Software Requirements Specification (SRS)

Revision History:

Date	Author	Description
2019.3.17	Rui Xing	Editing system capabilities
2019.3.18	Shuihan Zhang	Editing system context
2019.3.19	Yuru Wang	Editing quality requirements (non-functional requirements)
2019.3.19	Zheng Chen	Introduction/Concept of Operation
2019.3.20	Rui Zhu	Editing fundamental assumptions
2019.3.20	Rui Xing	Editing expected subsets
2019.3.21	Rui Xing, Shuihan Zhang, Yuru Wang, Rui Zhu, Shijie Wen	Editing use cases
2019.3.21	Zheng Chen	Quality Requirements/Expected subsets
2019.3.21	Zhi Zhou	Overall block diagram
2019.3.21	Zimu Hu	Edit functional documentation
2019.3.22	Rui Xing, Shuihan Zhang, Yuru Wang, Rui Zhu, Shijie Wen	Editing use cases
2019.3.22	Zheng Chen	Behavioral Requirements
2019.3.23	Zhi Zhou	Modify functional documentation
2019.3.23	Zheng Chen	Use Cases/Behavioral Requirements
2019.3.23	Zheng Chen	Fundamental Assumption/Appendices
2019.3.23	Rui Xing, Shuihan Zhang, Yuru Wang, Rui	Adding use case

	Zhu, Shijie Wen	
2019.3.23	Shijie Wen	Editing detailed requirements
2019.3.23	Rui Zhu	Editing expected changes
2019.3.23	Yuru Wang	Editing appendices
2019.3.24	Shijie Wen	Modifying detailed requirements
2019.3.24	Rui Xing	Editing introduction
2019.3.25	Zhi Zhou	Add Server System Context
2019.3.25	Zhi Zhou	Add System Input & Output
2019.3.25	Renxiang Zhu	Add Quality Requirements
2019.3.25	Renxiang Zhu	Integrate documents
2019.3.25	Yuanjin Li	Editing Software Requirements Specification
2019.3.26	Yifan Zhang	Editing the Detailed Requirments
2019.3.26	Zhongyu Wang	Editing the Quality Requirments
2019.3.26	Zheng Chen	Revise Use Cases and System Inputs and Outputs
2019.3.26	Qingzhong Chen	Revise Use Cases
2019.3.27	Zheng Chen	Revise Use Cases and Fundamental Assumption
2019.3.28	Zhi Zhou	Combine Learning Ducks’ Documents
2019.3.31	Zhi Zhou	Combine Revision History
2019.4.1	Zheng Chen	Remove some parts of administrator’s adding and moving functions and use cases.
2019.4.1	Yuanjin Li	Modify the Output
2019.4.1	Yifan Zhang	Modify the Input

2019.4.1	Yifan Zhang	Add the Definitions
2019.4.1	Yuanjin Li	Modify the use cases

1. Introduction

1.1 Intended Audience and Purpose

This document is intended to provided information guiding development process, ensuring that all system requirements are met. The following entities may find the document useful:

- Customer - This page will detail all of the web app requirements as understood by the production team. The customer should be able to determine that their requirements will be correctly reflected in the final product through the information found on this page.
- Development Team - Details of specific requirements that the final software build must include will be located here. Developers can use this document to ensure the software addresses each of these requirements.
- QA Team - By developing testing procedures founded in the system requirements, the QA Team can create a comprehensive testing regimen that will guarantee requirements are met.

1.2 How to use the document

Table of Contents:
1. Introduction
2. Concept of Operations - broad description of the purpose of the application
  2.1 System Context - details any specific system requirements the application will require to run
  2.2 System Capabilities - description in prose of all capabilities available to the user in the address book
  2.3 Use cases - A detailed look at each functional requirement, describing the application context both before and after an action is taken
3. Behavioral Requirements - How the application will interact with a user
  3.1 Input and output requirements - A description of allowed inputs and generated outputs
    3.1.1 Input - Describes any restrictions that will be placed on allowed input
    3.1.2 Output - Describes the range of outputs that can be generated
  3.2 Detailed Output Behavior - Output descriptions in prose
4. Quality Requirements - Requirements not pertaining to the function of the application will be listed here
5. Expected Subsets - Expected levels of functionality at checkpoints during development
6. Fundamental Assumptions - Some specifics about input, output, or behavior upon which other requirements are founded will be listed here
7. Expected Changes - Future features and directions the project is expected to take
8. Appendices - Details aiding the understanding of this document
  8.1 Definitions and acronyms - Any technical terms or abbreviations will be spelled out here for ease of use of the document
    8.1 Definitions - Definitions of technical or unusual terminology
    8.1.2 Acronyms and Abbreviations - Any abbreviated terms will be expanded here
  8.2 References - any external references necessary or helpful to understanding this document will be listed here

2. System Capabilities

2.1. System Context

Requires a system with a GUI display and browser because all of the operations are performed through a GUI and a browser.

Windows:

- Windows 10 (8u51 and above)
- Windows 8.x (Desktop)
- Windows 7 SP1
- Windows Vista SP2
- Windows Server 2008 R2 SP1 (64-bit)
- Windows Server 2012 and 2012 R2 (64-bit)

Mac OS X:

- Intel-based Mac running Mac OS X 10.8.3+, 10.9+

Linux:

- Red Hat Enterprise Linux 5.5+[1], 6.x (32-bit), 6.x (64-bit)[2]
- Red Hat Enterprise Linux 7.x (64-bit)[2] (8u20 and above)
- Ubuntu Linux 12.04 LTS, 13.x
- Ubuntu Linux 14.x (8u25 and above)
- Ubuntu Linux 15.04 (8u45 and above)
- Ubuntu Linux 15.10 (8u65 and above)

2.2. System capabilities

Intelligent light control system Web APP is a web program that supports user interaction. On the web page, the user logins the account according to his personal ID and password, and then carries on the concrete operation to the intelligent light control system. Different kinds of users have different rights to intelligent light control system. There are three different permissions: students, teachers and administrators. The system functions are as follows:

1.User login. Users must be students, teachers or administrators of some schools.

2.Check the state of the light. All users have this permission.

3.Check whether a room is occupied. All three users have this permission.

4.Check the state of the light sensor. In this function, users can see the situation of ambient light.

5.Turn on/off the lights. Student users can only turn on the light when it is off and the classroom is occupied, and turn off the light when it is on and the classroom is empty. When the relevant operation cannot be carried out, a window will pop up to show the reasons: For example, *There are people in the classroom, so you cannot turn off the lights*. Teachers and administrators directly force the lights to be on/off. Students, teachers and administrators can operate the switch of a light or the main switch of all lights.

6.Add/delete new rooms. Administrators have this permission.

