A Flood Monitoring and Warning System

# Summary

The impact caused by floods is projected to be on an increase, with the number of people affected and the property damage caused by this is expected to increase exponentially. FMWS enable the early detection of floods, and the subsequent dissemination of warnings to individuals and/ or relevant bodies. Literature review conducted found that existing FMWS projects suffered from a few drawbacks: reliability of sensor data, scalability and usability.

This project used a client-server architecture in the development of the FMWS. The clients in this project were comprised of the sensing layer – sensors and microcontrollers. The server was comprised of a website, web service and database.

The sensing layer made use of ultrasonic sensor for water level detection, flow rate sensor, temperature and humidity sensors. The use of multiple sensors mitigated the anomalies of the ultrasonic sensor, and its accuracy shortcoming due to the effect of temperature gradients. The ESP 8266 microcontroller on the nodeMCU development board was used in the sensing layer.

Sensor data read from the sensing layer is posted to the web layer via a web service. Authentication was performed to ensure authorized end-points post data to the service. Users were capable of being registered to the system via a website, allowing them to receive alerts on imminent and ongoing floods. Through the website, information was also provided on safety, preparedness and guidelines for imminent, ongoing and post floods.

Through an administrator dashboard, an administrator was capable to creating a location to be monitored on the system. The administrator was able to define the thresholds needed in the detection of floods as need, allowing the system to keep up with seasonal changes. Sensing devices were able to be registered to the FMWS, and deployed to particular locations.

Evaluation of the project revealed that the objectives set out for the project had been met. There were, however gaps found to be addressable in future works of the project – in particular in the design of a casing of the sensing devices.

# Introduction

## Objectives

The project aims at:

1. Using sensors to measure environmental weather variables in the detection of floods.

2. Program a controller to receive and analyze sensor and meteorological data to aid in the identification of floods.

3. Designing and testing web-based services to allow monitoring and receiving of flooding warning by users.

## Significance

The development of the FMWS shall enable warning of users in areas susceptible to fluvial flooding on imminent floods. Through the system the civilians would be able to view live sensor data on their regions, and through SMS or a website, receive notifications on imminent floods.

The system shall provide civilians information on safety and evacuation procedures for imminent floods. The system shall also allow users to access a dedicated helpline for rescue, guidance and counseling services.

The system shall contribute to the body of work done on the FMWS field. Current and future work proposed from the system could help in the realization of innovative and applicable FMWS to be used in the real world.

# Literature Review

**…Reducted…**

# Methodology

## System Requirements

The requirements of the system could be categorized as either functional or non-functional

### Functional Requirements

1. The system shall allow collection of sensor data for flooding risk assessment.

2. The system shall allow storage of the sensor data.

3. The system shall provide civilians warning in the case of imminent floods.

4. The system shall provide instructions and helpful information for response and evacuation in the case of imminent floods.

5. The system allow viewing of logged sensor data by interested parties.

### Non-Functional Requirements

1. The user experience is to be smooth.

2. The system should be protected from data injection by unauthorized parties.

## System Block Diagram

The block diagram of the system is shown in Figure 1. The proposed FMWS can be viewed of as being comprised of three main layers: the sensor layer, the web layer and the user layer. This section discusses the overall structure and functioning of the FMWS based on this layering.

The sensor layer is comprised of sensors and a microcontroller. It was apparent from the previous works that the use of just a single ultrasonic sensor could lead to inaccuracies and outliers due to the impact of temperature on the sensor readings. As such, multiple sensors are to be used to mitigate this effect and improve the accuracy of the system.

An ultrasonic sensor is to be used to measure the water level. A water flow sensor is to be used to measure the rate of flow of water. A temperature sensor is to measure the ambient temperature of the area while a humidity sensor is to measure the humidity of the area.

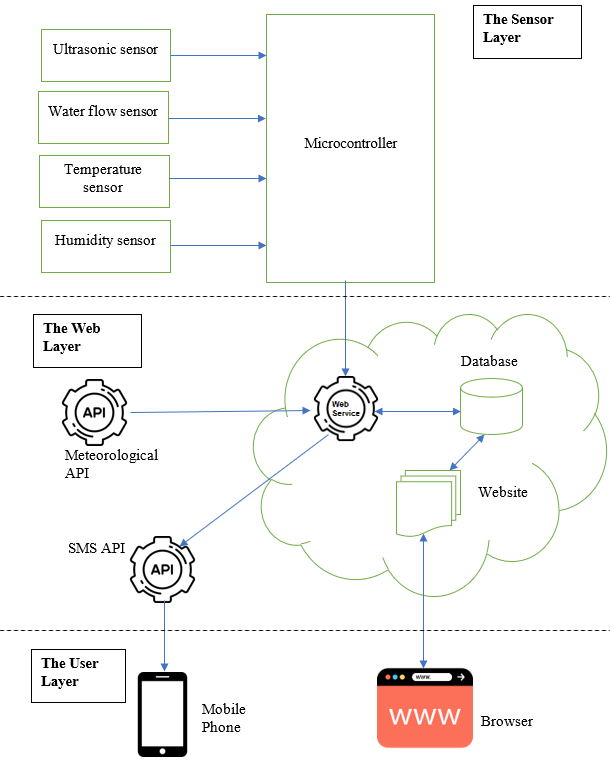


Figure 6: System block diagram

A microcontroller would be used to periodically read the sensors. The microcontroller would then preprocess the sensor data read before sending them to the web layer for storage and analytics. The TCP/IP protocol stack is to be used for communication between the microcontroller and the web layer. This communication would be achieved using an internet-enabled interface on the microcontroller.

The web layer is comprised of: a web service, a database and website. Data sent by the microcontroller would be received by the FMWS web service. The web service is to authenticate the data to ensure a registered node sent it, while identifying the region of the node. The service would then parse the received data to obtain the individual data items sent from the sensor layer. By accessing public meteorological APIs, the FMWS web service would combine the received sensor data with rainfall reading and forecasts for that particular region, and based on these current and previous reading made, determine whether the are was at risk of flooding.

A database would be used to log the data received from the sensor layer. This would be helpful in the decision-making of the FMWS service, as the service would have access to previous records rather than just the current data in the risk assessment of floods. In addition to that, the database would contain generalized information about the location of deployment and access credentials of the sensor systems deployed on the field. Thie implication of this would be a smoother deployment, authentication and identification of sensor layer nodes. Furthermore, the database would store the threshold and set points used in the risk assessment of floods. Using this design, rather than hard-coding the threshold on the microcontroller, would provide flexibility in the policy for flooding risk assessment, ensuring the FMWS kept abreast with changing climatic conditions.

The website would be used as the interface between users of the FMWS and the whole system. Through the website, interested parties would be able to view logged data, receive warning on imminent floods, view instructions on evacuation procedures and have access to a dedicated helpline. In addition to just serving regular users, the website is to allow admins to define the flooding risk assessment thresholds.

An SMS API would be used to enable the web service send out warning SMS to users registered to the region served by a sensor node, from which an imminent flood was determined. The use of the SMS would be complementary to information available on the website, ensuring that the users could have multiple channels of communication with the FMWS.

The user layer would be comprised of the users making use of the FMWS. Two broad classes of users would exist: civilians and admins. Civilians would register to the FMWS based on their region, and provide information necessary to contact them such as their phone numbers. By SMS or the website, they would have access to the FMWS. Admins would be able to define the policy needed for the risk assessment in the FMWS. They would have access to the system via their browsers.

