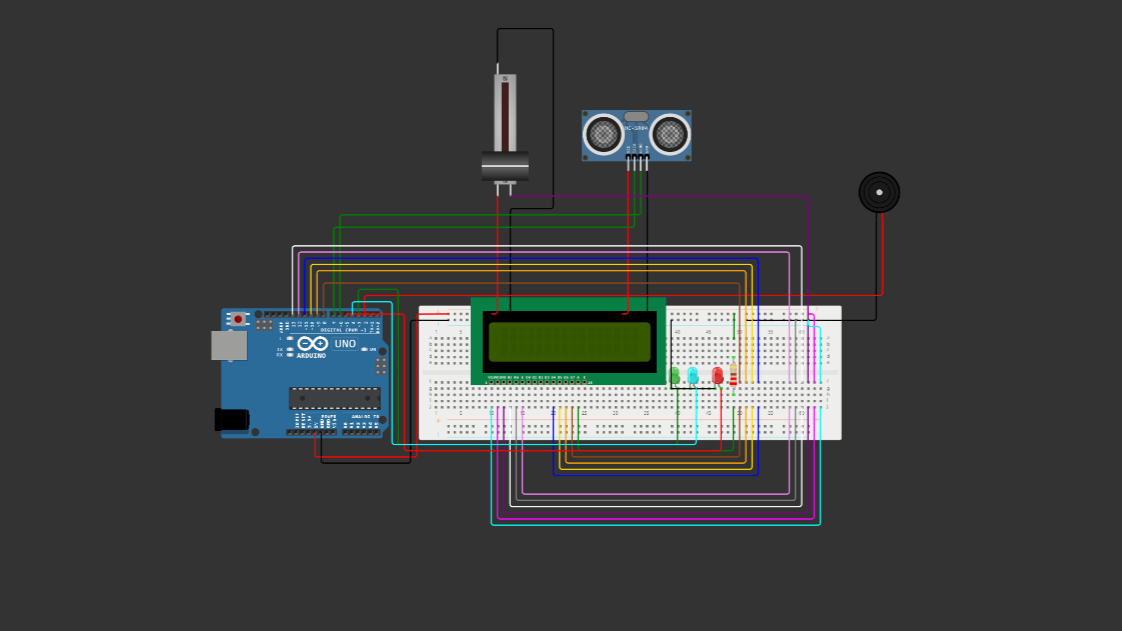
Flood Monitoring & Early Warning

Phase-3

**Design for Flood Monitoring & early warning**



C++ code for flood monitoring & Early warning

#include <LiquidCrystal.h>

LiquidCrystal lcd(13, 12, 11, 10,  9, 8);

int trig = 7;

int echo = 6;

int red = 5;

int blue = 4;

int green = 3;

int buz = 2;

int distance;

long duration;

void setup()

{

 pinMode(red,OUTPUT);

 pinMode(blue,OUTPUT);

 pinMode(green,OUTPUT);

 pinMode(buz,OUTPUT);

 pinMode(trig,OUTPUT);

 pinMode(echo,INPUT);

**Serial**.begin(9600);

 lcd.begin(16,2);

 lcd.setCursor(1,0);

 lcd.print("Distance:");

}

void loop()

{

 digitalWrite(trig,LOW);

 digitalWrite(trig,HIGH);

 delayMicroseconds(10);

 digitalWrite(trig,LOW);

 float duration = pulseIn(echo,HIGH);

 // Distance = speed \* Time

 float distance = (0.0343 \* duration)/2 ;

 lcd.setCursor(10,0);

 lcd.print(distance);

  if(distance>250)

  {

   lcd.setCursor(3,1);

   lcd.print("Safe Zone  ");

   digitalWrite(red,LOW);

   digitalWrite(green,HIGH);

   digitalWrite(blue,LOW);

   digitalWrite(buz,LOW);

   delay(1000);

  }

   else if(distance>200&&distance>100)

  {

   lcd.setCursor(3,1);

   lcd.print("Medium Zone");

   digitalWrite(red,LOW);

   digitalWrite(green,LOW);

   digitalWrite(blue,HIGH);

   digitalWrite(buz,LOW);

   delay(1000);

  }

  else if(distance<100)

