

---

# VOICE CONTROL BASED HOME AUTOMATION SYSTEM

---

16:332:567 SOFTWARE ENGINEERING COURSE PROJECT  
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

FALL, 2013

EDITED BY

LI XINYU

GU YUE

GUO JIAQI

NAYYAR VIDUR

SHU CHANG

ZHANG LUFAN

*The State University of New Jersey  
Rutgers*

FOR DETAILS

[HTTPS://SITES.GOOGLE.COM/SITE/VCHOMEAUTOMATION/](https://sites.google.com/site/vchomeautomation/)

# Contents

<b>1</b>	<b>CONTRIBUTION BREAKDOWN</b>	<b>5</b>
1.1	Contribution Breakdown of Report 1 . . . . .	5
1.2	Contribution Breakdown of Report 2 . . . . .	5
1.3	Contribution Breakdown of Report 3 . . . . .	6
1.4	Contribution Breakdown of Demo 1 . . . . .	7
1.5	Contribution Breakdown of Demo 2 . . . . .	7
<b>2</b>	<b>SUMMARY OF CHANGES</b>	<b>9</b>
2.1	Optimization of User Interface . . . . .	9
2.2	Expanding Server Database . . . . .	9
2.3	Intelligent Server Design . . . . .	9
2.4	Abundant Operations for Smart Elements . . . . .	9
2.5	Larger Library for Key Words . . . . .	9
2.6	Efficient Communication Protocol Design . . . . .	9
<b>3</b>	<b>Customer Statement of Requirements(CSR)</b>	<b>10</b>
3.1	Problem Statement . . . . .	10
3.1.1	Identify the human languages . . . . .	10
3.1.2	Abundant operations for elements . . . . .	10
3.1.3	Application on cellphone platform . . . . .	11
3.1.4	Feedback information . . . . .	11
3.1.5	Wireless communication and remote controlling . . . . .	11
3.1.6	User data . . . . .	12
3.1.7	Check the status of elements . . . . .	13
3.2	Glossary of Terms . . . . .	13
<b>4</b>	<b>System Requirements</b>	<b>17</b>
4.1	Enumerated Functional Requirements . . . . .	17
4.2	Enumerated Nonfunctional Requirements . . . . .	18
4.3	On-Screen Appearance Requirements . . . . .	20
<b>5</b>	<b>Functional Requirements Specification</b>	<b>22</b>
5.1	Stakeholders . . . . .	22
5.1.1	Internal parts . . . . .	22
5.1.2	External parts . . . . .	22
5.2	Actors and Goals . . . . .	22
5.3	Use Cases . . . . .	22
5.3.1	Casual Description . . . . .	22
5.3.2	Use Case Diagram . . . . .	25
5.3.3	Traceability Matrix . . . . .	32
5.3.4	Fully-Dressed Description . . . . .	32
5.4	System Sequence Diagrams . . . . .	34

<b>6</b>	<b>User Interface Specification</b>	<b>40</b>
6.1	Preliminary Design . . . . .	40
6.1.1	Log In . . . . .	40
6.1.2	Main Interface . . . . .	43
6.1.3	Music Interface . . . . .	46
6.1.4	Light interface . . . . .	49
6.2	User Effort Estimation . . . . .	51
6.2.1	User login . . . . .	51
6.2.2	User logout . . . . .	52
6.2.3	Light controlling . . . . .	52
6.2.4	Music controlling . . . . .	52
6.2.5	Modify the brightness . . . . .	52
6.2.6	Modify the volume of music . . . . .	53
6.2.7	Elements status check . . . . .	53
6.2.8	Control information feedback . . . . .	53
6.2.9	Light interface . . . . .	53
<b>7</b>	<b>Domain Analysis</b>	<b>55</b>
7.1	Domain Model . . . . .	55
7.1.1	Domain model-General . . . . .	55
7.1.2	Domain model-Server . . . . .	56
7.1.3	Domain model-Arduino . . . . .	57
7.1.4	Domain Model-The communication protocol . . . . .	59
7.2	System Operation Contracts . . . . .	59
7.2.1	Light controlling . . . . .	59
7.2.2	Music controlling . . . . .	61
7.2.3	Modify the brightness . . . . .	61
7.2.4	Modify the volume of music . . . . .	61
7.2.5	Elements status check . . . . .	61
7.2.6	User login . . . . .	61
7.2.7	Create new users accounts . . . . .	62
7.2.8	Remote user . . . . .	62
7.2.9	User log out . . . . .	62
7.2.10	Check the authority of user . . . . .	62
7.2.11	Network error notification . . . . .	62
7.3	Mathematical Model . . . . .	63
7.3.1	The fuzzy recognition model . . . . .	63
<b>8</b>	<b>Interaction Diagrams</b>	<b>64</b>
8.1	Design Patterns . . . . .	64
8.2	UML Diagrams . . . . .	66
8.3	Alternative solution description . . . . .	75

<b>9</b>	<b>Class Diagram and Interface Specification</b>	<b>78</b>
9.1	Class Diagram . . . . .	78
9.2	Data Types and Operations Signature . . . . .	80
9.3	Traceability Matrix . . . . .	91
9.4	Object Constraint Language (OCL) Contracts . . . . .	91
<b>10</b>	<b>System Architecture and System Design</b>	<b>94</b>
10.1	Architectural Styles . . . . .	94
10.2	Package Diagram . . . . .	96
10.3	Database . . . . .	97
10.4	The thread algorithm of the Server . . . . .	98
<b>11</b>	<b>Hardware Development</b>	<b>100</b>
11.1	ARDUINO . . . . .	102
11.2	ATtiny-85 . . . . .	104
11.3	Part List detail and their role . . . . .	105
11.4	Integrating the Hardware . . . . .	112
11.5	Central Hub . . . . .	114
11.6	Light-Controller Node . . . . .	117
11.7	Music-Controller Node . . . . .	120
11.8	Communication between the hardware . . . . .	122
11.9	Working of Infrared Communication . . . . .	122
11.10	Working of Radio-Frequency Communication . . . . .	123
11.11	Multiplexing, Modulation, Algorithms and Protocols . . . . .	123
11.12	User Interface from Hardware . . . . .	127
11.13	Testing . . . . .	129
<b>12</b>	<b>Algorithm</b>	<b>144</b>
12.1	The keywords extraction algorithm . . . . .	144
12.2	The smart control algorithm . . . . .	144
12.3	The reminder algorithm . . . . .	145
<b>13</b>	<b>User interface design and implementation</b>	<b>146</b>
<b>14</b>	<b>Design of tests</b>	<b>150</b>
<b>15</b>	<b>Past Works and Future Plan</b>	<b>155</b>
15.1	Past Works . . . . .	155
15.2	Future Plans . . . . .	155
<b>16</b>	<b>Reference</b>	<b>157</b>



# 1 CONTRIBUTION BREAKDOWN

## 1.1 Contribution Breakdown of Report 1

Responsibility	Li Xinyu	Gu Yue	Guo Jiaqi	Nayyar Vidur	Shu Chang	Zhang Lufan
Project Management(10')	50%	30%	10%			10%
Section1 (9')			100%			
Section2 (6')		33%		33%		34%
Section3 (30')	7%	23%	22%	48%		
Section4 (15')		20%				80%
Section5 (25')	40%				60%	
Section6 (5')	40%	20%			20%	20%
Total (100')	18.5	16	16.5	16.5	16.5	16

Table 1: CONTRIBUTION BREAKDOWN FOR REPORT 1.

## 1.2 Contribution Breakdown of Report 2

Responsibility	Li Xinyu	Gu Yue	Guo Jiaqi	Nayyar Vidur	Shu Chang	Zhang Lufan
Section1 (30')		20%	57%	10%	13%	
Section2 (10')		100%				
Section3 (15')	41%			13%		46%
Section4 (4')	100%					
Section5 (11')						100%
Section6 (12')					100%	
Section7 (18')	40%			60%		
Total (100')	17	16	17	16	16	18

Table 2: CONTRIBUTION BREAKDOWN FOR REPORT 2.

All parts under Hardware Development have been designed and documented by Vidur Nayyar, but have not been mentioned in the contribution distribution table above. The contribution distribution table above is as per the guidelines.

### 1.3 Contribution Breakdown of Report 3

Report 3	Li Xinyu	Gu Yue	Guo Jiaqi	Nayyar Vidur	Shu Chang	Zhang Lufan
Percent	16.8	16.8	16.8	16.8	16.8	16

Table 3: CONTRIBUTION BREAKDOWN FOR REPORT 3.

All parts under Hardware Development have been designed and documented by Vidur Nayyar, but have not been mentioned in the contribution distribution table above. The contribution distribution table above is as per the guidelines.

## 1.4 Contribution Breakdown of Demo 1

Responsibility	Li Xinyu	Gu Yue	Guo Jiaqi	Nayyar Vidur	Shu Chang	Zhang Lufan
Part1 (35')	8.8	7.6	7.2	8	0.6	2.8
Part2 (10')	4.5	1.5				
Part3 (0')						
Part4 (10')	1.9	1.5	1.6	5		
Part5 (8')	3.3	0.8	0.3	1.5	0.2	0.9
Part6 (5')	1.3	1.1	0.3	1	0.7	0.6
Part7 (10')	3.3	1.5	0.3	3	1	0.3
Part8 (5')	5					
Part9 (0')						
Part10 (2')				2		
Part11 (3')	0.3		2.1	0.1		0.1
Part12 (12')	7.4	1.2	0.4	2.4	0.5	0.3
Total (100')	35.8	15.2	12.2	25	5	5
Equalize (100')	23.8	17.2	15.2	20	15	7

Table 4: CONTRIBUTION BREAKDOWN FOR DEMO 1.

## 1.5 Contribution Breakdown of Demo 2

Because we have two different demo2 in our group, here we will list two different contribution breakdown tables.

Table 5 is the contribution breakdown for the team of Xinyu Li, Yue Gu, Jiaqi Guo;

Table 6 is the contribution breakdown for the team of Chang Shu, Lufan Zhang, Vidur Nayyar.

Responsibility	Li Xinyu	Gu Yue	Guo Jiaqi
Part1 (35')	12.9	12.9	9.2
Part2 (10')	4.1	3.6	2.3
Part3 (0')			
Part4 (10')	3	3	4
Part5 (8')	3	3	2
Part6 (5')	1.8	1.8	1.4
Part7 (10')	3	3.5	3.5
Part8 (5')	5		
Part9 (0')			
Part10 (2')			2
Part11 (3')			3
Part12 (12')	3	5.25	3.75
Total (100')	35.8	33.05	31.15

Table 5: CONTRIBUTION BREAKDOWN FOR DEMO 2(Xinyu L, Jiaqi G, Yue G).

Responsibility	Shu Chang	Zhang Lufan	Nayyar Vidur
Part1 (35')	11.66	11.66	11.66
Part2 (10')	3.33	3.33	3.33
Part3 (0')			
Part4 (10')	3.2	1.6	4.9
Part5 (8')	2.7	2.7	2.7
Part6 (5')	1.6	1.6	1.6
Part7 (10')	3.3	3.3	3.3
Part8 (5')	5		
Part9 (0')			
Part10 (2')	1	1	
Part11 (3')	1.5		1.5
Part12 (12')	2.6	4.4	5
Total (100')	34.5	31	34.5

Table 6: CONTRIBUTION BREAKDOWN FOR DEMO 2(Chang Shu, Lufan Zhang, Nayyar Vidur).

## **2 SUMMARY OF CHANGES**

### **2.1 Optimization of User Interface**

In the final version of our home automation system, the user interface had been optimized. Most buttons and backgrounds are reprocessed by Photoshop and it looks more pretty than the first demo.

### **2.2 Expanding Server Database**

In this version of server, the database has been expanded to store more information. In the detail of design, the database stores the history of users operations additionally, in order to analyze the habits of different users to make server more intelligent.

### **2.3 Intelligent Server Design**

In the first demo, the function of server is to receive and detect the information send from the google voice and send the digital signal to the smart elements. In this version, the server had been redesigned. The main difference of this demo is intelligence of server. Based on the history of database of each users, the server can analyze the different habits of them and make the corresponding operation with different users, different locations and different time.

### **2.4 Abundant Operations for Smart Elements**

We make more operations in the final demo. In the first demo, we can only operate one lights and play and pause the music player. In this demo, there are two lights in different locations and more songs in the playlist. And with different kind of operation, we can turn on or off the different lights, or play the next or previous songs in the playlist.

### **2.5 Larger Library for Key Words**

In the redesigned server, the library of key words had been expanded. For example, turn on and switch on both can be recognized now for turning on the lights. This change is made for the different habits of speaking and make the server smarter.

### **2.6 Efficient Communication Protocol Design**

With the new designed protocol of this demo, the communication is more efficient. The UI offers a text box for users to input the IP address of server. And the server can show its IP address for users. Also with the new algorithm of code, thae server now can receive different informations from devices in the same time.

## 3 Customer Statement of Requirements(CSR)

### 3.1 Problem Statement

As the modern software system, home automation control system should achieve any requirements from users by its intelligence and generalization. In our home automation system, the requirements of customer can be concluded as below:

#### 3.1.1 Identify the human languages

It is impossible for any users to understand the instructions or algorithm of programming. What they can do is control the elements with human languages and signal of body. So the users require the system to identify the human languages instead of machine instructions or other programming languages.

**Human voice instruction** Voice is the most directly and simple way for people to convey their requirement. For example, when the sunlight is insufficient inside the room, the user can just send a voice message just like Turn on the light to modify the brightness of the room. This condition requires the excellent flexibility because different users maintain the various ways of speaking and the system should be compatible to cater to a large number of people. And this requirement is the core case of the whole system.

**Simple gestures** Some users also demand the system to be equipped with the alternative controlling ways. And gesture is a good choice. Gesture s the easiest way of communicating for some one who cannot speak. For example the user can slide one of his fingers to modify the volume of MP3 player. The technology of identifying gestures for system is complex in algorithm and hardware design, but it can really simplify the operation of users.



Figure 1: Model of User Input.

#### 3.1.2 Abundant operations for elements

In general the users have more requirements to control the elements of the system than just turning it on and off. For the lamps, if the users want to

modify the brightness of the room, users can change the brightness of the lamp instead of turning it on or off. Or if the user is using the MP3 player, he can modify the volume or skip to the next or to the last song. This requirement is important because the users in general have their own customs of using the system and the system should make the users comfortable with the abundant operations of the elements.

### **3.1.3 Application on cellphone platform**

When the users are using the system, it is inconvenient and impossible to hold several terminals such as cellphones, iPads and controllers. For users, it is significant to concentrate the functions of different devices into one terminal. In general, the most frequently used device is cellphone for users. So the users require the system to be developed on the cellphone platform such as Android and IOS. As one application of mobile device, the users can log into the home automation system directly with less cost than a whole controller. Android is the most widely used mobile system and it can be compatible with a large number of users if the system developed on Android. Another reason is that the Android system can easily process the voice information because it is a Linux based OS and it can easily access Google Voice. Also Android primarily uses for touchscreen devices [1]. So for users the Android system is the best choice of platform for our home automation system.

### **3.1.4 Feedback information**

At any point of time, the user might want to know the status of the different elements of the house. For this purpose the user would require the system to send a feedback to the user interface. The users should know whether the system recognizes the correct information and execute the right actions. Also if the identification failed, the users should be announced to send the messages again or cancel the action by the feedback information.

### **3.1.5 Wireless communication and remote controlling**

The users prefer controlling the elements at their house remotely using a cell phone. So the system should support wireless communication. Another problem is that WIFI and Bluetooth and other high-speed type of communication cannot be available all the families in general. So the system should use the TCP/IP protocol between the communication of server and the central controller. In addition, the users demand the wireless communication functions of elements of the home automation system because to increase the flexibility of the system by allowing the user to change the position of the various elements of the home automation system. This flexibility is lost in case of wired system because of the constraints due to using wires.

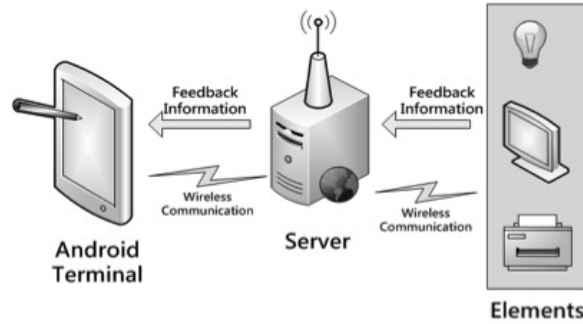


Figure 2: Model of Communication.

### 3.1.6 User data

The users demand the system to storage the data of different IDs. In general users have two aspects of requirements of users data. First the users have their own customs of living. With the data of history of special users, the system can easily recognized the messages the users send, and create a user friendly interface for different users. Also the system can create several classes for many users. With this function, the system can limit some functions for special classes or groups. For example, if the user A and B have been divided into the class Kids, A and B cannot operate some limited elements such as microwaves. In this way the system can also guarantee the safety of the house and its users.

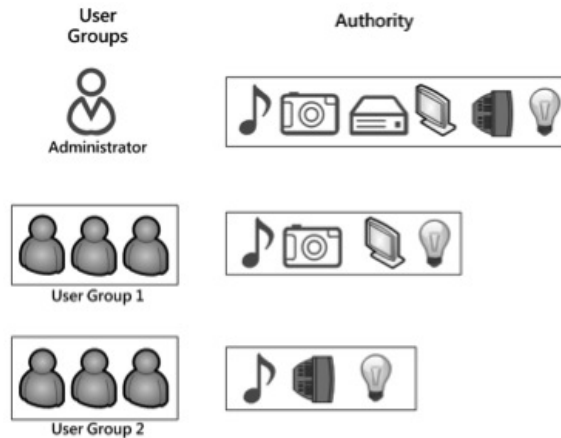


Figure 3: Model of User Group.



### 3.1.7 Check the status of elements

The users should have the authority to check the status of the element. That requires a new window in UI design to display the status of any elements in the system. And A running log is also needed to assist the people taking care of the maintenance of the system to provide the better user experience.

## 3.2 Glossary of Terms

**1. Network service:** a data storage, manipulation, presentation, communication or other shared function provided by one device and consumed by others.

**2. Client (computing):** software that accesses a remote service on another computer.

**3. Machine learning:** a branch of artificial intelligence, concerns the construction and study of systems that can learn from data. For example, a machine learning system could be trained on email messages to learn to distinguish between spam and non-spam messages. After learning, it can then be used to classify new email messages into spam and non-spam folders.

**4. Decision tree learning:** Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value.

**5. Classifier:** An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data to a category.

**6. Thread of execution:** In computer science, a thread of execution is the smallest sequence of programmed instructions that can be managed independently by an operating system scheduler. A thread is a light-weight process. The implementation of threads and processes differs from one operating system to another, but in most cases, a thread is contained inside a process.

**7. Socket:** A network socket is an endpoint of an inter-process communication flow across a computer network. Today, most communication between computers is based on the Internet Protocol; therefore most network sockets are Internet sockets.

**8. Packet:** A network packet is a formatted unit of data carried by a packet-switched network. Computer communications links that do not support packets, such as traditional point-to-point telecommunications links, simply transmit data as a bit stream. When data is formatted into packets, the bandwidth of the communication medium can be better shared among users than if the network were circuit switched.

**9. TCP/IP:** The Internet protocol suite is the networking model and a set of communications protocols used for the Internet and similar networks. It is commonly known as TCP/IP, because its most important protocols, the Transmission Control Protocol (TCP) and the Internet Protocol (IP) were the first networking protocols defined in this standard.

**10. Android software development:** Android software development is the process by which new applications are created for the Android operating system. Applications are usually developed in the Java programming language using the Android Software Development Kit, but other development tools are available.

**11. Android SDK:** The Android software development kit (SDK) includes a comprehensive set of development tools.[6] These includes 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, Windows XP or later; for the moment one can develop Android software on Android itself by using [AIDE - Android IDE - Java, C++] app and [Android java editor] app.

**12. Application programming interface:** An application programming interface (API) specifies how some software components should interact with each other. In addition to accessing databases or computer hardware, such as hard disk drives or video cards, an API can be used to ease the work of programming graphical user interface components. In practice, many times an API comes in form of a library that includes specifications for routines, data structures, object classes, and variables. In some other cases, notably for SOAP and REST services, an API comes as just a specification of remote calls exposed to the API consumers.

**13. Internet protocol suite:** The Internet protocol suite is the networking model and a set of communications protocols used for the Internet and similar networks. It is commonly known as TCP/IP, because its most important protocols, the Transmission Control Protocol (TCP) and the Internet Protocol (IP) were the first networking protocols defined in this standard. It is occasionally known as the DoD model, because the development of the networking model was funded by DARPA, an agency of the United States Department of Defense.

**14. Java class file:** Java class file is a file (with the .class filename extension) containing a Java bytecode which can be executed on the Java Virtual Machine (JVM). Java class file is produced by Java compiler from Java programming language source files (.java files) containing Java classes. If a source file has more than one class, each class is compiled into a separate class file. JVMs are available for many platforms, and the class file compiled in one platform will execute in a JVM of another platform. This makes Java platform-independent.

**15. UI:** The user interface, in the industrial design field of human-machine interaction, is the space where interaction between humans and machines occurs.

**16. User experience:** involves a person's behaviors, attitudes, and emotions about using a particular product, system or service. User experience includes the practical, experiential, affective, meaningful and valuable aspects of human-computer interaction and product ownership. Additionally, it includes a person's perceptions of system aspects such as utility, ease of use and efficiency.

**17. HCI:** human computer interaction

**18. Adobe Photoshop:** Adobe Photoshop is a graphics editing program developed and published by Adobe Systems.

**19. Demo:** Within the computer subculture known as the demo scene, a non-interactive multimedia presentation is called a demo (or demonstration). Demo groups create demos to demonstrate their abilities in programming, music, drawing, and 3D modeling.

**20. HUI:** Handset User Interface.

**21. Customer** The enterprise or department that might want to deploy our system.

**22. Arduino Mega** The Arduino Mega is a microcontroller board based on the ATmega 1280. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

**23. Database** A database is a collection of information that is organized so that it can easily be accessed, managed, and updated. In one view, databases can be classified according to types of content: bibliographic, full-text, numeric, and images.

**24. Delivery** This term can refer (i) to the process of extraction of the package from the box by the receiver or (ii) to the whole process of delivery (from the booking to the actual delivery).

**25. Inter-building delivery:** A delivery process where both the sender and the receiver are in different buildings.

**26. Intra-building delivery:** : A delivery process where both the sender and the receiver are in the same building.(may or may not be in the same floor).

**27. Mobile interface:** A web-based user interface working on Android smart phones, where users can view and execute operations through this application. These operations include: book a delivery, check status of package/mail, personal information management, and system maintenance.

**28. Properties of security:** Refers to the confidentiality, authenticity, integrity and non-repudiation properties of a secure information transfer.

## 4 System Requirements

### 4.1 Enumerated Functional Requirements

In this section, we will extract and analysis the basic and main functional requirements [2] from Customer Statement of Requirements (CSR). By using User Stories [3], we may make the system requirements easier to understand. The table 3.1 is the functional requirements, which are extracted from CSR.

As we know, customers and end users are always interested in the requested functional and costs, which means our groups work should base on the customers requirements, gathering the main requirements from the customers is the first step, however, these requirements cannot be used directly, because customers just present the requirements for us but do not consider if these requirements can be realized or not and do not regard the priorities in their requirements. Hence, we should negotiate with the customers to decide which part is essential, which is desired but not mandatory, which is optional, and which will realize in future.

Basing on the principles mentioned above, we issued the requirements from the CSR. Most of the customers consider the basic system should contain some important functions such as turn on or turn off the lamp, locking or opening the door, and start or stop the music system in their home (ST-3), and they prefer all these functions can be control by voice. This requirement is the basic request of most customers and considering the voice control application are widely used in cellphone market, it can be realized, that why we give it 5 PW. Cellphone application is also a basic and essential requirements for the customers, considering more and more people prefer using cellphone in their daily life, the home automation should also be applied in this field, which is the most convenient and comfortable method to control the home devices. ISO [4] and Android are two types application platforms, considering the market information and development cost, our group decide to use Android application platform [5] to develop our voice based home automation system (ST-4). Besides this two parts, using this application in a remote place is also an important points for the customers, they wish to use this system without the limitation of the distance, therefor, applying the WIFI and Bluetooth and other high-speed type of communication is necessary for us (ST-7). These parts are the main points in the customers requirements. There might be some other points which the customers mentioned, considering the some people prefer using button rather than voice or they may be inconvenient to use voice control in some places, button control is also needed in our design (ST-5), and some others informed us that they wish to know the statues of their home devices and encourage us put into this function in the system (ST-10).

Again, the main points of the system is (ST-3) and (ST-4), to realize this two part is the basic goal of the whole application, at the same time, some related parts may be added into our system, such as voice command device[6] by hu-

ID	PW	User Story
ST-1	4	As a user, I can use the control system by human language and receive the feedback information, which are also human language.
ST-2	1	As a user, I can use my body signal or gestures to turn off or on the lamp or the music system in my house.
ST-3	5	As a user, I can turn on or turn off my home lamp and my music stereo by voice control.
ST-4	5	As a user, I can use this system by my cellphone.
ST-5	3	As a user, I wish to use some buttons to control my house devices.
ST-6	3	As a user, I can get the feedback information, such as whether the system recognizes the correct information and executes the right actions, and I also want to announce and send the messages again or cancel the action by the feedback information. Also the system should notify if the state of some element has been altered manually.
ST-7	4	As a user, I can use this system to control my home devices even I am far away from my home.
ST-8	2	As a user, I can set some limited functions for special people.
ST-9	3	As a user, I can add or eliminate new devices into the system.
ST-10	2	As a user, I can check the statues of my home devices by this system.

Table 7: Functional User Stories.

man languages, remote pattern, button control, statues checking, and feedback information. However, even some points of the requirements are given 3 PW and 2 PW, the costs and limitation of technology make them cannot be realized (ST-9, ST-8). As for the body signal or gestures control, only small group customers required it and the technology is very hard for us, for these reasons we define it as 1 point.

## 4.2 Enumerated Nonfunctional Requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that define specific behavior or functions [7]. According to the FURPS [8] definition of non-functional requirements, we would like to divide our non-functional requirements into 5 parts: Accessibility, Usability, Reliability, Performance and supportability. And all the requirements will be

treated as user stories in this section [9].

**Accessibility** Accessibility is the degree to which a product, device, service, or environment is available to as many people as possible. Accessibility can be viewed as the "ability to access" and benefit from some system or entity. In this case our system should be accessible to different kind of people and different kind of devices. People can use our control system in different kind of devices based on android platform (ST-11).

**Usability** Usability is the ease of use and learnability of a human-made object. Though we make our system based on voice control there are still some concerns about usability. The voice recognition systems usually have troubles with some voice with accent, however a users requires should be accepted and execute. If the system cannot recognize or fully recognize the users command, it should give user some related options based on the command (ST-12).

Also a system based on voice control doesnt mean the system can only controlled by voice commands. Users should be able to control the system by press the buttons or by typing the settings (ST-13). With some touch based control method people can control the system without talking.

**Reliability** Reliability emphasizes dependability in the lifecycle management of a product. Dependability, or reliability, describes the ability of a system or component to function under stated conditions for a specified period of time. People usually live in a house for 5 years or longer [10], so the both the software and the hardware should be able to working a longtime without maintenance (ST-14).

The hardware in the system should be safe enough to control the switch of some electronic equipment. If user wants to cut off the power, the user should be able to turn on and turn off the smart element by hand (ST-15).

Also a user should be able to track all the command has been made and the system should keep tracking users commands (ST-16). The log helps user and the system to understand the users habits better and improve the quality of voice recognition by predicate users commands.

**Performance** Performance is characterized by the amount of useful work accomplished by a system compared to the time and resources used. A user should get feedback in a very short time after he gives the commands (ST-17), just like talking to a real people. The processing time here should be short enough and the system should relay on the server to do the recognition job and complex calculation works.

ID	PW	User Story
ST-11	3	As a user, I should be able to control my system in different kind of devices based on android platform.
ST-12	4	As a user, I should be given some related options based on my commands if my commands are not recognized correctly.
ST-13	4	As a user I should be able to control the system with my android device based on touch screen or input but not only with voice commands.
ST-14	3	As a user, I should be able to use both the software on the android and the hardware for a longtime without maintenance.
ST-15	2	As a user I should be able to track all the commands I have sent.
ST-16	3	As a user, I should be able to turn on and turn off the smart element in my home with my hands.
ST-17	4	As a user, I should receive some feedbacks after I send a voice commands, whether the commands has been recognized correctly or not.
ST-18	3	As a user, I should be able to find some documentation about the home automation system online and fix some little problems with the help of the documentation.
ST-19	2	As a user I should be able to update my software on the android if necessary .

Table 8: Non-functional User Stories.

**Supportability** Supportability refers to the ability of technical support personnel to install, configure, and monitor computer products, identify exceptions or faults, debug or isolate faults to root cause analysis, and provide hardware or software maintenance in pursuit of solving a problem and restoring the product into service. Incorporating serviceability facilitating features typically results in more efficient product maintenance and reduces operational costs and maintains business continuity. (ST-18) suggests that there should be a set of documentation of the system so that the system can be maintained with some documentation. Also (ST-19) suggests that there should be a software update to keep the software working in the best condition.

### 4.3 On-Screen Appearance Requirements

Our program is home automation based on voice, that means users can control their home more conveniently, what they need to do is just having a conversation with their mobile phones. This UI design [11] is based on Android system [12].

UI [13] is important, why we say that? As we all know, most users of the prod-



uct dont have the enough knowledge of electronic and software, what they can know about their electronic product always via UI. What they can get about the information of product is through clicking the button or rolling along the screen. Thus, our destination of UI design is making a kind of clear and neat interface, which can help us attract more people to generate interest in our product. To be clear can make sure that users can understand the functions of the device [14] and can easily learn how to use it, and guarantee that they wont be confused by what they see. To be neat is not equal to to be simple, it means that our interface should make users feel comfortable, we try our best to make them achieve their purpose with the least steps. At last, we should make our interface looks harmonious and attractive [15].

We cannot promise that all users are satisfied with our UI, in fact, no designer is able to do it, but if over 80

ID	PW	Requirement
ST-20	5	The interface shall make users be able to input their voice through convenient, clear and easy way.
ST-21	4	The interface shall give users feedback via image or word. The interface should inform users that their voice is received when it is recording.
ST-22	3	The interface shall offer additional button which users can control the system by clicking button in some certain situation.
ST-23	3	The interface shall show users the status of their devices by icon or buttons.
ST-24	2	The interface should support new devices to guarantee that users do not need to abandon their interface when new devices are added.
ST-25	2	The interface shall jump to right desk when it receives certain information and signal.
ST-26	1	The interface should own appropriate overall arrangement and color which can make users feel comfortable when they use it.

Table 9: On-Screen Appearance Requirements .

The pictures below is the preliminary drafts of the interfaces of our system which will be polished and embellished in future, it shows the basic function of our program. Overall, it has 4 separate interfaces: log in interface which is created to guarantee the security and privacy of users [16], main interface through which users can input command, music interface and light interface [17] through which users can check the status of music and light.

## 5 Functional Requirements Specification

### 5.1 Stakeholders

In general, the stakeholders of our system can be divided into two different parts: the internal parts and the external parts.

#### 5.1.1 Internal parts

**Developers:** the developers of our system are a person or a group people who code and design the processes of the whole program, our developers have the right to access all section of the system and interest to realize the function for the system, which required by the end users and customers.

#### 5.1.2 External parts

**End users:** end users may a person or a group who interest to use and investigate our system.

**Customers:** homeowners or a group people who share a home are our main customers, they always interest the reliability, value, function, cost performance, and quality of our system.

**Investigator:** investigator may be a person or a group people who are interested in market, customers, and the user and customer feedback.

### 5.2 Actors and Goals

Our systems actors and goals are indicated in the table 5.

### 5.3 Use Cases

#### 5.3.1 Casual Description

**UC-1 Light controlling** Allow the users to turn on or off the lights by sending the voice message to the server from the Android based cellphone.

**UC-2 Music controlling** Allow the users to control the music player by their voice. And the server sends the control instruction to the MCU after processing the voice message.

**UC-3 Modify the brightness** The users can modify the brightness of the light by their voice command. Depending on the voice command, the system will increase or decrease the brightness of the light.

<b>Actor</b>	<b>Category</b>	<b>Goals</b>
Users	Initiating	To turn on or turn off the light and music (by voice), to control the light intensity and music volume (by voice).
Device System (Collector)	Initiating and Participating	To collect and compile the information from the users and send the code information to the central sever.
Device System (Server)	Initiating	To analysis the information from the device collector, sending the feedback to the user interface and signal to the arduino.
Arduino	Initiating and Participating	To collect the signal from the sever and control the light and music switches.
Light Switch	Participating	To turn on or turn off the light and control light intensity.
Music Switch	Participating	To turn on or turn off the music and control music volume.
Communication Network	Participating	To transmit signal by wifi and internet.
Administrator	Initiating	To manage the administrators own authorities and enable administrator increase or decrease their authorities in the management interface.

Table 10: Actors and Goals

**UC-4 Modify the volume of music** It allows the users to increase or decrease the volume of music player by sending the related voice message to the music system.

**UC-5 Elements status check** The system allows the users to check the current status of elements in system. It is can either be called upon by the user or if the state of the device is changed manually. The elements can send the status information to the server and the server process this information. The elements can send the status information to the system and the system can relay the status to the user by displaying it on the user interface.

**UC-6 User login** It allows the authorized users to access into the system.

**UC-7 Create new users accounts** The system allows the new users to create their own accounts. By following a few easy steps given in the user interface.

**UC-8 Remote user** It allows the administrator to remote the deserted users. The deserted usernames cannot be used again and their data in server will be deleted.

**UC-9 User log out** The system allows the administrator to manage the authorities of different users. Based on the requirements, in our system the different users have different authorities to access the elements. And the system administrator can increase or decrease their authorities in the management interface.

**UC-10 Authority management** The system allows the administrator to manage the authorities of different users. Based on the requirements, in our system the different users have different authorities to access the elements. And the system administrator can increase or decrease their authorities in the management interface.