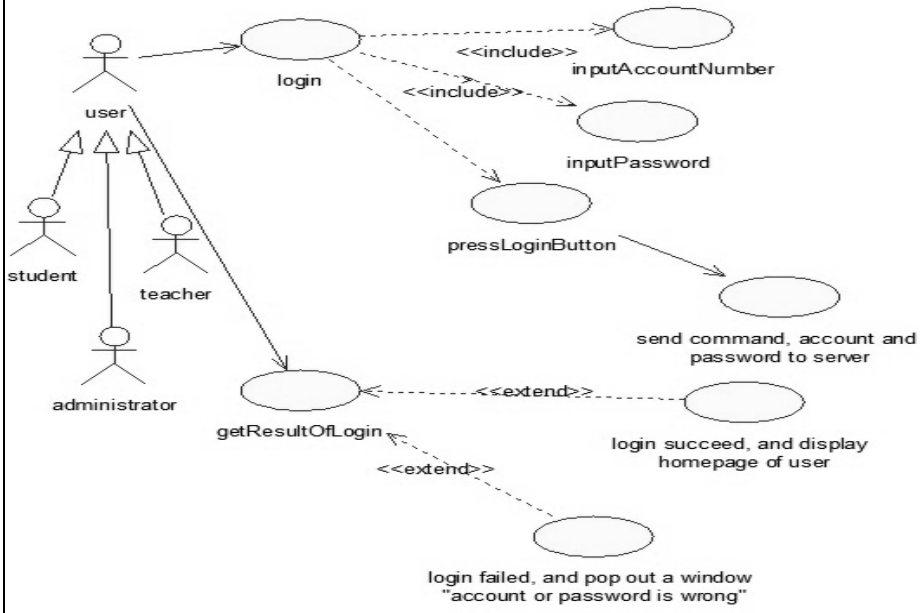
7.Add/delete sensors. Administrators have this permission. There are three kinds of sensors: switch sensor, light sensor and Presence sensor.

8.Add/delete actuators (lights). Administrators have this permission.

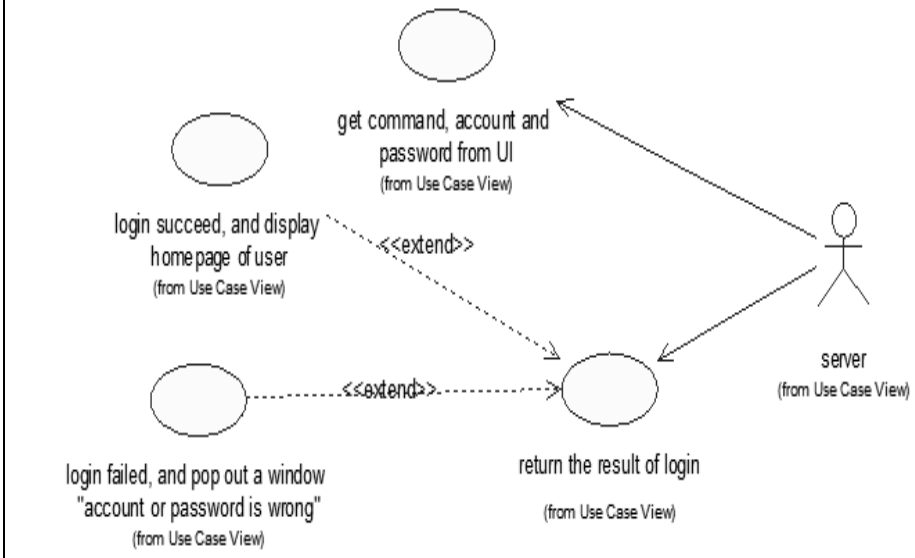
2.3. Use cases for Customers

2.3.1 User login

Use Case	user login		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	User Login and go into the light system		
Goals	User Go into the light system		
Summary	Login by inputting account number, password and press login button.		

Actors	user	
Trigger	Inputting account number, password and press login button.	
Precondition	None	
Basic Flow	Actor	System
1	User(student, teacher and administrator)input account number and password.	
2	User press login button	
2		Login part of UI gets the account number and password.
		Login part of UI sends command, account number and password to server
3	user get the result of login. If login succeed, the homepage of user will be displayed. If login fails, a window will be poped out, “account or password is wrong" .	
Frequency		
Type	Primary	
Postconditions	The web page is displayed.	
Chart		
Alternate Flow	*Actor*	System
1	User(student, teacher and administrator) Register account	Login part of UI will let you input account number, email and password and save it.
2	User forget password	Login part of UI will let you input email and account number. And it will send a link to your email and let you change your password.
3		

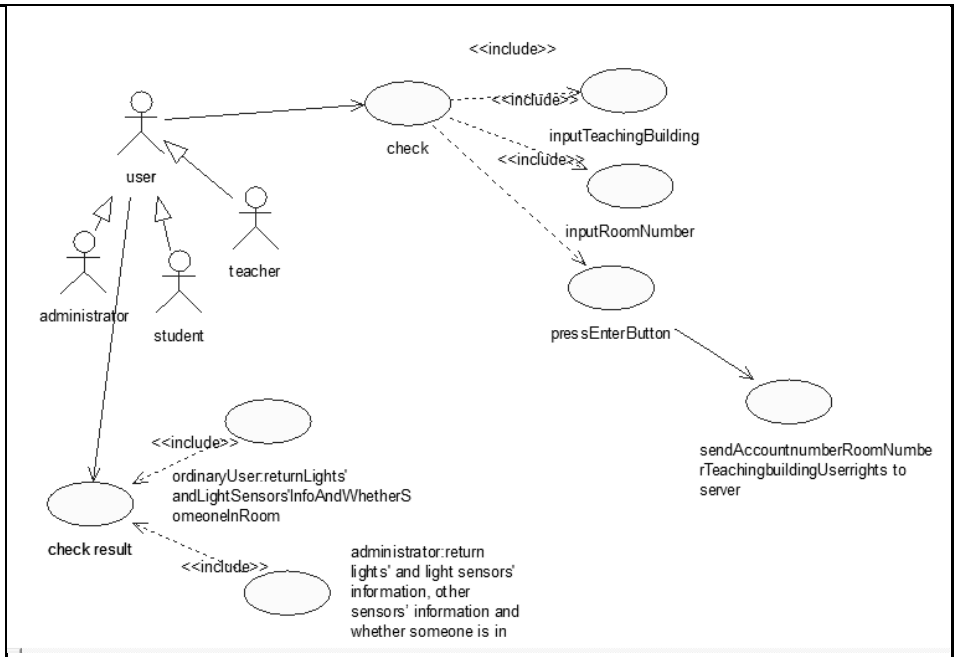
2.3.2 Verify login

Use Case	verify login		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	verify login		
Goals	server get login information, verify it and then go into the light system		
Summary	Server get information and verify it.		
Actors	server		
Trigger	user press login button.		
Precondition	None		
Basic Flow	Actor		System
1	command, account number and password to server		
2	Server returns back result of login.		
3			UI displays the result of login. If login succeed, the homepage of user will be displayed. If login fails, a window will be poped out, “account or password is wrong" .
Frequency			
Type	Primary		
Postconditions	The web page is displayed.		
Chart			
Alternate Flow	*Actor*		System

1	User(student, teacher and administrator) Register account	Login part of UI will let you input account number, email and password and save it.
2	User forget password	Login part of UI will let you input email and account number. And it will send a link to your email and let you change your password.
3		

