INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

PROJECT TITLE –

WEB INTEGRATED IOT BASED AUTOMATED IRRIGATION AND CONTROLLING SYSTEM

Mentor – Dr. Sanjai Singh

Group Member –

1. Amey Rajul Shamkuwar	IEC2017084
2. Rakesh Kumar	IEC2017032
3. Pradumna Shivanand Tamkute	IEC2017059

Abstract

With the advent of humankind, there had been several problems faced by all humankind in this ultimate transformation. The basis of humankind is on availability of clean and drinkable water. With the rise of population and demand for clean water increasing rapidly and the availability of water is decreasing at an alarming rate. Agriculture sector is a major part of Indian Household for centuries and is one of the key areas where demand for water is very high. With alarming decline in water resources, the agriculture area is facing a major setback. With the proposed system the severity of the situation is being taken into consideration. This system comprises Smart solution for Farming and Irrigation for better management of water and ultimately electricity through the use IOT based automation. The system will alert the farmer about the previous water given to crops along with real time analysis of available water in the tank and the need to plant through server. It will store all the information on server which can later be accessed by providing credentials and farmer can take some steps to ensure proper growth of plants.

I. Introduction

Agriculture sector is the backbone of Indian economy with a whopping share of 15.87% .[1][2] This sector is getting some major setbacks in recent times due to various factors. With the rising cost of investment in various crop related primary necessities and low income every year farmers are getting frustrated. One of the primary reasons for this is the rising surplus of the electricity charges for irrigating the crops at timely manner.

Water crisis is major concern for the mankind at present. According to National Institution for Transforming India, a government think tank, which released a report recently showing that, without drastic action, water demand in the country will exceed supply by 2030.[3] So it’s natural that farmers are already facing a scarcity of water. Also, rainwater, which is a major source of water for centuries has shown drastic decline over the years. Many parts of the countries are facing drought like situation due to this.

With so much at stake and so little options available farmers are forced to totally or for the most part depend on ground water. This water is pumped by electric motors which requires huge amount of electricity. To reduce the load on the electricity distribution system, electricity is cut for the major part of the day and is available for little time at odd hours. Farmers are pushed to visit farms at night which sometimes proves hazardous.

With the advent of IOT technology farming can see major upsurge in its production along with becoming major profitable section through cutting costs on various extra resources which are being wasted due to human error and negligence. It is being estimated that Smart farming and IOT- driven agriculture can pave the way for Third green revolution. This will also help in able to produce enough food to rapidly rising population of world, which is being estimated to reach 9.7 Billion by the end of 2050 and whopping 11 Billion by the end of 21 Century.[4]

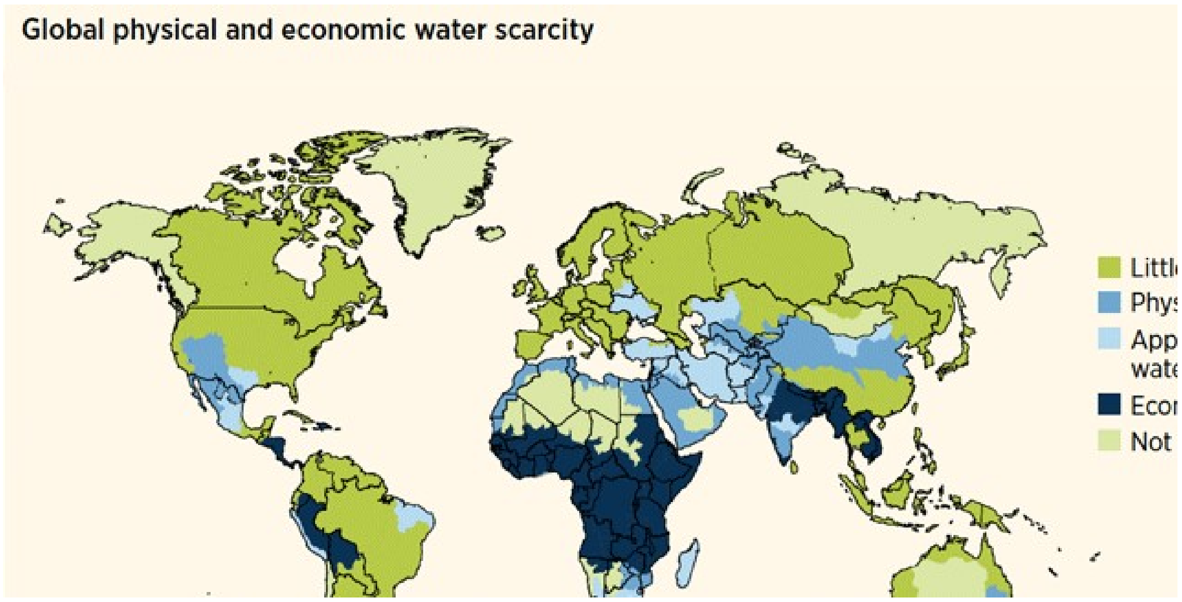


Fig 1 – Global physical and economic water scarcity

Image source – United Nations – Water for life decade (https://www.un.org/waterforlifedecade/scarcity.shtml)

To overcome this problem and this smart irrigation system is developed which will undertake the following objectives:

Objectives

- Conservation of water and electricity
- Automation of irrigation system
- Accessibility of farm system through web from anywhere
- To provide optimum water irrigation to crops
- Developing web based and mobile based app to control the system

Advantages of Smart Irrigation system

- Water wastage can be minimised since adequate water will provide at timely manner no extra water will be wasted due human error or negligence.
- Cost of electricity bills can be cut down since water pumping requires huge amount of electricity which can become payload in overall cost of farming
- The system is compact and can be cost effective once taken into account overall advantages and reduction of electricity charges.

