android Device

The final step in the communication process is between the web server and the Android application written in Java. It is possible to send push notifications whenever a new door open event is detected by the web server, however, for the purposes of this project, the Android application makes GET requests to the server. The Android application uses a variety of libraries that are available through the Android Studio but the library used to make the requests is the Volley library.

4 Future Scope

This design can be used as a reference for further applications to be developed with the current sensor architecture, and it provides a framework using the Raspberry Pi through which other sensors can be added to the smart home network.

4.1 Reed Switch Applications

The door sensor provides a way of seeing whether or not a door has been opened. The most obvious way to extend this application is to record the amount of time a door is open. While this may be useful to individual end users, there lies a possible business interest in knowing how often the doors to a store are opened

and closed. This information can then be used by businesses to lower their energy costs.

A second future application for this architecture could include connecting a Bluetooth module to the Mega 2560 board to identify individuals entering through the door by pairing with an end users phone. It could also be used by parents to check on a child that may try to leave home at an odd hour.

4.2 Further Raspberry Pi Applications

Since the Raspberry Pi uses an RF receiver it can not only receive the transmission from the magnetic reed switch but also be outfitted with the ability to receive transmissions from other RF-enabled devices in the household. The Raspberry Pi could be used as a hub for RF-enabled smart home devices throughout the house. The concept can be used to develop a more advanced home security system that would include PIR motion sensors placed in other areas of the house to detect intrusion while homeowners are away. Using, RF-enabled bulbs the smart home system can be extended to include energy management inside the house.

The Raspberry Pi can serve as a fog computing device which can store information locally before sending it to the cloud server. This can be useful when applications might need real time support or low latency that cannot be provided by the cloud service.

4.3 Android Application Improvements

The Android application for this project, written in Java, is a basic application. Future work may involve adding time zone support from the device so as not to depend solely on Unix timestamps. Multiple options to view and display the data located in the MongoDB database, can also be added. A calendar feature can be integrated so users can have a more robust view of the door open events in their household. The system can be enhanced to display alerts to users in the form of SMS or email in the event of trespassing.

5 Limitations

Potential issues may arise through interference on the 433 Hz RF frequency. Many home devices use RF signals to communicate and at a given time there may be more than one RF receiver trying to send signals to the Raspberry Pi or it could be picking up signals that it wasn't intended to receive. An interference testing with the RF units can be done as a part of future work. In the case of multiple transmitters attempting to communicate with the Raspberry Pi, there would need to be a registration system in place on the Raspberry Pi that kept track of incoming signals and their sources. However, the architecture proposed here does not provide that support.

6 Conclusion

This paper presents an architecture that can be used as framework to build a low-cost smart home security system. Using affordable components such as microcontrollers from Elegoo and Raspberry Pi and RF signals as a communication channel between these devices, it was possible to develop an IoT system that allows users of a household to view when a particular door has been opened. Schematics for connecting the different components have been provided along with figures to demonstrate them. The data flow between each of these devices have been explained and potential issues that may arise have been discussed. Finally, future work in this area along with potential use cases for this architecture have also been discussed.

References

1. Kodali, R.K., Jain, V., Bose, S., Boppana, L.: IoT based smart security and home automation system. In: 2016 International Conference on Computing, Communication and Automation (ICCCA), pp. 1286–1289. IEEE, April 2016
2. Tanwar, S., Patel, P., Patel, K., Tyagi, S., Kumar, N., Obaidat, M.S.: An advanced Internet of Thing based security alert system for smart home. In: 2017 International Conference on Computer, Information and Telecommunication Systems (CITS), pp. 25–29. IEEE, July 2017
3. Gupta, P., Chhabra, J.: IoT based Smart Home design using power and security management. In: 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH). IEEE (2016)
4. Piyare, R., Tazil, M.: Bluetooth based home automation system using cell phone. In: 2011 IEEE 15th International Symposium on Consumer Electronics (ISCE), pp. 192–195. IEEE, June 2011
5. Behera, A.R., Devi, J., Mishra, D.S.: A comparative study and implementation of real time home automation system. In: 2015 International Conference on Energy Systems and Applications, pp. 28–33. IEEE, October 2015
6. Howedi, A., Jwaid, A.: Design and implementation prototype of a smart house system at low cost and multi-functional. In: Future Technologies Conference (FTC), pp. 876–884. IEEE, December 2016
7. Panwar, A., Singh, A., Kumawat, R., Jaidka, S., Garg, K.: Eyrie smart home automation using Internet of Things. In: Computing Conference, 2017, pp. 1368–1370. IEEE, July 2017
8. Baraka, K., Ghobril, M., Malek, S., Kanj, R., Kayssi, A.: Low cost arduino/android-based energy-efficient home automation system with smart task scheduling. In: 2013 Fifth International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN), pp. 296–301. IEEE, June 2013
9. ShariqSuhail, M., ViswanathaReddy, G., Rambabu, G., DharmaSavarni, C.V.R., Mittal, V.K.: Multi-functional secured smart home. In: 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 2629–2634. IEEE (2016)
10. Gunputh, S., Murdan, A.P., Oree, V.: Design and implementation of a low-cost Arduino-based smart home system. In: 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN), pp. 1491–1495. IEEE, May 2017

11. Vineeth, K.S., Vamshi, B., Mittal, V.K.: Wireless voice-controlled multi-functional secure ehome. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 2235–2240. IEEE, September 2017
12. Sunehra, D., Tejaswi, V.: Implementation of speech based home automation system using Bluetooth and GSM. In: 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), pp. 807–813. IEEE, October 2016