**Flood Monitoring and Early Warning System.**

**Phase-4**

Development Part 2:

Tasks:

* Installing the required libraries and configuring the components used.
* Use web development technologies to create a platform that performs real-time transit information.
* Creating alert systems using IoT Platforms like cayenne or ThingSpeak.

**Transmitting the data to the server:**

This code reads data from the ultrasonic sensor and sends it to a server (you can set up a web server to receive this data for real-world applications):

import RPi.GPIO as GPIO

import dht11

import time

import datetime

import MySQLdb

# initialize GPIO

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

# read data using pin 14

instance = dht11.DHT11(pin=14)

GPIO.setup(16,GPIO.OUT)

def THRESHOLD():

return 20

class ultraSound:

TRIG = 20

ECHO = 21

#LED=16

def \_\_init\_\_(self):

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(self.TRIG,GPIO.OUT)

GPIO.setup(self.ECHO,GPIO.IN)

#GPIO.setup(self.LED,GPIO.OUT)

def readData(self):

GPIO.output(self.TRIG, False)

#print "Waitng For Sensor To Settle"

time.sleep(2)

GPIO.output(self.TRIG, True)

time.sleep(0.00001)

GPIO.output(self.TRIG, False)

pulse\_start=0;

pulse\_end=0;

while GPIO.input(self.ECHO)==0:

pulse\_start = time.time()

while GPIO.input(self.ECHO)==1:

pulse\_end = time.time()

pulse\_duration = pulse\_end - pulse\_start

distance = pulse\_duration \* 17150

distance = round(distance, 2)

time.sleep(0.4)

return distance;

sensor=ultraSound();

while True:

remarks\_val="Nothing to Worry"

result = instance.read()

distance=sensor.readData();

dateandtime=datetime.datetime.now()

if result.is\_valid():

print("Time: " + str(datetime.datetime.now()))

print("Temperature: %-3.1f C" % result.temperature)

print("Humidity: %-3.1f %%" % result.humidity)

print "Distance:",distance,"cm"

time.sleep(3)

print"----------------------------------------------------"

if(distance<=THRESHOLD()):

GPIO.output(16,GPIO.HIGH)

print "WARNING THE LEVEL OF WATER HAS RAISED BEYOND THE THRESHOLD LEVEL"

remarks\_val="WARNING THE LEVEL OF WATER HAS RAISED BEYOND THE THRESHOLD LEVEL"

else:

GPIO.output(16,GPIO.LOW)

print"----------------------------------------------------"

conn=MySQLdb.connect(host="localhost",user="root",passwd="shrestha5555",db="minor")

cur=conn.cursor()

sql\_query="""INSERT INTO Data (DateTime,Temperature,Humidity,Distance,Remarks) VALUES (%s,%s,%s,%s,%s)"""

cur.execute(sql\_query,(dateandtime,result.temperature,result.humidity,distance,remarks\_val))

conn.commit()

conn.close()

**Ultrasonic Sensor:**

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(16,GPIO.OUT)

def THRESHOLD():

return 30

class ultraSound:

TRIG = 20

ECHO = 21

#LED=16

def \_\_init\_\_(self):

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(self.TRIG,GPIO.OUT)

GPIO.setup(self.ECHO,GPIO.IN)

#GPIO.setup(self.LED,GPIO.OUT)

def readData(self):

GPIO.output(self.TRIG, False)

#print "Waitng For Sensor To Settle"

time.sleep(2)

GPIO.output(self.TRIG, True)

time.sleep(0.00001)

GPIO.output(self.TRIG, False)

pulse\_start=0;

pulse\_end=0;

while GPIO.input(self.ECHO)==0:

pulse\_start = time.time()

while GPIO.input(self.ECHO)==1:

pulse\_end = time.time()

pulse\_duration = pulse\_end - pulse\_start

distance = pulse\_duration \* 17150

distance = round(distance, 2)

print "Distance:",distance,"cm"

time.sleep(0.4)

return distance;

sensor=ultraSound();

while True:

sensor.readData();

if(sensor.readData()<=THRESHOLD()):

GPIO.output(16,GPIO.HIGH)

print "WARNING THE LEVEL OF WATER HAS RAISED BEYOND THE THRESHOLD LEVEL"

else:

GPIO.output(16,GPIO.LOW)

**Hardware interfacing with Cayenne IoT platform**

Cayenne is an IoT project builder platform that allows you to create IoT applications and projects by connecting various sensors and devices to the Cayenne platform. Here's a step-by-step guide to configuring Cayenne for a flood management and early warning system using sensors and devices:

Step 1: Create a Cayenne Account

Go to the Cayenne website (https://cayenne.mydevices.com) and create a new account if you don't already have one.

