

SwitchDoc Labs

The Smart Garden System Configuration and Operations Manual

July 2020
Version 1.2



Table of Contents

Why you should build this project	7
What is The Smart Garden System?	8
Ready to begin?.....	9
Using this Manual:.....	9
Assembling the Smart Garden System	11
Initial Testing of your Smart Garden System	15
Configuration of your Smart Garden System	19
SGS Valve Configure Window	21
Valve Configuration Menu	22
SGS Valve Report	23
SGS Name Change	24
SGS DM (Debug / MySQL / Manual Watering / Metric / English)	25
SGS Mail and Text Notifications.....	27
SGS Pixel / Solar Configuration / Station Altitude	28
SGS Weather Configuration	29
SGS Blynk and ThunderBoard Configuration	30
SGS Physical Pin Configurations.....	31
SGS Camera / Rest / MQTT Configuration	32
Dash Application Screens	33
Dash Application Screens	34
SGS Status Tab	35
Valve Graphs Tab	36
Weather Tab	37
Hydroponics Tab	38
Herb Garden Tab	38
Next Events Tab	39
P/V Programming Tab.....	40
Logs Tab	41
Documentation Tab	42
Disclaimer	43

Why you should build this project

Imagine buying a fully functional computer for \$35 and then tracking the weather at your home, and even watering your houseplants from your phone while away on vacation? Imagine learning a little about electronics without messing with soldering, resistors, and transistors? Ever been curious about “coding” but have no interest in taking a class? This great little project will take you less than six hours at your kitchen tables and when you are done, you will become familiar with:

- How a simple computer works (Raspberry Pi)
- What valves, pumps and solenoids do
- The purpose of a relay
- Python coding language
- How to connect your devices to the Internet and your phone

This is the ultimate IoT you have been reading about – the Internet of Things. If you are a tinkerer and love to invent stuff, this is for you. Engage your kids and learn something new together!

What is The Smart Garden System?

Imagine building your own remote monitoring and management system for your indoor or outdoor garden? How about sharing your garden and the weather worldwide? You can do all this and learn a little about electronic circuitry **without touching a soldering iron**. You can measure soil moisture and use that as water your plants or garden just the right about of water.

Highly expandable! You can have your Raspberry Pi Base Unit inside your house and have multiple wireless control units in your outdoor garden, greenhouse or in the upstairs bedroom. Up to 250 wireless control units can be connected to one Raspberry Pi base unit. This allows you to control your truly MASSIVE garden. Or your small one. Either way!

This is a perfect highly expandable project kit for kids with some help from the adults and for adults trying to learn some new things. We have done this before with our successful [OurWeather Kickstarter](#) so we know what we are talking about. People all over the world have built the [OurWeather](#) weather station with great success. Grove connectors to wire everything up so you don't need to solder! And you cannot reverse them and blow things up. [Here is our tutorial on the Grove system.](#)

The Smart Garden System Features

- Control Valves, Pumps and Lights
- Measure your Soil Moisture in your yard or in your home
- Show your results on the Internet and your Phone

Weather expansion kit:

- Monitor weather in your yard, including wind speed and direction, rain, UV and sun intensity, temperature, and humidity.
- Show your results on the Internet and your Phone

Easy to build. Easy to learn about the IOT (Internet Of Things) and the Raspberry Pi.

Ready to begin?

Find a clean, dry indoor workspace where you can spread out the parts without them being disturbed.

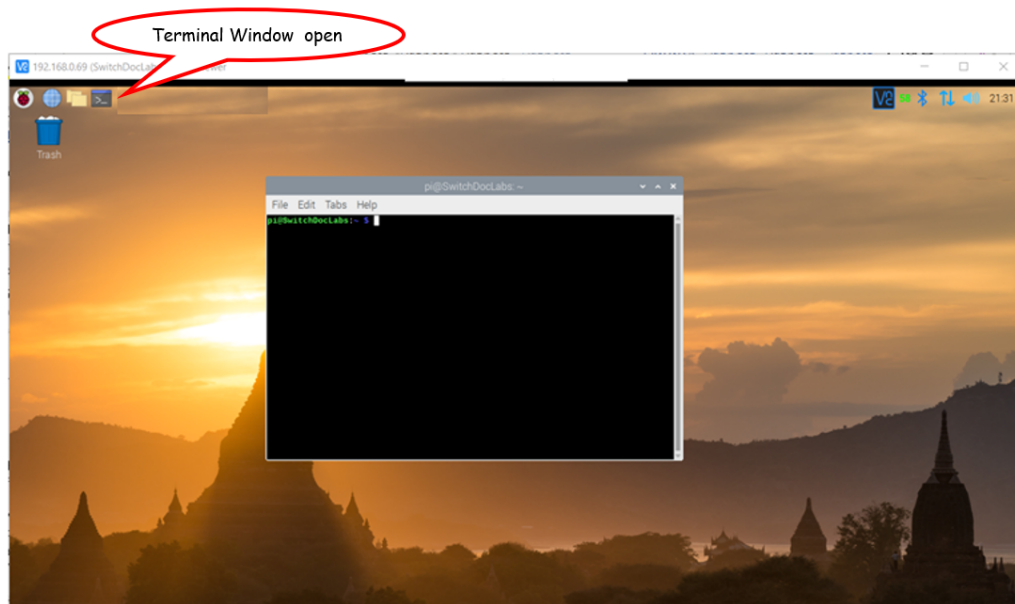
- Static is your enemy. You can control static by touching a grounded object such as a radiator with one hand when touching an electronic component with the other. If you are a geek you can buy an anti-static mat for as little as \$25.
- Always keep water away from the electronics and power supply! Keep that coffee cup at a distance!
- The Smart Garden System electronics are designed for indoor use only and should be placed in a dry environment. Of course, certain parts are designed for outdoor use, such for monitoring soil moisture and watering your yard.
- This is not a toy! Keep it out of reach of young children and pets.
- SwitchDoc Labs assumes no liabilities in the use of this kit, beyond the refund of the purchase price.

Using this Manual:

You do not need to learn how to code. You will be given specific commands to type into the Terminal Window. These commands are written in a popular programming language called Python. Therefore, you are in fact writing code.

Along the way you will pick up some tidbits, for example, the command the entry “sudo” tells the software you are a super-user, allowing you to do stuff not available to normal users. Sudo is often followed by the word “python” to tell the software to act on your command in Python language. Finally, pay close attention to your type case: “Python” and “python” are not the same command.

When we talk about the Terminal Window, we are referring to this on the Raspberry Pi:



When you need to type something in the Terminal Window, we will show what you should type in this font and color:

```
cd SDL_Pi_SmartGardenSystem2
```

What is displayed in the terminal window looks like this:

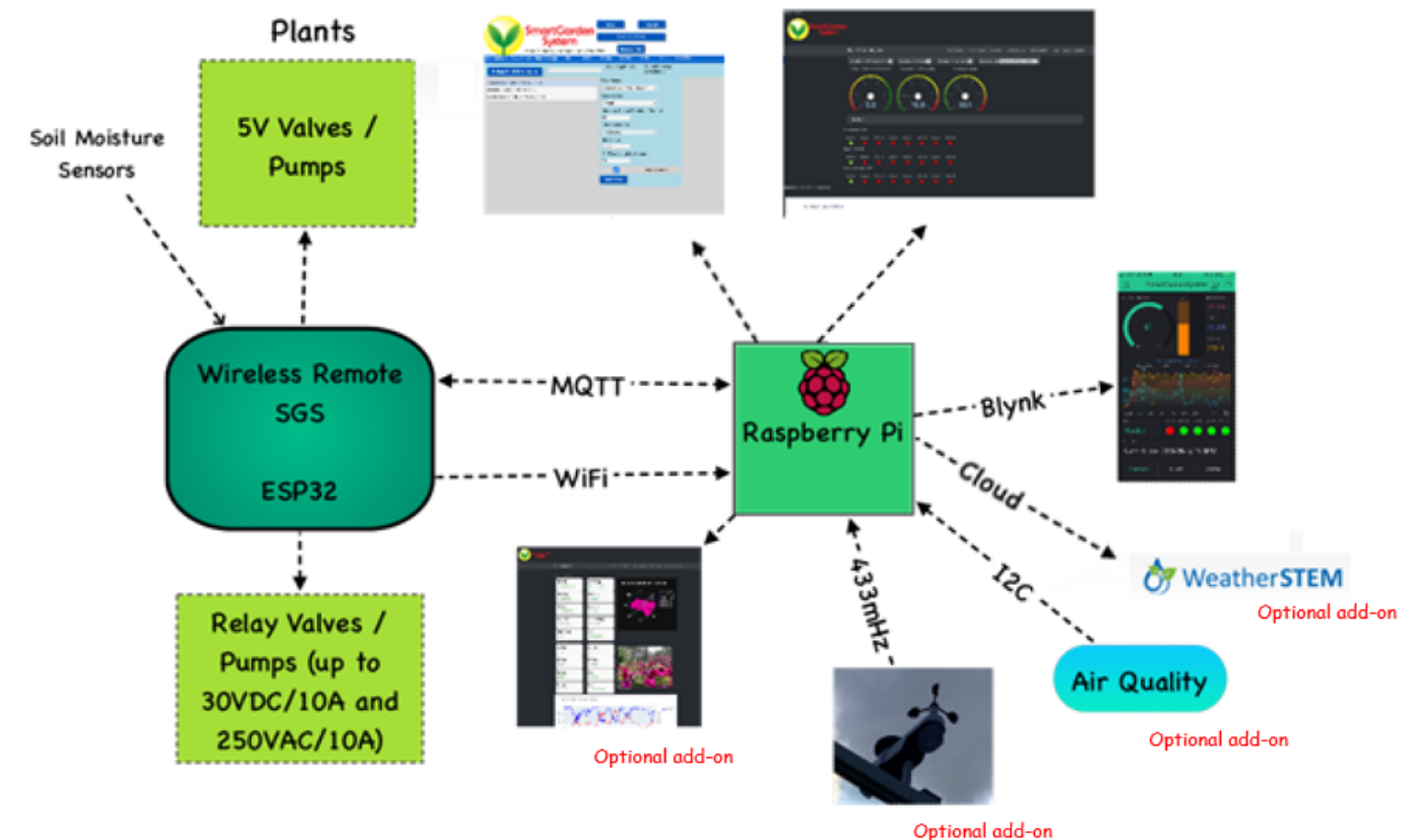
```
SGSConfigure.py:24: DeprecationWarning: AppURLopener style of invoking requests is
deprecated. Use newer urlopen functions/methods
  myURLopener = AppURLopener()
```

Assembling the Smart Garden System

Begin your project by assembling the Smart Garden System components. You will need to complete steps outlined in the [Smart Garden System Assembly and Test Manual](#).

Step 1: With your Smart Garden System up and running, you now want to make sure your Wireless Extender unit(s) are plugged in and connected to your Wi-Fi network.

