



**VIT**<sup>®</sup>  
Vellore Institute of Technology  
(Deemed to be University under section 3 of UGC Act, 1956)

## **SCHOOL OF COMPUTER SCIENCE** **ENGINEERING**

### **“SMART GARDEN MANAGEMENT SYSTEM”**

By

Jaivant Vassan **19BCE2322**

Divakar. A.S **19BCE2337**

Kabhilan S **19BCE2339**

Sandheep B S **19BCE2341**

Prithvi Saran S **19BCE2344**

**Project Report**  
*of*

**CSE2006 – MICROPROCESSOR & INTERFACING**

**Fall Semester 2021-22**

Submitted to:

**Faculty: Prof. Manish Kumar**

**Date: 28.11.2021**

**Slot: L29+L30**

**Signature:**

<b>S.NO</b>	<b>CONTENT</b>	<b>PAGE NO.</b>
1.	Abstract	2
2.	Introduction	3
3.	Literature Survey	4,5,6
4.	Drawback of the existing work	11
5.	Proposed work (Block Diagram, pin diagram, flow chart)	12
6.	Hardware/Software Requirements	15
7.	Implementation	17
8.	Challenges Faced and Design Constraints	19
9.	Screenshots	20
10.	Result & Conclusion	22
11.	References	23
12.	Appendix (code)	24

## **ABSTRACT**

Technology has made significant advances in every aspect of life, including industry and agriculture. Our survival is largely reliant on agricultural progress. Researchers are attempting to incorporate current technologies into agriculture in order to establish new strategies for improving agricultural and production health. People in big cities are having trouble maintaining their homemade gardens due to a lack of qualified gardeners. The best way to solve this problem is to modernize traditional gardening methods while also keeping a close eye on plant growth. As a result, the suggested system aims to provide a smart means of monitoring plant growth through the use of automation and IoT technology. The internet of things offers a variety of uses for agricultural growth and monitoring. The main theme of this project is to maintain gardening by smart monitoring it. The Raspberry Pi microprocessor is used to collect data from the garden such as temperature, moisture, luminosity, and humidity of the soil through sensors and store it in the cloud data, which can be accessed through our smartphone by creating a garden monitoring app, where you can control the water supply for the plants in the garden and maintain the health status of the garden. As a result of this methodology, we can maintain our gardens in our hectic schedules, and we can do automatic watering for our gardens dependent on the temperature. The main benefit of this system is that it creates an ideal climate for plant development, reduces water usage, and adds greenery to a garden or terrace.

## **1. Introduction:**

Plants are recognized to be a vital source of survival and aid in the purification of polluted air. Few people feel obligated to plant a tree, and some see it as a pastime. Planting a tree entails more than just burying a seed ball in the ground; there are other aspects to consider. Some plants require extra attention for optimal growth. Some plants are planted only for the purpose of display and homegrown agriculture. For photosynthesis to occur, the plant must be given the proper environment and watered on a regular basis. We also know that a single type of soil or nutrient is insufficient for all plants to thrive. Each plant has its own traits that help it produce a high yield.

To solve all of these issues, the best option is to keep an eye on the garden. The concept of the Internet of Things emerges from the fast-rising use of the internet, multipurpose gadgets (laptops, smartphones, tablets, etc.), and objects created to perform a specific purpose. IoT refers to a network of items or entities that can communicate with one another over the Internet. Things are predicted to become active participants in the fields of business, social processes, information, and communication by utilizing IOT. They must be able to interact with the environment as well as communicate with one another while exchanging and altering environmental data and information. The mechanisms that develop services and initiate activities, with or without human participation, have an automatic influence on it. In this regard, we offer a model that could effectively care for a garden without the need for human intervention.

This can be accomplished with a low-cost credit card-sized Raspberry Pi that is controlled via the internet and runs on Android. The purpose of the application is to create programs that allow a distant user to communicate with one or more interface cards using a smartphone, web browser, and Raspberry Pi card. If you only need to read simple analog signals, such as those from a temperature or light sensor, the Raspberry Pi is an excellent choice. This technology not only monitors the growth but also sends out alarms when there is a problem with the growth or if the environment is not right. The Internet of Things can be used to develop this type of system (IOT). IOT is defined as a phenomenon in which a system or a gadget function with the assistance of the internet.

## 2. Literature review:

[1] *IoT based smart garden monitoring system using NodeMCU microcontroller*

**Mubashir Ali, Nosheen Kanwal, Aamir Hussain, Fouzia Samiullah, Aqsa Iftikhar, Mehreen Qamar**

**August 2020**

Technology brings a remarkable advancement in every field of life, whether its industry or agriculture. Our lives are essentially dependent on agricultural development. Researchers are working to integrate modern technologies in agriculture to develop new practices for the enhancement of healthy agriculture and production. Internet of things is a domain of computer science that provides mechanisms and techniques to interconnect a wide range of digital devices to automate the real-life systems. In big cities, peoples facing problems in their homegrown gardens regarding the maintenance and availability of proper gardeners. This research paper has proposed an IoT based approach for smart garden monitoring using NodeMCU 8 microcontroller that helps the users in identifying current parameters of temperature, moisture, and humidity of their homegrown plants and gardens. A prototype has been implemented to show the real illustration of the proposed approach. An android mobile application has been developed to display the real-time profiles of environmental factors like temperature, moisture, and humidity. With the help of this system, users will be able to treat their gardens in a better way in terms of plant health and growth. This research work replaces the need for gardeners and issues faced during the maintenance of gardens in big cities. The purpose of this research is to introduce and prosper the IoT innovation towards smart cities in our society.

[2] *Smart Gardening: A solution to your gardening issues*

**Bhadra Mayuri, Chakraborty Niloy, and Mukherjee Adrika**

**February 2020**

The technology which could make our life prosper within the walls could also help to create our own corner of nature nourish. In this project, a smart gardening system has been proposed that utilizes the concept of the Internet of Things (IoT). The main objective of this project is to optimize water usage during gardening and remotely maintain the garden. In this system, important information related to the plant, like, temperature, relative humidity, and the soil moisture is continuously recorded in a cloud-based Database. Artificial Intelligence (AI) based planning is done in regular intervals for watering the plants and provide adequate lighting in the garden area for aesthetics. The real-time sensor status can be directly monitored and controlled by the end-users of the garden through his or her smartphone using Telegram application. A plant recognition model has been introduced in this system, where a Convolutional Neural Network (CNN) [1] based deep-learning algorithm classifies the plant categories. Moreover, this model also informs the end

user about the health of the plant. The technology which could make our life prosper within the walls could also help to create our own corner of nature nourish. In this project, a smart gardening system has been proposed that utilizes the concept of the Internet of Things (IoT). The main objective of this project is to optimize water usage during gardening and remotely maintain the garden. In this system, important information related to the plant, like, temperature, relative humidity, and the soil moisture is continuously recorded in a cloud-based Database. Artificial Intelligence (AI) based planning is done in regular intervals for watering the plants and provide adequate lighting in the garden area for aesthetics. The real-time sensor status can be directly monitored and controlled by the end-users of the garden through his or her smartphone using Telegram application. A plant recognition model has been introduced in this system, where a Convolutional Neural Network (CNN) [1] based deep-learning algorithm classifies the plant categories. Moreover, this model also informs the end user about the health of the plant.