## System Flowchart

The system flowchart is shown in figure 2.

The sensor layer and web layer can be viewed as running in parallel. When a sensor layer node starts up, it reads its sensor data, then sends them to the web layer. The sensor layer node then sleeps for a given duration, before reiterating the read, send sleep infinite loop again.

In the web layer, when a POST request is received from a sensor layer node, the node is first authorized. Further processing is terminated if authorization fails, and the layer waits for another request. If the node is authorized, the received sensor data is parsed and stored in a database. The data is then evaluated against previously stored data and other data points using a risk assessment algorithm. If it is deemed that the area served by the sensor node is at flood risk, and if no previous flood alerts had been sent for that region, the users serviced by that node are fetched, and issued alerts for the imminent flood.

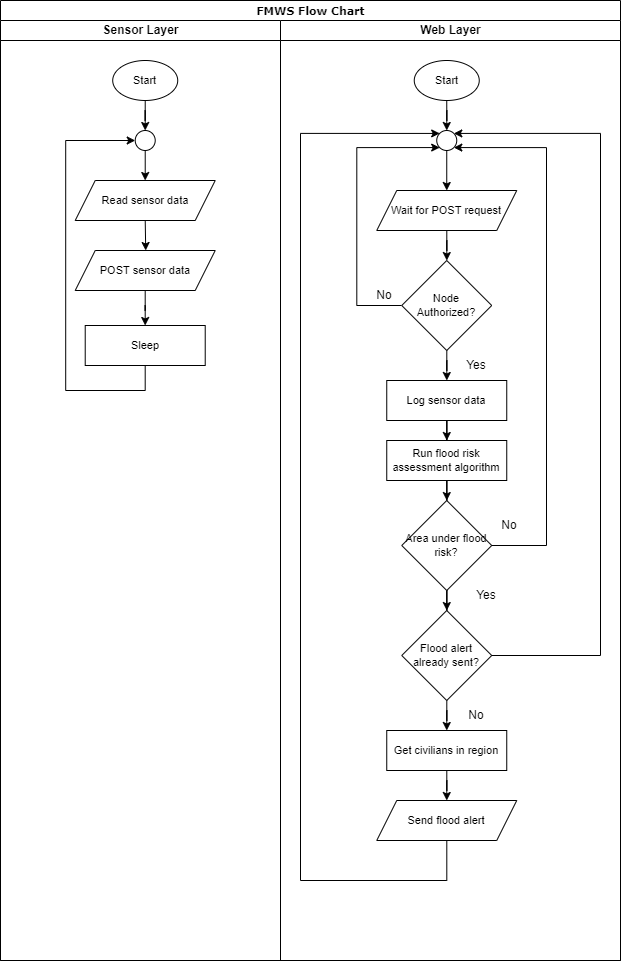


Figure 7: System flow chart

# Selection of Components

This section documents the components selected to achieve the objectives of the project while adhering to the functional and functional requirements proposed for the system while implementing the logic described in the previous chapter. This documentation shall cover the selection of the components according to the three layers proposed for the system: sensor layer, web layer and user layer.

## Sensor Layer Components

The sensor layer shall be made of both hardware and software components.

### Hardware Components

The hardware components of the sensor layer are as follows:

**Microcontroller**

The ESP 8266 nodeMcu was chosen to server as the microcontroller for this project. The device would be responsible for reading the sensor data, and through the internet, transmit them to the web service.



Figure 8: ESP 8266 nodeMcu

The device has the following specifications [19]:

* Operating voltage: 3.3V DC
* Input voltage: 7-12V
* Digital I/O pins: 16
* Analog I/O pins: 1
* TCP/IP Stack
* WiFi interface

**Ultrasonic Sensor**

The HC-SR04 sensor was selected to server this purpose. The sensor would be responsible for measuring the water level of the area.

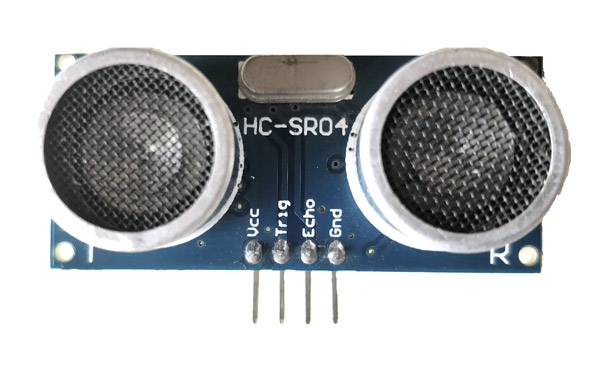


Figure 9: HC-SR04 sensor

The sensor has the following specifications [20]:

* Operating voltage: 5V
* Practical measuring distance: 2cm – 80cm
* Accuracy: 3mm

**Water Flow Sensor**

A G¾” water flow sensor was selected for this project. The sensor would be responsible for detecting the flow rate of fluvial water.



Figure 10: Water flow sensor

The sensor has the following specifications [21]:

* Working voltage: 5-24V DC
* Flow rate range: 1-60L/min
* Sensing technology – hall sensor

**Temperature & Humidity Sensors**

The DHT22 sensor was selected to serve for both temperature and humidity measurement.

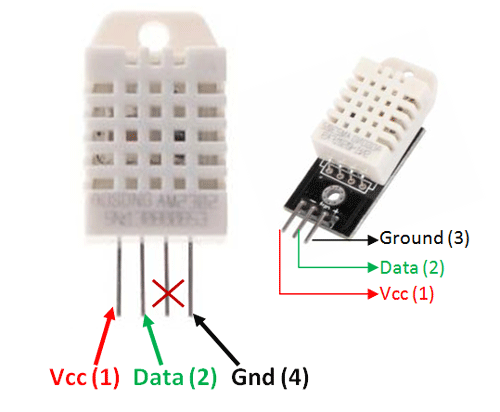


Figure 11: DHT22 sensor

The sensor has the following specifications [22]:

* Operating voltage: 3.5-5.5V
* Output: serial
* Temperature range: -40°C – 80°C
* Humidity range: 0% - 100%
* Resolution: 16 bit
* Accuracy: ±0.5°C and ±1%

### Software Components

The microprocessor shall be programmed in Arduino programming language that is a c/c++ dialect using the Arduino IDE.

## Web Layer Components

The web layer shall mostly be concerned with software components. The following programming language and frameworks shall be used in the development of the web layer:

**HTML, CSS and Bootstrap –** these shall be used in the design of the website’s user interface (UI). Bootstrap shall be used to facilitate rapid development of the website while providing a responsive design to cater to different screen sizes (desktop or mobile).

**JavaScript –** this programming language shall be used to make the website responsive.

**Php –** this programming language shall be used to code the web service functionality.

**Twilio –** API to enable sending of SMS

**MySQL –** relational database management system (RDBMS) to store the system’s database.

The XAMPP stack shall be used to provide the web server, MySQL database and php support, enabling development and testing on a localhost environment rather than a blown-out production environment. This shall provide greater flexibility of development, testing and debugging of the project.

## User Layer Components

The user layer components would be the hardware and software components needed to run a web browser. Any browser, whether on desktop or mobile, on any operating system is to be supported.