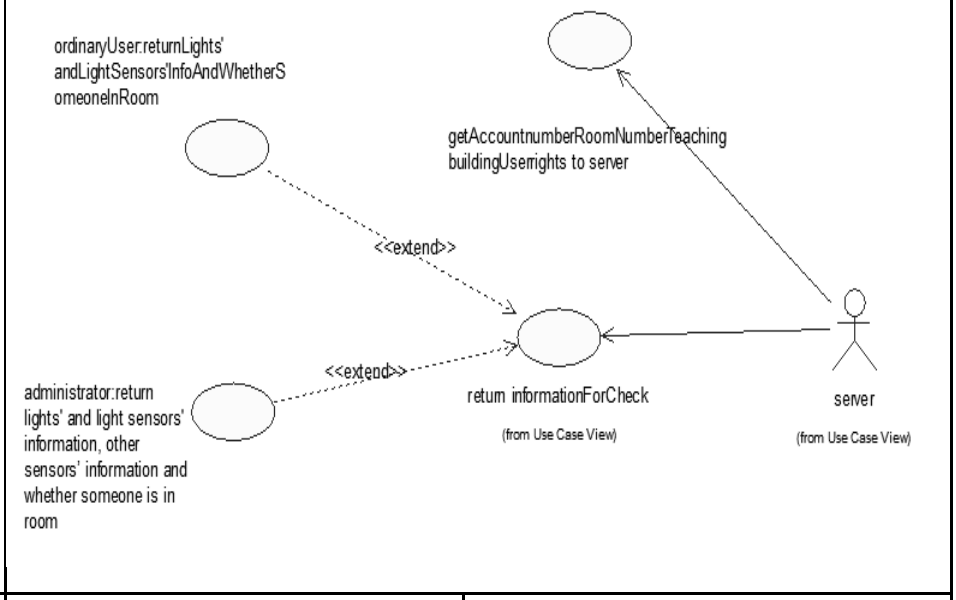
2.3.3 Check the state of lights or light sensors or check whether someone is in room

Use Case	check the state of lights or light sensors or check whether someone is in room		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	check the state of lights or light sensors or check whether someone is in room		
Goals	check the state of lights or light sensors or check whether someone is in room		
Summary	Check all states of lights and sensors and whether someone is in room by inputting room number and choosing teaching building.		
Actors	user		
Trigger	inputting room number and choosing teaching building		
Precondition	Login and press “lights and sensors”		
Basic Flow	Actor		System
1	User inputs teaching building name and room number and press enter button.		
2			To server: UI part will send account number, room number, teaching building and user’s current right.
3	The user check results.		
4			If the user is an ordinary user(student or teacher), the server will return lights' and light sensors' information and whether someone is in room. If the user is an administrator, the server return lights' and light sensors' information, other sensors’ information and whether someone is in room.
Frequency			
Type	Primary		
Postconditions	The state of light are displayed.		

Chart		
Alternate Flow	*Actor*	System
1		

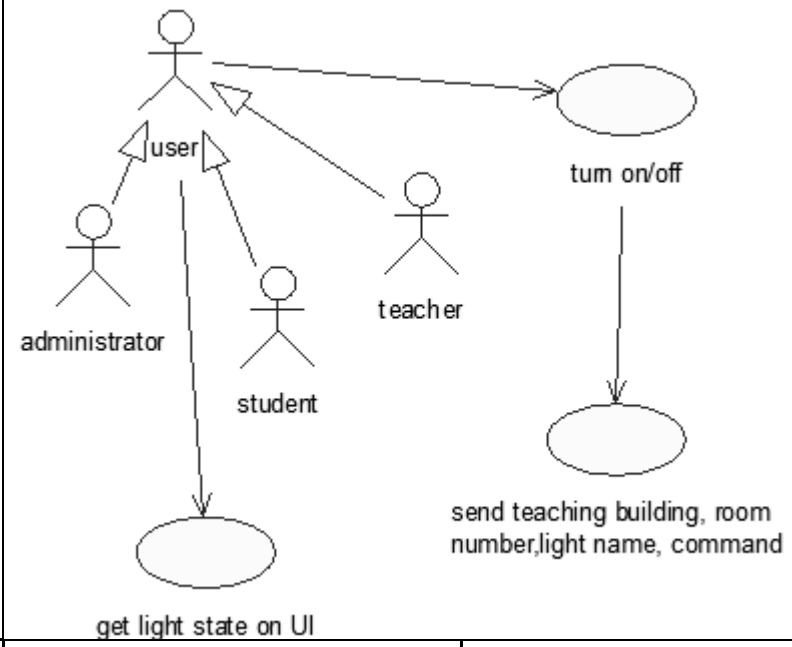
2.3.4 Server checks

Use Case	Server checks		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	Server checks.		
Goals	Server checks the state of lights or light sensors or check whether someone is in room		
Summary	Server checks all states of lights and sensors and whether someone is in room		
Actors	server		
Trigger	UI sends check command to server		
Precondition	Login and press “lights and sensors”		
Basic Flow	Actor		System
1	From UI : server gets account number, room number, teaching building and user’s current right.		
2			Server return information for checking
3	If the user is an ordinary user(student or teacher), the server will return lights' and light sensors' information and whether someone is in room. If the user is an administrator, the server return lights' and light sensors' information, other sensors’ information and whether someone is in room		
Frequency			

Type	Primary	
Postconditions	The state of light are displayed.	
Chart		
Alternate Flow	*Actor*	System
1		

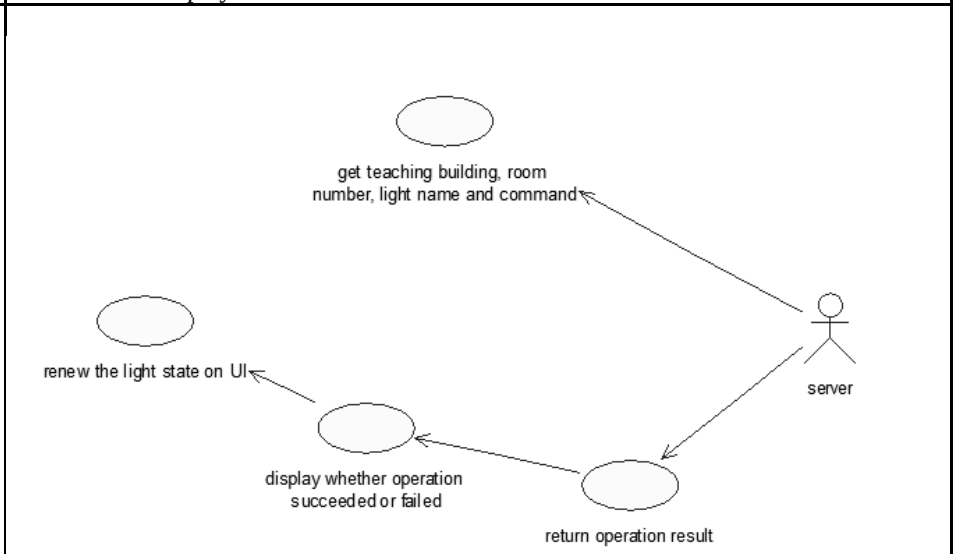
2.3.5 User turns on/off

Use Case	User Turn on/off		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	User turns on/off the lights		
Goals	User turns on/off the lights		
Summary	User turns on/off the lights		
Actors	user		
Trigger	Choose room number and choose teaching building and choose lights. Finally press the turn on/off button.		
Precondition	Login and check		
Basic Flow	Actor		System
1	User presses turn on/off button		
2			UI part will send teaching building name, room number, light name and command to server.
3			Server return operation result
4	UI will display that the operation		
Frequency			
Type	Primary		
Postconditions	The result is displayed.		

