IOT PHASE-4

This phase involves creating a web application that displays real-time water level data from IoT sensors and issues flood warnings when needed.

HTML, CSS, and JavaScript Web Application:

Create a folder for your web application project and create the following files within it:

- index.html for the main HTML structure.
- Web.py for code integration.

Index.html

```
<!DOCTYPE html>

<html>

  <head>

    <script type="text/javascript">

      function load()

      {

      setTimeout("window.open('http://127.0.0.1:5000/', '_self');", 500);

      }

      </script>

     <style>

     .Line2 {

      border-left: 180px solid lightgray;
      height:{{ blue_line }}px;
    }
    .Line1 {
      border-left: 180px solid #28a1f7;
      height: {{ gray_line }}px;
    }
    .Div1 {
    width: 100%;
    height: 100%;
    margin: 1px 0;
    background-color: #ffffff;
    background-image: url({{url_for('static', filename='water_tank.jpeg')}});
    }
    </style>
</head>
<body onload="load()">
  <center>
    <br>
    <br>
  <table name="Table1" >
    <tr>
      <td class="Div1">
        <table align="bottom" name="Table2"  height=420px style="border-spacing: 44px;">
```

```
            <tr>
              <td >
               <br>
 <br> <br>
                 <br>
                <div class="Line2"></div>
                <div class="Line1"></div>
              </td>
            </tr>
          </table>
        </td>
      </tr>
    </table>
    </center>
  </body>
</html>
```

Web.py

You will need o install flask to receive data from the Arduino and process it.

```
pip install Flask
```

Code:

```
import time
from flask import *
import RPi.GPIO as GPIO
```

```
from datetime import datetime

#----------------------------------------------------------
# Setup
app = Flask(__name__)
TRIG = 11
ECHO = 12

def setup():
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(TRIG, GPIO.OUT)
        GPIO.setup(ECHO, GPIO.IN)

def distance():
        GPIO.output(TRIG, 0)
        time.sleep(0.000002)
        GPIO.output(TRIG, 1)
        time.sleep(0.00001)
        GPIO.output(TRIG, 0)
        while GPIO.input(ECHO) == 0:
                a = 0
        time1 = time.time()
        while GPIO.input(ECHO) == 1:
                a = 1
        time2 = time.time()
        during = time2 - time1
        return during * 340 / 2 * 100
#----------------------------------------------------------
# Flask
@app.route('/')
def index():
    setup()
    dis = distance()
    percent = (dis-3)/.24
    blue_line = round(percent*2.6)
    gray_line = 260-blue_line
    return render_template('index.html', blue_line=blue_line, gray_line=gray_line)
```

To integrate above code into the early warning platform:

```
def index():
    return app.send_static_file('index.html')
@app.route('/get_data')
def get_data():
    water_level = 50  # Example value, replace with your actual data
    return jsonify({'water_level': water_level})
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```