  {

   lcd.setCursor(3,1);

   lcd.print("Danger Zone");

   digitalWrite(red,HIGH);

   digitalWrite(green,LOW);

   digitalWrite(blue,LOW);

   digitalWrite(buz,HIGH);

  }

}

Python code for flood monitoring & Early warning

# Flood Detection System

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BOARD)

lcd\_rs = 13

lcd\_en = 12

lcd\_d4 = 11

lcd\_d5 = 10

lcd\_d6 = 9

lcd\_d7 = 8

trig = 7

echo = 6

red = 5

blue = 4

green = 3

buz = 2

GPIO.setup(red, GPIO.OUT)

GPIO.setup(blue, GPIO.OUT)

GPIO.setup(green, GPIO.OUT)

GPIO.setup(buz, GPIO.OUT)

GPIO.setup(trig, GPIO.OUT)

GPIO.setup(echo, GPIO.IN)

def setup():

GPIO.output(red, GPIO.LOW)

GPIO.output(blue, GPIO.LOW)

GPIO.output(green, GPIO.LOW)

GPIO.output(buz, GPIO.LOW)

GPIO.output(trig, GPIO.LOW)

GPIO.output(trig, GPIO.HIGH)

time.sleep(0.00001)

GPIO.output(trig, GPIO.LOW)

lcd\_init()

print\_distance()

def lcd\_init():

lcd = LiquidCrystal(lcd\_rs, lcd\_en, lcd\_d4, lcd\_d5, lcd\_d6, lcd\_d7)

lcd.begin(16, 2)

lcd.setCursor(1, 0)

lcd.print("Distance:")

def print\_distance():

while True:

GPIO.output(trig, GPIO.LOW)

GPIO.output(trig, GPIO.HIGH)

time.sleep(0.00001)

GPIO.output(trig, GPIO.LOW)

duration = pulseIn(echo, GPIO.HIGH)

distance = (0.0343 \* duration) / 2

lcd.setCursor(10, 0)

lcd.print(distance)

if distance > 250:

lcd.setCursor(3, 1)

lcd.print("Safe Zone ")

GPIO.output(red, GPIO.LOW)

GPIO.output(green, GPIO.HIGH)

GPIO.output(blue, GPIO.LOW)

GPIO.output(buz, GPIO.LOW)

time.sleep(1)

elif distance > 200 and distance > 100:

lcd.

**DEFINITION:**

**Importing Libraries:**

import RPi.GPIO as GPIO: Imports the RPi.GPIO library, which allows interaction with the Raspberry Pi's GPIO pins.

import time: Imports the time module, used for adding time delays.

**GPIO Setup:**

Sets the GPIO mode to GPIO.BOARD, which refers to using physical pin numbering.

Defines pin numbers for various components (LCD, ultrasonic sensor, LEDs, buzzer).

Sets up the GPIO pins as either input or output.

**Setup Function:**

def setup(): Initializes and configures the various components for the flood detection system.

Sets the initial states for the LEDs and ultrasonic sensor trigger pin.

Calls lcd\_init() and print\_distance() functions.

It is missing the continuation of the code inside this function.

**LCD Initialization Function (lcd\_init):**

def lcd\_init(): Initializes the LCD using a LiquidCrystal instance (assumed to be a custom class).

Sets up the LCD with specified pin connections and sets the cursor position to start displaying the distance.

**Print Distance Function (print\_distance):**

def print\_distance(): Contains a loop to continuously measure and display the distance using the ultrasonic sensor.

Triggers the ultrasonic sensor to measure distance and calculates the distance based on the duration of the echo.

Displays the distance on the LCD.

Controls LEDs and buzzer based on the measured distance to indicate the safety zone.

It is incomplete and missing the continuation of the code inside this function.

The code appears to be a part of a larger project to create a flood detection system using ultrasonic sensors to measure water levels. The LEDs and buzzer are used to indicate the safety zone based on the measured distance.

However, it's important to note that the code is incomplete and seems to have some errors, particularly within the setup() and print\_distance() functions. Additionally, the LiquidCrystal class is referenced but not defined in the provided code, which could be a custom class or an external library.

If you have a specific question or need further assistance with a particular aspect of the code or the flood detection system, feel free to ask!

**Finally:**

**This code essentially creates a distance measuring system using an ultrasonic sensor and provides visual and auditory feedback based on the measured distance. The LCD displays the current distance, and the LED color and buzzer indicate the safety zone.**