# Implementation

## Sensor Layer

The DHT11 sensor and HC-SR04 ultrasonic sensor was connected to the nodeMCU 8266 development board. Since physically connecting and reading values from the flow rate sensor in the prototype FMWS system was a challenge, a potentiometer was used to simulate the fluctuations of values that would be expected in the field. A USB connection was used to power the nodeMCU 8266 development board and enable uploading and debugging of the device. A WiFi connection was used to provide internet access to the board. The sensing layer setup was as shown in figure 12 and 13.

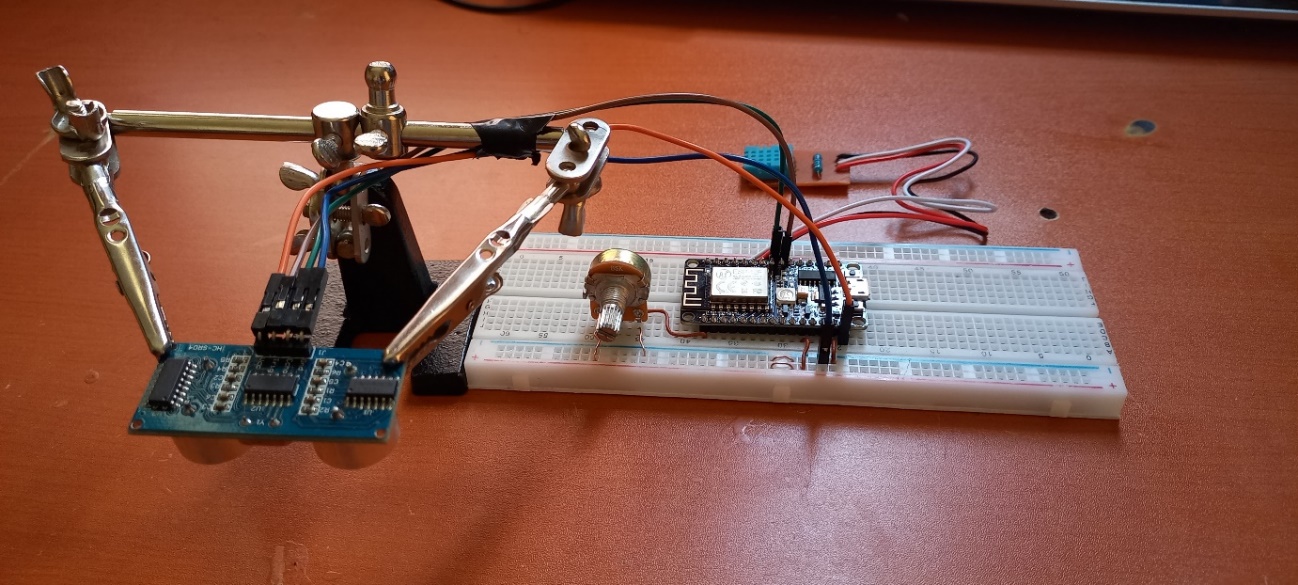


Figure 12: Sensing layer setup (oblique)

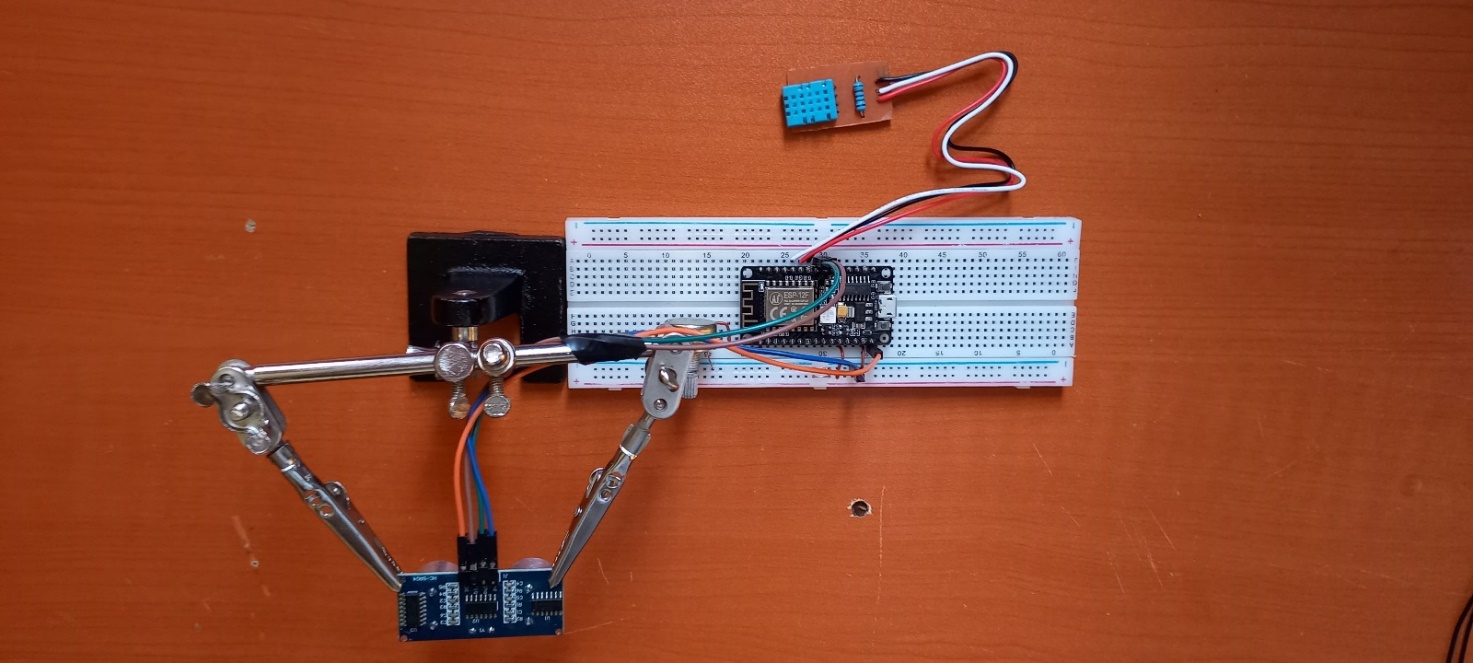


Figure 13: Sensing layer setup (aerial)

The 8266 microchip was programmed using the Arduino IDE. The pinout defined for the development board was as follows:

|  |  |
| --- | --- |
| **nodeMCU pin** | **Allocated to** |
| GPIO2 | DHT signal |
| GPIO12 | HC-SR04 Trigger |
| GPIO14 | HC-SR04 Echo |
| ADC0 | Potentiometer |

Table 1: nodeMCU pinout

The complete code is provided in the appendix item 1 of this report. The main functions of the program were:

|  |  |
| --- | --- |
| **Function** | **Purpose** |
| void setup() | Initialize Serial connection to PC, WiFi connection, DHT sensor and ultrasonic sensor |
| void loop() | Infinite loop that coordinated other functions. The loop: requested for distance, flow rate, humidity and temperature reading from their corresponding function, and passed these values as parameters to a function to post the data. The loop then slept, before beginning execution again. |
| Float getUltrasonicDistance() | Obtained distance between the sensor and water by using the time of flight principle. |
| Float getTemperature() | Read temperature from the DHT sensor |
| float getHumidity() | Read humidity from the DHT sensor |
| bool postData(float distance, float flowrate, float temperature, float humidity) | Received distance, flow rate, temperature and humidity sensor readings as parameters. The function then created a http connection to the web server, formatted a HTTP header and body with the parameters form encoded, and using the POST http method posted these. The function returned true if a 200 response code was received on posting the data, else it returned false. |

Table 2: ESP8266 functions and purpose

## Web Layer

Development and hosting of the FMWS website was done on the localhost environment by using the XAMPP stack. Apache hosted the website, the database was developed in MySQL and the website coded mostly from php. The website was accessible from the nodeMCU by virtue of them being on the same WiFi network.