**UC-11 Check the authority of user** Administrator can check the authorities of all users in order to management. And the users can check their own permissions to know whether they can do the operations.

**UC-12 Control information feedback** It allows the users to check the feedback information of voice message in order to check whether the voice have been identified by the server. If the message identification failed, the feedback information can lead the user to send again or cancel the operation.

**UC-13 Network error notification** The system offers the notification of network error if the signal of network is weak. The signal can be separate into 2 parts: Between cellphone and server; between server and MCU. In such an event, the system will notify the user using the interface device.

### 5.3.2 Use Case Diagram

All the use case diagrams are showed as follows:

User Case 1 Light Controlling
<p><b>Related Requirements:</b> ST-3,ST-1,ST-2,ST4,ST20</p> <p><b>Initiating Actor:</b> User</p> <p><b>Actors Goal:</b> To turn on or off the light</p> <p><b>Participating Actor:</b> Light and Arduion</p> <p><b>Precondition:</b> User is an authorized person</p> <p><b>Success End Condition:</b> The light is turn on or off by the users instruction</p> <p><b>Failed End Condition:</b> Server has not identified the message of user</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>← 1. Cellphone displays the main menu of application.</p> <p>→ 2. User selects the light function.</p> <p>← 3. Cellphone displays the light function interface.</p> <p>→ 4. Player sends their voice message to the server.</p> <p>← 5. Server processes the message and sends a signal to the Arduino.</p> <p>← 6. The Arduino receives the signal from server and communicates it to the MCU and invokes UC5(status check).</p> <p>← 7. The MCU receives the signal from the Arduino and turns on or off the switch.</p>

Table 11: User Case 1 Light Controlling .

<b>User Case 2 Music Controlling</b>
<b>Related Requirements:</b> ST-3,ST-1,ST-2,ST4,ST20 <b>Initiating Actor:</b> User <b>Actors Goal:</b> To turn on or off the music player <b>Participating Actor:</b> Music player and Arduion <b>Precondition:</b> User is an authorized person <b>Success End Condition:</b> The music player is turn on or off by the users instruction <b>Failed End Condition:</b> The message of user have not been identified by server <b>Extension Point:</b> <b>Flow of Events for Main Success Scenario:</b> ← 1. Cellphone displays the main menu of application. → 2. User selects the music function. ← 3. Cellphone displays the music function interface. → 4. Player sends their voice message to the server. ← 5. Server processes the message and sends a signal to the Arduino. ← 6. The Arduino receives the signal from server and communicates it to the MCU and invokes UC5(status check).. ← 7. The MCU receives the signal from the Arduino and turns on or off the switch.

Table 12: User Case 2 Music Controlling .

<b>User Case 3 Modify brightness</b>
<b>Related Requirements:</b> ST-3,ST-1,ST-2,ST4,ST20 <b>Initiating Actor:</b> User <b>Actors Goal:</b> To modify the brightness of the light <b>Participating Actor:</b> Light and Arduion <b>Precondition:</b> User is an authorized person <b>Success End Condition:</b> The brightness of the light has been modified follow the users instruction <b>Failed End Condition:</b> Server has not identified the messages of user <b>Extension Point:</b> <b>Flow of Events for Main Success Scenario:</b> ← 1. Cellphone displays the main menu of application. → 2. User selects the light function. ← 3. Cellphone displays the light function interface. → 4. Player sends their voice message to the server. ← 5. Server processes the message and sends a signal to the Arduino. ← 6. The Arduino receives the signal from server and communicates it to the MCU and invokes UC5(status check). ← 7. The MCU receives the signal from the Arduino and modifies the brightness.

Table 13: User Case 3 Modify brightness.

User Case 4 Modify the Volume of Music
<p><b>Related Requirements:</b> ST-3, ST-1,ST-2,ST4,ST20</p> <p><b>Initiating Actor:</b> User</p> <p><b>Actors Goal:</b> To modify the volume of the music player</p> <p><b>Participating Actor:</b> Music player and Arduion</p> <p><b>Precondition:</b> User is an authorized person</p> <p><b>Success End Condition:</b> The volume of music player has been modified follow the users instruction</p> <p><b>Failed End Condition:</b> Server has not identified the messages of user</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>← 1. Cellphone displays the main menu of application.</p> <p>→ 2. User selects the music function.</p> <p>← 3. Cellphone displays the music function interface.</p> <p>→ 4. Player sends their voice message to the server.</p> <p>← 5. Server processes the message and sends a signal to the Arduino.</p> <p>← 6. The Arduino receives the signal from server and communicates it to the MCU and invokes UC5 (status check).</p> <p>← 7. The MCU receives the signal from the Arduino and modifies the volume.</p>

Table 14: User Case 4 Modify the Volume of Music.

User Case 5 Elements Status Check
<p><b>Related Requirements:</b> ST-6, ST-10, ST-24</p> <p><b>Initiating Actor:</b> User or Arduion</p> <p><b>Actors Goal:</b> To check the current status of smart elements in system</p> <p><b>Participating Actor:</b> Light, Music player, Server and micro-controller</p> <p><b>Precondition:</b> a. User is an authorized person. b. Communication between the server and smart elements is connected</p> <p><b>Success End Condition:</b> The status of elements display on the interface of cellphone</p> <p><b>Failed End Condition:</b> The status information conveys failure between server and smart elements</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>← 1. Cellphone displays the main menu of application.</p> <p>→ 2. User selects the status check function.</p> <p>← 3. The elements send a status message to the MCU and MCU sends the information to the Arduino.</p> <p>← 4. Arduino sends message to the server and server sends to the cellphone after processing.</p> <p>← 5. Cellphone receives the message and displays the information of status on UI.</p>

Table 15: User Case 5 Elements Status Check.

<b>User Case 6 User Login</b>
<p><b>Related Requirements:</b> ST-8</p> <p><b>Initiating Actor:</b> User, Administrator</p> <p><b>Actors Goal:</b> To login the system by the username and password</p> <p><b>Participating Actor:</b> Server</p> <p><b>Precondition:</b> Users account has been created in the server</p> <p><b>Success End Condition:</b> Users account has been created in the server</p> <p><b>Failed End Condition:</b> The user enters the incorrect username or the password is not corresponding with the username</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>← 1. Cellphone displays the login interface.</p> <p>→ 2. User enters the username and password.</p> <p>← 3. Cellphone send the username and password to the server.</p> <p>← 4. The server checks the username and password in the database and sends the feedback information to the cellphone.</p> <p>← 5. The cellphone display the main menu to the user.</p>

Table 16: User Case 6 User Login.

<b>User Case 7 Create New User Accounts</b>
<p><b>Related Requirements:</b> ST-8</p> <p><b>Initiating Actor:</b> User</p> <p><b>Actors Goal:</b> To create the new accounts for the new users and store in server</p> <p><b>Participating Actor:</b> Server</p> <p><b>Precondition:</b> Username of new account has not been occupied and the number of users of system less than the maximum users the server can hold</p> <p><b>Success End Condition:</b> The user account has been created and the information has been stored in the server</p> <p><b>Failed End Condition:</b> The username has been occupied or the password is out of requirement</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>Include: Login (UC-6)</p> <p>← 1. Cellphone displays the creating new account interface.</p> <p>→ 2. User enters the personal information.</p> <p>← 3. Cellphone identifies whether the username and password obey the rules.</p> <p>← 4. Cellphone send the information to server and the server stores the information about the new accounts.</p> <p>← 5. Server sends the feedback information to the cellphone and the cellphone display the login interface to the user.</p>

Table 17: User Case 7 Create New User Accounts.



<b>User Case 8 Remove Users</b>
<b>Related Requirements:</b> ST-8 <b>Initiating Actor:</b> Administrator <b>Actors Goal:</b> TTo remove a deserted or illegal user <b>Participating Actor:</b> Server <b>Precondition:</b> a. The information of deserted or illegal user is stored in server. b. Current user must be administrator. <b>Success End Condition:</b> Administrator deletes the information of users in server <b>Failed End Condition:</b> The users information cannot be found in server <b>Extension Point:</b> <b>Flow of Events for Main Success Scenario:</b> Include: Login (UC-6) ← 1. Cellphone displays the user management interface. → 2. Administrator selects the user that will be removed. ← 3. Cellphone send the remove instruction to server. ← 4. Server deleted the related information of user. ← 5. Server sends the feedback information to the cellphone and the cellphone display the current users list.

Table 18: User Case 8 Remove Users.

<b>User Case 9 User Log Out</b>
<b>Related Requirements:</b> ST-8 <b>Initiating Actor:</b> User, Administrator <b>Actors Goal:</b> To log out from the system and protect personal information <b>Participating Actor:</b> Server <b>Precondition:</b> User has already logged into the system <b>Success End Condition:</b> The user has logged out of the system and no operations can be done <b>Failed End Condition:</b> None <b>Extension Point:</b> <b>Flow of Events for Main Success Scenario:</b> Include: Login (UC-6) ← 1. Cellphone displays the main menu of application. → 2. User clicks the log out button. ← 3. Cellphone clear the information of current user and display the login interface.

Table 19: User Case 9 User Log Out.

User Case 10 Authority Management
<p><b>Related Requirements:</b> ST-8</p> <p><b>Initiating Actor:</b> Administrator</p> <p><b>Actors Goal:</b> To manage the authorities of different users</p> <p><b>Participating Actor:</b> Server</p> <p><b>Precondition:</b> a. Current user must be the administrator. b. There are 1 user at least in the database.</p> <p><b>Success End Condition:</b> The authorities of users have been modified after management</p> <p><b>Failed End Condition:</b> The authorities of users have not be changed</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>Include: Login (UC-6)</p> <p>← 1. Cellphone displays the management interface of administrator.</p> <p>→ 2. Administrator selects the users need to be modified.</p> <p>← 3. Cellphone display the current authorities of this user.</p> <p>→ 4. Administrator modifies the authorities of this user.</p> <p>← 5. Cellphone send the modified information to the server and the server modifies the information in database.</p> <p>← 5. Server sends the feedback information to the cellphone and displays the current authorities of user on interface.</p>

Table 20: User Case 10 Authority Managemen.

User Case 11 Check the Users Authority
<p><b>Related Requirements:</b> ST-8</p> <p><b>Initiating Actor:</b> User</p> <p><b>Actors Goal:</b> To check the current users authority</p> <p><b>Participating Actor:</b> Server</p> <p><b>Precondition:</b> a. Authorized user have logged into the system. b. Network between server and cellphone should be connected</p> <p><b>Success End Condition:</b> The users authority displays on interface</p> <p><b>Failed End Condition:</b> The communication between server and cellphone has been interrupted</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>Include: Login (UC-6)</p> <p>→ 1. User selects the authority function.</p> <p>← 2. Cellphone sends the check instruction to the server.</p> <p>← 3. Server checks the database and sends the information of authority back to the cellphone.</p> <p>← 4. Cellphone displays the authority on the interface.</p>

Table 21: User Case 11 Check the Users Authority.

User Case 12 Control Information Feedback
<p><b>Related Requirements:</b> ST-1, ST-12, ST17, ST21</p> <p><b>Initiating Actor:</b> Server</p> <p><b>Actors Goal:</b> Send the feedback information to the user interface</p> <p><b>Participating Actor:</b> User, cellphone</p> <p><b>Precondition:</b> Network between server and other parts of system should be connected</p> <p><b>Success End Condition:</b> The feedback information sends to the cellphone and displays on the screen</p> <p><b>Failed End Condition:</b> The communication between server and other parts has been interrupted</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>→ 1. The server identifies the voice message after user sending the voice messages.</p> <p>← 2. The server processes the feedback information and sends back to the cellphone.</p> <p>← 3. Cellphone receives the feedback and displays on the interface.</p>

Table 22: User Case 12 Control Information Feedback.

User Case 13 Network Error Notification
<p><b>Related Requirements:</b> ST-8, ST17</p> <p><b>Initiating Actor:</b> Cellphone</p> <p><b>Actors Goal:</b> To give a notification of network error to the user</p> <p><b>Participating Actor:</b> User</p> <p><b>Precondition:</b> The communication between cellphone and other parts of system has been interrupted</p> <p><b>Success End Condition:</b> The network error notification has been displayed on the cellphone</p> <p><b>Failed End Condition:</b> The network has no error currently</p> <p><b>Extension Point:</b></p> <p><b>Flow of Events for Main Success Scenario:</b></p> <p>Include: Login (UC-6)</p> <p>← 1. Cellphone checks the current status of network.</p> <p>← 2. If the network status is abnormal, the cellphone displays an error notification on the screen.</p>

Table 23: User Case 13 Network Error Notification.

### 5.3.3 Traceability Matrix

The Traceability Matrix is showed in the Table 19 in next page.

### 5.3.4 Fully-Dressed Description

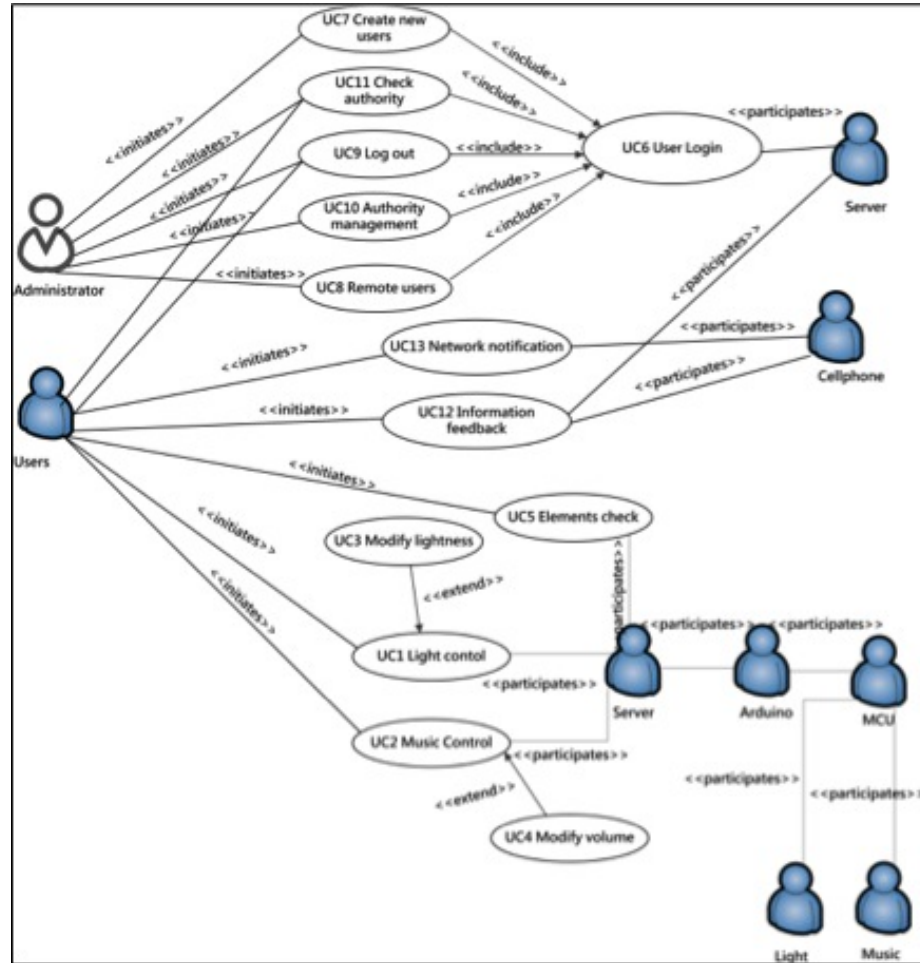


Figure 4: Use Case Diagram.

REQ	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13
ST1	4	x	x	x	x								x	x
ST2	1	x	x	x	x									
ST3	5	x	x	x	x									
ST4	5	x	x	x	x									
ST6	3					x								
ST8	2						x	x	x	x				
ST10	2					x								
ST12	4												x	
ST17	4												x	x
ST20	5	x	x	x	x								x	x
ST21	4												x	
ST24	2													
<b>Max</b>	<b>PW</b>	5	5	5	5	3	2	2	2	2	2	2	4	4
<b>Total</b>	<b>PW</b>	20	20	20	20	5	2	2	2	2	2	2	16	10

Table 24: Traceability Matrix

## 5.4 System Sequence Diagrams

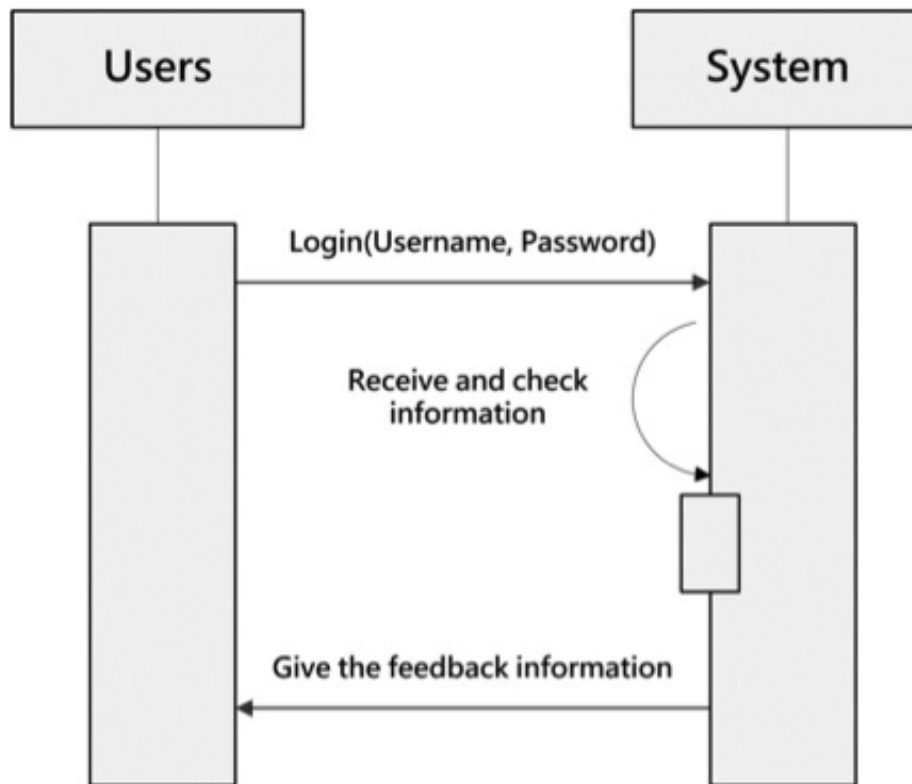


Figure 5: Login sequence diagram UC-6.

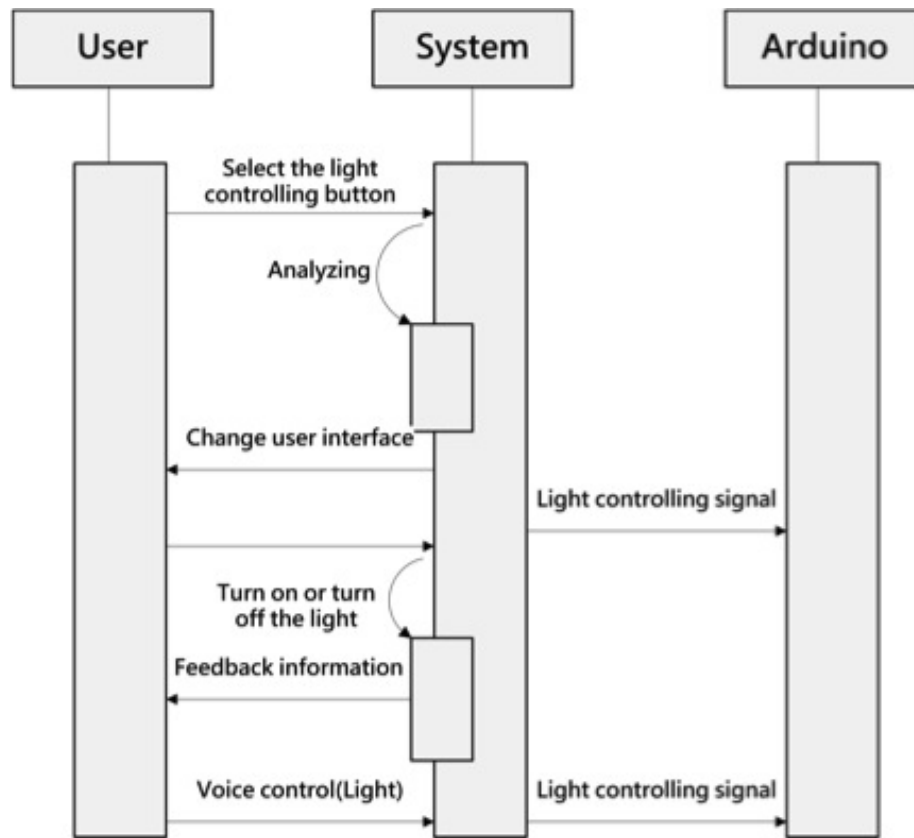


Figure 6: Light controlling sequence diagram UC-1.

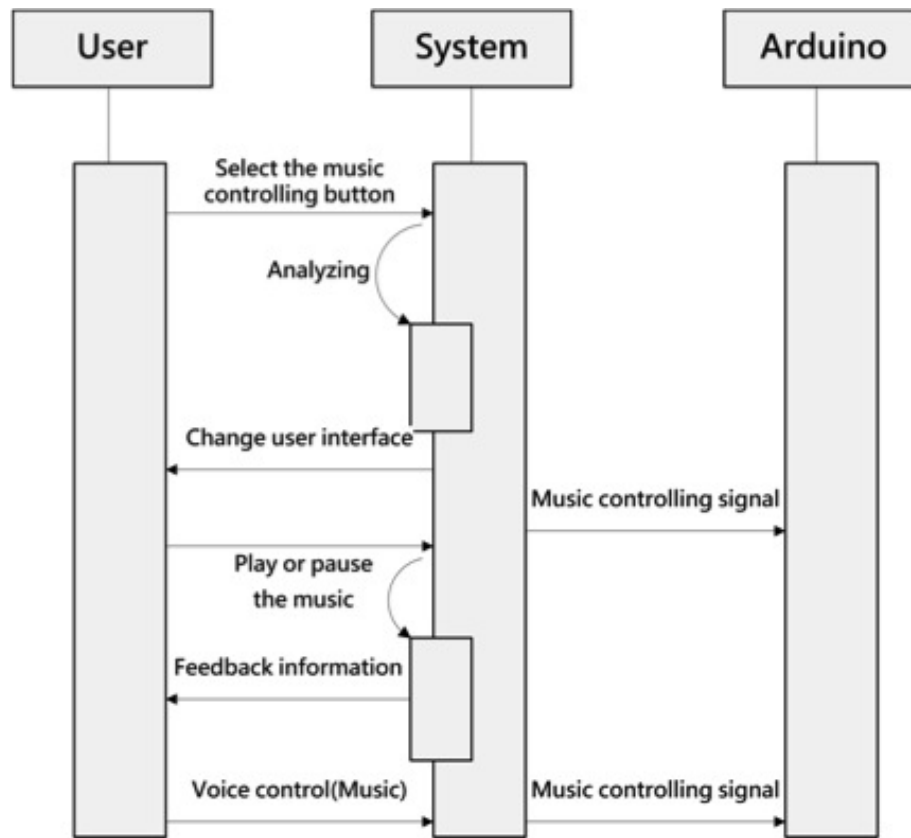


Figure 7: Music controlling sequence diagram UC-2.



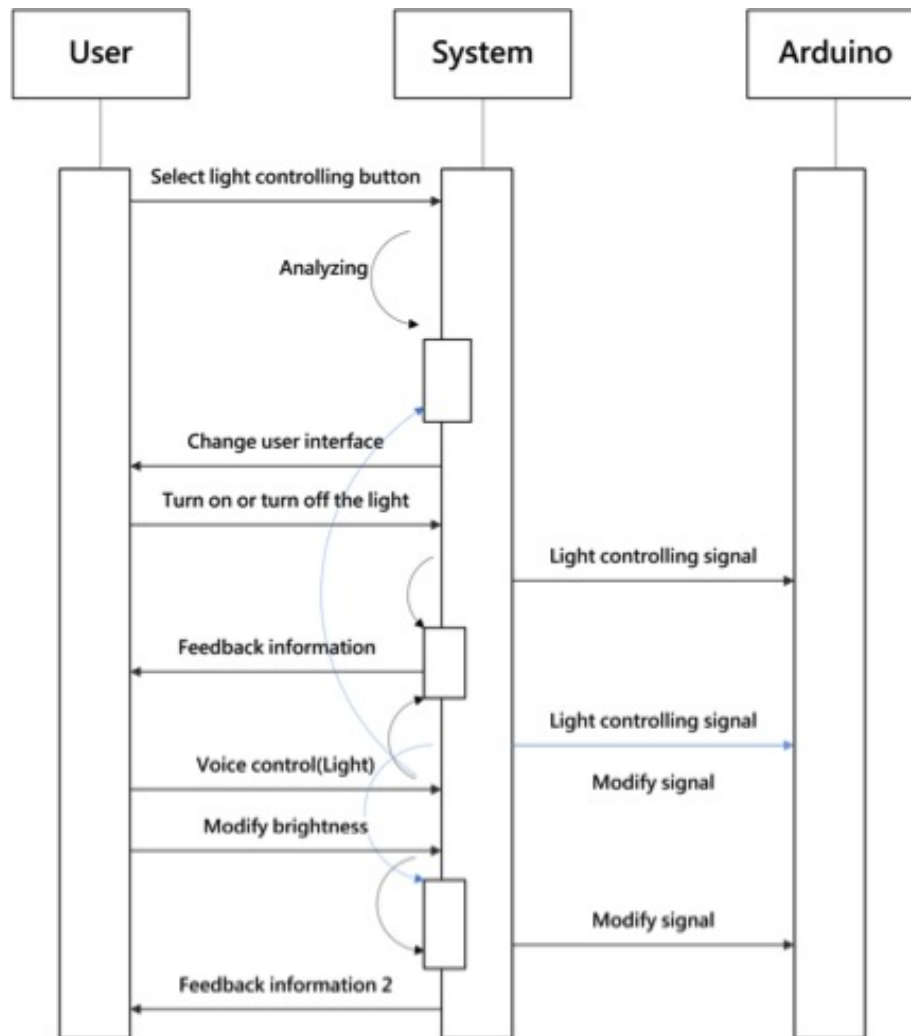


Figure 8: Modify the volume of music sequence diagram UC-4.

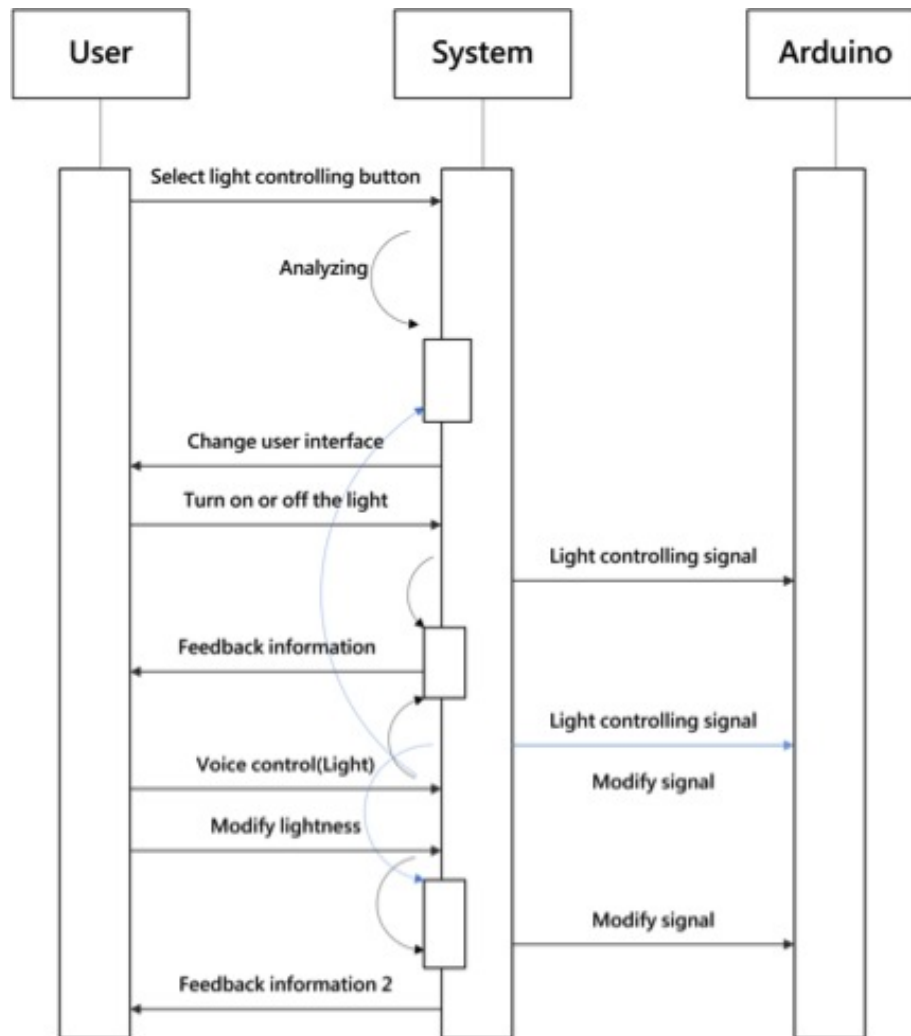


Figure 9: Modify the brightness sequence diagram UC-3.

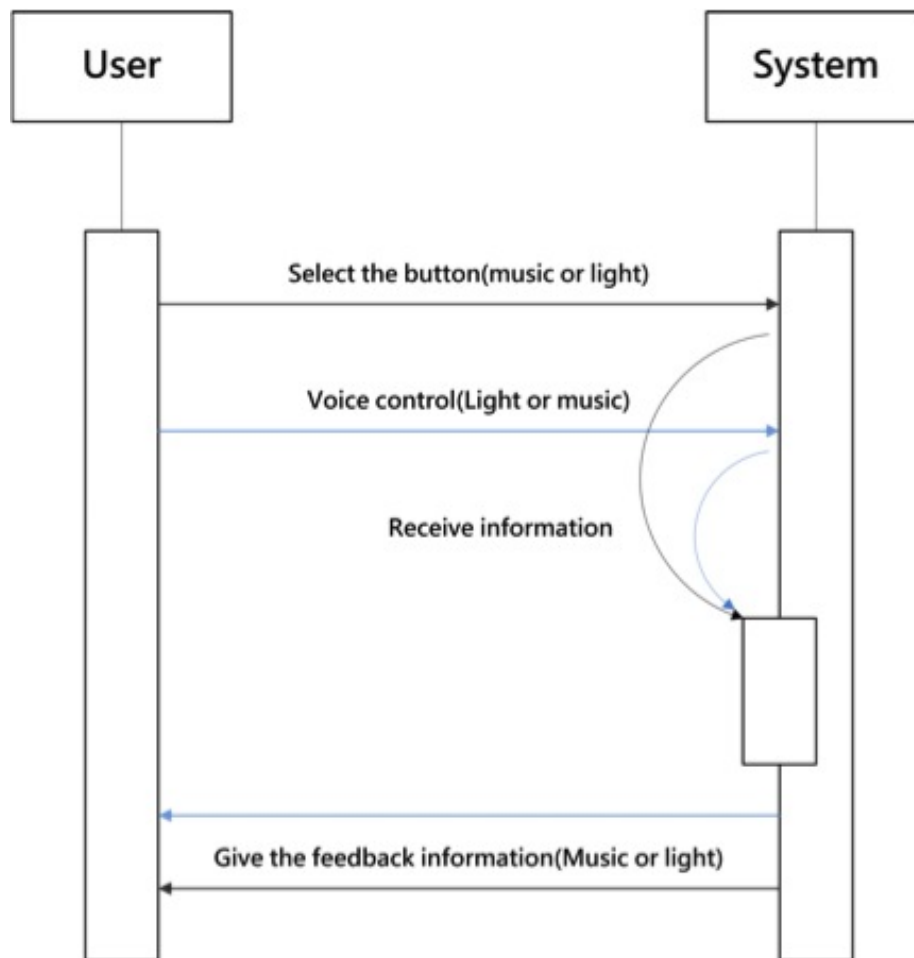


Figure 10: Create new users accounts sequence diagram UC-7.

The figures show some important use case system sequence diagram. In each diagram, we provide two methods to help user to control the system, the black arrow means the user completes the processes by button and the blue arrow means the users completes the processes by voice control.

## **6 User Interface Specification**

### **6.1 Preliminary Design**

From this section, we will give you a detail explanation about our interface design, and show you step by step that how users can run their house with minimized approach. We do effort to make our interface be friendly and clear.

Users can try input passport at most 3 times. If they type in wrong number three times succession, the screen would locked for about 5 minutes which is designed in order to protect users privacy.

#### **6.1.1 Log In**

When users enter our system, they are prompt with log in interface (shown in figure 11), which can guarantee their security, authorization and privacy. What the users need to do is inputting their password through the number keyboard showing on the desk. If they key in a wrong number accidentally, instead of keying in the entire password, they can modify it by clicking on the button at bottom-right to delete only the wrong one, which can minimize their effort effectively. Then users can tab the blue circle button which with an arrow on it to log in. When they fill in the right password, they will come to the main interface, otherwise, the interface will show users that the password is incorrectly, thus they can try it again (shown in figure 12). However, if they input wrong numbers more than 3 times, the system will be locked automatically in case of malevolent act of intruding.

The image shows a log-in interface on a dark background. At the top center is a circular icon containing a white silhouette of a person's head and shoulders. Below this icon are two input fields. The first field is labeled "user name:" and contains three asterisks "\*\*\*". The second field is labeled "password:" and contains seven asterisks "\*\*\*\*\*". To the right of these fields is a blue circular button with a white right-pointing arrow. Below the input fields is a numeric keypad consisting of a 4x3 grid of blue buttons with white text. The buttons are arranged as follows:

1	2	3
4	5	6
7	8	9
exit	0	<

Figure 11: Log in interface; users can enter the main interface after keying in correct passport.