Smart Garden System Block Diagram



SwitchDoc Labs

Step 2: Determine your Raspberry Pi IP address on your network. Follow the procedures here: <https://learn.pimoroni.com/tutorial/raspberry-pi/finding-your-raspberry-pi>

In a terminal window, change directories down to:

```
cd SDL_Pi_SmartGardenSystem2
```

Run SGSConfigure.py

```
sudo python3 SGSConfigure.py
```

You will see something like this:

```
pi@SwitchDocLabs:~/SDL_Pi_SmartGardenSystem2 $ sudo python3 SGSConfigure.py
SGSConfigure.py:24: DeprecationWarning: AppURLOpener style of invoking requests is
deprecated. Use newer urlopen functions/methods
  myURLOpener = AppURLOpener()
remi.server      INFO      Started httpserver http://0.0.0.0:8001/
remi.request     INFO      built UI (path=/)
SGS.JSON File does not exist
SGSConfiguration.JSON File does not exist
```

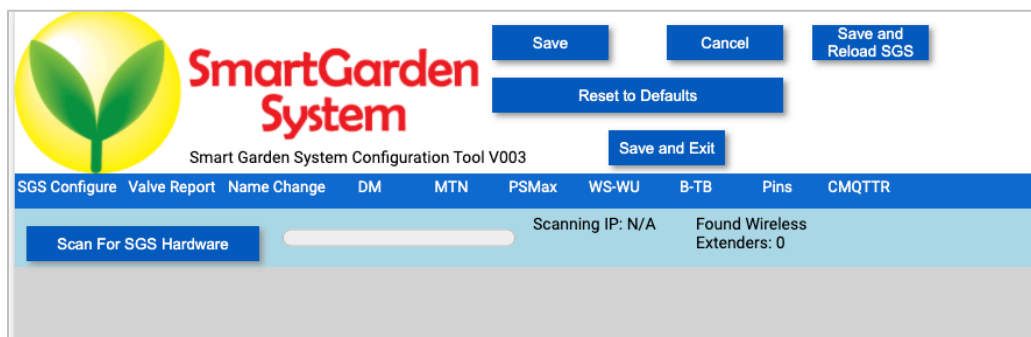
Now open a browser window (either on the Raspberry Pi or on another computer on your local WiFi network) and enter this URL:

On your Raspberry Pi: <http://127.0.0.1:8001/>

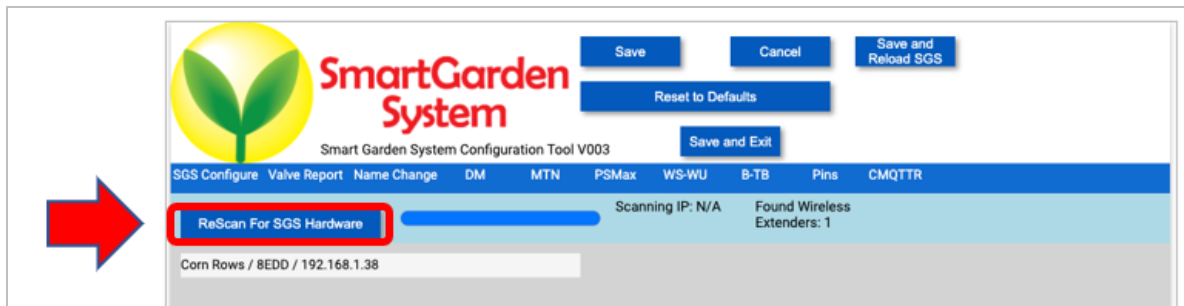
If you do this on another computer type: <http://xxx.xxx.xxx.xxx:8001/>

NOTE: The “xxx.xxx.xxx.xxx” is the IP address of **your** Raspberry Pi that you wrote down above.

You will see this screen on your browser:



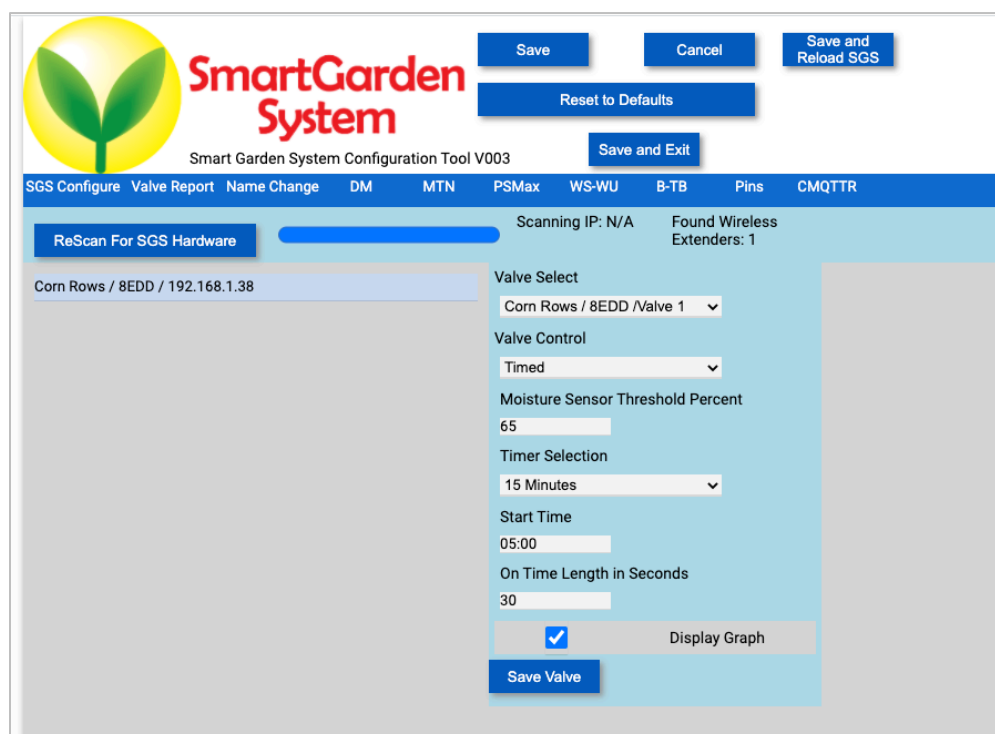
Step 3: Click on the “Scan For SGS Hardware Button”. The first thing we do is have the SGSConfigure software scan for all your wireless units. Your base unit comes with one system. This takes a while (15 or 20 minutes). Click on the “Scan For SGS Hardware Button”.



SGSConfigure found one SGS Wireless Unit: “Corn Rows/8EDD/192.168.1.38”. Note that we had named this unit “Corn Rows” earlier. Yours will probably show up blank.

Step 4: Click on the Wireless Unit on the screen to open up the Valve configuration menus. “Corn Rows/8EDD/192.168.1.38” in our example.

Select **valve 1** – using the Valve Select menus (make sure you still have the USB Light Stick plugged into Valve 1) to turn on for 30 seconds every 15 minutes and click Show Graph. Note that you can only select the Timer Selection and Start Time if you have selected “Timed” in the Valve Control dropdown menu. You should see the screen shown here: Finally click the “Save Valve” button. **If you do not do this, the valve changes are NOT saved.**



Step 5: Click on the “DM” Tab on your menu.

Click on enable SW Debugging (you can turn this off later).

Click on enable MySQL logging. The default MySQL password on the SDL SD Card is “password”.

Click on Save and Exit which saves your JSON files for SGS2 and quits the SGSConfigure program.

Note: On some systems, you may have to hit either “ctrl-c” or the “ctrl-|” to get the server to quit in the terminal window.

Look at the JSON files to see what you have done! SGS.JSON is the general configuration file, while SGSConfiguration.JSON is the valve/pump/timers configuration file for all the wireless extender units.

```
pi@SwitchDocLabs:~/SDL_Pi_SmartGardenSystem2 $ more SGS.JSON
```

```
{ "key": "value", "ProgramName": "SmartGardenSystem2", "ConfigVersion": "001", "S
WDEBUG": true, "enable_MySQL_Logging": true, "English_Metric": false, "MySQL_Pas
sword": "password", "mailUser": "yourusername", "mailPassword": "yourmailpasswor
d", "notifyAddress": "you@example.com", "fromAddress": "yourfromaddress@example.
com", "enableText": false, "textnotifyAddress": "yournumber@yourprovider", "enab
lePixel": false, "pixelPin": "21", "SolarMAX_Present": false, "SolarMAX_Type": "
LEAD", "BMP280_Altitude_Meters": "626.0", "Sunlight_Gain": "High", "weather": fa
lse, "USEWEATHERSTEM": false, "INTERVAL_CAM_PICS_SECONDS": "60", "STATIONKEY":
"", "WeatherUnderground_Present": false, "WeatherUnderground_StationID": "KWXXXX
X", "WeatherUnderground_StationKey": "YYYYYY", "USEBLYNK": false, "BLYNK_AUTH":
"", "AS3935_Lightning_Config": "[2,1,3,0,3,3]", "REST_Enable": false, "Camera_Ni
ght_Enable": false, "MQTT_Enable": false, "MQTT_Server_URL": "", "MQTT_Port_Numb
er": "5900", "MQTT_Send_Seconds": "500", "manual_water": true, "Tank_Pump_Level"
: "15.0", "WirelessDeviceJSON": [{"return_value": 0, "id": "8EDD", "name": "Corn
Rows", "ipaddress": "192.168.1.38", "hardware": "esp32", "return_string": "8EDD
,1,1,1,1,024", "connected": true}], "UltrasonicLevel": "4"}
```

```
pi@SwitchDocLabs:~/SDL_Pi_SmartGardenSystem2 $ more SGSConfiguration.JSON
```

```
{ "SGSConfigVersion": "001", "Valves": [{"id": "8EDD", "ValveNumber": 1, "Control
": "Timed", "MSThresholdPercent": "65", "TimerSelect": "15 Minutes", "StartTime
": "05:00", "OnTimeInSeconds": "30", "ShowGraph": true}, {"id": "8EDD", "ValveNum
ber": 2, "Control": "Off", "MSThresholdPercent": "65", "TimerSelect": "Daily", "
StartTime": "05:00", "OnTimeInSeconds": "10", "ShowGraph": false}, {"id": "8EDD"
, "ValveNumber": 3, "Control": "Off", "MSThresholdPercent": "65", "TimerSelect":
"Daily", "StartTime": "05:00", "OnTimeInSeconds": "10", "ShowGraph": false}, {"
id": "8EDD", "ValveNumber": 4, "Control": "Off", "MSThresholdPercent": "65", "Ti
merSelect": "Daily", "StartTime": "05:00", "OnTimeInSeconds": "10", "ShowGraph":
false}, {"id": "8EDD", "ValveNumber": 5, "Control": "Off", "MSThresholdPercent"
: "65", "TimerSelect": "Daily", "StartTime": "05:00", "OnTimeInSeconds": "10", "
ShowGraph": false}, {"id": "8EDD", "ValveNumber": 6, "Control": "Off", "MSThresh
oldPercent": "65", "TimerSelect": "Daily", "StartTime": "05:00", "OnTimeInSecond
s": "10", "ShowGraph": false}, {"id": "8EDD", "ValveNumber": 7, "Control": "Off"
, "MSThresholdPercent": "65", "TimerSelect": "Daily", "StartTime": "05:00", "OnT
imeInSeconds": "10", "ShowGraph": false}, {"id": "8EDD", "ValveNumber": 8, "Cont
rol": "Off", "MSThresholdPercent": "65", "TimerSelect": "Daily", "StartTime": "0
5:00", "OnTimeInSeconds": "10", "ShowGraph": false}]}
```

