SCHOOL OF ELECTRONICS ENGINEERING(SENSE)
VELLORE INSTITUTE OF TECHNOLOGY
VELLORE

IOT BASED FLOOD MONITORING AND ALERTING SYSTEM

EMBEDDED PROJECT REPORT

By

Jaisuriyaa.P(20BEC0296)

Rahul.D(20BEC0387)

Ajith Kumar.P (20BEC0703)

JanarthanP.M (20BEC0508)

SUBMITTED TO Dr. Biswajit Dwivedy

During Winter Semester for the course

ECE3502-IOT Domain Analysis- J COMPONENT

Declaration by Authors:

This is to declare that this report has been written by us as part of our coursework. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be plagiarized, we shall take full responsibility for it.

Jaisuriyaa.P (20BEC0296)

Ajith Kumar.P (20BEC0703)

Rahul.D(20BEC0387)

Janarthan.P.M(20BEC0508)

Place: VIT, Vellore

Date: 15 April 2022

Acknowledgement

Without Dr. Biswajit Dwivedy Sir’s crucial and kind assistance, this report would not have been feasible. His ability to inspire us really aided the success of our endeavour. Even when we faced challenges or difficulties, he never stopped encouraging us and helped us get through them. Without his consistent support and direction, it would not have been feasible. He motivated us and provided intelligent advice and original tips that were beneficial to our endeavour. Additionally, we would like to express our gratitude to the VELLORE INSTITUTE OF TECHNOLOGY's authority for providing us with this chance and subject. We had the chance to attend and pick up a lot of fresh knowledge that would benefit our future and careers.

Abstract :

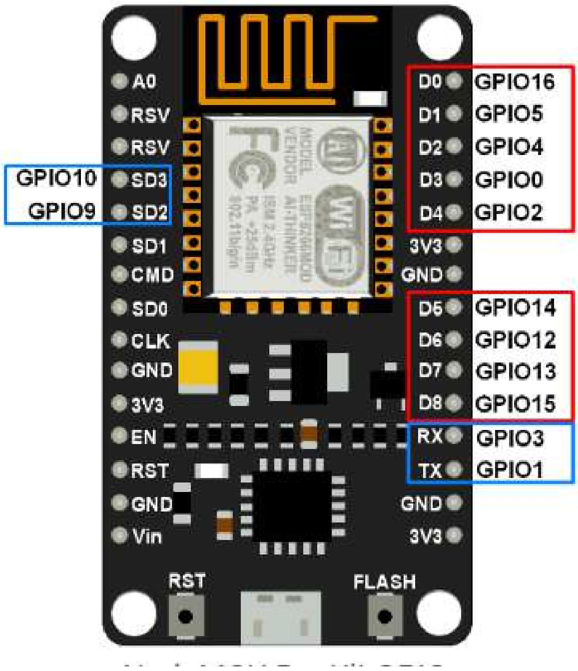
Flooding is considered one of the most destructive natural disasters in the world. In countries like India with climatic conditions occurrence of heavy rain fall and subsequent discharge of water leads to Flood. Flooding creates major damages to life, their habitats and the economy By installing of flood and rain alerting systems near major waterways vital information can be provide so that lives and property can be protected. Normal Weather monitoring and alerting systems are not quick and accurate enough to predict floods in time to prevent personal or environmental damages. The government has to spend tons of money in flood mitigation plans to help the victims and also to reduce the number in the long run damages that can occur after flooding. Since Most of the flood alerting systems involves high cost they are deployed on select locations based on priority. The purpose of this project is to sense the water level and rainfall 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. If the rainfall level is high, then it alerts people through LED signal

Introduction :

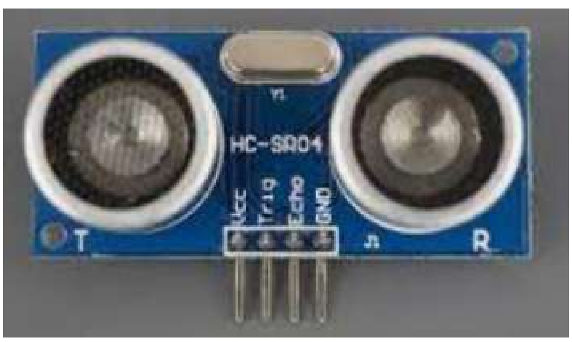
The extreme climatic changes due to the effect from various human activities such as pollutions, cutting of innumerable trees and too much of gas emission are the some of the main reason for natural disasters that occur in worldwide . The most common factor that cause major damage to life, property and country's economy is the flood .Flooding is brought on by an increased quantity of water in lake or river when it is overflowing. When a dam fractures and abruptly releasing a massive quantity of water not only houses and property are damaged , sewage overflow and chemical spillage also leads to a variety of diseases afterwards. To manage these kind of situations and alert people understanding of increased water level, increased rainfall level and speed of water flow are valuable for discovering potential seriousness of the flood. This project presents the details of how the data - like flood level and rain intensity are collected from sensors and made available on cloud and sending alert messages by using Thingspeak-an IOT platform and a wifi module (ESP-8266) and short message service (SMS) to relay data from sensors to computers or directly alert the People of that area through their mobile phone. The data from the IOT cloud can be accessed by android smart phones at anytime from anywhere in the world using the mobile app things view. The alert notification regarding the flood level in river will be send to thingspeak. In this project we make use of a cost effective system using ESP8266 and sensors, to measure rise of water level in rivers and water bodies and rainfall level and alert government authorities and people instantly by transmitting information using IOT.

