**Software Requirements Specification (SRS)**

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 2019.3.25 | Li Yuanjin | Editing Software Requirements Specification |
| 2019.3.26 | Zhang Yifan | Editing the Detailed Requirments |
| 2019.3.26 | Wang Zhongyu | Editing the Quality Requirments |
|  |  |  |
|  |  |  |
|  |  |  |

**1.  Introduction**

**1.1    Intended Audience and Purpose**

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

Primary Customer - This page will detail all of the application 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.

User - A prospective user will be able to use this document to identify the main functionailty included in the application. Furthermore, the application will have a set of system requirements before the application can be run. Details regarding these requirements can be found here.

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. System Capabilities - 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 requirments - 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  
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. Apendicies - 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  
    8.1.1 Definitions - Definitions of technical or unusual terminology  
    8.1.2 Acronyms and Abreviations - Any abreviated 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**

This product is a control system for intelligent lighting systems which developed for university classrooms. The system can realize the control of the switch state of the lights in the classroom by analysis according to the information of the light sensor and the infrared sensor sent to the server and the received network request.

**2.1    System Context**

The system needs to be used on the PC side, so the system environment applied is as follows:

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)
* RAM: 128 MB
* Disk space: 128 MB
* Processor: Minimum Pentium 2 266 MHz processor

Mac OS X:

* Intel-based Mac running Mac OS X 10.8.3+, 10.9+
* Administrator privileges for installation

Linux:

* Oracle Linux 5.5+1
* Oracle Linux 6.x (32-bit), 6.x (64-bit)2
* Oracle Linux 7.x (64-bit)2 (8u20 and above)
* 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**

The control system can be divided into two main modes: automatic control mode and human command mode. In the automatic control mode, the system can automatically determine whether it is necessary to change the state of the lamp and execute it by acquiring sensor information. In the human command mode, the system performs different operations according to the authority of the command. When the authority level of the instruction is high, the instruction can be executed, however, when the authority level of the instruction is low, in order to prevent the misoperation behavior, the situation is judged by acquiring the sensor information to intelligently control the light in the room.

**2.3 Use Cases**

**2.3.1  Automatic light-off mode**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | *Automatic light-off mode* | | |
| Version | 1.0 | Created | *2019-3-25* |
| Author | Li Yuanjin | | |
| Source | *Requirement* | | |
| Purpose | power saving intelligently | | |
| Goals |  | | |
| Summary | Under the automatic mode, no one in the house for more than 1 minute, turn off the light | | |
| Actors | IC | | |
| Trigger | *Light sensor gets no signal for more than 1 minute* | | |
| Precondition | Automatic mode | | |
| Basic Flow | *Actor* | | System |
| 1 |  | | Light sensor gets no signal for more than 1 minute ,Sever sends a package to IC |
| 2 | IC give light a signal to turn off | |  |
| 3 |  | |  |
| Frequency |  | | |
| Type | Primary | | |
| Posconditions | The project assigment is created | | |
| Chart | Server  Light  Intelligent  Control  Light Sensor | | |
| Alternate Flow | *Actor* | | System |

 **2.3.2  Automatic light-on mode**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | *Automatic light-on mode* | | |
| Version | 1.0 | Created | *2019-3-25* |
| Author | Li Yuanjin | | |
| Source | *Customer* | | |
| Purpose | Turn on the light | | |
| Goals |  | | |
| Summary | Under the automatic mode, someone presses the button, turn on the light. | | |
| Actors | IC | | |
| Trigger | Someone presses the button | | |
| Precondition | Automatic mode | | |
| Basic Flow | *Actor* | | System |
| 1 |  | | Sever give IC a signal that button is activated |
| 2 | IC give light a signal to turn on | |  |
| 3 |  | |  |
| Frequency |  | | |
| Type | Primary | | |
| Posconditions | The project assigment is created | | |
| Chart | Light  Button  Control System | | |
| Alternate Flow | *Actor* | | System |

**2.3.3  Command-light mode**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Use Case | | *Command-light mode* | | | | |
| Version | | 1.1 | | Created | *2019-3-31* | |
| Author | | Zhang yifan | | | | |
| Source | | *Requirement* | | | | |
| Purpose | | turn the light on or off correctly | | | | |
| Goals | |  | | | | |
| Summary | | A user issues a command to change the light through the Server, then the Intelligent Control(our system) make a judgement and return the correct answer. | | | | |
| Actors | | Intelligent Control system | | | | |
| Trigger | | *Light sensor* | | | | |
| Precondition | |  | | | | |
| Basic Flow | | *Actor* | | | System | |
| 1 | | Server: Send instruction to change the state  of the light | | |  | |
| 2 | |  | | | Intelligent Control: Return ‘open’ or ‘close’  or ‘not change’ or ‘exception’  or ‘exception’ | |
| 3 | |  | | |  | |
| Frequency | |  | | | | |
| Type | | Primary | | | | |
| Posconditions | | The project assigment is created | | | | |
| Chart | | IC  Server | | | | |
| Alternate Flow | | *Actor* | | | System | |