Initial Testing of your Smart Garden System

Step 6: Start your SGS2 system in a terminal window by typing:

```
sudo python3 SGS2.py
```

You will see something like this as it scrolls across your screen. By the way, if you turn Software debugging ON (table "DM" - Debug configuration on) in SGSConfigure you will see much more on your terminal window.

```
pi@SwitchDocLabs:~/SDL_Pi_SmartGardenSystem2 $ sudo python3 SGS2.py
b''
b''
#####
SGS2 Version 014 - SwitchDoc Labs
#####

Program Started at:2020-07-14 11:00:10

SGS.JSON File exists
SGSConfiguration.JSON File exists
-----
Local Devices
-----
OLED:                Not Present
BMP280:              Present
DustSensor:          Not Present
-----
Checking Wireless SGS Devices
-----
Corn Rows - 8EDD:    Present
subscribing to  SGS/8EDD

-----
Plant / Sensor Counts
-----
Wireless Unit Count: 1
Sensor Count:  4
Valve Count:  8
-----
Other Smart Garden System Expansions
-----
Weather:                Not Present
GardenCam:              Present
SunAirPlus:             Not Present
SolarMAX:               Not Present
Lightning Mode:         Not Present
MySQL Logging Mode:     Present
UseBlynk:               Not Present
-----
-----
Scheduled Jobs
-----
Jobstore default:
  blinkLED (trigger: interval[0:00:05], next run at: 2020-07-14 11:00:19 PDT)
  checkForButtons (trigger: interval[0:00:10], next run at: 2020-07-14 11:00:24 PDT)
  statusLEDs (trigger: interval[0:00:15], next run at: 2020-07-14 11:00:29 PDT)
  checkForAlarms (trigger: interval[0:00:15], next run at: 2020-07-14 11:00:29 PDT)
  manualCheck (trigger: interval[0:00:15], next run at: 2020-07-14 11:00:33 PDT)
  valveCheck (trigger: interval[0:01:00], next run at: 2020-07-14 11:01:18 PDT)
  tick (trigger: interval[0:05:00], next run at: 2020-07-14 11:05:14 PDT)
  readWiredSensors (trigger: interval[0:08:20], next run at: 2020-07-14 11:08:34
PDT)
```

```
updateDeviceStatus (trigger: interval[0:12:00], next run at: 2020-07-14 11:12:14
PDT)
```

```
-----
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "1", "timestamp":
"07/14/2020 18:02:18", "valvestate": "V10000000"}'
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "1", "timestamp":
"07/14/2020 18:02:48", "valvestate": "V00000000"}'
```

When 15 minutes have passed (after the USB Light Stick has turned on and off) minutes,
hit "ctrl-c" to quit.

With software debugging ON, you would have seen this:

```
pi@SwitchDocLabs:~/SDL_Pi_SmartGardenSystem2 $ sudo python3 SGS2.py
b''
b''
```

```
#####
SGS2 Version 014 - SwitchDoc Labs
#####
```

Program Started at:2020-07-14 10:24:57

SGS.JSON File exists
SGSConfiguration.JSON File exists

Local Devices

OLED: Not Present
BMP280: Present
DustSensor: Not Present

Checking Wireless SGS Devices

myURL= http://192.168.1.38/setValves?params=admin,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
return= {'return_value': 0, 'id': '8EDD', 'name': 'Corn Rows', 'ipaddress':
'192.168.1.38', 'hardware': 'esp32', 'return_string': '', 'connected': True}
Corn Rows - 8EDD: Present
MQTT: Sending CONNECT (u0, p0, wr0, wq0, wf0, c1, k60) client_id=b'SGS2'
MQTT: Received CONNACK (0, 0)
subscribing to SGS/8EDD
MQTT: Sending SUBSCRIBE (d0, m1) [(b'SGS/8EDD', 0)]
MQTT: Received SUBACK

Plant / Sensor Counts

Wireless Unit Count: 1
Sensor Count: 4
Valve Count: 8

Other Smart Garden System Expansions

Weather: Not Present
GardenCam: Present
SunAirPlus: Not Present
SolarMAX: Not Present
Lightning Mode: Not Present
MySQL Logging Mode: Present
UseBlynk: Not Present

myURL= http://192.168.1.38/enableMoistureSensors?params=admin,1,1,1,1
myURL= http://192.168.1.38/readMoistureSensors?params=admin

MoistureSensorStates
[{'id': '8EDD', 'sensorType': 'C1', 'sensorNumber': '1', 'sensorValue': '77.85',
'timestamp': '2020-07-14 10:25:05'}, {'id': '8EDD', 'sensorType': 'C1',
'sensorNumber': '2', 'sensorValue': '100.00', 'timestamp': '2020-07-14 10:25:05'},

```

{'id': '8EDD', 'sensorType': 'C1', 'sensorNumber': '3', 'sensorValue': '100.00',
'timestamp': '2020-07-14 10:25:05'}, {'id': '8EDD', 'sensorType': 'C1',
'sensorNumber': '4', 'sensorValue': '100.00', 'timestamp': '2020-07-14 10:25:05'}}
-----
-----
Scheduled Jobs
-----
Jobstore default:
    blinkLED (trigger: interval[0:00:05], next run at: 2020-07-14 10:25:06 PDT)
    checkForButtons (trigger: interval[0:00:10], next run at: 2020-07-14 10:25:11 PDT)
    statusLEDs (trigger: interval[0:00:15], next run at: 2020-07-14 10:25:16 PDT)
    checkForAlarms (trigger: interval[0:00:15], next run at: 2020-07-14 10:25:16 PDT)
    manualCheck (trigger: interval[0:00:15], next run at: 2020-07-14 10:25:20 PDT)
    valveCheck (trigger: interval[0:01:00], next run at: 2020-07-14 10:26:05 PDT)
    tick (trigger: interval[0:05:00], next run at: 2020-07-14 10:30:01 PDT)
    readWiredSensors (trigger: interval[0:08:20], next run at: 2020-07-14 10:33:21
PDT)
    updateDeviceStatus (trigger: interval[0:12:00], next run at: 2020-07-14 10:37:01
PDT)
-----
MQTT: Received PUBLISH (d0, q0, r0, m0), 'SGS/8EDD', ... (175 bytes)
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "4", "timestamp":
"07/14/2020 17:25:26", "enableSensors": "1,1,1,1,", "sensorValues":
"77.85,100.00,100.00,100.00,", "sensorType": "C1,C1,C1,C1"}'
Sensor Message Recieved
-----
Processing MQTT Sensor Message
-----
MoistureSensorStates
[{'id': '8EDD', 'sensorType': 'C1', 'sensorNumber': '1', 'sensorValue': '77.85',
'timestamp': '2020-07-14 10:25:27'}, {'id': '8EDD', 'sensorType': 'C1',
'sensorNumber': '2', 'sensorValue': '100.00', 'timestamp': '2020-07-14 10:25:27'},
{'id': '8EDD', 'sensorType': 'C1', 'sensorNumber': '3', 'sensorValue': '100.00',
'timestamp': '2020-07-14 10:25:27'}, {'id': '8EDD', 'sensorType': 'C1',
'sensorNumber': '4', 'sensorValue': '100.00', 'timestamp': '2020-07-14 10:25:27'}]
-----
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
nextMoistureValveSensorCheck = 2020-07-14 10:15:00
nextMoistureValveSensorCheck = 2020-07-14 10:30:00
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PUBLISH (d0, q0, r0, m2), 'b'SGS/8EDD/Valves'', ... (113 bytes)
MQTT: Received PUBLISH (d0, q0, r0, m0), 'SGS/8EDD', ... (97 bytes)
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "1", "timestamp":
"07/14/2020 17:27:05", "valvestate": "V10000000"}'
Valve Change Received
Timer Fired! Next Fire= 2020-07-14 10:30:00
MQTT: Received PUBLISH (d0, q0, r0, m0), 'SGS/8EDD', ... (97 bytes)
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "1", "timestamp":
"07/14/2020 17:27:35", "valvestate": "V00000000"}'
Valve Change Received
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
The time is: 2020-07-14 10:30:01.355894
>>>>>>Valve Check<<<<<<
MQTT: Sending PUBLISH (d0, q0, r0, m3), 'b'SGS/8EDD/Valves'', ... (113 bytes)
MQTT: Received PUBLISH (d0, q0, r0, m0), 'SGS/8EDD', ... (97 bytes)
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "1", "timestamp":
"07/14/2020 17:30:05", "valvestate": "V10000000"}'
Valve Change Received
Timer Fired! Next Fire= 2020-07-14 10:45:00
nextMoistureValveSensorCheck = 2020-07-14 10:45:00
MQTT: Received PUBLISH (d0, q0, r0, m0), 'SGS/8EDD', ... (97 bytes)

```

```

Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "1", "timestamp":
"07/14/2020 17:30:35", "valvestate": "V00000000"}'
Valve Change Received
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
The time is: 2020-07-14 10:35:01.355834
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
MQTT: Received PUBLISH (d0, q0, r0, m0), 'SGS/8EDD', ... (175 bytes)
Wireless MQTT Message received: b'{"id": "8EDD", "messagetype": "4", "timestamp":
"07/14/2020 17:35:27", "enableSensors": "1,1,1,1", "sensorValues":
"77.85,100.00,100.00,100.00", "sensorType": "C1,C1,C1,C1"}'
Sensor Message Recieved
-----
Processing MQTT Sensor Message
-----
MoistureSensorStates
[{'id': '8EDD', 'sensorType': 'C1', 'sensorNumber': '1', 'sensorValue': '77.85',
'timestamp': '2020-07-14 10:35:27'}, {'id': '8EDD', 'sensorType': 'C1',
'sensorNumber': '2', 'sensorValue': '100.00', 'timestamp': '2020-07-14 10:35:27'},
{'id': '8EDD', 'sensorType': 'C1', 'sensorNumber': '3', 'sensorValue': '100.00',
'timestamp': '2020-07-14 10:35:27'}, {'id': '8EDD', 'sensorType': 'C1',
'sensorNumber': '4', 'sensorValue': '100.00', 'timestamp': '2020-07-14 10:35:27'}]
-----
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
>>>>>>Valve Check<<<<<<
MQTT: Sending PINGREQ
MQTT: Received PINGRESP
The time is: 2020-07-14 10:40:01.355954

```

Configuration of your Smart Garden System

Step 7: Now we will go through all the configuration screens and describe the buttons and fields. You do not have to follow this in the order presented. Remember, you can pick and choose and always go back and change the configuration later and reload your configuration into SGS on the fly without even stopping the program.

