**FLOOD MONITORING AND ALERTING SYSTEM TINKERCAD SIMULATION**

College of Engineering and Architecture

**BOHOL ISLAND STATE UNIVERSITY**

Main Campus, Tagbilaran City, Bohol

Mark Darel B. Saligumba

Clifford Jay E. Dela Peña

Michael Jhon O. Madrona

Jesson D. Pitlo

**FLOOD MONITORING AND ALERTING SYSTEM TINKERCAD SIMULATION**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A Research Project Proposal Presented to the Faculty of the College of Engineering and Architecture

**BOHOL ISLAND STATE UNIVERSITY**

Main Campus, Tagbilaran City, Bohol

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In Partial Fulfillment of the Requirements of the Course

**CPE 03 - Microprocessor Systems**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mark Darel B. Saligumba

Clifford Jay E. Dela Peña

Michael Jhon O. Madrona

Jesson D. Pitlo

June 2022

**IMAGINATIVE ABSTRACT**

Flood is a major known natural calamity that causes significant damage to the environment and living beings. So, during these times of calamity, it is essential to have emergency alerts of water level status at river beds. In this project, the objective is to use an ultrasonic sensor to check the water level of the rivers to see if they are in normal condition or not. If they reach beyond the limit, then it alerts people through LED signals and buzzer sound. This system will display the data of the water level measured on LCD display. The device will use ultrasonic sensor to sense the river levels and will use Arduino Uno to process these data. Timely detection of floods which results in prevention of risks and accidents. With this project it can save many people's 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 happen every time.

**Chapter 1**

**INTRODUCTION**

Flooding is one of the most common and destructive calamities caused by a combination of natural and human factors such as heavy rainfall, typhoons, poor drainage system and other local factors. In 2020, flash floods caused the death of more than six thousand people across the globe (Alves, 2022). The Philippines is one of the most vulnerable to tropical storms and typhoons expecting 20 typhoons annually with an average of six to nine of them reaching landfall (Strobl 2019). The Philippines has 421 rivers in total around the country and since it is highly exposed to flooding due to several cyclones and heavy rainfall, the National Risk Reduction and Management Council (NDRRMC) are expecting reports of floods within the area of heavy rain condition.

Despite having National Warning Centres, there are still reports of floods that are not recorded especially in remote areas such as islands. Water levels rapidly rise in small and medium-sized rivers and do not leave enough time to issue warning alerts. A Flood Monitoring and Alerting System are frequently used to keep track of an increase in water levels. It is typically installed on bridges, measures water levels in real time, and sends data continually to a centralized data system.

In this study, it will focus on Flood Monitoring System and Alerting System using Arduino Uno as the microcontroller. It is open-source electronics that creates command to the other parts of the circuit. In this circuit, the Arduino is connected to an ultrasonic sensor to sense the water level in river beds, temperature sensor to predict the humidity level, buzzer and LEDs for alerting system, and Arduino IDE for the codes, upload programs and communication.

**Statement of the Problem**

The Philippines is highly exposed to flooding, the consequence of severe cyclones and heavy rainfall. The risks from flooding are worsen by land-use change such as urbanization and logging (The World Bank Group, 2021). On November 12-13, 2020, typhoon Ulysses lashed the main island of Luzon with heavy rainfall triggering extensive flooding affecting several regions. Tens of thousands of homes in low-lying suburbs of the national capital were submerged in roof-level floods, trapping residents in their rooftops while awaiting rescue (Office for the Coordination of Humanitarian Affairs Services, 2020). Local authorities in the Philippines called it the worst in the region in four decades. In these times of calamities, it is crucial to have a flood monitoring and alerting system to warn residents with the danger of an upcoming flood. Implementing an Arduino-based flood monitoring and alerting system is one of the most effective methods that can help prevent the possibility of flood victims.

**Objectives of the Study**

The following specific objectives will be undertaken:

* Design a circuit diagram of flood monitoring and alerting system using tinkercad simulation.
* Develop a code for the Arduino and Ultrasonic Sensor according to the desired work.
* Develop a working simulation of flood monitoring and alerting system using tinkercad simulation.

