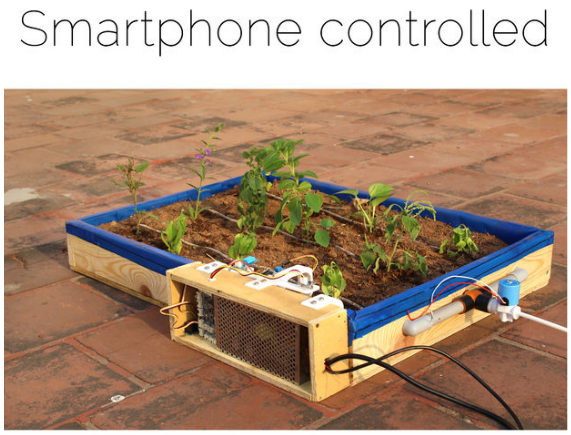
Raspberry Pi Powered IOT Garden

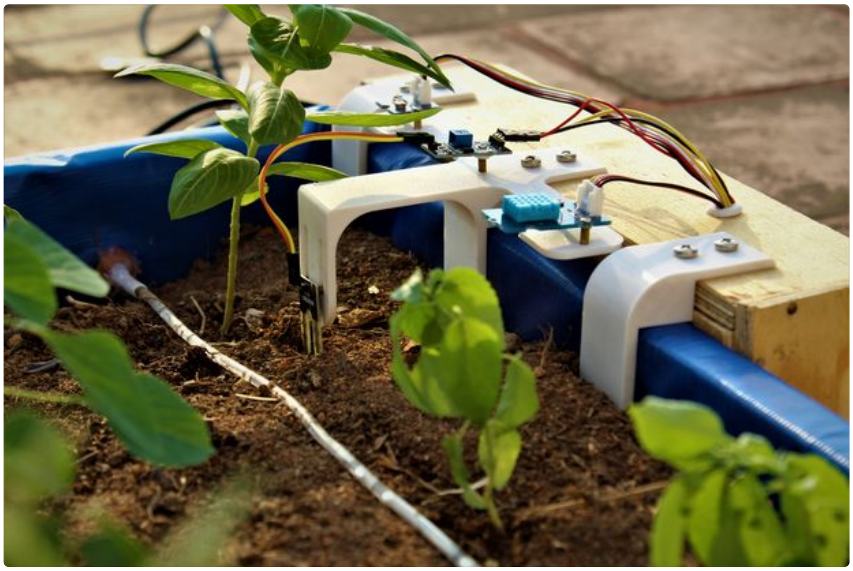
by Technovation

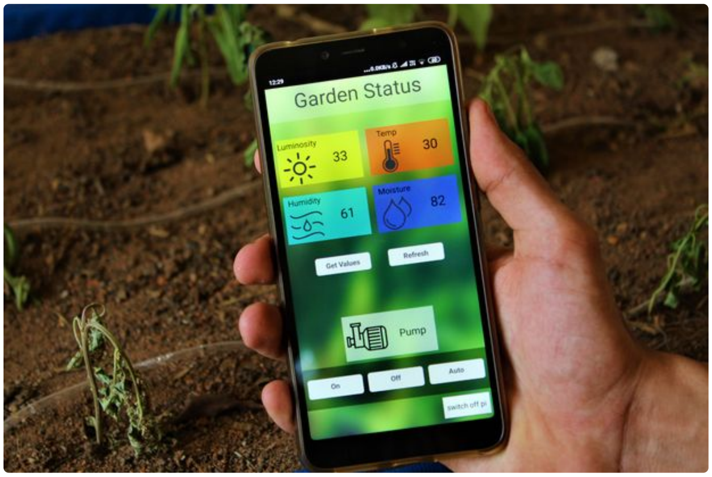
One of the primary objectives of this project was to be able to maintain the well-being of a garden using the power of the Internet of Things (IoT). With the versatility of the present tools and software, our planter is integrated with sensors that monitor the real-time status of the plants. We built a smartphone app that let's one access the data and take needed actions if necessary.

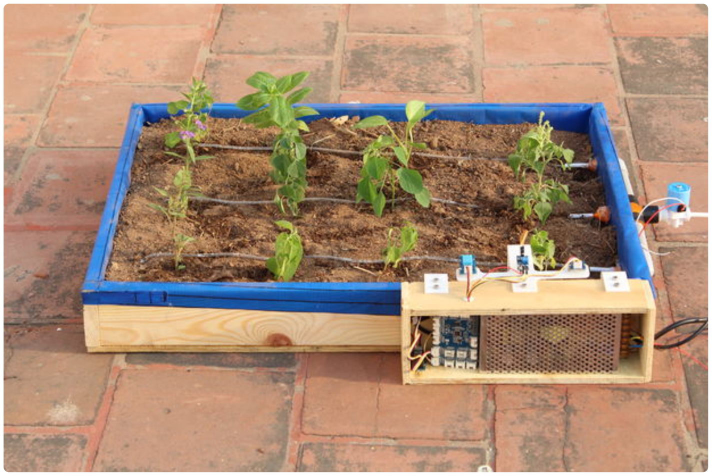
The design of our planter is scalable, low-cost and easy to build, making it the perfect option to add greenery to one's terrace or backyard. The smart garden has proven to be more efficient in water consumption and facilitates maintenance and monitoring.

Follow on to learn how to make your very own database and app, by creating a garden that can be monitored by a click of a button!









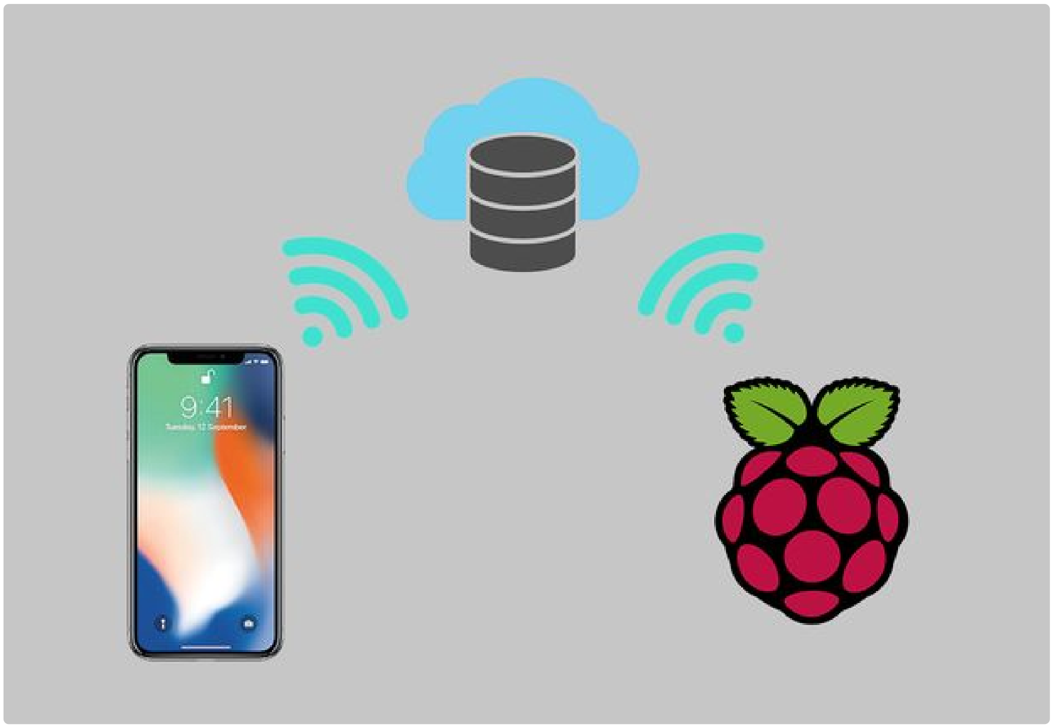
Step 1: Overview of the IOT System

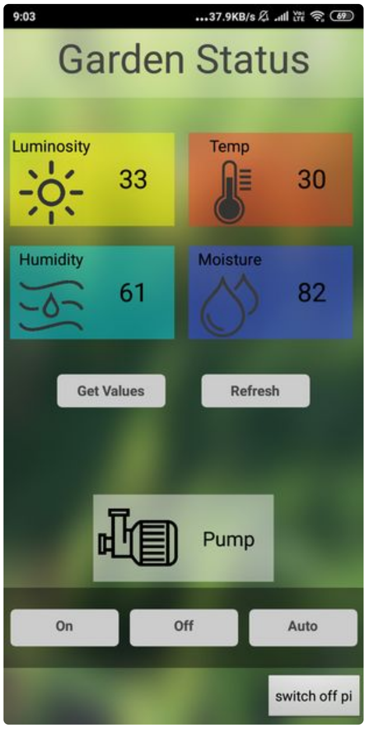
The Iot system functions through the following processes. A Raspberry Pi is used to relay useful information of the garden, such as luminosity, humidity and the moisture content in the soil from various sensors into a cloud database. Once the information is in the cloud, it can be accessed from anywhere using a smartphone app that we built. This process is reversible too, the user can send instructions, such as the state of the water pump, back to the garden which will execute the required commands.