The first thing to figure out is what your Raspberry Pi IP address is. Follow the procedures here:

<https://learn.pimoroni.com/tutorial/raspberry-pi/finding-your-raspberry-pi>

Step 8: In a terminal window, change directories to:

```
cd SDL_Pi_SmartGardenSystem2
```

Run SGSConfigure.py:

```
sudo python3 SGSConfigure.py
```

You will see something like this:

```
pi@SwitchDocLabs:~/SDL_Pi_SmartGardenSystem2 $ sudo python3 SGSConfigure.py
SGSConfigure.py:24: DeprecationWarning: AppURLopener style of invoking requests is
deprecated. Use newer urlopen functions/methods
  myURLopener = AppURLopener()
remi.server      INFO      Started httpserver http://0.0.0.0:8001/
remi.request     INFO      built UI (path=/)
SGS.JSON File does not exist
SGSConfiguration.JSON File does not exist
```

Now open a browser window (either on the Raspberry Pi or on another computer on your local WiFi network) and enter this URL:

On your Raspberry Pi: <http://127.0.0.1:8001/>

On your Raspberry Pi:

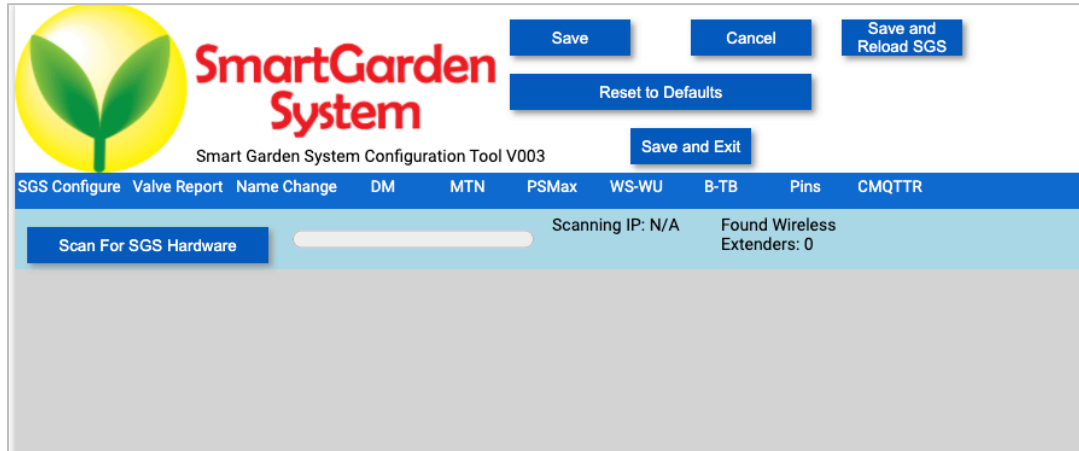
```
http://127.0.0.1:8050/
```

On another computer type:

```
http://xxx.xxx.xxx.xxx:8050/
```

Where “xxx.xxx.xxx.xxx” is the IP address of your Raspberry Pi that you wrote down above.

You will see this screen on your browser (or one of the other pages).



Notice the blue boxes on the top with key commands such as Save, Cancel and Restore to Default. We should understand what they do:

Save Button – Saves your configuration into SGS.JSON and SGSConfiguration.JSON. Does not reload the SGS program with your values.

Cancel Button – Clears all your changes and reloads the original JSON files.

Reset to Defaults – Resets the configuration to the factory default values

Save and Exit – Saves your configuration into SGS.JSON and SGSConfiguration.JSON and exits the program. Does not reload the SGS program with your values.

Save and Reload SGS – Saves the JSON files and dynamically reloads your SGS program with the new values. This will take about 20 or 30 seconds for the SGS program to recognize your request and carefully reinitialize the SGS program.

Smart Garden System
Smart Garden System Configuration Tool V003

Buttons: Save, Cancel, Save and Reload SGS, Reset to Defaults, Save and Exit

Menu: SGS Configure, Valve Report, Name Change, DM, MTN, PSMAX, WS-WU, B-TB, Pins, CMQTT

ReScan For SGS Hardware

Scanning IP: N/A Found Wireless Extenders: 4

Right Flowers / 149D / 192.168.1.34
Right Pots / 529D / 192.168.1.37
Corn Rows / 8EDD / 192.168.1.38
GardenRoom / 14D5 / 192.168.1.42

Valve Select
Right Flowers / 149D / Valve 2

Valve Control
MS#1/Right Flowers/149D

Moisture Sensor Threshold Percent
65

Timer Selection
30 Minutes

Start Time
05:00

On Time Length in Seconds
20

☒ Display Graph

Save Valve

- **Scan For SGS Hardware** - (Rescan for SGS Hardware) – Clicking this button will scan (or rescan) all your local IP numbers on your local network looking for SGS Wireless Units. It will take about 10 minutes to scan your entire network. You can watch the progress in the Scanning IP and Found Wireless Extenders fields. Rescanning your network does not clear the valve and sensor programming on your current units.
- **The Wireless Extender Buttons** - When you select a Wireless Unit by clicking on the Wireless Extender name, you can then add and edit all the valves or pumps on your Wireless unit. The fields of the name are: Name of Unit / Internal ID of Unit / IP Address of Unit. When you click on the unit, a valve configuration screen comes up.

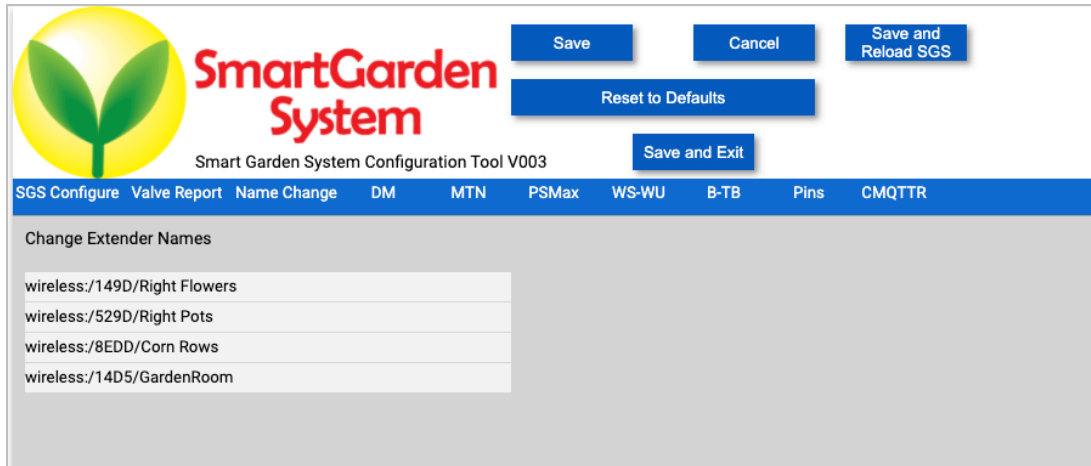
Valve Configuration Menu

The image displays two screenshots of the Valve Configuration Menu. The left screenshot shows the 'Valve Select' dropdown set to 'None Selected' and the 'Valve Control' dropdown open, showing a list of moisture sensors. The right screenshot shows 'Valve Select' set to 'Right Flowers / 149D / Valve 1' and 'Valve Control' set to 'Daily'. It also shows 'Start Time' as 05:00 and 'On Time Length in Seconds' as 10.

- **Valve Select** – Select Valve 1 -8 on the current Wireless Unit
- **Valve Control** – There are two choices. You can select a “timed” valve (turns on and off at a specified length and a specified timing interval) or have the valve controlled by any of the moisture sensors in the entire system. The fields of Timer Selection and Start Time cannot be changed unless you have selected a “timed” value.
- **Moisture Sensor Threshold Percent** – This is the threshold for the pump or valve to be turned on by the selected sensor. It defaults to 65 Percent. By the way, if you set this above 100%, then the valve will be always on. This is one of the ways to turn a valve or fan on or off permanently.
- **Timer Selection** – This dropdown menu selects the repetition frequency for the timed valve. The first fire of your timer will be after the Start Time and after the current time aligned on the appropriate clock time. For example, selecting a 15-minute interval at 13:12 (use 24 hour time in the field) will start your first on time at 13:15, the next at 13:30, and so on.
- **Start Time** – This is the time to start calculating the next “timed event” or when to do the on and off event in the case of the Daily event.
- **On Time Length in Seconds** – When a timed event happens, this is where you specify how long to turn the valve or pump on in seconds. This is a second way to make a valve or pump turn on permanently. For example, if you select a 15-minute timed event and set them On Time Length In Seconds to something longer than 15 minutes (over 900), then the Valve will never turn off. Very handy!
- **Display Graph** – If you click this, then the Moisture Sensor selected (if you are connected to a moisture sensor event) and the Valve On/Off graph will be displayed on the Dash_App.
- **Save Valve** – This saves the configuration of the valve. Warning: Selecting another valve will clear your changes if you do not hit “Save Valve”.

SGS Valve Report

The Valve Report tab shows the current configuration of all your valves. This is a great place to check if you have configured your valves the way you intended.

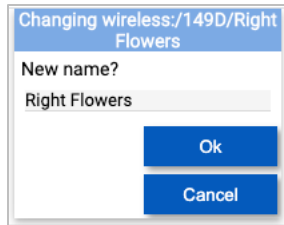



The image shows the 'Smart Garden System Configuration Tool V003' interface. At the top left is a logo of a green plant with a yellow sun-like background. To its right is the text 'SmartGarden System' in red and black. Below this is 'Smart Garden System Configuration Tool V003'. On the right side of the top section are five buttons: 'Save', 'Cancel', 'Save and Reload SGS', 'Reset to Defaults', and 'Save and Exit'. Below the top section is a blue navigation bar with the following tabs: 'SGS Configure', 'Valve Report' (which is highlighted), 'Name Change', 'DM', 'MTN', 'PSMax', 'WS-WU', 'B-TB', 'Pins', and 'CMQTTR'. The main content area below the navigation bar is titled 'Change Extender Names' and contains a table with four rows of text:

wireless:/149D/Right Flowers
wireless:/529D/Right Pots
wireless:/8EDD/Corn Rows
wireless:/14D5/GardenRoom

SGS Name Change

The Name Change tab is used to change the Wireless Extender Names. Click on the name of the extender to bring up the name change popup.





