Water Board Deployment Guide

MIT CityFARM

This guide focuses on the deployment of a Water Board, from installing the operating system on the Raspberry Pi, to configuring and setting up the board to make measurements.

Items needed:

1. Water Board Working network
2. Internet connection
3. Server running database and web management interface
4. Server URL or IP Address written down
5. Linux computer with SD Card reader
6. 8Gb SD Card and Raspberry Pi

Steps:

1. Files and information needed.

1.1 To quickly get up and running, you may choose to use the MIT CityFARM’s prebuilt Arch Linux, and Image Preparation Utility, however you are free to set up the Operating System however you like.

The prebuilt Arch Linux images can be downloaded on CityFARM’s Google Drive, and the utility and required files can be downloaded from GitHub on the following links:

Image: <https://drive.google.com/file/d/0Bzu5DJ7GsPj5WG1rY1U2VHl6czA/view>

Utilities & Files: <https://github.com/MIT-CityFARM/sensor-board>

By downloading the contents of the *“demo”* folder on GitHub, along with the image file, you’ll have all the needed files.

1.2 Copy the contents of the *“rpi”[1]*, *“utilities”[2]* and unzip the image file to a common folder. You must have the following files inside the same folder:

*- config.json*

*- image\_utility.py*

*- rpi\_service.py*

*- rpi\_service.service*

*- rpi\_sh.sh*

*- sensor\_terminal.py*

*- serialsensor.py*

*- archlinux\_cityfarm\_image\_3\_12\_34\_ARCH.img* (Currently the latest)

1.3 Determine the path of the SD Card.

To Determine the SD Card’s path, first disconnect the SD Card. Then on a terminal window, list the files on *“/dev”*:

*user$ ls /dev*

Which will output something similar to the following:

bus loop1 sda2 tty0

cdrom pts sda5 ttyUSB0

cdrw ram0 stderr

dvd sda stdin

dvdrw sda1 stdout

Now insert the SD Card, and execute the command again:

*user$ ls /dev*

bus loop1 sda2 stderr

cdrom pts sda5 stdin

cdrw ram0 sdb stdout

dvd sda sdb1 🡨 tty0

dvdrw sda1 sdb2 🡨 ttyUSB0

Observe that the disk sdb, and, some of its partitions (sdb1 and sdb2) have appeared on the *“/dev”* folder. This means that (in this case) *“/dev/sdb”* is the SD Card, and the path we are looking for.

Note that *“sdb”* is an example, which may vary according to each system.

1. Configuring Arch Linux image and burning image on SD Card.

2.1 We can now proceed to executing the *“image\_utility.py”* and start configuring the Arch Linux image.

On a terminal window, go to the directory containing the files from step 1.2:

*user$ cd /path/to/files*

You may check you’re in the right folder by listing the files:

*user$ ls*

archlinux...img rpi\_service.service

config.json rpi\_sh.sh

image\_utility.py sensor\_terminal.py

rpi\_service.py serialsensor.py

2.2 Execute *“image\_utility.py”:*

The *“image\_utility.py”* takes 2 required arguments, and optional additional files:

*image\_utility.py TARGET\_DISK SOURCE\_IMAGE [ADDITIONAL\_FILES]...*

Target Disk: The SD Card path, determined in step 1.3

Source Image: Arch Linux image on the current folder

Additional files (Optional): Any files the user wants to be added to the custom image

On the same terminal window, execute the command­[3]:

*user$ sudo ./image\_utility.py /dev/sdb archlinux\_...\_ARCH.img*

[sudo] password for user:

The command must be run as superuser, hence the *“sudo”*  command at the beginning, it’ll then ask for the user’s password. Type the password and hit enter.

The utility will ask for the following information:

Enter the database server URL or IP Address:

Enter the database username:

Enter the database password:

Enter the required information, that should be known beforehand. Breakdown of information:

Server URL/IP: URL or IP Address to the server running the database used for storing sensor data and settings.

Username and Password: Login information for authentication on the database, this is not the server’s user login information.