The following are some of the key features of our garden :

- Real-time feedback of the garden's various sensors
- Database of the garden's health status
- Global monitoring and operating capacities
- Drip irrigation system
- App controlled water system
- Automatic watering schedules

We decided to use Google's Firebase as the intermediary of our IOT system, to create our own free cloud database. We then used MIT's App Inventor to create a smartphone application which is compatible with the Firebase database and the Raspberry Pi. It can also communicate with the database with the help of a free Python library.





Garden Status

Luminosity 33

Temp 30

Humidity 61

Moisture 82

Step 2: Materials Needed:

The materials needed to make the iot planter can be easily found in local or online shops. The following list is a description of all the parts needed.

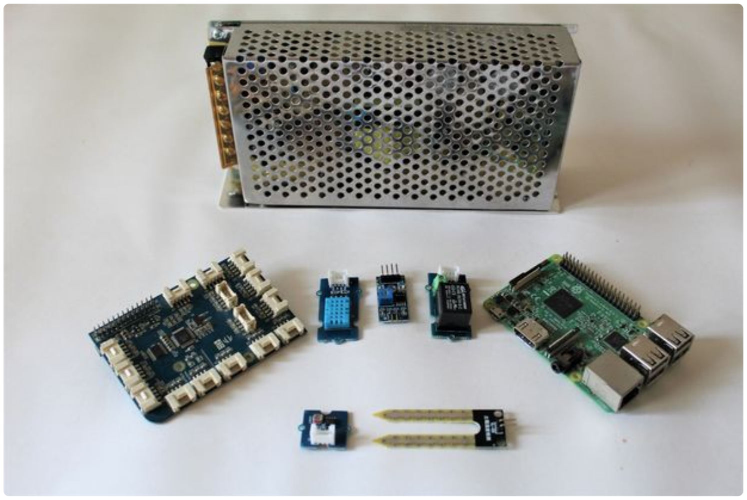
HARDWARE :

- **1" Pine Wood Plank** - dimensions; 300cm x 10cm (as the wood will be outdoor, we would recommend treated wood)
- **1/4" Plywood** - dimensions; 120cm by 80cm
- **Tarpaulin Sheet** - dimensions; 180cm x 275cm
- **PVC Pipe** - dimensions; length 30cm, Dia 2cm
- **Surgical Tube** - dimensions; 250cm
- **Elbow Joint** x 2
- **Wood Screw** x 30

ELECTRONICS :

- **Rasberry Pi3 Model B**
- **Grove Pi + Sensor Shield**
- **12V Solenoid Valve**
- **Humidity and Temperature Sensor (dht11)**
- **Moisture Sensor**
- **Luminosity Sensor**
- **Relay Module**
- **12V Power Supply**

The total cost of this project is roughly *50 USD*

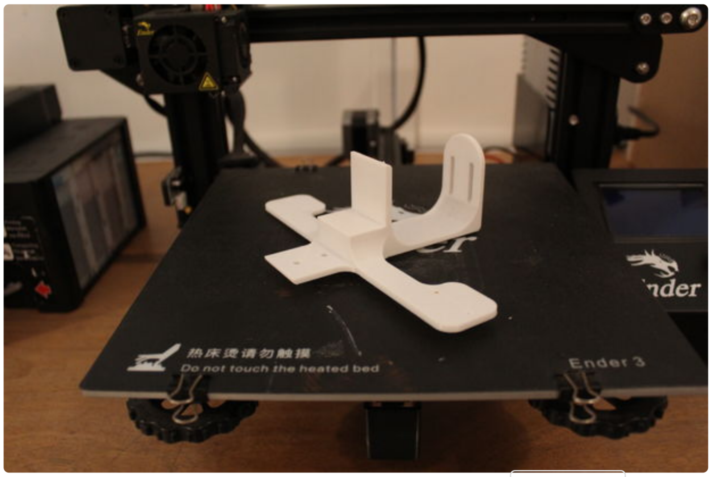


Step 3: 3d Printed Parts

Various components that needed to be customised for this project were made with the help of 3d printing. The following list contains the complete list of parts and their printing specifications. All the STL files are provided in a folder attached above, allowing one to make their needed modifications if necessary.

- **Pipe Joint** x 1, 30% infill
- **Nozzle Adaptor** x 3, 30% infill
- **Tube Plug** x 3, 10% infill
- **Hook** x 2, 30% infill
- **Sensor Mount** x 1, 20% infill
- **Valve Adaptor** x 1, 20% infill
- **Wiring Cover** x 1, 20% infill

We used our Creality Ender 3 to print the parts, which took around 8 hours for the 12 parts.

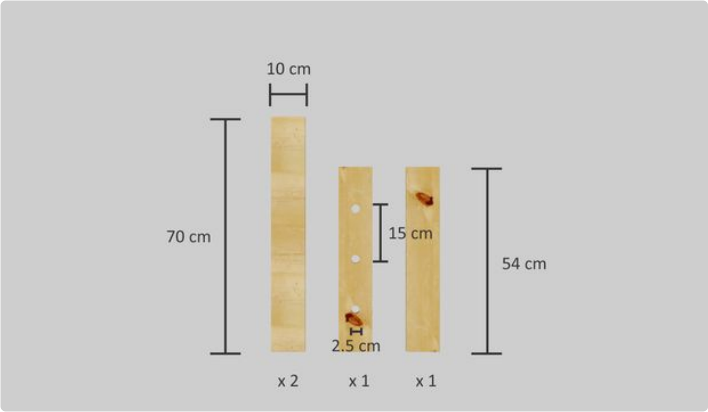


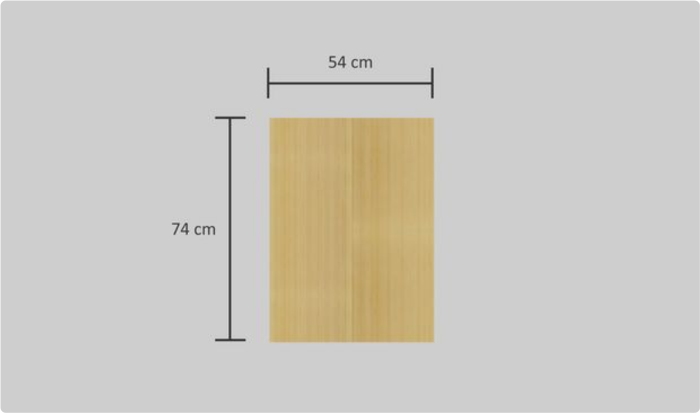
https://www.instructabl...

Download

Step 4: The Plans

One is not restricted to the dimensions that we chose to make our planter, but attached above are all the details required to make the project. In the following steps one can refer do these images to cut the wood.

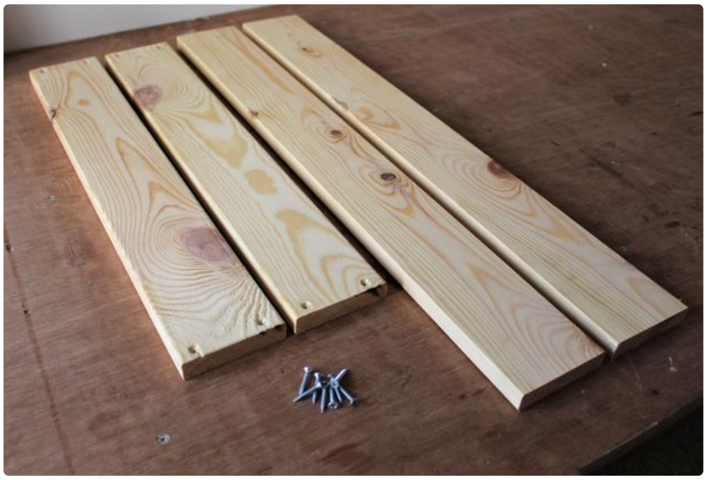




Step 5: Building the Sides

To hold the plants we decided to make a planter structure out of wood. The inner dimensions of our box is 70cm by 50cm with a height of 10cm. We used pine wood planks to build the sides.

Using a circular saw we cut the four pieces to length (dimensions attached above). We drilled pilot holes at the marked spots and countersunk the holes so that the screw heads sat flush. Once done, we drove in 8 wood screws while making sure the sides were square which secured the frame.