Chart		
Alternate Flow	*Actor*	System
1		

2.3.6 Server turns on/off

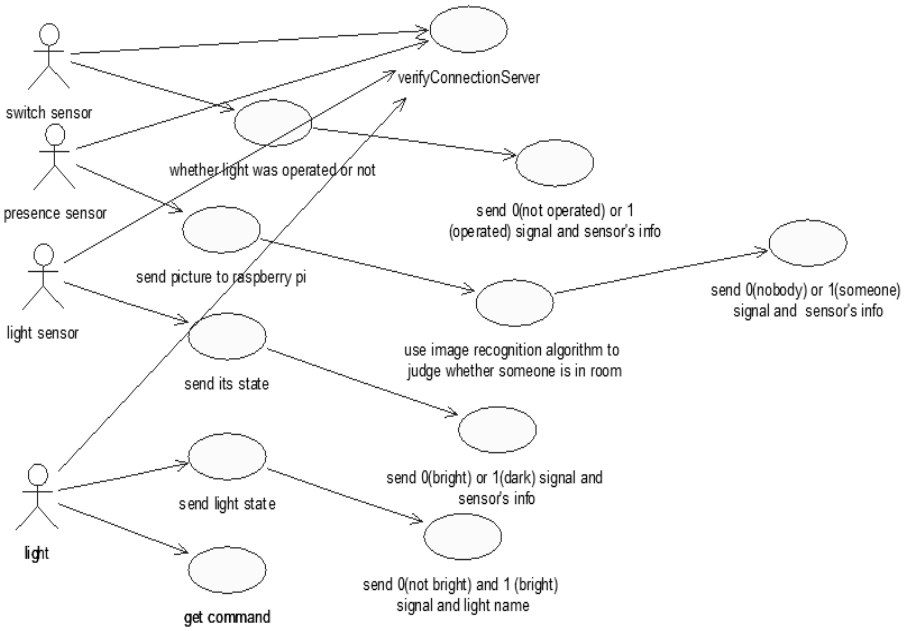
Use Case	Server turn on/off		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	Server turns on/off the lights		
Goals	Server turns on/off the lights		
Summary	Server turns on/off the lights		
Actors	user		
Trigger	User presses the turn on/off button.		
Precondition	Login and check		
Basic Flow	Actor		System
1	server gets teaching building name, room number, light name and command to server.		
2	Server return operation result		
3			UI will display that the operation succeeded or failed . After that, UI will renew light state.
Frequency			
Type	Primary		

Postconditions	The result is displayed.	
Chart		
Alternate Flow	*Actor*	System
1		

2.3.7 Hardware sends signals and gets command

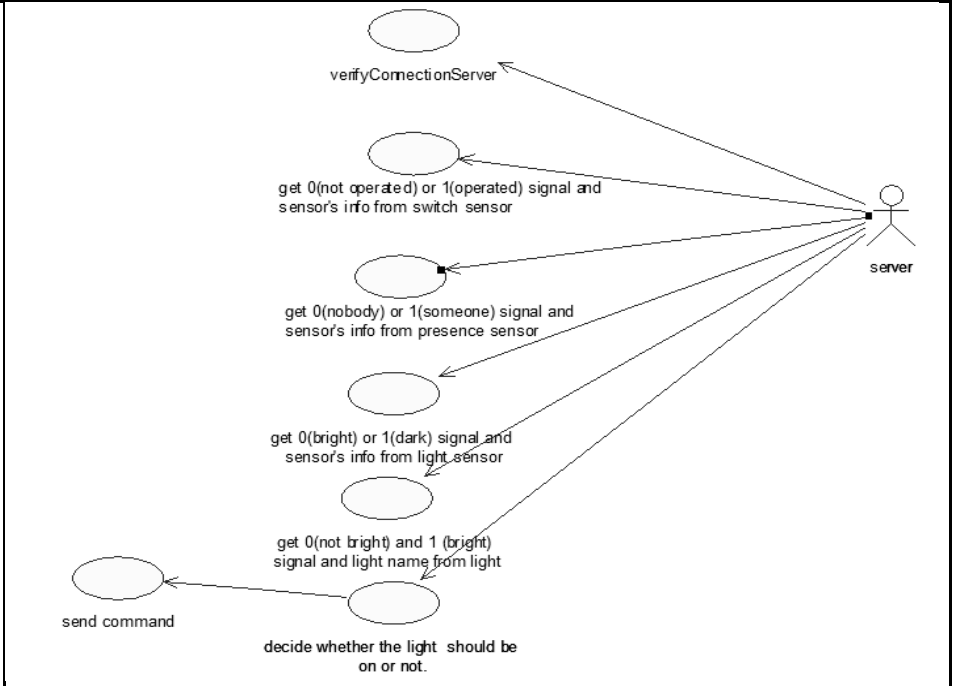
Use Case	hardware sends signals and gets command		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	hardware sends signals and gets command		
Goals	hardware sends signals and gets command		
Summary	hardware sends signals and gets command		
Actors	user		
Trigger	Sensors send their data to communication module.		
Precondition			
Basic Flow	Actor		System
1	Communication module verify connection to the server		
2			Server will accept the connection and tell communication module.

3	3.1 Switch sensor tells communication module whether light was operated or not. 3.2 Presence sensor send a picture to raspberry pi to communication module. 3.3 Light sensor send its state to communication module. 3.4 Light send its state to communication module.	
4		4.1 Communication module sends the switch sensor's information and 0(not operated)/1(operated)signals toserver. 4.2 Communication module uses image recognition algorithm to judge whether someone is in room. And then it send 0(nobody) or 1(someone) signal and presence sensor's information to server. 4.3 Communication module send 0(bright) or 1(dark) signal and light sensor's information to server. 4.4 Communication module send 0(not bright) and 1 (bright) signal and light name to server.
5	light gets command from server.	
Frequency		
Type	Primary	
Postconditions		

Chart		
Alternate Flow	*Actor*	System
1		

2.3.8 Server gets signals from hardware

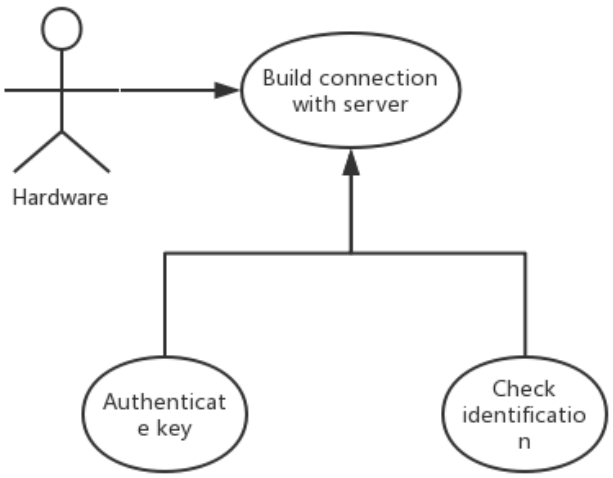
Use Case	Server gets signals from hardware		
Version	1.0	Created	3-23-19
Author	Zheng Chen		
Source	User stories		
Purpose	Server gets signals from hardware		
Goals	Server gets signals from hardware		
Summary	Server gets signals from hardware		
Actors	user		
Trigger	Sensors send their data to communication module.		
Precondition			
Basic Flow	Actor		System
1	server verifies connection from hardware.		
2	2.1 server gets the switch sensor's information and 0(not operated)/1(operated)signals. 2.2 server gets send 0(nobody) or 1(someone) signal and presence sensor's information. 2.3 server gets 0(bright) or 1(dark) signal and light sensor's information. 2.4 Server gets 0(not bright) and 1 (bright) signal and light name.		