### **[3]** *Web Based Smart Irrigation System Using Raspberry Pi*

**Dr. Dhiraj Sunehra**

**2019**

The major population of India mainly depends upon Agriculture. Nowadays the ground water levels are also decreased due to global warming and uncertain rain fall. The conventional irrigation techniques have some problems like farmer should visit the field regularly even at night, then he may face some hazards like snake bite, electric shock and etc. In this paper a web based smart irrigation system using Raspberry Pi is implemented. The proposed system checks the temperature and moisture content present in the soil. The motor is also automatically controlled ON and OFF, when the 3-phase supply is present in the field and also with the flow of water. The acquired temperature and moisture content of the soil are sent to Raspberry Pi using nRF24L01 transceiver. Raspberry Pi posts the received values on the webpage. When we enter the acquired parameter values in the soil test blogs, the webpage shows the suggestions of pesticides and crops that need to be taken by the farmer to increase the yield. The system can reduce the water consumption and wastage of water.

### **[4]** *Smart Home Garden Irrigation System Using Raspberry Pi*

**S.N. Ishak, N.N.N.Abd Malik, N.M. Abdul Latiff, N. Effiyana Ghazali, M. A. Baharudin**

**November 2017**

Irrigation system is a method of allowing water to drip slowly to the roots of plants, either onto the soil surface or directly onto the root zone, through solenoid valve. However, it is found that the market price of the system is expensive for small area coverage. Thus, this paper proposes a design for smart home garden irrigation system that implements ready-to-use, energy efficient, and

cost effective devices. Raspberry Pi, which is implemented in this system is integrated with multi-sensors such as soil moisture sensors, ultrasonic sensors, and light sensors. This proposed system managed to reduce cost, minimize waste water, and reduce physical human interface. In this paper, the relay is utilized to control the switching of solenoid valve. The system also managed to measure moisture of the soil and control the solenoid valve according to human's requirements. It is conducted with Graphical User Interface (GUI) using Android application to activate watering activity. Email notification is also sent to the home user for alert purposes either for normal or critical operations. An experimental setup has been tested and it is proven that the system can intelligently control and monitor the soil moisture levels in the experiment field.

## **[5]** *Automated Smart Irrigation System using Raspberry Pi*

**Tanu Sahu, Ashok Verma**

**August 2017**

Water is the most essential contribution for upgrading agricultural productivity and therefore expansion of water system has been a key format in the improvement of farming in the nation. An Automated Sprinkler irrigation method distributes water to crops/plants by spraying it over the crops/plants like a natural rainfall. In this thesis we will develop an automated sprinkle system that will help a farmer/people to know about his field, and the status of his plant at his home or he may be residing in any part of the world. This work will help the farmers to irrigate the farmland in a very efficient manner with automated irrigation system based on soil, humidity, weather. This sprinkler system will provide control for soil temperature, moisture sensing to ensure plants is watered when there is demand, live streaming and also provide the temperature, humidity sensing, forecast lookup from other weather services. Whenever there is a change in temperature, humidity and current status of rain of the surroundings these sensors sense the change in temperature and humidity and gives an interrupt signal to the raspberry pi. Water excess irrigation not only reduces plants production but also damages soil fertility and also causes ecological hazards like water wasting and salinity. In recent years the awareness of water and energy conservation has resulted in the greater use of sprinkler system. Currently the automation is one of the important roles in the human life. It not only provides comfort but also reduce energy, efficiency and time saving. Now a day the industries are using an automation and control machines which are high in cost and not suitable for using in a farm & garden field. So in this work we will design a smart irrigation technology based on IoT using Raspberry pi. The proposed sprinkler system will be low in cost and usable by the Indian farmers. Raspberry pi is the main heart of the overall system.

**Base:** The base is defined as the period from the first to the last watering of the crop just before its maturity. It is also known as base period. It is denoted by  $\cdot s \cdot$  and expressed in number of days. The base period for some common crops are given in the below table.

Crop	Base in days
Rice	120
Wheat	120
Maize	100
Cotton	200
Sugarcane	320

**Delta:** Each crop requires certain amount of water per hectare for its maturity. If the total amount of water supplied to the crop (from first to last watering) is stored on the land without any loss, then there will be a thick layer of water standing on that land. This depth of water layer is known as Delta for the crop. It is denoted by  $\Delta$  and expressed in cm. Delta for some common crops is given in the below table.

Kharif	Delta in cm
Rice	125
Maize	45
Ground nut	30
Millet	30
Rabi crop	35
Wheat	40
Mustard	45
Gram	30
Potato	75

**Duty:** The duty of water is defined as number of hectares that can be irrigated by constant supply of water at the rate of one cumec throughout the base period. It is expressed in hectares/cumec. and is denoted by  $\cdot o \cdot$ . The duty of water is not constant. but it varies with various factors like soil condition, method of ploughing, method of application of water, etc. The duties of some common crops are given in the below table.

Crop	Duty in hectares/cumec
Rice	900
Wheat	1800
Cotton	1400
Sugarcane	800



Crop	Total growing period (days)	Crop	Total growing period (days)
Alfalfa	100-365	Millet	105-140
Banana	300-365	Onion green	70-95
Barley/Oats/Wheat	120-150	Onion dry	150-210
Bean green	75-90	Peanut/Groundnut	130-140
Bean dry	95-110	Pea	90-100
Cabbage	120-140	Pepper	120-210
Carrot	100-150	Potato	105-145
Citrus	240-365	Radish	35-45
Cotton	180-195	Rice	90-150
Cucumber	105-130	Sorghum	120-130
Eggplant	130-140	Soybean	135-150
Flax	150-220	Spinach	60-100
Grain/small	150-165.	Squash	95-120
Lentil	150-170	Sugarbeet	160-230
Lettuce	75-140	Sugarcane	270-365
Maize sweet	80-110	Sunflower	125-130
Maize grain	125-180	Tobacco	130-160
Melon	120-160	Tomato	135-180

### Region-Based Moisture Requirements:

Climate Zone	Mean daily temperature		
	Low (less than 15°C)	Medium (15- 25°C)	High (more than 25°C)
Desert/arid	4-6 mm	7-8 mm	9-10 mm
Semi-arid	4-5 mm	6-7 mm	8-9 mm
Sub-humid	3-4 mm	5-6 mm	7-8 mm
Humid	1-2 mm	3-4 mm	5-6 mm

**Average Crop Requirements Over Growing Period:**

S.no	Crop	Temperature Required (°C)	Moisture Required (cm)
1.	Rice	22-32	150-300
2.	Wheat	10-26	75-100
3.	Millet	27-32	50-100
4.	Gram	20-25	40-45
5.	Sugar Cane	21-27	75-150
6.	Cotton	21-30	50-100
7.	Oilseeds	20-30	50-75
8.	Tea	20-30	150-300
9.	Coffee	15-28	150-250
10.	Banana	20-30	120-220
11.	Beans	10-27	30-50
12.	Cabbage	15-21	35-50
13.	Melon	18-35	40-60
14.	Onion	20-25	35-55
15.	Potato	15-20	50-70
16.	Peas	10-18	35-50
17.	Tomato	15-32	40-80

