Software Requirements Specification (SRS)

Revision History:

Date	Author	Description
2019.3.21	Zhi Zhou	Overall block diagram
2019.3.21	Zimu Hu	Edit functional documentation
2019.3.23	Zhi Zhou	Modify functional documentation
2019.3.25	Zhi Zhou	Add Server Logic into User Test Case
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 Requirments
2019.3.25	Renxiang Zhu	Integrate documents

1. Introduction

1.1 Intended Audience and Purpose

This document is for the customers and everyone who joins in this project. In this document, we will explain how every part of the system could work together. What users could do and what will happen. By using the Use Case, we want not only user but also every developer could know what they can do and what information they can get from the system. And if this document passed by everyone, all work should be finished follow it.

1.2 How to use the document

In this document, all the situations the users can faced to will be found. In the second part of this document, which is the Use Case's part, users can look up what they can do in what situations. And when users follow the Use Case, what will happen is written clearly. For every developer, what information and operations could other groups can provide for you is also said specifically. When the project finished, we will also use this document to check if all the requirements can be solved. And if everyone accepts what the document written, when the developers finish all the functions, the project will be finished completely.

2. System Capabilities

2.1 System Context

System requirements:

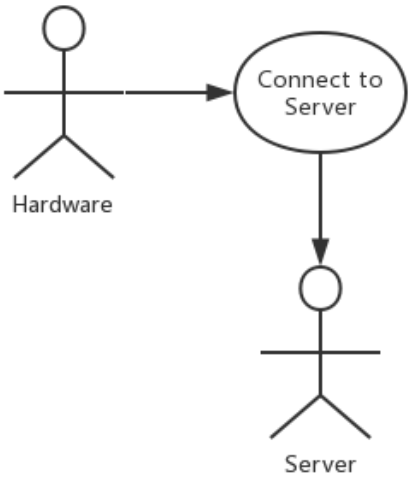
If you use web to login the system, you need a browser with Chrome or IE core. If you use APP to login in the system, you need a phone with android system.

2.2 System capabilities

Use Cases

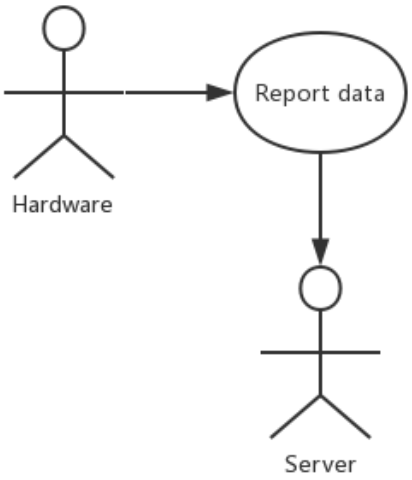
2.1.1. Hardware connects to server

Use Case	Hardware connects to server.	
Version	V1.0 Created	*2019.3.25*
Author	Zhi Zhou	
Source	Hardware	
Purpose		
Goals	Build connects between server and hardware.	
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	Hardware	Raise a connecting request.
2	Server	Authenticate hardware's key. (Move to alternate flow 1 when error)
3	Server	Authenticate whether hardware is registered in the database. (Move to alternate flow 1 when error)
4	Server	Build connection with Hardware.
Frequency		
Type	Primary	
Postconditions	Connection is built.	

Chart		
Alternate Flow	*Actor*	System
1	Server	Reject the connecting request.

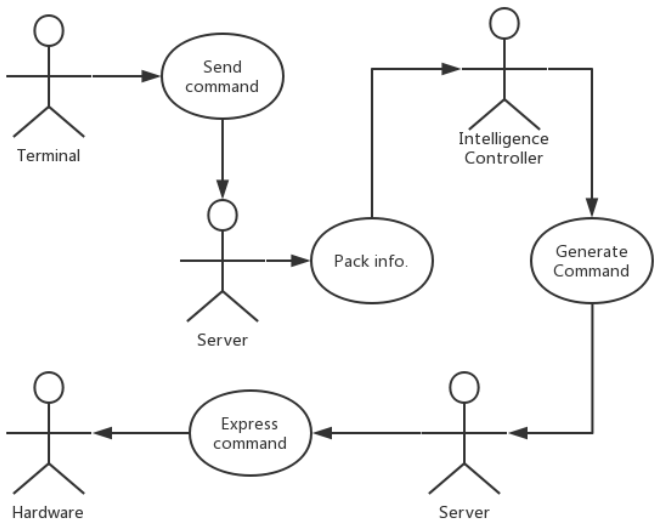
2.1.2. Hardware reports data

Use Case	Hardware reports data	
Version	V1.0 Created	*2019.3.25*
Author	Zhi Zhou	
Source	Hardware	
Purpose		
Goals	Report sensors' data to server	
Summary	Report sensors' data to server.	
Actors	Hardware	
Trigger	Sensors' data changed.	
Precondition	Connection is built.	
Basic Flow	*Actor*	System
1	Hardware	Send sensors' data to server through socket. (Move to alternate flow 1 when failed.)
2	Server	Record the data in memory.
Frequency		
Type	Primary	
Postconditions	Data is sent.	

