C. Report No.3 Software Requirement Specification

1. User Requirement Specification

Nowadays, there are many reports about the unhealthy side-effects found in the foods we buy and eat. Large food which is introduced by chemical to grow food quicker and use pesticides to prevent loss from being destroyed by worms. In this country, it’s cheaper for a family to eat fast food or meats than it is to eat organically grown fruits and vegetables. In addition to the foods, we are developing industry and modern too fast with a lot of harmful effects for environment, air quality as poor as less space to relax after work at home. A small garden in house which is not only makes your house look good but they also keeps you healthy with fresh air and safe fruit or vegetables; this is also the best place to reduce stress and lighten mind which cannot be done being inside four walls. But we have to face too much pressure and differences works in day, this make us have less time to take care a garden day by day. Come from real needs, users want a system which help them to take care a small garden in house less time, smarter and flexibility. The system should meet the below needs:

1.1. Monitoring the status of garden

Users can tracking indexes of garden through sensors.

Users can monitor the status of garden from anywhere in anytime.

**1.2. Response with expected problems**

User can be notified when unexpected problems occurred of weather with their garden (rain, high temperature ...).

System can automatically react with problems which is harmful for garden.

**1.3. Manage the system**

Users can control one or multi devices to solve some specific tasks such as limit volume water for some kind of plant in blooming lately.

Users can manage all devices, tracking reliability and durability of devices; add, remove or configure a specific devices.

Users can start or stop any automatically actions in their garden

**1.4. Schedule**

Users can create a plan measurements indexes from sensors.

Users will be received auto schedule after processing indexes on server.

Users need to be notified some reminder with situations which meet plant’s requirement or auto actions deadline nearly.

**1.5. Consumption and durability**

All sensors which used in the system, have to a long life low power.

All sensors have a great durability when usually contact with water day by day.

**1.6. High security**

Users have to be authenticated before using the system

Data and commands which transmit in system via RF and Wi-Fi should be protected.

**1.7. Interoperability**

New end devices can be added to the system and interact with other ones regardless of manufacturing origin such as sensors, pump, traction motor.

2. System Requirement Specification

2.1 External Interface Requirement

2.1.1. User interface

The interface must be designed to be satisfied the following requirement:

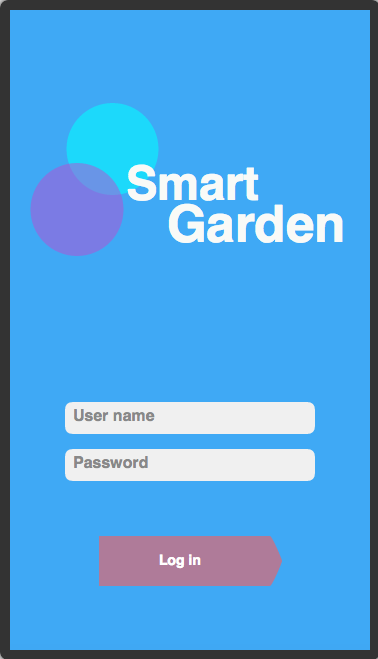
The interface is divided by tabs, which will allow users to easily switch between different parts of the program.

Be simple and user-friendly.

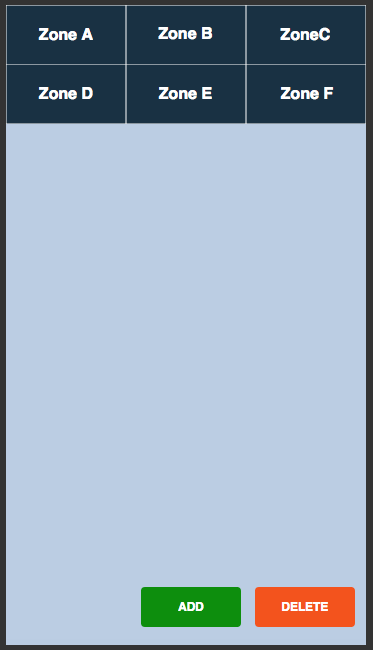
Meet all the main functions and easily to identify each of functions.

Use obvious icons to avoiding misunderstanding.

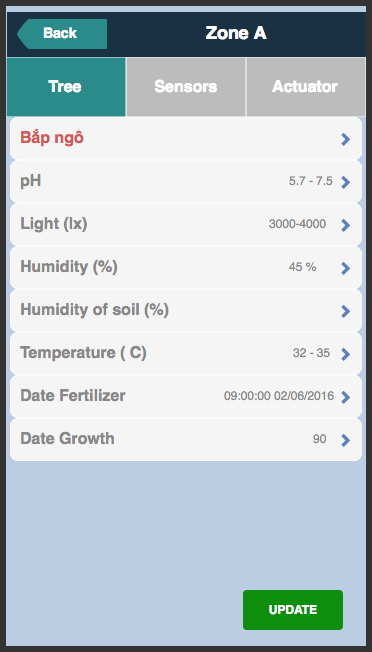
User interface use Graphical User Interface must be simple, clear, and easy to use.

**2.1.1.1. Login**

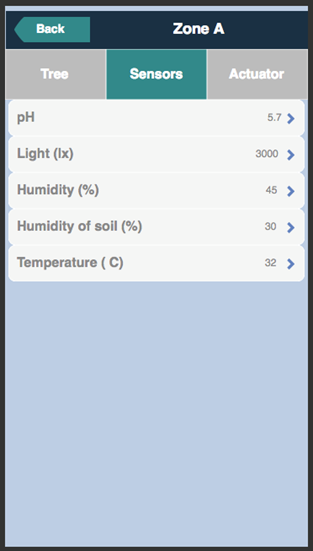
**Figure 3. “Login” screen**

**2.1.1.2. Zone management**

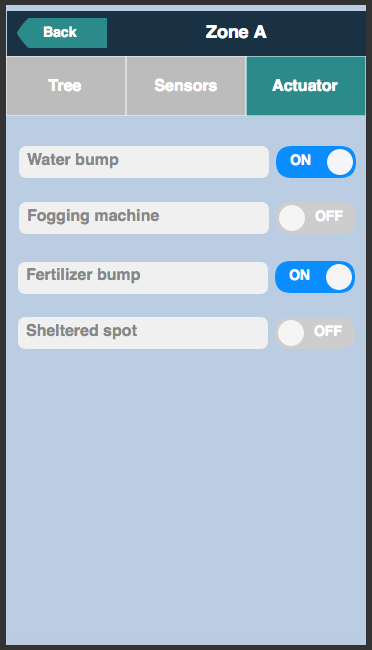
**Figure 4. “Zone management” screen**

**2.1.1.3. Tree information**

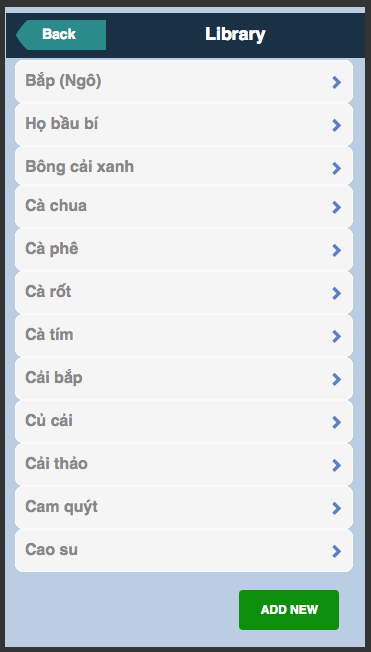
**Figure 5. “Tree information” screen**

**2.1.1.4 Sensors management**

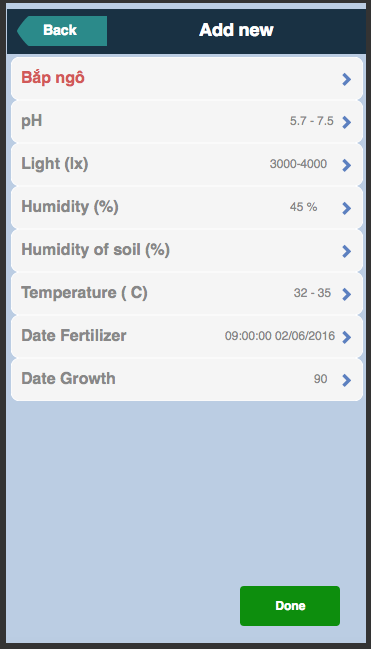
**Figure 6. “Sensors information” screen**

**2.1.1.5. Actuator management**

**Figure 7. “Actuator management” screen**

**2.1.1.6. Library**

**Figure 8. “Library” screen**

**2.1.1.7. Add new**

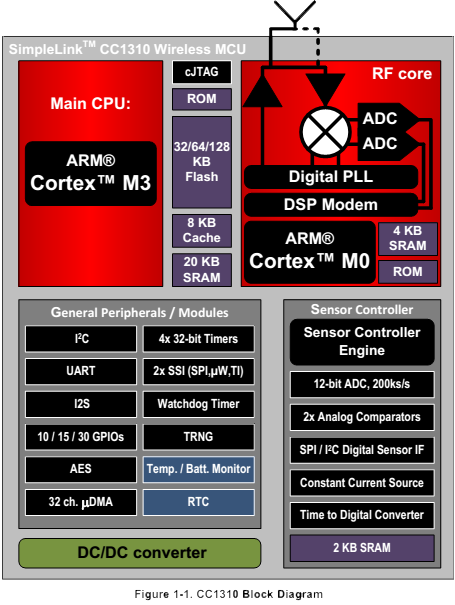
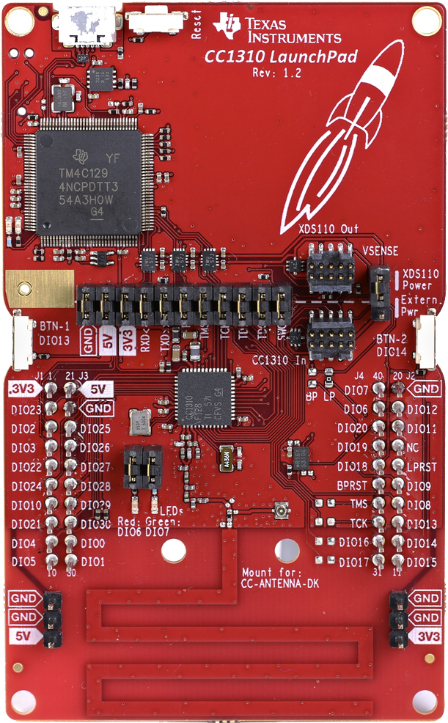
**Figure 9. “Add new” screen**