The Login information will be used to store and retrieve the board’s settings, log, and sensor data, so it is required. If the information is incorrect you will not be able to configure the board through the web interface, nor will the board be able to upload sensor data.

After entering the 3 pieces of information required, the utility will proceed to mount the image, copy the files, set the permissions, add the server information then unmounts the image. This is the typical output during this process:

All files found.

Target Disk: /dev/sdb

Source Image: archlinux\_cityfarm\_image\_3\_12\_34\_ARCH.img

Press Enter to continue...

Press Enter if the information is correct

Mounting image...

umount: mount\_point\_archlinux\_cityfarm\_image\_3\_12\_34\_ARCH.img: not found

Mounting succesful

Copying files:

config.json

rpi\_service.py

rpi\_service.service

rpi\_sh.sh

sensor\_terminal.py

serialsensor.py

Setting permissions...

Adding connection settings to config file.

Unmounting image.

Image preparation finished, press Enter to continue...

The error on line 2 is simply because the utility tries to unmount currently mounted images, you may simply ignore it.

If all information is satisfactory, you may press enter to burn the newly created custom image to the SD Card.

Unmounting disk partitions (errors are expected).

umount2: Invalid argument 🡨 (error mentioned above)

umount: /dev/sdb: not mounted

Burning image to SD Card at /dev/sdb

Image burning may take as long as 15 minutes, please wait.

The burning process may take as long as 15 minutes (If using a virtual machine, this process may take even longer).

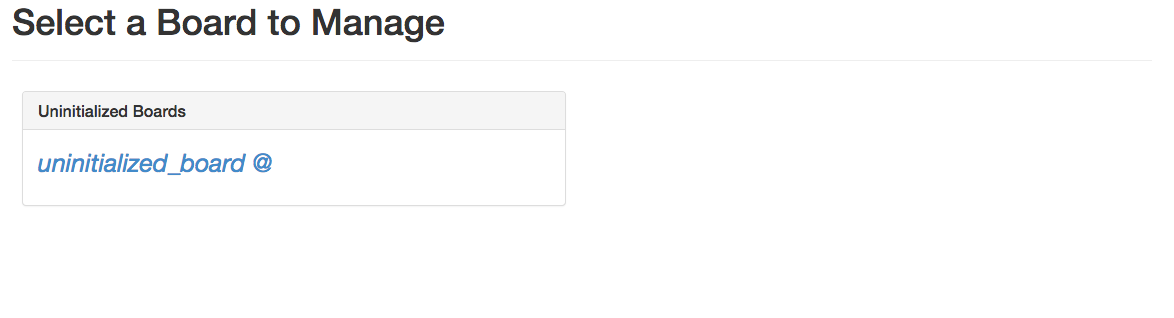
Once the process is complete, the disk will be automatically ejected, and you are free to remove it.