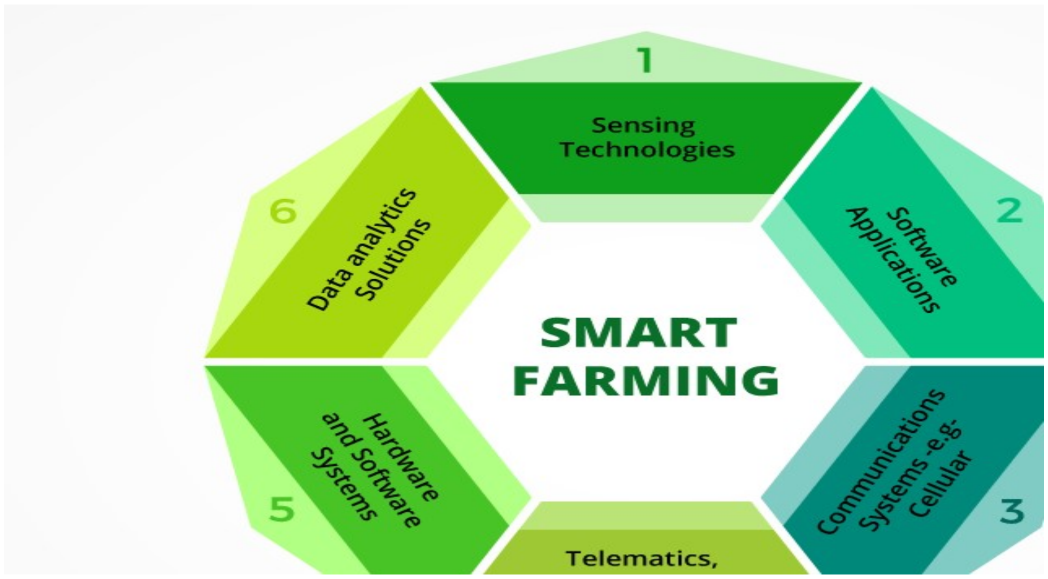


Fig 2– Use of Smart farming in agriculture
Image source – Beecham Research
(https://medium.com/sciforce/smart-farming-or-the-future-of-agriculture-359f0089df69)

II. Working principle

This model utilises the ability of Raspberry pi, which a microcontroller, to compute various conditions for the irrigation. The input to the Raspberry pi system is from three sensors namely – Soil Moisture Sensor, Temperature and Humidity sensor and Ultrasonic Sensor. Soil moisture sensor works on the principle of resistance change in the soil and gives output for moisture content of the soil. Temperature and Humidity sensor works on the principle of temperature variation of resistance. Ultrasonic sensor measures the distance between objects and works on principle of reflection of Ultrasonic waves and speed of the waves.

The model works in real time and it gets reading of moisture contents of the soil and checks with a threshold value based on the crop which is being taken in the form. If the moisture is low then watering is done. Also the temperature is check so that on very hot day extra water can be provided in order to ensure that plants get enough water considering the evaporation of water due to heat. Ultrasonic sensor checks the water level in the tank to ensure proper water level is always maintained for the proper irrigation of fields.

The web portal can be accessed from anywhere to check the current status of the farm and also to analyse the water given in the past which is being stored on a separate independent sever. This information can also be used by agricultural department for the analysis of water and crop management in a particular area.

The mobile based application can be used to check real time status of farm and also some special functions can be controlled by human intervention such as not to give water on a particular day if weather forecast is suggesting that rainfall might occur. It will also be able to give past analysis and condition of crop on any particular time and the amount of water being given to crop based on the crop requirement.

