instructables

Terrarium Controller

by kevin_scheyltjens

In this instructable I will explain how to make your own Terrarium controller.

We will be doing several steps in this instructable such as;

- Wiring an electrical circuit which holds certain components as:

heat lamps, a sensor and fans.

- creating a mysql database on the Raspberry pi

- write data to the database

- reading data from the database

- creating a flask webserver to show the data

-build a case to contain the electrical circuit

The terrarium controller will be controlled mainly by the rasberry pi. Keep in mind that this build is for a small terrarium however by changing a few things you could use this for larger builds.

This build will have a price tag of about €90.

Step 1: Components

The components are;

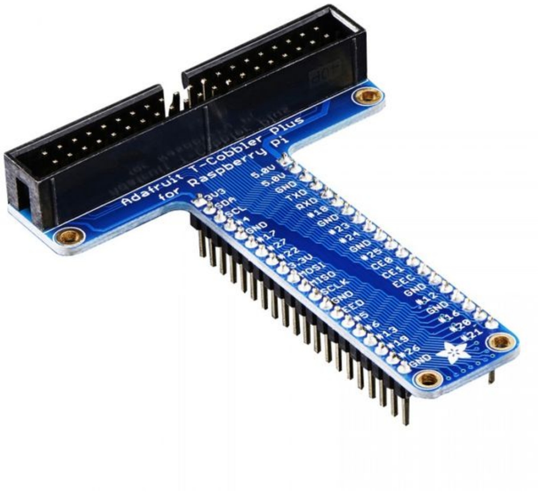
- **Raspberry PI 3 B**
- **T-cobbler plus**
- **A DHT22 sensor**
- **2x 5v blower fans**
- **A relay module (either 2 channel or 4 channel)**
- **2x heatlamps (one for day, one for night)**
- **2x light fixtures**
- **a power plug and cable**
- **2x transistor (BC517)**
- **2x diode**
- **a LCD (HD44780)**
- **40x jumper wires**
- **a number of small screws**
- **resistors (10k, 1k)**
- **breadboard (small one is okay.)**
- **heat shrink sleaves**

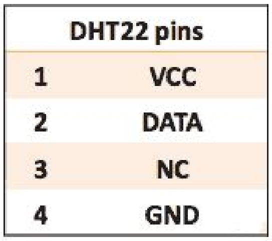
we will also be using a few tools such as;

- **a soldering iron**
- **screwdrivers (flat head or cross head)**
- **a wire stripper**
- **a side cutter**

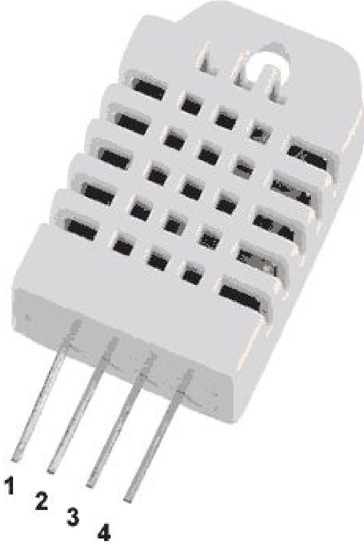
These components are all you need to create this build, the tools will be used during the wiring of the circuit.
You can just connect the pins with a breadboard or directly onto the pi or pi T cobbler, however this is not recommended. Because the jumper wires may come loose and fall off.

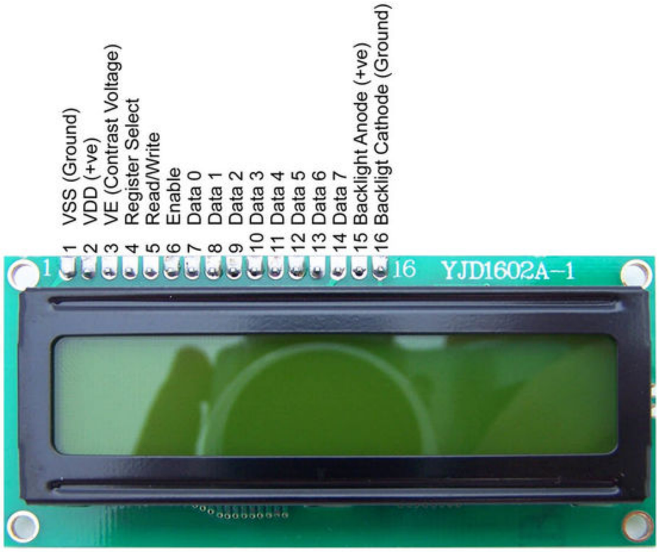
So it's highly recommended to solder some of the more delicate things such as the DHT22 sensor.