**Sensor Ranges:**

Sensor	Range in Standard units	Sensor Depiction Range
Humidity	5-99% RH	0-1023
Temperature	-40-80 °C	0-1023
Luminosity	0-1000 Lux	0-1023
Moisture	0-950	0-1023

## Nutrient Requirements:

Supplied from air and water	Supplied from soil and fertiliser sources	
	Macronutrients	Micronutrients
Carbon (C)	<b>Nitrogen</b> (N)	Zinc (Zn)
Hydrogen (H)	Phosphorous (P)	Copper (Cu)
Oxygen (O)	<b>Potassium</b> (K)	Iron (Fe)
	Sulphur (S)	Maganese (M)
	Calcium (Ca)	Boron (B)
	Magnesium (Mg)	Chlorine (Cl)
		Molybdenum (Mo)
		Cobalt (Co)

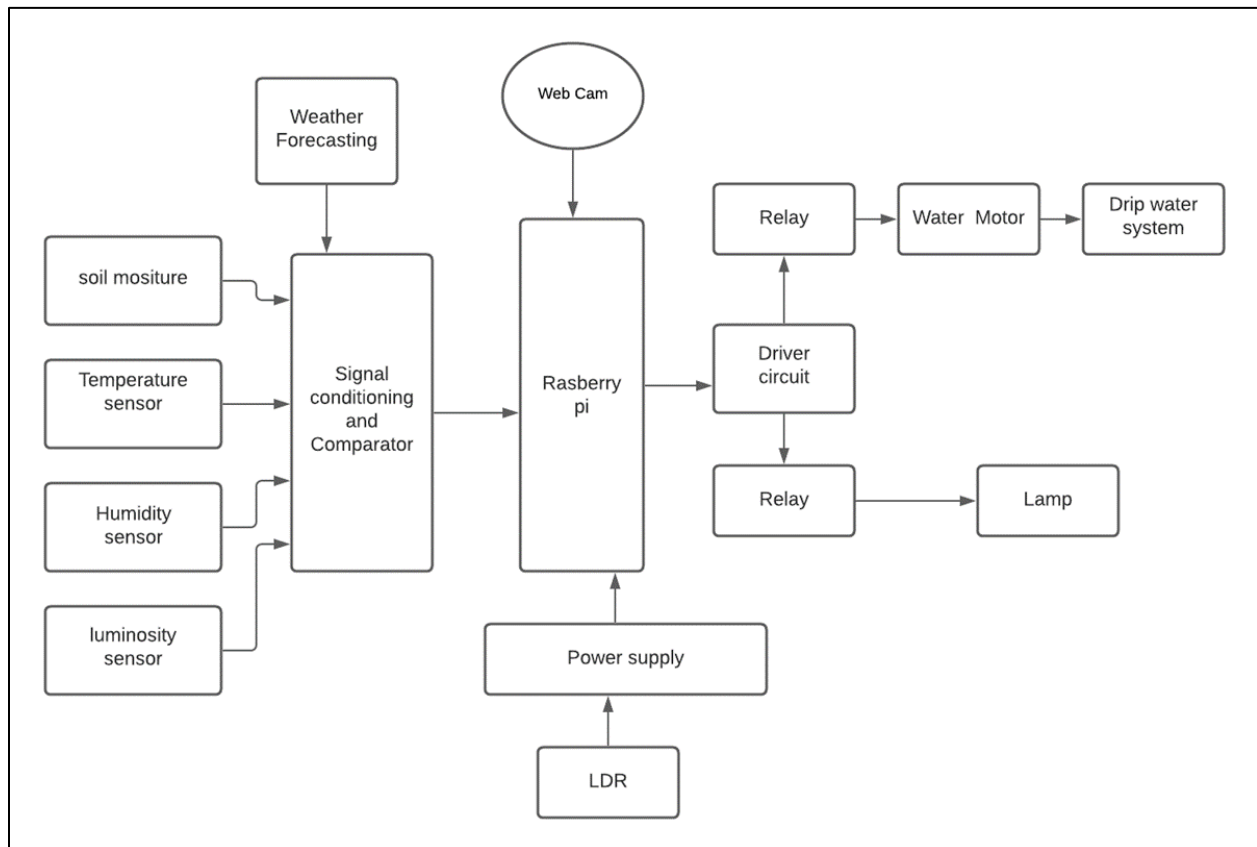
### **3. Drawback in the Existing Work:**

The existing works were not well researched and implemented. They missed out on many features like ventilation etc. They were also generally configured to provide water irrespective of the plant types. The existing works were also limited to 1 or 2 plants and did not have the potential to be used large scale because of their lack of research. We have done our own research and considered parameters like ventilation which will give the system a potential to be used in a large-scale basis.

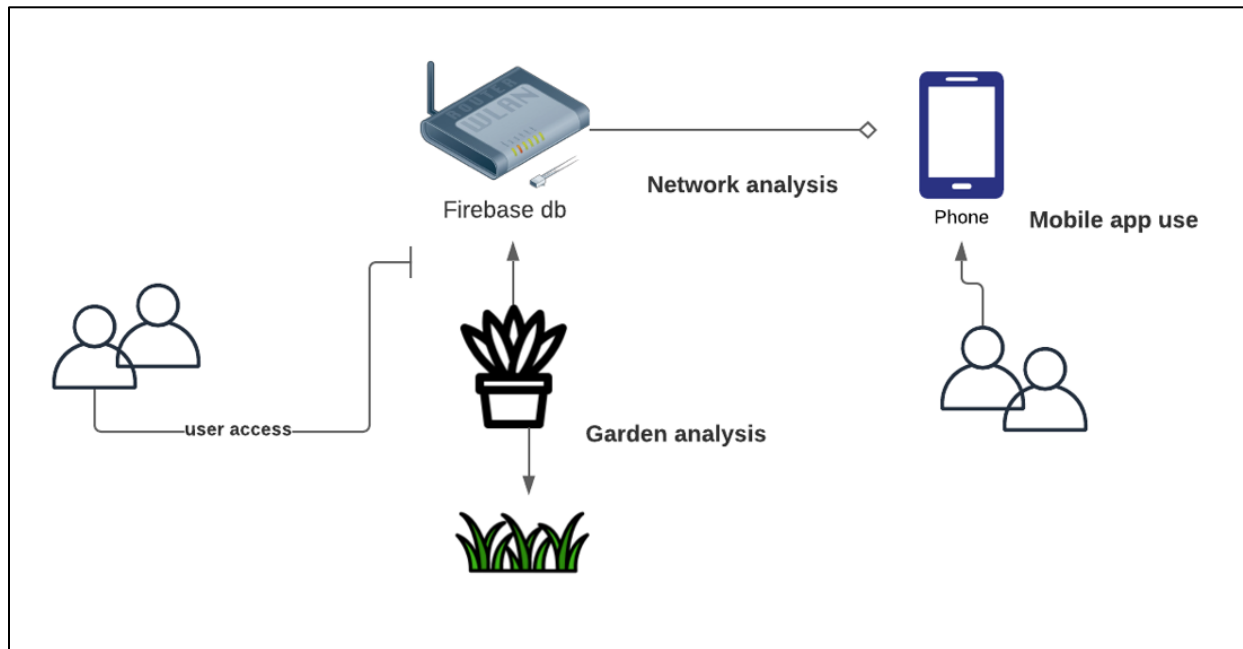
## 4. Proposed Work:

It usually consists of a central microcontroller to which other objects are connected. The smart garden consists of Raspberry Pi as a hub to which different types of sensors such as moisture sensor, humidity sensor, temperature sensor are connected. Sensors are connected to their respective positions and these sensors send the data to Raspberry Pi. Firebase is a database available on the internet in which real-time values of the sensor are updated every second. Android application is developed using android studio software. Within the software, the connectivity between the application and firebase will be made. So, the user can monitor the parameters from anywhere. Watering of garden varies with the type of soil. Hence the values of the sensors are predetermined for automation purposes inside the software. Whenever the user finds need of watering the garden, a switch in the application will automate the process or manually turn the motor on to water the plants. This helps in complete maintenance of the garden.