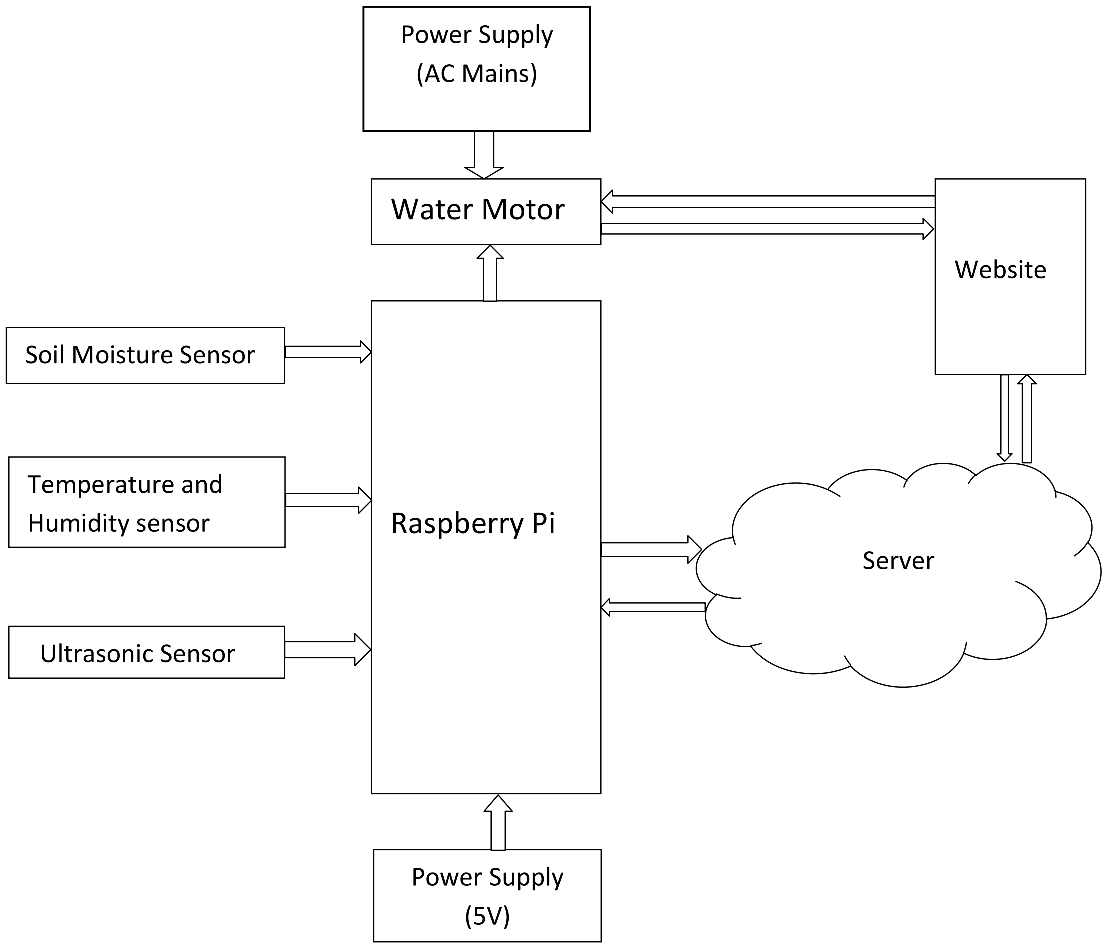


Fig.3 – Proposed system

III. Hardware description

1) Raspberry Pi 3 – Model

The Raspberry Pi 3 Model B is an upgraded ARMv7 multi core processor and Gigabytes of RAM, this seems to bea pocket computer and able to moved from being a 'toy computer' to a real desktop PC requirements. The figure:2 shows the circuit board diagram of raspberry Pi. The big upgrade is a move from the BCM2836 (single core ARMv6) to BCM2837 (quad core ARMv7). The Processor speed increases 2 times that is equal to multi core processors. By efficiently using architecture the speed may increases from 4 to 7.5 times. This processor improves the quantity performance web browsing and game playing. The Pi 3 will run the all other daughter boards at 99 % efficiency.
Pi 3 works on ‘sudo apt-upgrade’. Pi 3 is a quad core 64-bit CPU and on-board WiFi and Bluetooth. The RAM remains 1GB and there is no change to the USB or Ethernet ports. However, the upgraded power management should mean the Pi 3 can make use of more power hungry USB devices .

2) Soil Moisture Sensor

Precision soil moisture consists two probes that are inserted in to soil. When the current pass through the probes, the soil contains low moisture offer a less resistance and passes high current. That is variable resistance is the parameter to identify the level of soil moisture.

3)* Temperature sensor (LM35*)

The LM35 sensor series are precision integrated circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature.

4) RELAY

A relay is used to control the A.C motors from the controlled DC signal. It can isolate one operated electrical circuit to another. The principle behind the electromagnet operates the close and opens the circuit. Relays are applied wide area electronics circuits such as industrial control circuits a high power amplifiers, telephone exchanges etc.

5) **HC-SR04 Ultrasonic sensor**

This sensor uses sonar and capable to determine the distance of object, which is not easily affected by sunlight. It is also packaged with a transmitter and a receiver.

IV. Software description

1) Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

2) Python

Python is a widely used high-level, general purpose programming language which can work quickly and integrate systems more effectively. It is an open source language which is easy to read the syntaxes so everyone can understand it easily. Also python supports multiple programming prototypes including object-oriented, imperative and functional programming styles.

3) HTML

HTML is a standard markup language used for creating web pages and web applications. In system it is used for displaying final result on web page so that it can monitored from any place in world.

4) Adafruit

Arduino Libraries. Libraries are files written in C or C++ (.c, .cpp) which provide your sketches with extra functionality (e.g. the ability to control an LED matrix, or read an encoder, etc.). ... To install your own library, create a folder inside ARDUINO/hardware/libraries with the name of your library.