**Significance of the Study**

The study Flood Monitoring and Alerting System TinkerCad Simulation aims to provide proper information and knowledge. Furthermore, the study could be of importance to the following:

**Future Researchers.** The information presented may be used as reference data for the future researchers in conducting similar study. This study will be served as guide for further development and improvement.

**School Administration.** The product of this study would serve as a guide to the school service for making a local business program with the help of students of being a creative and resourceful student to any kind of research.

**Community.** This research will be beneficial to the community that is near a water source such as river and sea. Through this research, the researchers can help the community in alerting in case of a flood phenomenon.

**Electrical Students.** This study will help the electrical students to develop and enhance their knowledge and be more creative. It will also improve their analytical skills which can be used in the electrical industry.

**SCOPE AND LIMITATIONS**

This study aims to design and simulate a Flood Monitoring and Alerting System using Tinker Cad Simulation.

• This study focuses only on simulating Flood Monitoring and Alerting system using TinkerCad web application.

• This Flood Monitoring and Alerting System use HC-SR04 Ultrasonic Sensor as main sensor.

**Theoretical Background**

Flooding is one of the major disasters occurring in various parts of the world. With climate change, the frequency and severity of floods are projected to increase (Allen et al., 2019). Over the past decades, flood management has shifted from structural measures (i. e., physical flood protection structures) to non-structural measures, such as the distribution of flood warnings (UNISDR, 2018).

A local flood warning system typically serves a small community situated in a headwater area and exposed to flash floods or rapid riverine floods. Forecasts of such flood events are characterized by short lead times and large uncertainties. A Bayesian theory is formulated for a local flood warning system built of three functional components: monitor, forecaster, and decider. The theory offers a modelling framework and mathematical concepts necessary for (1) developing optimal decision rules for issuing warnings based on imperfect forecasts, (2) evaluating system performance statistically, and (3) computing the ex-ante economic benefits from a system (Zheng, 2021). Bayes' Theorem, named after 18th-century British mathematician Thomas Bayes, is a mathematical formula for determining conditional probability. Conditional probability is the likelihood of an outcome occurring, based on a previous outcome having occurred in similar circumstances. Bayes' theorem provides a way to revise existing predictions or theories (update probabilities) given new or additional evidence.

Formula,

A, B = events

P(A|B) = probability of A given B is true

P(B|A) = probability of B given A is true

P(A), P(B) = the independent probabilities of A and B

Alerts are characterized by a higher lead-time (i.e., the time between warning issuing and predicted impact) and/or lower predicted impact compared to emergency warnings. The latter are used for more severe flood forecasts and shorter lead-times (Golding, 2009). Flood warnings often fail to be received, understood or evoke adequate responses (O’Sullivan et al., 2012; Rollason et al., 2018; Sukhwani et al., 2019; Zhu et al., 2010). To reach all targeted audiences, both traditional media (e.g., radio and TV) as well as digital communication channels (e.g., web sites and applications, social media) should be employed for flood warning dissemination. The inadequacy of responses to flood warnings has two common causes: low individual risk perception and a lack of self-efficacy. Put simply, people underestimate the risk posed by floods and/or perceive themselves as unequipped to mitigate this risk. Thus, effective flood warnings should raise individual flood risk perception, hereby increasing the likelihood of recipients to follow recommended protective actions. To evoke adequate action among recipients, flood warnings should contain all the information required to take sufficient action. Besides the characteristics of a warning itself, an individual’s response to a flood warning is shaped by personal attributes (e.g., age, knowledge of hazard, trust in authorities) and situational factors (e.g., personal experience with floods, location of housing) that influence personal risk perception (Kellens et al., 2013; Lechowska, 2018; Wachinger et al., 2013).

**Definition of Terms**

**Alerting system.** A system that allows alert messaging, machine-to-person communication that is important or time sensitive.

**Arduino software (IDE).** An open-source software, which is used to programme the Arduino boards, and is an integrated development environment, developed by arduino.cc. It allows to write and upload code to Arduino boards

**Arduino uno.** A low-cost, flexible, and easy-to-use programmable open-source microcontroller board that can be integrated into a variety of electronic projects. This board can be interfaced with other Arduino boards, Arduino shields, Raspberry Pi boards and can control relays, LEDs, servos, and motors as an output.