**2.3.4  Time setting mode**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | *Time setting mode* | | |
| Version | 1.1 | 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 |  | | |
| Summary | A administrator issues a command to change the time periods through the Server,  then the Intelligent Control(our system) make a judgement based on the data from Data Base, and return the answer. | | |
| Actors | Intelligent Control system, App/Web, Server, Data Base | | |
| Trigger | Data Base | | |
| Precondition | The command came from a administrator. | | |
| Basic Flow | *Actor* | | System |
| 1 | App/Web :a Command to set the time period | |  |
| 2 |  | | Server: ask and get data from DB |
| 3 | Server: send data to Intelligent Control | |  |
| 4 |  | | Intelligent Control: Make decision and send answer to Server |
| 5 | Server: send data to Data Base | |  |
| 6 |  | | Data Base: save the new settings |
| Frequency |  | | |
| Type | Primary | | |
| Posconditions | The project assigment is created | | |
| Chart | Server  APP/WEB  IC  DB | | |
| Alternate Flow | *Actor* | | System |

**2.3.5 Rules setting mode**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | *Rules setting mode* | | |
| Version | 1.1 | Created | *2019-3-31* |
| Author | Zhang yifan | | |
| Source | Requirement | | |
| Purpose | (The administrator) Set the rules of our system, including priority and orders | | |
| Goals |  | | |
| Summary | A user issues a command to change the rules through the Server, then the  Intelligent Control(our system) make a judgement based on the data from Data Base, and return the correct answer. | | |
| Actors | Intelligent Control system, App/Web, Server, Data Base | | |
| Trigger | Data Base | | |
| Precondition | The command came from a administrator. | | |
| Basic Flow | *Actor* | | System |
| 1 | App/Web :a Command to set the rules | |  |
| 2 |  | | Server: ask and get data from DB |
| 3 | Server: send data to Intelligent Control | |  |
| 4 |  | | Intelligent Control: Make decision and send new rules to Server |
| 5 | Server: send data to Data Base | |  |
| 6 |  | | Data Base: save the new settings |
| Frequency |  | | |
| Type | Primary | | |
| Posconditions | The project assigment is created | | |
| Chart | Server  APP/WEB  IC  DB | | |
| Alternate Flow | *Actor* | | System |

**3.    Detailed Requirements**

**3.1 System Inputs and Outputs**

This system will automatically ask for data about the sensors and lights from the Server in cycles.

And when a instruction arrives at the Server, it will sends data to our system, including the state of the light, the the relevant sensors’ data, the user’s information and the instruction.  
The sensors now have three kinds, which are the movement sensor , the luminance sensor and the button sensor.

**3.1.1 Inputs**

\*Light state: One input required, 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: Every sensor has three input{kind, online, value}, the first one - kind is listed as follows:

{

\*kind: it is a string type, has three values in{Movement, Luminance, Button}

\*online: it is a boolean type, whose value is true or false. True means the sensor is online, while false

means the opposite.

\*value: {

It is a numerical type.

For the “Luminance” the brighter the room, the lager the value.

For the “Movement”, if there are someone in the room, the value will be larger than zero, so if no one in the room, the value is zero.

For the “Button”, if the user wants to change the light’s state, the value will be larger than zero, so if no one in the room, the value is zero.

}

\*User information: At least one input required, a numerical type means user’s priority{1, 2, 3}. The priority has 3 levels that level\_1 means the lowest priority for students, level\_2 means the middle priority for the button sensor and level\_3 means the highest priority for teachers and administrators.

\*Instruction information: At least one input required, a boolean type. True means that the instruction would like to open the light, while false means the opposite.

**3.1.2 Outputs**

This system will send its output to the Server.

\*Result: One output required, it can be 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.

**4   Quality Requirements (Non-functional Requirements)**

**Behavioral (run-time) qualities:**

* Performance

the elapsed time between the IC receive the command from server and the IC

system’s response must be <= 100 ms

* Security

No body can change the rule except administrator

* Reliability

The IC can ignore those commands illegal or too often such as

1. Students want to turn on the light when there is nobody in the room.
2. Students want to turn on the light when the light is enough.
3. IC received more than one command during 1 second.
4. IC received the student’s command after teacher’s command one minute later.(the IC must set how long does each command can effect or shield other commands)

* Availability

If receives an illegal command, IC should return the error information to the server such as :

1. Condition error: students want to turn off the light but somebody is in the room.(there is confliction between students’ commands and sensors’ information)
2. Authority error: students want to turn off the light during the specific time set by administrator.

**Developmental (design-time) qualities:**

* Modifiability
* Maintainability
* Portability
* Reusability
* Understandability

**5. Expected Subsets**

The core of this intelligent control system is to be able to analyze the priority of turning on/off the light according to the authority and sensor status.

**6.    Fundamental Assumptions**

The application can run on any system.

Developing Software: not sure.

Developing Environment: Raspberry Pi

**7.    Expected Changes**

7.1.We expect that the system can be expended to serve for multiple rooms and multiple lights. We can group rooms and light to fit our needs in different situation.

**8.    Appendices**

**8.1    Definitions and acronyms**

**8.1.1    Definitions**

|  |  |
| --- | --- |
| **Keyword** | **Definitions** |
|  |  |
|  |  |
|  |  |
|  |  |

**8.1.2    Acronyms and abbreviations**

|  |  |
| --- | --- |
| **Acronym or**  **Abbreviation** | **Definitions** |
| IC | Intelligent Controller |
|  |  |
|  |  |

**8.2    References**