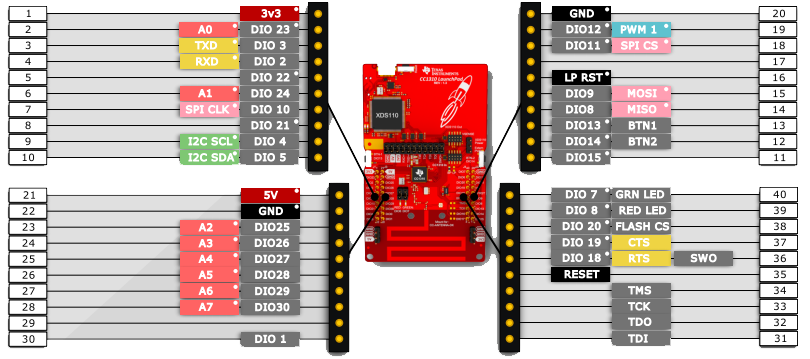
2.1.2. Hardware interface

The system use variety sensors which must transmit data via RF to Controller to process and update via Wi-Fi to Internet where users can tracking system with mobile. So the hardware interface that the system using must be designed to be satisfied the following requirement:

* Low cost hardware sensors module which have reliability data and long endurance.
* Strong MCU have multi communication with sensors like I2C, SPI, UART; multi connection in system via RF, Bluetooth and Wi-Fi.
* Easy replace a node or a parts that not affect other parts in system

2.1.2.1. Device MCU – CC1310 Launchpad (Texas Instruments)





* ***Microcontroller***

- ARM Cortex M3

- EEMBC CoreMark Score: 142\* - EEMBC ULPBench Score: 158\*

- Up to 48MHz Clock Speed

- 128KB of In-System Programmable Flash

- 8KB of SDRAM for Cache

- 20KB of Ultralow Leakage SRAM

- 2-Pin cJTAG and JTAG Debugging

- Supports Over-the-Air Upgrade (OTA)

* ***Ultralow Power Sensor Controller***

- Can Run Autonomous From the Rest of the System

- 16-bit Architecture - 2KB of Ultralow Leakage SRAM for Code and Data

* ***Peripherals***

- All Digital Peripheral Pins Can Be Routed to Any GPIO

- Four General-Purpose Timer Modules (Eight 16-Bit or Four 32-Bit Timers, PWM Each)

- 12-Bit ADC, 200 ksamples/s, 8-Channel Analog MUX

- Continuous Time Comparator

- Ultralow Power Clocked Comparator

- Programmable Current Source

- UART – 2× SSI (SPI, MICROWIRE, TI) – I2C – I2S

- Real-Time Clock (RTC)

- AES-128 Security Module

- True Random Number Generator (TRNG)

- Support for Eight Capacitive Sensing Buttons

- Integrated Temperature Sensor for ARM

* ***Low Power***

- Wide Supply Voltage Range: 1.8 to 3.8 V

- Active-Mode RX: 5.5 mA

- Active-Mode TX at +10 dBm: 12.9 mA

- Active-Mode MCU 48 MHz Running Coremark: 2.5 mA (51 µA/MHz)

- Active-Mode MCU: 48.5 CoreMark/mA

- Active-Mode Sensor Controller at 24 MHz: 0.4 mA + 8.2 µA/MHz

- Sensor Controller, One Wake Up Every Second Performing One 12-Bit ADC Sampling: 0.85 µA

- Standby: 0.6 µA (RTC Running and RAM and CPU Retention)

- Shutdown: 185 nA (Wakeup on External Events)

* ***RF Section***

- Excellent Receiver Sensitivity –124 dBm using Long-Range Mode, –110 dBm at 50 kbps

- Excellent Selectivity: 52 dB

- Excellent Blocking Performance: 90 dB

- Programmable Output Power up to +14 dBm

- Single-Ended or Differential RF Interface

- Suitable for Systems Targeting Compliance with Worldwide Radio Frequency Regulations

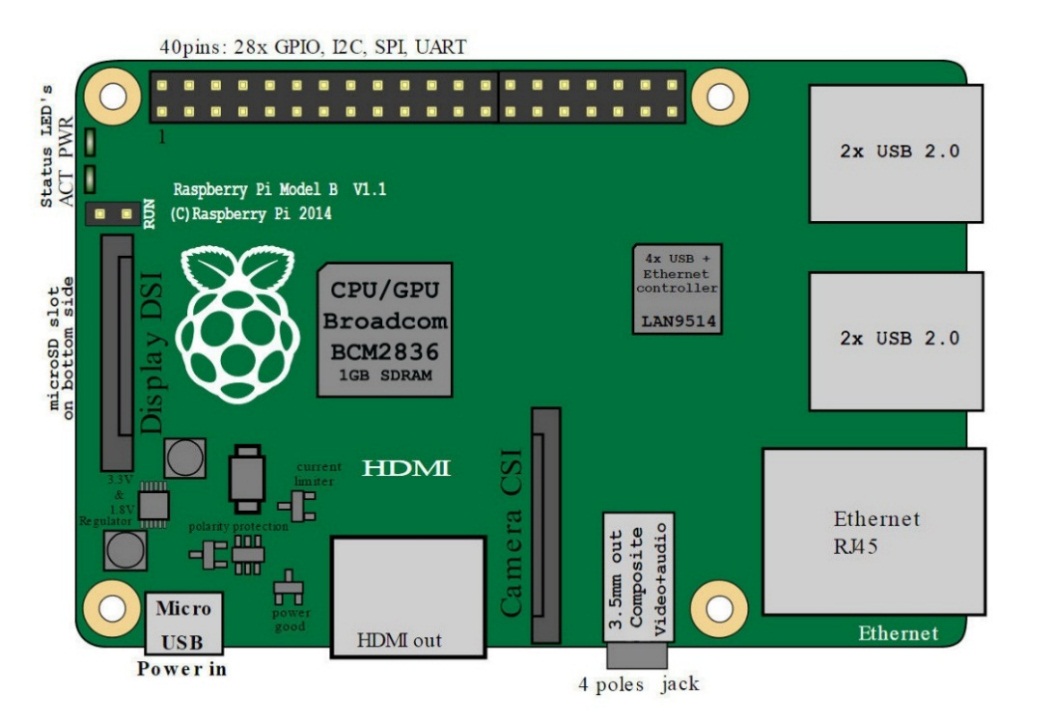
• ETSI EN 300 220, EN 303 131, EN 303 204 (Europe)

• FCC CFR47 Part 15 (US)

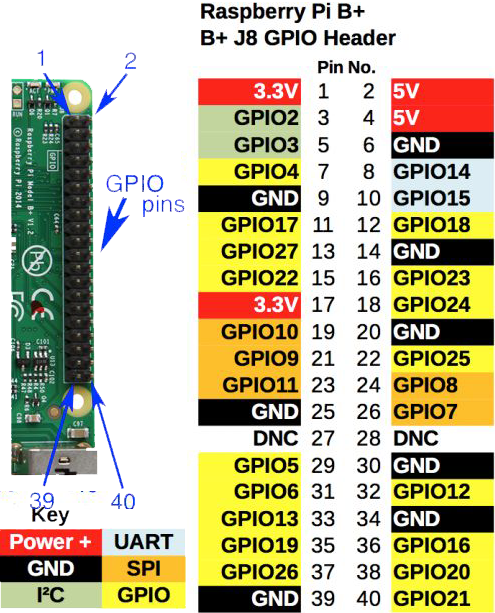
• ARIB STD-T108 (Japan)

- Wireless M-Bus and IEEE 802.15.4g PHY

2.1.2.2. Device MCU – Raspberry Pi 3 Mode B

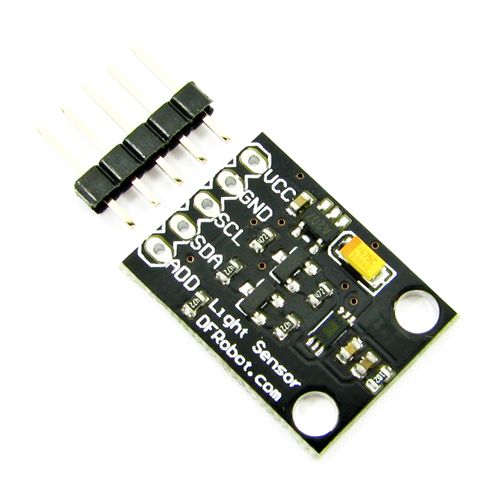


* *A 1.2GHz 64-bit quad-core ARMv8 CPU*
* *802.11n Wireless LAN*
* *1GB RAM*
* *4 USB ports*
* *40 GPIO pins*
* *Ethernet port*
* *Micro SD card slot (now push-pull rather than push-push)*

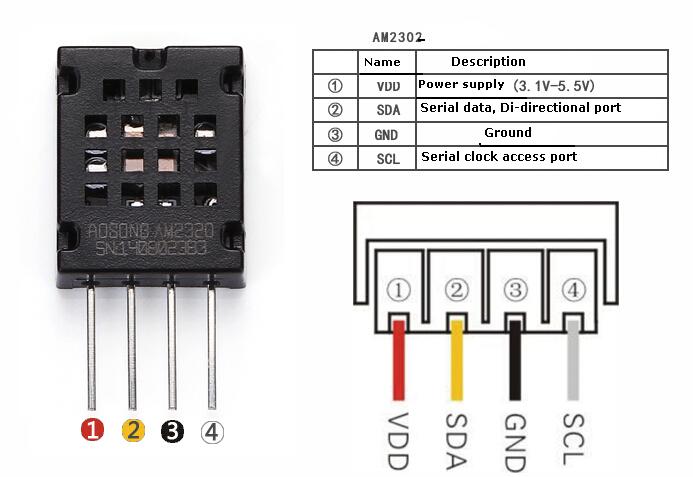


2.1.2.3. Sensors

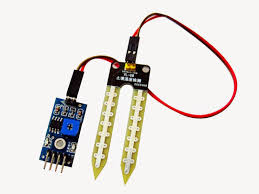
*+ Light sensor:* which can get intensity of light around in a zone. Recommend: module BH1750 FVI