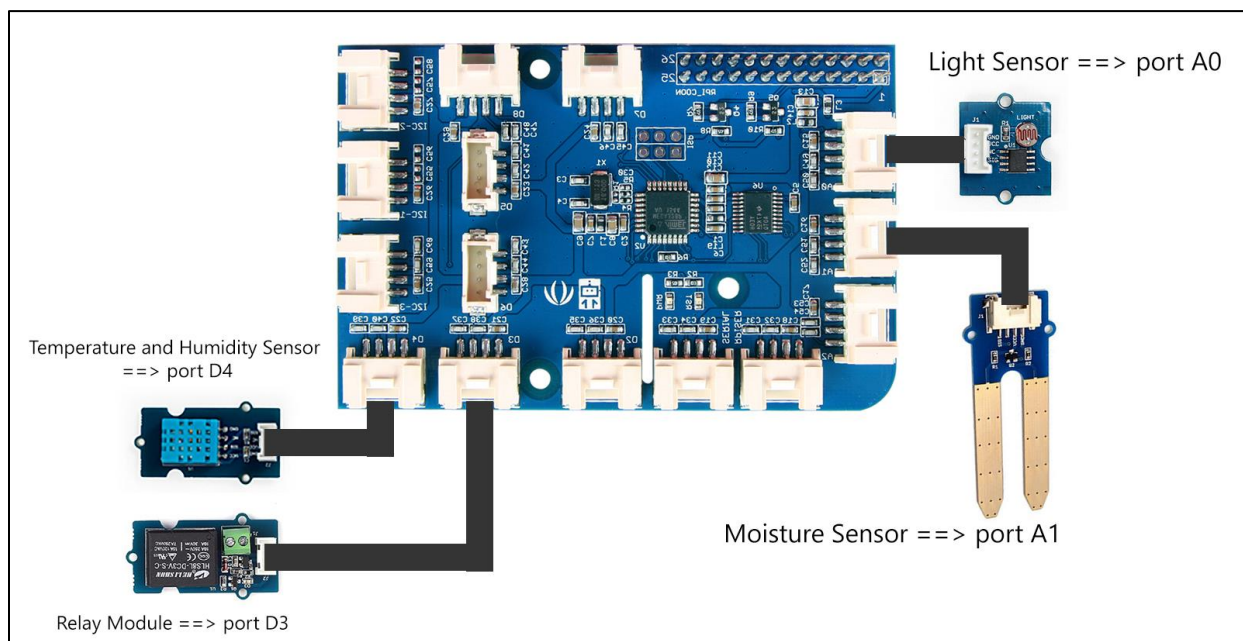
### 4.a. Block Diagram:



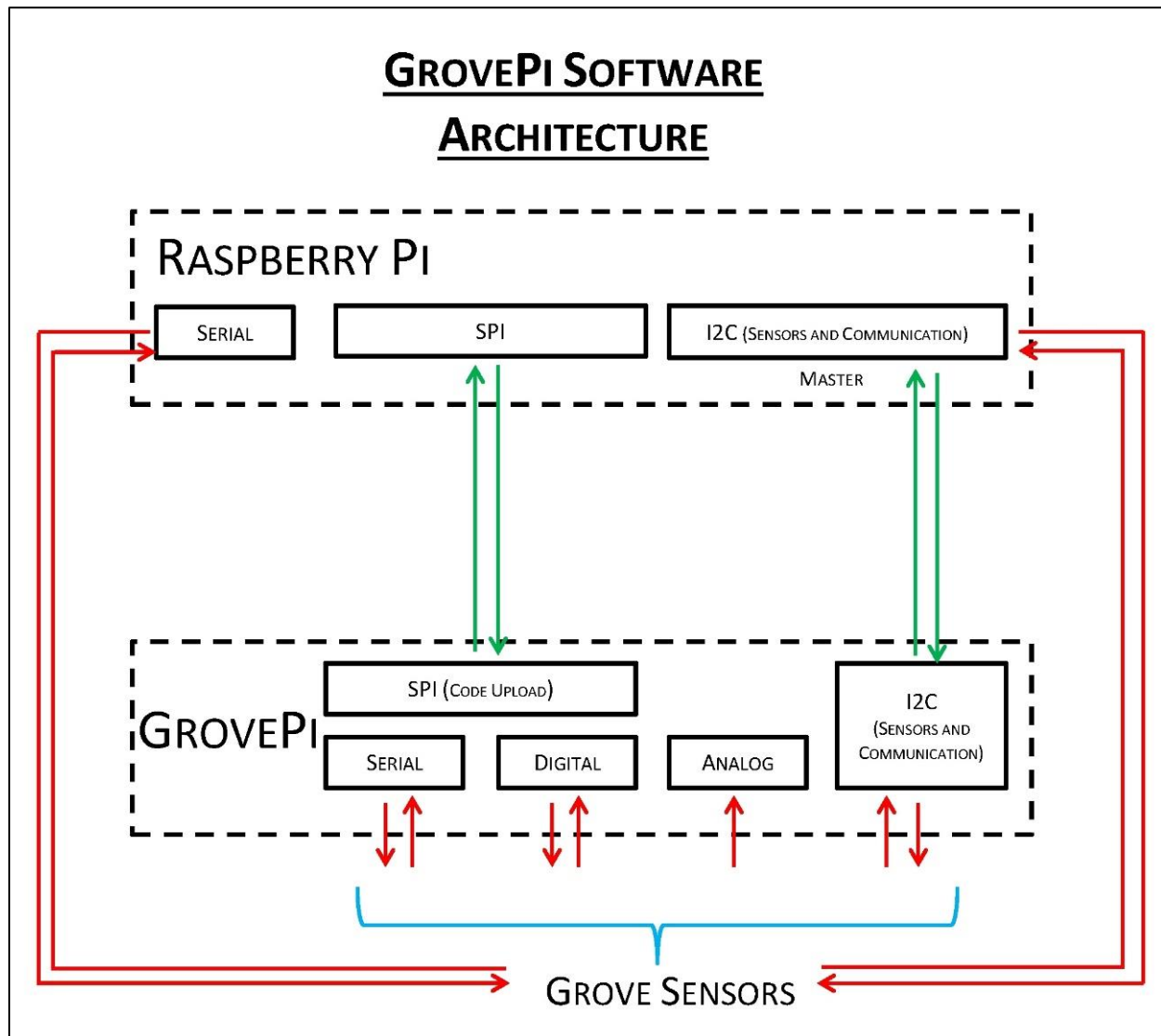
#### 4.b. Flow Diagram:



#### 4.c. Wiring Diagram:



#### 4.d. Interface Diagram:



## **5. Hardware/Software Requirements:**

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

### **Reason For Choosing Raspberry Pi:**

The key reason we chose raspberry pi over a standard Arduino microcontroller is the database component; relaying the data collected from the sensor to the database (and consequently to the app interface) is key to our model and also, our automation algorithm utilizes processed scientific data of different varieties of plants (which we have uploaded to the firebase database) to better streamline the care given to the plants through our actuators setup.