The image shows a user login interface. At the top, there is a circular icon of a person's head and shoulders. Below this, there are two input fields: "user name:" followed by three asterisks (\*\*\*) and "password:" followed by seven asterisks (\*\*\*\*\*). To the right of these fields is a blue circular button with a white right-pointing arrow. Below the input fields, a red error message reads "incorrect input, please try again...". At the bottom of the interface is a numeric keypad with a 4x3 grid of buttons. The buttons are arranged as follows:

1	2	3
4	5	6
7	8	9
exit	0	<

Figure 12: users cannot enter our system without authorization.

### 6.1.2 Main Interface

With the authorization, the user is presented with main interface (shown in figure 9-3), it means user can start control their house by voice now. On the top of the screen, he or she can see the detailed status of information of his/her specific devices. Knowing the status of devices, users can make command better and more efficiently. For example, if the status of light is on, they do not need to turn on it again. In other words, users can make better judgment when they know more.

Apart from the list on top, users can see that the main interface includes other 4 parts:

**Microphone** Microphone is the essential and indispensable part to a voice control device, when users want to output command, they just need to click on this button, then the circle background of microphone change its color from red to gray (shown in figure9-4). Meanwhile, the phone will start to record what they say, and then send it to Arduino over Wi-Fi. Users can be informed that their phones with that change are receiving their voice. When they finish the command, they need to click the Microphones button again, the circle background is back to red. Finally, it will receive the feedback and transfer it to users via voice system. (The rest three buttons are just needed when users cannot control by voice in some special situation. Otherwise, users can approach to their destination via voice. To sum up, the buttons shown on interface provide users with more options so that they can choose the way they prefer.)

**Music** One click on this button, they will come into the music interface, and then they can do more detailed things.

**Light** Through click on this button, the system would redirect users into the light interface.

**Exit** This button simply exits from system.

Our target is to make main interface as simple as possible in order to make users would not be bothered with too many useless option and messy buttons. Meanwhile, our interface is designed to satisfy the most principal functions.

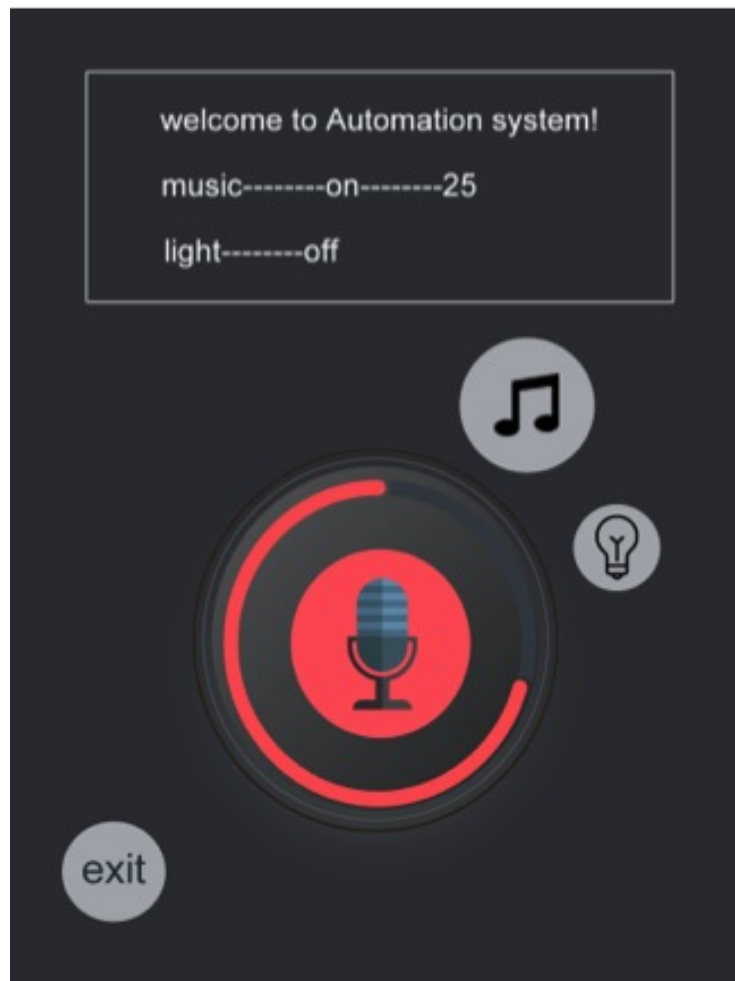


Figure 13: Main interface which users can click the Microphone button to make command.



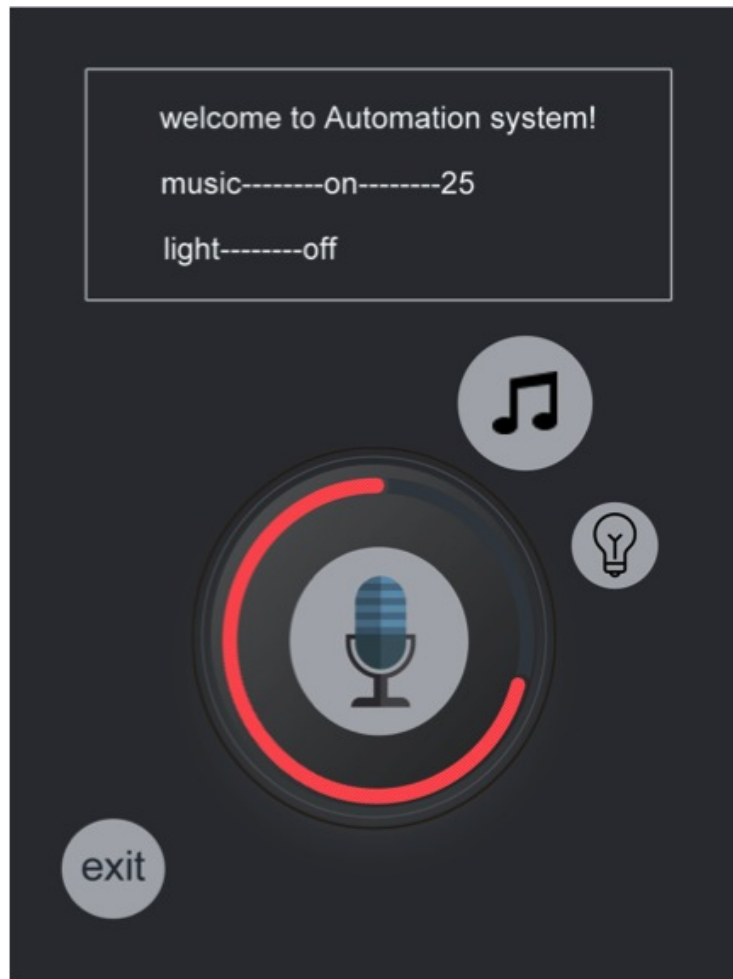


Figure 14: Main interface when record users voice.

### 6.1.3 Music Interface

When the mobile phone receive the command about the music, it will jumps into music interface. In this interface, users can command the music and check the status of music more clearly and more detailed. Users can see a button line, it includes play(shown in figure 9-5), pause(shown in figure 9-6), and volume (shown in figure 9-7 and figure 9-8) button, and the number shown under the button line is the degree of volume. If users want to change the status of music, they can click on the microphone button and speak, and then click the button again after finishing their order. If they want to go back to main interface, they can say back to main interface or just click the back button; they can choose what you want.

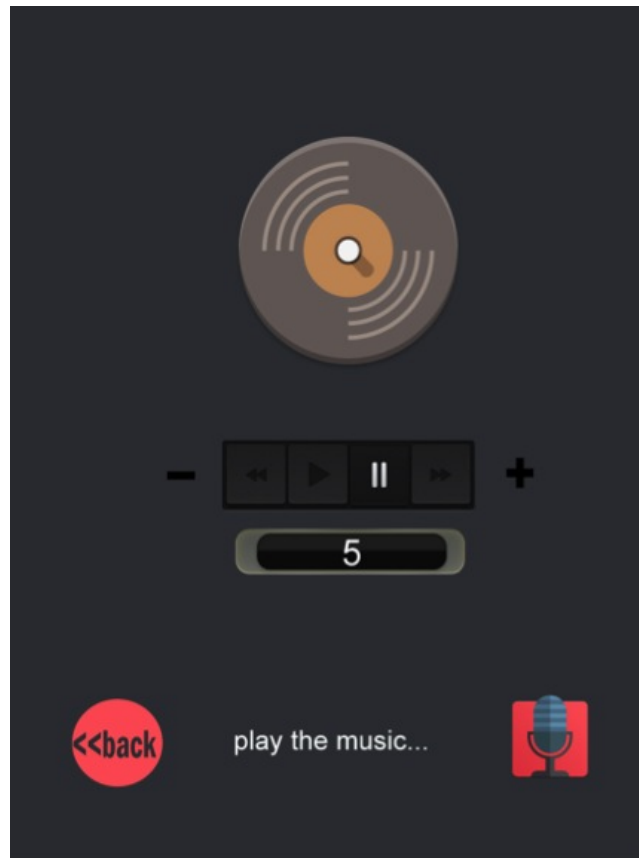


Figure 15: Music interface play the music.

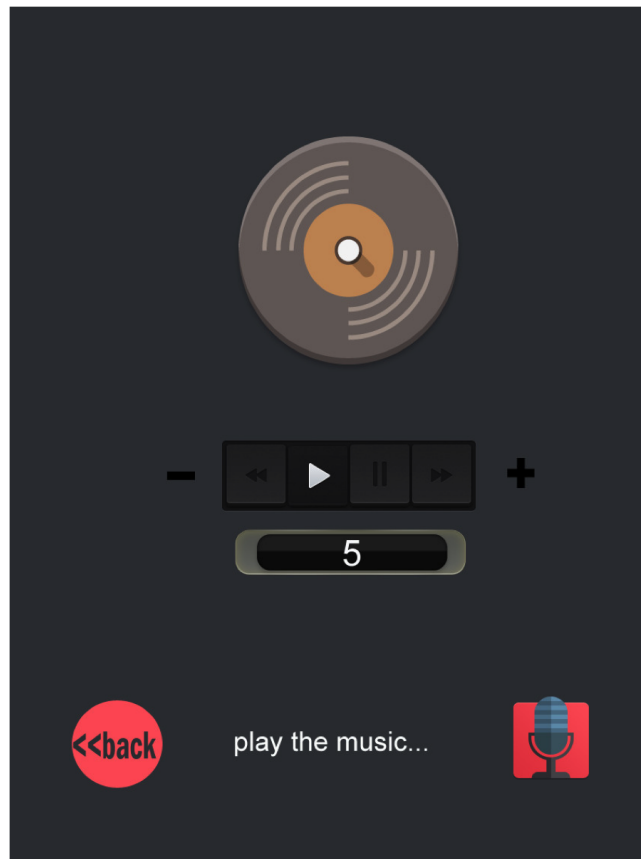


Figure 16: Music interface turn off the music.

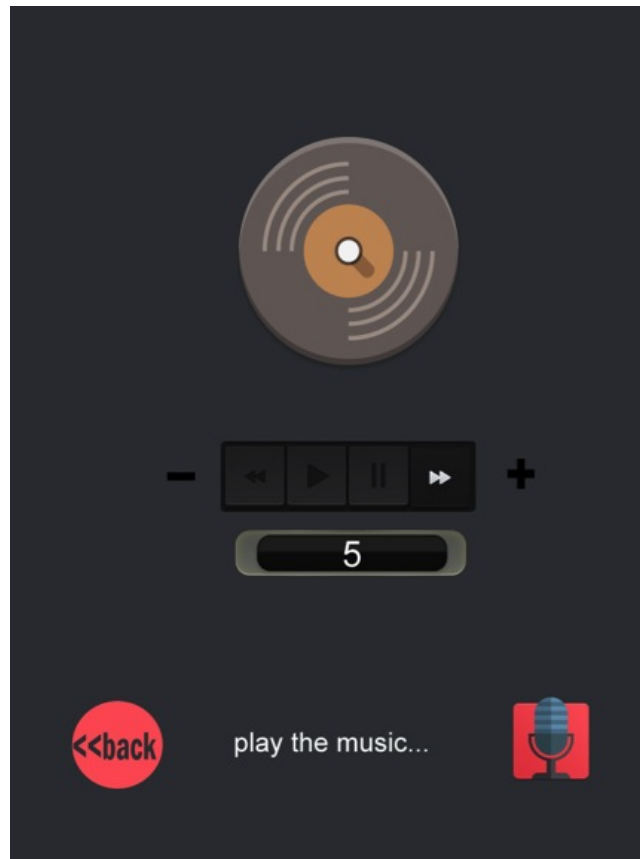


Figure 17: Turn up the music.

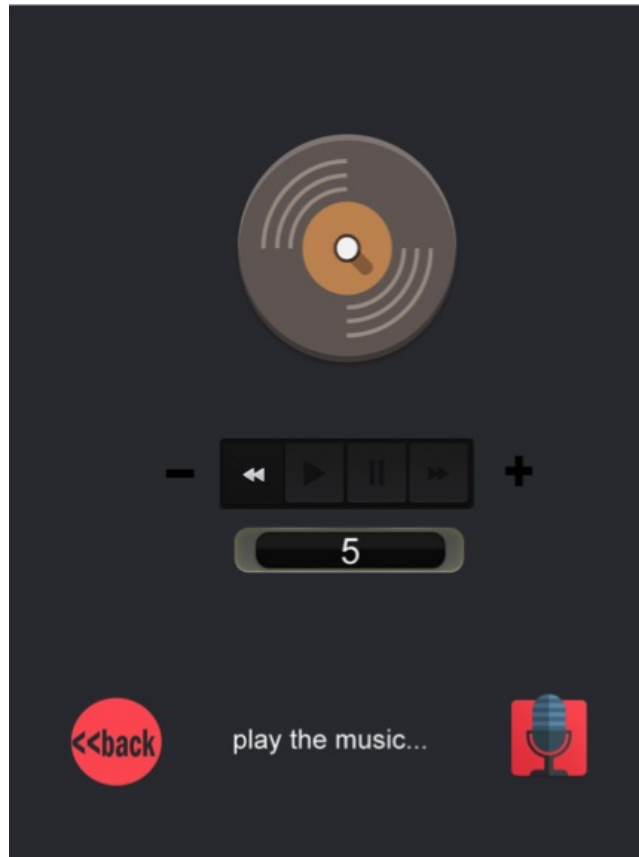


Figure 18: Turn down the music.

#### 6.1.4 Light interface

After getting the signal about light, users are prompted with light interface. The picture of light can basically show whether the light is on or not, users can know the status of their lights simply through the dynamic picture or the button at the bottom of the screen. User can also control the brightness of the light through the button in the middle of screen, the number near the button can show the user the intensity of their light, we consider that the number can offer the user more direct and clear feelings which make them control their light more appropriately. Again, when the user wants to back to main interface, they can click the back button or back via voice.

Finally, as we can see from the whole interfaces, which are shown above, what user needs to do is very casual and simple. To log in, user needs type their

passport and click one button. To input their command, user needs to click the Microphone button two times. Generally, after these two steps, users can control their devices by voice all the time. As a result, our interface design can basically satisfy our requirement demand.

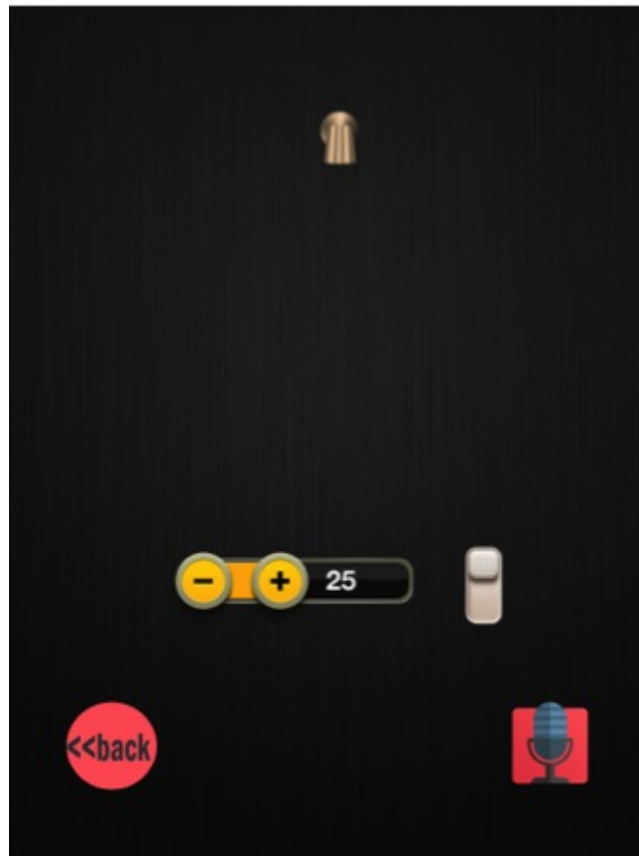


Figure 19: Turn up the music.

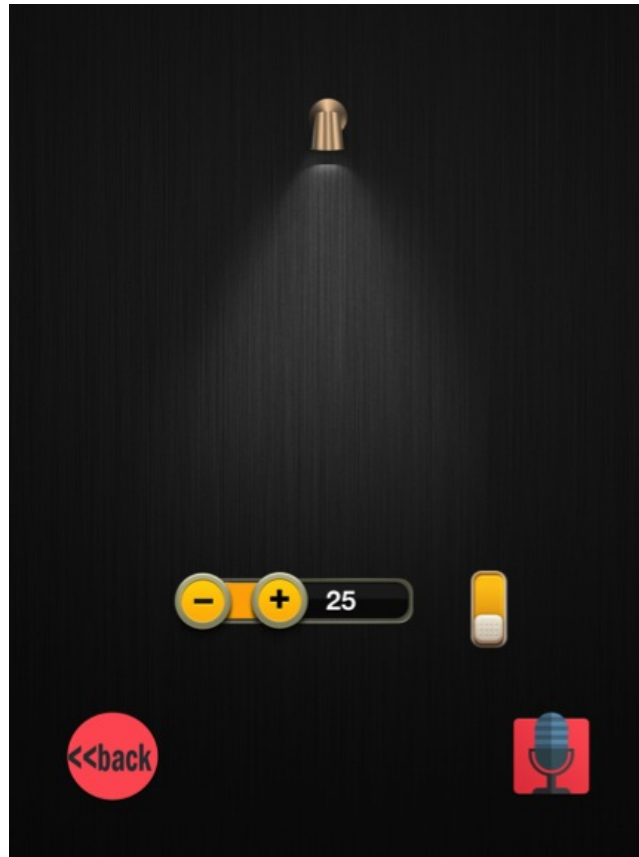


Figure 20: Turn down the music.

## 6.2 User Effort Estimation

### 6.2.1 User login

Respective user cases: UC-6

Navigation and Data Entry: 2 clicks and 10 20 keystrokes

1 Click auto-home to enter into the system.

2 Enter the username.

3 Enter the password.

4 Click the login button to check the username and password. If the word is correct, the user enable log in.

5 If the password is incorrect, user can enter the password again.

6 If the password is incorrect, user can click on return button to exit the system.

### **6.2.2 User logout**

Respective user cases: UC-9

Navigation and Data Entry: 1 2 clicks

Prerequisite: 1. User login (3 clicks and 8 keystrokes)

- 1 Click exit button to return to the login interface.
- 2 Click the return button to the cellphone main interface.
- 3 If the users do not want to return the login interface, click on return button to logout directly.

### **6.2.3 Light controlling**

Respective user cases: UC-1

Navigation and Data Entry: 1 3 clicks

Prerequisite: 1. User login (3 clicks and 8 keystrokes)

- 1 Click the light figure button to enter the light-controlling interface.
- 2 Push down the light button to turn on the light.
- 3 Push on the light button to turn off the light.
- 4 User enables use voice to control the light overall process click on the click on the microphone graphic and then use the voice to turn on or turn off the light.

### **6.2.4 Music controlling**

Respective user cases: UC-2

Navigation and Data Entry: 1 3 clicks

Prerequisite: 1. User login (3 clicks and 8 keystrokes)

- 1 Click the music figure button to enter the music-controlling interface.
- 2 Click the music play button to turn on the music.
- 3 Click pause to pause the music.
- 4 User enable use voice to control the music overall process click on the click on the microphone graphic and then use the voice to play or pause the music.

### **6.2.5 Modify the brightness**

Respective user cases: UC-3

Navigation and Data Entry: 1 3 clicks

Prerequisite: 1. User login (3 clicks and 8 keystrokes); 3. Light controlling - the light should be turn on (0 3 clicks)

- 1 Press on + and then drag the button to modify the brightness
- 2 User enable use voice to modify the brightness overall process click the micro-



phone graphic and then use voice to modify the brightness



### **6.2.6 Modify the volume of music**

Respective user cases: UC-4

Navigation and Data Entry: 1 3 clicks

Prerequisite: 1. User login (3 clicks and 8 keystrokes);

4. Music controlling (0 3 clicks)

1 Click  and  to modify the volume of the music

2 User uses voice to modify the volume of the music click the microphone graphic and then use voice to modify the volume of the music.

### **6.2.7 Elements status check**

Respective user cases: UC-5

Navigation and Data Entry: 0 clicks

Prerequisite: 1. User login (3 clicks and 8 keystrokes)

1 When the use login the system the music and light status will show on the screen, user enable check these information directly.

### **6.2.8 Control information feedback**

Respective user cases: UC-12

Navigation and Data Entry:

Prerequisite: 1. User login (3 clicks and 8 keystrokes)

This section will be divided into several parts to discuss

1 User uses button to control the system The feedback information will show on the screen directly, hence the user effort estimation depend on what operation the user chooses refer to the 1, 2, 3, 4, 5, 6

2 User uses voice to control system.

1: If the voice control fails, the feedback information will show on the screen directly.

2: If the voice controls successes, the feedback information will show on the next screen directly.

### **6.2.9 Light interface**

Respective user cases: UC-12

Navigation and Data Entry: 2 clicks and 50 keystrokes

- 1 Click sign in button to enter the create new accounts interface
- 2 Enter the new username
- 3 Enter the new password
- 4 Enter the new email address
- 5 Click finish button to complete the sign in process.

## 7 Domain Analysis

### 7.1 Domain Model

Voice based home motivation has many parts. Devices controlled or monitored may include electronic ovens, motion sensors, music players. Most, if not all, of the useful sensors that this project plans to utilize will send or process a single data value. This makes it very convenient to lay out a generic way to process the inputs from, or outputs to these devices. Each device can then be reduced to an object bearing a single data value to report to the system, or an object receiving a single data value from the system to govern the device's actions, or both. These standardized objects will be the device nodes as shown in the diagram. The data values obtained from these devices can then be categorized into analog, digital, or none (if the device hasn't a value to report, or cannot be controlled). We give our solution in three major fields, the application in the android platform, the server and related algorithm, the intelligent elements at home and the communication through different network

#### 7.1.1 Domain model-General

The general domain model is showed in figure 21, also the object lifelines is showed in figure 22.

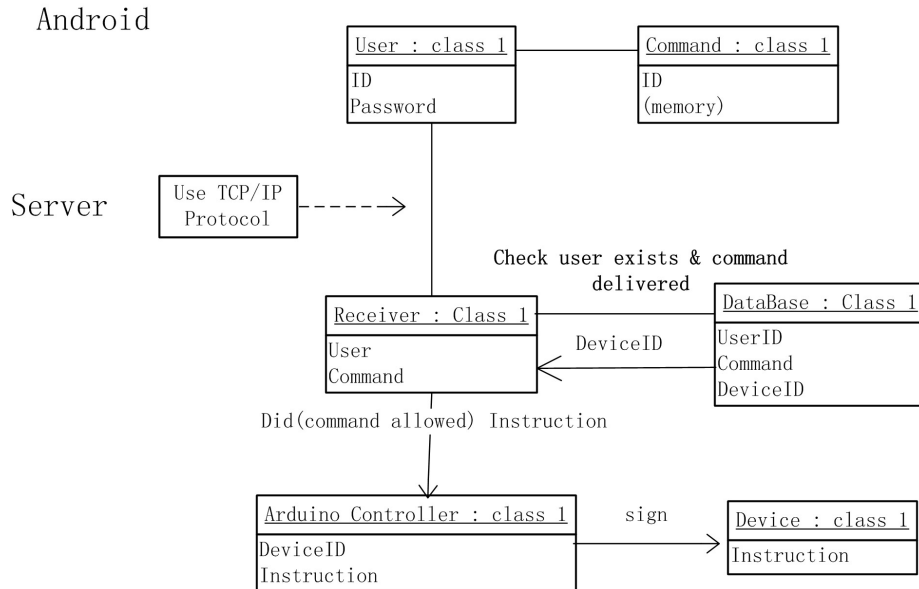


Figure 21: General domain model.

We have 3 layers. One is the Android layer which stores the information the user sent in and pass requirement of the user or the Admin to the server through the Internet. And the second layer, the Server layer, which receives the requirement the Android layer gets, and compute it to signals easy to understand by the Arduino Module by Wi-Fi or Internet. The last layer, which constructed with an Arduino Module and a MCU, is for receiving and control the electronic devices in home, the music player and the lights,etc.

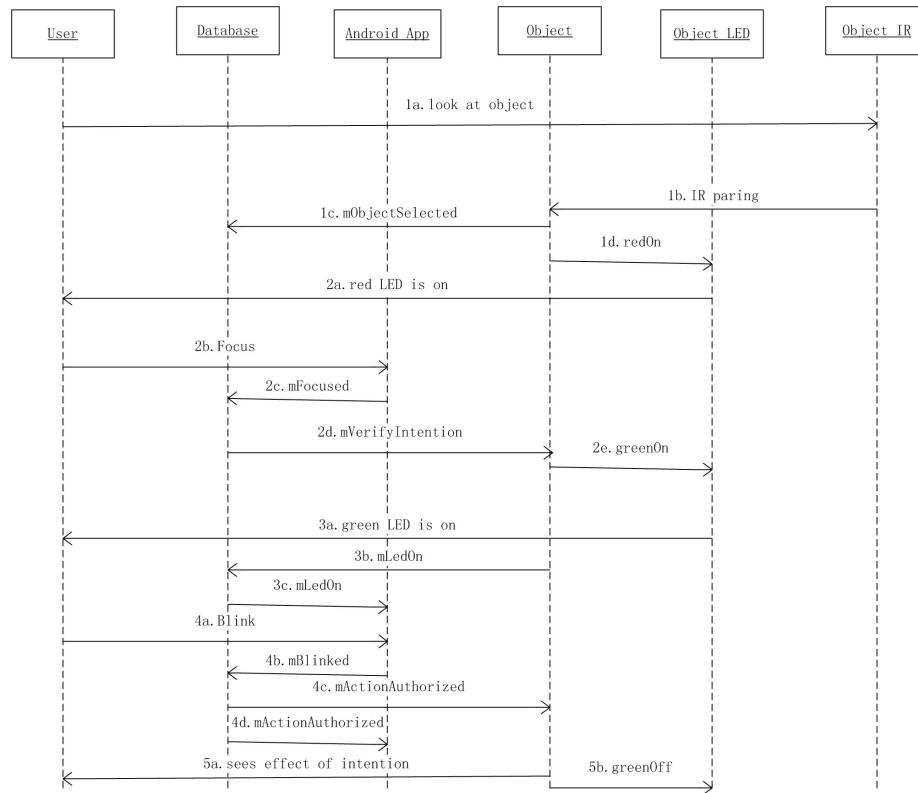


Figure 22: Object lifelines.

This is the object lifeline of the whole project, which illustrate how the user controlled the electronic devices in home by the Voice Based Home Automation.

### 7.1.2 Domain model-Server

The server is the brain of the whole home automation system. The cell phone or some other android platforms do not understand the users requirements and these devices just do some dictation works. The server gets the users words by communication with devices through networks; and then, the server analysis

the words and sentences people talk.

The analysis work can be extremely difficult. Our solution is starting with locate some key words in the sentences and try to figure out peoples requirements with those key words. Thats probably the natural way we use to understand each others talking.

We try to understand peoples requirements with some machine learn method. For example, if you want to turn on the light, you may say turn on the light, if you want turn off the light, the sentence should be turn off the light. We use a filter to pick out some top level keywords like Light, then we pick up some lower level key words like turn on, turn off, and then we can pick up some lowest level key words like 50 percent.

The server should also have the ability to communicate with the intelligent elements at home and the devices. When the server complete the analysis, the server should send some commands to the Smart elements to control them work in right way. Meanwhile the server should also send some feedback to the user to remind them the requirements have been accomplished.

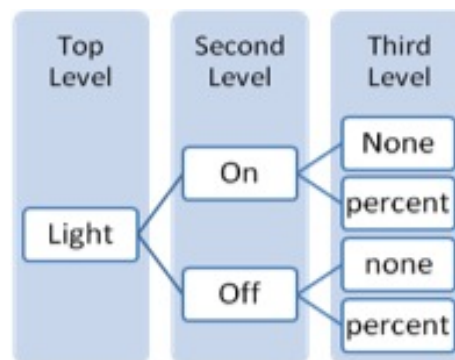


Figure 23: Server Model.

### 7.1.3 Domain model-Arduino

The android platform should have ability to communicate with server via LTE network or Wi-Fi. Anyway, the communication should base on the Socket protocol. Socket communication is widely used nearly everywhere and Google also gives the developer some free APIs to use.

After the command given by the user is sent to the server, in the server it is required to generate a specific code for each command. A serial code is generated

and communicated with the Arduino over the Wi-Fi.

For example:

If the code is 001501-02-01-1000-150111

We can split it up as:

*001501: start of code*

*02: user code (for e.g. 02- Vidur, 03- Chang)*

*01: device code (for e.g. 01- light, 02- fan)*

*1000: action code (for e.g. 1000- ON, 1001- OFF, 1010- status)*

*150111: end of code*

The Arduino will read this serial code and relay it to the MCU over the Infra-red bandwidth to perform the desired action. This process can be broken up into smaller modules to make it easier for the reader to understand.

**The First model can be considered the work done by the Arduino. It is required to do the following jobs:**

- Receive the signal sent by the server over the Wi-Fi.
- Decode the received signal and recognize what work and by which device is required to be done.
- Generate a binary code to be transmitted by the required infra - Red transmitter towards the device on the job needs to be performed (For Example a light which needs to be turned ON).
- Wait for the feedback from the Micro Controller Unit (MCU) regarding the status of the device on which the job needs to be done.
- Generate an alarm and reset the MCU, in case no feedback is received in the required amount of time.
- In case the feedback is received, send a confirmation signal to the MCU and relay the status of the MCU to the server.
- In case if the state of the device is changed manually, send the status to the server through Wi-Fi.
- If authentication of the user fails a number of times, Lockdown the whole house and sound the alarm.

**The Second model can be considered the work done by the Micro Controller Unit (MCU). It is required to do the following jobs:**

- Receive the signal sent by the Arduino over the Infrared bandwidth.
- Check the state of the device it is connected to.
- In case the device is already in the state, in which the user wants it to be, send a feedback to the Arduino communicating the same, so that the server can tell the user that the device is already in the required state.( For example if the user wants to turn ON the light and the light is already in the ON state.)
- In case the device is not in the state, in which the user wants it to be in. Then change the state of the device through the electronic circuit and send a feedback to the Arduino regarding the same.
- Keep on sending the feedback code until it receives a confirmatory signal from

the Arduino.

-In case if the state of the device is changed manually, send the status to the Arduino.

The arduino is based on MCU and with that micro-controller we can use AD/DA to control the luminance of the light, we can play or stop the music. With thousands of open source projects in the Internet, we can do a lot of things with MCU based elements at home.

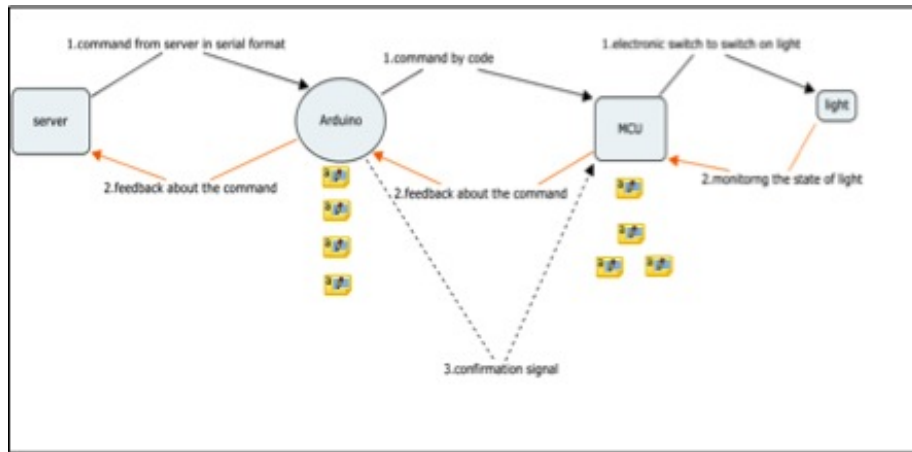


Figure 24: General description diagram.

#### 7.1.4 Domain Model-The communication protocol

The communication is the chain to connect the server the devices and the smart elements at home together as a system. It is a challenge for us to build a reliable communication between different devices and different networks.

However, though devices and network changes, the IP address and the mac address stay the same (in general cases). The communication based on Socket is suitable for our apply situation. However we still need to build a communication protocol in application layer. The protocol should include some key elements like the time, the event, the key command, and some labels to label the state like in processing, done and failed.