Several php files were used in coding the website:

|  |  |
| --- | --- |
| **Function** | **Purpose** |
| config.php | Contained access credentials needed to connect to the MySQL database |
| login.php | Enabled user login. Admins were directed to restricted page on login inaccessible by ordinary users. Session variables were used to facilitate persistent sessions. |
| Logout.php | Logged a user out by deleting his/her session variable. |
| Sign\_up.php | Registered a user to the service under a particular location, allowing them to receive flood alerts. |
| Index.php | Displayed data logs and corresponding graphs for different areas for the sensor data associated with the area. |
| Guidelines.php | Provided safety, preparedness and guidelines on what to do in the case of an imminent, on-going on post flood incident. |
| Helpline.php | Provided a helpline accessible for support, counselling and evacuation services. |
| Post\_log.php | A web service that enabled authentication of incoming sensor data, storing the data on the database on successful authentication, and checking if the data presented an imminent on ongoing flood scenario.  Authentication was done on password basis. If an imminent flood was flagged, the web service sent out SMSs to users under that area if a prior alert had not been sent.  The service returned the following return code on the following events:   |  |  | | --- | --- | | **Return Code** | **Event** | | 200 | The end-point sending the sensor data was successfully authenticated, and data entered into the database | | 401 | The end-point sending the data failed authentication | | 500 | The web service failed to connect to the database | | 400 | The sensor data sent was badly formatted | |

Table 3: Web layer main files and purpose

The MySQL database used for this system had the following tables:

|  |  |
| --- | --- |
| **Table** | **Purpose** |
| device | Inventory of sensor devices deployed on field |
| location | List of the areas considered for monitoring using sensing devices |
| user | Storage of registered user information on the system |
| data\_log | Storage of sensor data values posted from deployed sensors |
| flood\_alert | Storage of flood alerts delivered for different regions |

Table 4: MySQL database tables and purpose

The relations and schema on the database is as shown in figure 14:

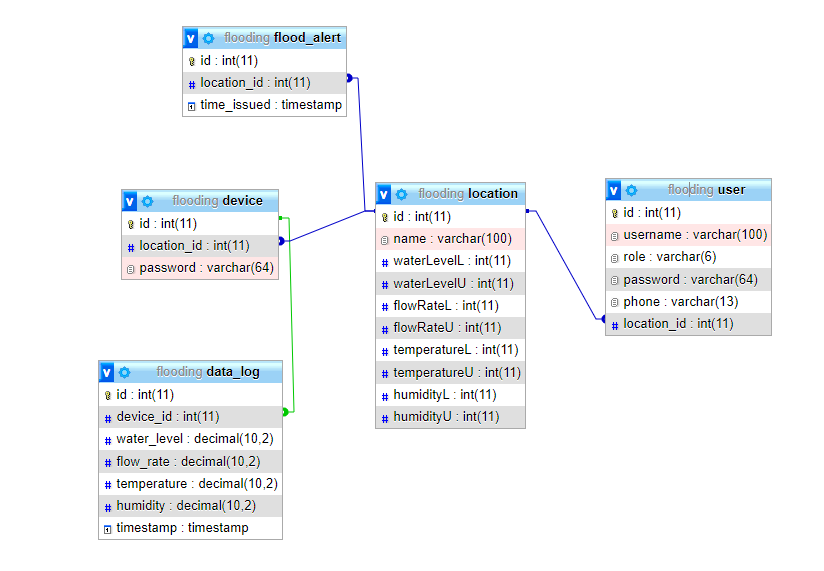


Figure 14: MySQL database schema

# Results

This chapter takes a sequential approach in the documentation of the results, from the registering and deployment of sensing devices, to the delivery of alerts to a registered user in the case of a flood.

## Administrator Preliminary Actions

**Admin login**

An admin (amdin1), allocated on the system on default was able to login into the FMWS as shown in figure 15.

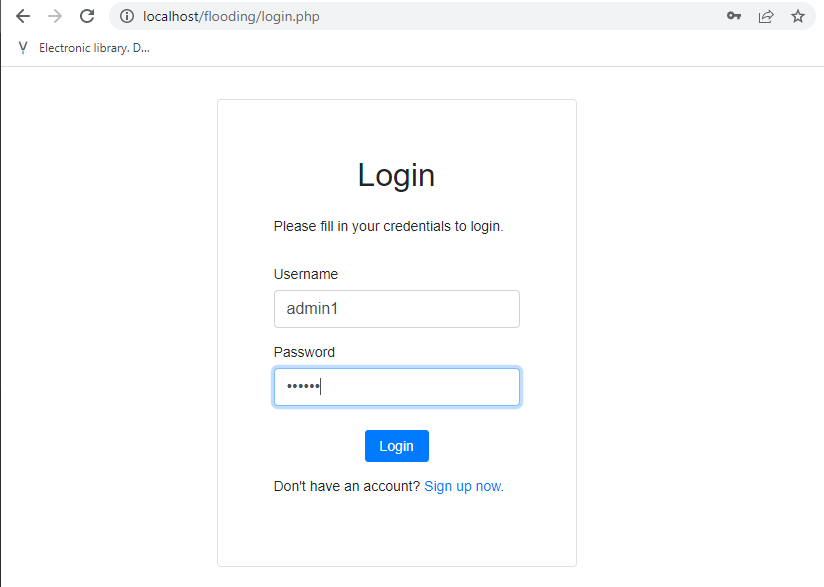


Figure 15: Login page

The administrator dashboard on login was as shown in figure 16. There were no locations created to be monitored or sensing devices registered to monitor the locations.