Chart		
Alternate Flow	*Actor*	System
1	Hardware	Try to reconnect.

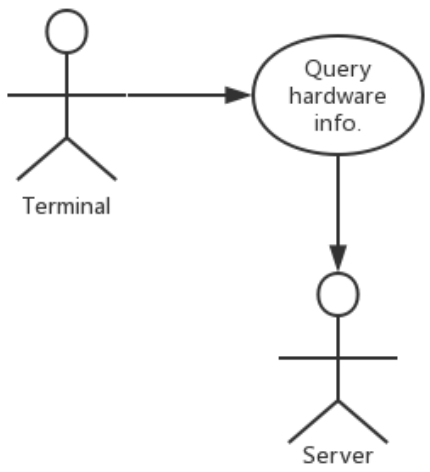
2.1.3. Terminal sends command

Use Case	Terminal sends command	
Version	V1.0 Created	*2019.3.25*
Author	Zhi Zhou	
Source	Terminal	
Purpose		
Goals	Give hardware the command after handled by intelligence controller.	
Summary	Server give intelligence controller the command submitted by the terminal. And then send the result generated by the intelligence controller to hardware.	
Actors	Terminal	
Trigger	Terminal sends command	
Precondition	Server and hardware is running	
Basic Flow	*Actor*	System
1	Terminal	Send command to server.
2	Server	Check user's authority. (Move to alternate flow 1 when failed.)
3	Server	Check whether the target is online. (Move to alternate flow 2 when target is offline)
4	Server	Pack necessary and related data, and send them to intelligence controller with command.
5	Intelligence Controller	Generate the command and return it to the server.
6	Server	Send command to hardware.
Frequency		

Type	Primary	
Postconditions	Hardware executed the command.	
Chart		
Alternate Flow	*Actor*	System
1	Server	Reject the command
2	Server	Tell terminal that the target is offline.

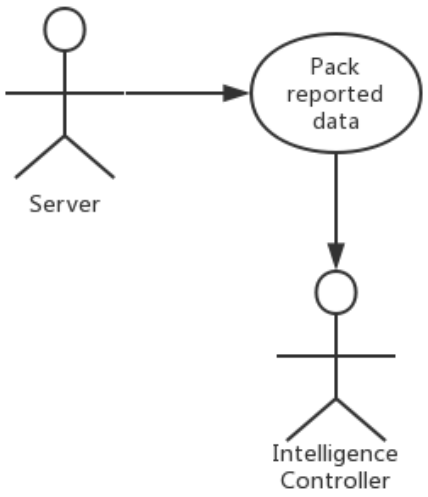
2.1.4. Terminal queries hardware's information

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

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

2.1.5. 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		
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	Server	Pack necessary and related data, and send
2	Intelligence Controller	Generate the command and return it to the server.
3	Server	Send command to hardware.
Frequency		
Type	Primary	
Postconditions	Hardware executed the command.	

Chart		
Alternate Flow	*Actor*	System

3. Detailed Requirements

3.1 System Inputs and Outputs

3.1.1 Inputs

The inputs send to the server when terminal 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 terminal 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 terminal 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.

3.1.2 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 terminal 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.

3.2 Detailed Output Behavior

4 Quality Requirements (Non-functional Requirements)

The system must show good behavior in many fields like Performance, Security, Availability, Reliability, Modifiability, Maintainability, Understandability.

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 jurisdiction. 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

Subsets one: Intelligent control technology interface module

This module is designed to connect with the raspberry pi which takes charge of the intelligent control of the whole light system. The server need to contact with the raspberry pi at any time.

Subset two: Server management module.

This module is in charge of the basic functions of the whole server.

Subset three: Hardware interface module.

Accept states from the hardware.

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.

8. Appendices

8.1 Definitions and acronyms

8.1.1 Definitions

Keyword	Definitions
Raspberry Pi	A kind of card computer

8.1.2 Acronyms and abbreviations

Acronym or Abbreviation	Definitions

8.2 References