### **Sensors Research:**

We chose to use few of GrovePi's wide variety of sensors for our model instead of standard raspberry pi compatible sensors. Grove sensors allow you to interact and monitor the world. Raspberry Pi lets you store and process data and bring the real world to the Raspberry Pi and the web. The GrovePi brings both Grove Sensors and the popular Raspberry Pi together in a simple, elegant, and open-source design. The GrovePi board slips over the Raspberry Pi. The Grove Sensors should be connected to the GrovePi board.

### **Moisture Sensor (Grove-moisture):**

This Moisture Sensor can be used for detecting the moisture of soil or judge if there is water around the sensor. This sensor is very easy to use, you can just simply insert it into the soil and read the data.

### **Temperature And Humidity Sensor (Grove DHT11 Sensor):**

This Temperature and Humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity, and the temperature is measured by a



negative temperature coefficient (NTC) thermistor. It has excellent reliability and long-term stability. This sensor will not work for temperatures below 0 degree.

### **Light Sensor(Grove Luminosity Sensor):**

The Grove - Light sensor integrates a photo-resistor (light dependent resistor) to detect the intensity of light. The resistance of photo-resistor decreases when the intensity of light increases. A dual OpAmp chip LM358 on board produces voltage corresponding to intensity of light (i.e. based on resistance value). The output signal is analog value, the brighter the light is, the larger the value.

## 6. Implementation:

### 6.a. Database Creation:

The system's first step is to construct a database. We make use of Google Firebase for this. By filling in the prerequisites in the Firebase console, we can establish a new project. We only need Firebase's database tools, so we go to the menu and choose 'database'. The database is then created, and test mode is enabled. Instead of using the "**cloud firestore**," we used a "**Realtime database**." After that, we set up some database rules. We take note of the database's security code from the 'Database Secrets' tab. As a result, we've built a cloud database that can be accessed from a smartphone or a **Raspberry Pi**. The database contains two main tables: the table that contains the control variables of the actuators and sensors for transfer of control between the app and automation program; and the other one contains the processed research data (the crop data) for the automation program's reference.

### 6.b. App Setup:

The smartphone application is the next component of the system. To create our own personalized software, we decided to use the MIT App Inventor. Using a Google account, we can quickly develop an app on the website. We can now design our UI and all of its modules using the app. We build a component to connect the app to the Firebase database, and we change the values for "FirebaseToken" and "FirebaseURL" to the values we noted earlier. The app allows the user to monitor the garden's atmospheric conditions round the clock (should he choose to). He/She could also choose to switch control of the actuators between himself and the automation program. The user can turn off the raspberry pi micro-controller from the app whenever he wants.

### 6.c. Building and Exporting the app:

We can now download and install the application. The software is compiled into an APK file that may be installed on any Android device if you select the build option. The file is transferred to the mobile device, and the application is subsequently installed.

### 6.d. Programming the Raspberry Pi:

We must flash the Raspberry Pi with the current version of "Raspbian for Robots" because we are utilizing GrovePi+ shield as an add-on for the Raspberry Pi. After that, we'll add another Python library to the mix. The essential libraries for requests and firebase are installed in the terminal. The code for the Raspberry Pi is then written and saved to a directory on the Raspberry Pi. Then, using the "python" command, we open the terminal and run the Python script.

The raspberry pi contains the python code whose features include:

1. 1.Constantly relaying sensor data from the sensors to the database. (Which is then relayed to the app)
2. 2.Listen to the database for changes, for instance, the python code constantly checks the motor state variable for change by the user. When the user changes the variable, the raspberry pi relays the command to the motor.
3. 3.The last and the most important feature of the python code is the automation program, which is a function that is triggered when the user relinquishes control of the garden network to the code.
4. 4.This program accounts for all variables in the network (ideal conditions for the crop type and current atmospheric conditions) and uses the actuators to create the ideal environment for the plants in the garden.

#### **6.e. Running the app:**

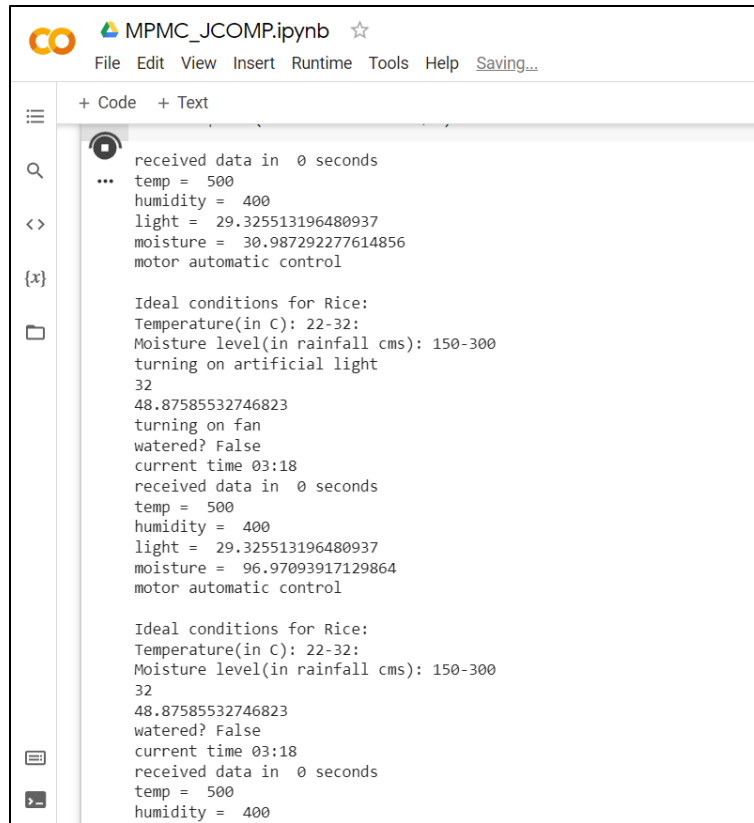
Our app's user interface is rather self-explanatory. The top four boxes show percentages of luminosity, temperature, humidity, and the moisture content of the soil in real time. These values can be modified by selecting "get values," which instructs the Raspberry Pi to update the cloud database, and then selecting "refresh," which refreshes the interface once the database has been updated. The drip irrigation system is located in the lowest area of the screen. The "on" button activates the water pump, whilst the "off" button deactivates it. The "auto" button uses the different sensor values to compute the exact amount of water required on a daily basis, watering the plants twice daily at 8 a.m. and 4 p.m.