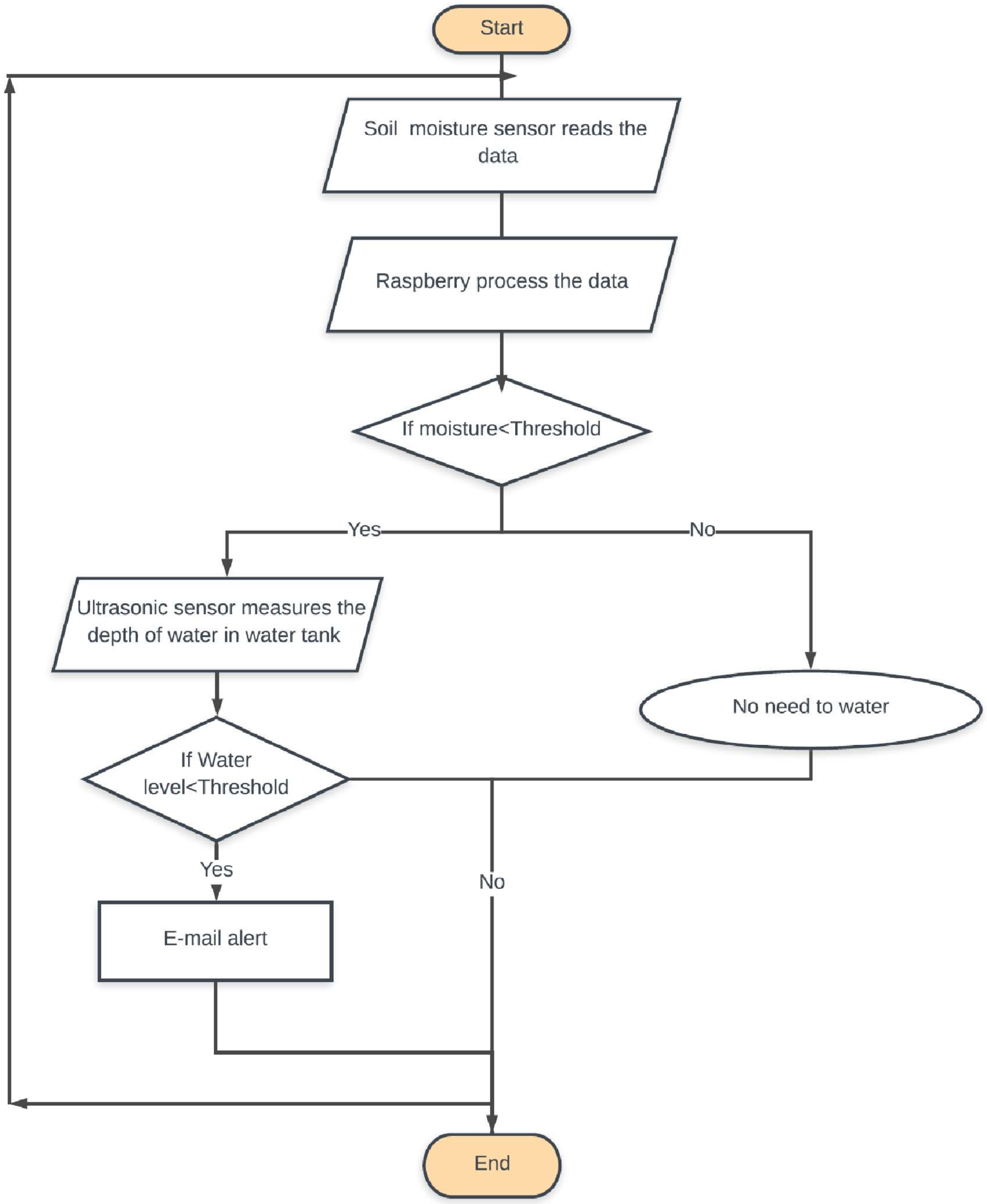
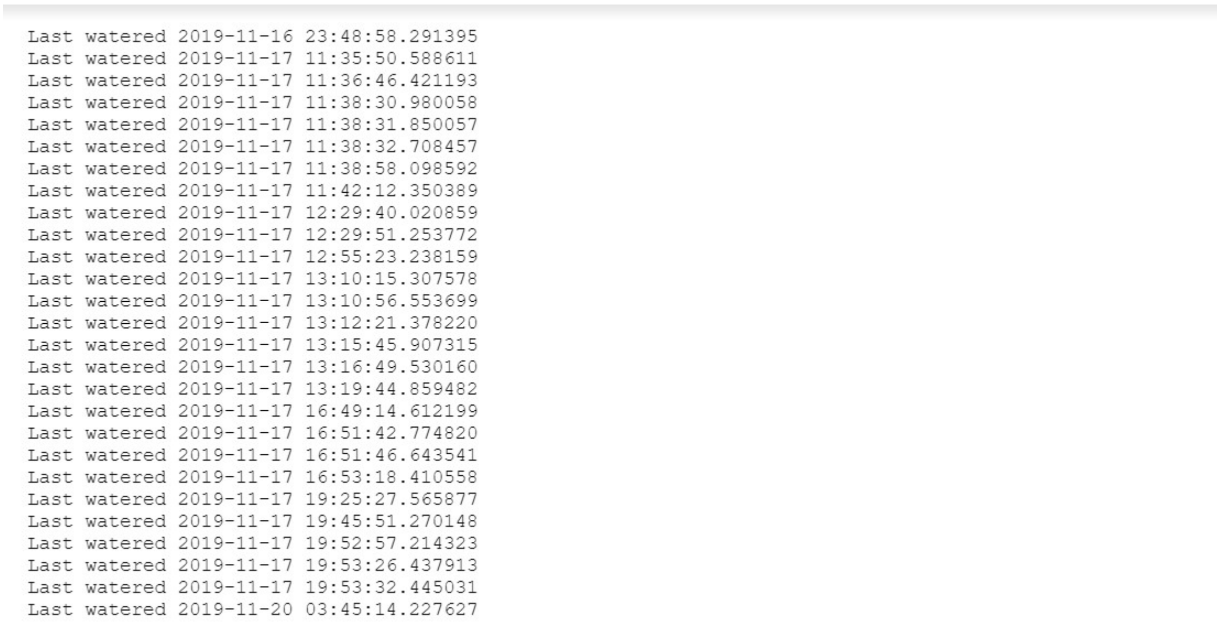