* Power Supply: 3.3V - 5V
* Light Range：0 - 65535 lx(Lux)
* Sensor Built-in: 16 bit AD converter
* Photo Diode
* SDA: Serial data, Di-directional port
* GND: Ground
* SCL: Serial clock access port
* Protocol: I2C

+ *Temperature and Humidity sensor:* get indexes of temperature and humidity in air. Recommend: AM2320

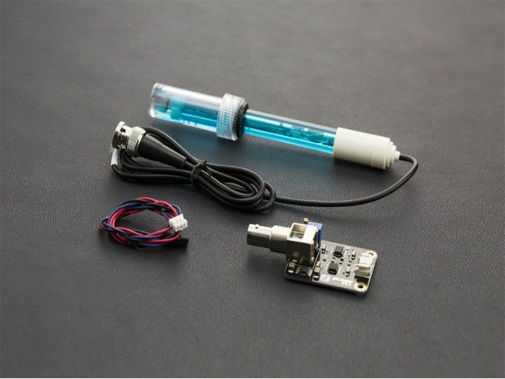
* Power supply: 3.1 V – 5.5 V
* SDA: Serial data, Di-directional port
* GND: Ground
* SCL: Serial clock access port
* Protocol: I2C

+ *Soil moisture sensor:* get index about humidity, water in soil



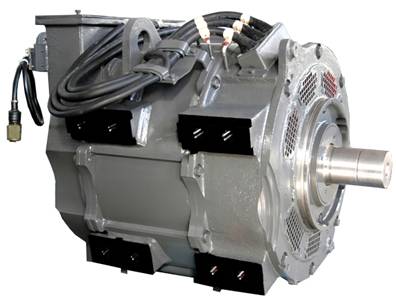
* Power input: VCC 5V
* Output: Analog, Digital

+ *PH sensor:* get index of pH degree in water

* Power input: VCC 5V
* Output: Analog, Digital

2.1.2.4. Actuators

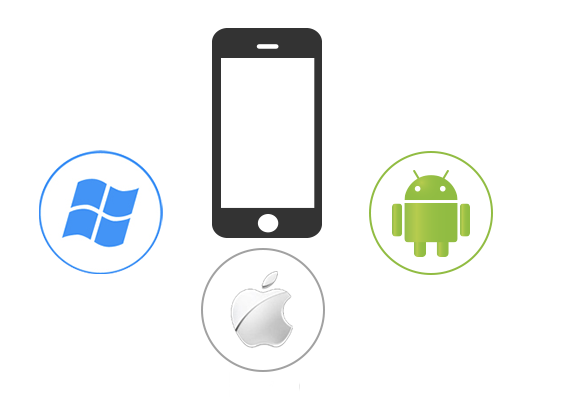
The system must control actuators such as water pump, traction motor, and nebulizer which are used with 220V AC with relay.





2.1.2.5. Device users

The system is managed by user via mobile, so hardware interface for users is most popular in mobile market such as mobile using android, iOS or windows phone. We are recommend android mobile.



2.1.3. Software Interface

- Mobile Application: Android OS (v4.0 or above)

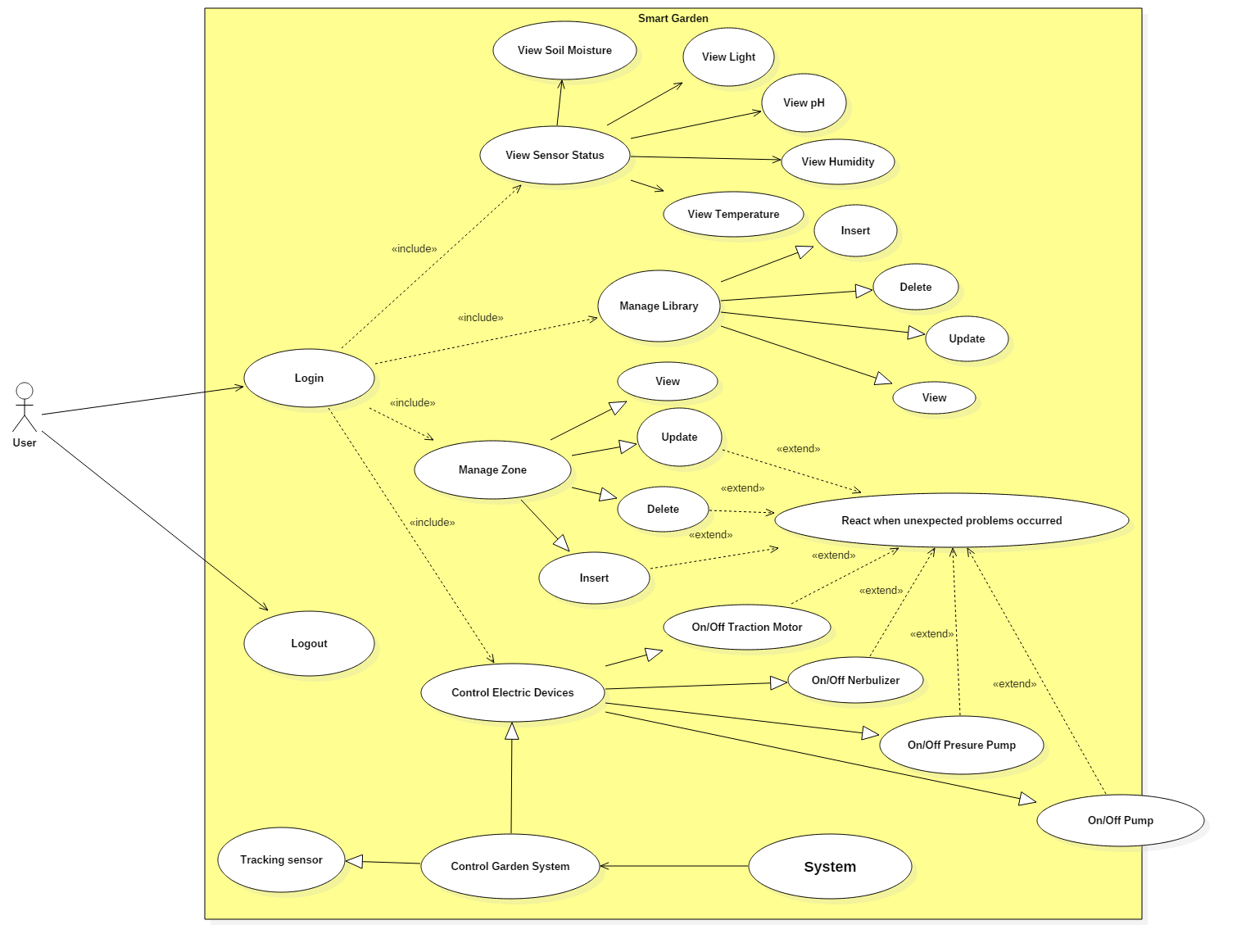
- Java Web server: Springs & Hibernate

- Database: MySQL

2.1.4. Communication Protocol

The system will applied to variety kinds of garden, small or large garden; it must satisfied distance communication in a garden. Users need a simple system which not affect too much garden’ space so electric wired between devices, so a wireless system is recommend with Wi-Fi and RF waves.

The device MCU in system have many kinds of protocol such as I2C, SPI, and UART to communicate with sensors. We recommend use I2C protocol to control easily and add more sensors to the systems.2.2 System Overview Use Case



**Figure 11. System overview use case**

2.3. List of Use Case

2.3.1. <User> Login

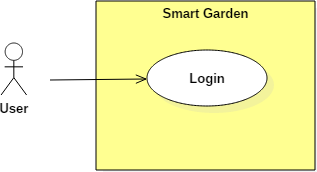


Figure 12. “Login” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC01** | | | |
| **Use Case No.** | **UC01** | **Use Case Version** | 2.0 |
| **Use Case Name** | Login | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - To manage and control the system, user must login into the system  **Goal:**  - Guest will become user to use function in the system after they login.  **Triggers:**  - User must input username and password which was provided and press button “Login”  **Preconditions:**  - User must have provided account from system.  **Post Conditions:**  - Success: User will moves to main page after login  - Fail: Display a popup message for errors  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access mobile application | Display Login Screen | | 2 | Fill 2 fields with username and password which was provided | Check validation of username and password | | 3 | Press “Login” button | Check login data in database  Log user into the system  Move to main page |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access mobile application | Display Login Screen | | 2 | Not fill enough fields or wrong format of textbox | Display "Username or password is wrong" message. | | No | Actor Action | System Response | | 1 | Access mobile application | Display Login Screen | | 2 | Fill invalid provided account’ information | Check validation of username and password | | 3 | Press “Login” button | Display "Invalid username or password" message. |   **Relationships:**  - NA | | | |

Table 9. “Login” specification.