Smart Garden System

Smart Garden System Configuration Tool V003


Buttons: Save, Cancel, Save and Reload SGS, Reset to Defaults, Save and Exit

Navigation: SGS Configure, Valve Report, Name Change, DM, MTN, PSMAX, WS-WU, B-TB, Pins, CMQTTR

Valve Configuration Report

ID	Unit Name	Valve Number	Control	MS Threshold	Time Select	Start Time	On Time (seconds)
149D	Right Flowers	1	Off	65	Daily	05:00	10
149D	Right Flowers	2	MS#1/Right Flowers/149D	65	30 Minutes	05:00	20
149D	Right Flowers	3	Off	65	Daily	05:00	10
149D	Right Flowers	4	Off	65	Daily	05:00	10
149D	Right Flowers	5	Off	65	Daily	05:00	10
149D	Right Flowers	6	Off	65	Daily	05:00	10
149D	Right Flowers	7	Off	65	Daily	05:00	10
149D	Right Flowers	8	Timed	65	15 Minutes	05:00	600
529D	Right Pots	1	Timed	65	15 Minutes	05:00	10
529D	Right Pots	2	Timed	65	15 Minutes	05:00	10

- **Debug Configuration** - enable SW Debugging – Checking this box enables software debugging on the SGS program and produces substantially more output to the console terminal window where you are running the SGS program.
- **MySQL Configuration** – Enable MySQL Logging – This tells the SGS program to log lots of information to the Raspberry Pi MySQL database for the Smart Garden System. Very useful and needed to produce graphs and charts in the Dash App. Especially required for the Weather Add-on for the Smart Garden System.
- **MySQL Configuration** – Password – This is the password on the “root” user of your MySQL application on your Raspberry Pi. Defaults to “password” which is the default password on the MySQL on the SwitchDoc Labs SD Card.
- **Enable Manual Watering** – Checking this box allows the Blynk App to manually turn valves on for a specified time. Uncheck this if you have a Garden that you don’t want people to manually turn on valves, but still want them to use Blynk to see what is going on with your Garden (like a school garden for example).
- **Tank Pump Level** – Future Enhancement.
- **Use Metric Units** – Checking this box will set the units used by the Smart Garden System to metric (versus the default of English units).



Smart Garden System

Smart Garden System Configuration Tool V003

Save

Cancel

Save and Reload SGS

Reset to Defaults

Save and Exit

SGS ConfigureValve ReportName ChangeDMMTNPSMaxWS-WUB-TBPinsCMQTTR

Debug / MySQL / MW Tab

Debug Configuration

☒enable SW Debugging

MySQL Configuration

☒enable MySQL Logging

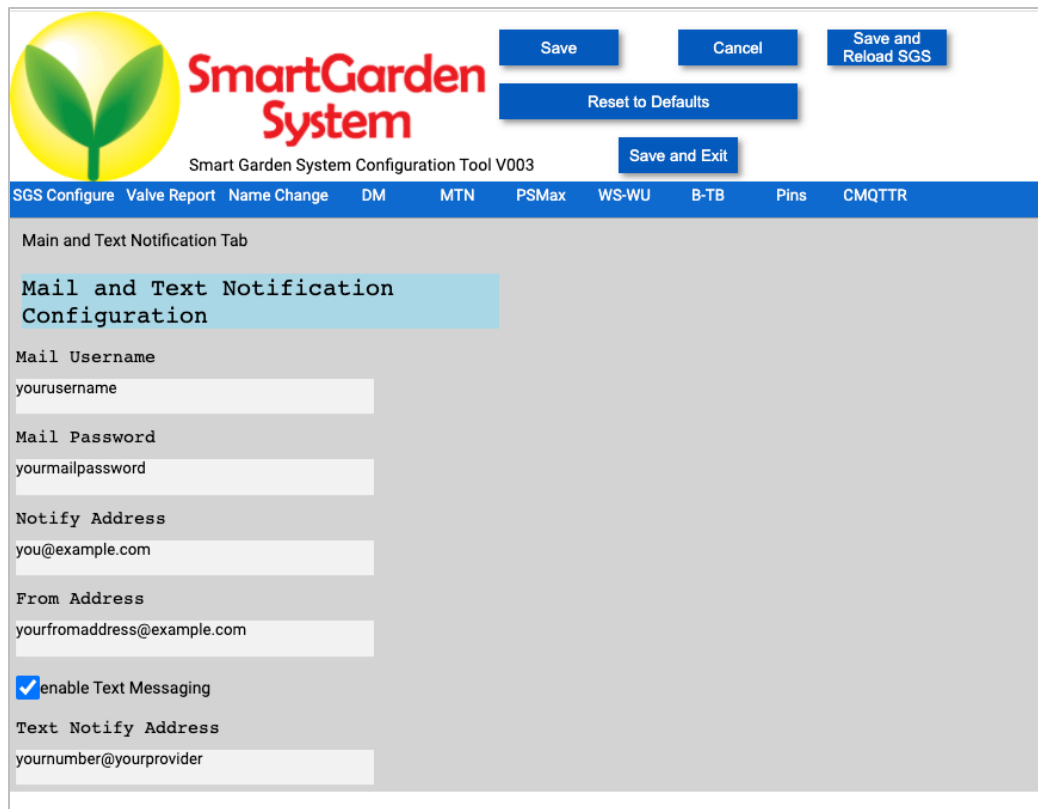
password

☒enable Manual Watering

Tank Pump Level

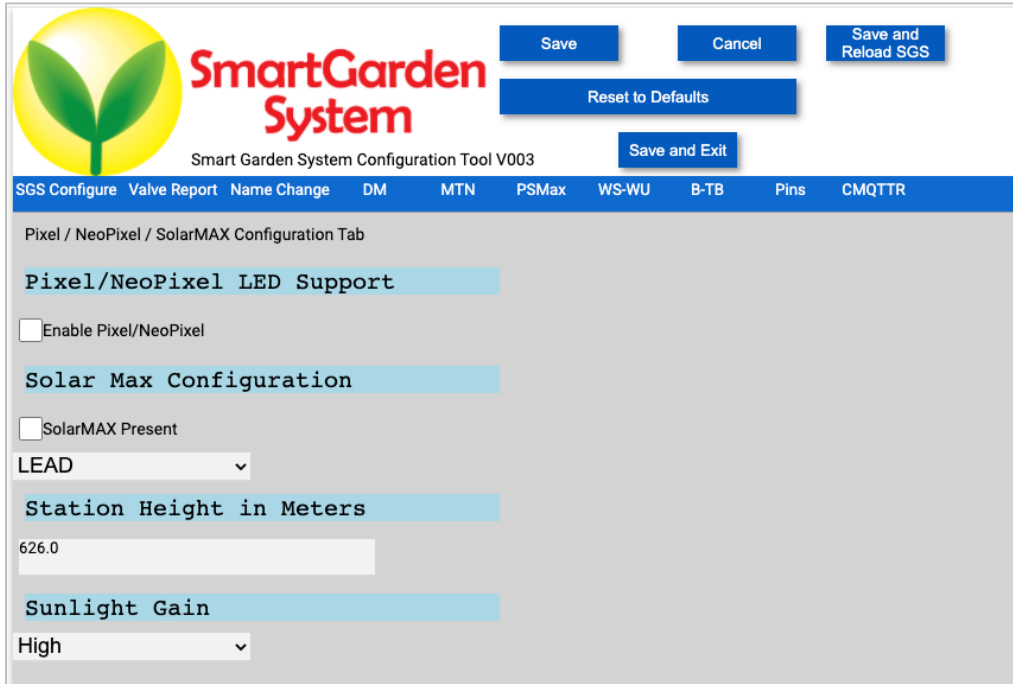
15.0

☐Use Metric Units (default English)



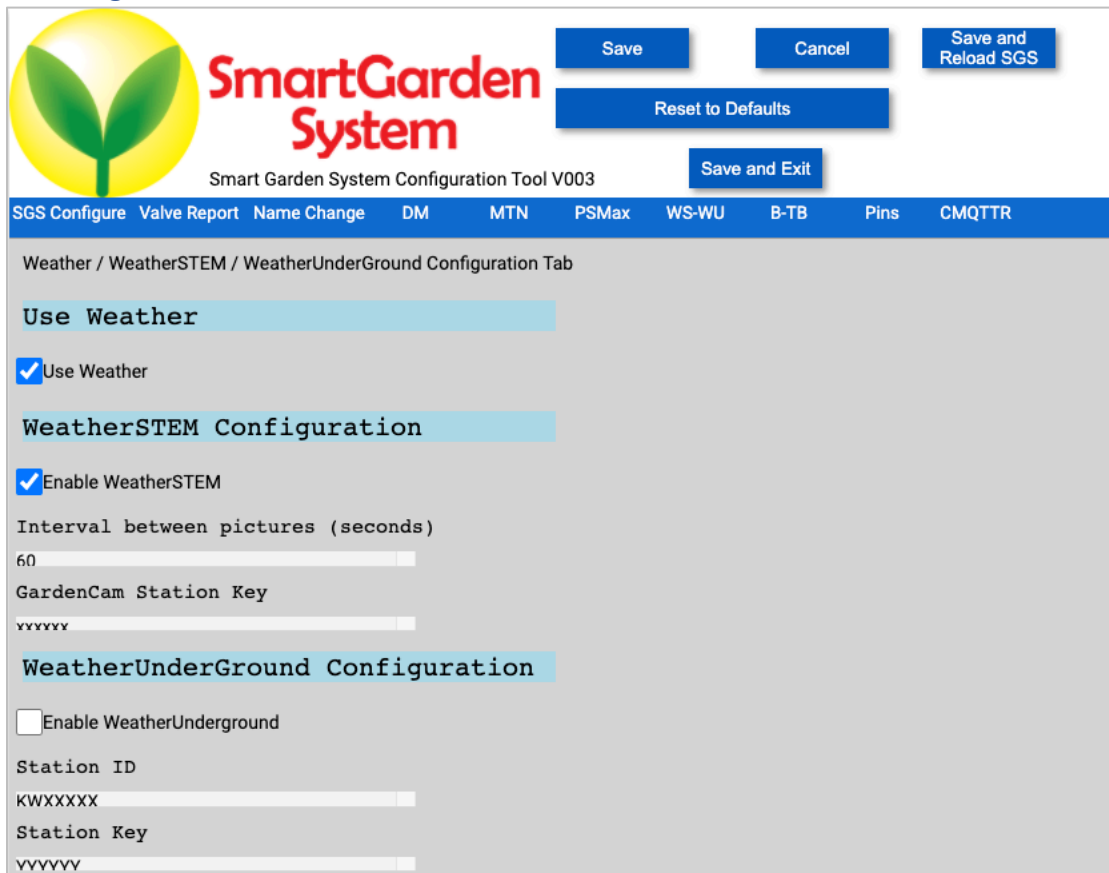
The image shows a web-based configuration tool for the Smart Garden System. At the top left is the Smart Garden System logo, which consists of a yellow circle with a green leaf inside. To the right of the logo is the text "Smart Garden System" in red. Below the logo and text is the version number "Smart Garden System Configuration Tool V003". On the right side of the top bar are five buttons: "Save", "Cancel", "Save and Reload SGS", "Reset to Defaults", and "Save and Exit". Below the top bar is a navigation menu with the following items: "SGS Configure", "Valve Report", "Name Change", "DM", "MTN", "PSMax", "WS-WU", "B-TB", "Pins", and "CMQTTR". The main content area is titled "Main and Text Notification Tab" and contains a section titled "Mail and Text Notification Configuration". This section has several input fields: "Mail Username" with the placeholder "yourusername", "Mail Password" with the placeholder "yourmailpassword", "Notify Address" with the placeholder "you@example.com", "From Address" with the placeholder "yourfromaddress@example.com", and "Text Notify Address" with the placeholder "yournumber@yourprovider". There is also a checkbox labeled "enable Text Messaging" which is checked.