Components Description:

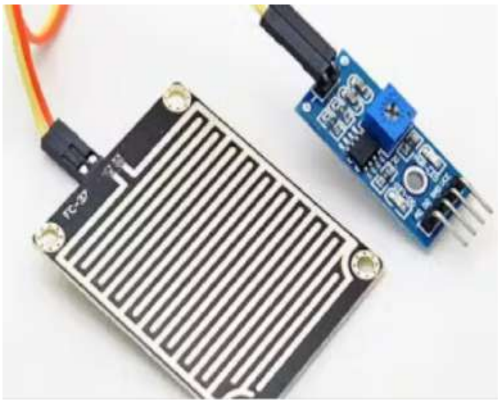
NodeMCU(ESP8266): It is a wifi-module that allows microcontrollers access to a Wi-Fi network. Using this the data can be send to the thingspeak cloud. This can be also used as the microcontroller to control the flow of operation like Arduino UNO. It has GPIO pin. GPIO (General-Purpose Input/Output) is a pin on an IC circuit. It can be either input pin or output pin, whose behaviour can be controlled at a run time. We have to include corresponding GPIO pins in the code based on the connections given in the ESP8266 microcontroller to make the application run. The only thing to take care of is that NodeMCU Devkit pins are numbered differently than internal GPIO notations of ESP8266 as shown in the below figure and table. For example, the D0 pin on the NodeMCU Devkit is mapped to the internal GPIO pin 16 of ESP8266.

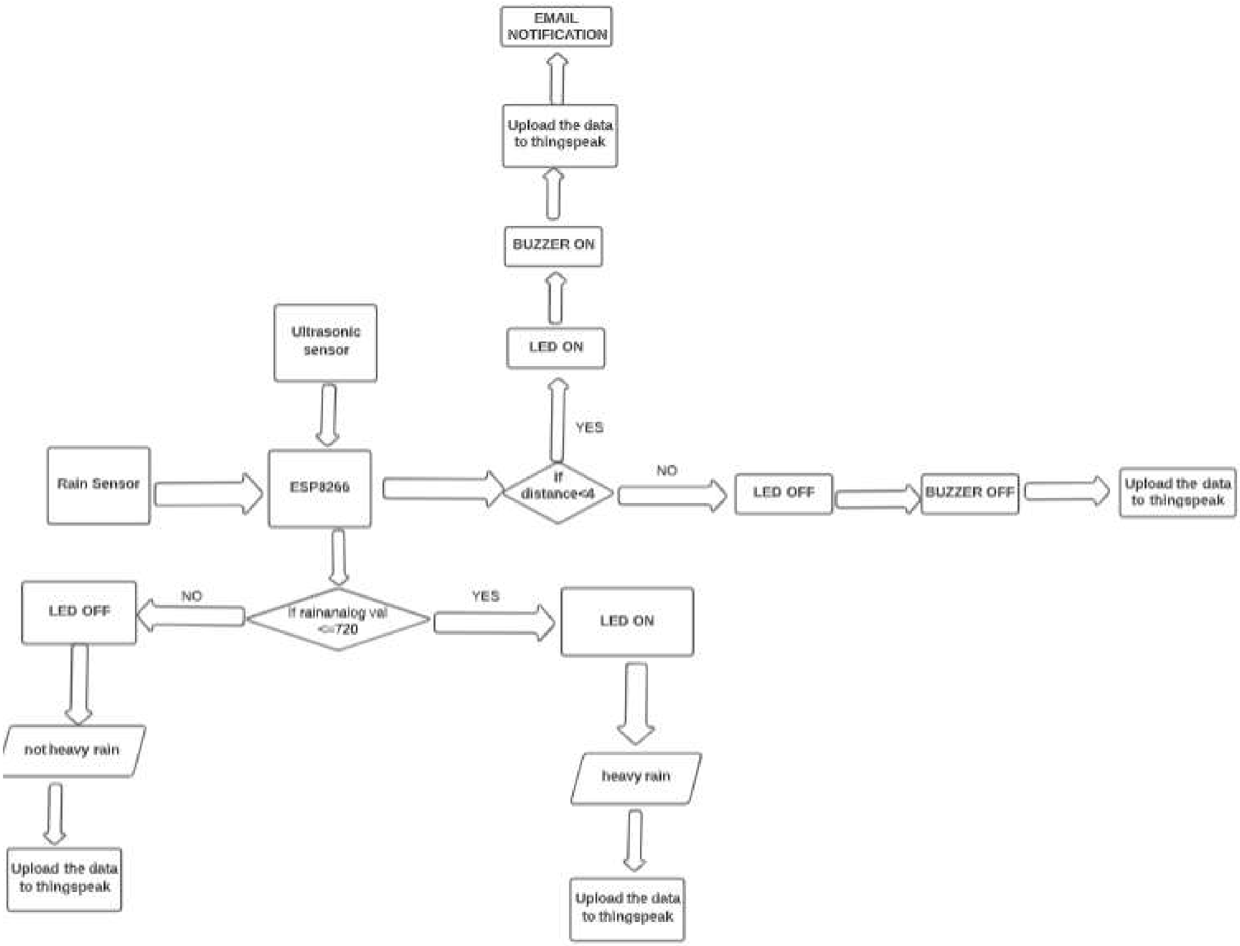


Ultrasonic Sensor: An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).In order to calculate the distance between the sensor and the object, the sensor measures the time it takes between the emission of the sound by the transmitter to its contact with the receiver. The formula for this calculation is D = ½ T x C (where D is the distance, T is the time, and C is the speed of sound ~ 343 meters/second). For example, if a scientist set up an ultrasonic sensor aimed at a box and it took 0.025 seconds for the sound to bounce back, the distance between the ultrasonic sensor and the box would be about 4.2875 meters. D = 0.5 x 0.025 x 343



Rain Sensor: It is used to detect whether the rain has occurred or not. If the rain has occurred the value of resistance present in the rain sensor decreases and the current flowing through the rain sensor increases. If the rain has not occurred the value of resistance present in the rain sensor increases and the current flowing through the rain sensor decreases. The value of the rain sensor will be lower if the rain detected by the rain sensor was more and the value of the rain sensor will be higher if the rain detected by the rain sensor was less