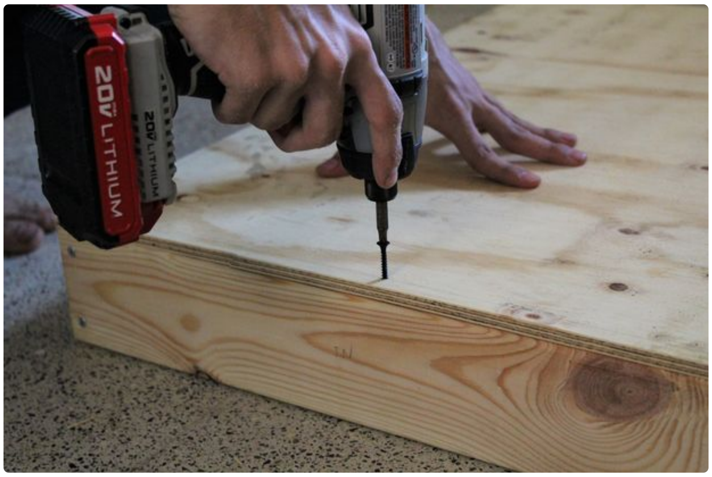
19.71
mm
in
OFF
ON
ORIGIN
ABS/0

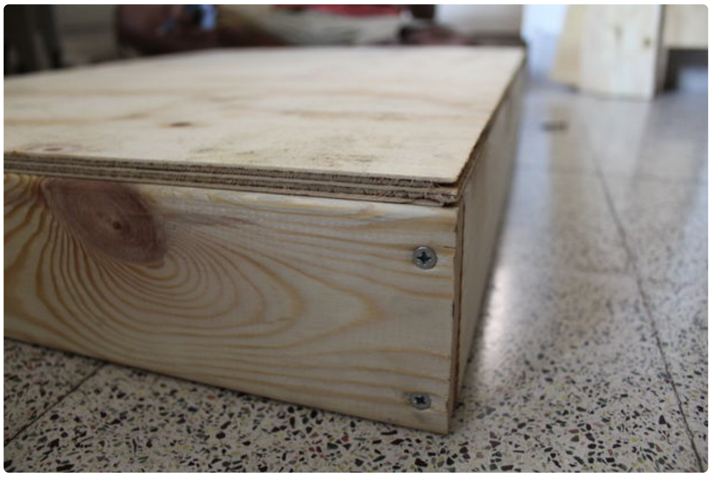
Coca-Cola
PORTER CABLE
20V
LITHIUM
4.0Ah

LITHIUM
PORTER CABLE

Step 6: Fitting the Bottom Panel

To make the bottom panel we cut a rectangular piece of 5mm plywood, which we then screwed in place to the side frame. Make sure the holes are countersunk so that the screws are flush with the base. The needed dimensions can be found attached above.

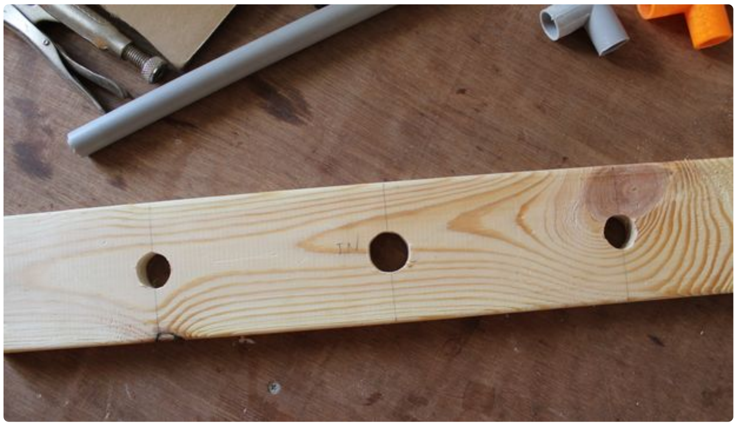




Step 7: Holes for the Pipe

Our planter is made to accommodate three rows of plants. Therefore for the drip irrigation system one side needs to hold the pipes for the water input.

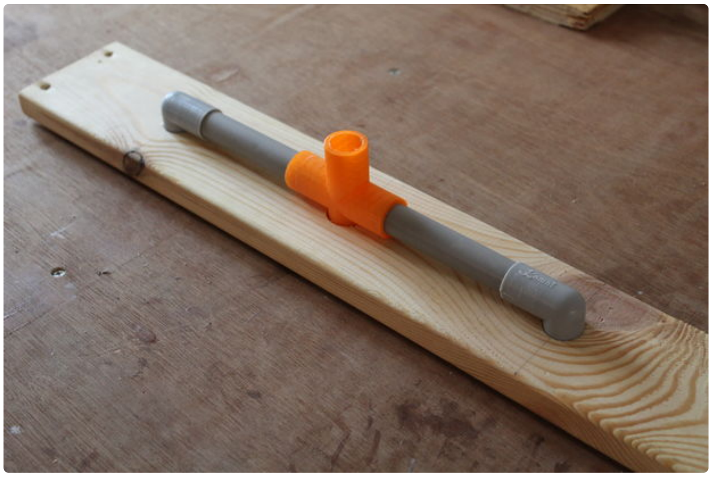
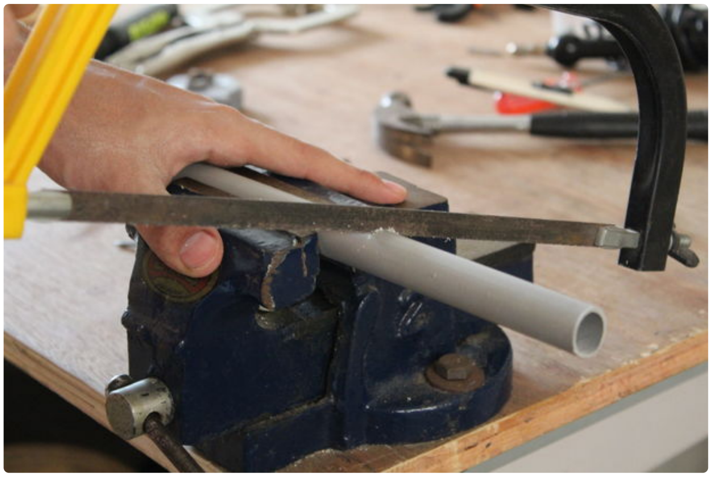
Start by measuring the diameters of the connectors and draw them out equidistantly on the shorter side of the frame. As we didn't have a forstner bit, we drilled a 10mm hole and then widened it with a jigsaw. To smooth out the rough edges one can use a Dremel till the connectors fit.

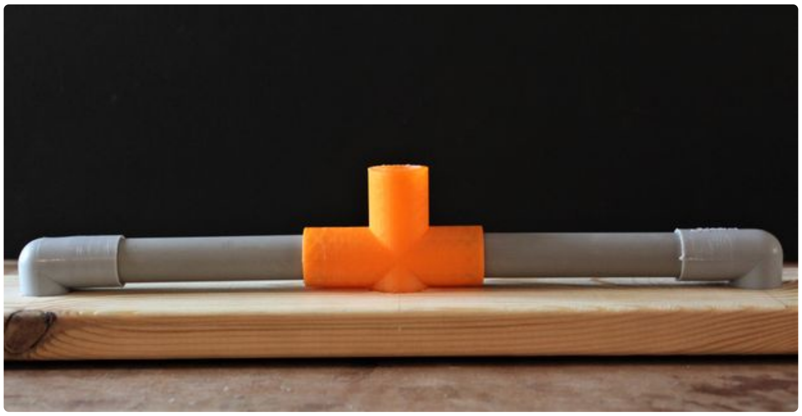


Step 8: Connecting the Water Pipes

To connect the joints simply cut two pieces of PVC pipe 12 cm long. Dry fit the set up to check if everything fits snuggly.

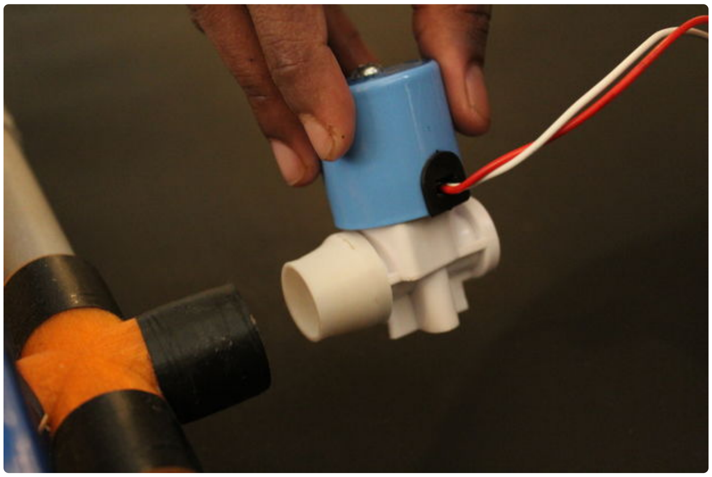
Then push in the 3d printed joint in the central hole and the two PVC elbow connectors on the opposing ends till they are flush. Attach the panel back to the frame and cap the connectors from the inside with the 3d printed adaptors. All connections are friction fit and should be watertight, if not, one could seal the joints with hot glue or Teflon tape

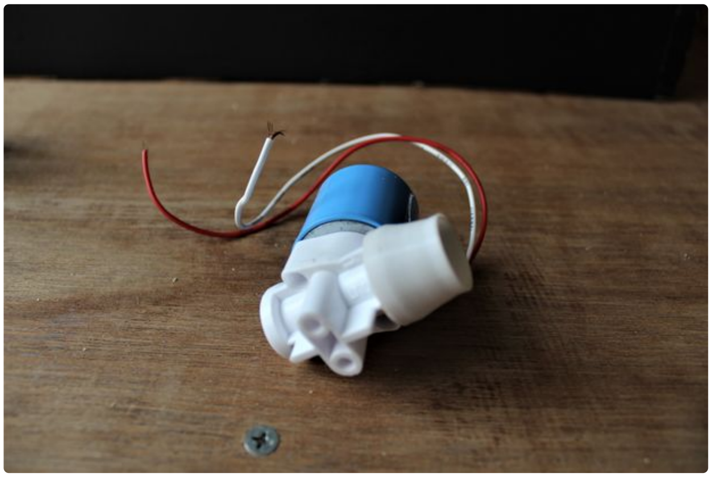





Step 9: Solenoid Valve

To control the flow of water into the drip irrigation system we used a solenoid valve. The valve acts as a gate that opens when an electrical signal is sent making it controllable automatically. To incorporate it, we attached one end to the water source and the other to the planter's water input pipe using an intermediary adaptor. It is important to connect the valve in the right orientation generally tagged as "IN" for the water input (a tap) and "OUT" for the water output (the planter).