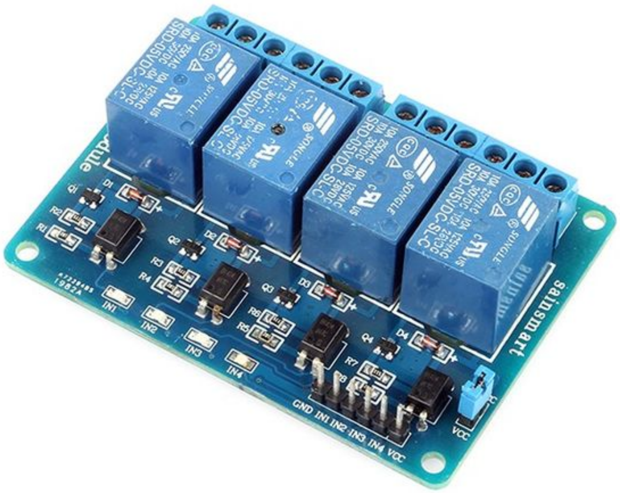




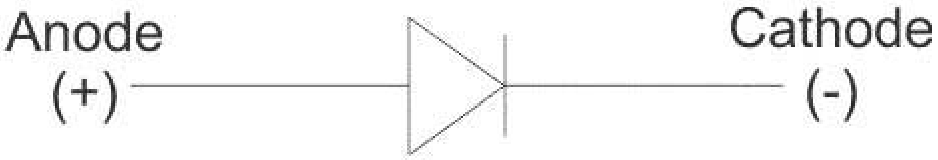
DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

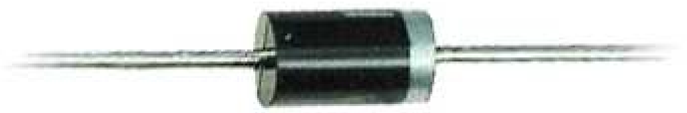


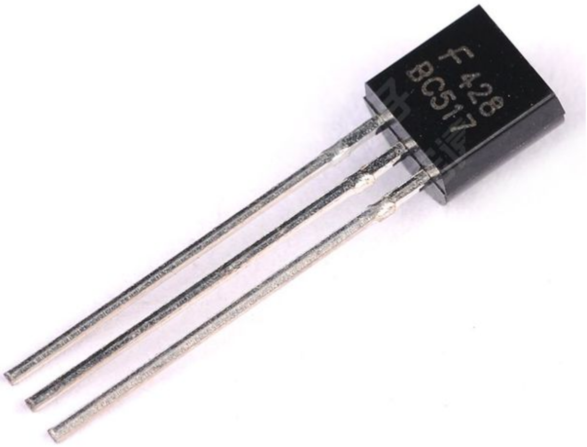


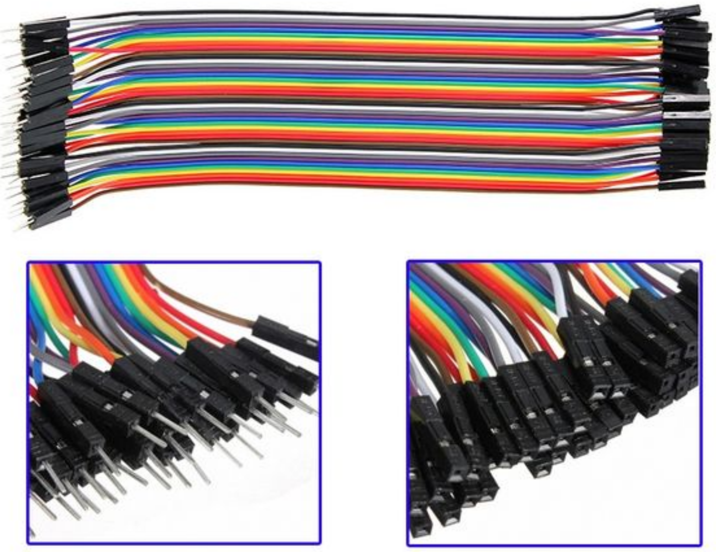


75W
NIGHT
HEAT LAMP


Anode
(+)
Cathode
(-)