## 7.2 System Operation Contracts

### 7.2.1 Light controlling

#### Preconditions:

The Light Smart Element has an Internet connection

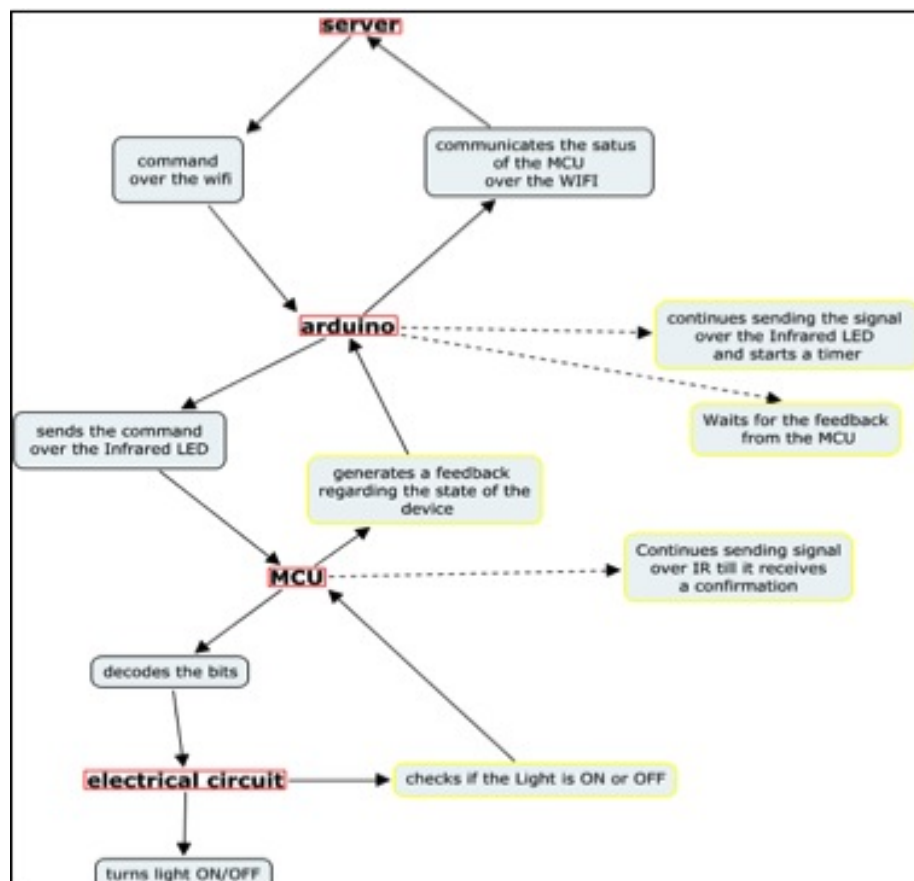


Figure 25: The Camp for the two modules.



The Light Smart Element can update it condition

**Post conditions:**

None.

### 7.2.2 Music controlling

**Preconditions:**

The Music Smart Element has an Internet connection

The Music Smart Element can update it condition

**Post conditions:**

The music Smart element can send updates of volume and music information

The Music Smart element can play music without Internet connection.

### 7.2.3 Modify the brightness

**Preconditions:**

The Light Smart Element has an Internet connection

The Light Smart Element has DAC to change the voltage

**Post conditions:**

None

### 7.2.4 Modify the volume of music

**Preconditions:**

The Music Smart Element has an Internet connection

The Music Smart Element can update it condition

The Music Smart Element has DAC to change the voltage

**Post conditions:**

The music Smart element can send updates of volume and music information

The Music Smart element can play music without Internet connection.

### 7.2.5 Elements status check

**Preconditions:**

The Smart Element has an Internet connection

The Smart Element can update it condition

**Post conditions:**

The Smart element can notify the changes when status changes.

### 7.2.6 User login

**Preconditions:**

The User has an account

The Server has the users information

**Post conditions:**

The server can record the users login history.

#### **7.2.7 Create new users accounts**

**Preconditions:**

The User does not have an account

The Server does not have the users information

**Post conditions:**

The server can record the users login history.

#### **7.2.8 Remote user**

**Preconditions:**

The User has an account

**Post conditions:**

None

#### **7.2.9 User log out**

**Preconditions:**

The User does not have an account

The User is logged in

**Post conditions:**

The server can record the users log out history.

#### **7.2.10 Check the authority of user**

**Preconditions:**

The User does not have an account

The Server have the users information

**Post conditions:**

The server can update the users information via Internet.

#### **7.2.11 Network error notification**

**Preconditions:** The Smart element has Internet connection

The server has Internet connection

The smart Elements at home have Internet connection

**Post conditions:**

The system is able to rebuild the connection of the whole system.

## 7.3 Mathematical Model

In this section we are going to talk about the mathematical models in our system.

### 7.3.1 The fuzzy recognition model

Though we use Google s voice dictation APIs to do the voice recognition work, and Google did a good job in voice recognition, the recognition results are still affected by multiple factors. The noise in the environment, the accent and even the talking speed can have some bad influence on the accuracy of recognition.

However, we want to build a system that even the server only recognize a part of the command the system is still able to give user some feedbacks and options.

In the natural command sentences there are usually 4 types of words: nouns, verbs, adjectives and some other words. By detecting these words we can know or partly know the users idea. For example if a user want to turn on the light he will say, Turn on the light. Here Turn on is the verb and it tells us the action need to be done. Light here is the noun, which shows us the subject the user want to control.

In our server algorithm, we classify the input command in 4 levels: Subject, actions, degrees, and other information. The sever first segments the every single words in the command and then put the words in different levels by cross compare the words in different word pools.

Then we can know exactly what is the users requirement. If the recognition of tome part of the command is failed we can still provide user some action options by the information from other level. The level model and discription is showed in table 20.

Level	Description	Addition
1	Nouns	AIIf missing, system provide some information about the types of Smart Elements based on Level 2 words.
2	Verbs	If missing, System give some optional actions based on level 1 words.
3	Words about degrees	If missing the system do the action without degree information and send user some feedbacks.
4	Others	No use.

Table 25: Different levels of words.

## 8 Interaction Diagrams

### 8.1 Design Patterns

There are three types of Design Patterns; creational Patterns, Structural Patterns, and Behavioral Patterns. Creational patterns are ones that create objects, rather than having you instantiate objects directly. This gives your program more flexibility in deciding which objects need to be created for a given case. Structural types concern class and object composition. They use inheritance to compose interfaces and define ways to compose objects to obtain new functionality. And most of behavioral design patterns are specifically concerned with communication between objects. The sub class of types of each type is showing below in the diagram below. Obviously the behavioral design is better for us.

Here are the specific details of the sub types of Behavioral Pattern.

- 1.Chain of responsibility delegates commands to a chain of processing objects.
- 2.Command creates objects which encapsulate actions and parameters.
- 3.Interpreter implements a specialized language.
- 4.Iterator accesses the elements of an object sequentially without exposing its underlying representation.
- 5.Mediator allows loose coupling between classes by being the only class that has detailed knowledge of their methods.
- 6.Memento provides the ability to restore an object to its previous state (undo).
- 7.Observer is a publish/subscribe pattern which allows a number of observer objects to see an event.
- 8.State allows an object to alter its behavior when its internal state changes.
- 9.Strategy allows one of a family of algorithms to be selected on-the-fly at run-time.
- 10.Template method defines the skeleton of an algorithm as an abstract class, allowing its subclasses to provide concrete behavior.
- 11.Visitor separates an algorithm from an object structure by moving the hierarchy of methods into one object.After comparing each design pattern carefully, the Iterator, Observer, State can be used as Design Pattern of our project, among which the Observer suites our project the most.

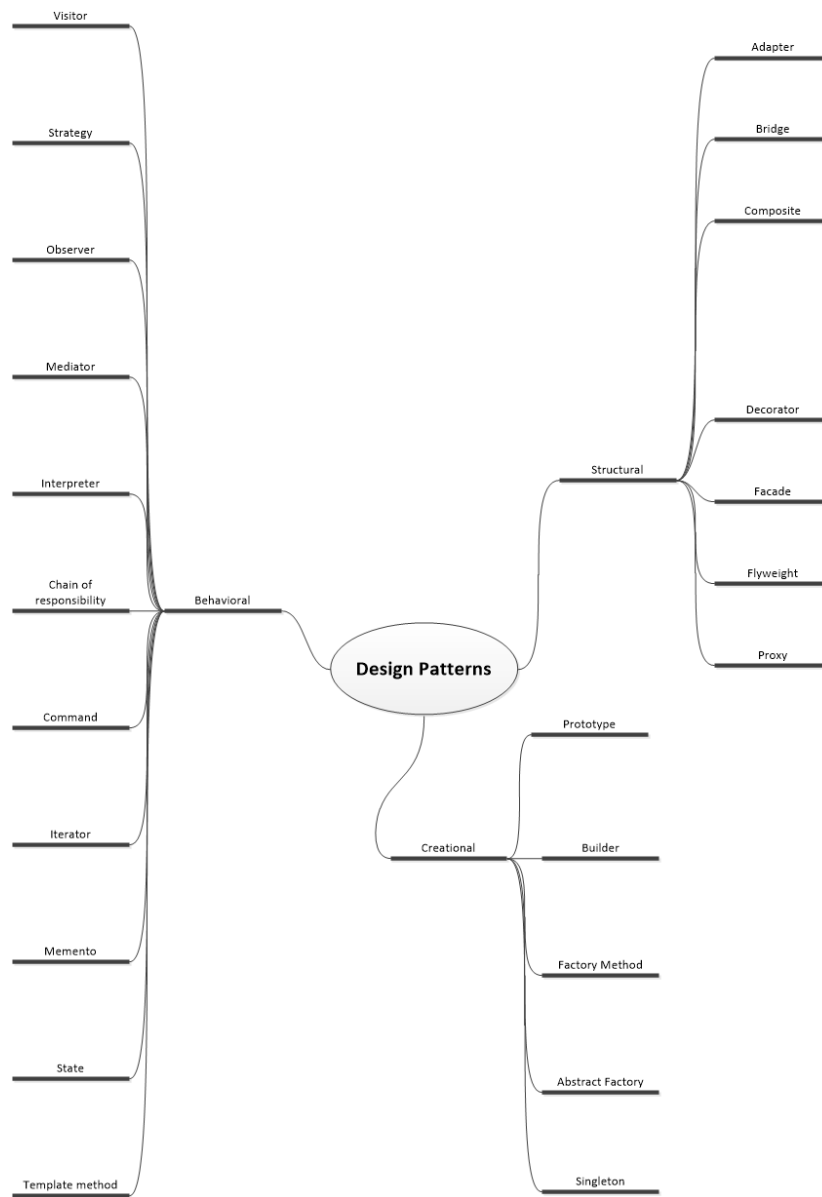


Figure 26: Design Patterns

## 8.2 UML Diagrams

### Use Case 1-User Case 4: Controlling and Modification of Elements

#### Use Case 12: Control Information Feedback

User Case 1-User Case 4 allow the users to control and modify the light and music player by their voice messages. These are the main functions of the home automation system. The diagram below shows the general process of the system. There are 4 actors involved in the process: User, Server, Arduino, and Elements. The user selects the control function on the main UI, and UI controller displays the referred control UI to the user. After the user sending the voice message to the server, the server will process the message with the core algorithm and coding the message with the communication protocol then send to the Arduino and the elements. This process involves the feedback information between server and user via cellphone. When the server identifies the message failed, the feedback function will be activated to send the error notification to the user interface.

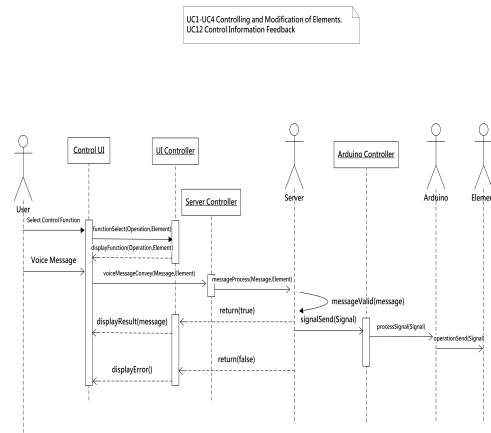


Figure 27: UC1-UC4,UC12

## Use Case 5: Elements Status Check

Use case 5 allows the user to check the current status of selected element. The purpose of this use case is to monitor the system and make sure all the elements are working normally to guarantee the security. This process will be achieved by the communication between user interface, server, arduino and elements like below. When the user selects the function, the cellphone will send a request instruction to server. And the server also sends the request instruction to the arduino and elements after coding the message. When the elements receives the request, they will check the status of themselves and send the feedback information to the server. Then the server can read the information and process it and send back to the user interface.

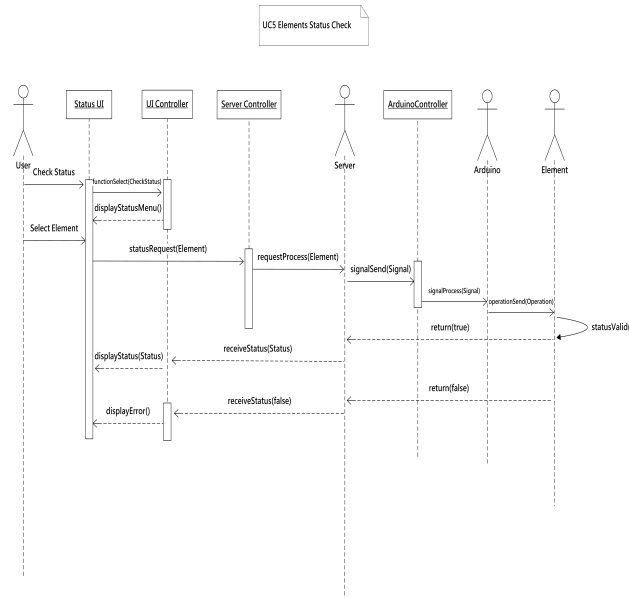


Figure 28: UC-5

## Use Case 6-User Case 11: User Operation and Management

In these series of use cases, they involve several operations for the user and administrator including log out and login action, authority management, create new account, remove account, and check the information of the users. The users can be classified into general users and administrator. There is only one administrator in the system who can modify the authority of users and remove users. Also the administrator has the full-permission of the operating to the elements. The general users have limited authorities to access the elements and database. The database located in the server has played an important role in this series of use cases. All the information of users has been stored in the database and the program should access the database to do the operation of users. The separated diagram of these use cases will be showed below.

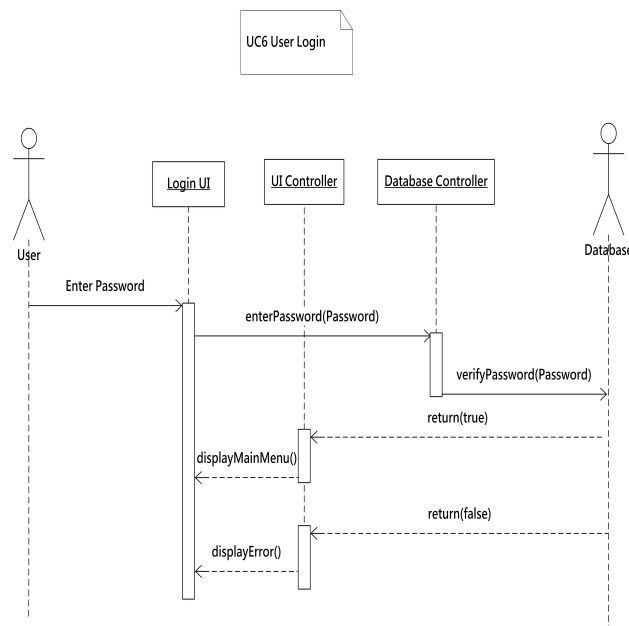


Figure 29: UC-6



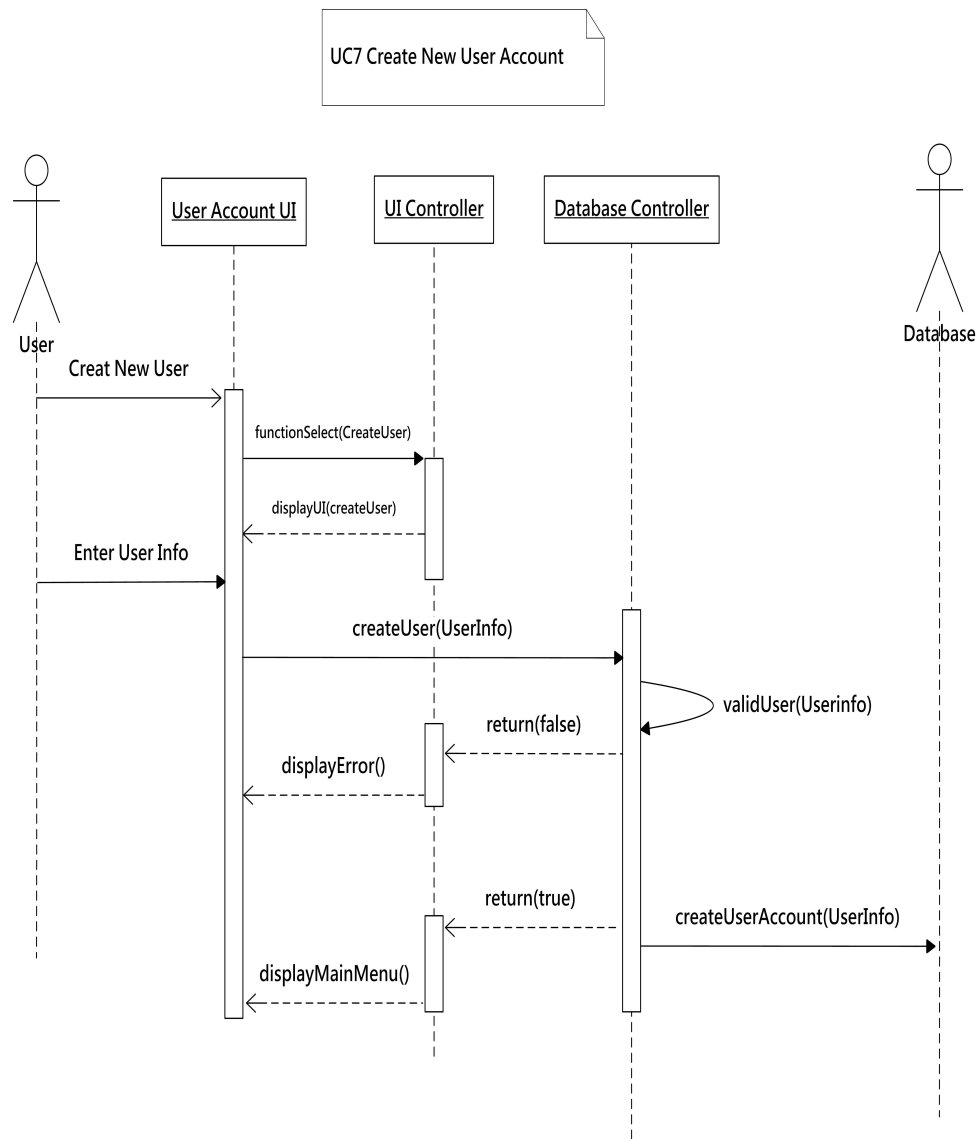


Figure 30: UC-7

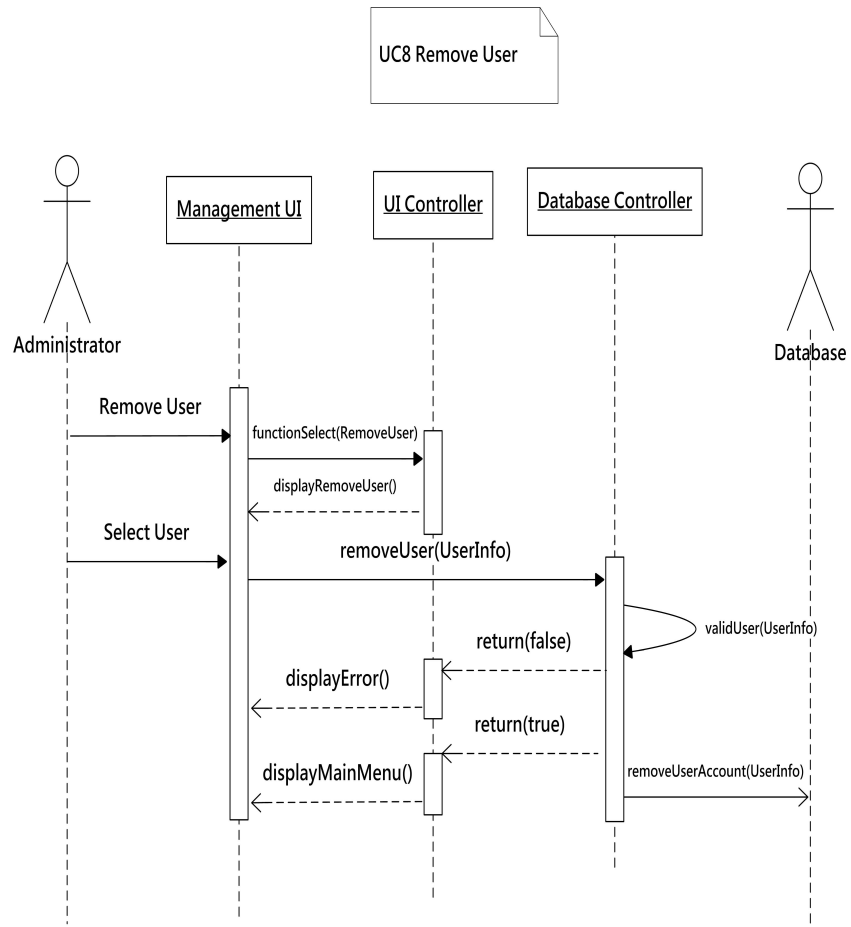


Figure 31: UC-8

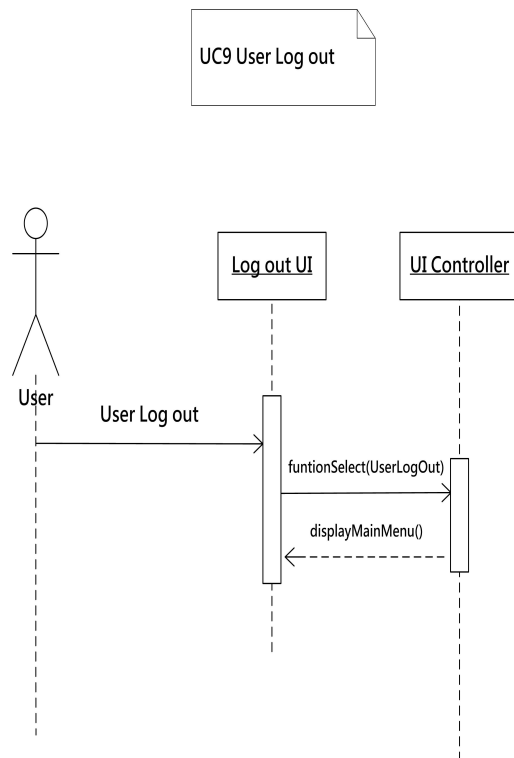


Figure 32: UC-9

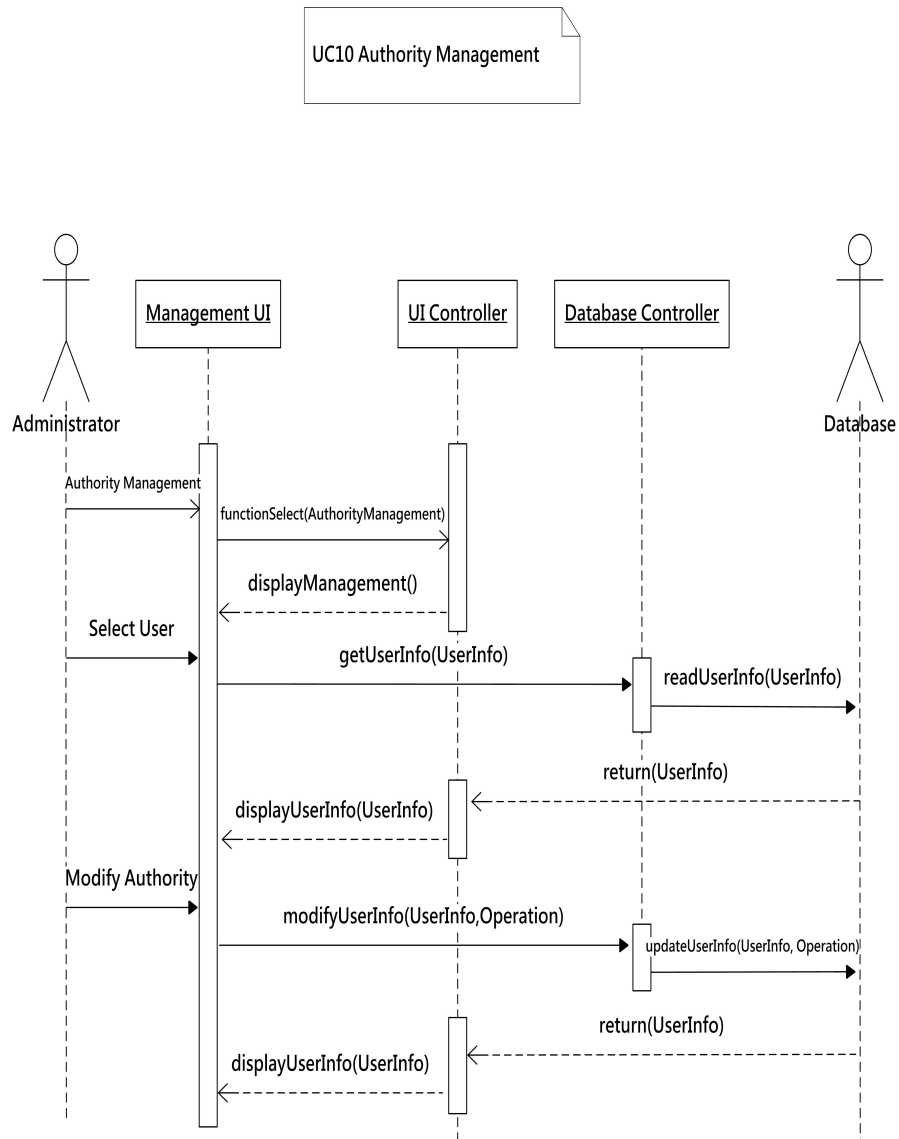


Figure 33: UC-10

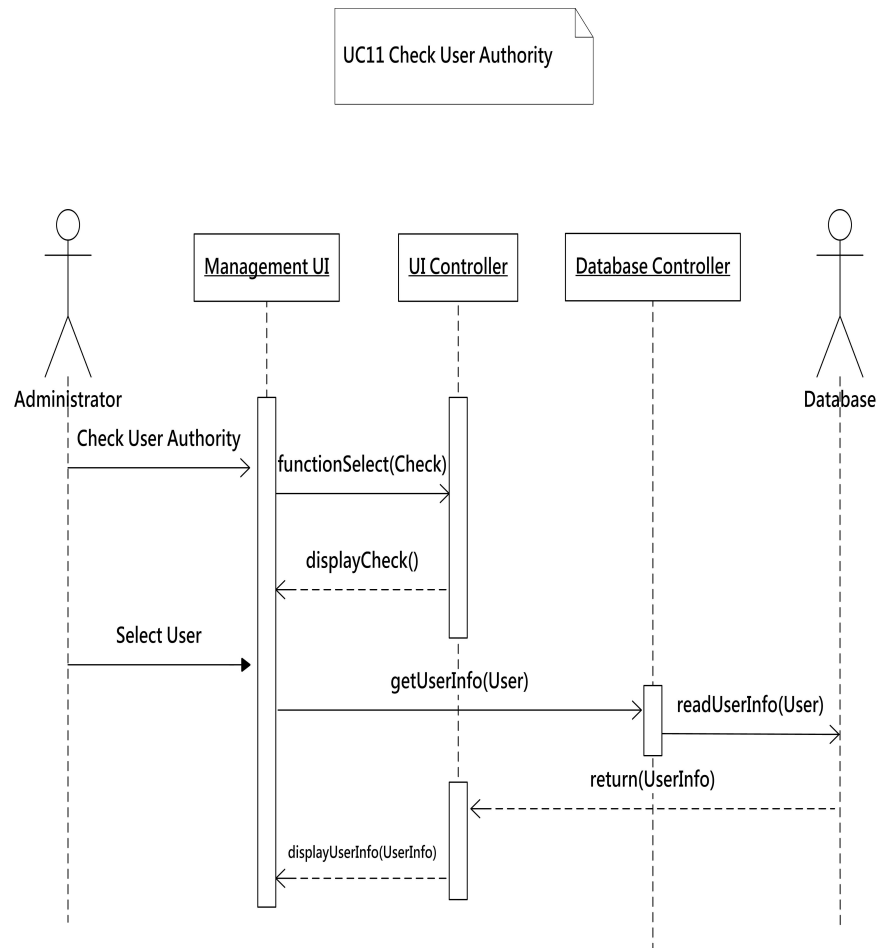


Figure 34: UC-11

### Use Case 13: Network Error Notification:

UC-13 offers a function of checking the network status for users. Because the communication between cellphone and server uses wireless internet, it can be unstable with several external reasons. So it is necessary for users to check the network whether connected. If the network has been interrupted, the system will send a notification to the user interface to remind the users and other operations based on the network will be paused. When doing this function, the cellphone will send a request instruction to the server. And the server can check the connection by some algorithm and send the feedback information to the cellphone. If the connection failed, the error notification will be pushed to the user interface.

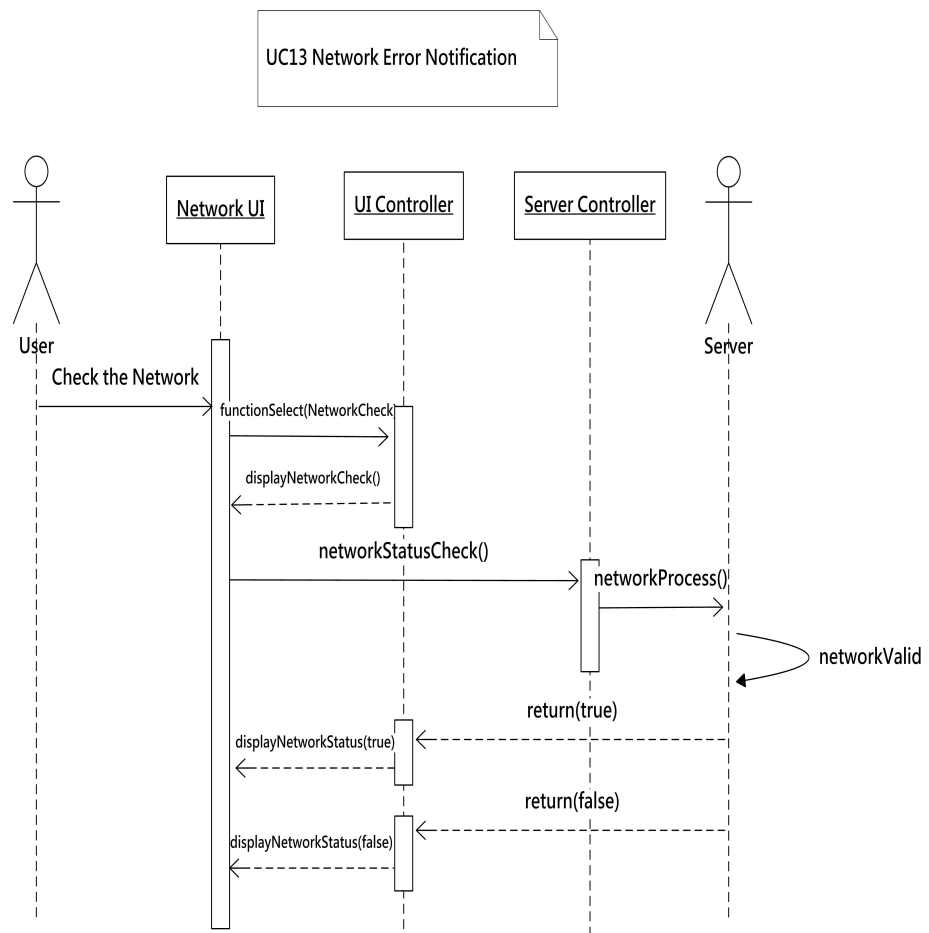


Figure 35: UC-13

### 8.3 Alternative solution description

**c.2.1 User Case 1— *Light Controlling*** The actor goal of this part is to turn on or turn off the light, hence, the basic requests from users is to control the light switch. There may exist a lot of alternative solutions to help us solve this problem, for example, we can implement button to realize these functions (considering the extra demands from customers and In order to avoid voice control problem, we add the button function into our system). However, compared with the voice control, it may cost more time and resource to complete the light controlling by button, we need more than three buttons to complete just one instruction, such as turn on the light or turn off the light, but for voice, we only need one sentence to solve such problem, customers and users do not need remember and care about button function, couple of words or a sentence enable complete the whole requirements. This method extremely reduces the elements of the external world which may interact with the system-to-be.

**c.2.2 User Case 2— *Music Controlling*** Just as the alternative solution mentioned above, voice control possesses a great amount of advantages than the button, however, in this part, we will explain the reason why we add the buttons into our system. As a part of our system, button also play an important role. At first design, we do not include the button design, because for a voice based system, button seems useless for the implementation. However, for music controlling part, we decide to add buttons, because music may influence our voice recognition and make our system not available. For these reason, in this part, button is a good design for the system compared with other alternative solutions (the only voice control system plays as an alternative solution in this part).

**c.2.3 User Case 3— *Modify the Brightness* / c.2.4 User Case 4— *Modify the Volume of the music*** For these two user cases, there is an alternative solution can be used for both of them. At the original design, we decide to use text message and keyboard to control the brightness and volume. However, compared with the voice control, this method may need too many interactions between the system-to-be and the users; as for the developers, they also need define many actions to realize the users requirements. Voice control, which bases on Android api by using the google voice, enable the developers save a lot of time and resource to realize the system and complete the requirements from customers.

**c.2.5 User Case 5— *Elements Status Check*** Elements Status Check is a basic request from the customers, at the original design, we provide several alternative solution to realize this function. For example, adding buttons to enable the users select which status they wish to check (checking the music player status or checking the light status), however, in the end we choose no button in this part, which means when the user logs into the system, they can check the information directly on the screen. Compared with former solution,

this may save customers time and also the resource of the system, at the same time, users enable decide what they will do by checking the status directly.

***c.2.6 User Case 6— User Login / c.2.7 User Case 7— Create New User Accounts*** For these two parts, we provide a solution at the beginning of the project, username and password are the basic elements and we just design these two elements in the login system and create new user accounts system at the original program. However, as for a great application in the cellphone, the update and push notification are also the necessary parts. Hence, the username and password are not enough for the systems development and the communication between the users and the developers, for these reason, we add email information request into the new user account. Compared with the original alternative solution, this solution may help the developers and users save time and convenient to both of them.