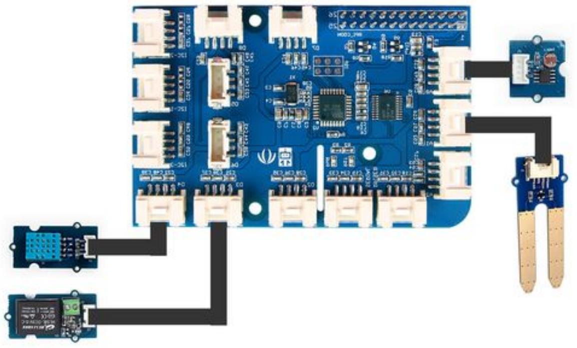


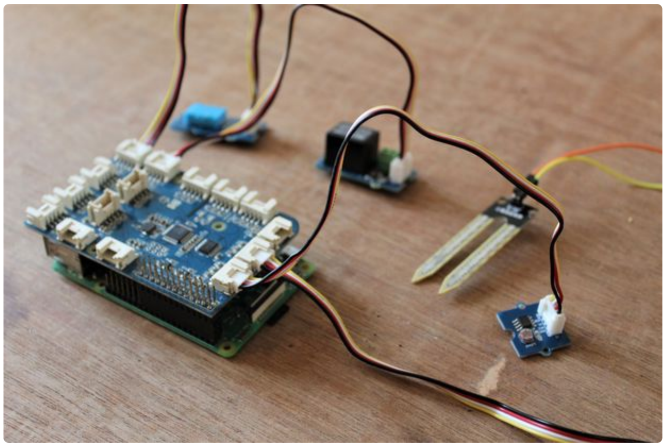
Step 10: Wiring the Electronics

Below is a table with the various modules and sensors with their respective ports on the grovepi+ shield.

- **Temperature and Humidity Sensor** ==> **port D4**
- **Relay Module** ==> **port D3**
- **Moisture Sensor** ==> **port A1**
- **Light Sensor** ==> **port A0**

Use the wiring diagram attached above as a reference.

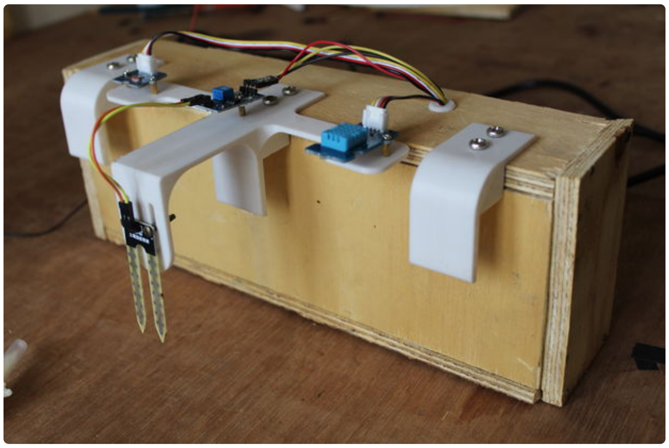


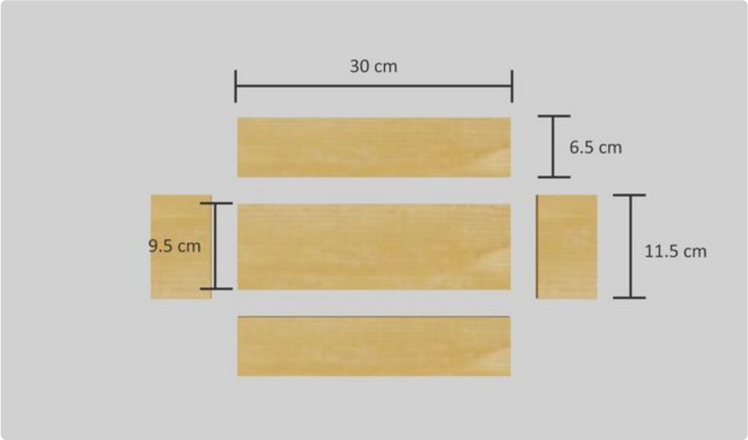


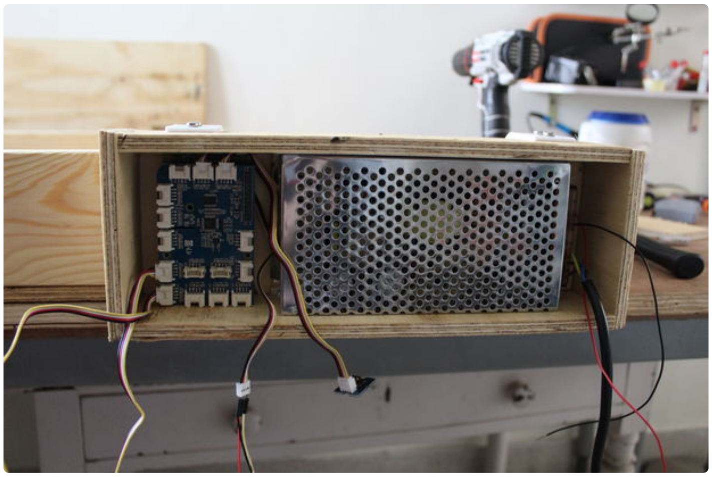
Step 11: Sensor Compartment

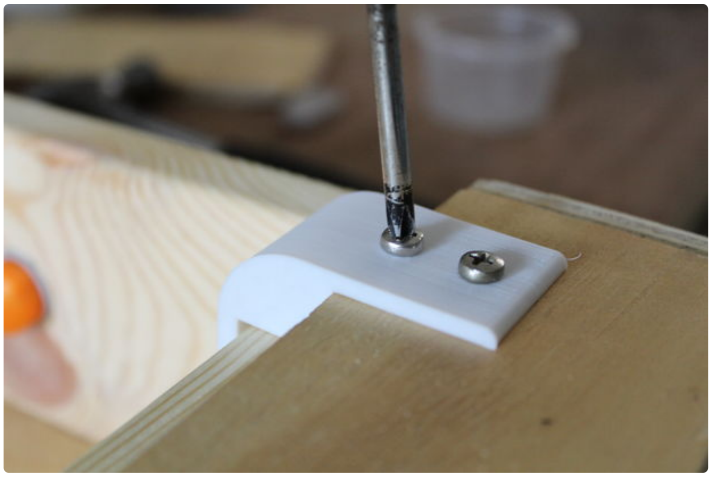
We built a compartment box that held all the electronics with the leftover plywood. We cut the wood according to the layout of the electronics and glued the pieces together. Once the glue had dried we mounted the power supply and Raspberry Pi into the compartment box, feeding the wires of the sensors through a slot. To cover the slots we pushed in printed covers to seal any gaps.

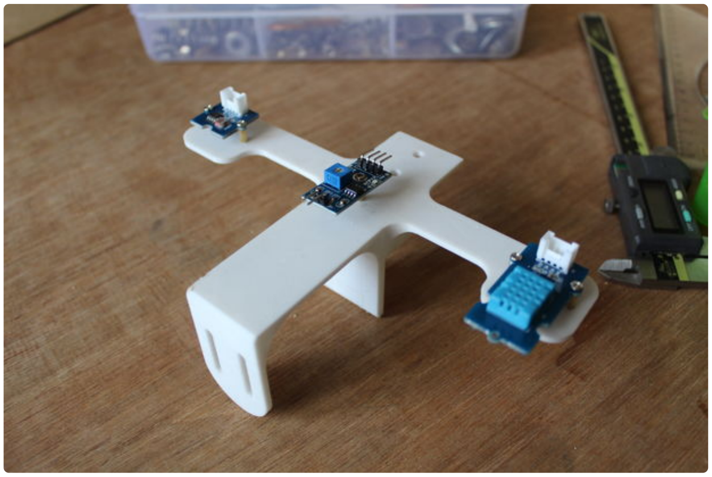
The Sensor Mount has holes to attach pegs on which you can mount the sensors. Attach the luminosity and humidity sensor on the top and the moisture sensor on the adjustable slot. To make the compartment box easily removable we screwed 3D printed hooks and the sensor mount which allowed the box to clip onto the main structure. This way, the electronic and iot system unit can be easily integrated to any planter.