2.3.2. <User> Library Overview Use Case

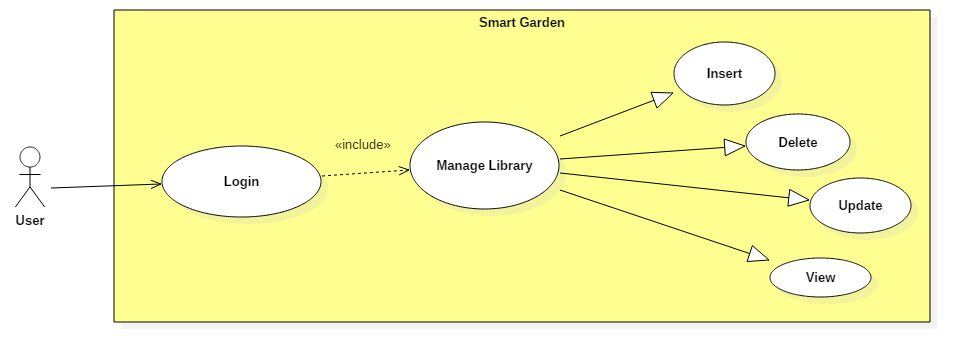


Figure 13. Library Overview use case

2.3.2.1 <User> Library View

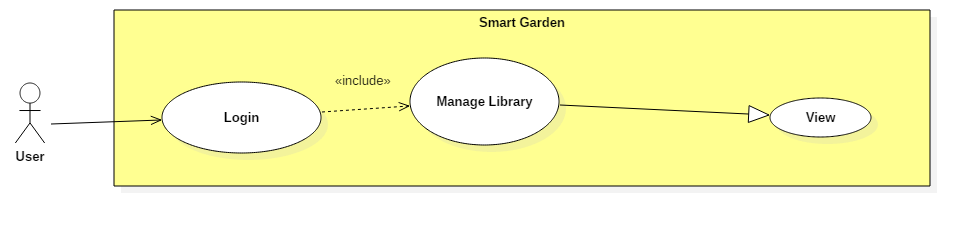


Figure 13.1. “Library View” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC02** | | | |
| **Use Case No.** | **UC02** | **Use Case Version** | 2.0 |
| **Use Case Name** | Library View | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User need to understand standard information of plants to manage system based on indexes of sensors. Library give user recommend information of tree.  **Goal:**  - User can understand easily about information of plants which they don’t need to find in other places outside system.  **Triggers:**  - NA  **Preconditions:**  - System must provide information on core database which join with a personal database for each user.  **Post Conditions:**  - Success: User can view list of plants which is on core database and personal database.  - Fail: Display a popup message for errors. Log errors to system in case conflict between personal database and core database.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Library page | Get Joined database between core database and personal database.  Show list of plants | | 2 | User choose one plant which they want | Navigate to Information of Plants  Show standard information which is load from core database |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Library page | Get Joined database between core database and personal database.  Show list of plants | | 2 | User touch “Add” button | Navigate to Add new page |  |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Library page | Get Joined database between core database and personal database.  Show list of plants | | 2 | User touch and hold on one or many kind of tree | Choose plants which user pressed | | 3 | User touch “Delete” button | Delete plants which user pressed  Update to database |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Errors from join database or system  Log errors. | | 2 | Waiting for information from database | Display errors message |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | User choose one plant which they want | Can’t get information of that plant | | 3 | Waiting for information | Display errors message |   **Relationships:**  - N/A | | | |

Table 10. “Library View” specification.

2.3.2.2 <User> Library Update

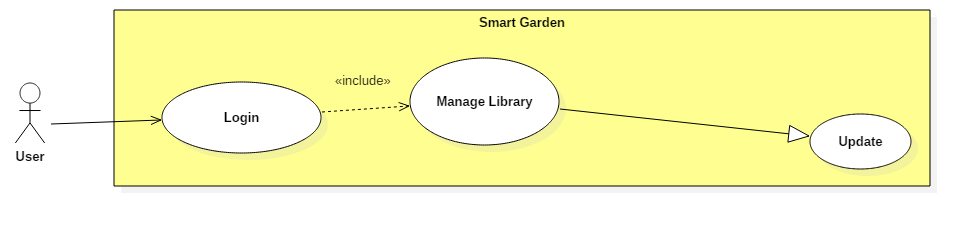


Figure 13.2. “Library Update” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC03** | | | |
| **Use Case No.** | **UC03** | **Use Case Version** | 2.0 |
| **Use Case Name** | Library Update | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User want to change standard information of specific plant. This use case allow user to change some information which is different with core database  **Goal:**  - System can work based on indexes which user changed. User can change threshold for sensors in system with their wish.  - User can change some indexes which they want. That help people have a real control with system to suitable with different conditions of garden.  **Triggers:**  - NA  **Preconditions:**  - Information which will change by user is exist on core database.  - The changed of user is not affect to data on core database, it will be added to personal database  **Post Conditions:**  - Success: User change data successful. Changed data is not affect to core database.  - Fail: Display a popup message for errors. Log errors to system in case conflict between personal database and core database.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch on one plant which they want | Navigate to Information of Plants  Show standard information which is load from core database | | 3 | Touch on each field of indexes and edit information | Validation indexes in fields. | | 4 | Touch “Done” button | Update to database  Display a message “Update Successful” |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | User touch and hold on an object | Turn on Delete button and allow to choose more than one object | | 3 | Touch on Delete button | Delete objects which user chose |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | User choose one plant which they want | Navigate to Information of Plants  Show standard information which is load from core database | | 3 | Touch on each field of indexes and edit information.  Input wrong data types of field | Validation data which user input  Display an error message. |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | User choose one plant which they want | Navigate to Information of Plants  Show standard information which is load from core database | | 3 | Touch on each field of indexes and edit information | Validation indexes in fields. | | 4 | Touch “Done” button | Update to database fail  Display a message “Update Fail” |   **Relationships:**  - N/A | | | |

Table 11. “Library Update” specification.

2.3.2.3 <User> Library Insert

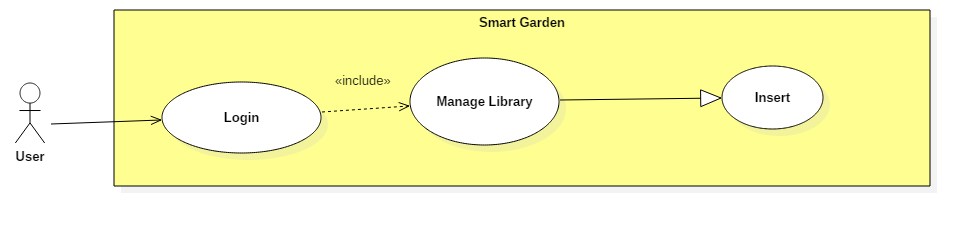


Figure 13.3. “Library Insert” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC04** | | | |
| **Use Case No.** | **UC04** | **Use Case Version** | 2.0 |
| **Use Case Name** | Library Insert | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - Allow user can add more to their database (personal) with some kind of plants which they want  **Goal:**  - User can control the system with any kind of plant, there are not depend too much on core database of system  **Triggers:**  - NA  **Preconditions:**  - System must provide a personal database for each user to avoid conflict between difference users on core database.  **Post Conditions:**  - Success: User can add any plants which they want to personal database which system can work on. Users set threshold of sensors to control actuators by themself.  - Fail: Display a popup message for errors. Log errors to system in case conflict between personal database and core database.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Library page | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch “Add” button | Navigate to Add new screen with fields which ask to input full data on | | 3 | User fill all fields in screen | Validation data which user input | | 4 | User touch “Add” button | Insert new data to database.  Display message “Add successful” |   **Alternative Scenario:**  - NA  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library page | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch “Add” button | Navigate to Add new screen with fields which ask to input full data on | | 3 | Users don’t fill enough fields in screen or input wrong types of data | Validation data which user input  Display message errors to user’s screen |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library page | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch “Add” button | Navigate to Add new screen with fields which ask to input full data on | | 3 | Waiting for information | Display errors message | | 4 | User touch “Add” button | Insert to database fail  Log errors  Display message error to user |   **Relationships:**  - N/A | | | |

Table 12. “Library Insert” specification.

2.3.2.4 <User> Library Delete

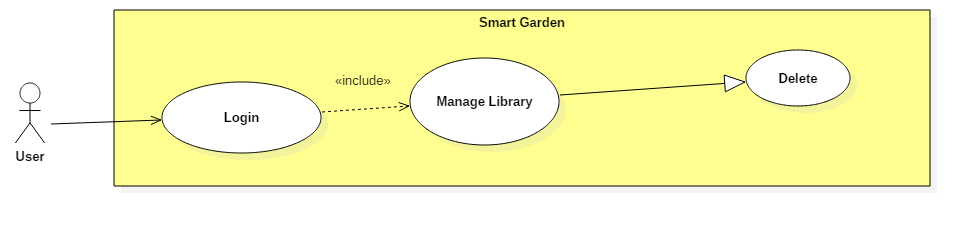


Figure 13.4. “Library Delete” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC05** | | | |
| **Use Case No.** | **UC05** | **Use Case Version** | 2.0 |
| **Use Case Name** | Library Delete | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - Allow user to delete one or many plants’ information in personal database  **Goal:**  - User can delete any data of plants which they want. That is not affect to core database of all system.  **Triggers:**  - NA  **Preconditions:**  - Information which will delete by user is exist.  **Post Conditions:**  - Success: User can delete one or many plants which they want.  - Fail: Display a popup message for errors. Log errors to system in case conflict between personal database and core database.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch and hold on one object plant in list | Turn on Delete button and allow to choose more than one object | | 3 | User touch on Delete button | Update to database to delete one or many object which user chose.  Update list of plants in Library Screen |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | User touch on one object of plant | Navigate to Library View Screen |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch and hold on one object plant in list | Turn on Delete button and allow to choose more than one object | | 3 | User touch on Delete button | Fail when update to database to delete one or many object which user chose.  Display a message error to user “Delete unsuccessful” |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access Library screen | Get Joined database between core database and personal database.  Show list of plants | | 2 | Touch and hold on one object plant in list | Turn on Delete button and allow to choose more than one object | | 3 | User touch on Delete button | Update to database to delete one or many object which user chose.  Fail when update list of plants in Library Screen  Display a message error to user |   **Relationships:**  - N/A | | | |

Table 13. “Library Delete” specification.