Start
Soil moisture sensor reads the data
Raspberry process the data
If moisture<Threshold
Yes
No
Ultrasonic sensor measures the depth of water in water tank
No need to water
If Water level<Threshold
Yes
No
E-mail alert
End

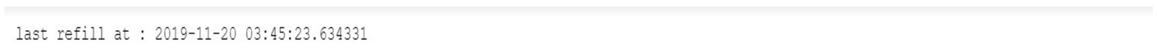
Fig 4 – Flow Chart of the system

V. Hardware implementation and results



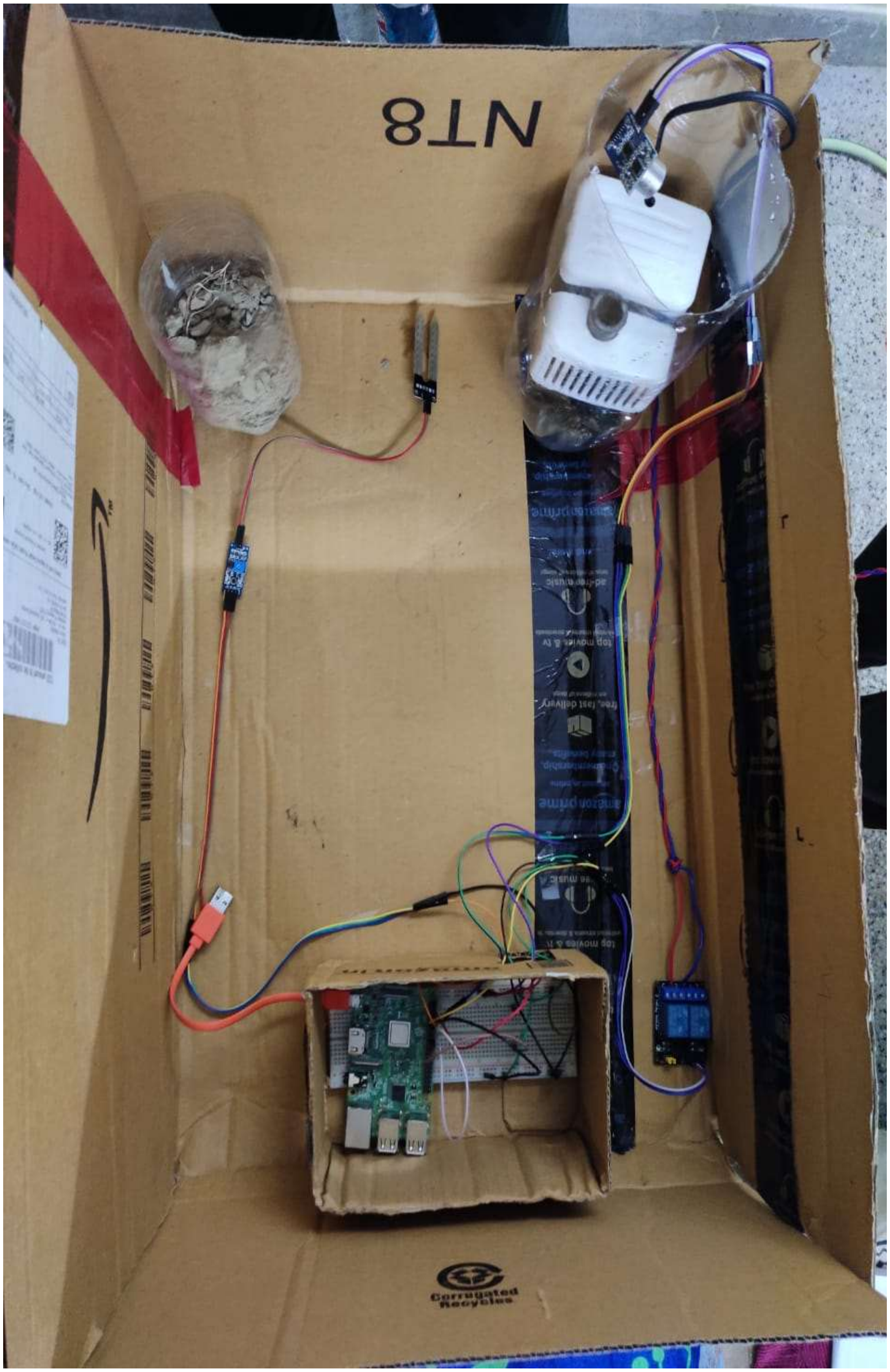
```
Last watered 2019-11-16 23:48:58.291395
Last watered 2019-11-17 11:35:50.588611
Last watered 2019-11-17 11:36:46.421193
Last watered 2019-11-17 11:38:30.980058
Last watered 2019-11-17 11:38:31.850057
Last watered 2019-11-17 11:38:32.708457
Last watered 2019-11-17 11:38:58.098592
Last watered 2019-11-17 11:42:12.350389
Last watered 2019-11-17 12:29:40.020859
Last watered 2019-11-17 12:29:51.253772
Last watered 2019-11-17 12:55:23.238159
Last watered 2019-11-17 13:10:15.307578
Last watered 2019-11-17 13:10:56.553699
Last watered 2019-11-17 13:12:21.378220
Last watered 2019-11-17 13:15:45.907315
Last watered 2019-11-17 13:16:49.530160
Last watered 2019-11-17 13:19:44.859482
Last watered 2019-11-17 16:49:14.612199
Last watered 2019-11-17 16:51:42.774820
Last watered 2019-11-17 16:51:46.643541
Last watered 2019-11-17 16:53:18.410558
Last watered 2019-11-17 19:25:27.565877
Last watered 2019-11-17 19:45:51.270148
Last watered 2019-11-17 19:52:57.214323
Last watered 2019-11-17 19:53:26.437913
Last watered 2019-11-17 19:53:32.445031
Last watered 2019-11-20 03:45:14.227627
```