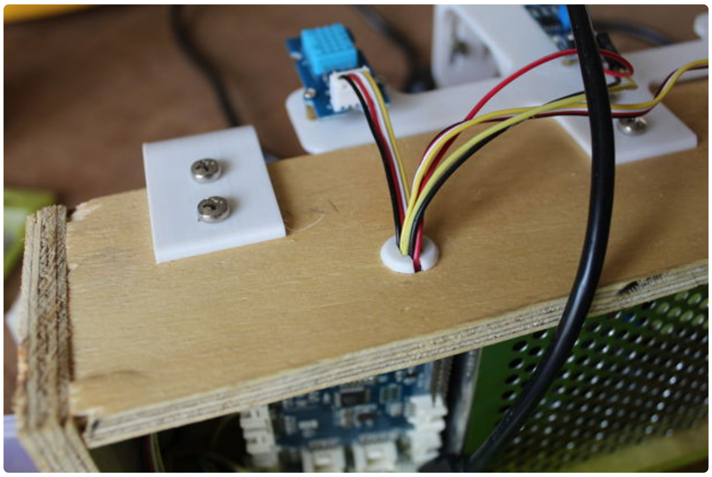


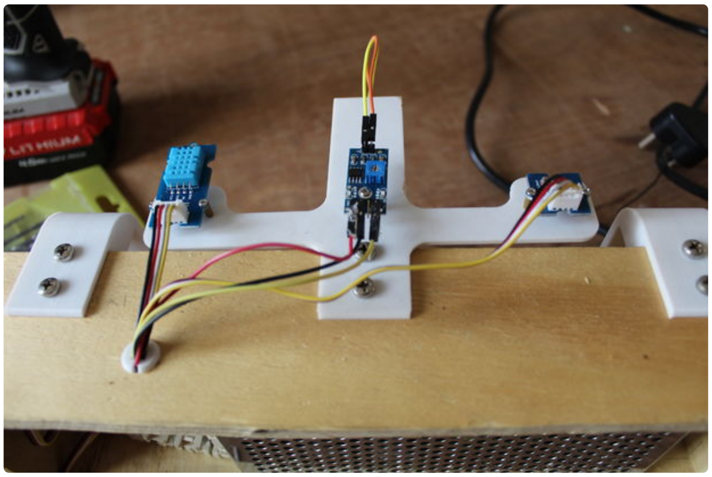

30 cm
6.5 cm
9.5 cm
11.5 cm









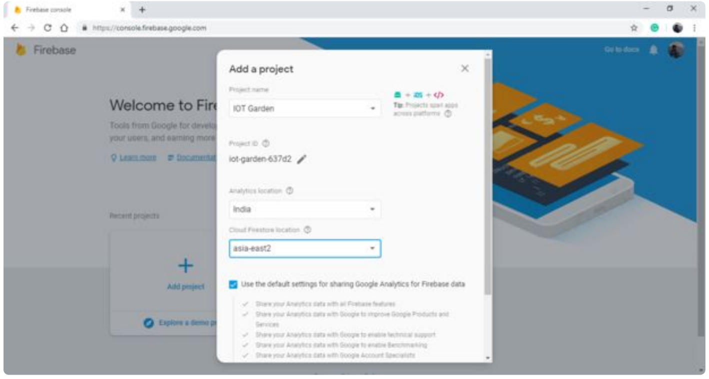


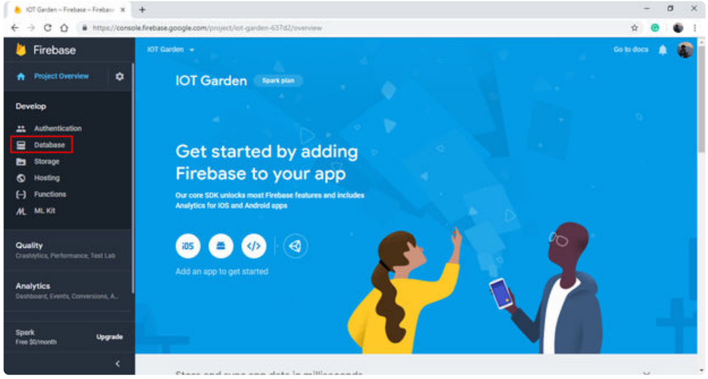
Step 12: Creating the Database

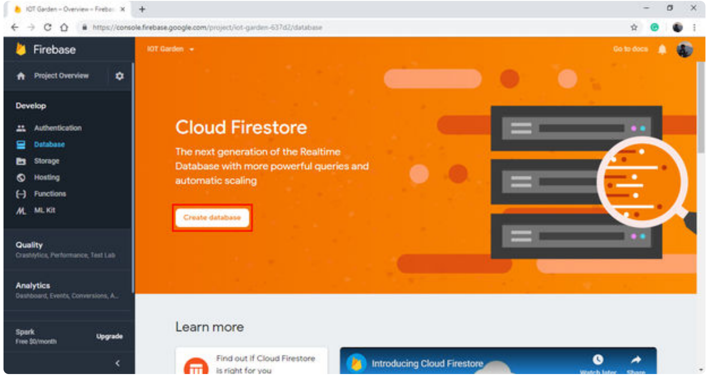
The first step is to create a database for the system. Click on the following link (Google firebase), which will lead you to the Firebase website (you will have to log in with your Google account). Click on the "Get Started" button which will take you over to the firebase console. Then create a new project by clicking on the "Add Project" button, fill in the requirements (name,details,etc) and complete by clicking on the "Create Project" button.

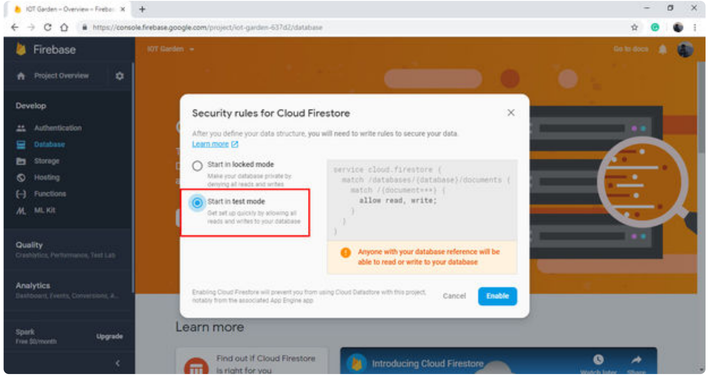
We just require Firebase's database tools, so select "database" from the menu on the left-hand side. Next click on the "Create Database" button, select the "test mode" option and click on "enable". Next set the database to a "realtime database" instead of the "cloud firestore" by clicking on the drop-down menu at the top. Select the "rules" tab and change the two "false" to "true", finally click on the "data" tab and copy the database URL, this will be required later on.

The last thing that you will need to do is to click on the gear icon next to project overview, then on "project settings", then select the "service accounts" tab, finally click on "Database Secrets" and note down the security code of your database. With this step complete, you have successfully created your cloud database which can be accessed from your smartphone and from the Raspberry Pi. (Use the pictures attached above in case of certain doubts, or just drop a question or comment in the comment section)

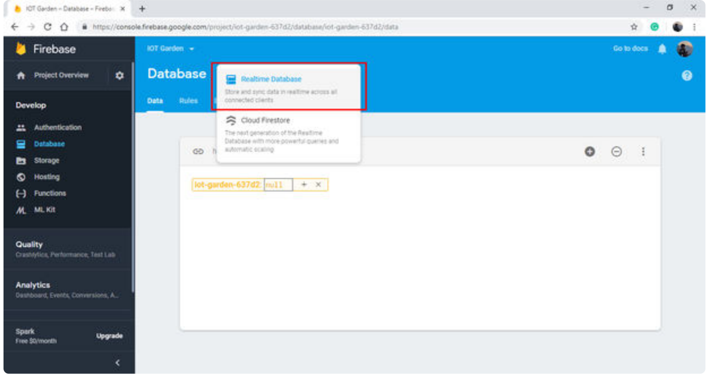


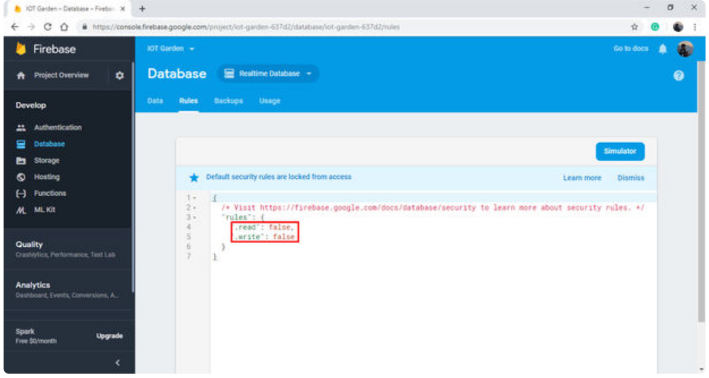



Firebase
Project Overview
Database
Cloud Firestore
The next generation of the Realtime Database with more powerful queries and automatic scaling
Create database
Learn more
Introducing Cloud Firestore

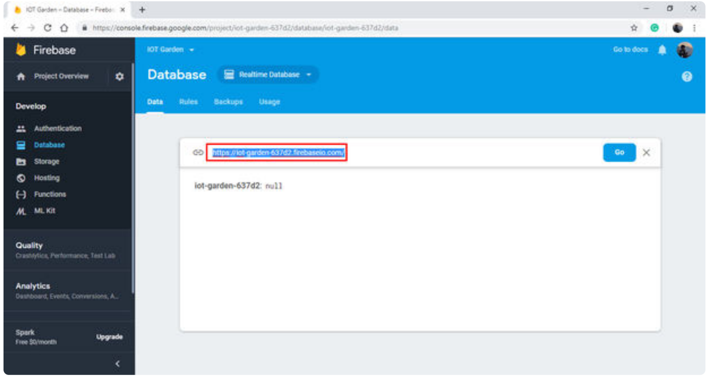

Security rules for Cloud Firestore
Start in locked mode
Start in test mode
Anyone with your database reference will be able to read or write to your database
Cancel
Enable

```
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write;
    }
  }
}
```