Methodology:

In this project, flood monitoring system using wireless sensor node is developed to observe the status of flood and rain which can alert people who will be in the area frequently affected by floods. When the sensors were triggered, all the data will be sent to Thingsspeak cloud to be viewed on user's smartphone via the wireless connection. Ultrasonic sensor and Rain sensor is used as input to the ESP8266 and power supply of it is given to power up the system to function well. The ultrasonic distance sensor is used to detect the flood level at a high prone area of flood (maximum is 4m away from it). It first measures the distance of water level from sensor and then converts it to percentage values by using the formula given. Based on the measured percentage and threshold values given by us it displays the percentage on a command prompt of Arduino IDE and takes specified action and same data is pushed to thingspeak cloud. The rain sensor display the analog values based on the rain level detected. If the rain level is high, the analog sensor value will be low and vice versa. The Thingsspeak cloud provides an interactive and easy to access platform for user or victim to get accurate information on the incoming floods and rain by displaying current condition of flood water level and rain level in real-time condition. When there are changes in flood level and the rain level , the graphs captured those data and change the measurements accordingly. So, when there is immediate change in the measurement, the LEDs will turn on which act as alerting purposes. ESP-8266 wifi module is used to send data and matlab analysis is used to generate alert SMS to the People when the flood water level reached a certain point of hazard. After this, the data will be collected from thingspeak in the file format of .csv. Now in that collected .csv file data we will be performing the machine learning algorithms like KNN, Random Forest, Logistic Regression, Decision – Tree to predict the occurrence of flood and rain per month and year. The accuracy of the above mentioned algorithm was calculated and graph was plotted for them.

Arduino Code:

```
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>
#define rainAnalog A0
const int trigPin1 = 5;
const int echoPin1 = 4;
const int redled = 0;
const int rainDigital = 12;
const int led = 13;
const int buzzer = 16;
long ch_no = 2092521;
const char * write_api = "ASGCZBPOCS61Z4ZH";//Replace with Thingspeak write
API
char ssid[] = "vivo1938";
char pass[] = "Jaigo@2012";
unsigned long startMillis;
unsigned long currentMillis;
const unsigned long period = 10000;
WiFiClient  client;
long duration1;
int distance1;
void setup()
{
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
  pinMode(redled, OUTPUT);
  digitalWrite(redled, LOW);
```

```
  pinMode(rainDigital, INPUT);
  pinMode(led,OUTPUT);
  pinMode(buzzer,OUTPUT);
  Serial.begin(9600);
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");
  Serial.println(WiFi.localIP());
  ThingSpeak.begin(client);
  startMillis = millis();  //initial start time
}
void loop()
{
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin1, LOW);
  duration1 = pulseIn(echoPin1, HIGH);
  distance1 = duration1 * 0.034 / 2;
  int rainAnalogVal = analogRead(rainAnalog);
  int rainDigitalVal = digitalRead(rainDigital);
  Serial.print("Rain Analog value = ");
  Serial.println(rainAnalogVal);
  //Serial.print("\t");
  Serial.print("Rain Digital value = ");
  Serial.println(rainDigitalVal);
  //Serial.print("\t");
  Serial.print("water level from ultrasonic sensor = ");
  Serial.println(distance1);
  delay(2000);
  if (distance1 < 4)
  {
    digitalWrite(redled, HIGH);
    digitalWrite(buzzer,HIGH);
  }
  else
  {
    digitalWrite(redled, LOW);
    digitalWrite(buzzer,LOW);
  }
if (rainAnalogVal<720){
  Serial.println("Heavy rain");
  digitalWrite(led,HIGH);
}
else{
  Serial.println("Not Heavy rain");
  digitalWrite(led,LOW);
```

```
}
  currentMillis = millis();
  if (currentMillis - startMillis >= period)
  {
    ThingSpeak.setField(1, distance1);
    ThingSpeak.setField(2, rainAnalogVal);
    ThingSpeak.writeFields(ch_no, write_api);
    startMillis = currentMillis;
  }
}
```

Logistic Regression in Machine :

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

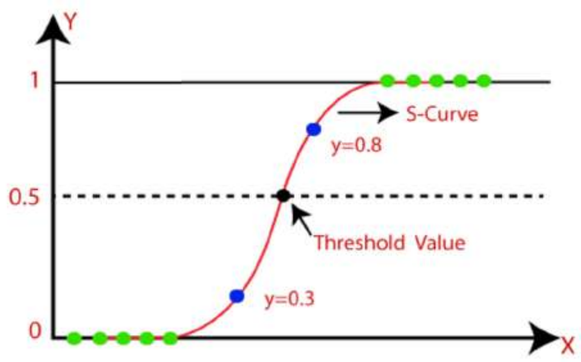
Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"

Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Steps in Logistic Regression:

To implement the Logistic Regression using Python, we will use the same steps as we have done in previous topics of Regression. Below are the steps:

- Data Pre-processing step
- Fitting Logistic Regression to the Training set
- Predicting the test result
- Test accuracy of the result(Creation of Confusion matrix)
- Visualizing the test set result.

Decision tree:

The decision tree algorithm is a popular machine learning technique used for both classification and regression tasks. It works by creating a tree-like model of decisions and their possible consequences, which is used to predict the outcome of a particular event or scenario. In a decision tree, the root node represents the entire dataset, and each subsequent node represents a feature or attribute of the data. The branches emanating from each node represent possible values or outcomes of the feature, leading to a decision or prediction at the leaf nodes. The decision tree algorithm recursively partitions the data based on the feature that maximizes the information gain or decrease in entropy at each split. This process continues until all the data is accurately classified or a stopping criterion is reached. Decision trees have several advantages, including being easy to interpret, handling both categorical and numerical data, and being able to handle missing values. However, decision trees can also be prone to overfitting, especially when the tree becomes too deep and complex. Overall, the decision tree algorithm is a powerful tool for building predictive models that can be easily understood and interpreted, making it a popular choice in many applications, including finance, healthcare, and customer analytics.