Screenshot from Last_watered.txt



```
last refill at : 2019-11-20 03:45:23.634331
```

Screenshot from last_refilled.txt



Hardware Model Implementation

VI. Working codes

1. **Web app template front end file**

```
<!DOCTYPE html>
<head>
  <title>{{ title }}</title>
</head>

<body>
  <h1>PLANT HELPLINE</h1>
  <h2>The date and time on the server is: {{ time }}</h2>

  <a href="/auto/water/ON"><button>Turn ON Auto Watering</button></a>
  <a href="/auto/water/OFF"><button>Turn OFF Auto Watering</button></a>
  <a href="/sensor"><button>Check Soil Status</button></a> <br>
  <a href="/water"><button>Water Once</button></a>
  <a href="/last_watered"><button>Check Time Last Watered</button></a>
  <a href="/dht"><button>Check Humidity and Hemperature</button></a>
  <a href="/mail"><button>Send Data to Researcher</button></a>
  <h2> {{ text }} </h2>
</body>
</html>
```

2. **Web app backend file using flask**

```
from flask import Flask, render_template, redirect, url_for
import psutil
import datetime
import water
import os
import stm

app = Flask(__name__)

def template(title = "HELLO!", text = ""):
    now = datetime.datetime.now()
    timeString = now
    templateDate = {
        'title' : title,
        'time' : timeString,
        'text' : text
        }
    return templateDate

@app.route("/")
def hello():
    templateData = template()
    return render_template('main.html', **templateData)

@app.route("/last_watered")
def check_last_watered():

    te=water.get_last_watered()
    te=te
    templateData = template(text=te)

    return render_template('main.html', **templateData)

@app.route("/sensor")
def action():
    status = water.get_status()
    message = ""
    if (status == 1):
        message = "Water me please!"
    else:
        message = "I'm a happy plant"

    templateData = template(text = message)
    return render_template('main.html', **templateData)

@app.route("/water")
def action2():
    water.pump_on()
    templateData = template(text = "Watered Once")
    return render_template('main.html', **templateData)

@app.route("/auto/water/<toggle>")
def auto_water(toggle):
    running = False
    if toggle == "ON":
        templateData = template(text = "Auto Watering On")
        for process in psutil.process_iter():
            try:
                if process.cmdline()[1] == 'auto_water.py':
                    templateData = template(text = "Already running")
                    running = True
            except:
                pass
        if not running:
            os.system("python3 auto_water.py&")
    else:
        templateData = template(text = "Auto Watering Off")
        os.system("pkill -f water.py")

    return render_template('main.html', **templateData)

@app.route("/dht")
def dht11():
    dd= water.humid()
    templateData = template(text=dd)
    return render_template('main.html', **templateData)

@app.route("/mail")
def mailme():
    #call function for all text files
    stm.email_data("humidity.txt")
    stm.email_data("last_refill.txt")
    stm.email_data("last_watered.txt")
    templateData = template(text="we mailed the samples")
    return render_template('main.html', **templateData)



if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)
```