**Emitter.** Emitter terminal is the heavily doped region as compared two base and collector. The work of the emitter is to supply charge carrier to the collector via the base.

**Flood Monitoring.** A system which monitors water levels in remote sites in rivers, creeks, canals and roads.

**LED.** Short for light-emitting diode. An electronic semiconductor device that emits light when an electric current passes through it.

**LM35 temperature sensor.** A temperature measuring device having an analog output voltage proportional to the temperature. It provides output voltage in Centigrade (Celsius).

**Microcontroller.** A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system.

**Programming.** The implementation of logic to facilitate specified computing operations and functionality.

**Receiver.** A hardware module or device used to receive signals of different kinds, depending on the context of the application.

**Smart Sensors.** Devices that take information from a physical environment and use embedded microprocessors and wireless communication to monitor, examine, and maintain various systems.

**Ultrasonic Sensors.** Sensors that measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target.

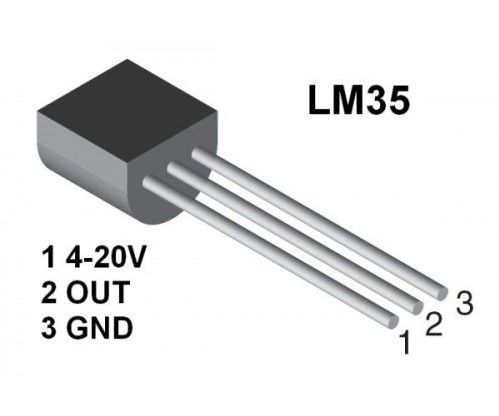
**Chapter 2**

**REVIEW OF THE RELATED LITERATURE**

Constructing a flood monitoring and alerting system is the main goal for this research. With that, this chapter involves a discussion of related literature and articles that will help to broaden the knowledge of the researchers in conducting this research.

2.1 LM35 Temperature Sensor

The LM35 (Figure 2.1) is a temperature sensor that produces an analog signal proportional to the current temperature. The output voltage can be easily interpreted to generate a temperature reading in degrees Celsius. The LM35 has an advantage over thermistors in that it does not require external calibration. It is also protected from self-heating by the coating. It can measure temperatures ranging from -55 to 150 degrees Celsius. When operated at optimal temperature and humidity levels, the accuracy level is high. The conversion of the output voltage to centigrade is also easy and straightforward.

**Figure 2.1**. **LM35 Temperature Sensor**

**2.2 LM35 Features and Specifications**

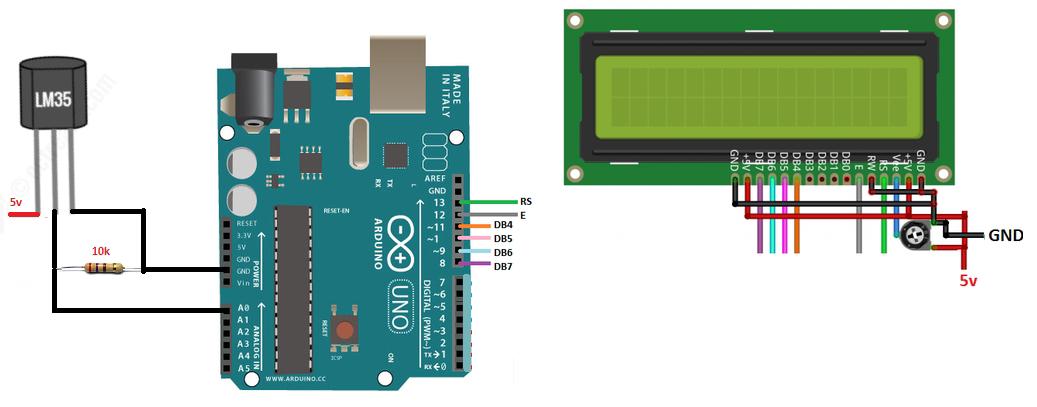
In Table 1. It shows the parameters of the LM35 Temperature Sensor. The Output voltage is directly proportional to Temperature i.e., there will be rise of 10mV or 0.01V for every 1°C rise in temperature.