2.3.3. <User> View Status Use Case

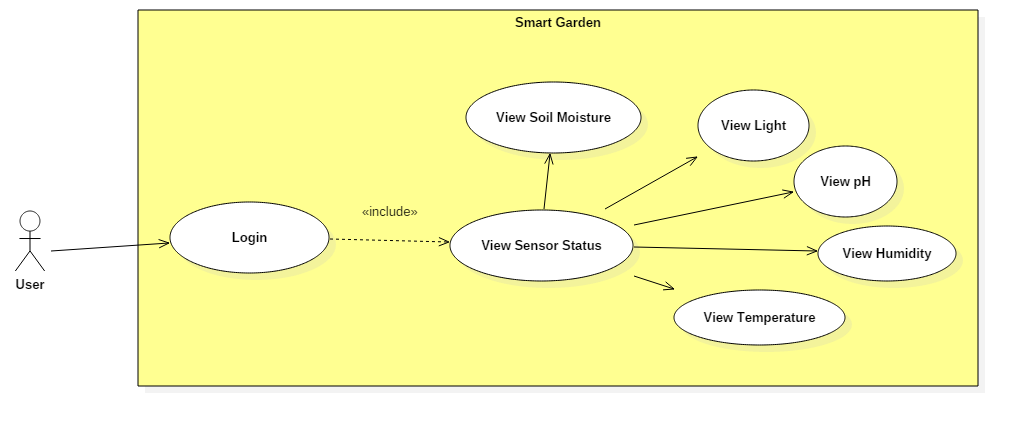


Figure 14. <User> View Status use case

2.3.3.1. View Temperature

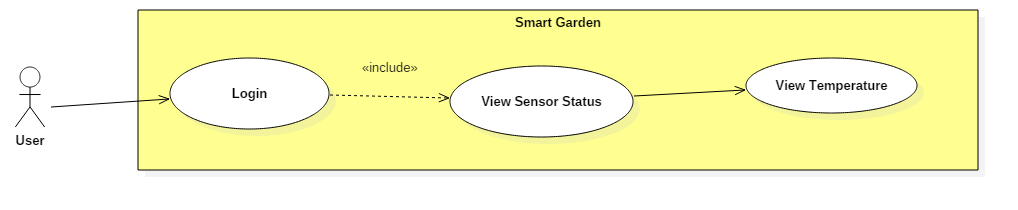


Figure 14.1. “View Temperature” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC06** | | | |
| **Use Case No.** | **UC06** | **Use Case Version** | 2.0 |
| **Use Case Name** | View Temperature | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User can monitor temperature via web server and mobile application.  **Goal:**  - Users can view temperature every time when they access to system  **Triggers:**  - After user logged to the system and touch on “Sensor” tab.  **Preconditions:**  - User must logged into the system.  - Temperature sensors must be connected to the system.  **Post Conditions:**  - Success: Information must be displayed in real time  - Fail: Display a message error to user, resent command to check status and ask user want to get indexes of temperature sensor again.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Temperature” line | Send command to get new index from temperature sensor  Display a message with new index |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on line of other sensors | Send command to get new index from other sensors  Display a message with new index |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Temperature” line | Fail when send command to get new index from temperature sensor  Display a message error about connect between sensor and controller. |   **Relationships:**  - N/A | | | |

Table 14. “View Temperature” specification.

2.3.3.2. View Humidity

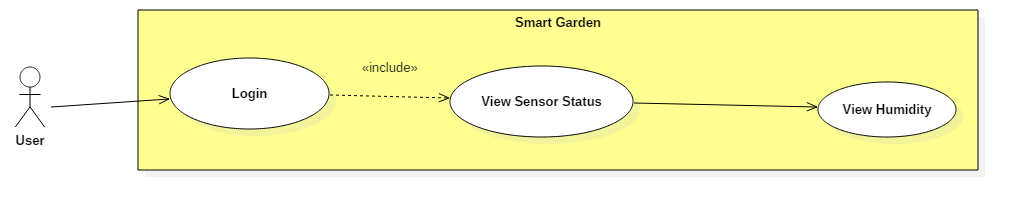


Figure 14.2. “View Humidity” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC07** | | | |
| **Use Case No.** | **UC07** | **Use Case Version** | 2.0 |
| **Use Case Name** | View Humidity | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User can monitor humidity in statistic via web server and mobile application.  **Goal:**  - Users can view humidity every time they access the system  **Triggers:**  - After user logged to the system and touch on “Sensor” tab.  **Preconditions:**  - User must logged into the system.  - Humidity sensors must be connected to the system.  **Post Conditions:**  - Success: Information must be displayed in real time  - Fail: Display a message error to user, resent command to check status and ask user want to get indexes of humidity sensor again.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Humidity” line | Send command to get new index from humidity sensor  Display a message with new index |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on line of other sensors | Send command to get new index from other sensors  Display a message with new index |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Humidity” line | Fail when send command to get new index from humidity sensor  Display a message error about connect between sensor and controller. |   **Relationships:**  - N/A | | | |

Table 15. “View Humidity” specification.

2.3.3.3. View Soil Moisture

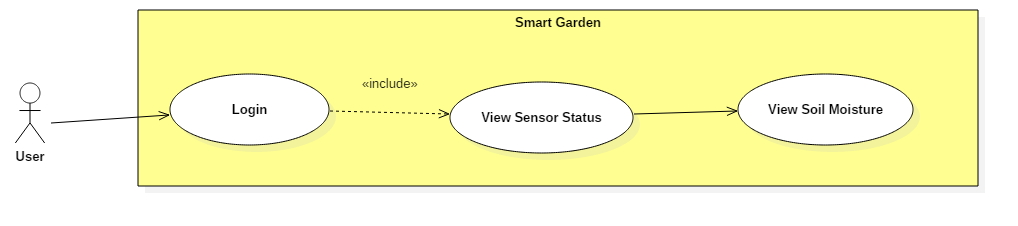


Figure 14.3. “View Soil Moisture” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC08** | | | |
| **Use Case No.** | **UC08** | **Use Case Version** | 2.0 |
| **Use Case Name** | View Soil Moisture | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User can monitor soil moisture via web server and mobile application.  **Goal:**  - Users can view soil moisture every time they access to system.  **Triggers:**  - After user logged to the system and touch on “Sensor” tab.  **Preconditions:**  - User must logged into the system.  - Soil moisture sensors must be connected to the system.  **Post Conditions:**  - Success: Information must be displayed in real time  - Fail: Display a message error to user, resent command to check status and ask user want to get indexes of soil moisture sensor again.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Temperature” line | Send command to get new index from temperature sensor  Display a message with new index |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on line of other sensors | Send command to get new index from other sensors  Display a message with new index |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Soil Moisture” line | Fail when send command to get new index from temperature sensor  Display a message error about connect between sensor and controller. |   **Relationships:**  - N/A | | | |

Table 16. “View Soil Moisture” specification.

2.3.3.4. View Light

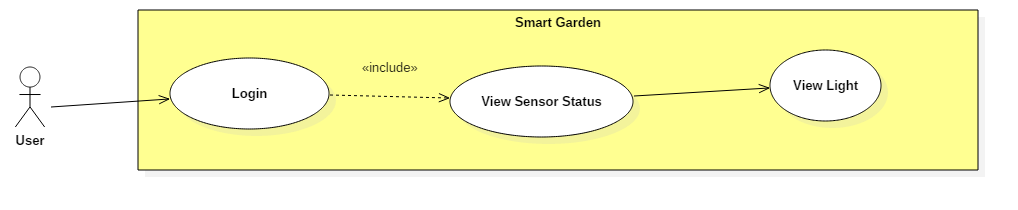


Figure 14.4. “View Light” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC09** | | | |
| **Use Case No.** | **UC09** | **Use Case Version** | 2.0 |
| **Use Case Name** | View Light | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User can monitor light via web server and mobile application.  **Goal:**  - Users can view light every time after they access to the system  **Triggers:**  - After user logged to the system and touch on “Sensor” tab.  **Preconditions:**  - User must logged into the system.  - Light sensor must be connected to the system.  **Post Conditions:**  - Success: Information must be displayed in real time  - Fail: Display a message error to user, resent command to check status and ask user want to get indexes of light sensor again.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Light” line | Send command to get new index from temperature sensor  Display a message with new index |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on line of other sensors | Send command to get new index from other sensors  Display a message with new index |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “Light” line | Fail when send command to get new index from temperature sensor  Display a message error about connect between sensor and controller. |   **Relationships:**  - N/A | | | |

Table 17. “View Light” specification.

2.3.3.5. View pH

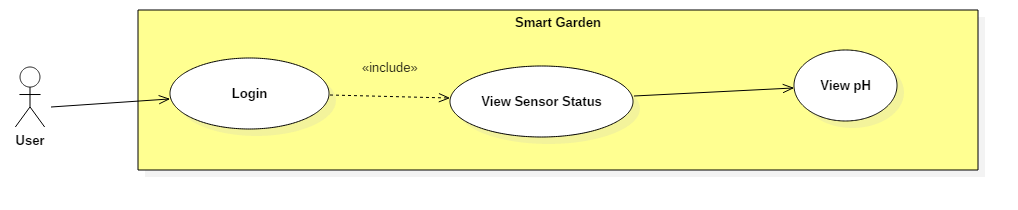


Figure 14.5. “View pH” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC10** | | | |
| **Use Case No.** | **UC10** | **Use Case Version** | 2.0 |
| **Use Case Name** | View pH | | |
| **Author** | Pham Hoang Chinh | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User can get pH via web server and mobile application.  **Goal:**  - Users can view pH every time after they access to the system  **Triggers:**  - After user logged to the system and touch on “pH” tab.  **Preconditions:**  - User must logged into the system.  - pH sensor must be connected to the system.  **Post Conditions:**  - Success: Information must be displayed in real time  - Fail: Display a message error to user, resent command to check status and ask user want to get indexes of light sensor again.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “pH” line | Send command to get new index from temperature sensor  Display a message with new index |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on line of other sensors | Send command to get new index from other sensors  Display a message with new index |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Touch on “Sensor” tab | Navigate to Sensor screen  Display current indexes of sensors in nearly time. | | 2 | Touch on “pH” line | Fail when send command to get new index from temperature sensor  Display a message error about connect between sensor and controller. |   **Relationships:**  - N/A | | | |