F 428
BC517



POWERPLUS
60W

fixPOINT®
250g
Ø 1,0 mm
Pb 38
Sn 60
Cu 2
Qualitäts-Lötzinn
Super solder wire

haupa
www.haupa.com

Step 2: Schematic and Breadboard Circuit

With above provided schematics it is possible to create the entire electrical circuit.
However, some extra explanation might be usefull into understanding how some of these components work.

The T-cobbler plus will be attached to the pins on the pi. (be carefull to look how you place it, since placing the cable wrong will fry the pins.) and then placed into a breadboard.

DHT22 Sensor;

In this build we use a DHT22 sensor, you could use a DHT11 sensor. However I find that it is better to use the 22 variant, because it is **more accurate** and has a **greater range** of measurement.

This sensor has 4 pins and requires one to wire a pull-up resistor to it. The **pull-up resistor** makes sure that the GPIO pin that reads from the DATA pin does not make any mistakes due to **floating current**. We will use a **10k ohm resistor** to connect the VCC(5v) to the DATA pin.

- **VCC** to **5V**
- **DATA** to **GPIO 12**
- **Null** to nothing
- **GND** to a **GND** pin of the pi (**ground**)
- **10K ohm resistor** From **VCC** to **DATA**

LCD (HD44780);

The LCD screen will be used to print the temperature and humidity that is read from the DHT22 sensor. This data will be sent from the sensor to the LCD class which controlls the LCD.

This is not hard to wire, but there are 16 pins. It is **important** to **wire** these pins **correctly** because you can easily short the LCD module.

- **VSS** to **GND**
- **VDD** to **5V**
- **V0** to **GND** with **1kohm resistor**
- **RS** to **GPIO 17**
- **RW** to **GPIO 27**
- **E** to **GPIO 22**
- **D0** to **GPIO 18**
- **D1** to **GPIO 23**
- **D2** to **GPIO 24**
- **D3** to **GPIO 25**
- **D4** to **GPIO 5**
- **D5** to **GPIO 6**
- **D6** to **GPIO 13**
- **D7** to **GPIO 19**
- **Anode** to **5V**
- **Kathode** to **GND**

Fans;

To wire the fans we will need to wire a transistor and a diode per fan. The transistor will allow us to controll when the fan goes on or off. With the diode we protect our components from shorting and getting rid of any risidual currents by creating a loop.

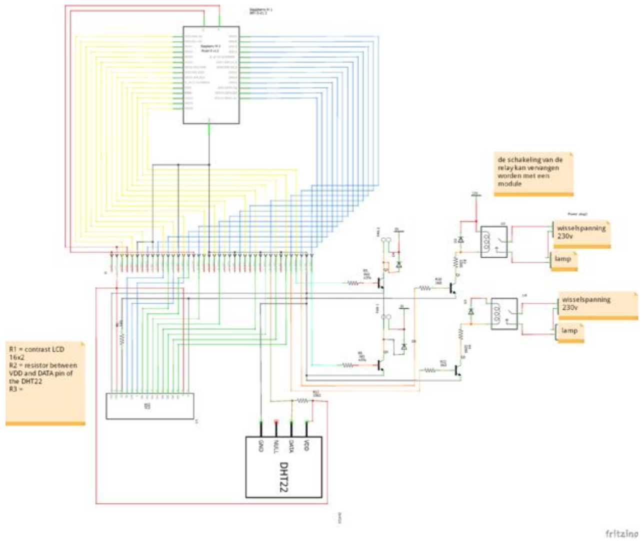
- **GND** to **Collector** of transistor
- **Power** to **5V** and **walled** side of **diode** (the white line on the diode)
- **Base** of transistor to **GPIO 26** with **1K resistor** (for the **second fan** we wire to **GPIO 21**)
- **Emmiter** of transistor to **GND**