KNN Algorithm:

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data. **Example:** Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

Random Forest:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Dataset used for Machine learning:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	SUBDIVISI	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANNUAL I	FLOODS
2	KERALA	1901	28.7	44.7	51.6	160	174.7	824.6	743	357.5	197.7	266.9	350.8	48.4	3248.6	YES
3	KERALA	1902	6.7	2.6	57.3	83.9	134.5	390.9	1205	315.8	491.6	358.4	158.3	121.5	3326.6	YES
4	KERALA	1903	3.2	18.6	3.1	83.6	249.7	558.6	1022.5	420.2	341.8	354.1	157	59	3271.2	YES
5	KERALA	1904	23.7	3	32.2	71.5	235.7	1098.2	725.5	351.8	222.7	328.1	33.9	3.3	3129.7	YES
6	KERALA	1905	1.2	22.3	9.4	105.9	263.3	850.2	520.5	293.6	217.2	383.5	74.4	0.2	2741.6	NO
7	KERALA	1906	26.7	7.4	9.9	59.4	160.8	414.9	954.2	442.8	131.2	251.7	163.1	86	2708	NO
8	KERALA	1907	18.8	4.8	55.7	170.8	101.4	770.9	760.4	981.5	225	309.7	219.1	52.8	3671.1	YES
9	KERALA	1908	8	20.8	38.2	102.9	142.6	592.6	902.2	352.9	175.9	253.3	47.9	11	2648.3	NO
10	KERALA	1909	54.1	11.8	61.3	93.8	473.2	704.7	782.3	258	195.4	212.1	171.1	32.3	3050.2	YES
11	KERALA	1910	2.7	25.7	23.3	124.5	148.8	680	484.1	473.8	248.6	356.6	280.4	0.1	2848.6	NO
12	KERALA	1911	3	4.3	18.2	51	180.6	990	705.3	178.6	60.2	302.3	145.7	87.6	2726.7	NO
13	KERALA	1912	1.9	15	11.2	122.7	217.3	948.2	833.6	534.4	136.8	469.5	138.7	22	3451.3	YES
14	KERALA	1913	3.1	5.2	20.7	75.7	198.8	541.7	763.2	247.2	176.9	422.5	109.9	45.8	2610.8	NO
15	KERALA	1914	0.7	6.8	18.1	32.7	164.2	565.3	857.7	402.2	241	374.4	100.9	135.2	2899.1	NO
16	KERALA	1915	16.9	23.5	42.7	106	154.5	696.1	775.6	298.8	396.6	196.6	302.5	14.9	3024.5	YES
17	KERALA	1916	0	7.8	22	82.4	199	920.2	513.9	396.9	339.3	320.7	134.3	8.9	2945.3	YES
18	KERALA	1917	2.9	47.6	79.4	38.1	122.9	703.7	342.7	335.1	470.3	264.1	256.4	41.6	2704.8	NO
19	KERALA	1918	42.9	5	32.8	51.3	683	464.3	167.5	376	96.4	233.2	295.4	54.1	2501.9	NO
20	KERALA	1919	43	6.1	33.9	65.9	247	636.8	648	484.2	255.9	249.2	280.1	53	3003.3	YES
21	KERALA	1920	35.2	5.5	24.1	172	87.7	964.3	940.8	235	178	350.1	302.3	8.2	3303.1	YES
22	KERALA	1921	43	4.7	15	171.3	104.1	489.1	639.8	641.9	156.7	302.4	136.2	15.8	2719.9	NO
23	KERALA	1922	30.5	21.4	16.3	89.6	293.6	663.1	1025.1	320.6	222.4	266.3	293.7	25.1	3267.6	YES
24	KERALA	1923	24.7	0.7	78.9	43.5	80	722.5	1008.7	943	254.3	203.1	83.9	41.6	3484.7	YES
25	KERALA	1924	19.3	2.9	66.6	111	185.4	1011.7	1526.5	624	289.1	176.5	162.9	50.4	4226.4	YES
26	KERALA	1925	4.1	16.5	76.9	93.4	258.2	688.8	593.5	554.1	158.8	295.4	223.7	98.8	3062.1	YES
27	KERALA	1926	28.6	5.8	23.1	55.8	222.6	563.9	885.2	536	322.7	216.7	88.8	16.2	2965.4	YES
28	KERALA	1927	18.8	35.3	49.6	86.5	265.4	720.2	888.2	315	335.6	135.8	137.6	6.8	2994.7	YES
29	KERALA	1928	12.7	65.9	51.3	121.1	81.9	590.7	420.6	553.2	75.9	321.5	155.2	52.7	2502.8	NO

Machine Learning code using python:

#KNN

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

import warnings
```

```
warnings.filterwarnings('ignore')
data = pd.read_csv("C:/Users/Jana.P.M/Desktop/IOT LAB/jcomp.csv")
data.head(10)
data.apply(lambda x:sum(x.isnull()), axis=0)
data['FLOODS'].replace(['YES','NO'],[1,0],inplace=True)
data.head(10)
x=data.iloc[:,1:14]
x.head(10)
y = data.iloc[:, -1]
y.head(10)
from sklearn import preprocessing
minmax = preprocessing.MinMaxScaler(feature_range=(0,1))
minmax.fit(x).transform(x)
from sklearn import model_selection,neighbors
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
print(y_train.tail(10))
print(x_train.head(10))
clf = neighbors.KNeighborsClassifier()
knn_clf = clf.fit(x_train,y_train)
y_predict = knn_clf.predict(x_test)
print('predicted chances of flood')
print(y_predict)
print(y_test.values)
 from sklearn.model_selection import cross_val_score
knn_accuracy =cross_val_score(knn_clf,x_test,y_test,cv=5,scoring='accuracy',n_jobs=1)
print(knn_accuracy)
print(knn_accuracy.mean())
```

 # Logistic Regression