Step 2: Add a New Device

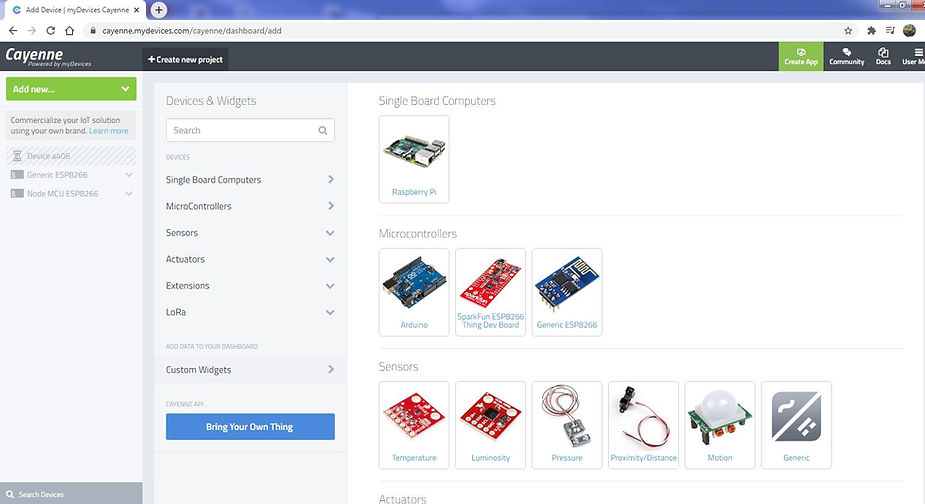
After logging in, click on the "Add New..." button and select "Device."

Add New Device here and select Generic ESP8266 for in this project.

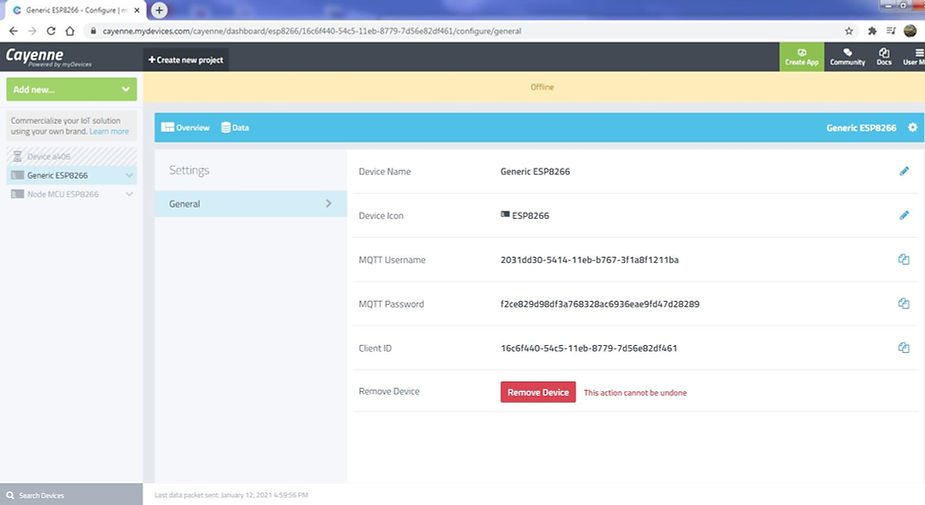
Step 3: Create a New Project

In your Cayenne dashboard, click on the "Create New Project" button.