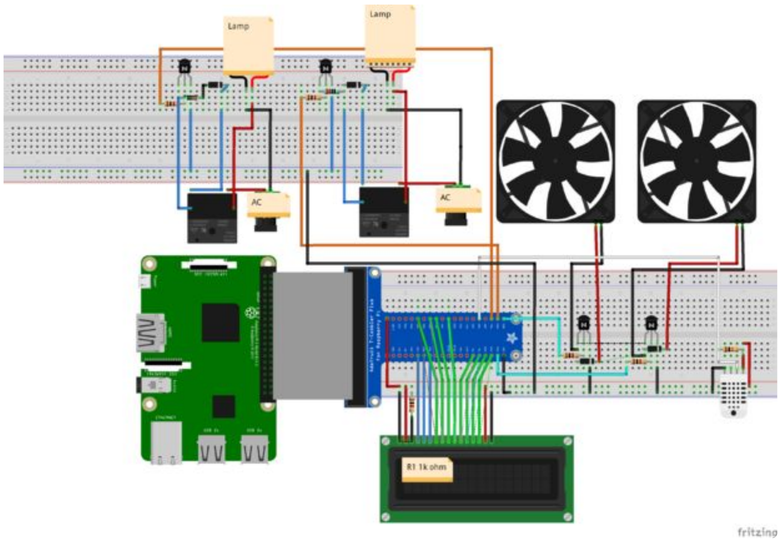
Relay module;

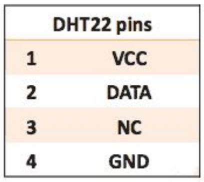
The wiring of the relay module may be one of the simpler wirings. Because we only need to provide a 5V power supply, a GND and then wire a closed circuit to the relay module. In the provided schematic it is wired using seperate transistors and diodes, but because we use the module we won't be using all of that circuit.

- **5V** to **5V**
- **GND** to **GND**
- **IN1** to **GPIO 16**

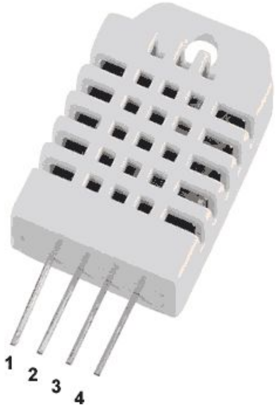
- **IN2** to **GPIO 20**
- **Left port** to **power plug + side**
- **Middle port** to **Light fixture**
- **Light fixture** to **power plug - side**

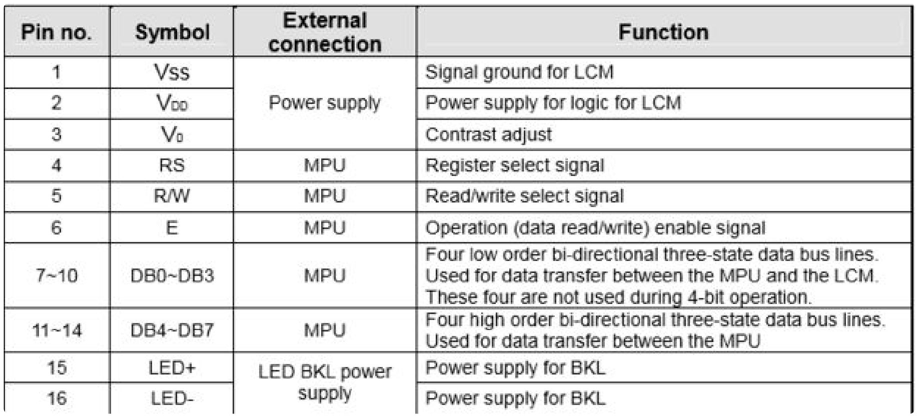

de schakeling van de relay kan vervangen worden met een module
wisselspanning 230v
lamp
wisselspanning 230v
lamp
R1 = contrast LCD 16x2
R2 = resistor between VDD and DATA pin of the DHT22
R3 =
GND
NULL
DATA
VDD
DHT22
fritzing





DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND





Pin no.	Symbol	External connection	Function
1	Vss	Power supply	Signal ground for LCM
2	V_{DD}		Power supply for logic for LCM
3	V_0		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU
15	LED+	LED BKL power supply	Power supply for BKL
16	LED-		Power supply for BKL

Step 3: The Case

I have made a small wooden box in which I place my breadboard on which i have wired my fans. I have screwed the T-cobbler to the side of the box, so that i can easily put in any wire on the inside. And still connect it to the pi which I put on the outside front of the box.