- **Mail and Text Notification Configuration** – Mail Username – The username for your email account – for Gmail (a good choice!) would be myname@gmail.com.
- **Mail and Text Notification Configuration** – Mail Password – The password for your email account.
- **Mail and Text Notification Configuration** – Text Notification Address – Text updates from Smart Garden System will be sent to this email address, customized for your mobile text provider (see this website: <https://www.dialmycalls.com/blog/send-text-messages-email-address>).
- **Mail and Text Notification Configuration** – From Address – Enter the email address you would like your notifications and email to come from. Note: For some mail servers, this will need to match your account information.



The image shows a web-based configuration tool for the Smart Garden System. At the top left is a logo with a green plant inside a yellow circle. To its right is the text "SmartGarden System" in red and black. Below this is "Smart Garden System Configuration Tool V003". On the top right are buttons: "Save", "Cancel", "Save and Reload SGS", "Reset to Defaults", and "Save and Exit". A blue navigation bar contains links: "SGS Configure", "Valve Report", "Name Change", "DM", "MTN", "PSMax", "WS-WU", "B-TB", "Pins", and "CMQTTR". The main content area is titled "Pixel / NeoPixel / SolarMAX Configuration Tab". It has four sections: "Pixel/NeoPixel LED Support" with an "Enable Pixel/NeoPixel" checkbox; "Solar Max Configuration" with a "SolarMAX Present" checkbox and a "LEAD" dropdown menu; "Station Height in Meters" with a text input field containing "626.0"; and "Sunlight Gain" with a dropdown menu set to "High".

- **Pixel Support** – Future Enhancement.
- **SolarMAX Configuration** – Future Enhancement.
- **Station Height in Meters** – Used for Weather Station add-on and the on-board barometer for calculating the Sea Level Barometric Pressure.
- **Sunlight Gain** – Future Enhancement.



The image shows the 'Smart Garden System Configuration Tool V003' interface. At the top, there is a logo with a green leaf inside a yellow circle, followed by the text 'Smart Garden System'. To the right of the logo are buttons for 'Save', 'Cancel', 'Save and Reload SGS', 'Reset to Defaults', and 'Save and Exit'. Below the logo and text is a navigation bar with tabs: 'SGS Configure', 'Valve Report', 'Name Change', 'DM', 'MTN', 'PSMax', 'WS-WU', 'B-TB', 'Pins', and 'CMQTTR'. The 'SGS Configure' tab is selected. Below the navigation bar, the text 'Weather / WeatherSTEM / WeatherUnderGround Configuration Tab' is displayed. The main content area has a light blue header 'Use Weather' with a checked checkbox 'Use Weather'. Below this is another light blue header 'WeatherSTEM Configuration' with a checked checkbox 'Enable WeatherSTEM'. Under 'WeatherSTEM Configuration', there is a label 'Interval between pictures (seconds)' with a value of '60' and a text input field. Below that is a label 'GardenCam Station Key' with a text input field containing 'yyyyyy'. The next section is 'WeatherUnderGround Configuration' with an unchecked checkbox 'Enable WeatherUnderground'. Below this are two text input fields: 'Station ID' with 'KWXXXXX' and 'Station Key' with 'yyyyyy'.

- **Use Weather** – Check this box if you have the Smart Garden System Weather add-on kit
- **WeatherSTEM Configuration – Enable WeatherSTEM** – Turns on support for WeatherSTEM and the GardenCam Cloud based server
- **WeatherSTEM Configuration – Interval between GardenCam pictures in seconds** – How often to send a picture to WeatherSTEM.
- **WeatherSTEM Configuration – GardenCam (SkyWeather) Station Key for WeatherSTEM** - Located on your Weather add-on Packing slip.
- **WeatherUnderground Configuration – Enable WeatherUnderground** – Legacy Support for WeatherUnderground. Not recommended for new installations.
- **WeatherUnderground Configuration – Weather Underground Station ID** – Legacy Support for WeatherUnderground. Not recommended for new installations.
- **WeatherUnderground Configuration – Station Key** – Legacy Support for WeatherUnderground. Not recommended for new installations.

SGS Blynk and ThunderBoard Configuration

The Smart Garden System mobile application is built on Blynk. See the Smart Garden System Blynk Installation manual for details.

SmartGarden System
Smart Garden System Configuration Tool V003

SGS Configure Valve Report Name Change DM MTN PSMax WS-WU B-TB Pins CMQTTR

Blynk / ThunderBoard AS3935 Tab

Blynk Configuration

☒ Enable Blynk

Blynk App Authorization
daff75da224b447b98ce2e5b0abe5312

ThunderBoard AS3935 Configuration

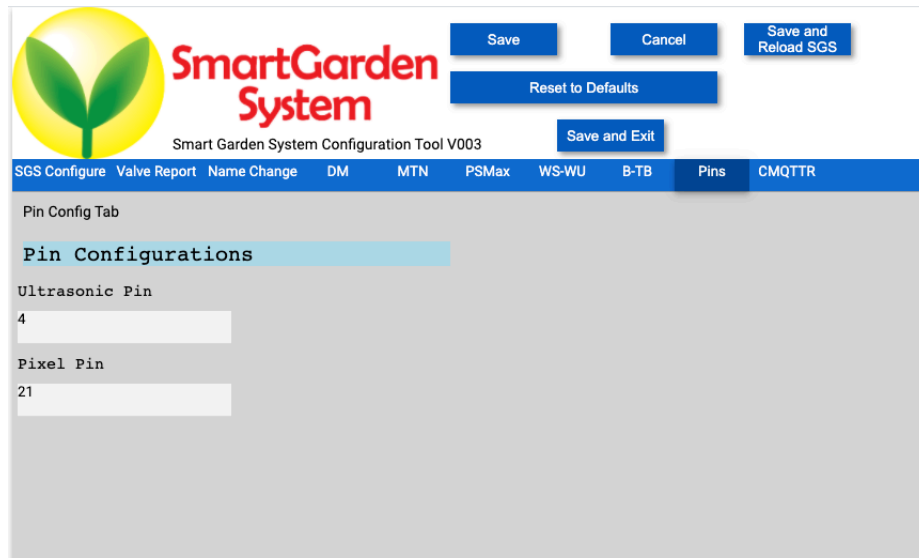
Format:[NoiseFloor, Indoor, TuneCap, DisturberDetection, WatchDogThreshold, SpikeDetection]

Thunderboard Configuration
[2,1,3,0,3,3]

- **Blynk Configuration – Enable Blynk** - Enables the usage of Blynk in the Smart Garden System.
- **Blynk Configuration – Blynk App Authorization** - This is an example number – it won't work for you. This is the number that ties your Blynk app on your phone to your Smart Garden System.
- **ThunderBoard AS3935 Configuration – Local Lightning Detection Kit** - Future Enhancement.

SGS Physical Pin Configurations

This screen is for physical pin mapping to your Raspberry Pi and Smart Garden System.



The screenshot shows the 'Smart Garden System Configuration Tool V003' interface. At the top left is a logo of a green plant in a yellow circle. To its right is the text 'SmartGarden System' in red and black. Below this is 'Smart Garden System Configuration Tool V003'. On the right side, there are five buttons: 'Save', 'Cancel', 'Save and Reload SGS', 'Reset to Defaults', and 'Save and Exit'. Below the header is a blue navigation bar with tabs: 'SGS Configure', 'Valve Report', 'Name Change', 'DM', 'MTN', 'PSMax', 'WS-WU', 'B-TB', 'Pins', and 'CMQTTR'. The 'Pins' tab is selected. Below the navigation bar is a 'Pin Config Tab' section. Under this, there is a 'Pin Configurations' section with a light blue background. It contains two entries: 'Ultrasonic Pin' with a value of '4' and 'Pixel Pin' with a value of '21'. Each entry has a text input field next to the value.

- **Pin Configuration** – Ultrasonic Pin – Future Enhancement
- **Pin Configuration** – Pixel Pin – Future Enhancement

SmartGarden System
Smart Garden System Configuration Tool V003

SGS Configure Valve Report Name Change DM MTN PSMAX WS-WU B-TB Pins CMQTT

Camera / MQTT / Rest Tab

Night Camera Enable

☐ Night Vision Enable

REST Interface

☐ REST Enable

MQTT Configuration (SGS OUT to other broker)

☐ MQTT Enable

MQTT Server URL

MQTT Server Port Number
5900

How Often MQTT Sent in Seconds
500

Save Cancel Save and Reload SGS Reset to Defaults Save and Exit

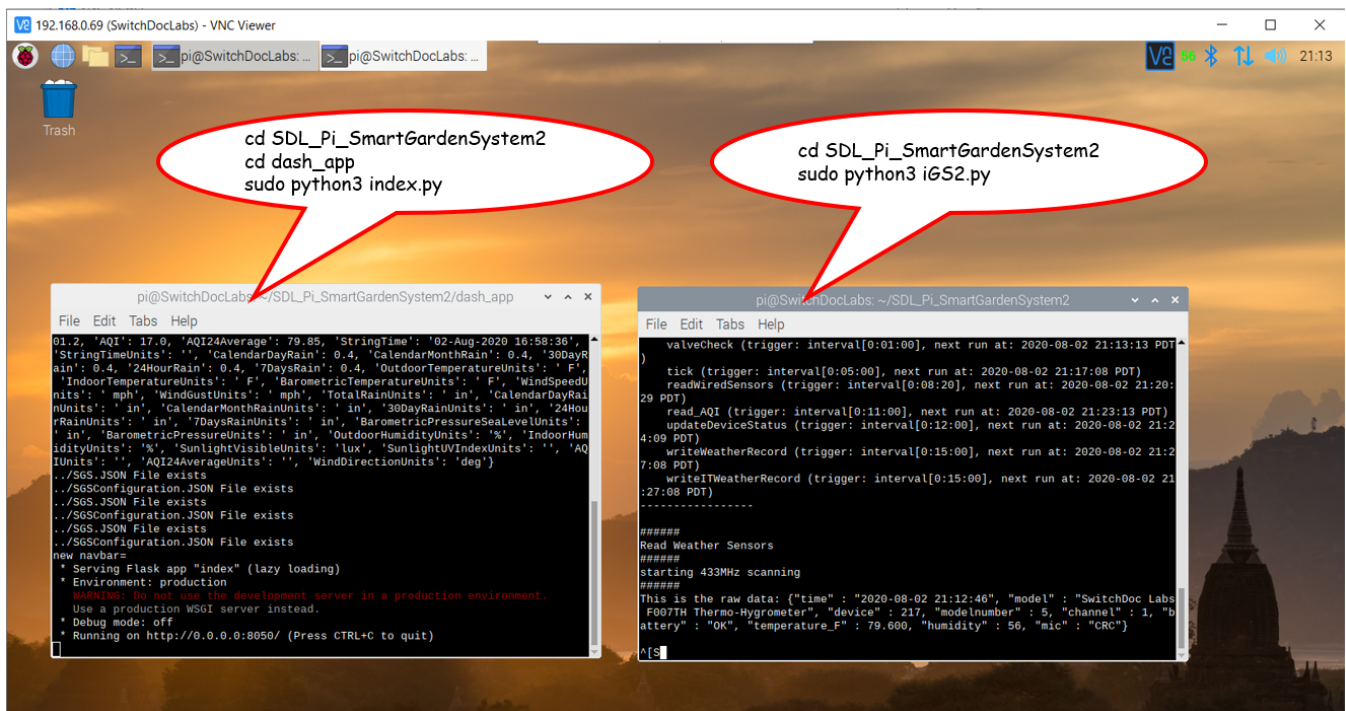
- **Night Camera Enable – Night Vision Enable** – Enables night vision option for the Garden Cam supplied with the Weather add-on – For future enhancement.
- **REST Interface – REST Enable** – Enables the REST interface to the Smart Garden System on the Raspberry Pi – Future enhancement.
- **MQTT Configuration** is for the use of an MQTT interface FROM the Smart Garden System to another computer of your choosing. For future enhancement.
- **MQTT Configuration – MQTT Enable** – For future enhancement.
- **MQTT Configuration – MQTT Server URL** – For future enhancement.
- **MQTT Configuration – MQTT Server Port Number** – For future enhancement.
- **MQTT Configuration – How Often MQTT Send in Seconds** – For future enhancement.