## **7. Challenges Faced and Design Constraints:**

The primary challenge with our project was that minor undesirable changes in the environment of the crops can have an adverse effect. Extreme weather conditions have also not been taken into consideration. To implement the system in a large scale, power would need to be supplied constantly and also the devices would need a network connection to be able to connect to the server. This is a major constraint as in many rural areas electricity supply is erratic and data/network coverage is virtually non-existent. Also to run the system efficiently a constant supply of water is necessary which may not be available to all areas and especially in drought hit areas.

A major problem we faced was in the implementation of our system as we were not able to get all of the required components. We were unable to obtain a Raspberry Pi3 Model B which we required for our proposed model. This was due to the global chip shortage caused by the Covid 19 pandemic as well as the floods in Chennai from where we tried to get it delivered.

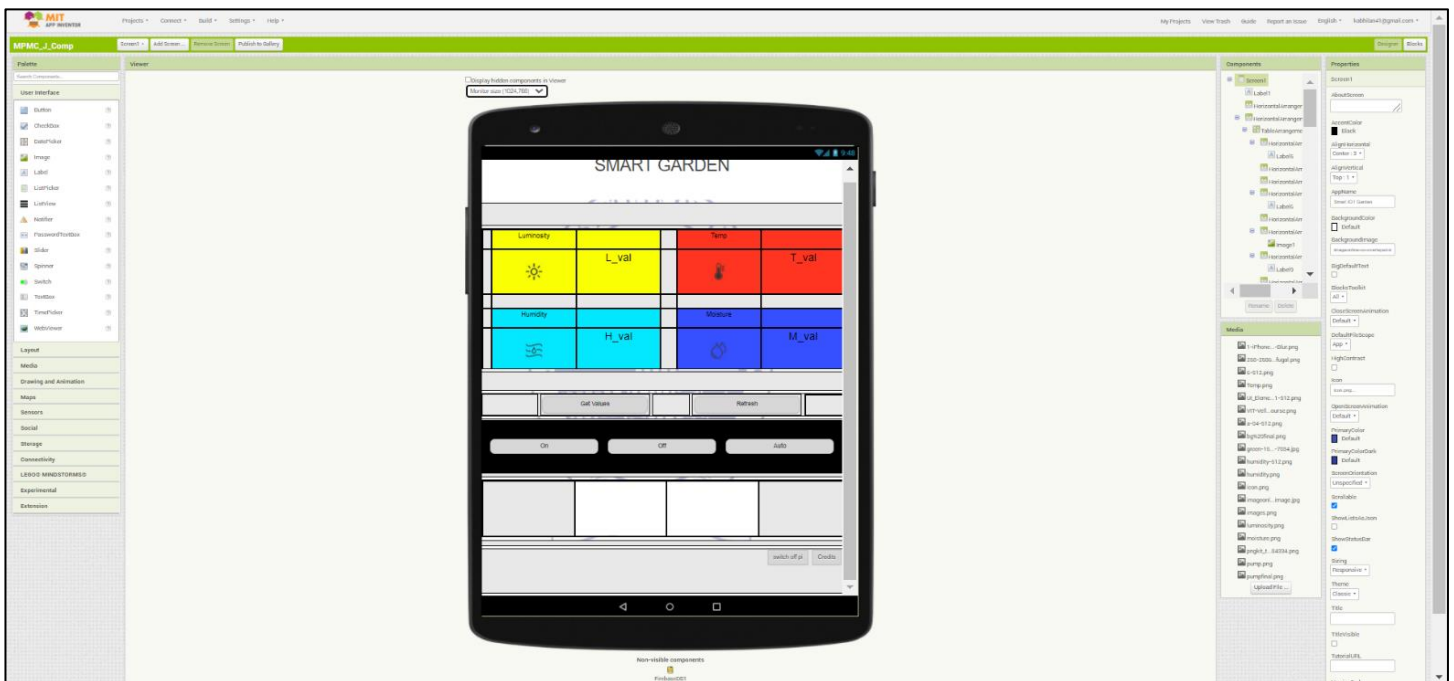
## 8. Screenshots:

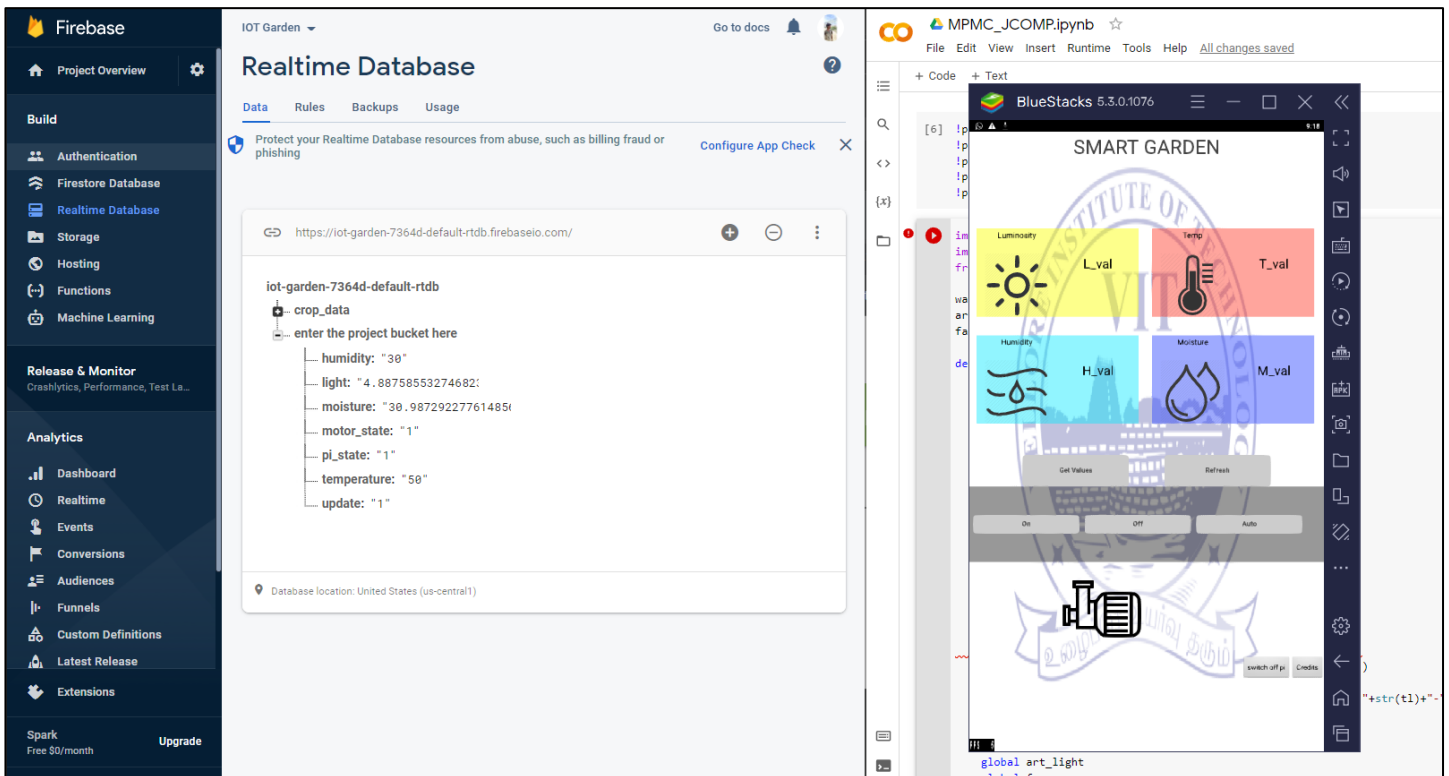
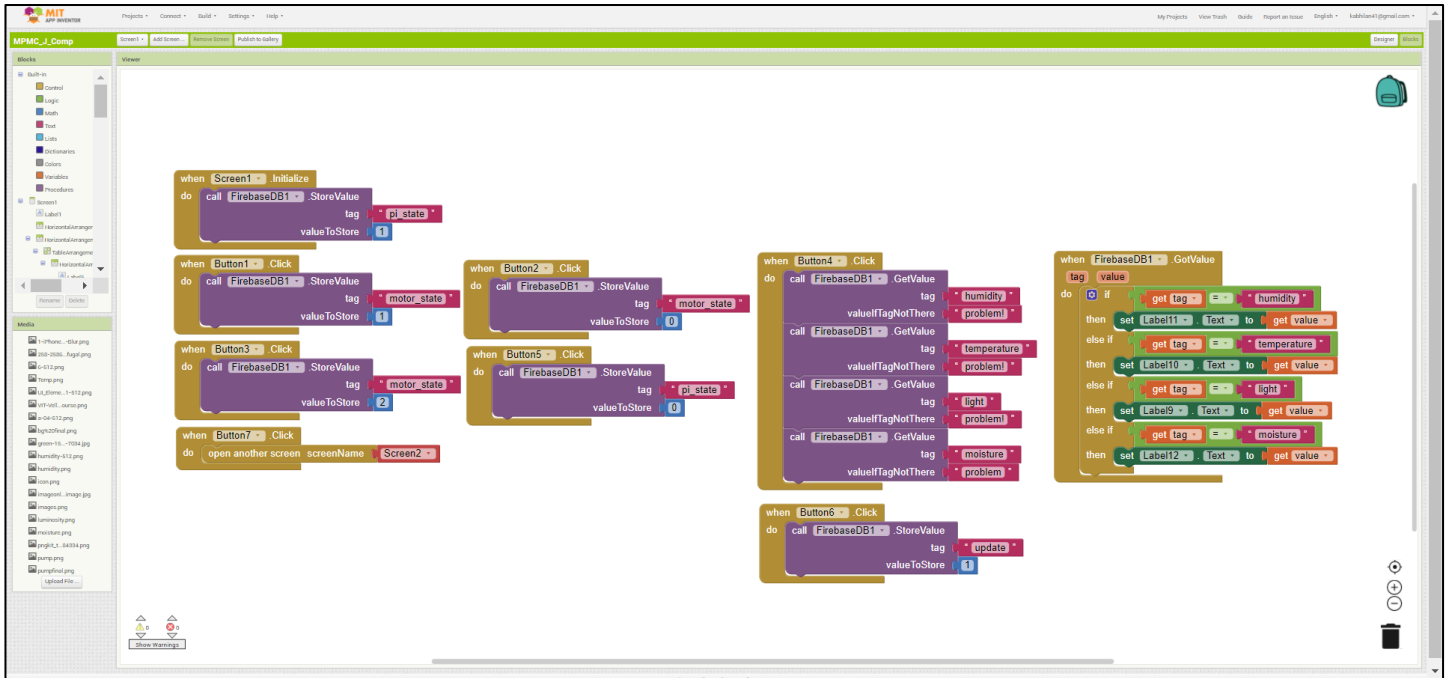


```
received data in 0 seconds
...
temp = 500
humidity = 400
light = 29.325513196480937
moisture = 30.987292277614856
motor automatic control

Ideal conditions for Rice:
Temperature(in C): 22-32:
Moisture level(in rainfall cms): 150-300
turning on artificial light
32
48.87585532746823
turning on fan
watered? False
current time 03:18
received data in 0 seconds
temp = 500
humidity = 400
light = 29.325513196480937
moisture = 96.97093917129864
motor automatic control

Ideal conditions for Rice:
Temperature(in C): 22-32:
Moisture level(in rainfall cms): 150-300
32
48.87585532746823
watered? False
current time 03:18
received data in 0 seconds
temp = 500
humidity = 400
```





## **9. Results:**

Thus, from the implementation we can see that the IOT Smart Garden System is able to efficiently manage the crops and plants automatically for a wide range of crops needing different requirements. Due to the cost-efficient nature of our system, it can be easily affordable by farmers to implement the system on a large scale. The system also saves the cost of manual labour which can thus be made use of elsewhere for further profit. Further, our system is precise and thus reduces wastage, i.e., it is economical.

## **10. Conclusion:**

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 this project was useful and helps to people to make their own automated garden.

## 11. References:

1. Ali, M., Kanwal, N., Hussain, A., Samiullah, F., Iftikhar, A., & Qamar, M. (2020). 'IoT Based Smart Garden Monitoring System using NodeMCU Microcontroller'. *Int. J. Adv. Appl. Sci*, 7(8), 117-124
2. Bhadra, Mayuri & Chakraborty, Niloy & Mukherjee, Adrika. (2020). 'Smart Gardening: A solution to your gardening issues'. 10.13140/RG.2.2.25898.57282.
3. Sunehra, D. (2019). 'Web based smart irrigation system using raspberry pi'. *International Journal of Advanced Research in Engineering and Technology*, 10(2).
4. Sambath, M., Prasant, M., Raghava, N. B., & Jagadeesh, S. (2019, November). 'Iot Based Garden Monitoring System'. In *Journal of Physics: Conference Series* (Vol. 1362, No. 1, p. 012069). IOP Publishing.
5. Jariyayothin, P., Jeravong-aram, K., Ratanachaijaroen, N., Tantidham, T., & Intakot, P. (2018, July). 'IoT Backyard: Smart watering control system'. In *2018 Seventh ICT International Student Project Conference (ICT-ISPC)* (pp. 1-6). IEEE.
6. Thamaraimanalan, T., Vivekk, S. P., Satheeshkumar, G., & Saravanan, P. (2018). 'Smart garden monitoring system using IoT'. *Asian Journal of Applied Science and Technology (AJAST)*, 2(2), 186-192.
7. Sahu, T., & Verma, A. (2017). 'Automated smart irrigation system using Raspberry Pi'. *International Journal of computer applications*, 172(6), 9-14.
8. Suma, N., Samson, S. R., Saranya, S., Shanmugapriya, G., & Subhashri, R. (2017). 'IOT based smart agriculture monitoring system'. *International Journal on Recent and Innovation Trends in computing and communication*, 5(2), 177-181.
9. Ishak, S. N., Abd Malik, N. N. N., Latiff, N. A., Ghazali, N. E., & Baharudin, M. A. (2017, November). 'Smart home garden irrigation system using Raspberry Pi'. In *2017 IEEE 13th Malaysia International Conference on Communications (MICC)* (pp. 101-106). IEEE.
10. Vineet Biswal, Hari M. Singh, W. Jeberson, Anchit S. Dhar (2015), 'Greeves: A Smart Houseplant Watering and Monitoring System', *International Journal of Science, Engineering and Technology Research (IJSETR)*, Volume 4, Issue 7, July 2015 ISSN: 2278 – 7798, 2015