Then I screwed the relay module to the botom of the box with the cable inputs (for the light fixture and power plug) to the back. I drilled a hole at the height of the inputs and gave it the same width so that i can easily put the cables through the hole.

To get the fans and sensor out of the box (considering we would place the fans and the sensor within the terrarium itself) I drilled a hole in the botom, this is with the idea that we put the controller box on top of the terrarium lid. As such we can just let the sensor into the terrarium via another hole in the lid.

To get a good view of the LCD which will show our temperature and humidity we will cut out a hole of the top of the box at the size of the screen of the LCD. Mind that there are some pins from the solder on the pins on the front of the LCD and there is a small outcropping on the front left side of the LCD. So we might need to make a few additional small insets.

To have an easy access to the contents of the box you may want to put a hinge on either the front of the box or back and connect it to the lid of the box. however if you do that, make sure that the jumper wires you used to wire the LCD are long enough. Or you might not even be able to open your box.

Step 4: Setting Up the Raspberry for Mysql

At this point we will start working with the raspberry pi. If you have worked with the pi before you will have already installed the os and may have fooled around with it, then we can just continue.

However for those new to the raspberry pi, we will have to start by installing an os to the pi. We do this by downloading an os and putting it onto an SD card. The **os** we will be using is called "**Raspian-jessie-with-pixel**". When you download the os you will get a file or Image, with which you can't do much, to use it we have to use a handy program called; "**Win32DiskImager**".

With this done look for the "**cmdline.txt**" file in the SD card and add "**ip=169.254.10.1**"

Then just put the SD card into the pi and connect it with an ethernet cable to the pc. By using a program called "**Putty**" and the IP you have previously inputted into the cmdline file you can connect to you pi.
The standard user for logging in on the pi will be;

```
- user : pi
```

```
- password : raspberry
```

after you have logged in, the best thing you can do is update your files and upgrade any package on your pi using following cmd lines.

```
- sudo apt-get update && sudo apt-get upgrade
```

Now we can start with installing mysql on the pi and creating a database.

Before we do anything we must install mysql onto the pi, we will use cmd lines again;

```
- sudo apt-get install mysql-server
```

```
- sudo apt-get install mysql-client
```

You will encounter some requests for permission, you can just accept them by pressing "y" and then enter.

next we will activate mysql on the pi :

```
- mysql -uroot -hlocalhost -p
```

After you have entered this you will be asked to enter a password, this password is the one you have entered during the install of the mysql server / client. if you have succesfully done all this you should see "mysql>" on the left instead of the name of your pi.

Finnaly we will create a user and a database.
while you have entered mysql by using above cmd line, enter; (caps is important here)

```
CREATE DATABASE database_input;
 CREATE USER 'Name'@'localhost' IDENTIFIED BY 'password';
 GRANT ALL PRIVILEGES ON database_input.* TO 'Name'@'localhost';
 FLUSH PRIVILEGES;
```

(replace name and password with a something of your choice)

You can quit the mysql space by pressing **CTRL+C**

Now there are a few different ways to get tables into your database:

1. **importing through pycharm**
2. **creating tables in pycharm**
3. **creating tables in pi**
4. **importing through filezilla**

I have created the tables through pycharm and will explain for that.

In pycharm;

Go to **VIEW** --> **TOOL WINDOWS** --> **DATABASE**
After opening the tool window for database press the **green +** in the top left of the tool window,
then press datasources and finally on mysql.

in the next window fill in the **name** of you database, the **user name** you have created and the **password** of this user.

then go to the **SSH/SSL tab**. Select **USE SSH TUNNEL** and input the info from your pi.

Finally we have to get a connection between our mysql database on the pi and the pycharm program which we use. you simply use the following cmd line in pi

```
sudo apt-get install python3-mysql.connector
```

Step 5: Setting Up Flask on Your Pi

Again we will connect to the pi through "putty", after we log in we will use the cmd line;

```
- sudo apt-get install python3-flask
```

Then you could either start writing code for a flask project using pycharm or just use the code I have written and change a few things which do not apply to you.

