FLOOD MONITORING AND EARLY WARNING

DEFINITION

  Flood is one of the major well known Natural Disasters. When water level suddenly rises in dams, river beds etc. Alot of Destruction happens at surrounding places. It causes a huge amount of loss to our environment and living beings as well. S0 in thesecase, it is very important to get emergency alerts of the water level situation in different conditions in the river bed.

The purpose of this project is to sense the water level in river beds and check if they are in normal condition. If they reach beyond the limit, then it alerts people through LED signals and buzzer sound,also it alerts people through SMS and Emails alerts when the water level reaches beyond the limit.

Software Components

1. [Arduino IDE](https://www.arduino.cc/en/software)
2. [Python 3.7 IDLE](https://www.python.org/downloads/)
3. [Bolt IoT Cloud](https://cloud.boltiot.com/)
4. [Bolt IoT Android App](https://play.google.com/store/apps/details?id=com.bolt.com.bolt)
5. [TWILLO SMS Messaging API](https://www.twilio.com/)
6. [MAILGUN EMAIL Messaging API Software components](https://www.mailgun.com/)

## Software Programming

After the successful completion of hardware setup. Now it’s the time to do software setup for the project. For that you have to first Download and Install Arduino IDE and Python IDE from the link given above in the software apps and online services section. Also Creating account on various online app services and noting down the important keys and ID’s. Below all the steps given to create account on online app services and noting down the keys.

**Step 1:**

**Creating an account on TWILLO and setting up TWILLO for sending SMS alerts.**

* Visit <https://www.twilio.com/>.
* Create account by clicking sign up, fill required details.
* Confirm your email.
* You will need to authenticate your phone number on which the SMS alerts will be notified.
* Enter the code sent to your phone
* When Prompted “Do you write code?” Click yes
* Select python as your programming language
* When prompted “What is your target today? “Choose” Twilio as a project.
* When prompted “What do you want to do first? “Choose” Send or receive a message.
* My First Twilio Project Dashboard page will open. Now you can Edit your Project as “My Project”.
* Get a trial number and save it somewhere and then choose to use this number.
* You will see the ACCOUNT SID and AUTH TOKEN.
* We will need Account Sid, Auth Token and Trial Number of these so save them somewhere

**Step 2**:

**Creating an account on MAILGUN and setting up MAILGUN for sending Email alerts.**

* Visit <https://www.mailgun.com/>.
* Create an account by clicking on the start sending option and by filling up details.
* Verifying your Account.
* Once you have verified your Email after that you have add your phone number.
* After Entering your number. Click on send activation code. After sometime you will receive one OTP. Enter the OTP. Click on Enter.
* After Creating account on MAILGUN go to the overview option. Click on API and Click on Python.
* After doing this so you will receive API Key and Sandbox URL. Save this both credentials somewhere you will be further using in this project.

**Step 3:**

**Creating an account on Bolt Cloud and Bolt Android App and Link the Bolt Module to Cloud.**

* Visit [https://cloud.boltiot.com](https://cloud.boltiot.com/).
* Create account using Email-Id and password. (Use the same email which was used to order hardware kit also use same email for app for linking the hardware to cloud.)
* After creating account on cloud. Then Download Bolt Android App from PLAYSTORE.
* Create an account on the Bolt app with the same email-Id then use the mobile hotspot for linking the Bolt Wi-Fi module to cloud.
* After successful linking of the device to the cloud then go to the cloud website. The Bolt device will show the device as online.
* Go to API section make the API as enable. Copy the API and save somewhere.
* Also copy the Bolt Device Id which is present on Bolt IoT dashboard and save it somewhere.

**Step 4:**

* After setting online app services and saving the keys somewhere. Now most important is to write code and allow sensors attached to microcontroller to take specific decisions.
* Basically this project contains two editors to write the code. First is Arduino IDE in that we will write the arduino code. Second the Python IDE in that we will write the configuration file and the main code. Also the download link of both the editor can find above in the online app services section

.

**Step 4.1:**

**Writing the code in the Arduino IDE**

* Open the Arduino IDE(Downloaded from the above section).
* Click on new file. Choose the correct file path to save the file. Give appropirate name to the file and add .ino extention to the file and save the file.
* Now the core part of the project is writing code for Arduino Uno. Below this line complete code is given. You can refer the below code.
* After writing the code. Verify the code and then upload the code to the specific Arduino using USB Cable type A. Remember while uploading select specific board you want to upload.