```
x_train_std = minmax.fit_transform(x_train)
```

```
x_test_std = minmax.transform(x_test)
print(x_train_std[:10])
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()
lr_clf = lr.fit(x_train_std,y_train)
y_predict = lr_clf.predict(x_test_std)
print('Predicted values')
print(y_predict)
print('Actual values')
print(y_test.values)
lr_accuracy =cross_val_score(lr_clf,x_test_std,y_test,cv=5,scoring='accuracy',n_jobs=1)
print(lr_accuracy.mean())
print(lr_accuracy)
```

Decision tree classification

```
from sklearn.tree import DecisionTreeClassifier
dtc_clf = DecisionTreeClassifier()
dtc_clf.fit(x_train,y_train)
dtc_clf_acc =
cross_val_score(dtc_clf,x_train_std,y_train,cv=3,scoring="accuracy",n_jobs=-1)
dtc_clf_acc
y_pred = dtc_clf.predict(x_test)
print('Predicted values')
print(y_pred)
print("actual values:")
print(y_test.values)
 dtc_accuracy=cross_val_score(dtc_clf,x_test_std,y_test,cv=5,scoring='accuracy',n_jobs=-1)
print(dtc_accuracy.mean())
print(dtc_accuracy)
```

Random forest classification

```
from sklearn.ensemble import RandomForestClassifier
rmf = RandomForestClassifier(max_depth=3,random_state=0)
rmf_clf = rmf.fit(x_train,y_train)
rmf_clf
y_pred = rmf_clf.predict(x_test)
print('Predicted values')
print(y_pred)
print("actual values:")
print(y_test.values)
rmf_clf_acc =
cross_val_score(rmf_clf,x_train_std,y_train,cv=5,scoring="accuracy",n_jobs=-1)
print(rmf_clf_acc.mean())
print(rmf_clf_acc)
 models = []
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier
models.append(('KNN', KNeighborsClassifier()))
models.append(('LR', LogisticRegression()))
models.append(('DT', DecisionTreeClassifier()))
models.append(('RF', RandomForestClassifier()))

names = []
scores = []
for name, model in models:
    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)
    scores.append(accuracy_score(y_test, y_pred))
```

```
    names.append(name)
tr_split = pd.DataFrame({'Name': names, 'Accuracy Score': scores})
print(tr_split)
import seaborn as sns
axis = sns.barplot(x = 'Name', y = 'Accuracy Score', data =tr_split )
axis.set(xlabel='Classifier', ylabel='Accuracy')
for p in axis.patches:
    height = p.get_height()
    axis.text(p.get_x() + p.get_width()/2, height + 0.005, '{:1.4f}'.format(height), ha="center")
plt.show()
```

Results:

	Name	Accuracy Score
0	KNN	0.958333
1	LR	0.958333
2	DT	0.791667
3	RF	0.875000

Figure 1. Accuracy obtained by the algorithm KNN, Logistic Regression, Decision-Tree, Radom Forest

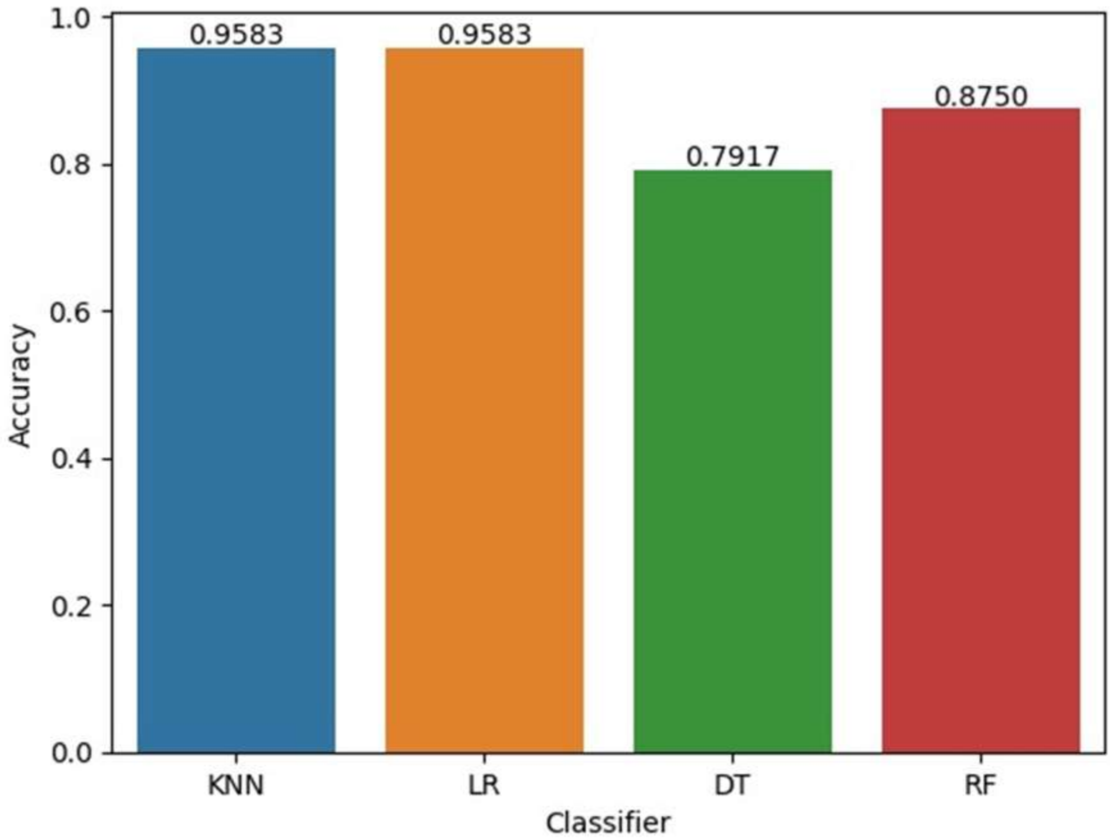


Figure 2. Bar chart plot of the Accuracy obtained by the algorithm KNN, Logistic Regression, Decision-Tree, Radom Forest