1. Setting up Raspberry Pi, and configuring through the web interface.

3.1 Plug in the SD Card and connect an Ethernet cable and power to the Raspberry Pi.

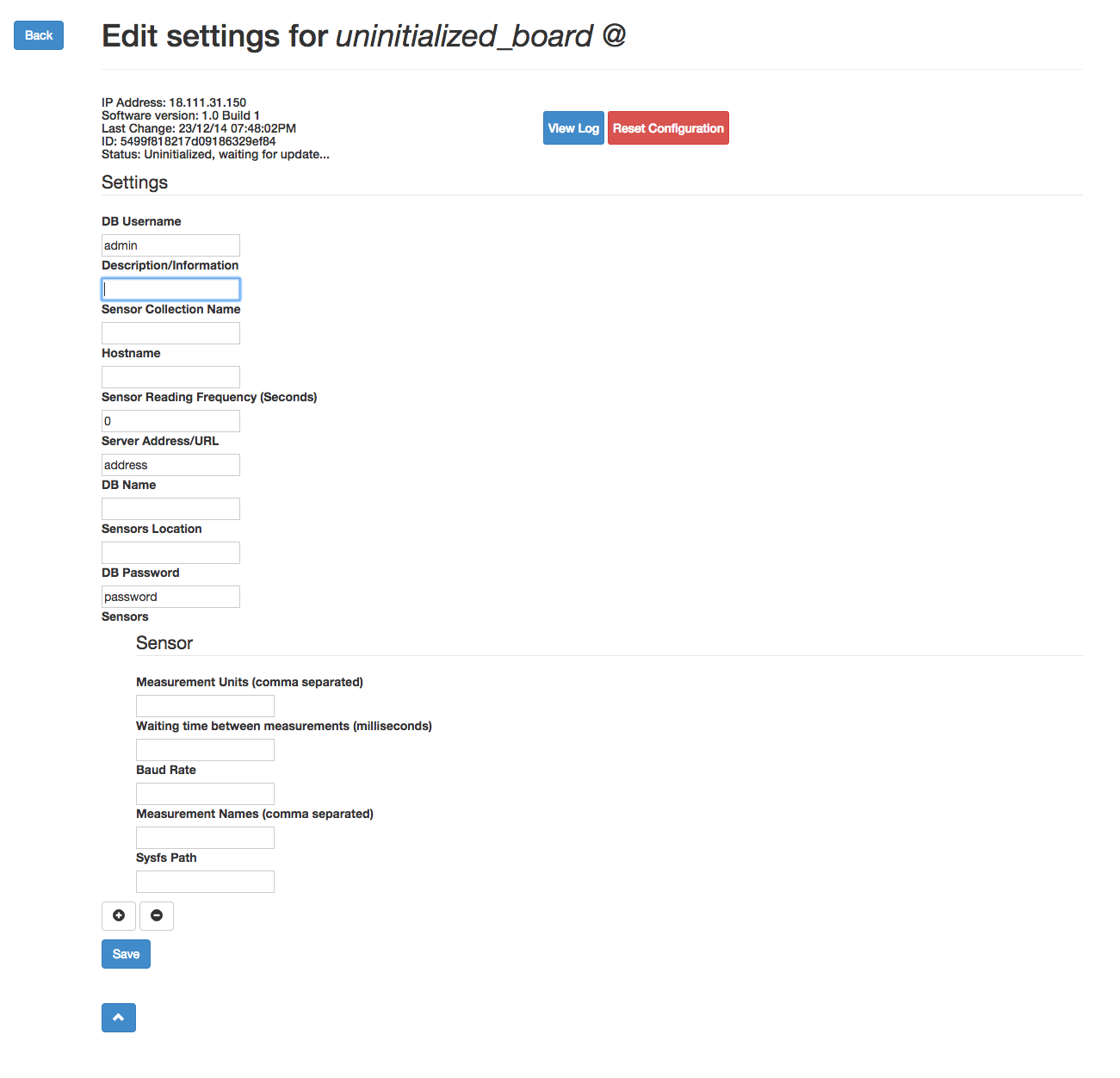
The Raspberry Pi will boot, and the *“rpi\_service.py”* will be executed automatically, a new empty configuration file will be added to the database (at the admin database, boards collection) and the configuration may now be edited through the web interface.

3.2 Go to the web interface’s address, and the following page should appear:



It is recommended that one board be added to the system at a time, so as the user may know which board is being deployed.

Select the uninitialized board, and the settings page for that board will appear:



In this page you can edit the boards settings and add/remove/edit sensor settings. The settings available for editing are:

* DB Username[4]
* DB Password[4]
* Description
* DB Name[4]
* Collection Name
* Hostname[5]
* Sensors Location
* Server Address[4]
* Sensor Reading Frequency[6]

Sensor settings:

* Measurement Units[6]
* Measurement Names[6]
* Baud Rate
* Waiting Time
* Sysfs path[7]