CODING

* //IOT Based Flood Monitoring And Alerting System.
* #include<LiquidCrystal.h>
* LiquidCrystal lcd(2, 3, 4, 5, 6, 7);
* const int in = 8;
* const int out = 9;
* const int green = 10;
* const int orange = 11;
* const int red = 12;
* const int buzz = 13;
* void setup() {
* Serial.begin(9600);
* lcd.begin(16, 2);
* pinMode( in , INPUT);
* pinMode(out, OUTPUT);
* pinMode(green, OUTPUT);
* pinMode(orange, OUTPUT);
* pinMode(red, OUTPUT);
* pinMode(buzz, OUTPUT);
* digitalWrite(green, LOW);
* digitalWrite(orange, LOW);
* digitalWrite(red, LOW);
* digitalWrite(buzz, LOW);
* lcd.setCursor(0, 0);
* lcd.print("Flood Monitoring");
* lcd.setCursor(0, 1);
* lcd.print("Alerting System");
* delay(5000);
* lcd.clear();
* }
* void loop() {
* long dur;
* long dist;
* long per;
* digitalWrite(out, LOW);
* delayMicroseconds(2);
* digitalWrite(out, HIGH);
* delayMicroseconds(10);
* digitalWrite(out, LOW);
* dur = pulseIn( in , HIGH);
* dist = (dur \* 0.034) / 2;
* per = map(dist, 10.5, 2, 0, 100);
* #map
* function is used to convert the distance into percentage.
* if(per < 0) {
* per = 0;
* }
* if (per > 100) {
* per = 100;
* }
* Serial.println(String(per));
* lcd.setCursor(0, 0);
* lcd.print("Water Level:");
* lcd.print(String(per));
* lcd.print("% ");
* if (per >= 80) #MAX Level of Water--Red Alert!{
* lcd.setCursor(0, 1);
* lcd.print("Red Alert! ");
* digitalWrite(red, HIGH);
* digitalWrite(green, LOW);
* digitalWrite(orange, LOW);
* digitalWrite(buzz, HIGH);
* delay(2000);
* digitalWrite(buzz, LOW);
* delay(2000);
* digitalWrite(buzz, HIGH);
* delay(2000);
* digitalWrite(buzz, LOW);
* delay(2000);
* }
* else if (per >= 55) #Intermedite Level of Water--Orange Alert!{
* lcd.setCursor(0, 1);
* lcd.print("Orange Alert! ");
* digitalWrite(orange, HIGH);
* digitalWrite(red, LOW);
* digitalWrite(green, LOW);
* digitalWrite(buzz, HIGH);
* delay(3000);
* digitalWrite(buzz, LOW);
* delay(3000);
* }
* else #MIN / NORMAL level of Water--Green Alert!{
* lcd.setCursor(0, 1);
* lcd.print("Green Alert! ");
* digitalWrite(green, HIGH);
* digitalWrite(orange, LOW);
* digitalWrite(red, LOW);
* digitalWrite(buzz, LOW);
* }
* delay(15000);
* }

**Step 5.2:**

**Writing the code in Python IDE.**

* For writing python code we will be using python IDE.
* In this project we will be making two python files. One will be saved in the name of conf.py and other will be main.py.
* The purpose of making two files is to make the code understandable. Also this both python files will be useful in sending sms and emails alerts to users.
* Now the most important part is arrived writing code in Python IDE. The full code is divided into two parts. The detailed code is given below.
* Open Python 3.7 IDE(Downloaded from the above section).
* Click on new file. Save the file in the name conf.py.
* **conf.py:** The file consists of important Api keys, Device id of Bolt IoT WiFi Module. Also it consists of important keys of Twillo and Mailgun respectively which will be further useful in this project.
* Below is the complete structure of conf.py file. Make sure that you add the updated Bolt API key, device id and Mailgun and Twillo details respectively:

CODING

#twillo details for sending alert sms

SID = 'You can find SID in your Twilio Dashboard'

AUTH\_TOKEN = 'You can find on your Twilio Dashboard'

FROM\_NUMBER = 'This is the no. generated by Twilio. You can find this on your Twilio Dashboard'

TO\_NUMBER = 'This is your number. Make sure you are adding +91 in beginning'

#bolt iot details

API\_KEY = 'XXXXXXXXX'

#This is your Bolt cloud API

Key.