3	The Server decides whether the light should be on or not.	
4		Communicatioin module sends command to lights.
Frequency		
Type	Primary	
Postconditions		
Chart		
Alternate Flow	*Actor*	System
1		

2.4. Use cases of Server

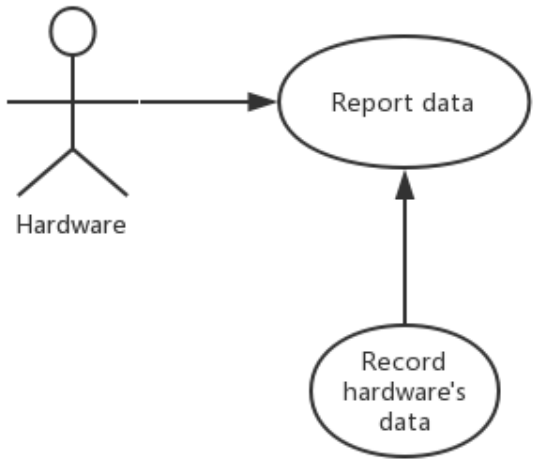
2.4.1 Hardware connects to server

Use Case	Hardware connects to server.		
Version	V1.0	Created	*2019.3.25*
Author	Zhi Zhou		
Source	Hardware		
Purpose	Build connects between server and hardware.		
Goals	Authenticate hardware's identification and build connections.		
Summary	Hardware raise a connecting request. After authenticating hardware's identification, server will build the connection.		
Actors	Hardware		
Trigger	Hardware boot.		
Precondition	Server is running		
Basic Flow	*Actor*		System
1	Raise a connecting request.		

2		Authenticate hardware's key. (Move to alternate flow 1 when error)
3		Authenticate whether hardware is registered in the database. (Move to alternate flow 1 when error)
4		Build connection with Hardware.
Frequency		
Type	Primary	
Postconditions	Connection is built.	
Chart		
Alternate Flow	*Actor*	System
1		Reject the connecting request.

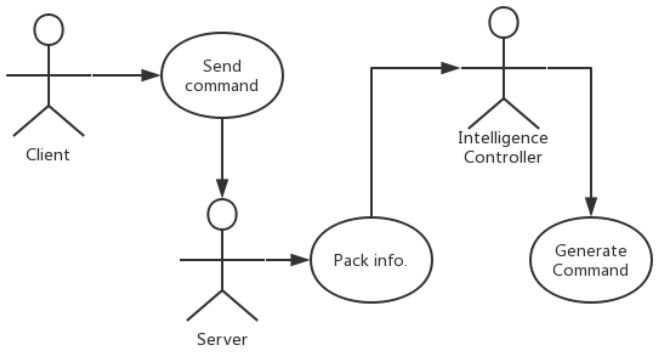
2.4.2 Hardware reports data

Use Case	Hardware reports data		
Version	V1.0	Created	*2019.3.25*
Author	Zhi Zhou		
Source	Hardware		
Purpose	Report sensors' data to server		
Goals	Send data and live package to server.		
Summary	Report sensors' data to server.		
Actors	Hardware		
Trigger	Sensors' data changed.		
Precondition	Connection is built.		
Basic Flow	*Actor*		System
1	Send sensors' data to server through socket. (Move to alternate flow 1 when failed.)		

2		Record the data in memory.
Frequency		
Type	Primary	
Postconditions	Data is sent.	
Chart		
Alternate Flow	*Actor*	System
1	Try to reconnect.	

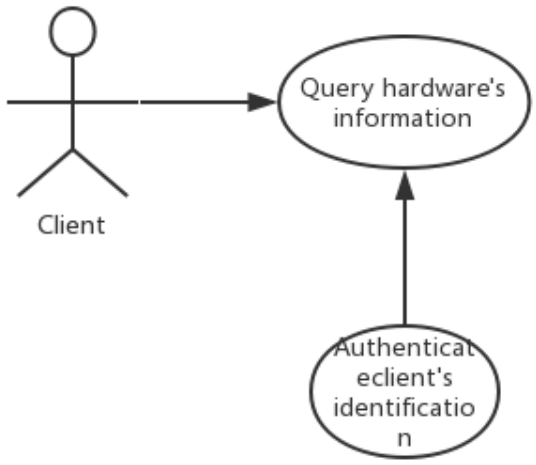
2.4.3 Client sends command

Use Case	Client sends command		
Version	V1.0	Created	*2019.3.25*
Author	Zhi Zhou		
Source	Client		
Purpose	Give hardware the command after handled by intelligence controller.		
Goals	Gather necessary data for IC, send data to IC, get command from IC and send command to hardware.		
Summary	Server give intelligence controller the command submitted by the client. And then send the result generated by the intelligence controller to hardware.		
Actors	Client		
Trigger	Client sends command		
Precondition	Server and hardware is running		
Basic Flow	*Actor*		System
1	Send command to server.		
2			Check user's authority. (Move to alternate flow 1 when failed.)
3			Check whether the target is online. (Move to alternate flow 2 when target is offline)

4		Pack necessary and related data, and send them to intelligence controller with command.
5	Generate the command and return it to the server.	
6		Send command to hardware.
Frequency		
Type	Primary	
Postconditions	Hardware executed the command.	
Chart		
Alternate Flow	*Actor*	System
1		Reject the command
2		Tell client that the target is offline.

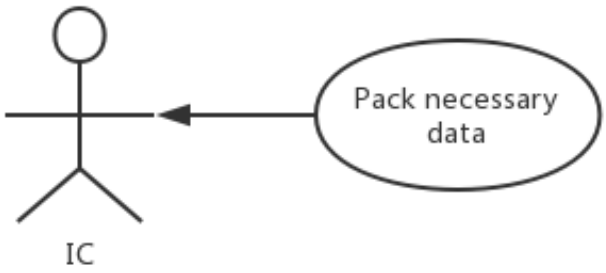
2.4.4 Client queries hardware's information

Use Case	Client queries hardware's information		
Version	V1.0	Created	*2019.3.25*
Author	Zhi Zhou		
Source	Client		
Purpose	Client got the hardware's information.		
Goals	Authenticate client's identification and then client got the hardware's information.		
Summary	Client raises a query request. After authenticating user's authority, server give client what it wants.		
Actors	Client		
Trigger	Client raises a request.		
Precondition	Server is running		
Basic Flow	*Actor*		System
1	Raise a query request.		
2			Authenticate user's authority. (Move to alternate flow 1 when error)
3			Report the data.
Frequency			
Type	Primary		
Postconditions	Client got the information.		

Chart		
Alternate Flow	*Actor*	System
1		Reject the query request.