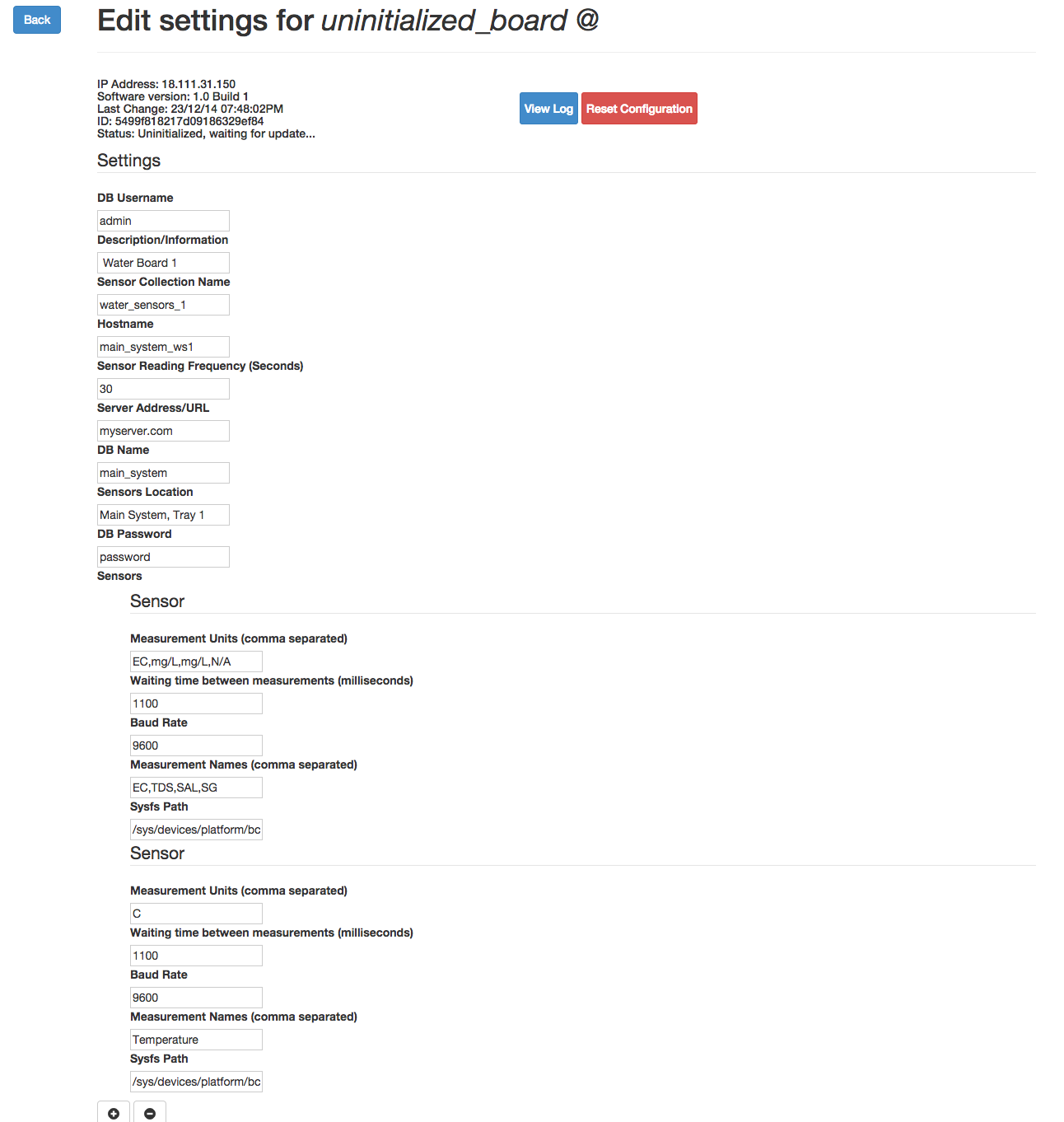
The status bar on the top of the page provides information about the board being managed, such as:

* IP Address (which may be used for ssh)
* Software Version
* Last Settings Change
* ID (given by the configuration entry in the database)
* Status[8]

This page also provides access to the current log (*“View Log”*) and a reset button to erase all configurations from the board (except server, username and password)

An example setup is provided at the end of this document[9]

Once saved, it may take a few seconds for the board to reload the changes. Once that happens, if the page is refreshed, *“Initialized”* should appear in the status report. And, all provided information is correct, the board should start reporting data. The log can always be checked to debug any problems that may arise.

 [9] Example initial configuration of board.