## 12. Appendix (Code):

### Python Code:

```
import time

import datetime

from firebase import firebase


watered = False

art_light=False

fan=False


def automate():

    thresh = 30

    lthresh=30

    fthresh=50

    name="Rice"

    tl=0

    th=0

    ml=0

    mh=0

    moisture = 706

    temp=500

    humidity=400

    light=300

    for i in range(0,17):
```

```

namec=firebase.get('crop_data/'+str(i) , 'name')

if namec==name:

    tl=int(firebase.get('crop_data/'+str(i) , 'temp_low'))

    th=int(firebase.get('crop_data/'+str(i) , 'temp_high'))

    ml=int(firebase.get('crop_data/'+str(i) , 'moist_low'))

    mh=int(firebase.get('crop_data/'+str(i) , 'moist_high'))

    break

    print("Ideal conditions for "+name+": \nTemperature(in C): "+str(tl)+"-
"+str(th)+" : \nMoisture level(in rainfall cms): "+str(ml)+"-"+str(mh))

    light=100*light/1023

    temp=100*temp/1023

    global watered

    global art_light

    global fan

    if light<lthresh and art_light==False:

        print("turning on artificial light")

        art_light=True

    else:

        if light>lthresh:

            print("turning off artificial light")

            art_light=False

    print(th)

    print(temp)

    if temp>th and not fan:

        print("turning on fan")

        fan=True

```

```

else:

    if temp<tl:

        print("turning off fan")

        fan=False

    print("watered?", watered)

    time = datetime.datetime.now().strftime("%H:%M")

    print("current time", time)

    if (((time > "17:00") and (time < "18:00")) or ((time > "10:45") and (time <
"11:30"))):

        if not watered:

            while True:

                #read moisture sensor value using Raspberry Pi

                moisture = 100 - (100*moisture/1023)

                if moisture < ml*0.1:

                    #turn motor off

                    watered = True

                    break

            else:

                #turn motor on

                print()

        else:

            watered = False

            #turn motor off

dht_sensor = 4

light_sensor = 0

```

```
moisture_sensor = 1
```

```
motor = 3
```

```
moisture=706
```

```
#motor - output pin
```

```
#temparature and humidity sensor - input pin
```

```
#light sensor - input pin
```

```
#moisture sensor - input pin
```

```
firebase = firebase.FirebaseApplication('https://iot-garden-7364d-default-  
rtadb.firebaseio.com/', None)
```

```
initTime = time.time()
```

```
while True:
```

```
    motor_state = firebase.get('/enter the project bucket here', 'motor_state')
```

```
    update = firebase.get('/enter the project bucket here', 'update')
```

```
    pi_state = firebase.get('/enter the project bucket here', 'pi_state')
```

```
    print("received data in ",int(time.time()-initTime),"seconds")
```

```
    initTime = time.time()
```

```
    if (pi_state == "0"):
```

```
        #turn motor off
```

```
        break
```

```
    #read temperature sensor reading
```

```

#read humidity sensor reading

#read light sensor reading

#read moisture sensor reading

temp=500

humidity=400

light=300

light = 100*light/1023

moisture = 100 - (100*moisture/1023)

print("temp = ",temp)

print("humidity = ", humidity)

print("light = ", light)

print("moisture = ", moisture)

if (update == "1"):

    print("updating db")

    firebase.put('enter the project bucket here', 'temperature', str(temp))

    firebase.put('enter the project bucket here', 'humidity', str(humidity))

    firebase.put('enter the project bucket here', 'light', str(light))

    firebase.put('enter the project bucket here', 'moisture', str(moisture))

    firebase.put('enter the project bucket here', 'update', str(0))

if (motor_state == "1"):

    #turn on motor

    print("motor turned on\n")

elif (motor_state == "2"):

    #automate

    print("motor automatic control\n")

```

```
    automate()
else:
    #turn off motor
    print("motor turned off\n")
```

### **Database JSON code:**

```
{
  "crop_data" : [ {
    "moist_high" : "300",
    "moist_low" : "150",
    "name" : "Rice",
    "temp_high" : "32",
    "temp_low" : "22"
  }, {
    "moist_high" : "100",
    "moist_low" : "75",
    "name" : "Wheat",
    "temp_high" : "26",
    "temp_low" : "10"
  }, {
    "moist_high" : "100",
    "moist_low" : "50",
    "name" : "Millets",
    "temp_high" : "32",
    "temp_low" : "27"
```

```
}, {  
  "moist_high" : "45",  
  "moist_low" : "40",  
  "name" : "Grams",  
  "temp_high" : "25",  
  "temp_low" : "20"  
}, {  
  "moist_high" : "150",  
  "moist_low" : "75",  
  "name" : "Sugar Cane",  
  "temp_high" : "27",  
  "temp_low" : "21"  
}, {  
  "moist_high" : "100",  
  "moist_low" : "50",  
  "name" : "Cotton",  
  "temp_high" : "30",  
  "temp_low" : "21"  
}, {  
  "moist_high" : "75",  
  "moist_low" : "50",  
  "name" : "Oilseeds",  
  "temp_high" : "30",  
  "temp_low" : "20"  
}, {
```

```

    "moist_high" : "300",
    "moist_low" : "150",
    "name" : "Tea",
    "temp_high" : "30",
    "temp_low" : "20"
  }, {
    "moist_high" : "250",
    "moist_low" : "150",
    "name" : "Coffee",
    "temp_high" : "28",
    "temp_low" : "15"
  }, {
    "moist_high" : "220",
    "moist_low" : "120",
    "name" : "Banana",
    "temp_high" : "30",
    "temp_low" : "20"
  }, {
    "moist_high" : "50",
    "moist_low" : "30",
    "name" : "Beans",
    "temp_high" : "27",
    "temp_low" : "10"
  }, {
    "moist_high" : "50",

```



```
"moist_low" : "35",  
"name" : "Cabbage",  
"temp_high" : "21",  
"temp_low" : "15"  
}, {  
  "moist_high" : "60",  
  "moist_low" : "40",  
  "name" : "Melon",  
  "temp_high" : "35",  
  "temp_low" : "18"  
}, {  
  "moist_high" : "55",  
  "moist_low" : "35",  
  "name" : "Onion",  
  "temp_high" : "25",  
  "temp_low" : "20"  
}, {  
  "moist_high" : "70",  
  "moist_low" : "50",  
  "name" : "Potato",  
  "temp_high" : "20",  
  "temp_low" : "15"  
}, {  
  "moist_high" : "50",  
  "moist_low" : "35",
```

```

    "name" : "Peas",
    "temp_high" : "18",
    "temp_low" : "10"
  }, {
    "moist_high" : "80",
    "moist_low" : "40",
    "name" : "Tomato",
    "temp_high" : "32",
    "temp_low" : "15"
  } ],
  "enter the project bucket here" : {
    "humidity" : "30",
    "light" : "4.887585532746823",
    "moisture" : "30.987292277614856",
    "motor_state" : "2",
    "pi_state" : "1",
    "temperature" : "50",
    "update" : "0"
  }
}

```