|  |  |
| --- | --- |
| Temperature-Voltage scale factor | +10 mV/°C |
| Measurement range | -55 °C to 150 °C |
| Supply Voltage | 4V – 30V |
| Current drain | 60 μA |
| Self-heating | 0.08 °C |
| Accuracy | ±3/4°C |
| Package | TO - 92 |

**Table 1.** **LM35 Temperature Sensor Parameters**

* 1. **LM35 Temperature Sensor with Arduino**

It is very simple to measure the temperature if a location using an Arduino and any of the commercial temperature sensors available. [Figure 2](file:///C:\Users\Richmond\Downloads\CPE-Yoya.docx#fig2).2 shows the pin configuration of the sensor with the Arduino uno. The input pin in the Circuit is the Analog pin A0 of the Arduino and connect the LM35 output pin. The +5V is applied to VCC pin of the sensor and ground the *Gnd* pin. The detected temperature is printed in the 16×2-character lcd.

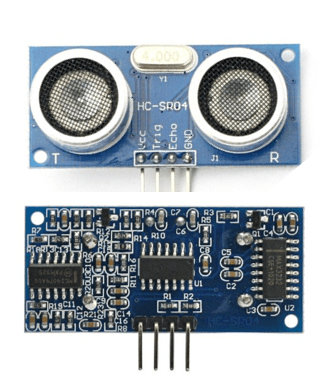


**Figure 2.2. LM35 Temperature Sensor, 16×2 LCD with Arduino**

* 1. **HC-SR04 Ultrasonic Sensor**

The HC-SR04 ultrasonic sensor (like the one shown in [Figure](file:///C:\Users\Richmond\Downloads\Objectives-of-the-Stud1.docx#fig21) 2.3) uses SONAR to determine the distance of an object just like the bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package from 2 cm to 400 cm or 1” to 13 feet. The operation is not affected by sunlight or black material, although acoustically, soft materials like cloth can be difficult to detect. It comes complete with ultrasonic transmitter and receiver module.

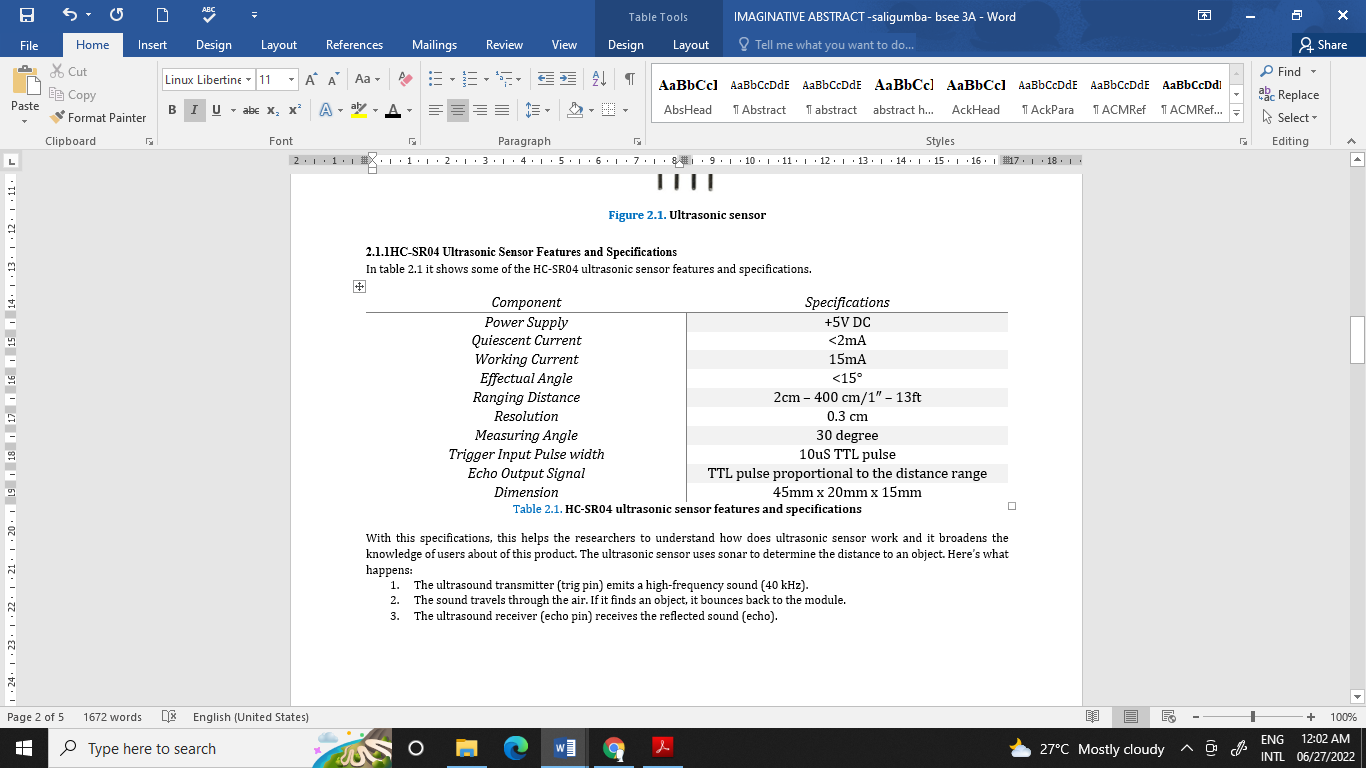
The ultrasonic sensor uses the reflection of sound in obtaining the time between the wave sent and the wave received. It usually sent a wave at the transmission terminal and receive the reflected waves. The time taken is used together with the normal speed of sound in air (340ms-1) to determine the distance between the sensor and the obstacle. The Ultrasonic sensor has been used by several researchers to sense the movements of the objects as they approach it.