2.4.5 Sensors' data affect the hardware

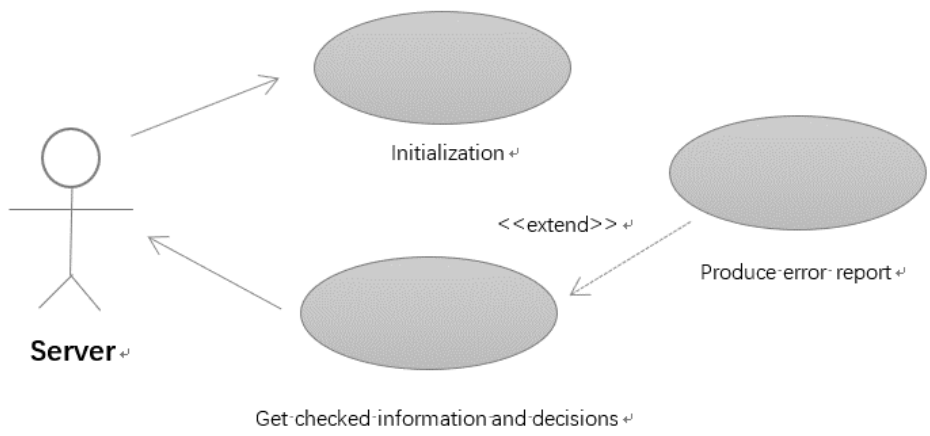
Use Case	Sensors' data affect the hardware	
Version	V1.0 Created	*2019.3.25*
Author	Zhi Zhou	
Source	Intelligence Controller	
Purpose	Hardware got the command.	
Goals	Hardware got the command.	
Summary	Server send intelligence controller's command to hardware.	
Actors	Server	
Trigger	Service received hardware's data.	
Precondition	Server is running and hardware just reported its data.	
Basic Flow	*Actor*	System
1		Pack necessary and related data, and send them to intelligence controller with command.
2	Generate the command and return it to the server.	
3		Send command to hardware.
Frequency		
Type	Primary	
Postconditions	Hardware executed the command.	

Chart		
Alternate Flow	*Actor*	System

2.5. Use cases of Intelligence Controller

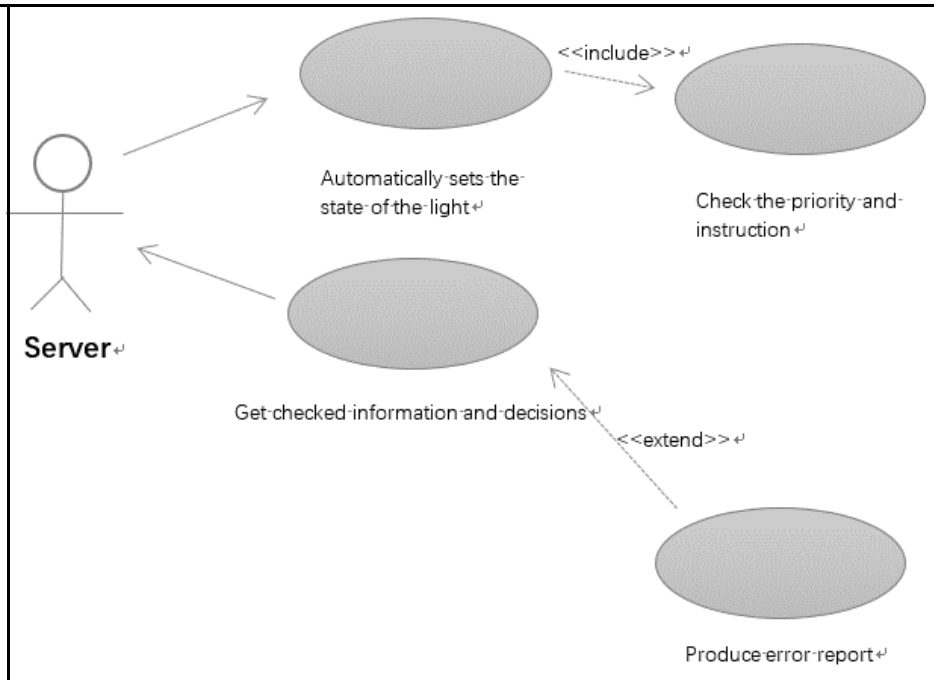
2.5.1 Initialize the system

Use Case	Initialize the system		
Version	1.0	Created	*2019-4-1*
Author	Li Yuanjin		
Source	Requirement		
Purpose	Initialize the system		
Goals	Make the system start to work		
Summary	Server give a signal to make the system initialized.		
Actors	Server		
Trigger	Customer start the system		
Precondition	None		
Basic Flow	*Actor*		System
1	Server give a package of the data to initialize the system		
2			Initialization and give a reply
Frequency	Once.		
Type	Primary		
Postconditions	The project assignment is created		

Chart		
Alternate Flow	*Actor*	System

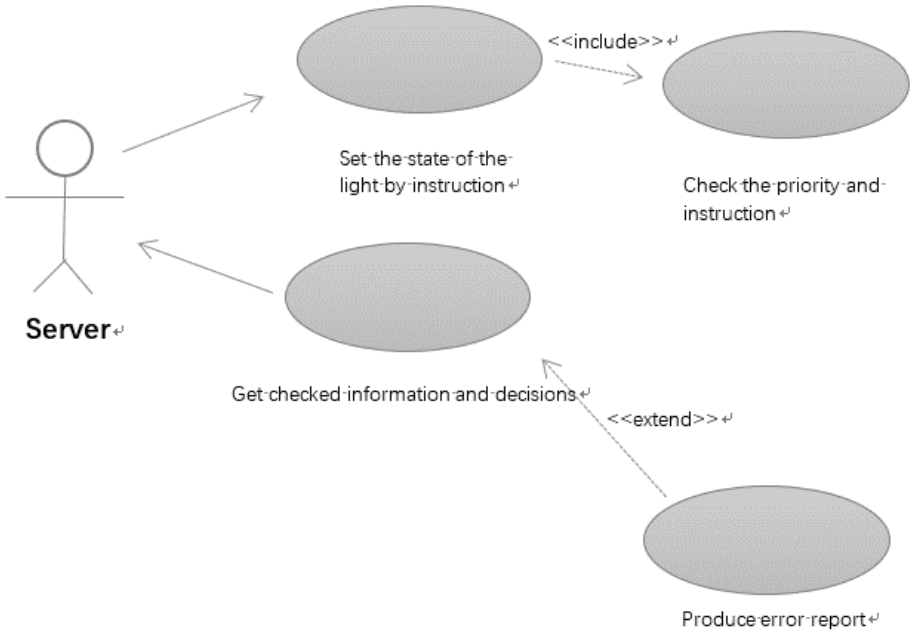
2.5.2 Automatic mode

Use Case	Automatic mode		
Version	2.0	Created	*2019-4-1*
Author	Li Yuanjin		
Source	Requirement		
Purpose	Power saving intelligently		
Goals	Control the status of the light		
Summary	Automatically sets the state of the light.		
Actors	Server		
Trigger	None		
Precondition	Automatic mode		
Basic Flow	*Actor*		System
1	Server give a package of the data		
2			Judge the situation, check the priority and instruction and give the command
Frequency	1 time in a minute		
Type	Primary		
Postconditions	The project assignment is created		