Table 18. “View pH” specification

2.3.4. <User> Manager Zone Overview Use Case

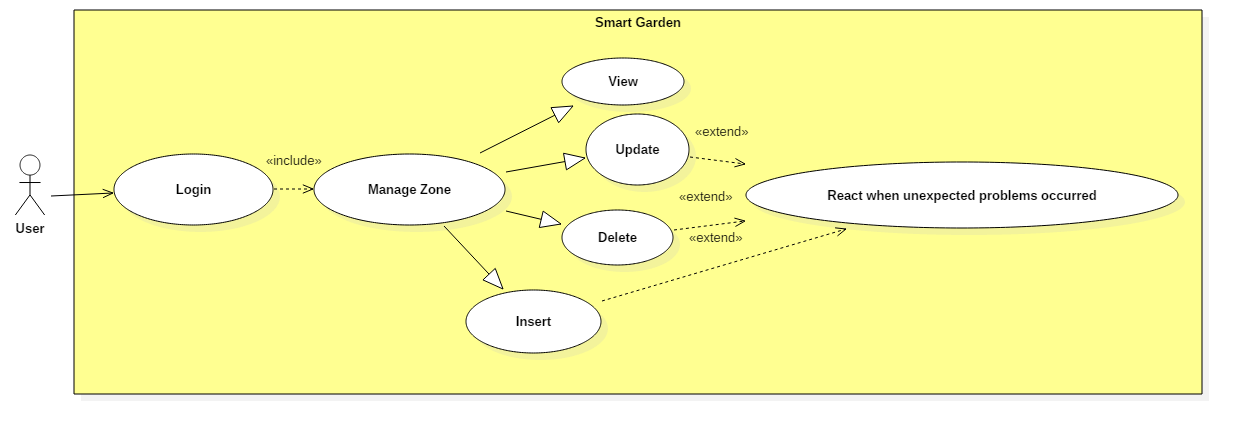


Figure 15. Manager Zone Overview use case

2.3.4.1 <User> Zone View

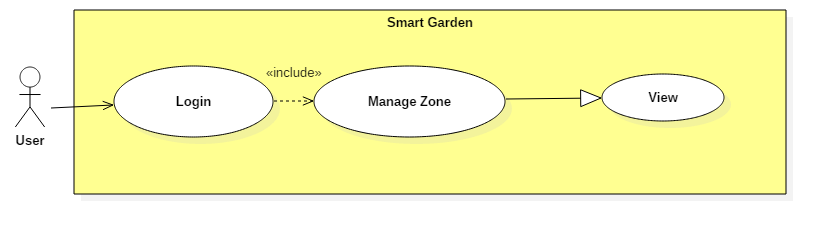


Figure 15.1. “Zone View” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC11** | | | |
| **Use Case No.** | **UC11** | **Use Case Version** | 2.0 |
| **Use Case Name** | Zone View | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User need to divide each plant to each zone because each plant has a different condition living.  **Goal:**  - User can divide each plant to each zone to make it easy to manager.  - User can view information of each zone.  **Triggers:**  - NA  **Preconditions:**  - System must provide information on core database which join with a personal database for each user.  **Post Conditions:**  - Success: User can view list of zone which is on core database and personal database.  - Fail: Display a popup message for errors. Log errors to system in case conflict between personal database and core database.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Zone page | Get information from database.  Show list of zone | | 2 | User choose one zone which they want | Navigate to Information of zone  Show standard information which is load from core database |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Zone page | Get information from database.  Show list of zone | | 2 | User touch “Add” button | Navigate to Add new page |  |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access Zone page | Get information from database.  Show list of zone | | 2 | User touch and hold on one or many zone | Choose zone which user pressed | | 3 | User touch “Delete” button | Delete zone which user pressed  Update to database |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Errors from join database or system  Log errors. | | 2 | Waiting for information from database | Display errors message |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone | | 2 | User choose one zone which they want | Can’t get information of that zone | | 3 | Waiting for information | Display errors message |   **Relationships:**  - N/A | | | |

Table 19. “Zone View” specification.

2.3.4.2 <User> Zone Update

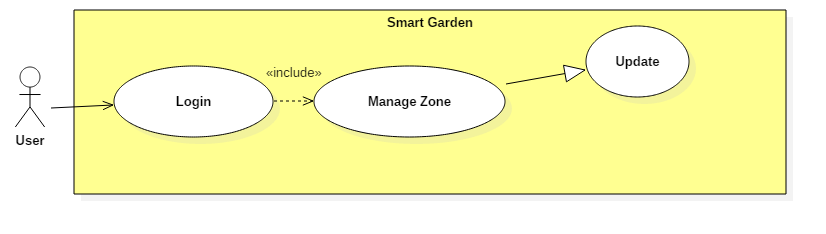


Figure 15.2. “Zone Update” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC12** | | | |
| **Use Case No.** | **UC12** | **Use Case Version** | 2.0 |
| **Use Case Name** | Library Update | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - User want to change information of zone. This use case allow user to change information.  **Goal:**  - System can work based on indexes which user changed. User can change name and device id of each zone.  **Triggers:**  - NA  **Preconditions:**  - The changed of user is added to personal database.  **Post Conditions:**  - Success: User change data successful.  - Fail: Display a popup message for errors. Log errors to system in case conflict.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone | | 2 | Touch on one zone which they want | Navigate to Information of zone  Show standard information which is load from database | | 3 | Touch on each field of indexes and edit information | Validation indexes in fields. | | 4 | Touch “Done” button | Update to database  Display a message “Update Successful” |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone | | 2 | User touch and hold on an object | Turn on Delete button and allow to choose more than one object | | 3 | Touch on Delete button | Delete objects which user chose |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone | | 2 | User choose one zone which they want | Navigate to Information of zone  Show standard information which is load from core database | | 3 | Touch on each field of indexes and edit information.  Input wrong data types of field | Validation data which user input  Display an error message. |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone | | 2 | User choose one zone which they want | Navigate to Information of zone  Show standard information which is load from core database | | 3 | Touch on each field of indexes and edit information | Validation indexes in fields. | | 4 | Touch “Done” button | Update to database fail  Display a message “Update Fail” |   **Relationships:**  - N/A | | | |

Table 20. “Zone Update” specification.

2.3.4.3 <User> Zone Insert

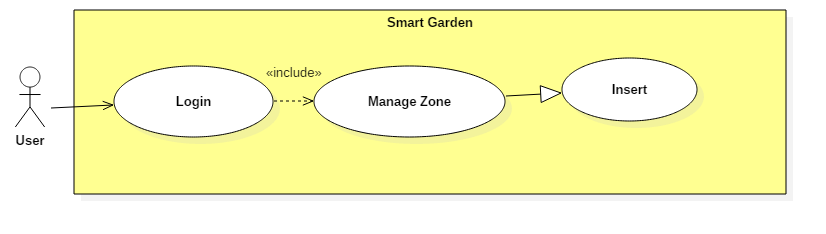


Figure 15.3. “Zone Insert” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC13** | | | |
| **Use Case No.** | **UC13** | **Use Case Version** | 2.0 |
| **Use Case Name** | Zone Insert | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - Allow user can add more zone to their database (each zone can be connected with new device).  **Goal:**  - User can add many zone add they want. Depend on how many devices they have.  **Triggers:**  - NA  **Preconditions:**  - Each zone has a device ID.  **Post Conditions:**  - Success: User can add any zone which they want to personal database which system can work on.  - Fail: Display a popup message for errors. Log errors to system in case conflict.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access zone page | Get information from database.  Show list of zone | | 2 | Touch “Add” button | Navigate to Add new screen with fields which ask to input full data on | | 3 | User fill all fields in screen | Validation data which user input | | 4 | User touch “Add” button | Insert new data to database.  Display message “Add successful” |   **Alternative Scenario:**  - NA  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone page | Get information from database.  Show list of zone | | 2 | Touch “Add” button | Navigate to Add new screen with fields which ask to input full data on | | 3 | Users don’t fill enough fields in screen or input wrong types of data | Validation data which user input  Display message errors to user’s screen |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone page | Get information from database.  Show list of zone | | 2 | Touch “Add” button | Navigate to Add new screen with fields which ask to input full data on | | 3 | Waiting for information | Display errors message | | 4 | User touch “Add” button | Insert to database fail  Log errors  Display message error to user |   **Relationships:**  - N/A | | | |

Table 21. “Zone Insert” specification.

2.3.4.4 <User> Zone Delete

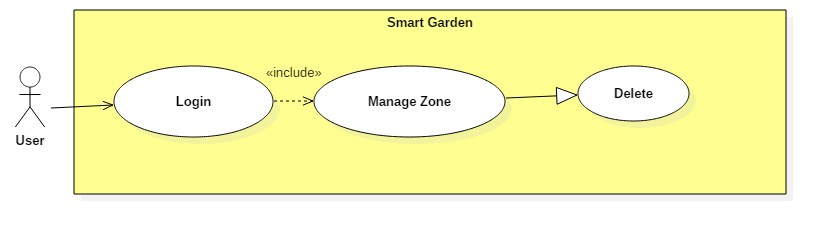


Figure 15.4. “Zone Delete” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC14** | | | |
| **Use Case No.** | **UC14** | **Use Case Version** | 2.0 |
| **Use Case Name** | Zone Delete | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - Allow user to delete zone information in database.  **Goal:**  - User can delete zone which they want.  **Triggers:**  - NA  **Preconditions:**  - Information which will delete by user is exist.  **Post Conditions:**  - Success: User can delete one or many zone which they want.  - Fail: Display a popup message for errors. Log errors to system in case conflict.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone. | | 2 | Touch and hold on one object plant in list | Turn on Delete button and allow to choose more than one object | | 3 | User touch on Delete button | Update to database to delete one or many object which user chose.  Update list of plants in zone Screen |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone. | | 2 | User touch on one object of zone | Navigate to Library View Screen |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone. | | 2 | Touch and hold on one object zone in list | Turn on Delete button and allow to choose more than one object | | 3 | User touch on Delete button | Fail when update to database to delete one or many object which user chose.  Display a message error to user “Delete unsuccessful” |  |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Access zone screen | Get information from database.  Show list of zone. | | 2 | Touch and hold on one object zone in list | Turn on Delete button and allow to choose more than one object | | 3 | User touch on Delete button | Update to database to delete one or many object which user chose.  Fail when update list of plants in Library Screen  Display a message error to user |   **Relationships:**  - N/A | | | |

Table 22. “Zone Delete” specification.