***c.2.8 User Case 8— Remove Users / c.2.11 User Case 11— Check the User's Authority*** Alt. solution: Server delete the all the account information but the administrator. The cost of this alternative solution is cheap; however, it is not practical. The server has no idea which user information should be deserted or is illegal, so it can only delete all the information and leaves the administrator only. Then the login in to the accounts will become inconvenient, for the accounts other than the administrator has all been deleted, which is obviously not our original intention. And it will not work when change the authorities of users.

***c.2.9 User Case 9— User Log Out*** Alt. solution: To log out from the system (without special protection to personal information). Alt. solution: To log out from the system (without special protection to personal information). In this case, the user information would not be deleted, which will possibly result in insecurity. The information of the user will be unsafe. And whats more, the abundant information of users will be stored in local storage or the storage of the server. If the data is stored in local storage, it will be a waste of the room. And it will be more serious if the data is stored in the server, because the user information should first be sent to the server, so along with the problem of storage consuming in the server, the workload of the whole communication will enlarge.

***c.2.10 User Case 10— Authority Management*** Alt. solution: Cellphone doesnt send the modified information to the server, instead it changes the authorities of this user directly. In this case, the server will not be able to modify the information in database. As long as the fail of updating in time, the whole system does not have an synchronous database, which will result in later disorder of data. Another result from this is the fail of checking the authority security.



***c.2.12 User Case 12— Control Information Feedback*** Alt. solution: Only have the server and the cellphone connected during the feedback query, resulting in asynchronous of other parts of the system. It is not bad to updating the other parts later when the whole networking is complete, however, the result may turn out to be disturbing to the UC13, which is notify the network error. Because if the information of the whole system could be updated during the network error, then it will no longer be a network error, and thus the network error will be difficult to classify.

***c.2.13 User Case 13— Network Error Notification*** Alt. solution: give notification whenever the communication between two parts of the whole system is interrupted. This would not be effective because the links between the parts of the system without the cellphone will not be frequent. And notify their network error will not be useful and it will take too much time

## 9 Class Diagram and Interface Specification

### 9.1 Class Diagram

dig.png

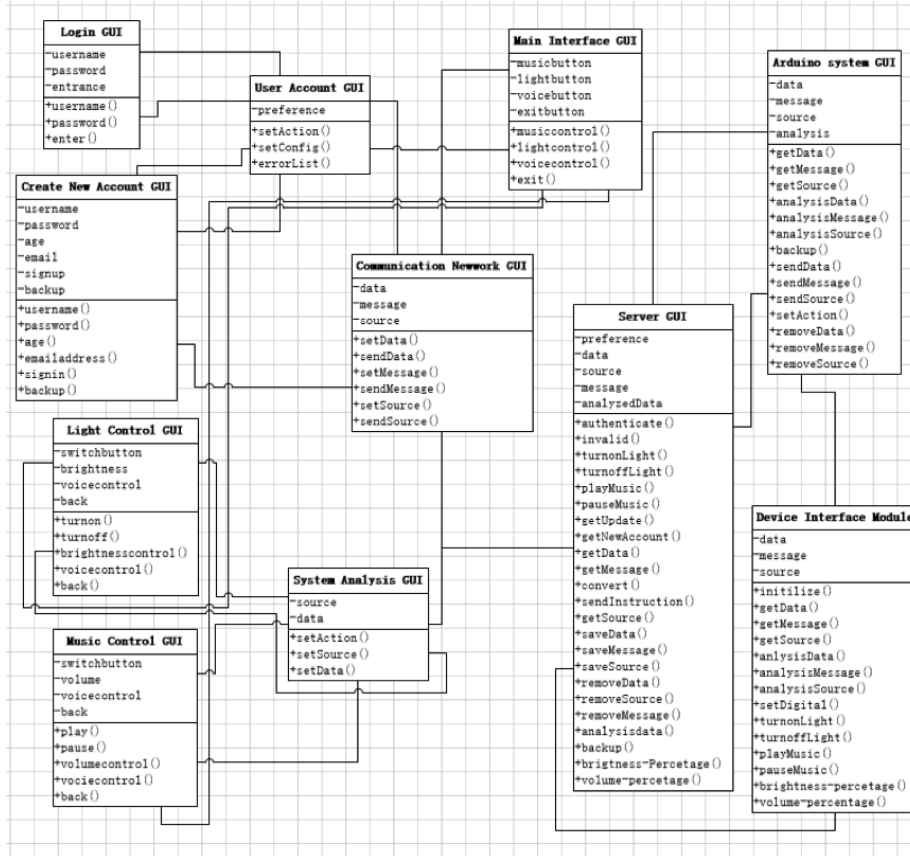


Figure 36: Class Diagram

From figure, we know that there are mainly three parts in the class diagram, the first part we call it UI part, in this part, there also have a lot of small GUIs, such as login, create new account, user account, main interface, light control, music control, and system analysis, the class diagrams of this part is mainly about the UI function. The second part is server part, this part also can be divided into two small parts: server GUI and communication GUI, this part is about the server operations and its data information. The last part is the Arduino, which is designed for the hardware, and Arduino also has two small parts: Ardiuno system GUI and device interface module GUI. Each class in the

figure connect with at least two other classes, and we will show the details for these classes in the following content.

As for the UI part, we can also divide it into two parts to introduce it, the first is the noncontrol part. This part contains login, create new account, user account, and main interface, the function of this part is to help the users to use the system, login GUI enable user type their usernames and passwords, and then user account enable get the information and begin to check the username and password, if it is correct, the user will go to the main interface part, at same time, if the user wish to create new account, the create new account GUI also provide the possible for the users to help them sign up. The main interface GUIs function is to help the user to choose which part they wish to control, it is the center part between noncontrol part and control part. Control part contains three parts: light control, music control and system analysis. The first two GUIs are mainly for the control function, users requirements, such as turn on or turn off the light, play or pause the music, brightness of the light, and volume of the music, can be satisfied by these GUIs, and the system analysis will analyze the source and data which come from the first two parts. The server part is the bridge between the UI and Arduino, the communication network enable the data and resource from UI part to be sent to the server, and the server will analyze all information and then send the corresponding information to the Arduino and the UI part.

The Arduino also can be divide into analysis part and module part, the first part will get the data and message from the server and then analyze and convert it to the instructions for the module, after that the module will execute the corresponding instructions. The system will work well when the whole parts work together, and also we should know that not all of the operations information will be sent to server, some application operations, such as back up and next, do not necessary for the server, only useful data or information will be translated into the server (control of the light and music, create new account). We will show the partial diagrams and more details in the next section.

## 9.2 Data Types and Operations Signature

In this part, we will present all types and signatures for the above data and operations. The class will be divided into several subclass.:

### 1 .Login GUI

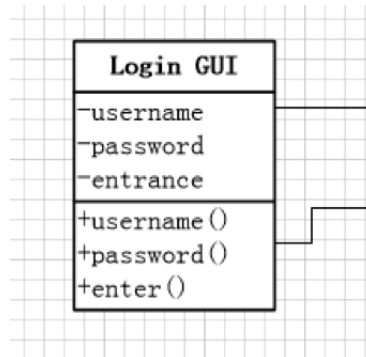


Figure 37: Login GUI

The data types and operation signatures: Username: java:string

Password: java:string

Username():void

Password():void

Enter():void

In this part, the users enable type their usernames and passwords to enter the whole system. When the user type username and password and click the enter button, the login GUI will send the corresponding information to the user account to check the user state. If the username and password correct, user will enter the system; if not, the user will get the error message.

## 2 .Create New Account GUI

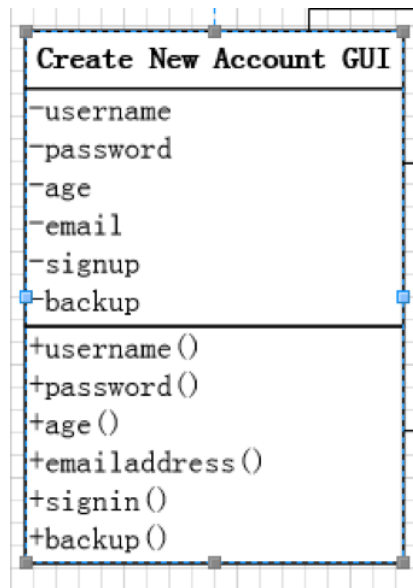


Figure 38: Create New Account GUI

Username: java:string  
Password: java:string  
email: java:string  
Username(): void  
Password():void  
Age():void  
Emailaddress():void  
Signin():void  
Backup():void

As for the create new account GUI, the user enable register new account in this part. The users should type new username, password, age, and email information into the system, and then click signup button to complete the process, and they also enable get back to the login state. When they finish the register, the create new account GUI will send these information to the server by using communication network GUI.

### 3 .User Account GUI

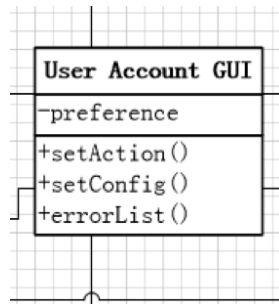


Figure 39: User Account GUI

Preference:java:string  
setAction():void  
setConfig():void  
errorList():void

When the user creates a new account, the create new account GUI will send the new information to the server to store it, and then back up to the login interface. As for the username and password, the user account GUI will check them, the user will enter the system if they correct or providing error information if not.

### 4 .Main Interface GUI

The data types and operation signatures:

Musicbutton:java:button  
Lightbutton:java:button  
voicebutton:java:button  
backbutton:java:button  
musiccontrol():void  
lightcontrol():void  
voicecontrol():void  
back():void

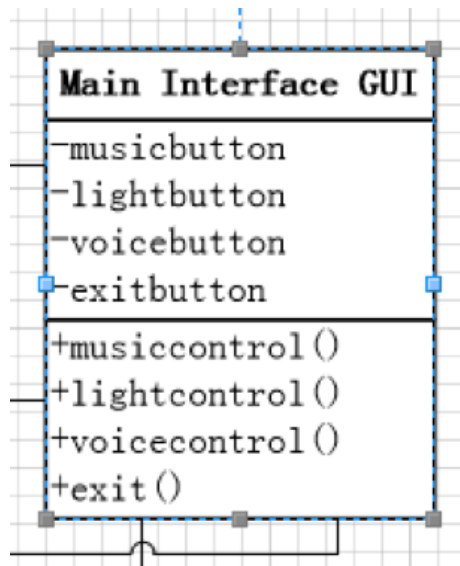


Figure 40: Main Interface GUI

The main interface GUI actually has three main functions, the first and most important part is the voice control. Users can use voice to control the system when they press the voice button. The second part is music button and light button, users enable choose to enter the music control interface or light control interface. The last part is exit part, users can exit the system by this button.

## 5 .Light Control GUI

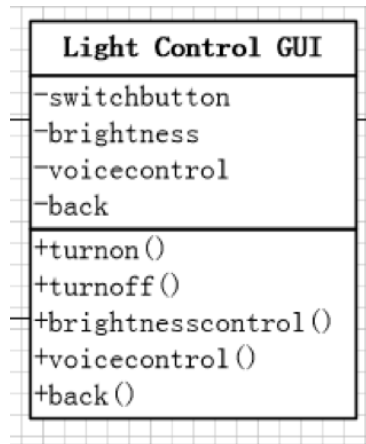


Figure 41: Light Control GUI

The data types and operation signatures:

Switchbutton:java:button

Brightness:java:button

Voicecontrol:java:button

back:java:button

turnon():void

turnoff():void

brightnesscontrol():void

voicecontrol():void

back():void

This part is mainly for the users who prefer button control, and at the same time, we also provide voice control here, the switch button enable turn on or turn off the light and the seek bar enable change the brightness of the light. Besides that back button enable the users to go back the main interface. The information the users required will be sent to the system analysis GUI in the end.



## 6 .Music Control GUI

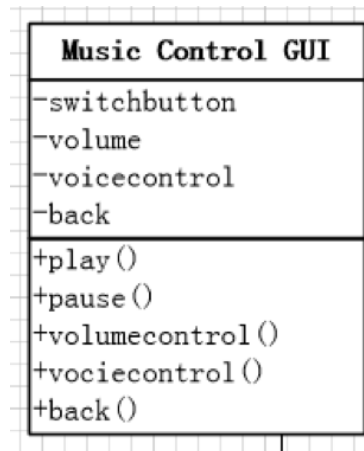


Figure 42: Music Control GUI

Switchbutton:java:button  
volume:java:button  
vociecontrol:java:button  
back:java:button  
play():void  
pause():void  
Volumecontrol:java:button  
Voicecontrol:java:button  
back():void

In this part, user can use voice to complete the action and also we provide button for them, switch button enable lay or pause the music, the volume can be adjusted by the volume button. At the same time, back button can help user go back to the main interface. All the data and information generated by this part will be sent to the system analysis GUI.

## 7 .System Analysis GUI

The data types and operation signatures:

source: java:string  
data: java:float  
setAction():void  
setSource():void  
setData():void

The system analysis GUI will analyze all data and source which generated by the music control part and light control part, these information will be set and send to the server part by using the communication network.

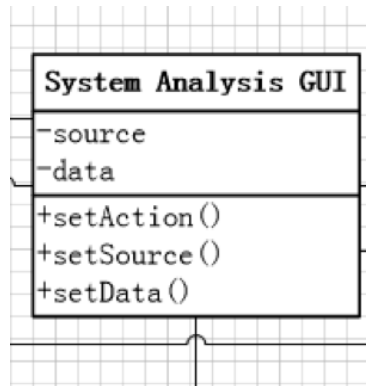


Figure 43: System Analysis GUI

## 8 .Communication Network GUI

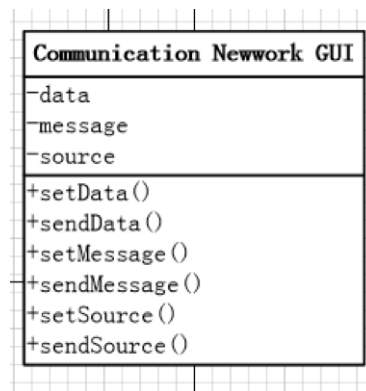


Figure 44: Communication Network GUI

The data types and operation signatures:

source: java:string

data: java:float

message:java:string

setMessage():void

setSource():void

setData():void

sendMessage():void

sendSource():void

sendData():void

Communication network are connected with almost all classes in this system.

The function of this part is very important, data transmission and conversion cannot realize without the communication network, it get message, data, and source from the other parts and then send them into the corresponding parts

## 9 .Server GUI

The data types and operation signatures:

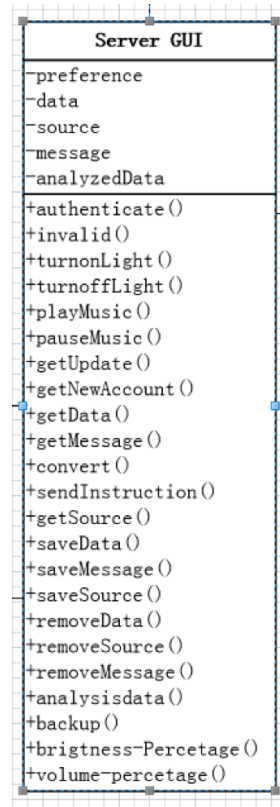


Figure 45: Server GUI

Preference:java:string  
Username: java:string  
Password: java:string  
email: java:string  
source: java:string  
data: java:float  
message:java:string  
analyzedData: java:float  
authenticate():bool

invalid():bool  
turnonLight():void  
turnoffLight():void  
playMusic():void  
pauseMusic():void  
getUpdata():void  
getNewAccount: void  
getData():float  
getMessage():string  
convert():void  
sendInstruction():void  
getSource():string  
saveData():float  
saveMessage():string  
saveSource():string  
removeData():void  
removeSource():void  
removeMessage():void  
analysisdata():void  
backup():void  
brightness-percentage():void  
volume-percentage():void

The server is the bridge between the hardware and the software, the user only use cellphone application and the information generated by the app will send into the server to store these data or message. After analyzing, server will send the analyzed data and message to the Arduino, and at the same time server need to give feedback to the software and hardware.

## 10. Arduino System GUI

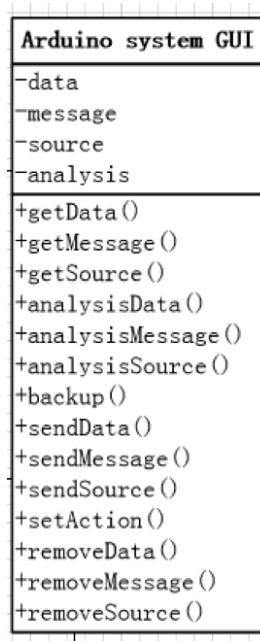


Figure 46: Arduino GUI

```
source: java:string
data: java:float
message:java:string
getData():float
getMessage():string
getSource():string
analyzedMessage: java:string
analyzedSource: java:string
analyzedData: java:float
backup():void
sendData():void
sendMessage():void
sendInsorce():void
removeData():void
removeSource():void
removeMessage():void
```

The Arduino system will get information from server and then it will analyze these information and convert them to the corresponding instruction for the module.

## 11. Device Interface Module

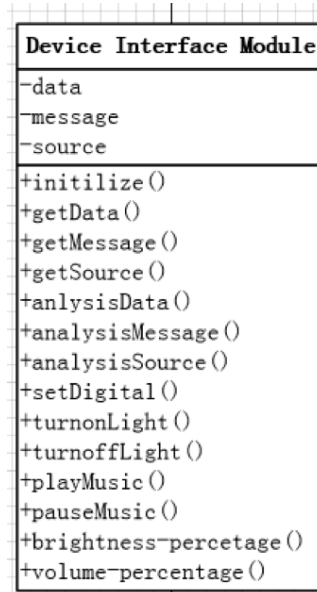


Figure 47: Device Interface Module GUI

```
source: java:string
data: java:float
message:java:string
initialize():void
getData():float
getMessage():string
getSource():string
analyzedMessage: java:string
analyzedSource: java:string
analyzedData: java:float
setDigital():void
turnonLight():void
turnoffLight():void
playMusic():void
pauseMusic():void
brightness-percentage():void
volume-percentage():void
```

The module enable receive the instruction from the Arduino system and execute the corresponding execution.

### 9.3 Traceability Matrix

From the figure, we know that the different GUIs will be involved in different interfaces, for example login GUI will be connected with the login interface, new account interface, main interface and server, and it will send or get data or message from these parts. We have already provide the involve information about these part in last section, hence, we will do not explain the relationship of every GUI and interfaces, however, some important information we should learn from the matrix. 1. User account GUI is the key among the UI part, it connect the login, create new account, and main interface. 2. System analysis GUI play an important role in the control part, whether the server can get the right information or not depends on whether the system analysis GUI work properly. 3. Server is the bridge between the user interface and Arduino. Server will get and send data or message from almost all parts and at the same time it should give some feedback to some specific parts. Besides that it also analyzes the data and message and convert them to make them match with the corresponding parts. 4. Communication network is the basic request for the whole project. Data transmission and message convert cannot work properly without the communication network.

### 9.4 Object Constraint Language (OCL) Contracts

As for the class diagram, there are some invariants, preconditions, and postconditions which we should consider and explain in our paper. The login GUI has some postconditions, the invariant of this part is entrance operation string; for other operations, when user and customers enter their username and password, the string of username and password will be send to the sever, and also the login GUI will receive the feedback string from the server, hence the preconditions are the login GUI and server GUI work normally, and at the same time communication network GUI work properly. As for create new account GUI, its preconditions are same as the login GUI, and it also has some other conditions we should consider: people who wish to enter the create new account interface should first get into the login interface, hence, before the create new account GUI we should get into the login GUI and in this part the invariants are back string and sign up string. After that we should consider main interface GUI, besides the conditions which we mentioned before, this part also need some other requests. The precondition of this part is login, all operations of the main interfaces can be executed until the login GUI finished it work. The invariants of this parts are the string of the exit button, music button, and light button. Light control GUI and music control GUI also has the conditions which we mentioned before, and they all has a same precondition: the main interface GUI. Every operations of these two parts cannot be executed before people get into the main interface GUI. The invariants of these two parts are switch buttons. As for the system analysis GUI, it has the same conditions as before and also need people choose instructions, for example, it will not work unless people choose to control the light or music by voice. Communication network GUI has the

most complicated conditions in this system, because it will be used in all of the system, hence it has the different conditions every time the instructions are sent between application and server, or server and arduino and it has no invariants. For the server GUI, its preconditions are there are message or instructions to the server which were sent from the other parts. In a brief, when user log into the system, every instructions which need contact with server will be the conditions for the server GUI. Arduino GUI will not work unless get instructions from the server, and the device module GUI will also not work unless get message from the Arduino GUI. In these above parts, they do not have the invariants.



	login interface	new account interface	light control interface	music control interface	main interface	communication network	server	arduino
	X	X			X		X	
Login GUI	X	X						
Create New Account GUI	X	X			X		X	
User Account GUI	X	X				X	X	
Main Interface GUI	X		X	X	X	X		
Light Control GUI			X		X	X	X	
Music Control GUI				X	X	X		
System Analysis GUI			X	X		X	X	
Communication Network GUI	X	X	X	X		X	X	X
Server GUI	X	X	X	X			X	X
Arduino System GUI						X		X
Device Interface Module						X		X

Figure 48: traceability matrix

## 10 System Architecture and System Design

### 10.1 Architectural Styles

Our project is too large to be completed by a single person in a short period. To make the development manageable, the best way is to divide it into components that can be developed (and later maintained) separately. Each component will be a work assignment for a team or individual. It is often thought that this decomposition is a management decision, determined primarily by the talent available. The decomposition is a critical design decision to be made on the basis of simple technical criteria. The result is a very unconventional, but easily maintained, design. And through discussion we decided to divide out project into 3 parts.

1. client and server architecture;
2. database architecture;
3. event-driven architecture;

In terms of that clients can make communication with their own house through user interface, we can simply treat our system as client and server architecture. When consider when clients register and they can set up their private account and preference, those information would be stored in database, for next time they log in, the system would call the certain data to control home intelligence which in our project is light and music, thus, we can see our system as data architecture. In our interface especially main interface, clients can see a list of the state of all of their home devices, when user wants to change their states, they can send their command by voice or clicking of the button to control the hardware, it makes our system fall under the event-driven model.

Now, I would introduce those three subsystem way.

**Client and server architecture:** Client/server architecture is an underlying framework that consists of many PCs and workstations as well as smaller number of mainframe machines, connected via local area networks and other type of computer networks. In our project, the destination is to establish the communication between clients and server so that clients can achieve their goal. And this architecture also allowed different clients to control the hardware through authorizing and identify them.

**Database architecture:** Database architecture is a collection of concepts that can be used to describe the structure of a data, and provide the necessary means to achieve this abstraction. It is composed of models, policies, rules or standards that govern which data is collected, and how it is sorted, arranged, integrated and put to use in data systems and in organizations. Data is usually one of several architecture domains that form the pillars of an enterprise architecture or solution architecture. In our project, database architecture can be divided into lower subsystem - data access subsystem and data store subsystem. Data access get the data from data store and communicate with intelligence through

server. Data store can hold the private information and preference of all the clients.

**Event-driven architecture:** is a pattern promoting the production, detection, consumption of, and reaction to events. From a formal perspective, what is produced, published, propagated, detected or consumed is a message called event notification, and not the event itself, which is the state change that triggered the message emission. For example, in our projects, clients can send their command or click on the button when they want to change the state of the light and music, their voice or the action just like a trigger to control the hardware.

## 10.2 Package Diagram

When modeling a large scale system, we are working with a high volume of model elements. We describe a model from different views and different phases, hence are in different types. UML package helps to organise and arrange model elements and diagrams into logical groups, through which we can manage a chunk of project data together. We can also use packages to present different views of the system's architecture. In addition, we can use package to model the physical package or namespace structure of the application to build. With this, we follow the convention of naming package like using lower case for Java package.

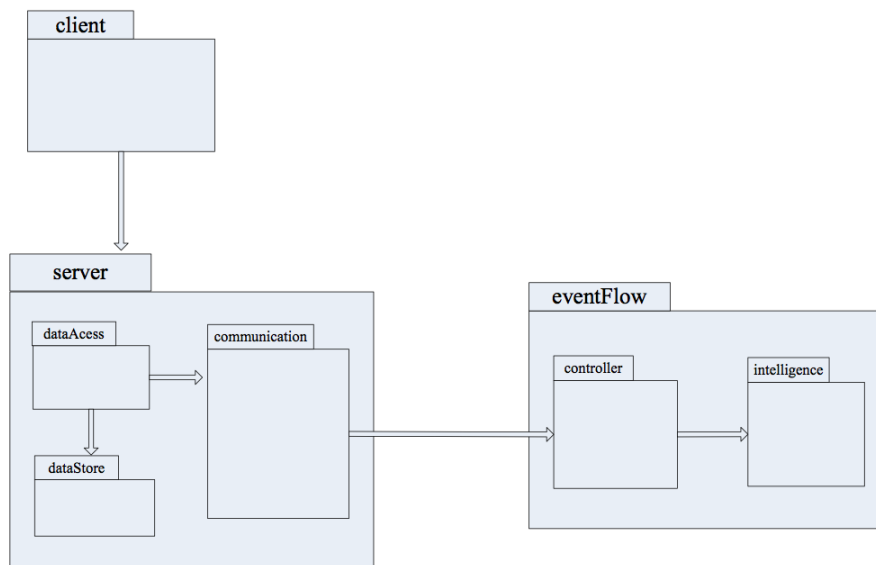


Figure 49: package diagram

### 10.3 Database

**A very simple database** In the user case we mentioned about the login and log out method. Generally, for the security purpose we put the user's account and passwords in the server. When the user request to login or log out, the device will send a message to the server and the server cross compares the username and passwords between the input and the data in the database. So far a very simple database is needed in our system, to keep the usernames and the related passwords for users to login and log out. This database is nothing but one table. We design it as follows:

#ID	Real Name	Account	Password	Address	Mark
1	Li Xinyu	Arthur	123456	812 Benner	0
2	Guo Jiaqi	Guo	123456	Busch	0
3	Nayyar	Vidur	123456	Dogglass	0

Table 26: A simple example of the basic table in the database

**Other requirements for the database** As we all know the Apps for the cellphones or tablets should be as light as possible so that it will keeps the system running smoothly and clean. However the home automation system needs to keep action records and the status of Smart elements. Thus we decide to put the action records into servers.

Also, we mentioned the status of the Smart elements at home should be tracked; we also need to keep records of the elements' status in the server.

Based on the requirements above we design some more tables in the database. They are showing in the table 2 and table 3.

#ID	Account	IP	Time	Element ID	Action
1	Arthur	128.98.12.128	12:40AM Oct.29 2013	L12	Turn on the light
2	Guo	128.98.12.128	12:40AM Oct 29 2013	L13	Turn off the music
3	Vidur	128.98.12.128	12:40AM Oct29 2013	M11	Turn up the light

Table 27: Table of action records

#ID	Account	Element ID	Time	Status	Mark
1	Arthur	L12	12:40AM Oct.29 2013	On 100	0
2	Arthur	L13	12:40AM Oct 29 2013	On 50	0
3	Arthur	M11	12:40AM Oct29 2013	Off	0

Table 28: Table of element status

**Relational Database** As we can see the three databases are related with each other, based on that we can use relational database to organize all the

tables in the database. The logic relationship between different table in the database is shown as follows:

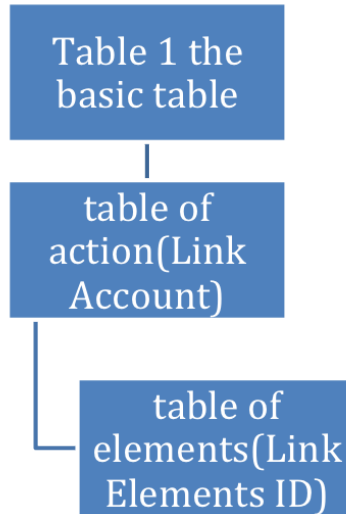


Figure 50: the logic of relational database

#### 10.4 The thread algorithm of the Server

The functional requirements of the server are to process the signals from clients and send commands to the Smart Elements. We have already discussed about those algorithm in previous chapters. In this section we are going to talk about the thread algorithm of the server. The algorithm serves the non-functional requirement that the server should be able to communicate with multiple clients at the same time.

In general, If the server have only one thread and once the server is in communication with a client, the thread is taken. No more clients can build the communication with the server because and the entire client is in a queue. When the communication of the previous client is finished, the thread will be freed and the server can build the communication with another client. Usually the communication is fast and the user can hardly feel the waiting time in the queue. However is the communication is blocked no clients in the queue can build the communication with the server until the current client is removed.

To avoid such problem we use multi-thread method to build the server. The server keeps search for the connection requirement. If there is a requirement, the server will start a new thread to communicate with the client while the main thread is still searching for the requirement.

The logic diagram of the algorithm is as follows:

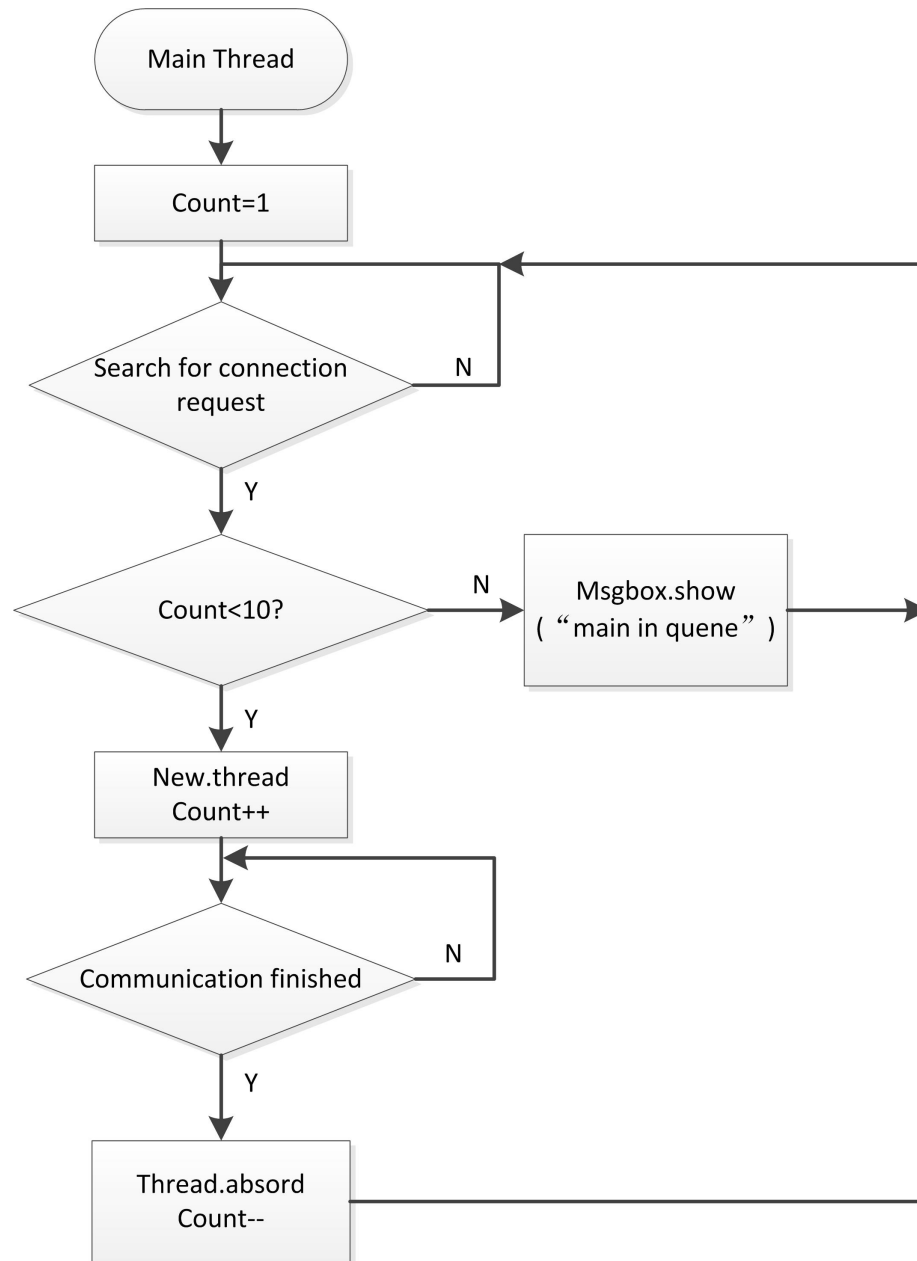


Figure 51: logic diagram of algorithm

## 11 Hardware Development

Hardware and controllers are an essential part of the projects that are expected to bring about some electrical or mechanical changes in its environment. To bring about any change in the physical world, the computer that runs a program can only send out signals in the form of electrical pulses. These pulses are of no use unless they are properly coupled to a well-developed hardware system that can perform the physical changes to realize what the software intended to do.

In the Voice Controlled Home Automation System, users requirements of turning on/off a light or changing the volume of a music player cannot be satisfied until the server is coupled with the hardware to perform the actual actions. For this purpose we interface a number of micro-controllers, relays, sensor, networking techniques and communication equipment with the server.

The hardware implementation is not just about connecting the wires and the components together; it forms the basis of many user case and system requirements. It requires proper and precision designing that is done while keeping in mind not just the users requirements but also the aspects of the system performance ranging from the systems power-performance ratio, cost of implementation, system usability, system reliability and system flexibility. There is a lot of programming involved in developing the hardware to run the controllers efficiently and to maintain the connectivity between it and all the mechanical devices it is in contact with.

This section of the project has been developed as a flexible module which has the ability of working as a stand alone module that can realize the basic requirement of the project that is HOME AUTOMATION, or can be in-cooperated with the server and google-voice to work as a VOICE CONTROLLED HOME AUTOMATION.