```
Predicted values
[1 1 1 0 0 1 0 0 1 0 0 1 1 0 0 0 0 0 1 1 0 1 1 0]
actual values:
[1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1]
```

Figure 3. Predicted and actual recorded values of RANDOM FOREST algorithm

```
Predicted values
[1 1 1 0 0 1 0 0 1 0 0 1 1 0 0 0 0 0 1 1 0 1 1 0]
actual values:
[1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1]
```

Figure 4. Predicted and actual recorded values of DECISION TREE algorithm

```
Predicted values
[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1]
Actual values
[1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1]
```

Figure 5. Predicted and actual recorded values of LOGISTIC REGRESSION algorithm

```
predicted chances of flood
[1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1]
[1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1]
```

Figure 6. Predicted and actual recorded values of KNN algorithm

Thingspeak:

Field1- Ultrasonic sensor data

Field2- Rain sensor data

Field3- Lamp indicator to turn on when the water level measurement distance from the ultrasonic sensor is less than 4cm otherwise the Lamp indicator will be turned OFF

Field4- Lamp indicator to turn ON whenever the rain sensor analog value is less than 720 otherwise the Lamp indicator will be turned OFF

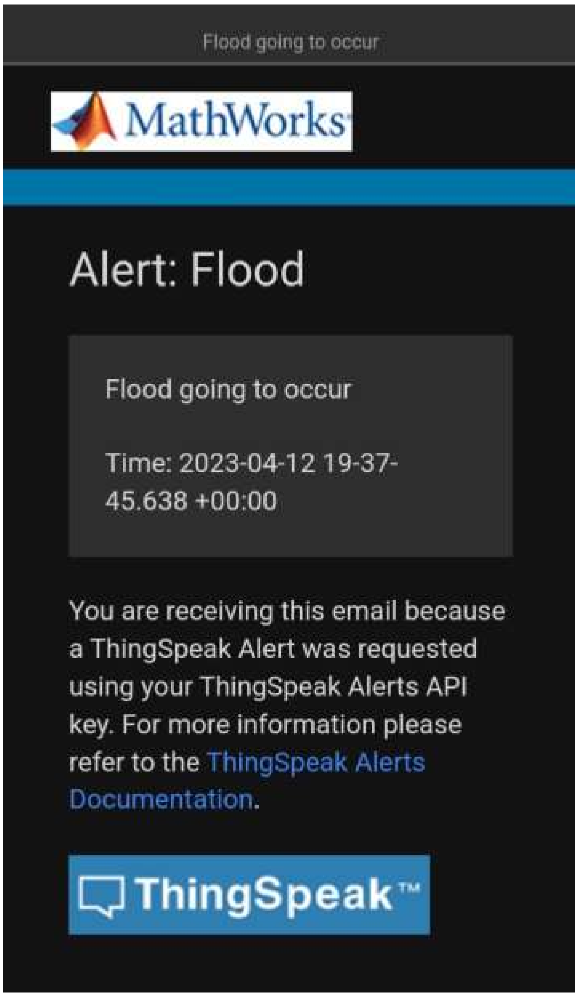


Figure 7. Email Notification from thingspeak

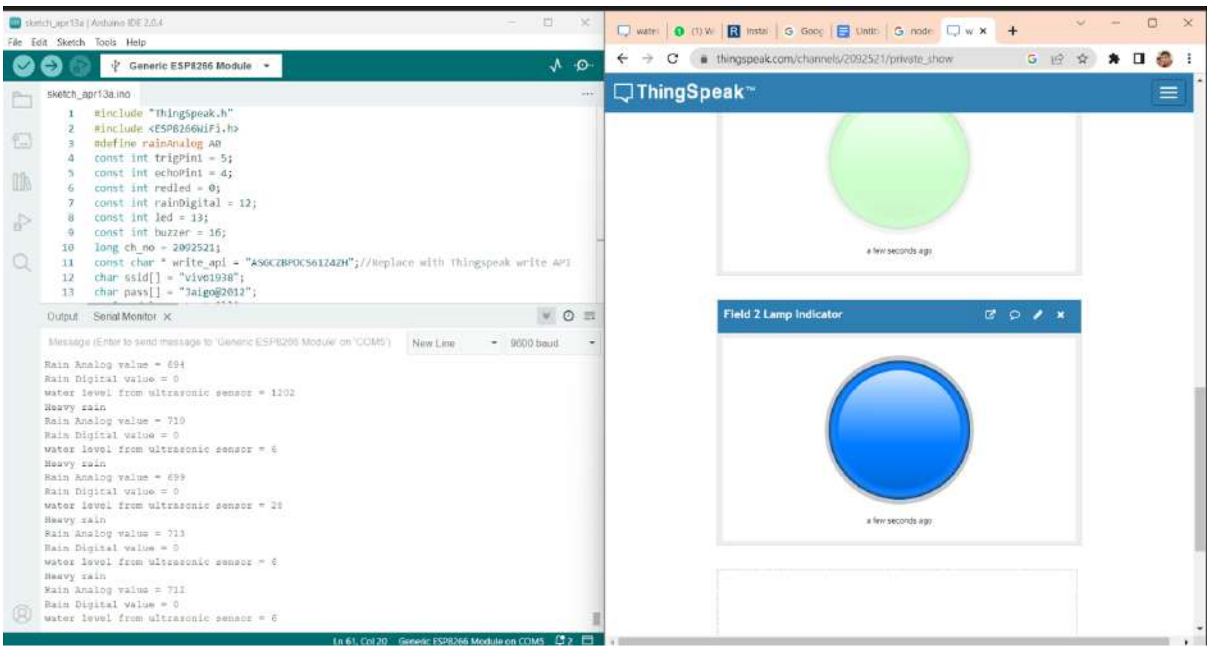


Figure 8. Heavy rain with less water level

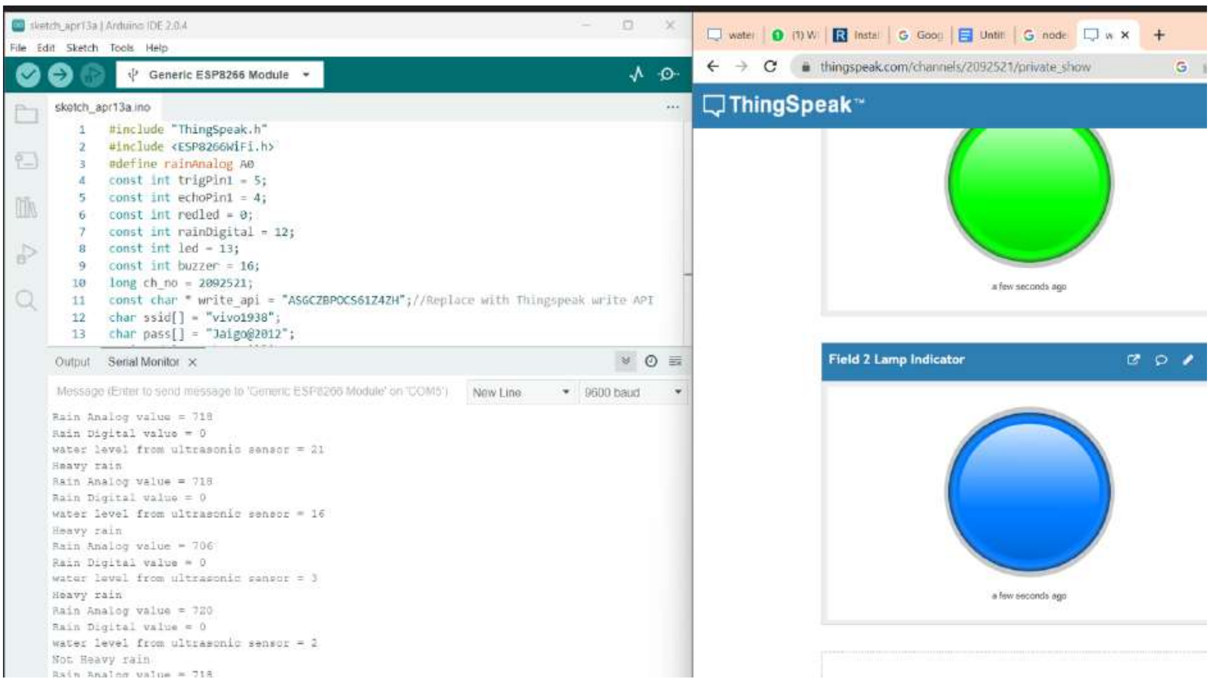


Figure 9. Heavy rain with more water level

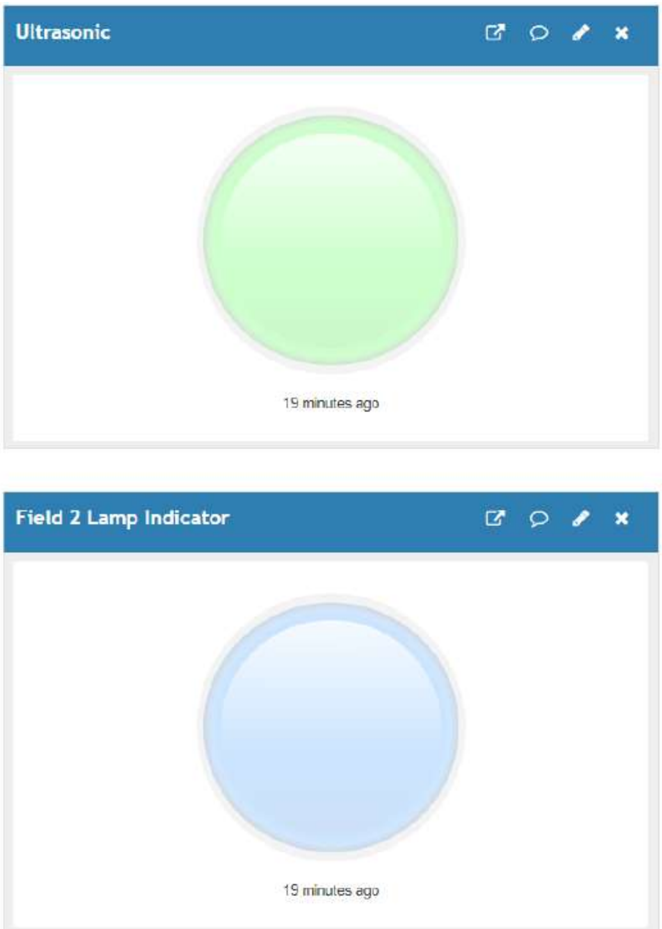


Figure 10. No rain with less water level

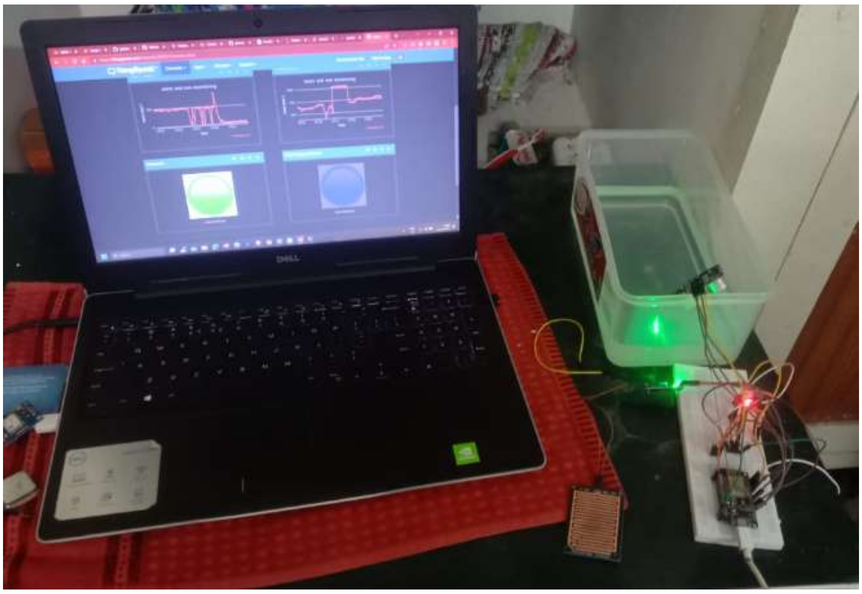


Figure 11. No rain with High water Level

1	created_a	entry_id	field1	field2
2	2023-04-0	1	8	1024
3	2023-04-0	2	6	1024
4	2023-04-0	3	3	1024
5	2023-04-0	4	3	1024
6	2023-04-0	5	3	1024
7	2023-04-0	6	3	790
8	2023-04-0	7	3	1024
9	2023-04-0	8	3	1024
10	2023-04-0	9	3	762
11	2023-04-0	10	3	704
12	2023-04-0	11	3	786
13	2023-04-0	12	3	1024
14	2023-04-0	13	3	1024
15	2023-04-0	14	2	1024
16	2023-04-0	15	2	1024
17	2023-04-0	16	2	1024
18	2023-04-0	17	2	1024
19	2023-04-0	18	2	945