Dash Application Screens

Dash is a Python framework for building web applications. It built on top of Flask, Plotly.js, React and React Js. It enables you to build dashboards using pure Python. Dash is open source, and its apps run on the web browser. The Smart Garden System uses Dash to display live information about what is going on in the Smart Garden System and in your Garden.

Step 9: You will open a second Terminal Window in your Raspberry pi to start Dash by typing the following:

```
cd SDL_Pi_SmartGardenSystem2
cd dash_app
sudo python3 index.py
```



Step: 10 Next, you will open your favorite browser to see live streaming data!

On your Raspberry Pi:

<http://127.0.0.1:8050/>

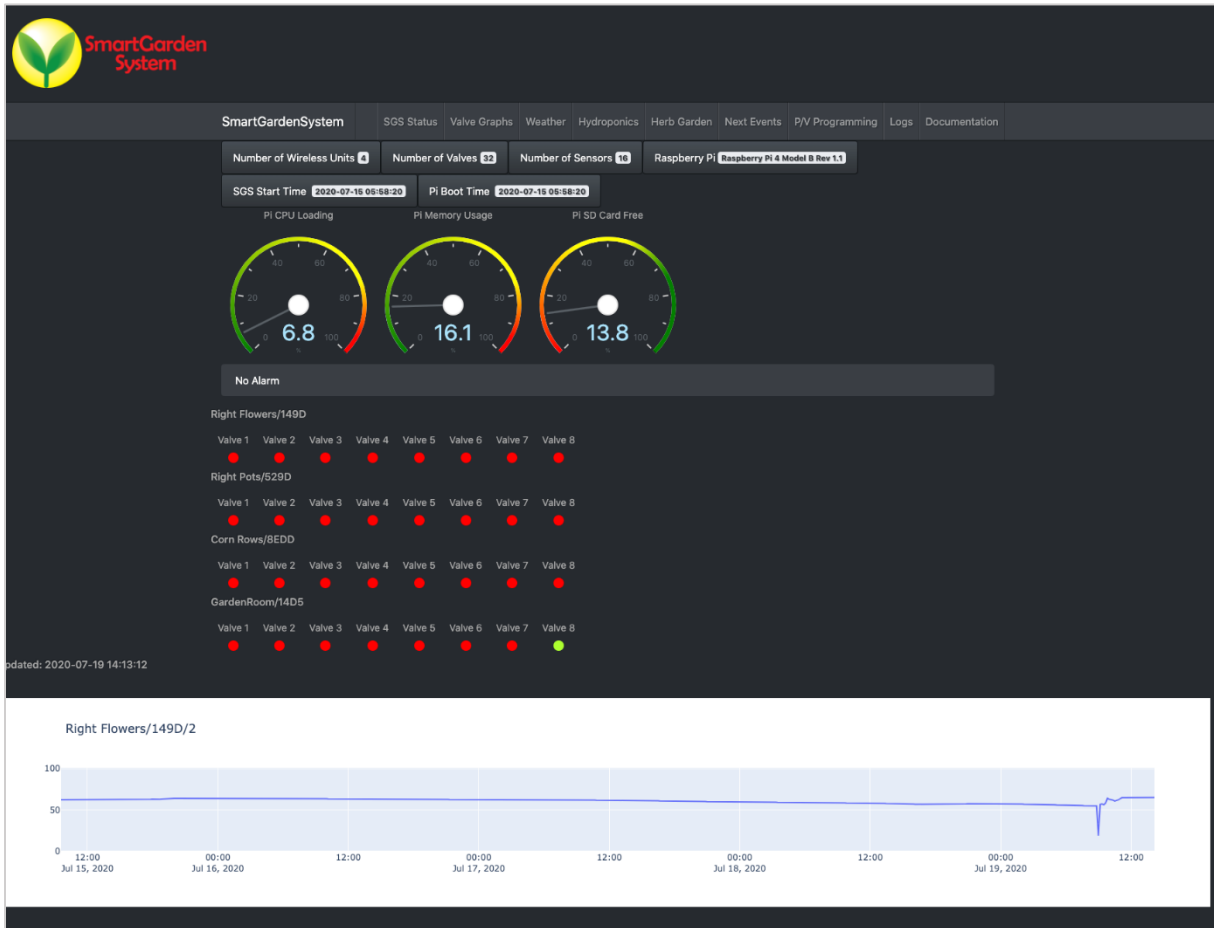
On another computer type:

<http://xxx.xxx.xxx.xxx:8050/>

Where "xxx.xxx.xxx.xxx" is the IP address of your Raspberry Pi that you wrote down above.

Dash Application Screens

After you start the dash app, you will have a screen with several tabs in your browser. Click on each respective tab to see the following screens.



SGS Status Tab

The SGS Status tab has a set of text boxes giving information about the current state of the Raspberry Pi and the Smart Garden System running. It is broken down into five different regions:



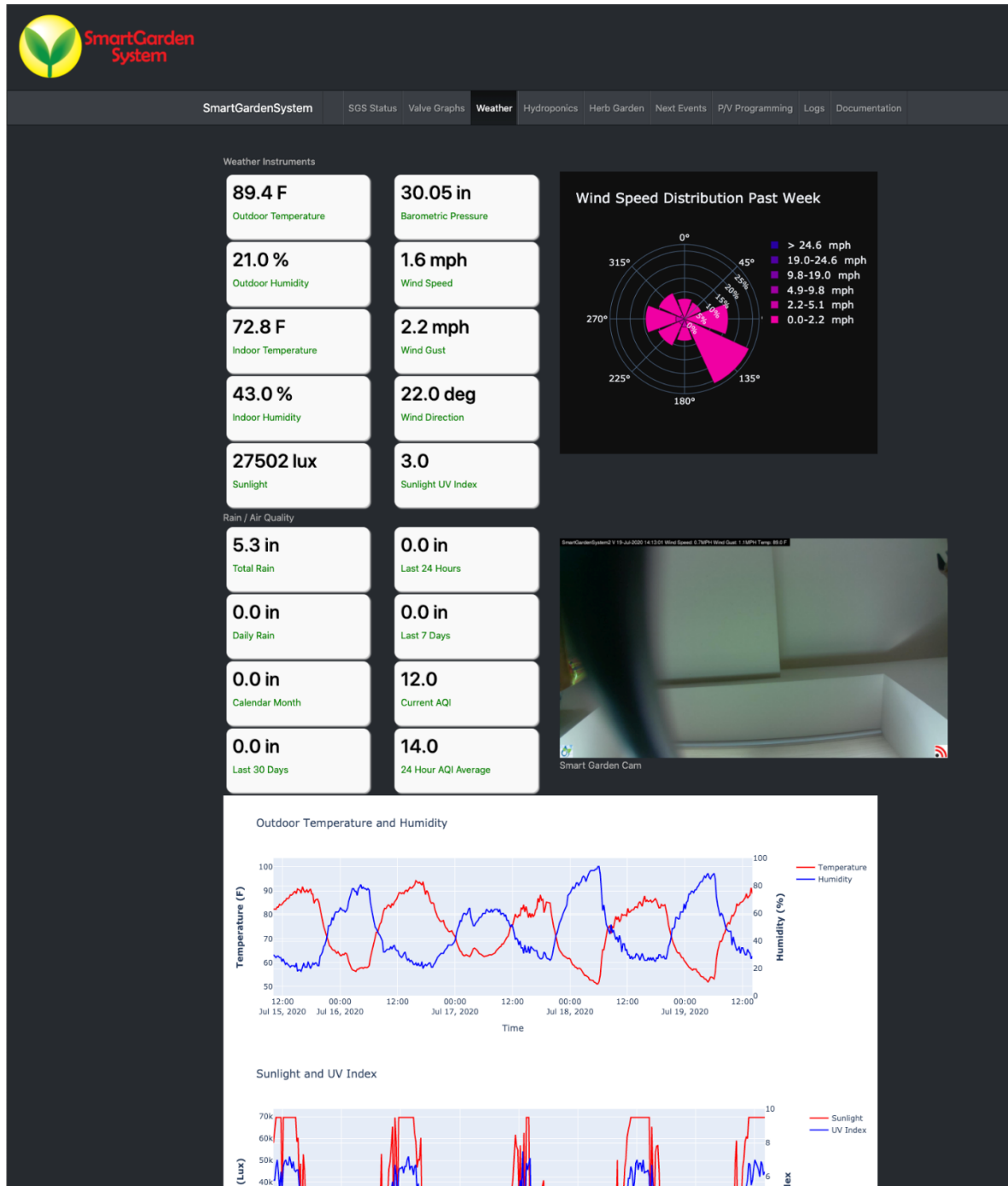
- **Text Blocks** Containing information about the SGS System and the type of Raspberry Pi running including time of the last boot of the Raspberry Pi.
- **Raspberry Pi Dashboard Gauges** showing the dynamic state of the Raspberry Pi.
- The **Alarm Reporting** Region – For future enhancement.
- **Current Valve State** for all the GSG Wireless Extenders
- **Selected Moisture Sensors** (from the “Save Graph” checkbox on the SGSConfigure program).

Valve Graphs Tab

This set of graphs shows when the selected valves (are turning on and off again, selected on the SGSConfigure Valve Screen).