2.3.5. <User> Control Electric Devices Overview Use Case

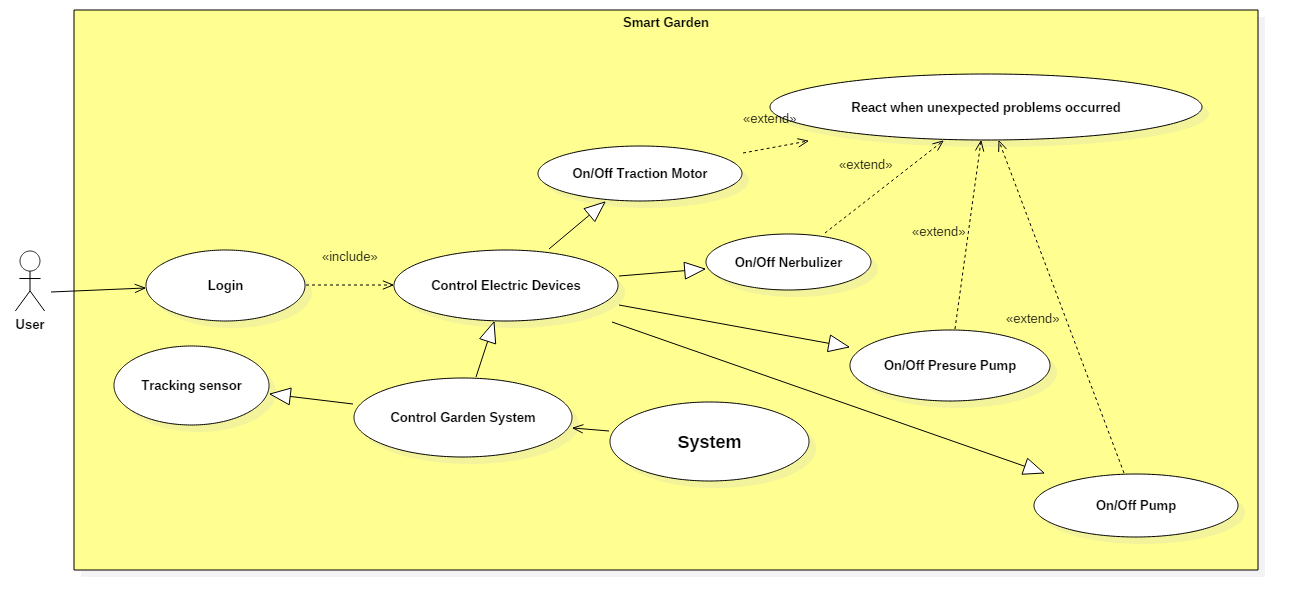


Figure 16. Control Electric Devices Overview use case

2.3.5.1 On off traction motor devices

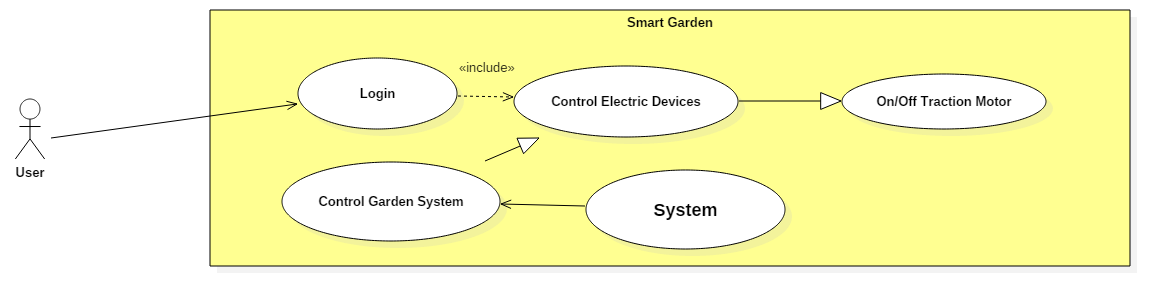


Figure 16.1. “On off traction motor” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC15** | | | |
| **Use Case No.** | **UC15** | **Use Case Version** | 2.0 |
| **Use Case Name** | On off traction motor | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - User  **Summary:**  - This function will be control traction motor (turn on or off) automatically or manual.  **Goal:**  - Devices turn on/off traction motor automatically by using sensors.  - Devices turn on/off traction motor in manual mode.  **Triggers:**  - Traction motor is connected to the system.  - Sensors is connect to the system.  **Preconditions:**  - System can read information from sensors.  **Post Conditions:**  - Success: Traction motor is turn on/off.  - Fail: Can’t connected with traction motor. Warning message will be shown “Can’t connected with traction motor”.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with traction motor in this Zone. | | 2 | Turn on/off traction motor. | Traction motor turn on/off. |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with traction motor in this Zone. | | 2 | Turn on/off traction motor. | Message will be shown “Can’t connect with device”. |   **Relationships:**  - N/A  **Business Rules:**  **-** Device turn on/off automatically based on information taking from sensors. Or user can turn it on/off manual. | | | |

**Table 23. “On off traction motor” specification.**

2.3.5.2 On off nebulizer devices

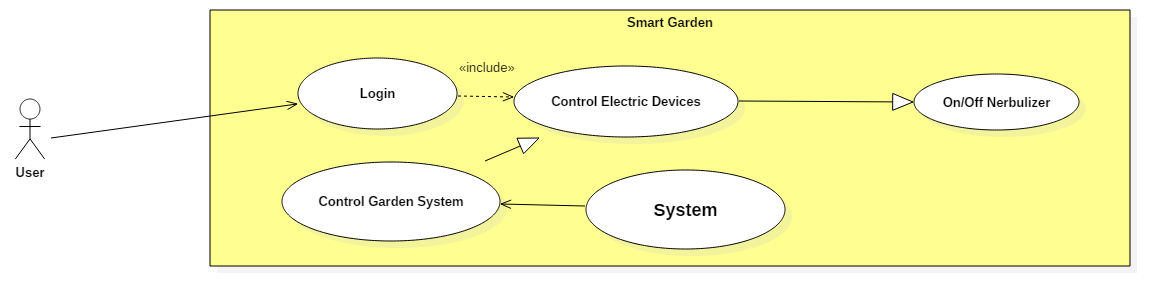


Figure 16.2 “On off nebulizer” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC16** | | | |
| **Use Case No.** | **UC16** | **Use Case Version** | 2.0 |
| **Use Case Name** | On off nebulizer | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - User  **Summary:**  - This function will be control nebulizer (turn on or off) automatically or manual.  **Goal:**  - Devices turn on/off nebulizer automatically by using sensors.  - Devices turn on/off nebulizer in manual mode.  **Triggers:**  - Nebulizer is connected to the system.  - Sensors is connect to the system.  **Preconditions:**  - System can read information from sensors.  **Post Conditions:**  - Success: Nebulizer is turn on/off.  - Fail: Can’t connected with nebulizer. Warning message will be shown “Can’t connected with nebulizer”.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with nebulizer in this Zone. | | 2 | Turn on/off nebulizer. | Nebulizer turn on/off. |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with nebulizer in this Zone. | | 2 | Turn on/off nebulizer. | Message will be shown “Can’t connect with nebulizer”. |   **Relationships:**  - N/A  **Business Rules:**  **-** Device turn on/off automatically based on information taking from sensors. Or user can turn it on/off manual. | | | |

**Table 24. “On off nebulizer” specification.**

2.3.5.3 On off pressure pump devices

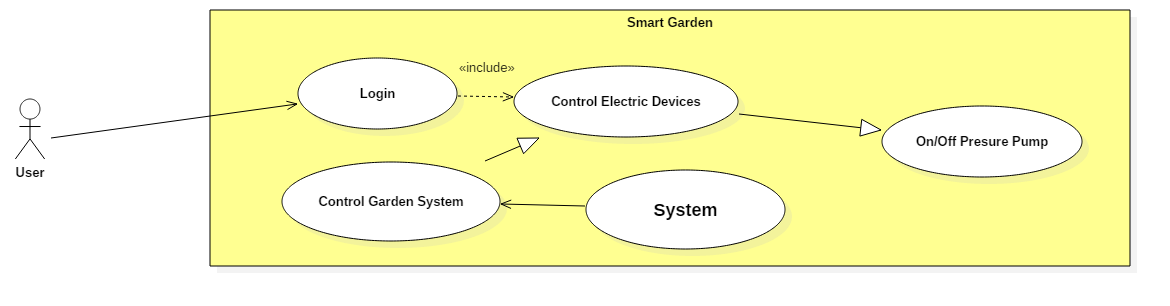


Figure 16.3 “On off pressure pump” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC17** | | | |
| **Use Case No.** | **UC17** | **Use Case Version** | 2.0 |
| **Use Case Name** | On off pressure pump | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - User  **Summary:**  - This function will be control pressure pump (turn on or off) automatically or manual.  **Goal:**  - Devices turn on/off pressure pump automatically by using sensors.  - Devices turn on/off pressure pump in manual mode.  **Triggers:**  - Pressure pump is connected to the system.  - Sensors is connect to the system.  **Preconditions:**  - System can read information from sensors.  **Post Conditions:**  - Success: Pressure pump is turn on/off.  - Fail: Can’t connected with pressure pump. Warning message will be shown “Can’t connected with pressure pump”.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with pressure pump in this Zone. | | 2 | Turn on/off pressure pump. | Pressure pump turn on/off. |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with pressure pump in this Zone. | | 2 | Turn on/off pressure pump. | Message will be shown “Can’t connect with pressure pump”. |   **Relationships:**  - N/A  **Business Rules:**  **-** Device turn on/off automatically based on information taking from sensors. Or user can turn it on/off manual. | | | |