3. Hardware implementation code on Raspberry pi

```
# External module imp
import RPi.GPIO as GPIO
import datetime
import time
#!/usr/bin/python
import sys
import ultra
import refill
import Adafruit_DHT

init = False

GPIO.setmode(GPIO.BOARD) # Broadcom pin-numbering scheme

def get_last_watered():
    try:
        f = open("last_watered.txt", "r")
        return f.readline()
    except:
        return "NEVER!"

def get_status(pin = 8):
    GPIO.setup(pin, GPIO.IN)
    return GPIO.input(pin)

def init_output(pin):
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, GPIO.LOW)
    GPIO.output(pin, GPIO.HIGH)

def auto_water(delay = 5, pump_pin = 7, water_sensor_pin = 8):
    consecutive_water_count = 0
    init_output(pump_pin)
    print("Here we go! Press CTRL+C to exit")
    try:
        while 1 and consecutive_water_count < 10:
            time.sleep(delay)
            wet = get_status(pin = water_sensor_pin) == 0
            if not wet:
                if consecutive_water_count < 5:
                    pump_on(pump_pin, 1)
                consecutive_water_count += 1
            else:
```

```
            consecutive_water_count = 0
    except KeyboardInterrupt: # If CTRL+C is pressed, exit cleanly:
        GPIO.cleanup() # cleanup all GPI

def pump_on(pump_pin = 7, delay = 1):
    init_output(pump_pin)

    water_level = ultra.dis()
    if water_level <10:
        refill.send()
        f=open("last_refill.txt","a+")
        f.write("last refill at : {}\n".format(datetime.datetime.now()))
        f.close()
        return False
    else:
        f = open("last_watered.txt", "a+")
        f.write("Last watered {}".format(datetime.datetime.now()))
        f.write("\n")
        f.close()
        GPIO.output(pump_pin, GPIO.LOW)
        time.sleep(1)
        GPIO.output(pump_pin, GPIO.HIGH)
        return True




def humid(dht_pin=3):
    dht_sensor = Adafruit_DHT.DHT11
    humidity, temperature = Adafruit_DHT.read_retry(dht_sensor, dht_pin)
    f = open("humidity.txt", "a+")

    if humidity is not None and temperature is not None:
        #f.open("humidity.txt","a+")
        f.write("Temp={0:0.1f}*  Humidity={1:0.1f}%".format(temperature, humidity))
        f.write("\n")
        f.close()
    else:
        f.write("Failed to get reading. Try again!")
        f.write("\n")
        f.close()
    try:
        f = open("humidity.txt", "r")
        return f.readline()
    except:
        return "NEVER!"
```

4. <u>Auto watering system code</u>

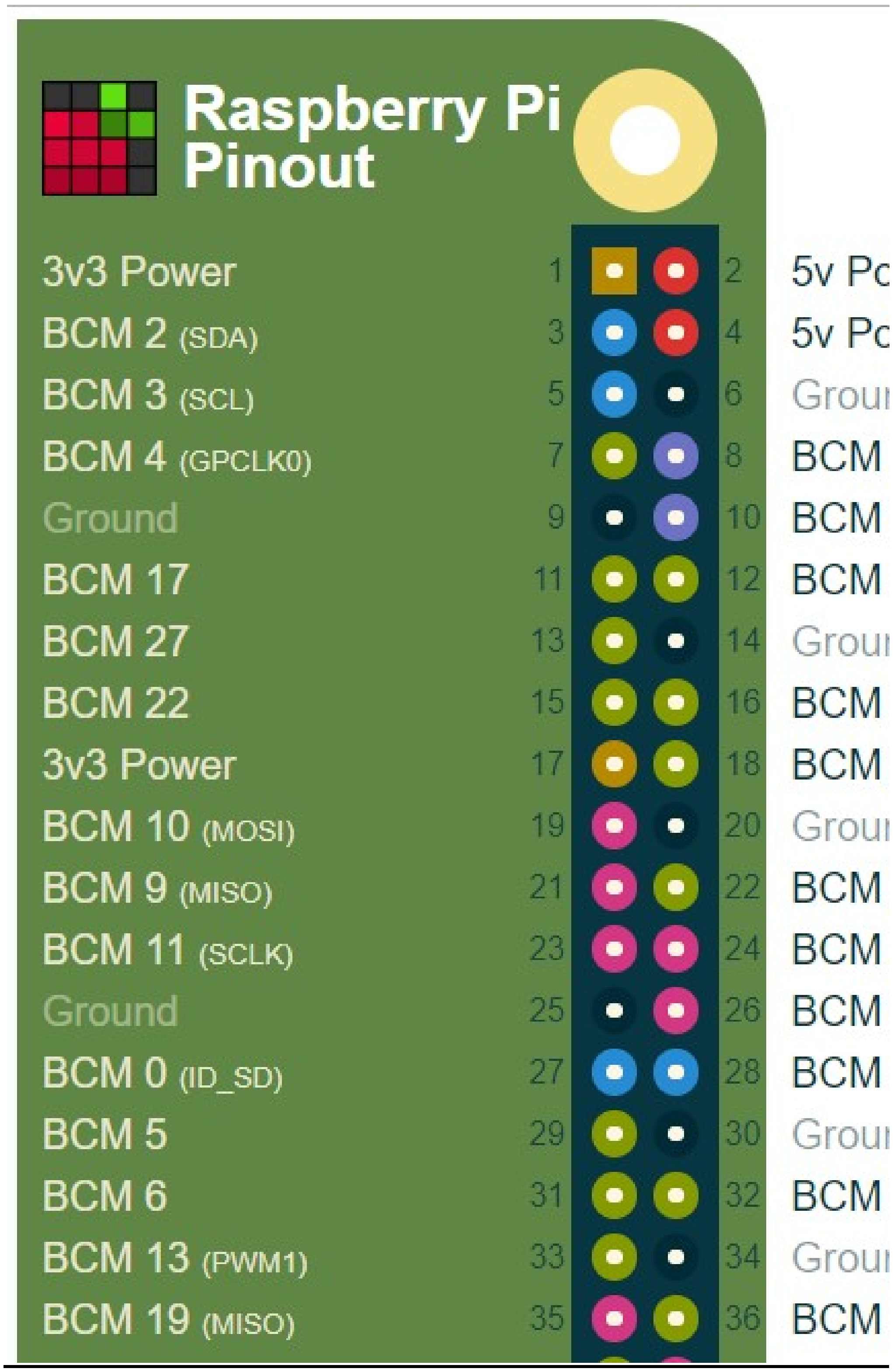
```
import water

if __name__ == "__main__":
    water.auto_water()
```