DEVICE\_ID = 'BOLTXXXXXXXXX' #This is the ID of your Bolt device.

#mailgun details for sending alert E-mails

MAILGUN\_API\_KEY = 'This is the private API key which you can find on your Mailgun Dashboard'

SANDBOX\_URL= 'You can find this on your Mailgun Dashboard'

SENDER\_EMAIL = 'test@ + SANDBOX\_URL' # No need to modify this. The sandbox URL is of the format test@YOUR\_SANDBOX\_URL

RECIPIENT\_EMAIL = 'Enter your Email ID Here'

 After writing the conf.py now the last part is to write the main.py code. This code will be help full to send SMS and email alerts when the water level crosses the threshold.

* Open the Python IDE.
* Click on new file. Save the file in the name main.py. Save the file in the same path where conf.py is saved.
* main.py: This file consists of the main coding facility. Discussed earlier it will be used to send SMS and emails alerts. It will be also help full to keep close monitor on water level to send alerts whenever required.
* Below is the complete code of main.py.

CODING

import conf  
from boltiot import Sms, Email, Bolt  
import json, time  
intermediate\_value = 55  
max\_value = 80  
mybolt = Bolt(conf.API\_KEY, conf.DEVICE\_ID)  
sms = Sms(conf.SID, conf.AUTH\_TOKEN, conf.TO\_NUMBER, conf.FROM\_NUMBER)  
mailer = Email(conf.MAILGUN\_API\_KEY, conf.SANDBOX\_URL, conf.SENDER\_EMAIL, conf.RECIPIENT\_EMAIL)  
def twillo\_message(message):  
 try:  
 print("Making request to Twilio to send a SMS")  
 response = sms.send\_sms(message)  
 print("Response received from Twilio is: " + str(response))  
 print("Status of SMS at Twilio is :" + str(response.status))  
 except Exception as e:  
 print("Below are the details")  
 print(e)  
def mailgun\_message(head,message\_1):  
 try:  
 print("Making request to Mailgun to send an email")  
 response = mailer.send\_email(head,message\_1)  
 print("Response received from Mailgun is: " + response.text)  
 except Exception as e:  
 print("Below are the details")  
 print(e)   
while True:  
 print ("Reading Water-Level Value")  
 response\_1 = mybolt.serialRead('10')  
 response = mybolt.analogRead('A0')  
 data\_1 = json.loads(response\_1)  
 data = json.loads(response)   
 Water\_level = data\_1['value'].rstrip()  
 print("Water Level value is: " + str(Water\_level) + "%")  
 sensor\_value = int(data['value'])  
 temp = (100\*sensor\_value)/1024  
 temp\_value = round(temp,2)  
 print("Temperature is: " + str(temp\_value) + "°C")  
 try:   
 if int(Water\_level) >= intermediate\_value:  
 message ="Orange Alert!. Water level is increased by " +str(Water\_level) + "% at your place. Please be Safe. The current Temperature is " + str(temp\_value) + "°C."  
 head="Orange Alert"  
 message\_1="Water level is increased by " + str(Water\_level) + "% at your place. Please be Safe. The current Temperature is " + str(temp\_value) + "°C."  
 twillo\_message(message)  
 mailgun\_message(head,message\_1)  
 if int(Water\_level) >= max\_value:  
 message ="Red Alert!. Water level is increased by " + str(Water\_level) + "% at your place. Please Don't move out of the house. The Current Temperature is " + str(temp\_value) + "°C"  
 head="Red Alert!"  
 message\_1="Water level is increased by " + str(Water\_level) + "% at your place. Please Don't move out of the house. The Current Temperature is " + str(temp\_value) + "°C."  
 twillo\_message(message)  
 mailgun\_message(head,message\_1)  
 except Exception as e:   
 print ("Error occured: Below are the details")  
 print (e)  
 time.sleep(15)

### Conclusion

Nowadays the Internet Of things (IoT) is broadly used in worldwide, this system will display the data of the water level measured on lcd display. This project can be very helpful to the Meteorological Department to continuously monitor the dams and river beds water level. With this project it can save many people lives by giving alerts when the water level crosses beyond the limit. This project is very cost-effective, flexible and productive in areas where flood conditions happens every time