**Table 25. “On off pressure pump” specification.**

2.3.5.3 On off pump devices

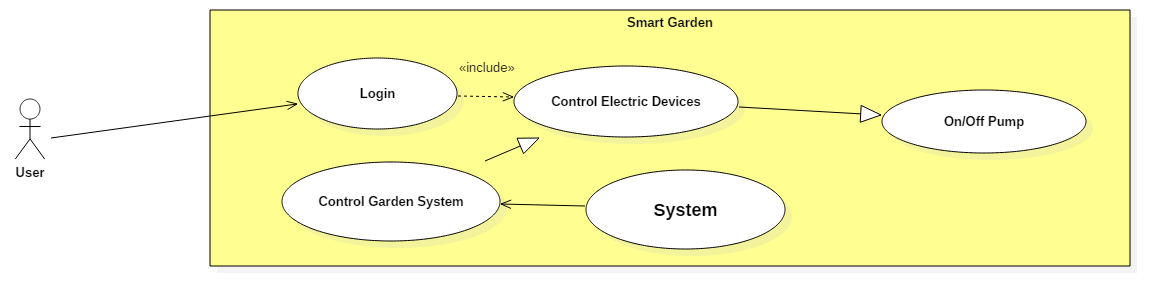


Figure 16.4 “On off pump” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC18** | | | |
| **Use Case No.** | **UC18** | **Use Case Version** | 2.0 |
| **Use Case Name** | On off pump | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - User  **Summary:**  - This function will be control pump (turn on or off) automatically or manual.  **Goal:**  - Devices turn on/off pump automatically by using sensors.  - Devices turn on/off pump in manual mode.  **Triggers:**  - Pump is connected to the system.  - Sensors is connect to the system.  **Preconditions:**  - System can read information from sensors.  **Post Conditions:**  - Success: Pump is turn on/off.  - Fail: Can’t connected with pump. Warning message will be shown “Can’t connected with pump”.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with pump in this Zone. | | 2 | Turn on/off pump. | Pump turn on/off. |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose Zone. | Connect with pump in this Zone. | | 2 | Turn on/off pump. | Message will be shown “Can’t connect with pump”. |   **Relationships:**  - N/A  **Business Rules:**  **-** Device turn on/off automatically based on information taking from sensors. Or user can turn it on/off manual. | | | |

**Table 26. “On off pump” specification.**

2.3.6. React when unexpected problems occurred

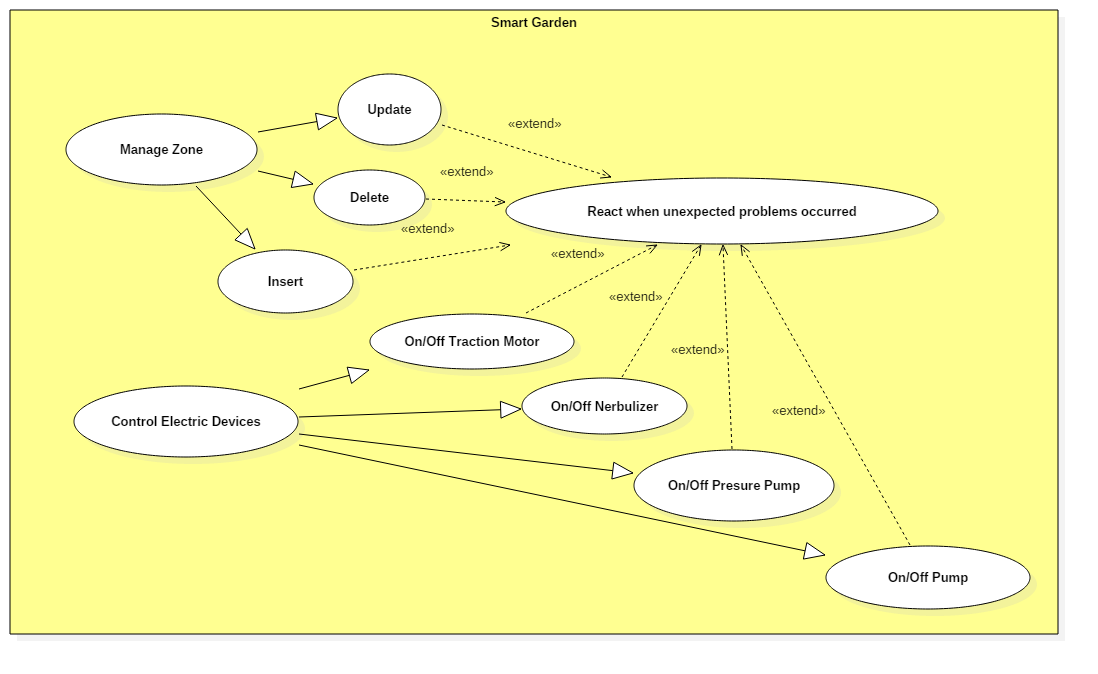


Figure 17. “React when unexpected problems occurred” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC19** | | | |
| **Use Case No.** | **UC19** | **Use Case Version** | 2.0 |
| **Use Case Name** | React when unexpected problems occurred | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - N/A  **Summary:**  - React when unexpected problems occurred.  **Goal:**  - Have plan B when unexpected problems occurred.  **Triggers:**  - N/A  **Preconditions:**  - N/A.  **Post Conditions:**  - Success: Display a popup message for this problem.  - Fail: N/A.  **Main Success Scenario:**  - N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**  - N/A  **Relationships:**  - NA | | | |

Table 27. “React when unexpected problems occurred” specification.

**2.3.7. Tracking sensor**

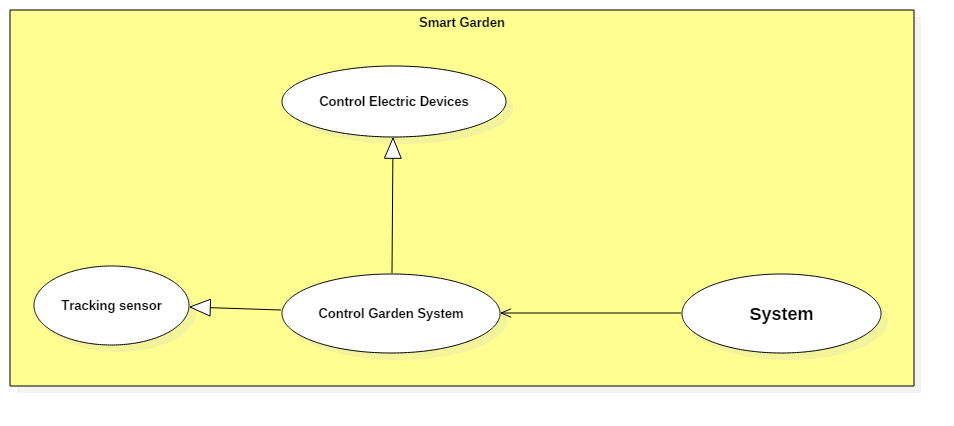


Figure 18. “Tracking sensor” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC20** | | | |
| **Use Case No.** | **UC20** | **Use Case Version** | 2.0 |
| **Use Case Name** | Tracking sensor | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - N/A  **Summary:**  - Tracking sensor.  **Goal:**  - Get information from sensor.  **Triggers:**  - N/A  **Preconditions:**  - N/A.  **Post Conditions:**  - Success: Get information from sensor successfully.  - Fail: Can’t connect to sensor.  **Main Success Scenario:**  - N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**  - N/A  **Relationships:**  - NA | | | |

**Table 27. “Tracking sensor” specification.**

**2.3.2. User logout**

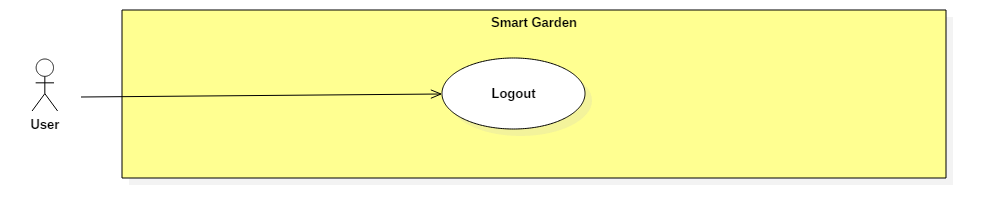


Figure 19. “User logout” use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC21** | | | |
| **Use Case No.** | **UC21** | **Use Case Version** | 2.0 |
| **Use Case Name** | User logout | | |
| **Author** | Huynh Huu Nghi | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - User  **Summary:**  - Logout from system  **Goal:**  - User can logout.  **Triggers:**  - N/A.  **Preconditions:**  - User login.  **Post Conditions:**  - Success: Logout successfully. Message will be shown “Logout successfully”.  - Fail: Can’t logout. Warning message will be shown “Can’t logout. Please try again”.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Logout | Logout from system  Message will be shown “Logout successfully”. |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Logout | Logout from system  Warning message will be shown “Can’t logout. Please try again”. |   **Relationship:**  **-** N/A  **Business Rules:**  **-** Logout from system. | | | |

**Table 28. “Logout” specification.**

**3. Software System Attribute**

**3.1 Usability**

The system is expected to apply for garden of users who has a garden in home but they don’t have enough time to take care or tracking usually. There are many plant’ indexes which gardener are tracking to get higher productivity in agriculture but the system has some basic sensors which can help users keep plants live and limit harmful weather with their garden.

System provide friendly GUI for users on their mobile which is closely with people in nowadays.

**3.2 Reliability**

System is expected to run continuously for years without errors (or in some cases recover by themselves if an error occurs).

Ensure that when system starts a "mission", it has a high probability of completing that mission without experiencing a failure.

System can safely be shut down for repair, or another way to repair without stopping system.

**3.3 Availability**

The server shall be working 24/7. When the system goes in under-maintenance, the page or application will display message "System is maintaining at the moment. Please check again later". Current system will work directly with index from sensors.

**3.4 Security**

Guarantee the data and application protection from being stolen and modified by encoding and decoding data.

**3.5 Maintainability**

All code shall be fully documented. All program files shall include comments concerning authorship and date of last change.

The code shall be modular to permit future modifications.

**3.6 Portability**

The system shall be designed to control and run on Web platform, Android platform. Make sure that control device must be portable device as smartphone, tablet…

Provided hardware can be plug-and-play

**3.7 Performance**

This is the system's performance characteristics:

Capacity: 1 end user a time.

Response time for a transaction:

Average: second

Maximum: seconds