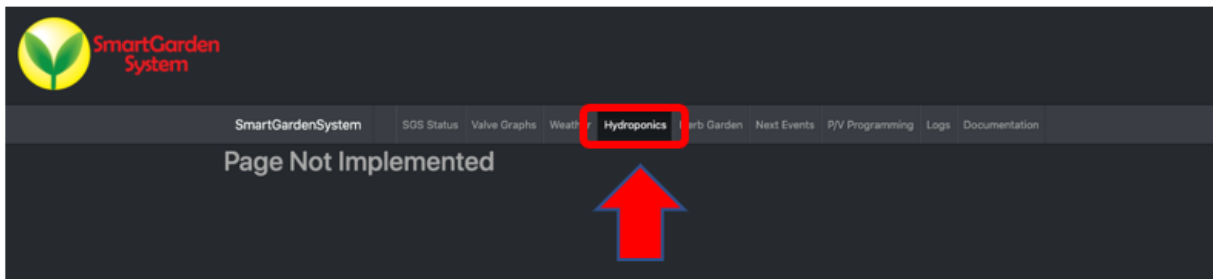
Weather Tab

The Weather Tab shows your current local weather conditions in your garden. This information comes from the optional SGS weather station add-on which includes 8 different sensors. Yes, we know the camera was NOT pointing at the garden. Silly engineers. The data from the optional Air Quality Sensor (at the bottom of the Weather Tab) is shown below:



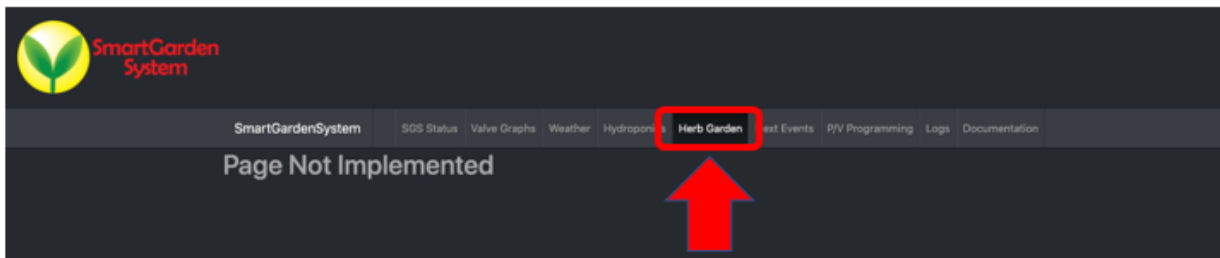
Hydroponics Tab

This tab will show the status of the Hydroponics add-on for the Smart Garden System.



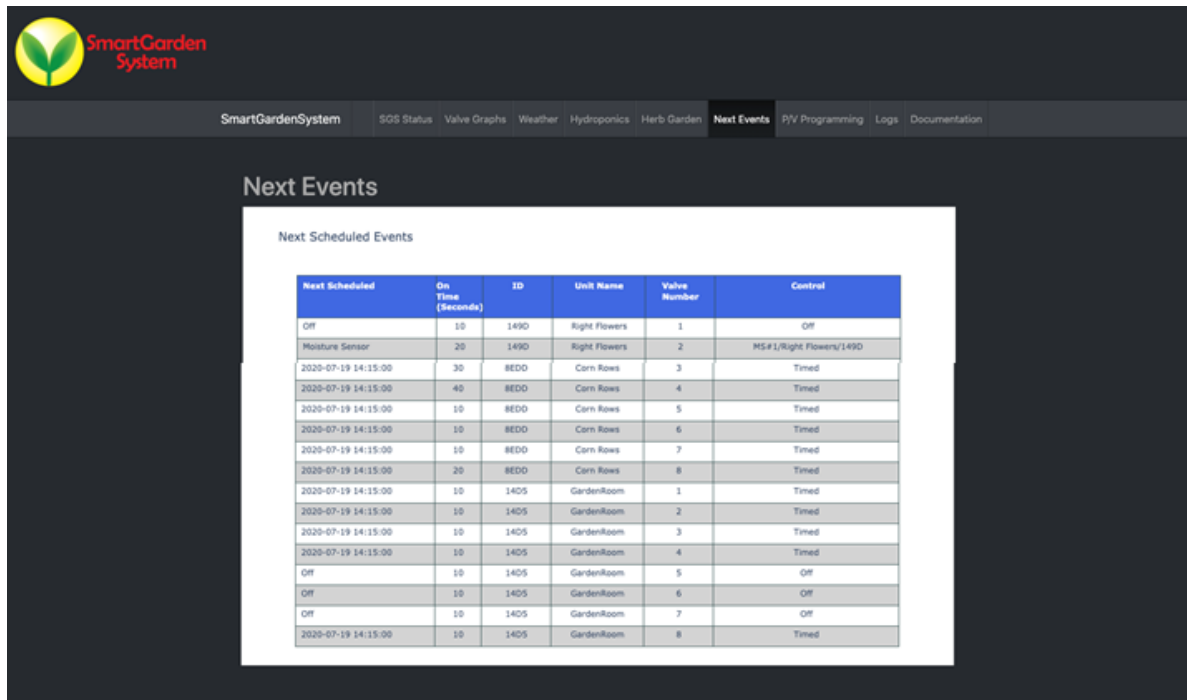
Herb Garden Tab

This tab will show all the status (including a camera) of your herb garden.



Next Events Tab

The next events tab shows a summary of all the valves in your Smart Garden System and is specifically focusing showing the next timed events for all of your valves. This is a great way of checking that your timed events are going to happen at the time you intended.



The screenshot displays the SmartGardenSystem web interface. The top navigation bar includes the logo and a menu with the following items: SmartGardenSystem, SGS Status, Valve Graphs, Weather, Hydroponics, Herb Garden, **Next Events**, RV Programming, Logs, and Documentation. The main content area is titled "Next Events" and contains a sub-header "Next Scheduled Events" above a table.


Next Scheduled	On Time (Seconds)	ID	Unit Name	Valve Number	Control
Off	10	1490	Right Flowers	1	Off
Moisture Sensor	20	1490	Right Flowers	2	MS#1/Right Flowers/1490
2020-07-19 14:15:00	30	8ED0	Corn Rows	3	Timed
2020-07-19 14:15:00	40	8ED0	Corn Rows	4	Timed
2020-07-19 14:15:00	10	8ED0	Corn Rows	5	Timed
2020-07-19 14:15:00	10	8ED0	Corn Rows	6	Timed
2020-07-19 14:15:00	10	8ED0	Corn Rows	7	Timed
2020-07-19 14:15:00	20	8ED0	Corn Rows	8	Timed
2020-07-19 14:15:00	10	1405	GardenRoom	1	Timed
2020-07-19 14:15:00	10	1405	GardenRoom	2	Timed
2020-07-19 14:15:00	10	1405	GardenRoom	3	Timed
2020-07-19 14:15:00	10	1405	GardenRoom	4	Timed
Off	10	1405	GardenRoom	5	Off
Off	10	1405	GardenRoom	6	Off
Off	10	1405	GardenRoom	7	Off
2020-07-19 14:15:00	10	1405	GardenRoom	8	Timed

P/V Programming Tab

The PV Programming tab is designed to show at a glance all the programming of your valves. Again, a great way to check your programming of your Smart Garden System.

Logs Tab

The Logs tab shows the last lines in the Log files maintained by the Smart Garden System. The main system log shows events, alarms, debugging events and status of your SGS wireless extenders. The Valve Log shows valve events, their source and any critical. The Sensor log shows you the latest sensor readings from your wireless extenders.



SmartGarden System

SmartGardenSystem | SGS Status | Valve Graphs | Weather | Hydroponics | Herb Garden | Next Events | P/V Programming | **Logs** | Documentation

System Log

TimeStamp	Level	Description
2020-07-19T12:20:33	INFO	Main Weather Sensors Found
2020-07-19T12:20:31	INFO	Indoor Weather Sensor Found
2020-07-19T12:19:46	INFO	Wireless Device ID 14D5 Active
2020-07-19T12:19:45	INFO	Wireless Device ID 8EDD Active
2020-07-19T12:19:43	INFO	Wireless Device ID 529D Active
2020-07-19T12:19:42	INFO	Wireless Device ID 149D Active
2020-07-19T12:19:38	INFO	Garden Cam Present
2020-07-19T12:19:38	CRITICAL	No Alarm

Valve Log

TimeStamp	DeviceID	Valve Number	State	Source	Seconds On	Valve Type
2020-07-19T14:00	14D5	8	1	Timer Event	10	
2020-07-19T14:00	14D5	4	1	Timer Event	10	
2020-07-19T14:00	14D5	3	1	Timer Event	10	
2020-07-19T14:00	14D5	2	1	Timer Event	10	
2020-07-19T14:00	14D5	1	1	Timer Event	10	
2020-07-19T14:00	8EDD	8	1	Timer Event	20	
2020-07-19T14:00	8EDD	7	1	Timer Event	10	
2020-07-19T14:00	8EDD	6	1	Timer Event	10	

Sensor Log

TimeStamp	DeviceID	Sensor Number	Sensor Value	Sensor Type On	TimeStamp Type
2020-07-19T14:11:38	14D5	4	100	C1	2020-07-19T14:08:45
2020-07-19T14:11:38	14D5	3	100	C1	2020-07-19T14:08:45
2020-07-19T14:11:38	14D5	2	100	C1	2020-07-19T14:08:45
2020-07-19T14:11:38	14D5	1	100	C1	2020-07-19T14:08:45
2020-07-19T14:11:38	8EDD	4	100	C1	2020-07-19T14:11:38
2020-07-19T14:11:38	8EDD	3	12.22	C1	2020-07-19T14:11:38
2020-07-19T14:11:38	8EDD	2	21.74	C1	2020-07-19T14:11:38
2020-07-19T14:11:38	8EDD	1	21.95	C1	2020-07-19T14:11:38

Documentation Tab

The documentation tab takes you to the Smart Garden System product page on the shop.switchdoc.com website. This is where you can find all the current documentation for the Smart Garden System in the “Downloads” section of this page.

FREE SHIPPING ON US ORDERS OVER \$199

SwitchDoc Labs

HomeShopTutorialsSupport ForumContact Us

QUG

Smart Garden System (V2) - Raspberry Pi based Smart Gardening Kit - No Soldering!

\$175.00

Quantity
1

ADD TO CART

Buy with **PayPal**

[More payment options](#)

The Smart Garden System - Raspberry Pi Plant Watering and Environmental Kit with Wireless Sensors

The Wireless Smart Garden System is finally here! Did you ever want to build your own remote monitoring and management system for your indoor or outdoor garden? Do you want to share your garden and the weather world wide? This project is for you. You can learn the Raspberry Pi and how to connect to the real world through this easy to build no-soldering kit. You can measure soil moisture and then use that as feedback to provide your plant or garden just the right about of water.

Highly expandable! You can have your Raspberry Pi Base Unit inside your house and have multiple wireless control units in your outdoor garden, greenhouse or in the upstairs bedroom. Up to 250 wireless control units can be connected up to one Raspberry Pi base unit. This allows you to control your truly MASSIVE garden. Or your small one. Either way!

And you get all the source code for the Smart Garden System software so you can do what you want. Modify the code to your own application! Add sensors! Add functions! Add more plants!

Quick Demo of the Smart Garden System

Disclaimer

SwitchDoc Labs, LLC takes no responsibility for any physical injuries and possession loss caused by those reasons which are not related to product quality, such as operating without following the operating manual and cautions, natural disasters, or force majeure.

SwitchDoc Labs, LLC has compiled and published this manual which covers the latest product description and specification. The contents of this manual are subject to change without notice.