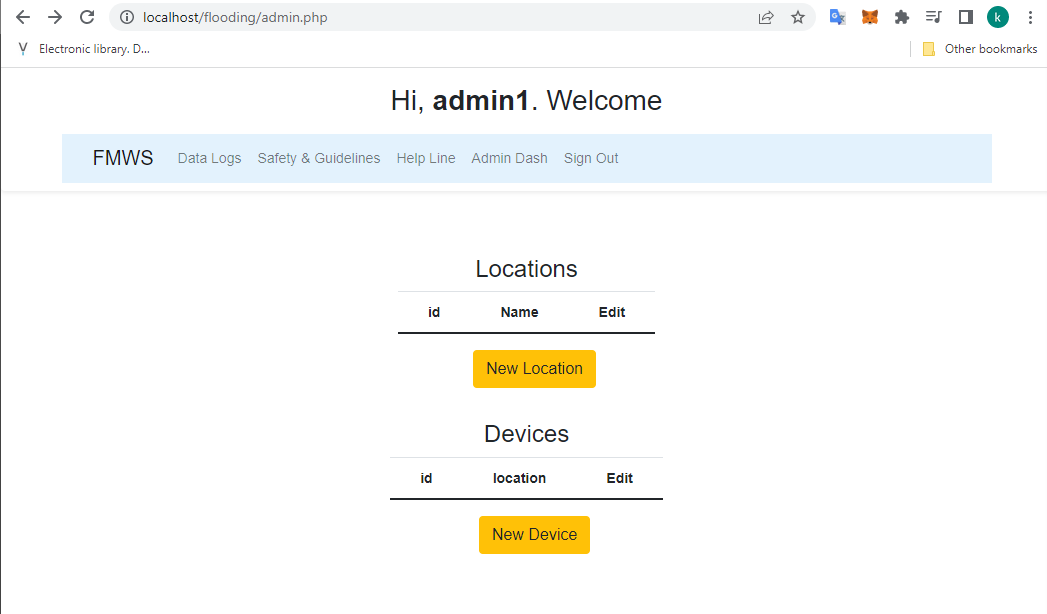


Figure 16: Admin dashboard (fresh)

**Creating a location**

It is expected that the FMWS be capable of monitoring the flood risks of multiple areas. As such, an administrator should be capable of defining a location on which a sensing device is to be deployed to monitor. In this case, an area named Lewe1 was created using the “create\_location.php” page. The limits and threshold for the area were not yet defined, since calibration for the area had not yet been done.

**Registering a sensing device**

It is expected that multiple sensing devices are to be deployed to the different regions being monitored by the FMWS. Authentication is also paramount in this, to ensuring that only valid end-points are capable of posting data to the FMWS. As such, sensing devices should be registered on the system. In this case, using the “create\_device.php” page, a sensing device was registered and allocated to the Lewes1 area that had been previously created as shown in figure 17. The field (Device Id) defined the unique ID of the sensing device, and the field (Device Password) defined the password of the device.

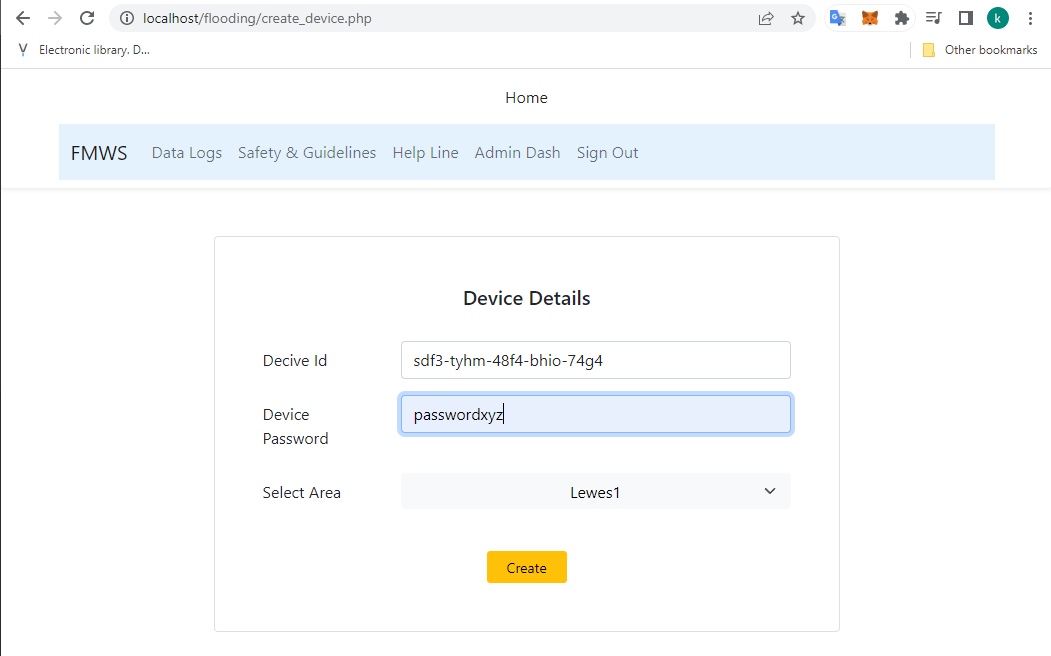


Figure 17: Registering sensing device

**Calibrating the sensing device to the area**

The threshold and limits defined for a particular area should be calibrated as per the environmental and water conditions of the area. This calibration is to involve the definition of water level and flow rate limits, and expected temperature and humidity.

In this run, in order calibrate the water level, the ultrasonic sensor was placed above an empty container of water. A different sketch was uploaded to the nodeMCU to continuously poll the distance from the ultrasonic sensor to the water body. The height noted when empty was 44.71cm as annotated red in figure 18. Water was filled to different levels until the container was full. The final height noted was 21.15cm as as annotated blue in figure 18. Of interest too, was an outlier value – 214.63cm as annotated green, during the calibration process.

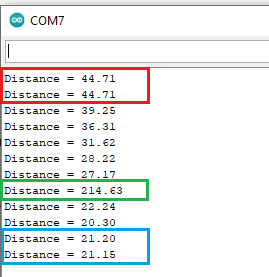
****

Figure 18: Water level calibration data

From these, a lower limit point of 32cm, just above the midpoint, was chosen to represent a point whereby a flood was deemed imminent. A critical limit 26cm was chosen to represent a point whereby a flood was in progress.

Water flow rate was simulated using a potentiometer. The potentiometer allowed a possible voltage variation of 0V – 3.3V. This range was mapped to values of range 0 – 100. The point 40 was chosen to represent a point were an imminent flood was due, and value 70 a point were a flood was in progress.

The temperature of the area was calibrated to have normal lower and upper limits of 2°C and 21°C as obtained from [23]. The relative humidity was calibrated in the range of 0 – 100 % as this parameter was not conclusively obtained in existing literature.

These values where updated for earlier created are by the admin as shown in figure 19.