It is a complex module that has been developed in a way that looks very simple to the user while operating the device. This module as per the user is just a connection between them and the device (for example a light). We have made the operation of the device look as simple as turning on the switch and observing the bulb to find out how brightly is it glowing. But the main challenge was to develop and program each protocol for each node and controller attached to the node.

- The two main components which form the backbone of the hardware development are the two micro-controller-units (MCU) used:
  1. ARDUINO Mega-series
  2. PIC 18F4520 Micro-controller
  3. ATtiny-85 (Due to their prominent role in this section, it would be worthwhile to read some technical specifications of them.)
- The other main concepts that govern the smooth working of the hardware



system and hence the system as a whole are:

1. Communication mode and equipment
2. Networking mode

Lets start with discussing the various parts involved.

## 11.1 ARDUINO

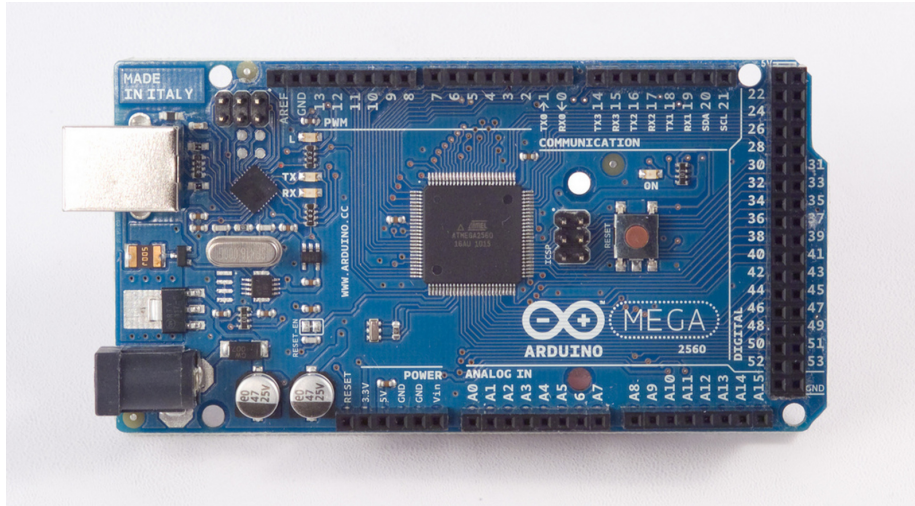


Figure 52: Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, MaxMSP).

**Arduino Mega 2560** The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller. The important feature that is being utilized in this project is that it has four Serial RX and TX Ports. Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage(recommended)	7-12V
Input Voltage(limits)	6-20V
Digital I/o Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8KB
EEPROM	4KB
Clock Speed	16 MHz

### Arduino Mega 2560's Specifications

**Communication** The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega16U2 on the board channels one of these over USB and provides a virtual com port to software on the computer. The Arduino software includes a serial monitor that allows simple textual data to be sent to and from the board.

**Programming** The Arduino Mega can be programmed with the Arduino software. The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming).

## 11.2 ATtiny-85

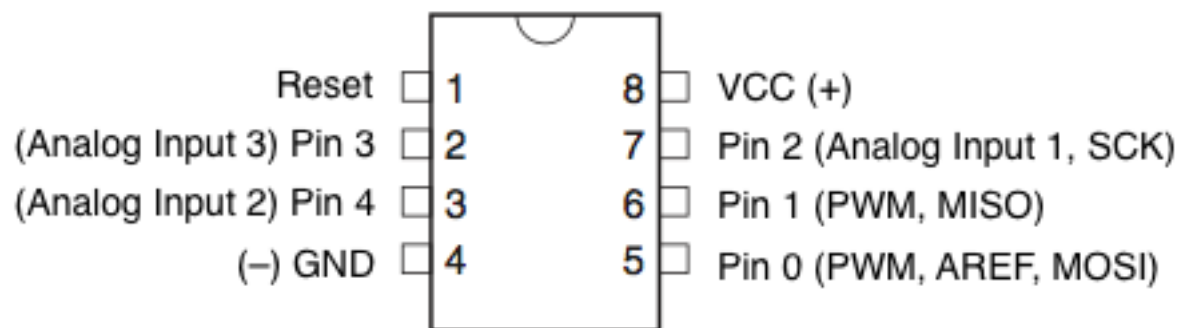
The high-performance, low-power Atmel 8-bit AVR RISC-based microcontroller combines 8KB ISP flash memory, 512B EEPROM, 512-Byte SRAM, 6 general purpose I/O lines, 32 general purpose working registers, one 8-bit timer/counter with compare modes, one 8-bit high speed timer/counter, USI, internal and external Interrupts, 4-channel 10-bit A/D converter, programmable watchdog timer with internal oscillator, three software selectable power saving modes, and debugWIRE for on-chip debugging. The device achieves a throughput of 20 MIPS at 20 MHz and operates between 2.7-5.5 volts.

By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

ATTiny-85 is the best choice for this project because of its compatibility with arduino. This chip is used as a replacement for PIC controller while using IR communication.

Parameter	Value
Flash (Kbytes)	8 Kbytes
Pin Count	8
Max. Operating Frequency	20 MHz
CPU	8-bit AVR
# of Touch Channels	3
max I/O Pins	6
Ext Interrupts	6

### ATtiny45 / ATtiny85



### 11.3 Part List detail and their role

**Infrared Transmitter:** This component is a LED (Light Emitting Diode), which emits light in the Infrared domain at 700 nanometers (nm) to 1millimeter(mm), which is below the visible spectrum. This comprises of one of the components that is responsible for sending communication signals between the two controllers (i.e. Arduino and PIC 18F4520 Microcontroller). It comprises of two pins, one is connected to the ground and the other to the output pin of the controller through a resistor. It emits when the output pin of the controller gives out a high pulse. We modulate the signal sent over this transmitter to omit the problem of interference and to utilize the maximum performance of the LED by passing maximum power through it for a small amount of time to avoid it from burning.

In this project an Infrared Emitters High Speed Emitter that operates at 5V, and requires 160mW of power to emit an Infrared signal at 940nm wavelength at 22Degree is used. Emitters emitting at 940nm are preferred because the receiver is most sensitive at this wavelength.

Infrared transmitter is used to make system as cost effective as possible, and because of the low cost of Infrared components, we can hook them to many devices to automate those devices. This system has been designed in a way to revolutionize home automation by providing maximum usability at a low cost. Another benefit of using Infrared communication is that the power consumption is minimum that is an important concern in case of the microcontroller attached to the devices (like door lock) that are powered by batteries.

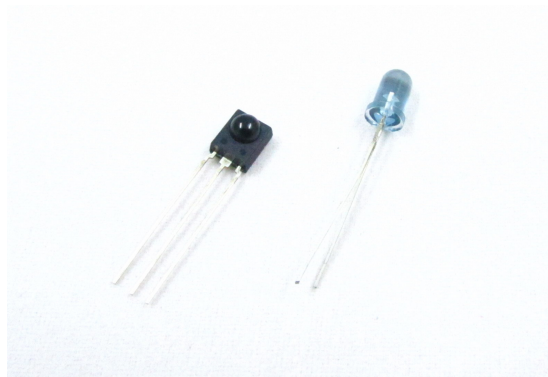


Figure 53: IR receiver and IR transmitter

**IR-Phototransistor:** This device is used as an infrared receiver to capture the transmitted signal, demodulate it, and passes it to the microcontroller. It acts like a normal transistor switching on and off depending upon the intensity of IR light being shined at it. As seen in figure X, The received IR signal is picked up by the IR detection diode on the left side of the diagram. This signal is amplified and limited by the first 2 stages. The limiter acts as an AGC circuit to get a constant pulse level, regardless of the distance to the handset. As you can see only the AC signal is sent to the Band Pass Filter. The Band Pass Filter is tuned to the modulation frequency of the handset unit. Common frequencies range from 30kHz to 60kHz in consumer electronics. The next stages are a detector, integrator and comparator. The purpose of these three blocks is to detect the presence of the modulation frequency. If this modulation frequency is present the output of the comparator will be pulled low, which can be detected by the controller to which it is attached.

For this application, we use an IR receiver that works for carrier frequency of 38KHz and can detect from a distance of 30 meters at a 45 degree view angle. Its operating voltage is 2.7 -5.5 volts and output current is 5 mA which can be easily coupled to the input pins of the controllers.

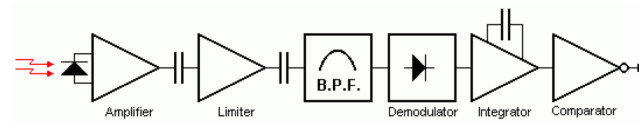
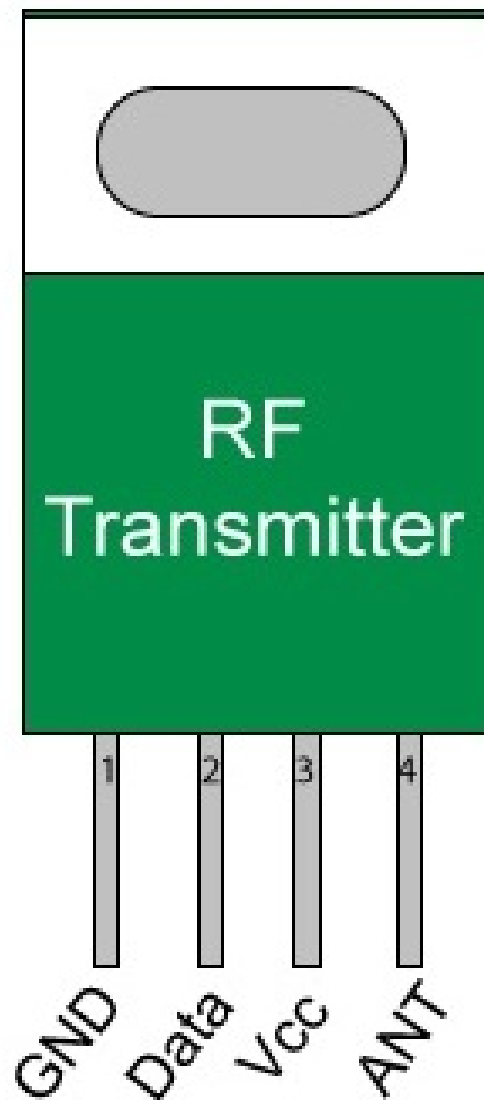


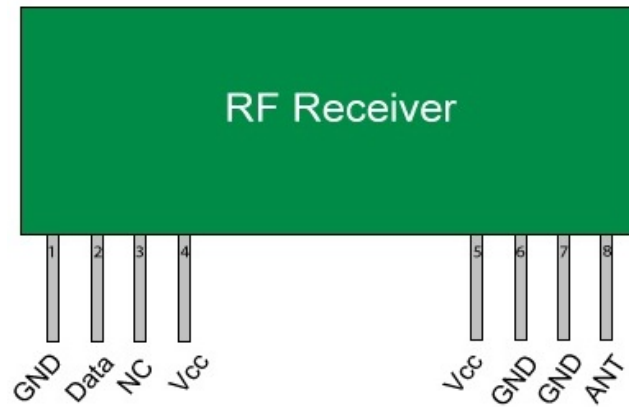
Figure 54: IR receiver(internal view)

**RF 434MHz transmitter Module:** The RF module, as the name suggests, operates at Radio Frequency. The corresponding frequency range varies between 30 kHz and 300 GHz. In this RF system, the digital data is represented as variations in the amplitude of carrier wave. This kind of modulation is known as Amplitude Shift Keying (ASK). It is a small 4 pin module that requires only power, ground and data. It is small enough to fit into any project and add wireless communication. These modules have up to 500 ft range in open space. The transmitter operates from 2-12V. Higher the Voltage, greater is the range of the transmitter. This is an upgrade from IR communication, as we dont need line of sight communication in this and the range is greater. An RF transmitter receives serial data and transmits it wirelessly through RF through its antenna connected at pin4. The transmission occurs at the rate of 1Kbps - 10Kbps.



**RF 434MHz receiver Module:** The rf interface receiver module is larger for demodulation and amplification circuitry. It also has a simple array of power, ground and data pins to connect up with the PIC's USART. The transmitted data is received by an RF receiver operating at the same frequency as that of the transmitter. These wireless receivers work with our 434MHz transmitters. They can easily fit into a breadboard and work well with microcontrollers to create a very simple wireless data link. Since these are only receivers, they will only work communicating data one-way, we would need two pairs (of different frequencies)

to act as a transmitter/receiver pair. These modules are indiscriminate and will receive a fair amount of noise. Both the transmitter and receiver work at common frequencies and don't have IDs. Therefore, a method of filtering this noise and pairing transmitter and receiver will be necessary.



**Relay and Manual-switch in a two-way switch configuration:** Relay is an electro-magnetic device that acts like a switch. The microcontroller can switch On/Off any device by controlling the relay. A two-way toggle switch is used as a manual switch. In short, relay is the switch whose state can be changed by the controller (by the voice command of the user) and the manual switch is the switch whose state can be changed physically by the user. The systems requirement was to have a system in which the user had the option to choose between switching On/Off the light either manually or by using the voice command. To realize this requirement, the hardware was designed in such a parallel and self-adapting way that will allow the use of a device manually and by using voice control at the same time without inducing any problem in the accessibility. The device is made to be self-adaptive by making the controller sense the users manual input and thereby alter its switching method as per the system requirement (ST-16).



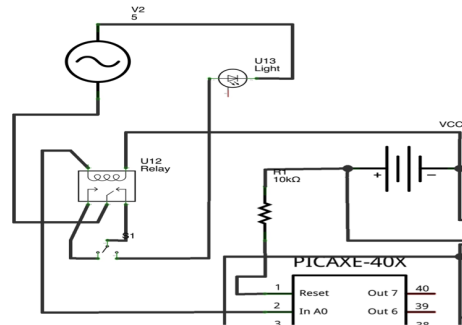


Figure 55: Smart switch

**LDR (Light Dependent Resistanc):** This device is used as a sensor. The special feature of this sensor is that the resistance of this sensor is inversely proportional to the light falling on it. We will use this device to get a real-time feedback of the luminance of the light or to determine if the light is on/off. If the light is ON, the LDR will offer minimum resistance and the voltage drop across the pin of the microcontroller would be maximum. In the case when the light is OFF, the voltage drop across the pin of the microcontroller would be minimum, this way the microcontroller can sense if the light is ON/Off. This not only gives the user a real-time status of the element (light) but also informs the user in case the light fails to respond in the desired manner. It realizes the following system requirements.(ST1,ST6,ST-10 ST-17)

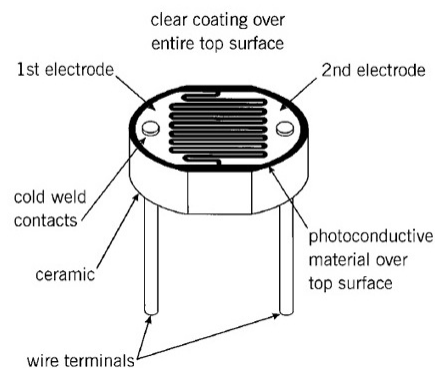


Figure 56: Light Dependent Resistor

**Electric Microphone:** is an acoustic-to-electric transducers or sensors that converts sound into an electrical signal. Higher the loudness of the sound, greater is the electric signal produced by the microphone. This property of the

microphone is used in the project to get a real-time feedback of the music player. By using the mic, the controller will be updated in real-time, with the volume of the music player. This information can be used to send as an feedback to the user.

The microphone is attached to the analog pin of the ATtiny-85 and gives a varied analog input corresponding to the ambient sounds. It realizes the following system requirements. (ST1,ST6,ST-10 ST-17)

**Push Button:** The push button is used as a mechanical sensor. This famous sensor is a button that sends a signal when pressed and does not send when not pressed. This device can be used to get a real-time feedback if the door is locked or not. This feature was not present in the original system requirements, but adding more us abilities to it show that the system is flexible and accepts future expansions and alterations in the system.

**Arduino Wi-Fi Shield:** The Arduino Wi-Fi Shield connects the Arduino to the Internet using 802.11 wireless specification (Wi-Fi). With this the arduino can be given commands over the Internet and the developer can have an access to it remotely.

Using this device helps us achieve the System requirement (18, 19) Supportability is realized as in case of an issue, the technician can remotely solve the problem and keep a log of it in his database. This can also be helpful in increasing the security of the system, in case a thief or malicious user interferes with the system, the system can inform the authorities in charge.

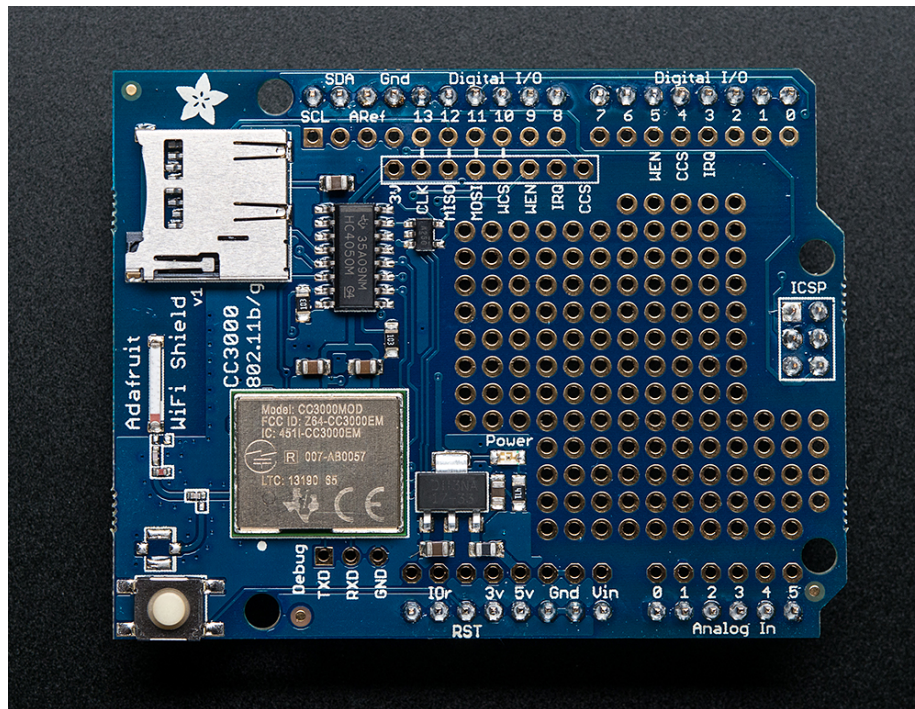


Figure 57: Arduino WIFI shield

**16 MHz Crystal:** This is the external clock of the PIC micro-controller. The controller can transmit data over this frequency. The reason for choosing a 16 MHz crystal oscillator is that arduino operates at 16MHz, for compatibility purpose we needed a microcontroller to operate at the same frequency.

## 11.4 Integrating the Hardware

**Star Network Topology** The Star Network Topology is used to connect the server to each of the smart elements controlled by the micro-controllers. The arduino is used as a central HUB that acts as a conduit to transmit messages. This consists of a central node, to which all other nodes (smart elements) are connected; this central node provides a common connection point for all nodes through a hub (arduino). In star topology, every node (Smart element) is connected to a central node called a hub (arduino). The switch is the server and the peripherals are the clients.

The only alteration made to the star topology is that arduino is an active hub and directs the feedback from each node (smart element) to the server instead of broadcasting it. The arduino only broadcasts the message from the server and directs the message towards all the nodes.

The star network topology is adopted in Voice control home automation for its advantages which are:

1. Better performance: star topology prevents the passing of data packets through an excessive number of nodes. At most, 3 devices and 2 links are involved in any communication between any two devices. Although this topology places a huge overhead on the central hub, with adequate capacity, the hub can handle very high utilization by one device without affecting others.
2. Isolation of devices: Each device is inherently isolated by the link that connects it to the hub. This makes the isolation of individual devices straightforward and amounts to disconnecting each device from the others. This isolation also prevents any non-centralized failure from affecting the network.
3. Benefits from centralization: As the central hub is the bottleneck, increasing its capacity, or connecting additional devices to it, increases the size of the network very easily. Centralization also allows the inspection of traffic through the network. This facilitates analysis of the traffic and detection of suspicious behavior.
4. Easy to detect faults and to remove parts.
5. No disruptions to the network when connecting or removing devices.
6. Installation and configuration is easy since every one device only requires a link and one input/output port to connect it to any other device(s).

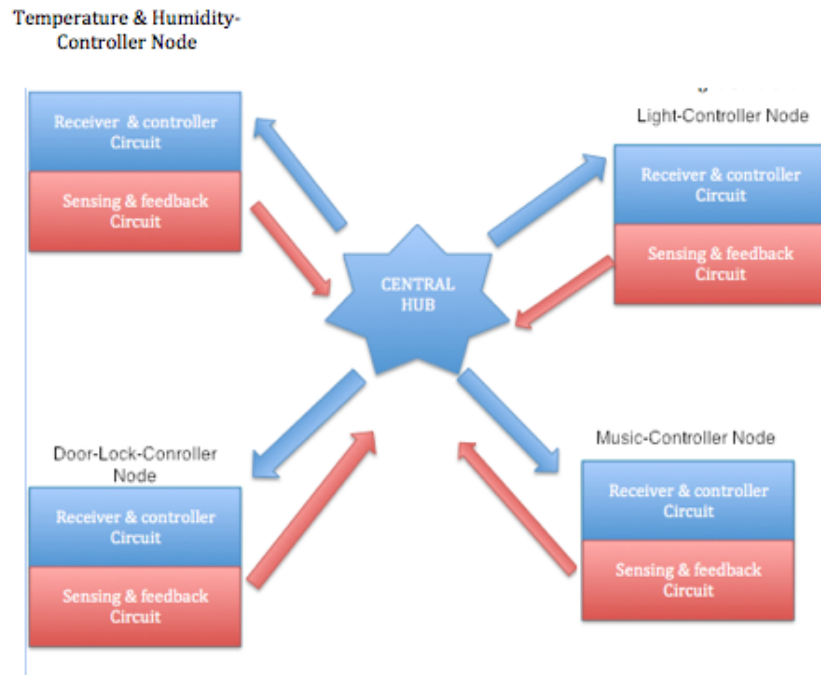


Figure 58: Diagram of our module connected in STAR

Using the STAR Topology as the basis of our connectivity, we can integrate the Components mentioned above in the form of nodes and assign each node some integrated responsibilities.

This module has been divided into 1 hub connected to 2 Basic nodes for the time being and has the capability of adding 20 more nodes without having to change any design or protocol. Each of the Nodes has two micro-controllers that have been programmed with different codes to achieve different usability from a similar design.

The two Basic nodes used are:

1. Light-Controller Node
2. Music-Controller Node

The other proposed nodes that will be added are:

1. Door-Lock-Controller Node
2. Temperature and Humidity-Controller Node

## 11.5 Central Hub

It is the main part of the star topology network formed in this module.

This is the part that accepts the users command either directly inputted to this node or through the inputs provided to the server connected to it.

This part is connected to all the nodes in this module and forms a connection between the hardware and the software. The central Hub even provides a UMI(User machine Interface) to make module independent enough to realize the basic requirements of the project.

The central HUB consists of:

1. Arduino Mega which is the main controller acting like the hub in the Star-Topology.
2. 344 MHz RF Transmitter
3. 315 MHz RF receiver

We use one frequency to transmit the data and data is received over a different frequency. This use of different frequency gives us the freedom to develop a duplex system that can communicate both ways without bothering about the interference.

We have used basic Radio Frequency transmitters and Receivers instead of XBee (commercial open source RF transceiver) so that we could implement our own protocol for communication between the nodes. Not using XBee (commercial open source RF transceiver) also reduces the cost of each node.

Figure below shows Arduino Mega being used as a Central-HUB

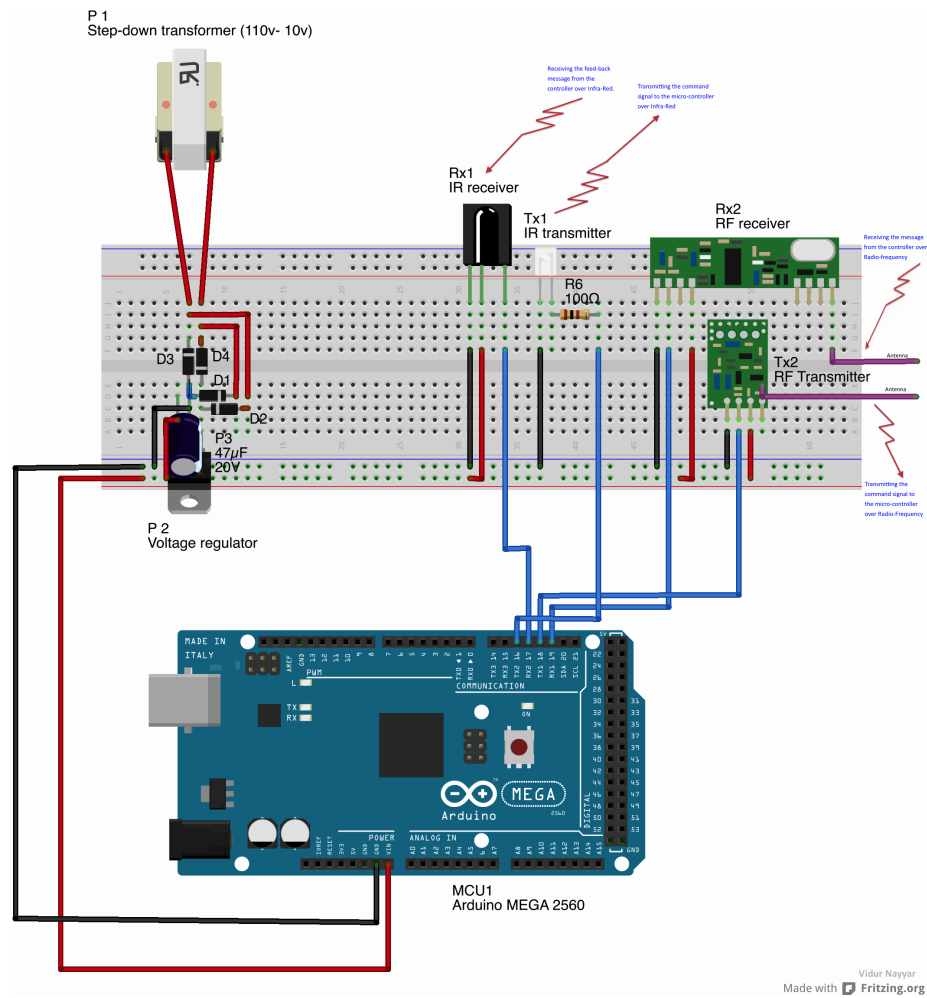


Figure 59: Board Diagram of Arduino as a Tranceiver HUB

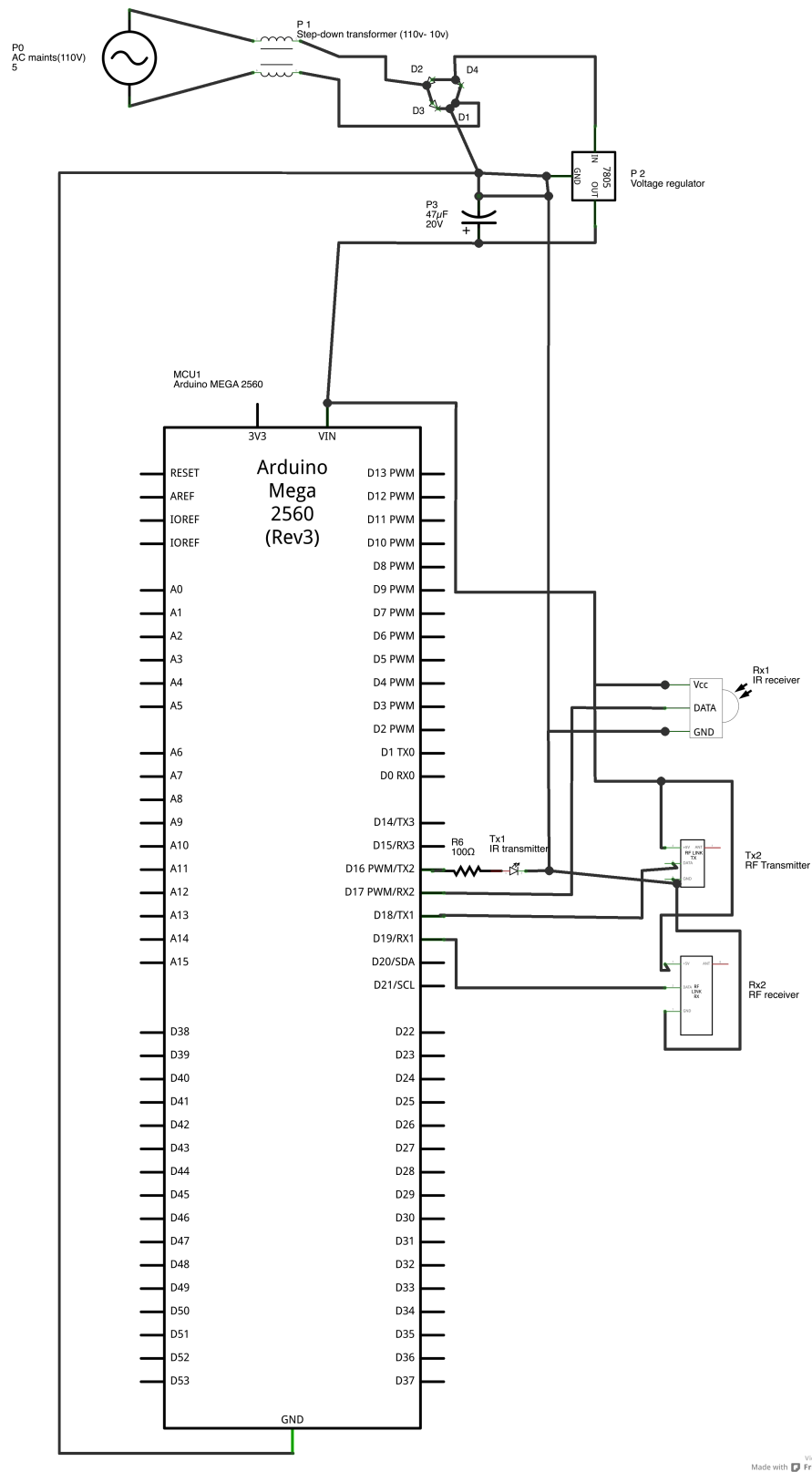


Figure 60: Schematic Diagram of Arduino as a Tranceiver HUB



## 11.6 Light-Controller Node

This is the node that actually alters the brightness of the light to which it is attached. It is connected wirelessly to the central HUB and waits for a command from it over Radio frequency. It is in perfect synchronization with the central HUB once the signal is received. This synchronization is achieved by using handshake mechanism (explained at the end).

It further divided into two parts:

1. Receiver and Controller circuit.
2. Sensing and Feedback circuit.

The Receiver and Controller circuit is the part that contains the RF receiver and waits for the command from the Central HUB. It then decodes the command and changes the brightness of the light depending on the command.

The Sensing and Feedback circuit is connected to the Receiver and Controller circuit as a slave and only activates when the Receiver and Controller circuit send a high pulse to one of its pins. This circuit then checks the brightness of the light from the LDR (Light Dependent Resistor) connected to it and provides a feedback back to the Central HUB through the RF Transmitter connected to it.

This arrangement adds a feature of letting the user know in real time if the command he/she gave has been actually been done. This feature further makes problem solving easy and to isolate the cause of any bug in the system. The Central HUB is programmed to raise an alarm in case no feedback is received after a command has been sent to a node. This alarm is reset only if a feedback is received from the same node.

Figure Below shows ATtiny 85 being used as a Light-Controller Node

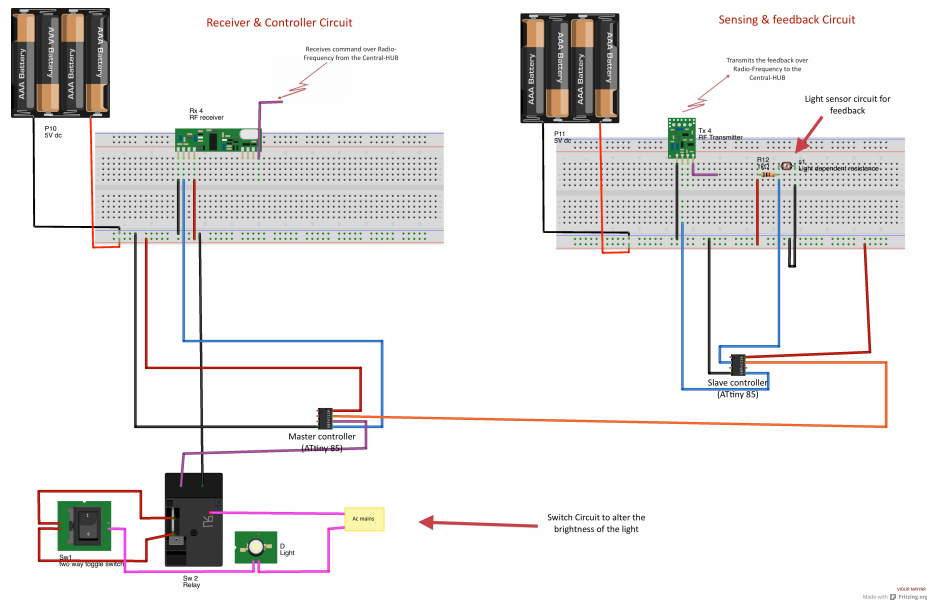


Figure 61: Bread-Board Diagram of ATtiny 85 as a Light-Controller Node

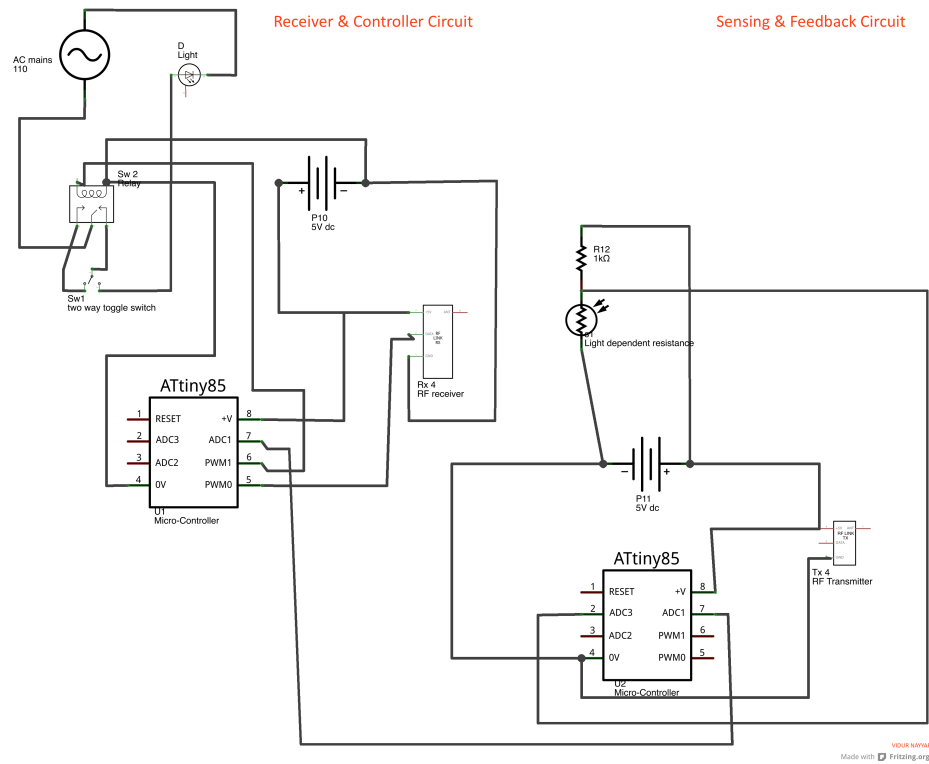


Figure 62: Schematic Diagram of ATtiny 85 as a Light-Controller Node

## 11.7 Music-Controller Node

This is the node that actually controls all the functionality of a music player that can be realized by the remote controller associated to that music player. It is connected wirelessly to the central HUB and waits for a command from it over Radio-frequency. It is in perfect synchronization with the central HUB once the signal is received. This synchronization is achieved by using handshake mechanism (explained at the end).