You can get the code from github , in the github repository you will find two directory's. One called **flask_site** and one called **terrariumRegelaar**. You can download both of them, though we will only use the flask_site in this step.

When you have downloaded the flask_site directory you will have to edit a bit of code in the dbclass.py file. you will have to open this file through **pycharm** so you better install it, if you haven't before.
The things you have to change are the name of the database(db), the username and the password. you can keep the host as localhost since you will be using it via the pi localhost. (picture 1)

To use this code you will have to open pycharm once again and then **create a new flask project**. It's important that it is a flask project. Else you will not be using a flask webserver when running the code and you will not have the correct librarys.

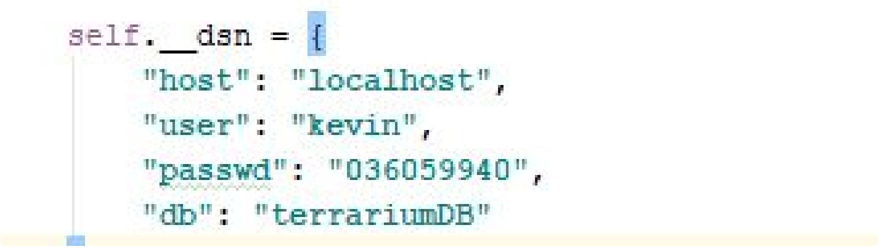
After you have done that just put the files inside the flask_site directory inside of your flask project and it should work. Next we will need to upload the files to the pi.

First you will need to add a deployement to your pycharm settings. (picture 2), then go to the interpreter (picture3) and add a new remote interpreter.

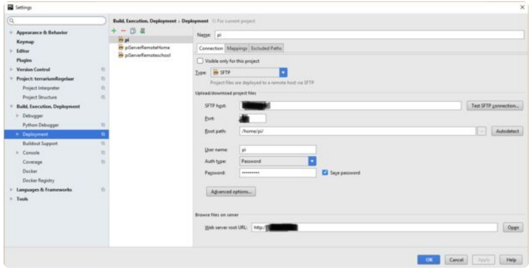
After that just use the deployment config, however be mindfull that where it says which python to use you should state that it is python3 and not just python. (picture 4)

when we have added the interpreter there are a few things we have to add to the interpreter.

click the green + in the interpreter screen (on the right side) and search for pigpio and mysql.connector. We will need the first for our DHT22 sensor and the second to connect our pi database to our pycharm.



```
self.__dsn = {
    "host": "localhost",
    "user": "kevin",
    "passwd": "036059940",
    "db": "terrariumDB"
```



Name: pi

☐ Visible only for this project

Type: SFTP

Project files are deployed to a remote host via SFTP

Upload/download project files

SFTP host: [redacted]

Port: [redacted]

Root path: /home/pi/

User name: pi

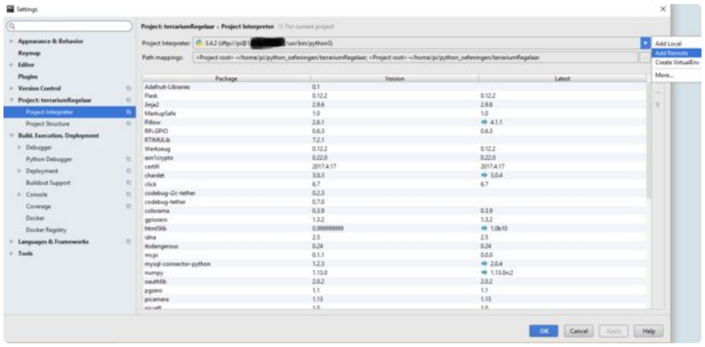
Auth type: Password

Password: ••••••••••

☑ Save password

Browse files on server

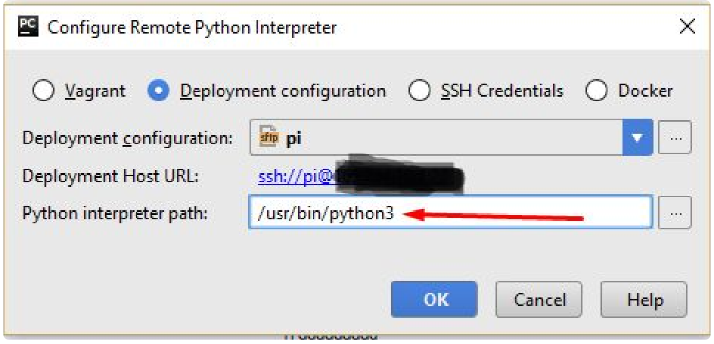
Web server root URL: http://[redacted]