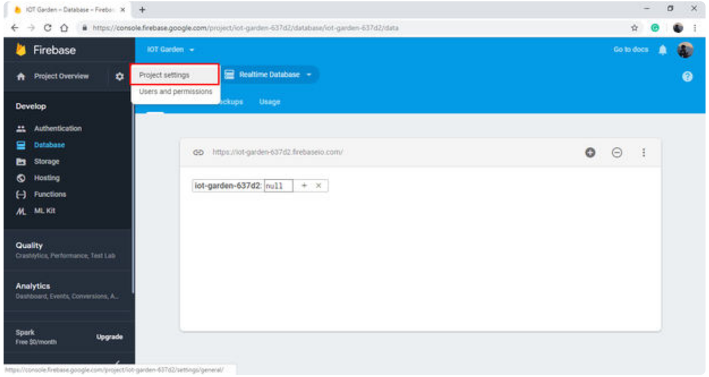


```
{
  /* Visit https://firebase.google.com/docs/database/security to learn more about security rules. */
  "rules": {
    ".read": false,
    ".write": false
  }
}
```

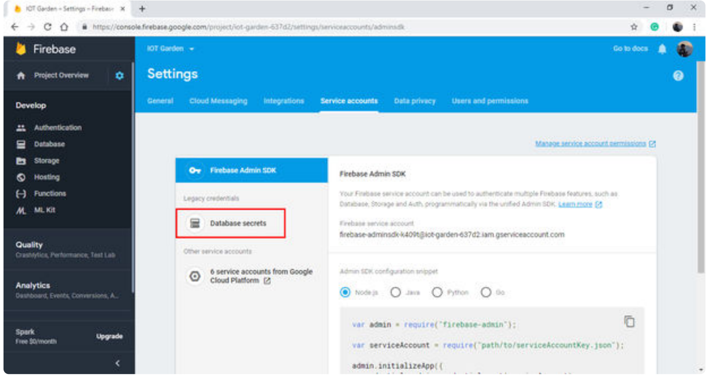


https://iot-garden-637d2.firebaseio.com/

iot-garden-637d2: null



iot-garden-637d2: null



Firebase Admin SDK

Your Firebase service account can be used to authenticate multiple Firebase features, such as Database, Storage and Auth, programmatically via the unified Admin SDK. Learn more

Firebase service account
firebase-adminsdk-k409t@iot-garden-637d2.iam.gserviceaccount.com

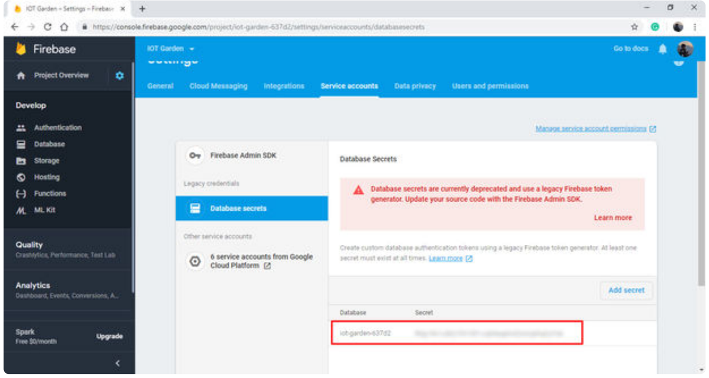
Admin SDK configuration snippet

● Node.js ○ Java ○ Python ○ Go

```
var admin = require("firebase-admin");

var serviceAccount = require("path/to/serviceAccountKey.json");

admin.initializeApp({
```



Database Secrets

Database secrets are currently deprecated and use a legacy Firebase token generator. Update your source code with the Firebase Admin SDK.

Create custom database authentication tokens using a legacy Firebase token generator. At least one secret must exist at all times. Learn more

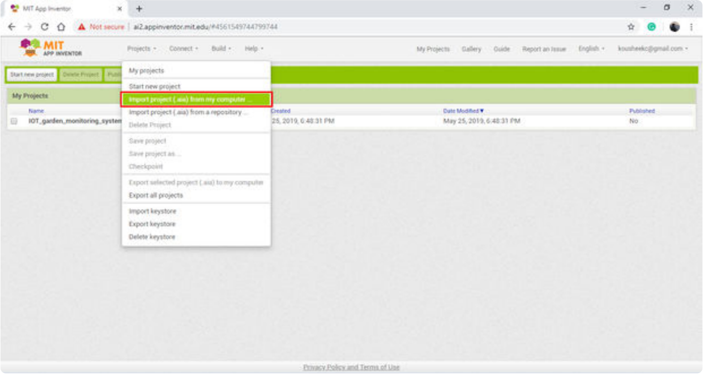
Database	Secret
iot-garden-637d2	[redacted]

Step 13: Setting Up the App

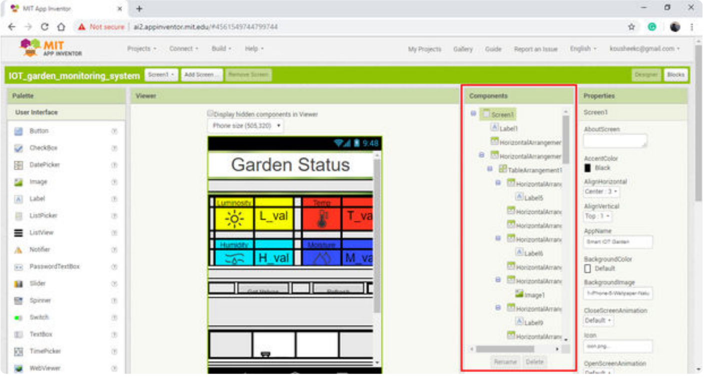
The next part of the IoT system is the smartphone application. We decided to use the MIT App Inventor to make our own customized app. To use the app that we created first open the following link (MIT App Inventor), which will lead you to their webpage. Next click on "create apps" towards the top of the screen and log in with your Google account.

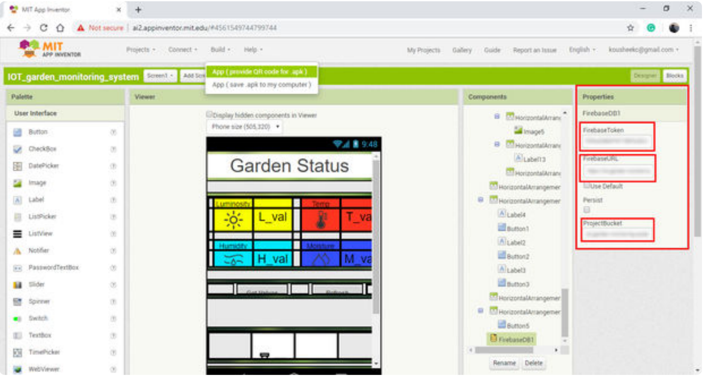
Download the .aia file that is linked below. Open the "projects" tab and click on "Import project (.aia) from my computer" next select the file that you just downloaded and click "ok". In the components window, scroll all down till you see "FirebaseDB1", click on it and modify the "FirebaseToken", "FirebaseURL" to the values that you had kept a note of in the previous step.

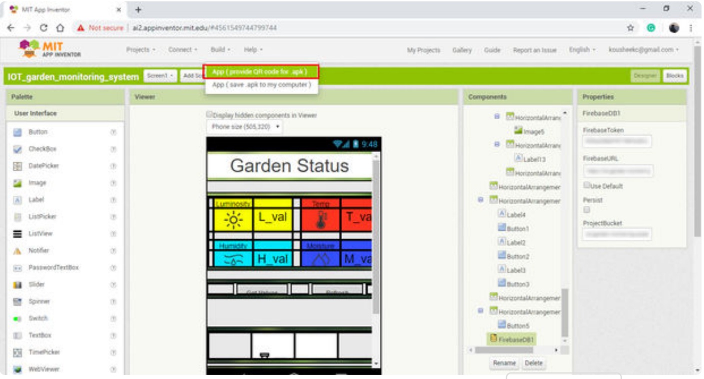
Once these steps are complete you are ready to download and install the app. You can download the app directly onto your phone by clicking on the "Build" tab and clicking on "App (provide QR code for .apk)" then scanning the QR code with your smartphone or clicking "App (save .apk to my computer)" you will download the apk file onto your computer which you need to shift onto your smartphone to then install.