Chart		
Alternate Flow	*Actor*	System

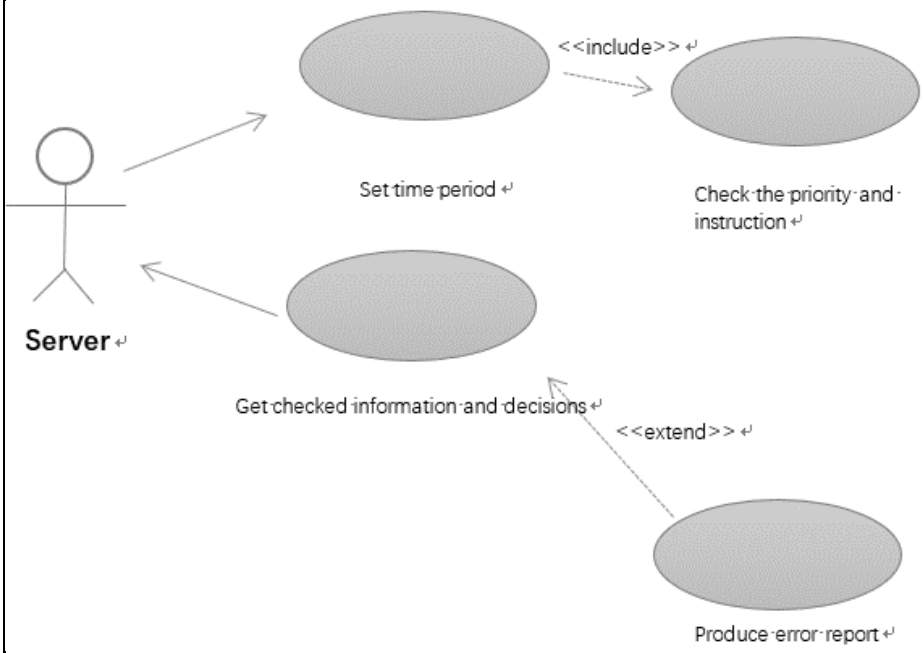
2.5.3 Command-light mode

Use Case	Command-light mode	
Version	2.0 Created	*2019-3-31*
Author	Zhang Yifan	
Source	Requirement	
Purpose	Turn the light on or off correctly by instruction	
Goals	Change the status of the light or give the error report	
Summary	A user issues an instruction to change the light through the server, then the Intelligent Control System (our system) make a judgement and return the result.	
Actors	Server	
Trigger	Someone gives an instruction to change the status of the light.	
Precondition	None	
Basic Flow	*Actor*	System
1	Server: Send instruction to change the state of the light	
2		Check the priority and instruction and make a decision back to the server
Frequency	2s	
Type	Primary	
Postconditions	The project assignment is created	

Chart		
Alternate Flow	*Actor*	System

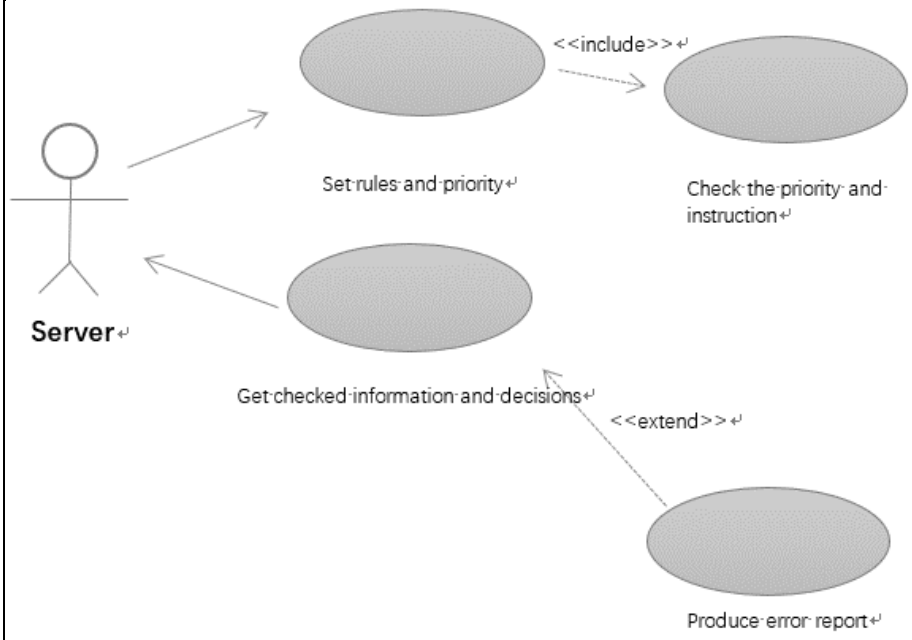
2.5.4 Time setting mode

Use Case	Time setting mode		
Version	2.0	Created	*2019-3-31*
Author	Zhang Yifan		
Source	*Requirement*		
Purpose	(The administrator) Set the time period that during these time slots our system will keep the light on or off all the time, until a teacher's or administrator's command change the state.		
Goals	Set the time period		
Summary	An administrator issues a command to change the time periods through the Server, then the Intelligent Control System (our system) make a judgement and return the results or the reason why he can't do it.		
Actors	Server		
Trigger	A command to change the time periods		
Precondition	The command came from an administrator.		
Basic Flow	*Actor*		System
1	Server sends data to Intelligent Control		
2			By checking the priority and instruction system make a decision and send it to Server
Frequency	2s		
Type	Primary		
Postconditions	The project assignment is created		

Chart		
Alternate Flow	*Actor*	System

2.5.5 Rules setting mode

Use Case	Rules setting mode		
Version	2.0	Created	*2019-3-31*
Author	Zhang Yifan		
Source	Requirement		
Purpose	(The administrator) Set the rules of our system, including priority and orders		
Goals	Set the rules		
Summary	A user issues a command to change the rules through the Server, then the Intelligent Control System (our system) make a judgement and return the results or the reason why he can't do it.		
Actors	Server		
Trigger	A command to set the rules.		
Precondition	The command came from an administrator.		
Basic Flow	*Actor*		System
1	Server sends data to Intelligent Control System		
2			By checking the priority and instruction system make a decision and send it to Server
Frequency	2s		
Type	Primary		
Postconditions	The project assignment is created		

Chart		
Alternate Flow	*Actor*	System

3. Detailed Requirements

3.1 System Inputs and Outputs for Customers

3.1.1 Inputs

The input of the application comes from the user.
Login interface comes at the beginning. There are two text boxes to be entered, account number and password.
In the navigation bar, there are "home page", "lights", "Sensors", "rooms", "current user identity" and "user personal information". Click on "lights" and there will be two drop-down menus of "building name" and "room number", "enter" and "return to the previous page" buttons on the left side of the interface. After clicking "Enter", there are all the lights in the room on the right side of the interface, as well as the switch of the lights, the check of the lights (full selection, reverse selection), the status of the light sensor and the prompt information box of the room.

Input at login interface:
* Account: must be made up of numbers. It can only be one of the teaching number, teacher's work number and administrator's ID number.
* Password: 6-20 characters.
* Login: Click on this button to enter the next interface with the correct account number and password.