Project Interpreter: 3.4.2 (sftp://pi@1[redacted]/usr/bin/python3)

Path mappings: <Project root>→/home/pi/python_oefeningen/terrariumRegelaar; <Project root>→/home/pi/python_oefeningen/terrariumRegelaar

Package	Version	Latest
Adafruit-Libraries	0.1	
Flask	0.12.2	0.12.2
Jinja2	2.9.6	2.9.6
MarkupSafe	1.0	1.0
Pillow	2.6.1	➡ 4.1.1
RPi.GPIO	0.6.3	0.6.3
RTIMULib	7.2.1	
Werkzeug	0.12.2	0.12.2
asn1crypto	0.22.0	0.22.0
certifi	2017.4.17	2017.4.17
chardet	3.0.3	➡ 3.0.4
click	6.7	6.7
codebug-i2c-tether	0.2.3	
codebug-tether	0.7.0	
colorama	0.3.9	0.3.9
gpiozero	1.3.2	1.3.2
html5lib	0.999999999	➡ 1.0b10
idna	2.5	2.5
itsdangerous	0.24	0.24
mcpi	0.1.1	0.0.0
mysql-connector-python	1.2.3	➡ 2.0.4
numpy	1.13.0	➡ 1.13.0rc2
oauthlib	2.0.2	2.0.2
pgzero	1.1	1.1
picamera	1.13	1.13
picraft	1.0	1.0



○ Vagrant ● Deployment configuration ○ SSH Credentials ○ Docker

Deployment configuration: pi

Deployment Host URL: ssh://pi@[redacted]

Python interpreter path: /usr/bin/python3

Step 6: Adding Your Code to Pi and Autorunning Your Files.

When you have setup your flask project it's time to get the code fixed for the actually components.

As I have mentioned in the step above, there where two file in the github, now we will use the directory "terrariumRegelaar".
We will need to make a new project in pycharm again, however instead of chosing a flask project, we will use a normal python project. Again just drag the files from the "terrariumRegelaar" directory into your new project. Setup the deployment and interpreter up like in the previous step. then just upload the code by right clicking you project name in the left hand side of pycharm and select upload to pi.

It has come to the point that you have made your electrical circuit, you have built your controller box and added all the necessary code to your pi. Now we need to autorun our files on the pi.

We do this by connecting to our pi, via putty and entering;

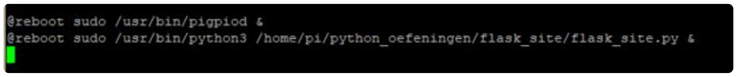
```
sudo crontab -e
```

at the end of this file we will add the lines from the picture added to this step.

if we wish to autorun our code for our controller itself we just add it by writing

```
@reboot sudo /usr/bin/python3 /home/pi/"file location"/"filename"
```

Then just reboot your pi and it should run by itself.



```
@reboot sudo /usr/bin/pigpiod &
@reboot sudo /usr/bin/python3 /home/pi/python_oefeningen/flask_site/flask_site.py &
```

Step 7: Thank You!

First of all thank you for taking the time to read through the entirety of this instructable. I am sure there are allot of mistakes in my sentences, in my spelling and in my explanations. So thank you for going through it all.

A bit about myself.

I am a student at Howest and I follow the major NMCT or New media and Communication Technology.
This instructable is part of a project for my finals of my first year and it will contain some or many mistakes in either coding, explanations or just mannerism. However I hope you enjoy reading and/or using this instructable to create something you can use.

This project is not finished by any means and is only a prototype. You can use any code I have written (if it is usefull) and tweak it or rewrite it.