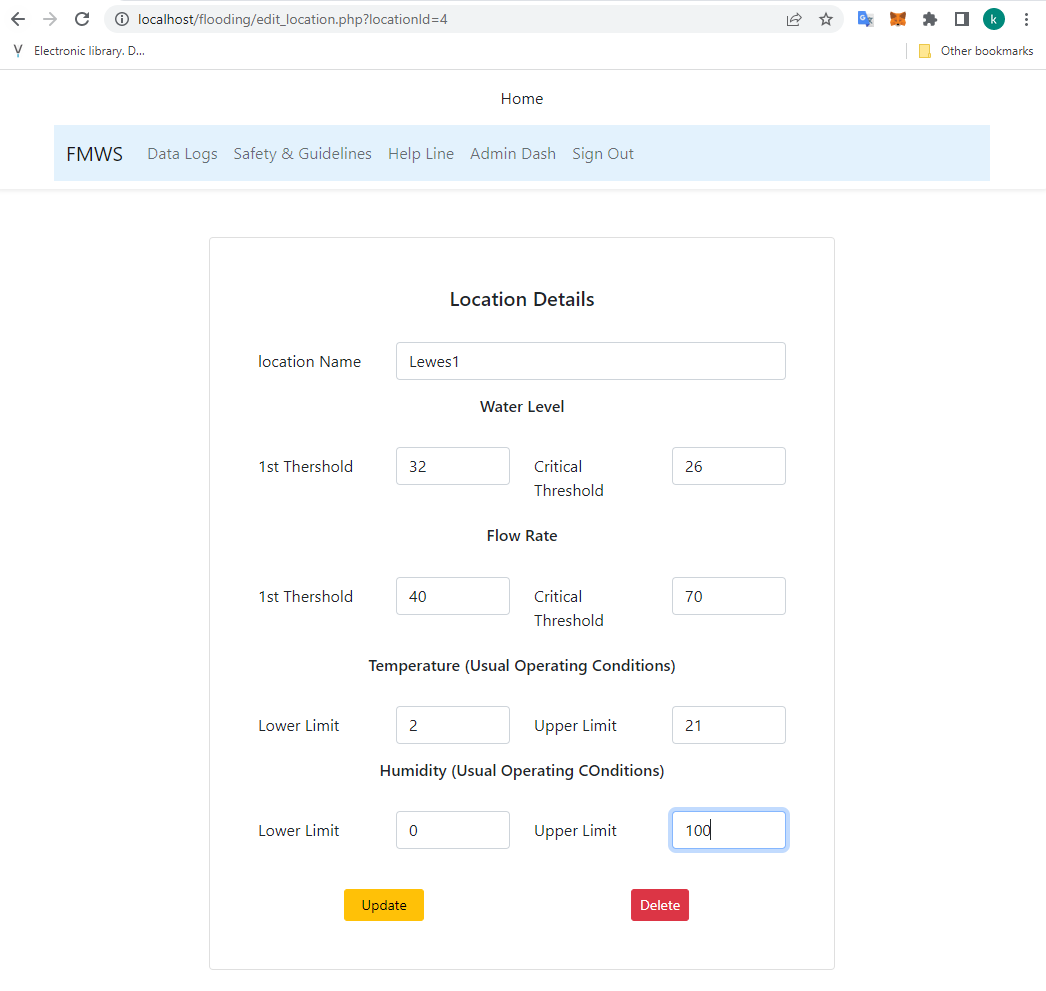
****

Figure 19: Updating location with values from calibration

The final view of the admin dashboard with the “Lewes1” location created and a device registered to monitor the area was as shown in figure 20.

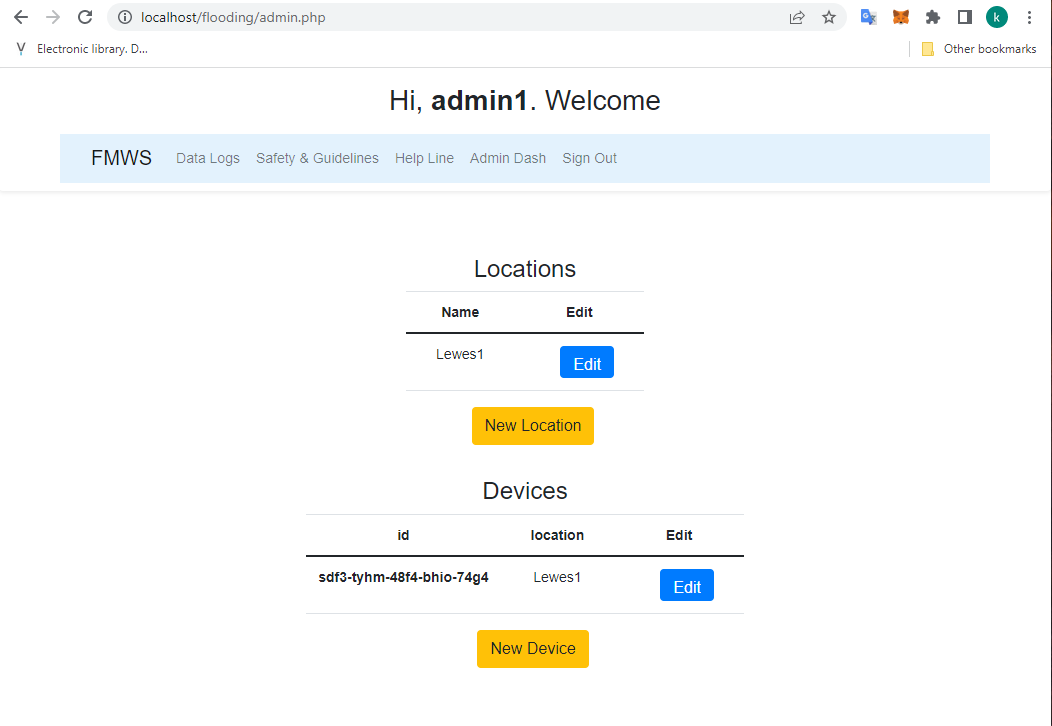


Figure 20: Admin dashboard (populated)

## Post-Calibration Operations

**Registering a User**

A test user registered to the FMWS. The user specified her username, password, phone number and area of stay as shown in figure 21.

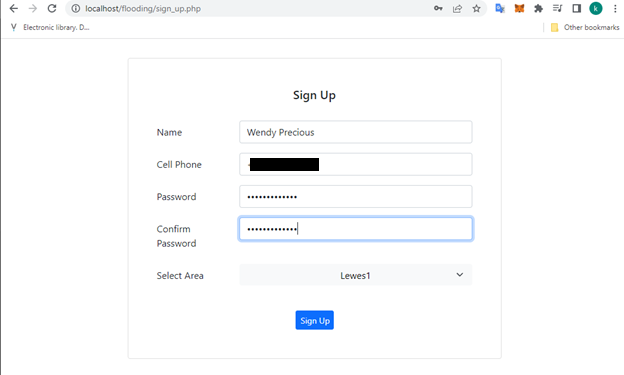


Figure 21: User signing up

**Simulating Varying Conditions**

The water level and simulated flow rate were varied to simulate varying conditions. The ultrasonic sensor of the sensing device was place on top of a bucket of water. The water container was first left empty for some while. Water was then progressively added to the container while increasing the potentiometer level to simulate water level rising and flow rate increase. This was done in stages until the container was full and flow rate close to maximum. Water was then progressively withdrawn from the container and the potentiometer wound down progressively until the container was empty and potentiometer close to origin. The data log posted is as shown in the appendix item 2, and the respective graph produced as shown in figure 22.