V. References

1. http://statisticstimes.com/economy/sectorwise-gdp-contribution-of-india.php
2. Ministry of Statistics and Programme Implementation (2018-2019) - http://mospi.nic.in/sites/default/files/press_release/Presss%20note%20for%20first%20advance%20estimates%202018-19.pdf
3. https://theweek.com/articles/850956/major-cities-india-are-starting-run-water
4. Based on prediction by United Nations at slow rate. https://www.un.org/development/desa/en/news/population/world-population-prospects-2019.html
5. Mobile integrated smart irrigation management and monitoring system using IOT – IEEE Xplore
   2011 - S. Vaishali ; S. Suraj ; G. Vignesh ; S. Dhivya ; S. Udhayakumar
6. Smart irrigation with embedded system - IEEE Xplore 2017 K KNamala ; Krishna Kanth Prabhu A V ; Anushree Math ; AshwiniKumari ; Supraja Kulkarni
7. Smart irrigation: A smart drip irrigation system using cloud, android and data mining – IEEE Xplore 2017 Subhashree Ghosh ; SumaiyaSayyed ; KanchanWani ; MrunalMhatre ; Hyder Ali Hingoliwala
8. IoT based smart irrigation monitoring and controlling system – IEEE Xplore 2018 Shweta B. Saraf ; Dhanashri H. Gawali
9. IoT based smart crop-field monitoring and automation irrigation system – IEEE Xplore 2018 R. Nageswara Rao ; B. Sridhar
10. *Smart Home Garden Irrigation System Using Raspberry Pi* – IEEE Xplore 2018 S.N. Ishak, N.N.N.Abd Malik, N.M. Abdul Latiff, N. EffiyanaGhazali, M. A. BaharudinUniversitiTeknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia
11. https://raspberrypi.stackexchange.com/questions/81635/soil-moisture-sensor-analog-output-is-0
12. http://www.piddlerintheroot.com/soil-moisture-sensor/
13. https://computers.tutsplus.com/tutorials/build-a-raspberry-pi-moisture-sensor-to-monitor-your-plants--mac-52875
14. https://www.raspberrypi.org/forums/viewtopic.php?t=207797
15. https://pinout.xyz/pinout/pin18_gpio24
16. http://mattrichardson.com/Raspberry-Pi-Flask/
17. https://stackabuse.com/read-a-file-line-by-line-in-python/
18. https://stackoverflow.com/questions/4465959/python-errno-98-address-already-in-use
19. https://www.geeksforgeeks.org/send-mail-attachment-gmail-account-using-python/
20. https://raspberrypi.stackexchange.com/questions/14105/how-does-python-gpio-bouncetime-parameter-work
21. https://sourceforge.net/p/raspberry-gpio-python/wiki/Inputs/
22. https://www.instructables.com/id/Soil-Moisture-Sensor-Raspberry-Pi/
23. https://superuser.com/questions/196166/linux-command-line-to-turn-off-proxy
24. Sangamesh Malge, Kalyani Bhole, "Novel, Low cost Remotely operated smart Irrigation system" 2015 International Conference on Industrial Instrumentation and Control (ICIC) College of Engineering Pune, India. May 28-30, 2015

Comments by the Panel



Pin diagram of Raspberry Pi 3 b