Under "sensors", click on the Add button and enter the following:
* Sensor types: Only one of three types can be selected from the drop-down menu.

Under "rooms", click the Add button and enter:
* Room number: Input cannot conflict with an existing room number. And it is less than 5 legal numbers or letters.

Input in basic information:
* Nickname: less than 20 characters
* ID number: less than 10 digits
* School: less than 30 characters
* Professional: less than 20 characters
* Class: less than 20 characters

"Modify password" input:
* Old passwords: 6-20 characters
* "New password": 6-20 characters.

3.1.2 Outputs

Display graphical user interface. Each current interface contains all text boxes or interactive buttons created for users to enter.

Output to the user:
Login interface:
* If the password or account is incorrect, a pop-up window will prompt "incorrect password or account".

Turn on the lights:
* If the user is a student and the room is occupied, when the "turn on" button is pressed, a pop-up window will prompt "the room is occupied, the students can not turn off the lights at will". If the room is unoccupied, when the "turn off" button is pressed, a window will pop up to indicate that "the room is unoccupied", and students can not turn on the light at will. If the switch is checked, similar.

3.2 Detailed Output Behavior for Customers

Login interface comes at the beginning. There are two text boxes to be entered, account number and password.

In the navigation bar, there are "home page", "lights", "Sensors", "rooms", "current user identity" and "user personal information". Click on "lights" and there will be two drop-down menus of "building name" and "room number", "enter" and "return to the previous page" buttons on the left side of the interface. After clicking "Enter", there are all the lights in the room on the right side of the interface, as well as the switch of the lights, the check of the lights (full selection, reverse selection), the status of the light sensor and the prompt information box of the room. From the administrator's perspective, there is a red remove button next to each light, and a green new one light button in the right place. The lower right corner of the interface has remove ticks.

Click on "sensors" and there will be two drop-down menus of "building name" and "room number", "enter" and "return to the previous page" buttons on the left side of the interface. Click "Confirm" and all the sensors and their status will appear on the right side of the interface.

Click on "rooms" and there will be a drop-down menu of "teaching building name", "confirmation" and "return to the previous page" buttons on the left side of the interface. Click on the "Confirm" button and all the room numbers in this building will appear on the right side of the interface.

Click on "User Personal Information" and the buttons "Basic Information" and "Modify Password" appear on the left side of the interface. After clicking on the "basic information", there will be "nickname", "ID

number", "school", "major" and "class" on the right side of the interface, as well as a "confirm modification" button. Click "Modify Password" and the text box of "New Password" and "Old Password" will appear on the right side of the interface, and the button "Confirm Modification" will appear.

3.4 System Inputs and Outputs for Developer

3.4.1 Inputs

The inputs send to the server when client queries hardware's data should be in the form of json which content is:

uid: The user's unique identification.
sid: User's secure ID.
hid: The hardware's unique identification.

The inputs send to the server when client want to operate a hardware should be in the form of json which content is:

uid: The user's unique identification.
sid: User's secure ID.
hid: The hardware's unique identification.
cmd: The command client sent.

The inputs send to server when hardware want to report their data should be in the form of json which content is:

data: The data which sensor want to report.

The inputs send to server when intelligence controller generated command should be in the form of json which content is:

data: The command that intelligence controller generated.

```
ROOM{
    *Room_id: the id of the room
    *Light state{
        *State: it can be a boolean type, whose value is true or false. True means that it is on now, while
        false means the opposite.
        ...
    }
    *Sensor state{
        *kind: it is a string type, has three values, {motion, light, button}
        *online: it is a boolean type.
        *value: It is a numerical type.
    }
};
Instruction{
    *User_priority: it is a numerical type and means user's priority
    *Instruction_type: the instruction has four kinds, { auto, instruction, time, rules}.
    *Extra_information: set time period or make rules.
};
Extra_information{
    *Data_about_time: .....
    *Data_about_rule: ......
    *Data_about_priority: ......
} ;
```

3.4.2 Outputs

The outputs send to intelligence controller from server when something need to do with hardware should be in the form of json which content is:

sensors: The list of sensors with their up-to-date data.
device: The device and its up-to-date data.
cmd: The command (Leave blank if there is no command existed.)
authority: The level of operator.

The outputs send to client when server report hardware's information should be in the form of json which content is:

hid: The hardware's unique identification.
online: Whether the hardware is online.
nickname: The nickname of hardware.
last: The timestamp of last update.
data: The hardware's data.

The outputs send to hardware when server send command should be in the form of json which content is:

data: The command.

The outputs send to the Server.
*Result: There outputs required, there are {value, room, hint}.
{
*value: it is a string type whose value is in set:{"open", "close", "null", "exception"} . "open" means turn on the light, "close" means turn off the light, "null" means do nothing and "exception" means don't change the light and send some error information to the Server.
*room: it is a numerical type that means the result for which room.
*hint: it is a string type, the content is for explaining the result when intelligent control system reject the command.
}

3.5 Detailed Output Behavior for Developer

4 Quality Requirements (Non-functional Requirements)

The system must show good behavior in many fields like Performance, Security, Availability, Reliability, Modifiability, Maintainability, Understandability.
Interface aesthetics:
Simple, comfortable and elegant.
Performance:
The system can respond the users' operation in less than 500ms
The hardware can respond the command in less than 1000ms
Security:
The system must have different authority. The administrator's jurisdiction must not be used by any other users.
Availability:
The user's operation must be judged strictly by control part. Every situation must have a solution even if the user has a wrong operation.
Reliability:
The system must be anti-interference. When some signal comes in a wrong way, the system should recognize it and give the respond.

Modifiability:

The system can be changed. When users need some new functions, we can add up them into the system.

Maintainability:

The system has to easily to be fixed. If some parts get wrong, it can easily to find some other things to take place.

Understandability:

The system must be easy for users. The UI and specification have to be good for users.

5. Expected Subsets

L0:

- Basic GUI.
- Users can log in. Ability to send data to back-end storage and call data from back-end storage.

L1:

- Better GUI
- Ability to add/remove actuators (lights). Administrators have this permission.
- Ability to add/delete new rooms. Administrators have this permission.
- Ability to add/remove sensors.

L2:

- Complete GUI for Intelligent Lighting Control
- Ability to see the status of the light. All three users have this permission.
- Check if a room is occupied. All three users have this permission.
- Ability to check the status of the light sensor. All three users have this permission.
- Ability to turn on/off the light. All three users have this right.

6. Fundamental Assumptions

Hardware: Raspberry pi 3B+, Camera, Light sensor, Light.
Software: Linux operating system， Python 3.6

7. Expected Changes

- Add light history analysis function.
- Add monitor function.
- Adjust the brightness of the light
- Personal Web Pages for Skin Change
- Provide personalized web customization
- Provide hotline for maintenance personnel.
- Provide multilingual support.
- Retrievable password and change password at any time
- Support binding mobile phone number and login by phone number.

8. Appendices

8.1 Definitions and acronyms

8.1.1 Definitions

Keyword	Definitions
Raspberry Pi	A portable single-board computer

8.1.2 Acronyms and abbreviations

Acronym or Abbreviation	Definitions
GUI	Graphical User Interface
IC	Intelligence controller

8.2 References