It has mostly same functionality as that of the Light-Controller Node except for the fact that:

1. The Receiver and Controller circuit of this node has a Infra Red led which can convert the user's command into Infra-red signal which performs various functions on the music player.
2. The Sensing and Feedback circuit is connected to a microphone and amplifier circuit designed and programmed by me to send a feedback with just 20% error in detecting the correct volume at which the player is playing. (Results from initial analysis and tests. Awaiting results of further tests)

Figure Below shows ATtiny 85 being used as a Music-Controller Node

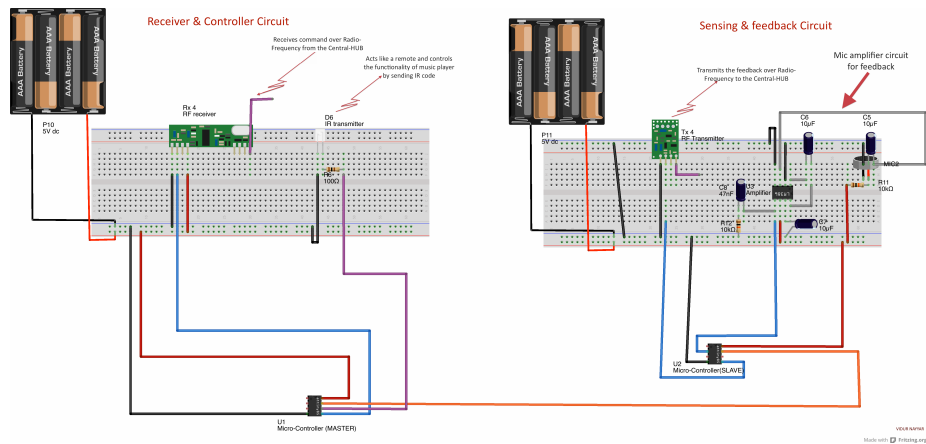


Figure 63: ATtiny-85 as a Music-Controller Node using a mic for feedback

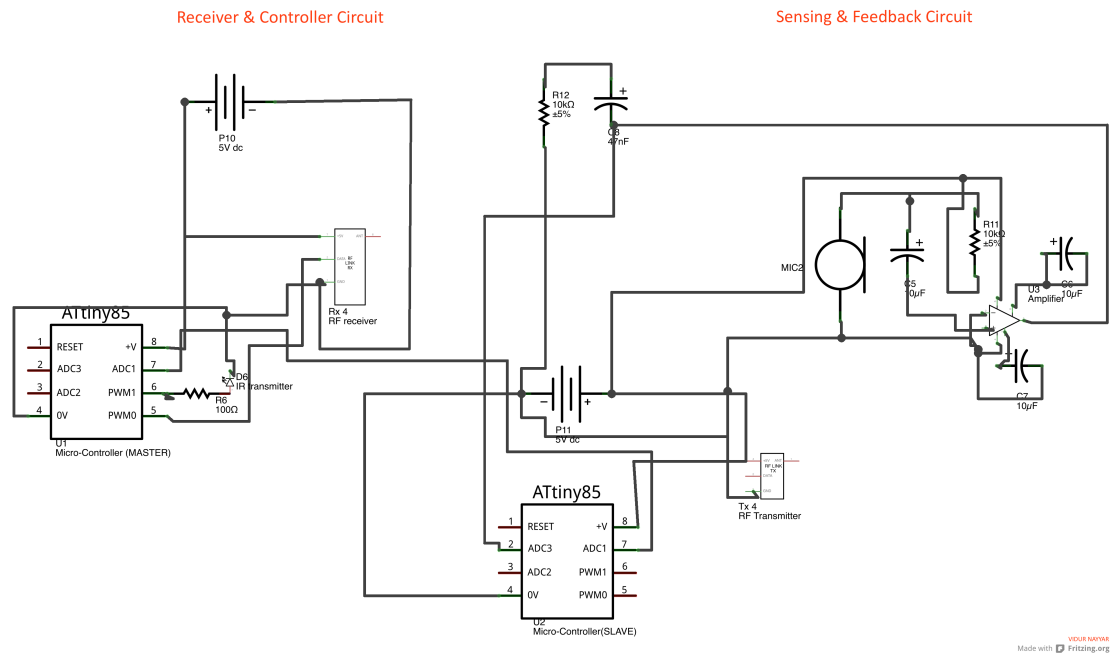


Figure 64: Schematic diagram of ATtiny-85 as a Music-Controller Node using a mic for feedback

## 11.8 Communication between the hardware

The communication between electronic devices is all about pulses of energy. Serial Communication involves changing the voltage of an electrical connection between the sender and receiver at a specific rate. Each interval of time represents one bit of information. The sender changes the voltage to send a value of 0 or 1 for the bit in question, and the receiver reads whether the voltage is high or low. There are two methods that the sender and receiver can use to agree on the rate at which bits are sent. In asynchronous serial communication, the rate is agreed upon mutually and clocked independently by sender and receiver. In synchronous serial communication, the sender pulses a separate connection high and low at a steady rate, controls it.

There are two common types of wireless communication in most peoples lives: infrared light communication and radio communication. The main difference between them, from a user or developers position, is their directionality. There are three types of devices common to both IR and RF systems: transmitters, which send a signal but cant receive one; receivers, which receive a signal but cant send one; and transceivers, which can do both. In this project we use a transmitter and receiver pair than a transceiver to save ourselves from the complexity of interference in transceivers. In the Voice based home automation system the transmitter keeps on sending the message till the time its receiver receives a feedback that the message has been sent. So we can say that the controller intelligently switches between the transmitter and receiver.

## 11.9 Working of Infrared Communication

IR communication works by pulsing an IR LED at a set data rate, and receiving the pulses using an IR photodiode. Its simply serial communication transmitted using infrared light. Since there are many everyday sources of IR light (the sun, incandescent light bulbs, any heat source), its necessary to differentiate the IR data signal from other IR energy. To do this, the serial output is sent to an oscillator before its sent to the output LED. The wave created by the oscillator, called a carrier wave, is a regular pulse thats modulated by the pulses of the data signal.

The receiver picks up all IR light but filters out anything thats not vibrating at the carrier frequency. Then it filters out the carrier frequency, so all thats left is the data signal. This method transmits data using infrared light without getting interference from other IR light sources.

The directional nature of infrared makes it more limited, but we use it in our project because it is cheaper than radio, and requires less power. As radios get cheaper, more power-efficient, and more robust, its less common to see an IR port on a computer. However, its still both cost-effective and power-efficient for line-of- sight remote control applications. In Home automation, power consumption parameter is important because the smart elements that are placed remotely operate on batteries.

### 11.10 Working of Radio-Frequency Communication

Radio relies on the electrical property called induction. Any time you vary the electrical current in a wire, you generate a corresponding magnetic field that emanates from the wire. This changing magnetic field induces an electrical current in any other wires in the field. The frequency of the magnetic field is the same as the frequency of the current in the original wire. This means that if you want to send a signal without a wire, you can generate a changing current in one wire at a given frequency, and attach a circuit to the second wire to detect current changes at that frequency. That's how radio works.

The distance that you can transmit a radio signal depends on the signal strength, the sensitivity of the receiver, the nature of the antennas, and any obstacles that block the signal. The stronger the original current and the more sensitive the receiver, the farther apart the sender and receiver can be. The two wires act as antennas. Any conductor can be an antenna. The length and shape of the antenna and the frequency of the signal all affect transmission. For a straight wire antenna is as follows: Antenna length =  $5,616 \text{ in.} / \text{frequency in MHz} = 14,266.06 \text{ cm.} / \text{frequency in MHz}$ .

In Voice controlled home automation we use digital RF COMMUNICATION. Digital radios superimpose digital signals on the carrier wave, so there must be a digital device on either end to encode or decode those signals. In other words, digital radios are basically modems, converting digital data to radio signals, and radio signals back into digital data. Though the power consumption in RF communication is more than in IR communication, but the increased range and omnidirectional property of Radio waves makes it more desirable.

### 11.11 Multiplexing, Modulation, Algorithms and Protocols

While transmitting via Radio Frequency or Infrared, anyone with a compatible receiver can receive your signal (in line of sight for IR communication). There's no wire to contain the signal, so if two transmitters are sending at the same time, they will interfere with each other. This is the biggest weakness of radio frequency and Infrared communication: a given receiver has no way to know who sent the signal it's receiving.

The only way to avoid such interference is to (modulate the pulse in case of infrared communication) multiplex the signal in case of Radio frequency communication. Sharing in radio communication is called multiplexing, and this form of sharing is called time division multiplexing. Each transmitter gets a given time slot in which to transmit. Multiplexing helps transmission by arranging for transmitters to take turns and to distinguish them based on frequency, but it doesn't concern itself with the content of what's being said. To make sure the message is clear, it's common to use a data protocol on top of using multiplexing. A number of protocols have been used to make the home automation system secure and error free.

**Handshake protocol:** A protocol similar to the handshaking protocol is adopted for the communication between arduino and the other controllers attached to make the elements of the house, smart elements. As-per the protocol that we have incorporated, when the user sends a command message to the arduino. The arduino relays the message to the controller either via RF or IR mode of communication, and keeps on sending the message till the controller replies with a feedback message. Once the feedback is received by the arduino, it needs to send an acknowledge message to the controller. The controller keeps on sending the feedback message till the time the arduino replies the feedback with an acknowledge message. This way the communication is reliable and in case of an error, the problem can be isolated.

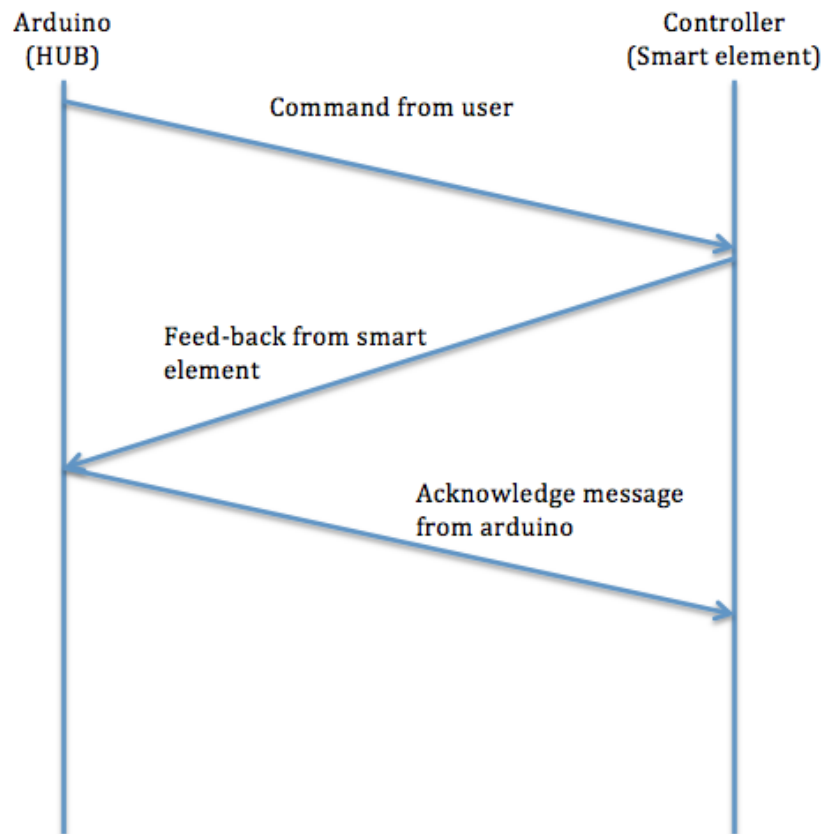


Figure 65: Hand-shake protocol

**Checksum algorithm:** The message bits communicated over Infrared and Radio frequency have to go through interference and noise, which may mix some noise or error bits in the original message. To detect such error bits and



reject the message with such errors, we incorporate an algorithm as a part of our protocol. According to this protocol, every time a message is received by the receiving circuit of the controller. The controller must check the message using the checksum algorithm and only proceed towards decoding the received message if the checksum algorithm is satisfied.

According to this algorithm, the message bits are divided into 3 parts, header, data and tail. The header contains the address information of the receptor, the data contains the information of the action that needs to be performed and the tail contains some bits used to check is the message is correct or if it has some errors in it. What the controller does to implement this algorithm is that it computes the sum of the timing of all the high bits in the message and compares it to the sum of timings of all the high bits in the tail. For the message to be considered error free and decoded, the sum of timing of high bits in the tail and the sum of timing of high bits in the tail should be equal. If they are not equal, it means it has error and the message is discarded and receptor waits for the next message.

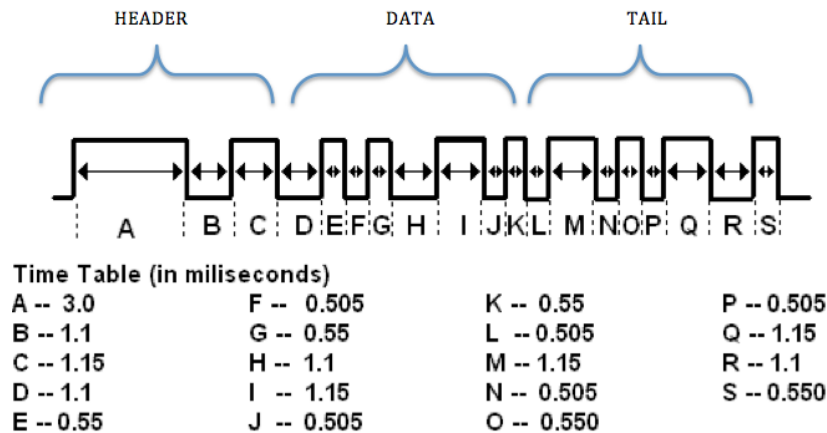


Figure 66: checksum algorithm

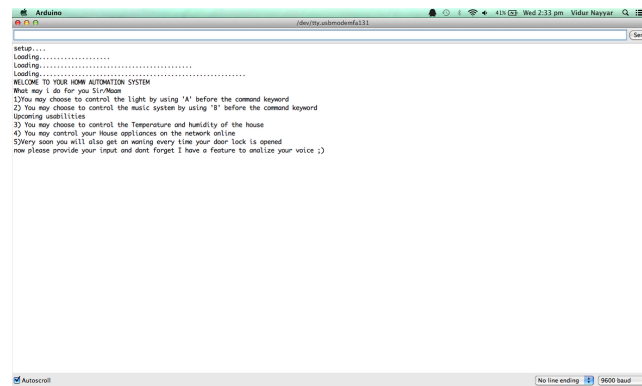
**Pulse-window algorithm:** During the communication test, it was seen that the noise signal was either too long a pulse or too short. So we defined another algorithm to be a part of the communication protocol. According to the Pulse-window algorithm, a higher limit and a lower limit of the accepted pulse is predefined. This way the receiver does not waste time in decoding or even checking the message for error in case it is not in the predefined range. This increases the efficiency of the receiver and saves it from overburden and errors. A high-low pair in the message is defined as a pulse. In our project we accept the message with pulse length from 4 to 12.

**Fuzziness algorithm:** In Infrared and Radio frequency communication there are always some signal errors and timing errors due to interference and timing latency of the controllers. To overcome this situation, and to make our system a little tolerant to errors, we introduce the Fuzziness algorithm. According to this algorithm if the received message has some difference in the pulse high time or pulse low time of any pulse in comparison to the saved pulse timings, it is acceptable till the time it is within the range of a predefined percentage of fuzziness.

Pulse is accepted if the Absolute value of (pulse time - saved pulse time) is less than  $(\text{fuzziness}/100) \times \text{pulse time}$

## 11.12 User Interface from Hardware

A crude User interface is provided at the hardware level that is capable to realize all the users requirements except for the requirement of controlling by voice command. This User interface can be accessed by typing in the command on a similar window as shown in the figure below. The commands are not so user friendly as they are encoded into simple alphabets that are understood by the controller. For example AA# to turn ON the light at 100% brightness and AB# to turn OFF the light. This way we can bypass the server and the mobile user interface and take control of the functionality of our house.



This User interface is also capable of displaying the feedback from the various nodes to which it is connected. For example if we enter the code BF#, it will display the volume at which the music system is playing. The Feed-back and command entered can be seen in the figure below. The developer for debugging and problem detection also can use this User interface. It also gives the user to himself detect the problem by reading the user manual.

```
returned:25%I received:  
AC#Turn light ON at 25%  
comand is:  
2  
ACB  
Got: A A [return_fbAfb1A  
100% brightness/ volume  
25alarm off  
Got: 90 90 The Light is glowing at:  
returned:25%I received:  
AF#Feedback  
comand is:  
5  
AFB  
return_fbAFB  
return_fbAFB  
return_fbAFB  
return_fbAFB  
Got: A B [return_fbBfb1B  
0% brightness/ volume  
23alarm off  
Got: 90 90 The Light is glowing at:  
returned:23%I received:  
AE#Turn light ON at 75%  
comand is:  
4  
AEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
Got: A E [return_fbEfb1E  
69alarm off  
Got: 90 90 The Light is glowing at:  
returned:69%
```

This is the encoded command received from the server

This is after decoding the command

This is the feedback received from the device

Feedback is 23% brightness

Feedback is 69% brightness

☒ Autoscroll

### 11.13 Testing

Bugs are closely linked with programming. The best way to remove these bugs is to perform extensive testing. The testing that has been performed to check the boundaries and quality of this module starts with unit testing and is followed up by integration and system testing.

a. As a part of unit test:

- To test the hardware each component was subjected to varied amount of power to test the deviation in its performance.
- Each component was subjected to random inputs and its outputs were mapped
- Each software module and program was probed with random inputs and its boundary conditions were checked
- After debugging, various tests were repeated to check if the new code has created any new errors.
- Debug points were inserted in the code to check if the code is running as desired.
- Debugging the Latencies in the software to match up to the practical expectations of the user.

b. Integration testing consisted of:

- Check the sent output and compare it with the input received by the other device
- Checking the output of one function and check its behavior when called by different functions and testing the returned value.
- Test the reliability of connection and test the effect of noise and other interference on it
- Keeping track of the Latencies and processing speed of various controllers to achieve timing synchronization between them, as this is a time sensitive system.

1. Test for RF transmitters and receivers (Also covers testing for integration between the components):

This test showed the result that the performance of the receiver and the transmitter were affected a lot and the distance at which they were communicating correctly reduced from 40 feet to just 10 feet. This made us realize that extra power supply needs to be used for the transmitter and receivers for them to communicate properly.

The test also burnt up a couple of ATtiny microcontrollers while using high power signal from the receiver unit to the pin of the controller. So a standard voltage of 5-9 volts is maintained.

Testing the various baud-rate of transmitting and receiving the data gave us an idea of achieving an handoff between high transmission rate and correctness of message reception over a certain distance. As we increased the baud-rate, more errors were incorporated in our reception. We finalized on a baud rate of 2000 bits per second, which let us communicate at a distance of 30-35 feet.

The use of external antenna also helps in increasing the range by 10%.

The image below maps the output of the receiver at close proximity with the transmitter working at 2000 bits per second:

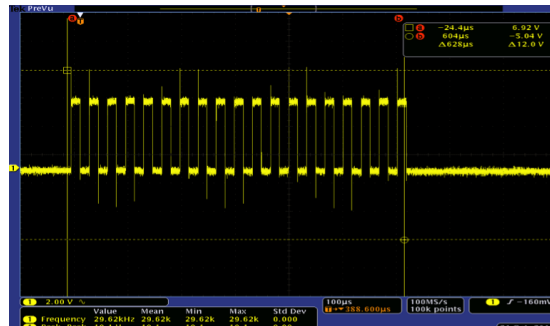


Figure 67: RF signal on the receiver

The image below maps the output of the receiver at a distance greater than 50 feet with the transmitter working at 2000 bits per second:

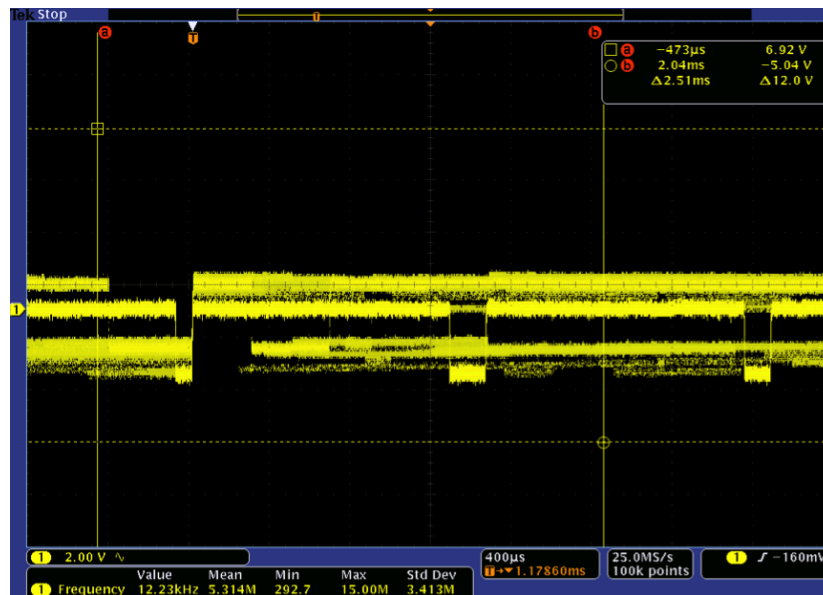


Figure 68: Noise on the RF receiver

2. Next we tested the IR led that is used to control the music player and we were looking into the following aspects:

- Debug points to view capturing timing data
- Counters verify that no transmitted bits were missed
- Debug points in arduino to affirm transfer accuracy arduino Mega to AT-tiny
- Debugging to make Arduino Mega and ATtiny compatible by considering their operating speed and timings.
- Debug points to affirm transfer accuracy between arduino.
- Test the response of arduino is two commands are executed at the same time
- Debugging the Latencies in the software to match up to the practical expectations of the user
- Distance from which the music player can be accurately controlled (not performed coz of lack of hardware)
- We tested the timing an the simulator and the image is below:



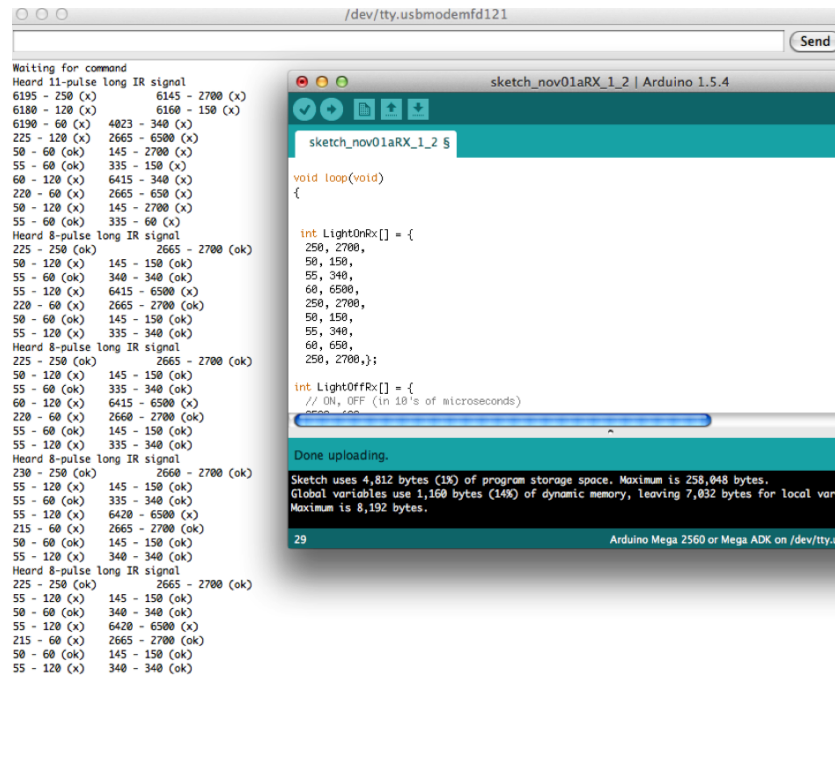


Figure 69: Checked for correctness of IR signal (values on arduino serial port monitor)

3. The Light dependent Resistor (LDR) was tested:

The effect of power variation was checked and the results showed that the Analog to Digital converter in the arduino gives different values over different power. To make the reading more independent, a separate source of power was dedicated to arduino.

The arduino reading over different intensity of light was monitored and recorded which was later compared to the Oscilloscope readings. The results were very accurate and the speed at which the results were updated was impressive (10milli second). The figure below shows the arduino reading of different intensity of light on the LDR. The corresponding figure shows the corresponding Oscilloscope readings.

The test result showed that arduino readings on its analog pin of more than 300 means complete darkness and at maximum brightness the readings were around 20. This enabled us to map these readings as shown in Figure 20. There are more test images in the folder named Test contd..

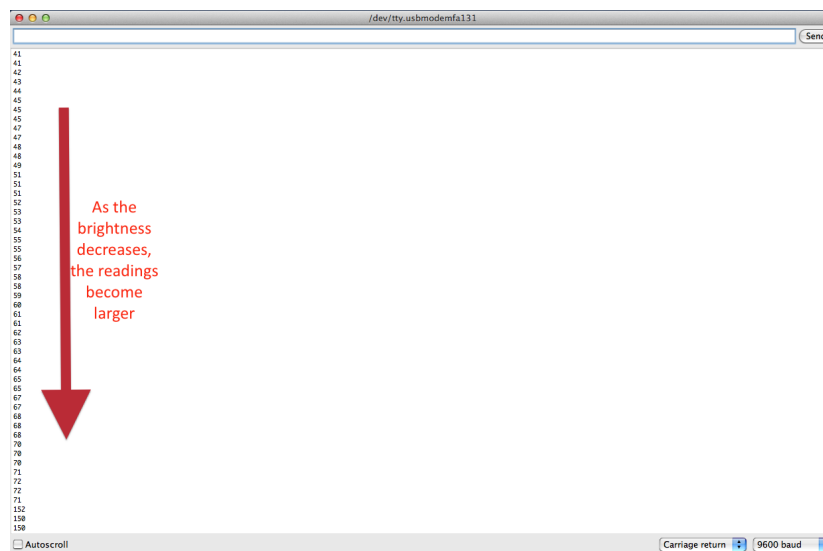


Figure 70: Raw readingsl (values on arduino serial port monitor)

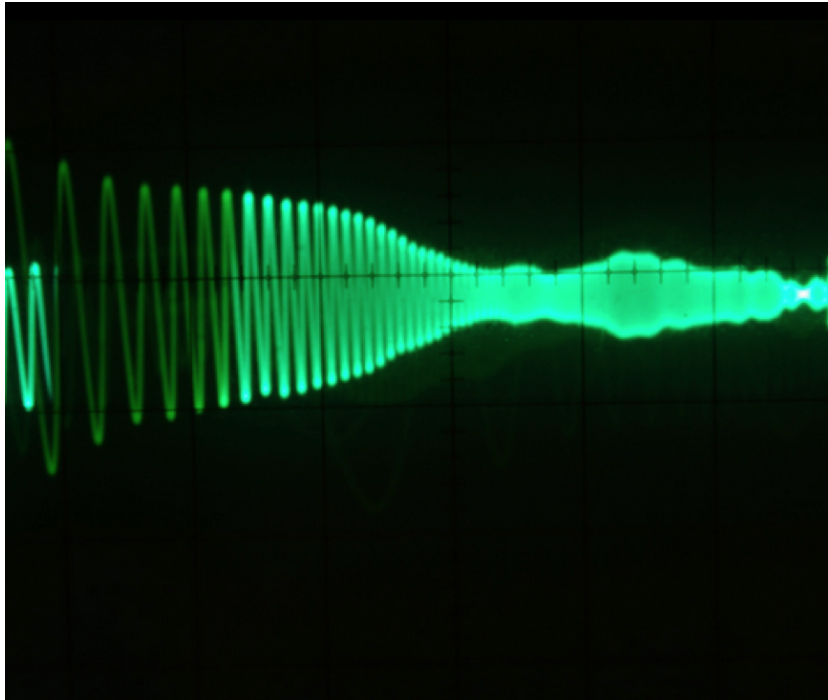


Figure 71: Oscilloscope reading for a dimming light

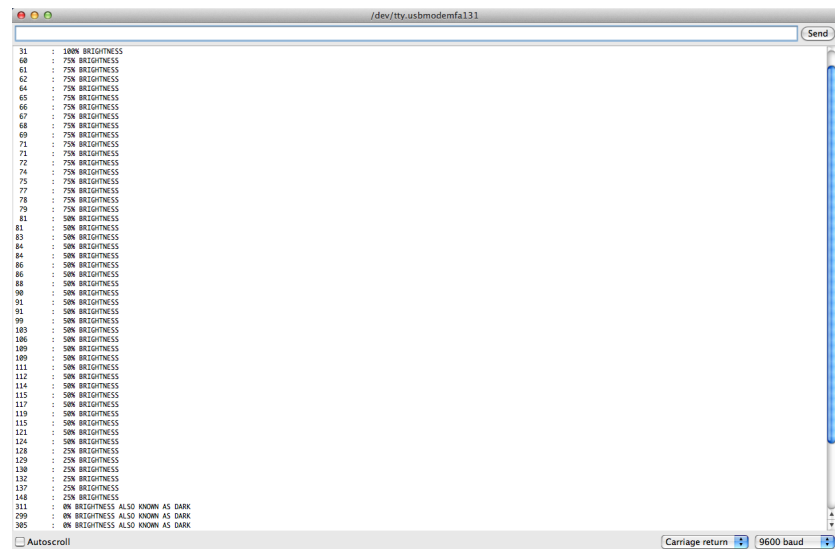


Figure 72: Mapped brightness to readings (values on arduino serial port monitor)

4. The Microphone sensor was tested:

The microphone was extensively tested over a various conditions and using different music. In fact this sensor was developed while testing because it was required to give an accurate for vast range of inputs.

The whole process of testing was a learning process and the results on the oscilloscope were taken as reference and the controller was programmed. The test results indicated that that after a certain volume, the amplitude of the wave becomes constant, but the frequency of encountering waves carrying high amplitude increases with the volume. Taking advantage of this knowledge, we have developed a program to use the root mean square of the average over a certain time period deduced from the tests. This gave us 80% accuracy in our test results.

The two figures below show us the difference in accuracy of the sensor when the volume is gradually increased. the figure below uses a system taking just the RMS value of the readings where as the corresponding figure gives the output depending on our own developed signal-processing algorithms.

Figure shows how we have mapped different waves into volume.