IOT_garden_monitoring_system — Created: May 25, 2019, 6:48:31 PM — Date Modified: May 25, 2019, 6:48:31 PM — Published: No







https://www.instructabl…

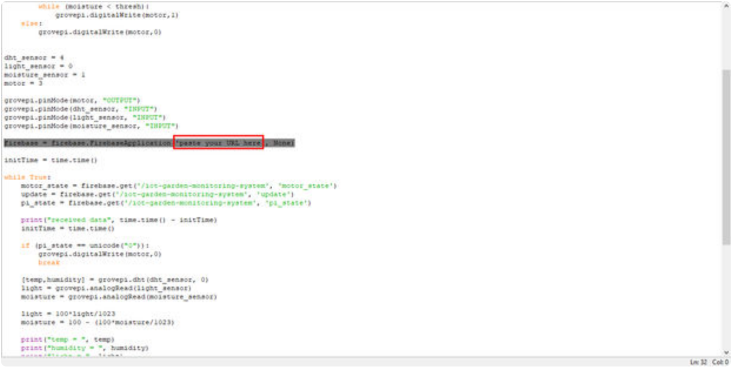
Download

Step 14: Programming the Raspberry Pi

The Raspberry Pi needs to be flashed with the latest version of Raspbian (Raspbian). In case you plan on using the GrovePi+ shield as we did, flash your Raspberry Pi with the latest version of "Raspbian for Robots" instead (Raspbian for Robots). Once you have flashed your Raspberry Pi you will need to install an additional python library. Open the terminal and paste the following commands:

1. **sudo pip install requests==1.1.0**
2. **sudo pip install python-firebase**

Once that is done, download the file attached below and save it onto a directory on your Raspberry Pi. Open the file and scroll down to line 32. On this line replace the part that says "paste your URL here" with your database's URL that you had noted earlier, make sure to paste the URL in between the ' 's. With this, you are done, open the terminal and run the python script using the "python" command.



```
        while (moisture < thresh):
            grovepi.digitalWrite(motor,1)
    else:
        grovepi.digitalWrite(motor,0)


dht_sensor = 4
light_sensor = 0
moisture_sensor = 1
motor = 3

grovepi.pinMode(motor, "OUTPUT")
grovepi.pinMode(dht_sensor, "INPUT")
grovepi.pinMode(light_sensor, "INPUT")
grovepi.pinMode(moisture_sensor, "INPUT")

firebase = firebase.FirebaseApplication('paste your URL here', None)

initTime = time.time()

while True:
    motor_state = firebase.get('/iot-garden-monitoring-system', 'motor_state')
    update = firebase.get('/iot-garden-monitoring-system', 'update')
    pi_state = firebase.get('/iot-garden-monitoring-system', 'pi_state')

    print("received data", time.time() - initTime)
    initTime = time.time()

    if (pi_state == unicode("0")):
        grovepi.digitalWrite(motor,0)
        break

    [temp,humidity] = grovepi.dht(dht_sensor, 0)
    light = grovepi.analogRead(light_sensor)
    moisture = grovepi.analogRead(moisture_sensor)

    light = 100*light/1023
    moisture = 100 - (100*moisture/1023)

    print("temp = ", temp)
    print("humidity = ", humidity)
```

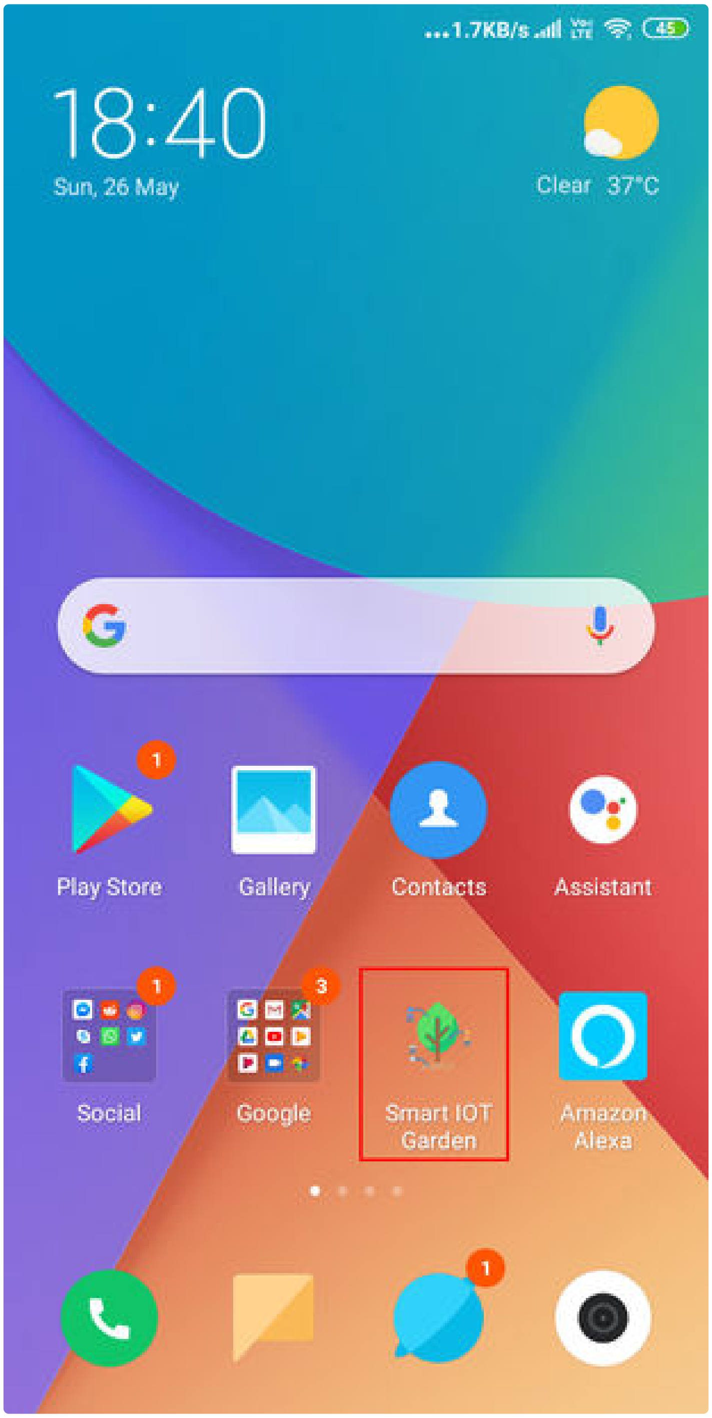
https://www.instructabl…

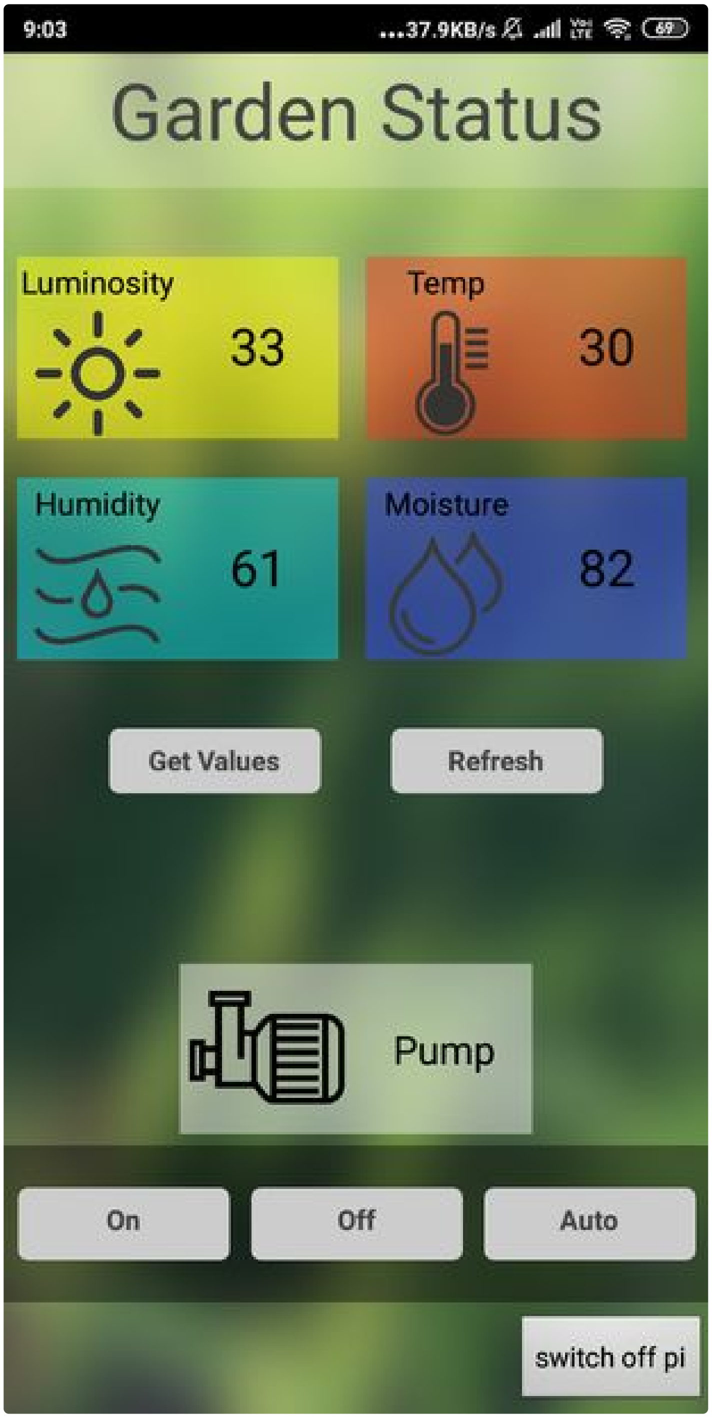
Download

Step 15: Using the App

The interface of our app is quite self-explanatory. The top four boxes display real-time values of luminosity, temperature, humidity and the soils moisture content in percentages. These values can be updated by clicking on the "get values" button which instructs the Raspberry Pi to update the cloud database followed by the "refresh" button which refreshes the screen once the database has been updated.