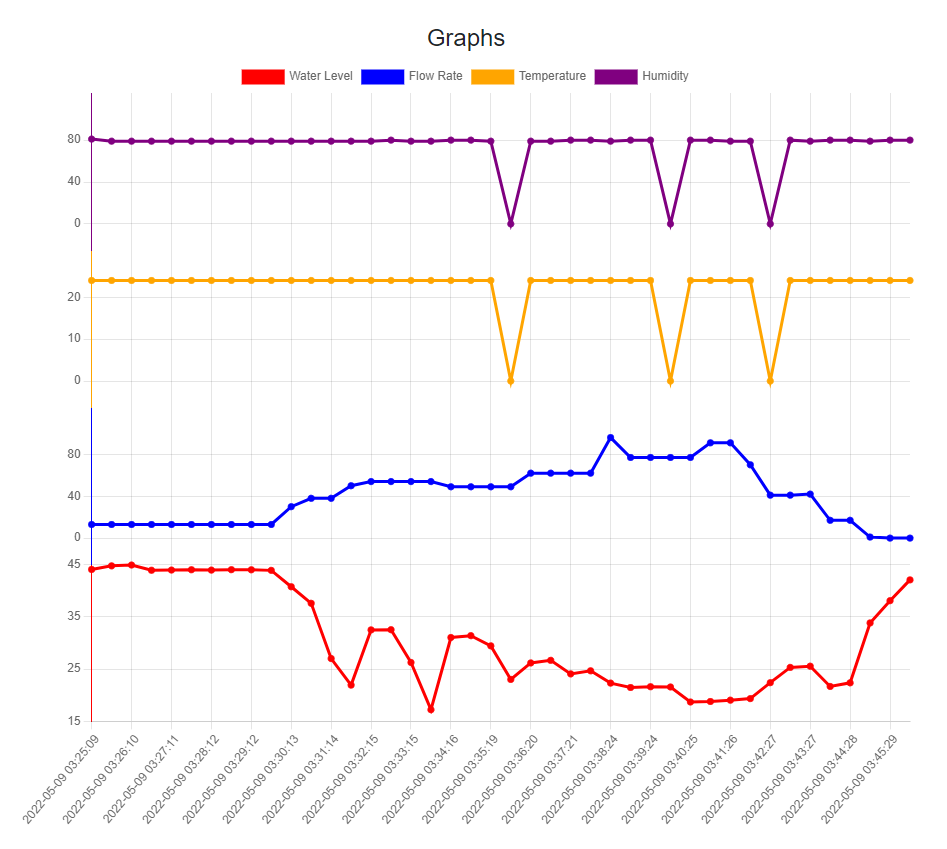


Figure 22: Graph for logged sensor data

**Receiving Flood Warning**

On increasing the water level and flow rate, the registered user received two SMSs on her phone: one warning her of an imminent flood event, and another alerting her of an ongoing flooding event. The first SMS was received at 03:34 AM, while the second SMS was received at 03:37 AM as shown in figure 23.



Figure 23: SMSs received

**Viewing Safety Guidelines**

The registered user was able to navigate to the website and view safety, preparedness and guidelines to consider in the event of a flood. The guidelines were divided into three: measures to consider in the case of an imminent flood alert, measures to adhere to in the case of an ongoing flood, and measures to consider after the flood as shown in figure 24.

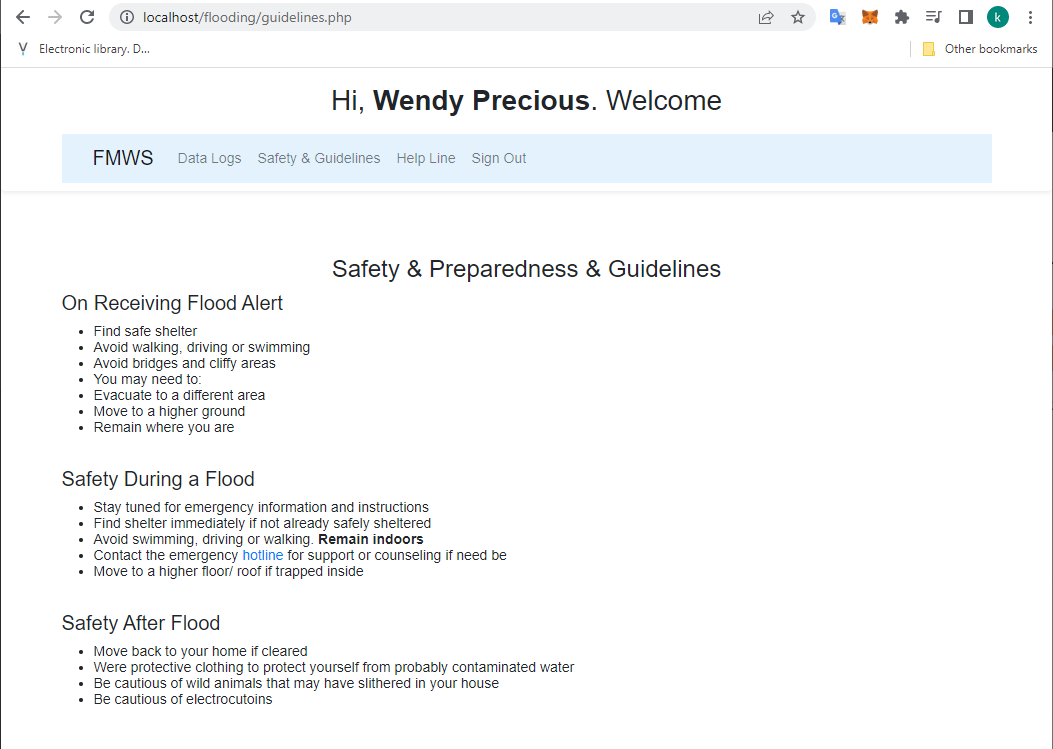


Figure 24: Safety, preparedness and guidelines

**Viewing Helpline**

The user was able to navigate to the website and view the helpline simulated for the FMWS as shown in figure 25. The services available via the helpline were listed, allowing the user to call for guidance and counselling, or evacuation support services.

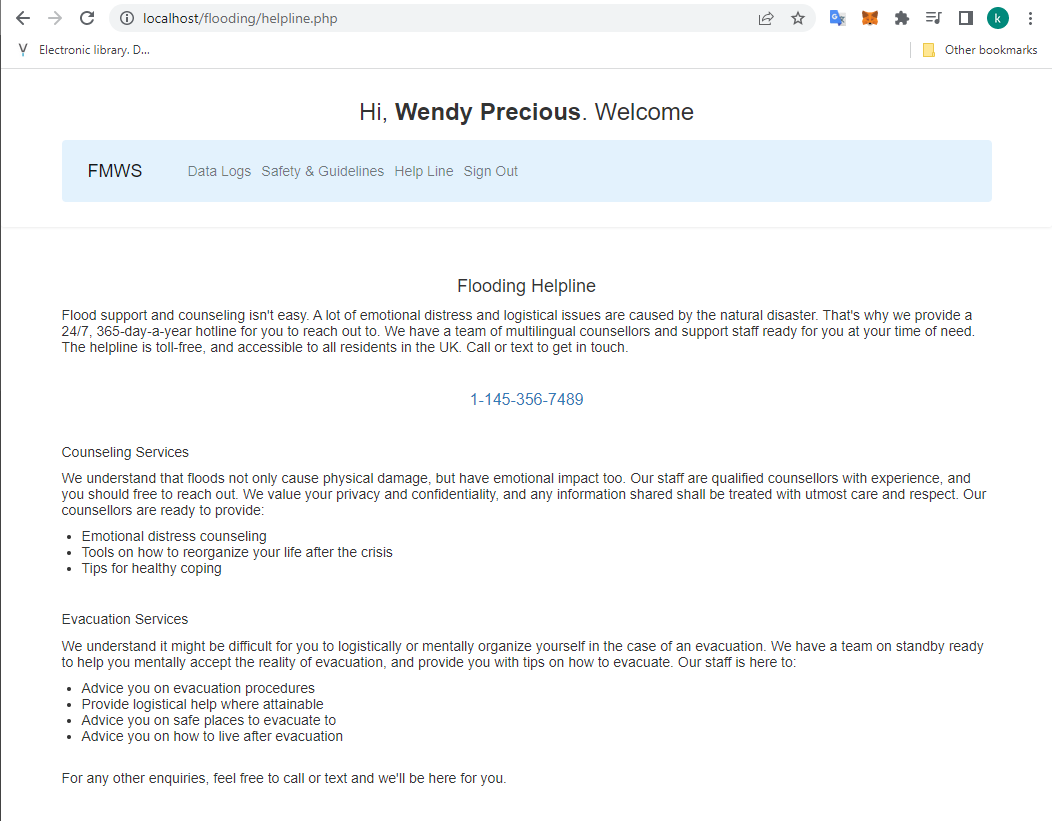
****

Figure 25: Helpline

# Discussion

This section analyzes the results obtained in the project from the perspective of the objectives set out for the project.