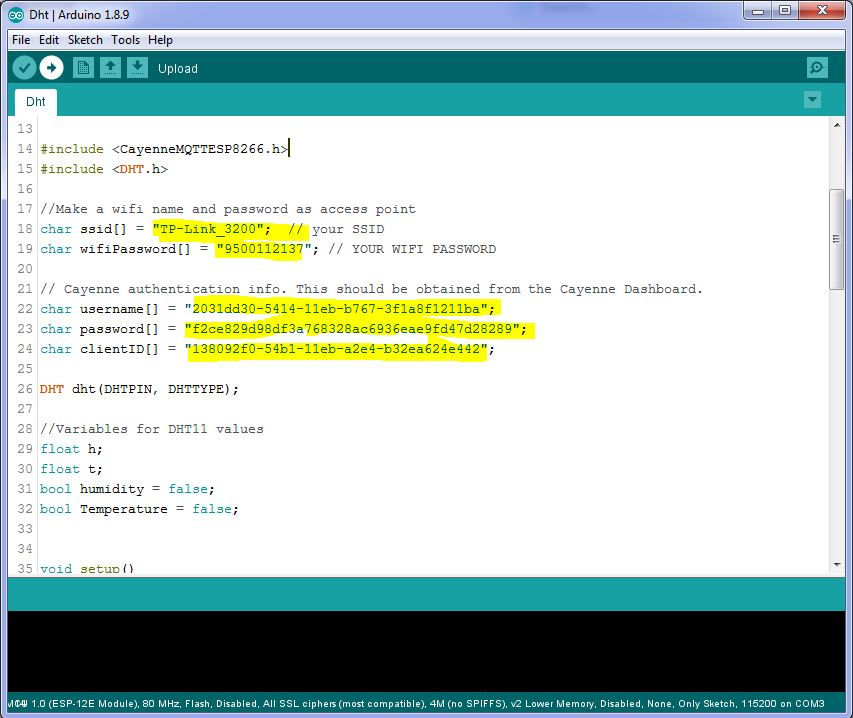
Give your project a name related to flood management and early warning.



Step 4: Configure device Generic ESP8266, MQTT username, password and client ID from Create App.



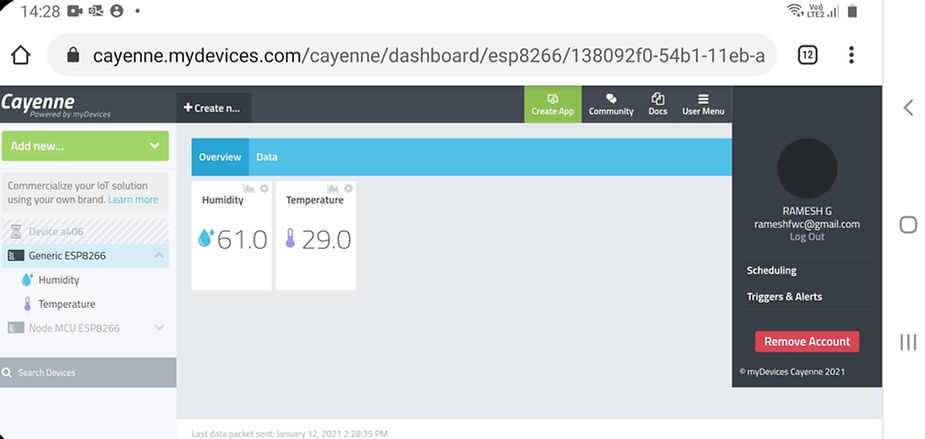
Step 5: Paste these respective details under username, password and client ID in Arduino source code, along with your Wi-Fi name and password.



After successfully compiling and uploading the code to NodeMCU, You will see ESP8266 connected to Wi-Fi. After the connection is established, the previous page is automatically updated on Cayenne. A new dashboard opens in the browser. Cayenne generates an ID and a device icon for your device.

Click on Custom Widgets and then value, and populate all fields . The channel number should be 1. (Make sure the channel number is same as in code.) Now, click on Add Widget.

When a connection is made, sensor data gets uploaded to Cayenne. Temperature and Humidity data on Cayenne.



The Data is received from the Sensors and are displayed.