The lower section of the screen is for the drip irrigation system. The "on" button turns on the water pump while the "off" button switches it off. The "auto" button utilizes the various sensor values to calculate the exact water needed on a daily basis and waters the plants twice per day at 8 AM and 4 PM.

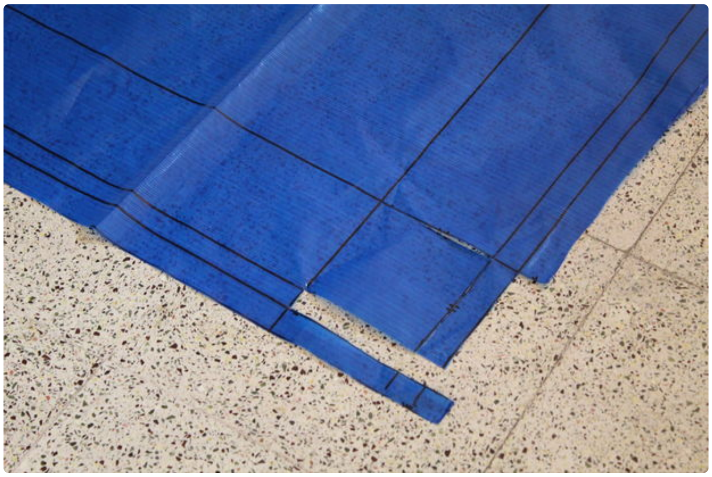
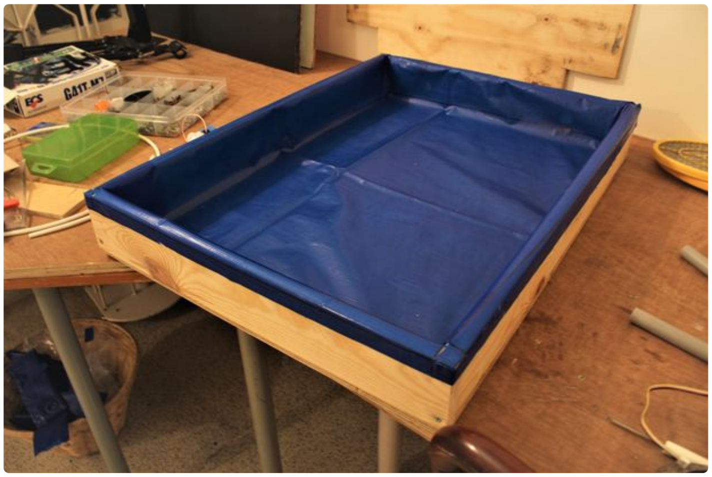




Step 16: Tarpaulin Liner

As the soil's moisture might rot the wood over time, we cut down a sheet of tarpaulin to size and lined it on the inner surface of the planter. Make sure to pull it over the sides and then finally hold it in place with some glue. Once done we filled in soil that we got from a local farm. Spread the soil evenly till the top and then embed the three rows of the drip irrigation tubing.

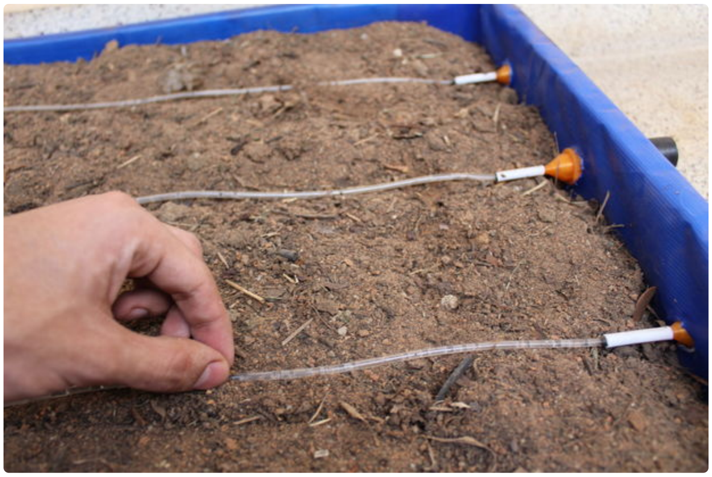
On the corner near the water pipes fit the electronic box and embed the moisture sensor into the soil. These makes the wiring job easier as the solenoid valve is near to the electronics and can be easily connected.

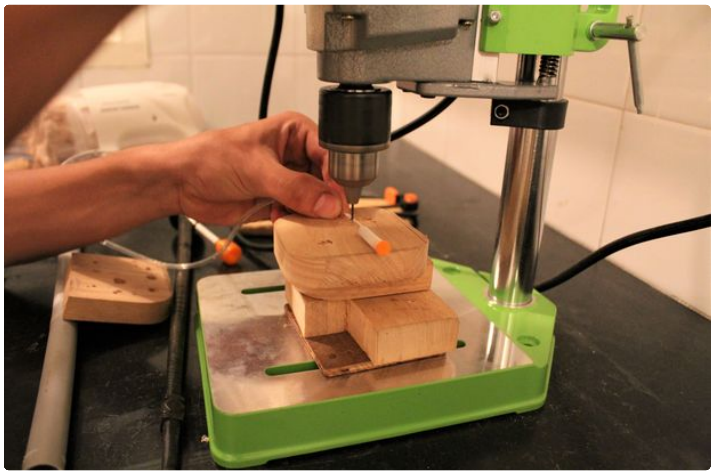



Step 17: Drip Irrigation System

Cut three pieces of the surgical tube stretching along the length of the planter (around 70cm) this will act as the main drip line for the plants. Therefore plan out the needed spacing between the plants and drill a 1mm hole and the intervals. Test if the water drips easily and enlarge the holes if needed. Use the three plugs to close the ends making sure the water is restricted to come out only from the drip holes.

Slightly embed the tubes in the soil and your ready to water your plants!





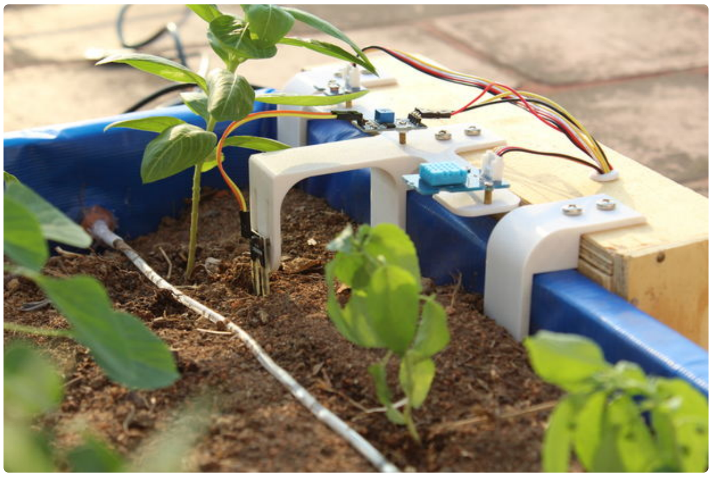
Step 18: Planting Results

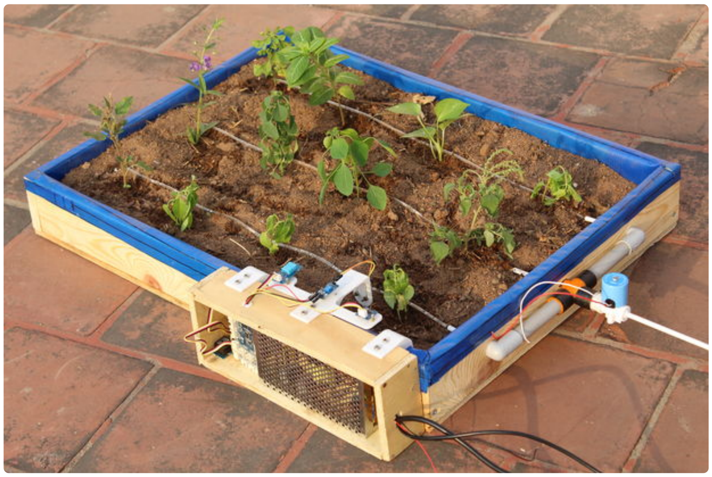
The pictures above are the results of the iot garden working for a month. The plants are healthy and we managed to grow herbs such as mint and coriander.

Through experimentation, we have noticed that the auto-mode saves close to 12% of water per day. As the plants are watered through drip irrigation, their roots grow straight giving more space to grow more plants in the planter. The only drawback that we observed was that the bigger plants need more soil depth. That said due to the modular construction one can easily add a deeper base to their requirements.

To conclude, this system not only makes your garden more efficient but also ensures the well-being of your plants as the real time data feedback provides a robust method to give the right amount of water and sunlight. We hope that the instructable was useful and that it will help you grow your very own iot garden.

Happy making!





Hi,

Great project! But i cant seem to find your link to the aia file even though i looked many times. Sorry but could you help post the link again? My students are trying this out . Thanks!

A neat project with great instructions! Great use of the internet of things as well!

Thanks!

Wow I was looking for something just like this, couldn't have found a better project. Thank you so much for sharing, it looks neat, simple and quite efficient!

Happy to know that this i'ble could be of help

Wonderful project and so well presented. The app looks cool too. A very efficient and eco friendly step towards healthy plants and reduction of water wastage. Good job :)

Thanks for the positive feedback