## Use of Sensors in FMWS

An earlier problem that had been discovered in the Literature Review chapter of the project was the unreliability of ultrasonic data measurement for obtaining accurate distances. This problem was experienced early in the calibration of the distance limits for the FMWS as shown in figure 18, where an anomalous reading of 214.63cm was encountered. As such, care had to be taken while handling this sensor. This project used a three-pronged approach to tackle this.

First, for any single time instance of reading sensor values from the different sensors connected to the microcontroller, 15 different ultrasonic reading were taken 50 microseconds apart and averaged as shown in the code snippet in figure 26. What this meant was that the effect of an anomaly would be evened out in the average value to be considered as distance from the ultrasonic sensor.

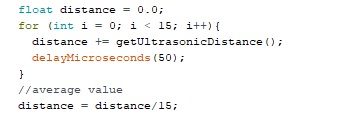


Figure 26: Averaging ultrasonic readings to mitigate anomalies

Second, the water flow rate sensor was used to complement the ultrasonic sensor. In order to determine if a flood was imminent, and the sensor values for water level and flow rate had crossed the critical threshold, but had crossed the lower limit, then both the water level and flow rate reading were checked as shown in figure 27. If both values were above the lower limit, an imminent flood was flagged. This action, in contrast to relying on just the ultrasonic sensor reading, was seen to improve the confidence with which an imminent flood could be flagged.

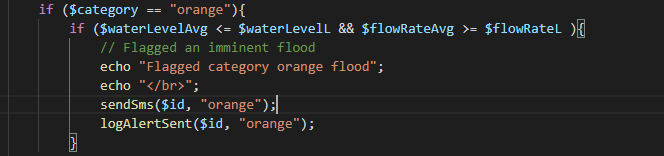


Figure 27: Combination of water level and flow rate to flag imminent flood

Finally, in order to flag whether a flood was imminent or ongoing, the data obtained from the sensor over the past 5 minutes was averaged. Considering that the sensing device sent out data updates every 30 seconds, this meant that any anomalous value captured within the 5-minute window would be even out from the previous data entries. This would in turn improve the confidence with which a flooding event could be flagged.

Temperature and humidity sensors were also used in this project. The sensors, however, were not critical in the flagging of a flood event.

## Microcontrollers in Embedded FMWS

The ESP 8266 microcontroller on the nodeMCU development board was used to read the sensor data, preprocess and transmit it to the web service. Programming of the microcontroller was conveniently achieved using the Arduino IDE. The performance of the microcontroller was consistent, and debugging easy. The development board facilitated rapid prototyping of the sensing layer.

The project was, however, unable to develop a casing for the sensing layer enclosing the microcontroller and sensors. Since the prototype was tested on a safe environment, this didn’t have any catastrophic effect on the execution of the project. Taken to a real-life case with the vagaries of weather, the prototype would have not sufficed. As such, it is seen advisable to design a housing unit of the sensing layer to protect it from the vagaries of weather.

## Web Services in FMWS

The web played an indispensable role in the development of this project. The design of the project envisioned that the FMWS system to be capable of monitoring different locations and respond appropriately to each. The web layer enabled this to happen.

An administrator was able to login to the FMWS website. The administrator was authorized to create a new location desired to be monitored for floods. The administrator was able to define the lower limit and critical thresholds for water level and flow rate for that area, together with normal temperature and humidity. The implication of this is that the FMWS could keep up with changing seasonal threshold demands.

The administrator was also capable to registering a sensing device with a particular 24byte ID and password on the system. The administrator could allocate the device to a desired location. What this meant was that sensing devices configured with unique IDs and passwords could be deployed to different locations according to demand. This also meant that the manufacture of the sensing devices could be outsourced, and the devices later registered to the system. This provides a more operational and financially feasible design.

Users were capable of registering with the FMWS, specifying the location of their residence and setting their phone numbers. This allowed the FMWS to send users under a particular area alert via SMS in the case of a flagged imminent or ongoing flood.

The data passed from the sensing devices to the web layer was stored on a MySQL database. This facilitated persistent storage, allowing reference to previous data in the decision-making of flood events. The data logged was displayed in tabulated form and graphically on the website. The graphical representation presented a visually aesthetic way to quickly make sense of the data logged, while the tabulated form allowed a more analytical/ empirical approach to it.

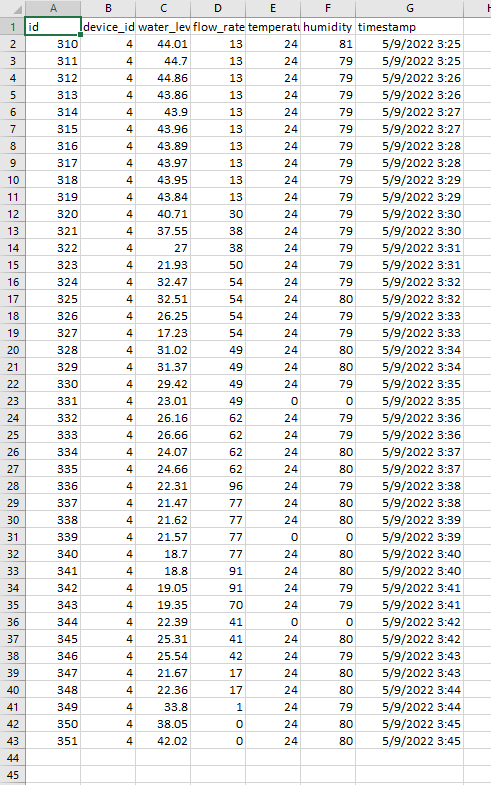
Through the website, individuals were also able to receive instructions and guidelines on how to conduct themselves in the case of an imminent or ongoing flood, and safety post floods. The website also allowed the display of a helpline conceived to be accessible 24/7 to aid in provision of guidance and counseling service, and evacuation help where needed.

By making use of the web, the policy implemented in the detection of floods was also made more accessible and flexible. This would have been in contrast to hard-coding the logic needed to monitor floods on the microcontroller – since this would have meant that on a real-life system where multiple sensing devices were to be deployed on the field, modification of logic would have been inefficient. In this design, the policy needed to flag floods needed only be coded once, and could be used to detect floods in different locations. In addition to this, the thresholds used in this process were easily accessible through the MySQL database in contrast to what would have been had they been hard coded on the microcontroller.

As such, the FMWS designed and implemented in this project is seen to be viable in large scale deployment. The division of the FMWS into the sensing layer as clients and the web layer as central server is seen to provide an elegant way to facilitating scalability of the system while maintaining operational and financial feasibility.

# Appendix – Redacted

**Sensor Data**

****