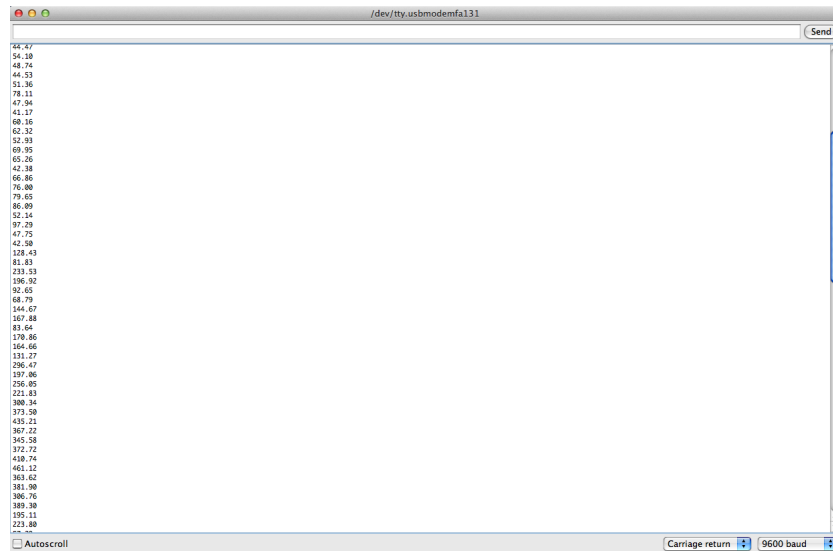


Figure 73: Readings with just RMS (high error)(values on arduino serial port monitor)



Figure 74: Readings after software processing (low error) (values on arduino serial port monitor)

The figures below depicts how the waves are mappend at different volume levels:

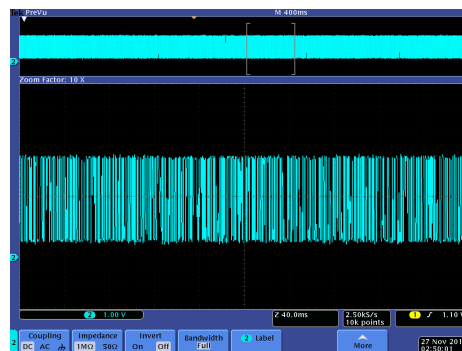


Figure 75: 100% Volume)

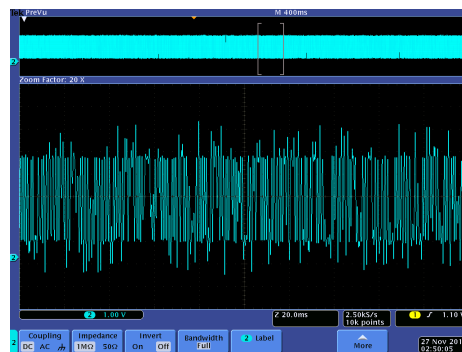


Figure 76: 75% Volume)

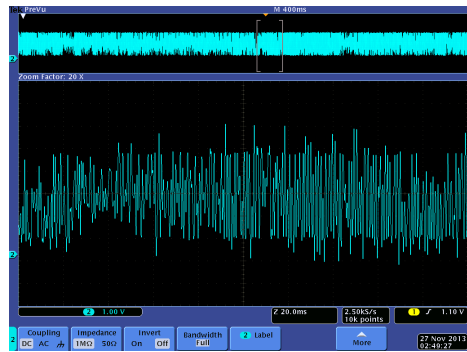


Figure 77: 50% Volume)

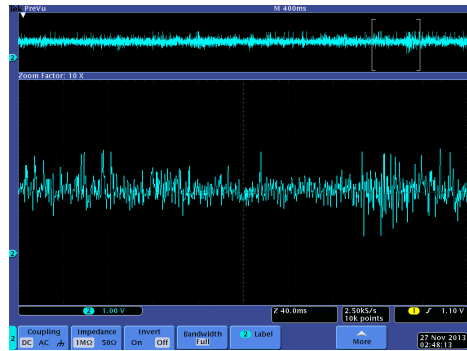


Figure 78: 25% Volume)

5) Testing of the integrity of the software was mainly performed by analyzing the response of the program to different commands. For testing and debugging, after every unit in the program the values of the variables are checked. This is done by dry-running the program and checking it and debugging it using the arduino serial port monitoring.

This helps us to isolate the bugs and record the response of each unit while working alone and also while integrated with the whole system.

All the debugging and Testing of the program for the controller were done using this method. The main advantage of using the serial monitoring port is that we can detect the course of the software at run-time. Figures below show the Serial-port-Monitoring of the program while it is being run. In the first figure, command AA#(to turn on the light at 100% brightness), AB#(to turn off the light) & AE#(to turn on the light at 75% brightness) has been shown. In the corresponding figure, command BA#(to set the volume at 100%) and BB#(to set the volume at zero %), the two boundary cases have been shown.

The third figure shows Serial.print command on arduino ide and the corresponding serial port monitoring for testing to give a better explanation of how the testing is done.

The fourth figure shows how the parts are integrated and tested. It shows the integration testing of central hub and light-controller node



```
returned:25%I received:  
AC#Turn light ON at 25%  
comand is:  
2  
ACB  
Got: A A [return_fbAfb1A  
100% brightness/ volume  
25alarm off  
Got: 90 90 The Light is glowing at:  
returned:25%I received:  
AF#Feedback  
comand is:  
5  
AFB  
return_fbAFB  
return_fbAFB  
return_fbAFB  
return_fbAFB  
Got: A B [return_fbBfb1B  
0% brightness/ volume  
23alarm off  
Got: 90 90 The Light is glowing at:  
returned:23%I received:  
AE#Turn light ON at 75%  
comand is:  
4  
AEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
return_fbAEB  
Got: A E [return_fbEfb1E  
69alarm off  
Got: 90 90 The Light is glowing at:  
returned:69%
```

This is the encoded command received from the server

This is after decoding the command

This is the feedback received from the device

Feedback is 23% brightness

Feedback is 69% brightness

☒ Autoscroll

Figure 79: Commands to change the brightness (values on arduino serial port monitor)

```
off
0 90 The music is playing at:
ad:2%I received:
dback
is:

B 0return_fbBfb1B
ghtness/ volume
off
0 90 The Light is glowing at:
ad:0%I received:
dback
is:

B 0return_fbBfb1B
ghtness/ volume
off
ght is glowing at:
ad:0%I received:
n OFF the music player
is:

_fb0fb10
off
sic is playing at:
ad:2%I received:
rease volume to 100%
is:

B 0return_fbBfb1B
ghtness/ volume
off
5 66 The music is playing at:
ad:0%
scroll
```

Feedback is 2%  
volume

Feedback is 0%  
volume

Figure 80: Commands to change the volume(values on arduino serial port monitor)



## 12 Algorithm

In this section we are going to talk about an algorithm we have used and some perspective algorithms.

### 12.1 The keywords extraction algorithm

As we know a command sentence usually consists of different kind of words. Some of the words in a command are important to the server while other words are not necessary. We use a word filter to extract some of the keywords from the command.

After analyzing the command sentences, we find that usually typical command contains three kinds of keywords, target, action and degree. Usually we use nouns to represent the target, we use verbs for action and we use adjectives or percentages for degrees.

Based on above we have three word pools, for target, action and degrees. Once a new word is input, the server segments every word and compare those words with words in the pool. If the words are same or share the similar meaning the server is able to extract the keywords.

The diagram is as follows:



Figure 83: The algorithm of keywords extraction

### 12.2 The smart control algorithm

Sometimes understanding the commands needs not only the keywords, the priori knowledge is equally important. If the command is simply "turn on the light" which light should be turned on by the system? We know if it is dinnertime the light in the dinner room should be turned on, if it is bedtime the bedroom's light should be turned on.

Based on the huge number of records of user's command history the system is able to conclude the user's habits and based on the user's habits the system can do the right thing based on some very simple command sentences.

As it is mentioned before, the Server is linked to an Access database, which keeps all the action records. Based on the machine learning method we can try to build the algorithm based on user's habit.

We use three most important factors as the key feature in the algorithm: time,

place and date. The main purpose of the algorithm is to sort the user's command and based on the command records to conclude some patterns. For example, based on the user's command records of in the living room we get the following table.

Among all 300 cases		
Actions	Time	Sequence
Turn on the light	6:00am-6:00pm	0
Turn on the light	6:00am-8:00pm	270
... ..		

Figure 84: Table 1 the statistic result of user's habit based on place

From the table above we can see that there is a high possibility that the user will turn on the light in the living room between 6:00pm to 9:00 pm. So if the user's command is turn on the light and the time is between 6:00pm to 9:00pm the system will turn on the light in the living room based on the records.

### 12.3 The reminder algorithm

Based on the user's habit, we are not only able to make the system understand the simple commands in different time, we can even make the server able to predict user's command.

There are some fixed schedule in our daily life like we turn on the light in our bedroom at the bedtime and we turn off the music when we leave home. We are able to calculate the percentage of some specific event. If the percentage of a certain event in a certain time is more than 90% we call it the high possibility event. The system will do these things automatically everyday. Also the list of high possibility will be updated every day to make sure there will be less statistics mistakes.

The classification of the event is in the following chat:

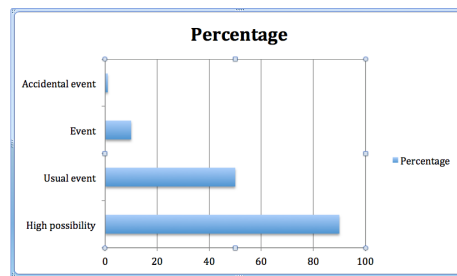


Figure 3 the classification of events

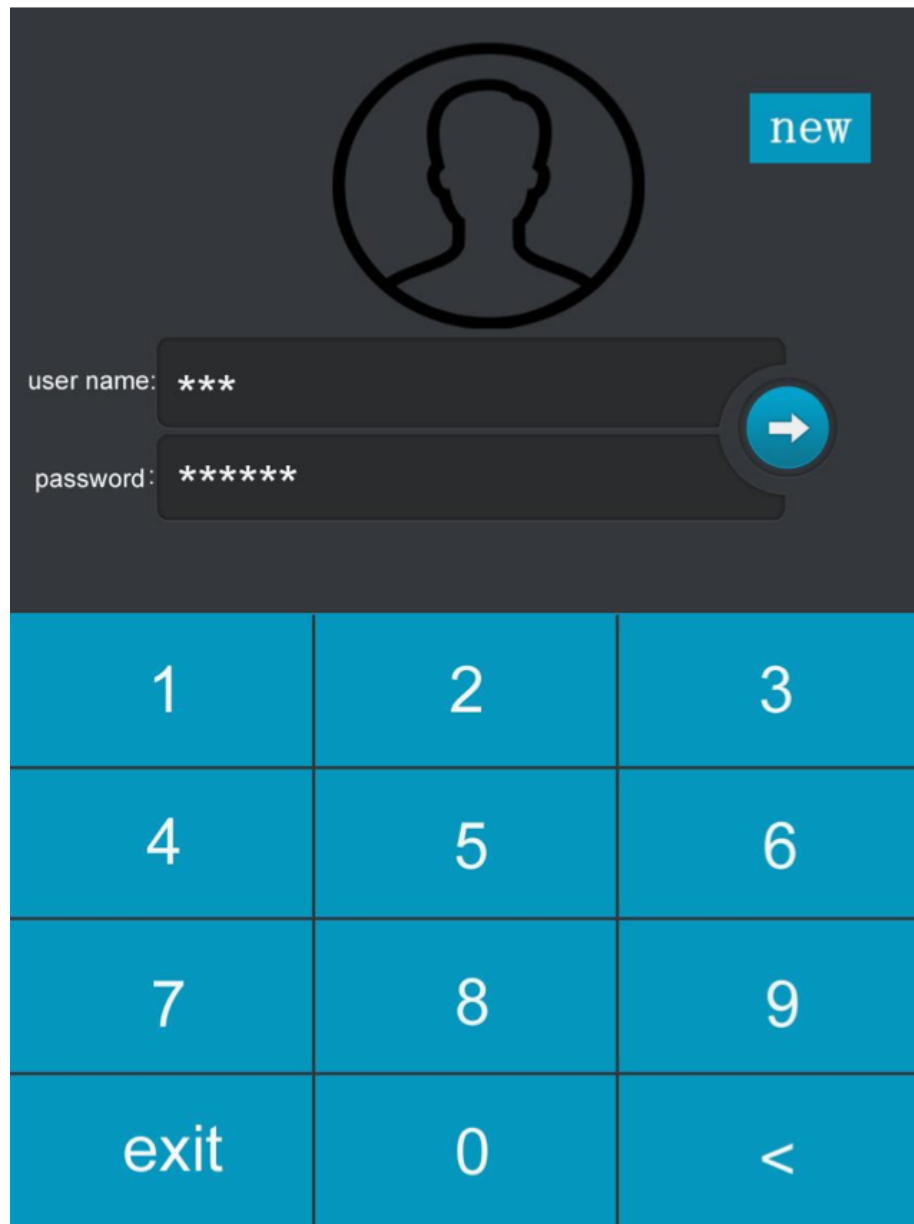
Figure 85: Table 1 the statistic result of user's habit based on place

## 13 User interface design and implementation

In report 1, we have made a lot of effort on our interface design, it can generally cooperate with server to realise the function of our product. However, we still found some problems existing in previous interface while preparing the demo1. Thus, with the help of suggestions of others, we try to modify it and to do some slight changes to make our interface more friendly to user. The following is the brief description and images of the primary changes to our interface (What I do not mention here still remained the same)

### a. Add register interface

In our previous design, we have owned login interface through which users can be endowed with authority. When consider that our project allows different users to work on it, it is necessary to create additional screen to satisfy this function. So we establish a new register interface. here comes our arrangement, when people using our home automation, the first interface is still the login page, in this page, we make a slight change that add a new button on it. If the user has not registered yet, they can click on the "create new user" button and come to the register interface. It contains a couple of text box, when users come to this interface, they can click the empty text box to input their basic information as they are required. The basic information includes name, age, password and their email. After enter the correct information, they can click on the "create" button to get authority. At the same time, the system would bring them back to the login interface.



The image shows a user registration interface. At the top, there is a dark grey header area. On the left, there is a circular icon of a person's head and shoulders. To the right of this icon is a blue button with the word "new" in white. Below the header, there are two input fields. The first is labeled "user name:" and contains three asterisks "\*\*\*". The second is labeled "password:" and contains six asterisks "\*\*\*\*\*". To the right of these fields is a blue circular button with a white right-pointing arrow. Below the input fields is a 4x3 grid of blue buttons. The buttons are labeled as follows: Row 1: 1, 2, 3; Row 2: 4, 5, 6; Row 3: 7, 8, 9; Row 4: exit, 0, <.

1	2	3
4	5	6
7	8	9
exit	0	<

Figure 86: Add register interface


b. Improve brightness adjusting button

Consider that adjust brightness with "+" button and "-" buttons is inconvenience to users, we decide to exchange those two buttons into a drag strip which

can dramatically decrease user effort. For example, if the user wants to adjust the lightness fro 10% to 90%, they have to do 8 clicks with old interface, with new interface, the user just needs to drag it at anywhere they want instead of clicking a button too many times. What's more, the drag sttrip can provide users with good visual presentation about the brightness. We believe those changes can improve te operability and avaiability of our interface without increasing its complexity. The current interface would make our product seems more reliable and with higher quality.

Figure 87: Improve brightness adjusting button





name:

email:

age:

password:

Figure 88: New user signup

## 14 Design of tests

### a. Casual Test Cases

**Use Case 1:** Light Controlling

User turn on or off the lights by sending the voice message to the server from the Android based cellphone.

**Use Case 2:** Modify the lightness

User control the music player by their voice. And the server sends the control instruction to the MCU after processing the voice message.

**Use Case 3:** Light Controlling

The users can modify the lightness of the light by their voice command.

**Use Case 4:** Modify the volume of music

The user can turn up/down the music by sending the related voice message to the server from the Android based cellphone.

**Use Case 6:** User log in

A user log in to his/her account in order to configure the alarm system. The tests confirm that the correct passwords and usernames allow the logins to the correct account.

**Use Case 8:** Remove user

User can remove the deserted user and their information.

**Use Case 12:** Control information feedback

The users can check the feedback of voice message in order to check whether the voice have been identified by the server. If the message identification failed, the feedback information can lead the user to send again or cancel the operation.

**Use Case 13:** Network error notification

The users can get the notification of network error from the user interface of Android cellphone if the signal of network is weak. The signal the users get is consist of 2 parts: Between cellphone and server; between server and MCU.

### b. Descriptive Test Cases

**Use Case 1:** Light Controlling

Success

- The light turns on when user press the light controlling button.

- The light turns on when user send a voice message containing the key words of turn on & light.

#### Failure

- User press the light controlling button, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User press the light controlling button, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.

#### **Use Case 2:** Music Controlling

##### Success

- The Music turns on when user press the light controlling button.
- The Music turns on when user send a voice message containing the key words of turn on & music.

#### Failure

- User press the music controlling button, but the signal of turn on the music does not go through to the server, and the music does not turn on.
- User press the music controlling button, the signal of turn on the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn on.
- User send a voice mail which contains the key words of turn on & music, but the signal of turn on the music does not go through to the server, and the music does not turn on.
- User send a voice mail which contains the key words of turn on & music, the signal of turn on the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn on.

### **Use Case 3:** Modify the lightness

#### Success

- The lightness grow/fade when user press the lightness controlling button.
- The lightness grow/fade when user send a voice message containing the key words of turn up/down & light.

#### Failure

- User press the light controlling button, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User press the light controlling button, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User send a voice mail which contains the key words of turn on light, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.

### **Use Case 4:** Modify the volume of the music

#### Success

- The Music turns up/down when user press the volume controlling button.
- The Music turns up/down when user send a voice message containing the key words of turn up/down & music.

#### Failure

- User press the music volume button, but the signal of turn up/down the music does not go through to the server, and the music does not turn up/down.
- User press the music volume button, the signal of turn up/down the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn up/down.
- User send a voice mail which contains the key words of turn up/down & music, but the signal of turn up/down the music does not go through to the server, and the music does not turn up/down.

- User send a voice mail which contains the key words of turn up/down & music, the signal of turn up/down the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn up/down.

#### **Use Case 6:** User Log In

Success

- User inputs username and password successfully. The system has to verify if the username and password exists and matches in the system. The match is made and the user is logged into his/her account.

Failure

- User inputs username and password, but when the system goes to verify, the username is incorrect and it doesnt match any username in the database and it displays a wrong username/ password error message.
- User inputs username and password, but when the system goes to verify, the password does not match with the one saved in the database and it displays a wrong username/password error message.
- User inputs username and password, but when the system goes to verify, neither the username nor the password match with the one saved in the database and it displays a wrong username/ password error message.

#### **Use Case 8:** Remove user

Success

- The user names and data is successfully removed and the deserted user-names cannot be used again.

Failure

- The user names and data is not removed and the deserted usernames can be used again.
- The user names and data is partially removed, the deserted usernames cannot be used again, however the data left takes up spaces in the server.

#### **Use Case 12:** Control information feedback

Success

- The user interface get feedback to check whether the voice have been identified by the server.

Failure

- The user interface get no feedback to check whether the voice have been identified by the server.
- The user interface get feedback to check whether the voice have been identified by the server, however the feedback is wrong.

**Use Case 13:** Network error notification

Success

- The user interface of Android cellphone display a text of connection notifications of both the signal between cellphone and server and the signal between server and MCU, which is right.

Failure

- The user interface of Android cellphone display no notification of the network status.
- The user interface of Android cellphone display a text of connection notifications of both the signal between cellphone and server and the signal between server and MCU, which is wrong.
- The user interface of Android cellphone display a text of connection notifications one of the signal between cellphone and server and the signal between server and MCU, which is wrong.

## 15 Past Works and Future Plan

### 15.1 Past Works

In this section we are going to talk about our tasks and plans. We have ten kinds of jobs in general, the system design related jobs and paper works.

We treat each week as a building circle so that our plan circle is a week. Table 8 shows the works and the related due time. Blue part means the tasks has already been finished; Green part represents the jobs we are working on; the yellow parts mean the tasks will be done in the future.

Now we accomplished all the tasks as we scheduled at the begining.

Task	SEP21	OCT4	OCT11	OCT18	OCT25	NOV1	NOV8	NOV15	NOV22	NOV29	DEC6	DEC13
Website Maintenance												
App UI Design												
App Frame Design												
Server Frame Design												
Smart Element Design												
Communication Design												
System De-Bugging I												
Problem Fix												
System De-Bugging II												
Problem Fix												
Report 1												
Report 2												
Report 3												
Demo 1												
Demo 2												
Archive Documentation												

### 15.2 Future Plans

Now we have a system consists of cellphone apps, a server and smart elements. And the connection between all the parts is the real internet which means you can actually use the whole system anywhere. What's more with the google voice, the voice recognition is pretty good and accurate.

We know the voice control based home automation system should actually based on the real requirements of users. So we might need to do a survey to collect the real needs to people's daily requirements. Then we can build more different smart elements to face the needs.

Also, I use some simple machine learn and statistics methods to extract the user's habits by checking the activity records in the database. We should improve the algorithms here to make the system smarter and faster.

Also we only build a very simple server with our laptop. In real use we need a real server and some security related algorithms should be added to the system.



## 16 Reference

- [1] **ANDROID (OS)**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/ANDROID\\_\(OPERATING\\_SYSTEM\)](http://en.wikipedia.org/wiki/Android_(operating_system))
- [2] **FUNCTIONAL REQUIREMENT**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/FUNCTIONAL\\_REQUIREMENT](http://en.wikipedia.org/wiki/Functional_requirement)
- [3] **USER STORIES**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [4] **ISO**. [HTTPS://DEVELOPER.APPLE.COM](https://developer.apple.com)
- [5] **ANDROID APPLICATION PLATFORM**. [WWW.MANTRAIS.COM/MOBILE/ANDROID-APPLICATION-DEVELOPMENT/](http://www.mantrais.com/mobile/android-application-development/)
- [6] **VOICE COMMAND DEVICE**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/VOICE\\_COMMAND\\_DEVICE](http://en.wikipedia.org/wiki/Voice_command_device)
- [7] **NON-FUNCTIONAL REQUIREMENT**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/NON-FUNCTIONAL\\_REQUIREMENT](http://en.wikipedia.org/wiki/Non-functional_requirement)
- [8] **FURPS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/FURPS](http://en.wikipedia.org/wiki/Furps)
- [9] **NON-FUNCTIONAL REQUIREMENT**. [HTTP://WWW.MOUNTANGOATSOFTWARE.COM/BLOG/NON-FUNCTIONAL-REQUIREMENTS-AS-USER-STORIES](http://www.mountangoatssoftware.com/blog/non-functional-requirements-as-user-stories)
- [10] **HOUSING**. [HTTP://MONEYNING.COM/HOUSING/THE-FIVE-YEAR-RULE-FOR-BUYING-A-HOUSE/](http://moneyning.com/housing/the-five-year-rule-for-buying-a-house/)
- [11] **UI-OVERVIEW**. [HTTP://DEVELOPER.ANDROID.COM/DESIGN/GET-STARTED/UI-OVERVIEW.HTML](http://developer.android.com/design/get-started/ui-overview.html)
- [12] **ANDROID**. [HTTP://WWW.ANDROID.COM/ABOUT/](http://www.android.com/about/)
- [13] **MOBILEPHONE UI DESIGN**. [HTTPS://WWW.GOOGLE.COM.HK/SEARCH?Q=MOBILE+UI+DESIGN&SAFE=STRICT&SA=X&TBM=ISCH&TBO=U&SOURCE=UNIV&EI=30LMUOSCLIHQ8QTBNOG4AG&VED=OCduQSAQ&BIW=1280&BIH=650&DPR=1](https://www.google.com.hk/search?q=mobile+ui+design&safe=strict&sa=x&tbm=isch&tbo=u&source=univ&ei=30LMUOSCLIHQ8QTBNOG4AG&ved=OCduQSAQ&biw=1280&bih=650&dpr=1)
- [14] **MOBILE-PATTERNS**. [HTTP://WWW.MOBILE-PATTERNS.COM/](http://www.mobile-patterns.com/)
- [15] **MOBILE APP DESIGN**. [HTTP://WWW.1STWEBDESIGNER.COM/DESIGN/MOBILE-APPS-DESIGNS/](http://www.1stwebdesigner.com/design/mobile-apps-designs/)
- [16] **GLADE**. [HTTPS://GLADE.GNOME.ORG/](https://glade.gnome.org/)
- [17] **HOME AUTOMATION AND CONTROL**. [HTTP://WWW.SAVANTSYSTEMS.COM/HOME\\_AUTOMATION\\_AND\\_CONTROL.ASPX](http://www.savantsystems.com/home_automation_and_control.aspx)
- [18] **SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [19] **STAKEHOLDERS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/STAKEHOLDERS](http://en.wikipedia.org/wiki/Stakeholders)
- [20] **USE CASE**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/USE\\_CASE#ACTORS](http://en.wikipedia.org/wiki/Use_case#actors)
- [21] **AUTOHOME**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF](http://eceweb1.rutgers.edu/~marsic/books/se/projects/autohome/2012-g1-report3.pdf)
- [22] **SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF](http://eceweb1.rutgers.edu/~marsic/books/se/projects/other/2012-g4-report3.pdf)
- [23] **SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [24] **SYSTEM SEQUENCE DIAGRAM**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/SYSTEM\\_SEQUENCE\\_DIAGRAM](http://en.wikipedia.org/wiki/System_sequence_diagram)
- [25] **SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [26] **SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF](http://eceweb1.rutgers.edu/~marsic/books/se/projects/autohome/2012-g1-report3.pdf)

[27]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF](http://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF)

[28]**EFFORT ESTIMATION**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/SOFTWARE\\_DEVELOPMENT\\_EFFORT\\_ESTIMATION](http://EN.WIKIPEDIA.ORG/WIKI/SOFTWARE_DEVELOPMENT_EFFORT_ESTIMATION)

[1]**ANDROID (OS)**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/ANDROID\\_\(OPERATING\\_SYSTEM\)](http://EN.WIKIPEDIA.ORG/WIKI/ANDROID_(OPERATING_SYSTEM))

[2]**FUNCTIONAL REQUIREMENT**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/FUNCTIONAL\\_REQUIREMENT](http://EN.WIKIPEDIA.ORG/WIKI/FUNCTIONAL_REQUIREMENT)

[3]**USER STORIES**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/)

[4]**ISO**. [HTTPS://DEVELOPER.APPLE.COM](https://DEVELOPER.APPLE.COM)

[5]**ANDROID APPLICATION PLATFORM**. [WWW.MANTRAIS.COM/MOBILE/ANDROID-APPLICATION-DEVELOPMENT/](http://WWW.MANTRAIS.COM/MOBILE/ANDROID-APPLICATION-DEVELOPMENT/)

[6]**VOICE COMMAND DEVICE**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/VOICE\\_COMMAND\\_DEVICE](http://EN.WIKIPEDIA.ORG/WIKI/VOICE_COMMAND_DEVICE)

[7]**NON-FUNCTIONAL REQUIREMENT**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/NON-FUNCTIONAL\\_REQUIREMENT](http://EN.WIKIPEDIA.ORG/WIKI/NON-FUNCTIONAL_REQUIREMENT)

[8]**FURPS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/FURPS](http://EN.WIKIPEDIA.ORG/WIKI/FURPS)

[9]**NON-FUNCTIONAL REQUIREMENT**. [HTTP://WWW.MOUNTAINGOATSOFTWARE.COM/BLOG/NON-FUNCTIONAL-REQUIREMENTS-AS-USER-STORIES](http://WWW.MOUNTAINGOATSOFTWARE.COM/BLOG/NON-FUNCTIONAL-REQUIREMENTS-AS-USER-STORIES)

[10]**HOUSING**. [HTTP://MONEYNING.COM/HOUSING/THE-FIVE-YEAR-RULE-FOR-BUYING-A-HOUSE/](http://MONEYNING.COM/HOUSING/THE-FIVE-YEAR-RULE-FOR-BUYING-A-HOUSE/)

[11]**UI-OVERVIEW**. [HTTP://DEVELOPER.ANDROID.COM/DESIGN/GET-STARTED/UI-OVERVIEW.HTML](http://DEVELOPER.ANDROID.COM/DESIGN/GET-STARTED/UI-OVERVIEW.HTML)

[12]**ANDROID**. [HTTP://WWW.ANDROID.COM/ABOUT/](http://WWW.ANDROID.COM/ABOUT/)

[13]**MOBILEPHONE UI DESIGN**. [HTTPS://WWW.GOOGLE.COM.HK/SEARCH?Q=MOBILE+UI+DESIGN&SAFE=STRICT&SA=X&TBM=ISCH&TBO=U&SOURCE=UNIV&EI=30LMUOSCLIHQ8QTBNOG4AG&VED=OCDUQSAQ&BIW=1280&BIH=650&DPR=1](https://WWW.GOOGLE.COM.HK/SEARCH?Q=MOBILE+UI+DESIGN&SAFE=STRICT&SA=X&TBM=ISCH&TBO=U&SOURCE=UNIV&EI=30LMUOSCLIHQ8QTBNOG4AG&VED=OCDUQSAQ&BIW=1280&BIH=650&DPR=1)

[14]**MOBILE-PATTERNS**. [HTTP://WWW.MOBILE-PATTERNS.COM/](http://WWW.MOBILE-PATTERNS.COM/)

[15]**MOBILE APP DESIGN**. [HTTP://WWW.1STWEBDESIGNER.COM/DESIGN/MOBILE-APPS-DESIGNS/](http://WWW.1STWEBDESIGNER.COM/DESIGN/MOBILE-APPS-DESIGNS/)

[16]**GLADE**. [HTTPS://GLADE.GNOME.ORG/](https://GLADE.GNOME.ORG/)

[17]**HOME AUTOMATION AND CONTROL**. [HTTP://WWW.SAVANTSYSTEMS.COM/HOME\\_AUTOMATION\\_AND\\_CONTROL.ASPX](http://WWW.SAVANTSYSTEMS.COM/HOME_AUTOMATION_AND_CONTROL.ASPX)

[18]**SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/)

[19]**STAKEHOLDERS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/STAKEHOLDERS](http://EN.WIKIPEDIA.ORG/WIKI/STAKEHOLDERS)

[20]**USE CASE**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/USE\\_CASE#ACTORS](http://EN.WIKIPEDIA.ORG/WIKI/USE_CASE#ACTORS)

[21]**AUTOHOME**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF](http://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF)

[22]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF](http://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF)

[23]**SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/)

[24]**SYSTEM SEQUENCE DIAGRAM**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/SYSTEM\\_SEQUENCE\\_DIAGRAM](http://EN.WIKIPEDIA.ORG/WIKI/SYSTEM_SEQUENCE_DIAGRAM)

[25]**SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/)

[26]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF](http://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF)

[27]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF](http://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF)

[28]**EFFORT ESTIMATION**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/SOFTWARE\\_DEVELOPMENT\\_EFFORT\\_ESTIMATION](http://EN.WIKIPEDIA.ORG/WIKI/SOFTWARE_DEVELOPMENT_EFFORT_ESTIMATION)

[29]**HARDWARE DOCUMENTATION**. [HTTP://WWW.FRITZING.ORG](http://WWW.FRITZING.ORG)

[30]**INFRARED COMMUNICATION**. [HTTP://WWW.SBPROJECTS.COM/KNOWLEDGE/IR/SIRC.PHP](http://WWW.SBPROJECTS.COM/KNOWLEDGE/IR/SIRC.PHP)

[31]**INFRARED COMMUNICATION PROTOCOL**. [HTTP://LEARN.ADAFRUIT.COM/TRINKET-GEMMA-IR-REMOTE-CONTROL/USING-IR-CODES-1](http://LEARN.ADAFRUIT.COM/TRINKET-GEMMA-IR-REMOTE-CONTROL/USING-IR-CODES-1)

[32]**COMMUNICATION PROTOCOL**. [HTTP://WWW.PIC-TRONICS.COM/IR-TRANSMITTER-AND-RECEIVER.PHP](http://WWW.PIC-TRONICS.COM/IR-TRANSMITTER-AND-RECEIVER.PHP)

[33]**SERIALPORT COMMUNICATION**. [HTTP://PLAYGROUND.ARDUINO.CC//CSHARP/SERIALCOMMSCSHARP](http://PLAYGROUND.ARDUINO.CC//CSHARP/SERIALCOMMSCSHARP)

[34]**INFRARED COMMUNICATION COMMUNICATION**. [HTTP://WWW.PYROELECTRO.COM/TUTORIALS/INFRARED\\_IR\\_RECEIVER/IR\\_THEORY\\_1.HTML](http://WWW.PYROELECTRO.COM/TUTORIALS/INFRARED_IR_RECEIVER/IR_THEORY_1.HTML)

[35]**WIRELESS COMMUNICATION**. [HTTP://WW1.MICROCHIP.COM/DOWNLOADS/EN/DEVICEDOC/ADN006.PDF](http://WW1.MICROCHIP.COM/DOWNLOADS/EN/DEVICEDOC/ADN006.PDF)

[36]**PIC MICROCONTROLLER**. [HTTP://WWW.MICROCHIP.COM/WWWPRODUCTS/DEVICES.ASPX?DDOCNAME=EN010297](http://WWW.MICROCHIP.COM/WWWPRODUCTS/DEVICES.ASPX?DDOCNAME=EN010297)

[37]**ARDUINO BASICS**. [HTTP://HTTP://WWW.NFIAUTOMATION.ORG/NFI\\_PROJECTS.HTML](http://HTTP://WWW.NFIAUTOMATION.ORG/NFI_PROJECTS.HTML)

[38]**COMMUNICATION PROTOCOL**. [HTTP://RATUL-ELECTRONICS.BLOGSPOT.COM/2012/03/RF-MODULE-TRANSMITTER-RECEIVER.HTML](http://RATUL-ELECTRONICS.BLOGSPOT.COM/2012/03/RF-MODULE-TRANSMITTER-RECEIVER.HTML)

[39]**ARDUINO BASICS**. [HTTP://HTTP://WWW.NFIAUTOMATION.ORG/NFI\\_PROJECTS.HTML](http://HTTP://WWW.NFIAUTOMATION.ORG/NFI_PROJECTS.HTML)

[40]**MICROPHONE BASICS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/MICROPHONE](http://EN.WIKIPEDIA.ORG/WIKI/MICROPHONE)

[41]**IR PROTOCOLS**. [HTTP://WWW.EPANORAMA.NET/LINKS/IRREMOTE.HTML](http://WWW.EPANORAMA.NET/LINKS/IRREMOTE.HTML)

[42]**STAR NETWORK**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/STAR\\_NETWORK](http://EN.WIKIPEDIA.ORG/WIKI/STAR_NETWORK)

[43]**HANDSHAKING PROTOCOL**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/HANDSHAKING](http://EN.WIKIPEDIA.ORG/WIKI/HANDSHAKING)

[44]**CHECKSUM**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/CHECKSUM](http://EN.WIKIPEDIA.ORG/WIKI/CHECKSUM)

[45]**MAKING THINGS TALK BY TOM IGOE**

[46]**VIRTUAL WIRE LIBRARY FOR ARDUINO WIRELESS CONNECTION**. [HTTP://WWW.AIRSPAYCE.COM/MIKEM/ARDUINO/](http://WWW.AIRSPAYCE.COM/MIKEM/ARDUINO/)