20	2023-04-0	19	5	1024
21	2023-04-0	20	4	1024
22	2023-04-0	21	2	1024
23	2023-04-1	22	8	950
24	2023-04-1	23	9	900
25	2023-04-1	24	4	860
26	2023-04-1	25	5	810
27	2023-04-1	26	10	770
28	2023-04-1	27	11	721
29	2023-04-1	28	12	703

Figure 12. Dataset obtained from thingspeak

Applications

This system can be implemented for the real-life operation purpose. The system will alert you with appropriate information about the Flood and rain. It can detect even an inch raise of water level, rainfall level and give alert. We have tested this system by real time for water level measurement and rainfall measurements successfully . If the water level and rain level increases, it will send an alert immediately. It can be used to monitor natural calamities. It can be used to monitor the floods water level and rainfall level. System has ability to send alert messages to the people through ESP Wi-fi module. Users can also monitor flood level and rainfall level from thingspeak cloud.

Conclusion: This project deals about the monitoring of flood level remotely from anywhere in the world using internet through a personal computer or Smartphone and alert the general public. In this project raw data loaded in the cloud can be visualized in graphical format with in a very short span of time at a remote desk/ mobile app. We have used Arduino, ESP and MALAB's Thingspeak IOT platform along with the sensors for the first time to monitor and alert flood situation. In this system, sensors work with low input power and detects the flood level in the river with a high accuracy. It can detect even an inch raise of water level and give alert. We have tested this system by real time for water level measurement successfully. If the water level increases, it will send an alert immediately. In this study, the prototype is only using a small scale of sensor detection within 50cm. In actual world, the system needs to detect the flood for about 1 to 2 meter if the system is placed at the riverside to detect flood. Besides, this prototype needs to be improved on the water-resistant features so that when the rain started to fall, it cannot damage the sensor node. A proper installation needs to be done so that the system can be put at any kind of surfaces to avoid it being fall down when water level rise up. Therefore, the system could help a huge number of victim's life whenever the future work could be done on it.

References:

1) Munawar, H. S., Hammad, A. W., & Waller, S. T. (2021). A review on flood management technologies related to image processing and machine learning. Automation in Construction, 132, 103916.
2) Rani, D. S., Jayalakshmi, G. N., & Baligar, V. P. (2020, March). Low cost IoT based flood monitoring system using machine learning and neural networks: flood alerting and rainfall prediction. In 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) (pp. 261-267). IEEE.
3) Chang, F. J., Hsu, K., & Chang, L. C. (Eds.). (2019). Flood forecasting using machine learning methods. MDPI
4) https://www.boltiot.com/projects/iot-based-flood-monitoring-and-alerting-system
5) https://www.hackster.io/Techatronic/water-level-indicator-using-nodemcu-50e173
6) https://diyi0t.com/rain-sensor-tutorial-for-arduino-and-esp8266/
7) Deowan, M.E., Haque, S., Islam, J., Hanjalayeamin, M., Islam, M.T. and Meghla, R.T., 2022, November. Smart Early Flood Monitoring System Using IoT. In *2022 14th Seminar on Power Electronics and Control (SEPOC)* (pp. 1-6). IEEE.
8) Prakash, C., Barthwal, A. and Acharya, D., 2022. FLOODWALL: A Real-Time Flash Flood Monitoring and Forecasting System Using IoT. *IEEE Sensors Journal*, *23*(1), pp.787-799.
9) Chhetri, S. and Mahamuni, C.V., 2022, October. Design and Implementation of an IoT-Edge Computing-based Flood Monitoring System for Early Warning: A Case of Nagaon District in Assam. In *2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)* (pp. 1-9). IEEE.
10) Maer, E.D. and Pop, A.A., 2022, September. Design of IoT based flood monitoring and alerting system. In *2022 IEEE 20th International Power Electronics and Motion Control Conference (PEMC)* (pp. 536-539). IEEE
11) Jan, O.R., Jo, H.S., Jo, R.S. and Kua, J., 2022. Real-Time Flood Monitoring with Computer Vision through Edge Computing-Based Internet of Things. *Future Internet*, *14*(11), p.308..