**Figure 2.3. Ultrasonic sensor**

* 1. **HC-SR04 Ultrasonic Sensor Features and Specifications.**

In table 2, it shows some of the HC-SR04 ultrasonic sensor features and specifications.

****

**Table 2. HC-SR04 ultrasonic sensor features and specifications**

With these specifications, this helps the researchers to understand how does ultrasonic sensor work and it broadens the knowledge of users about of this product. The ultrasonic sensor uses sonar to determine the distance to an object. Here’s what happens:

1. The ultrasound transmitter (trig pin) emits a high-frequency sound (40 kHz).
2. The sound travels through the air. If it finds an object, it bounces back to the module.
3. The ultrasound receiver (echo pin) receives the reflected sound (echo).

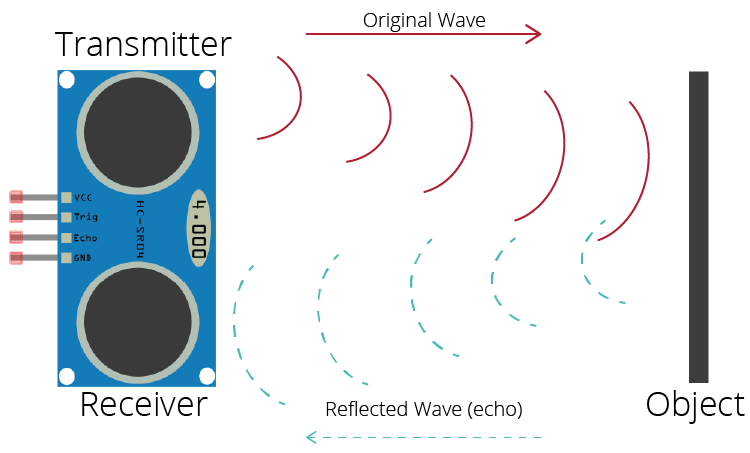
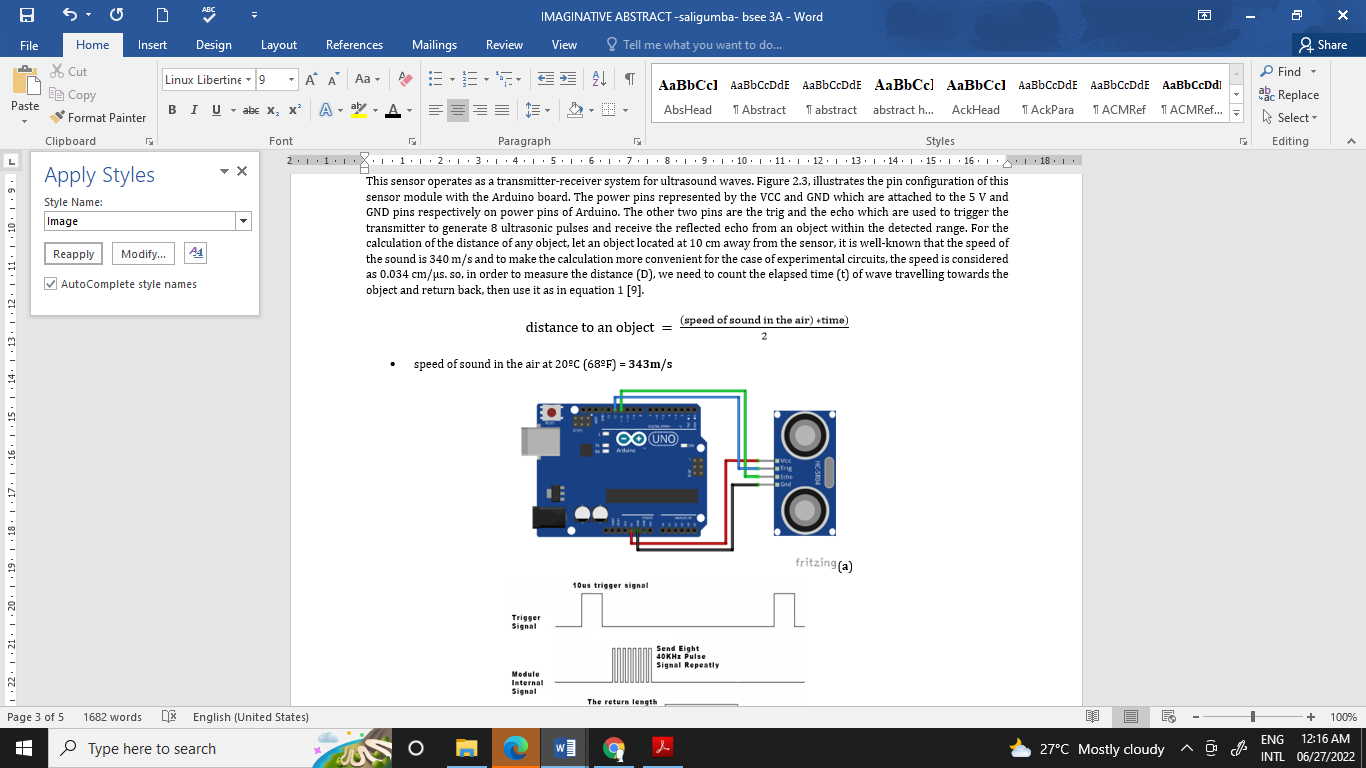


Figure 2.4. Shows how Ultrasonic sensor work

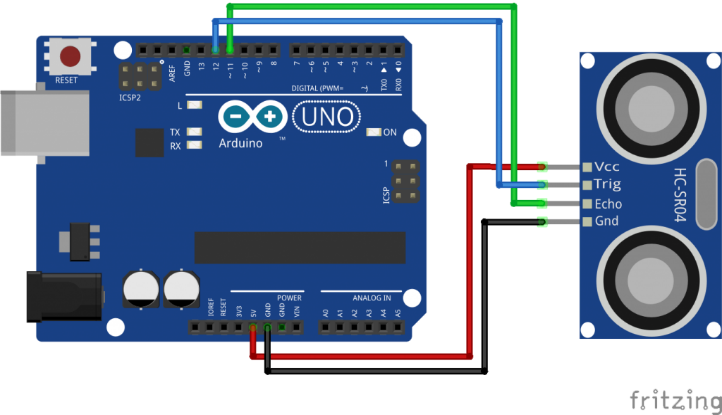
Figure 2.4 shows how ultrasonic sensor work, ultrasonic sensor uses the reflection of sound in obtaining the time between the wave sent and the wave received. It usually sent a wave at the transmission terminal and receive the reflected waves.

* 1. **Configuration of HC-SRO4 ultrasonic sensor with Arduino**

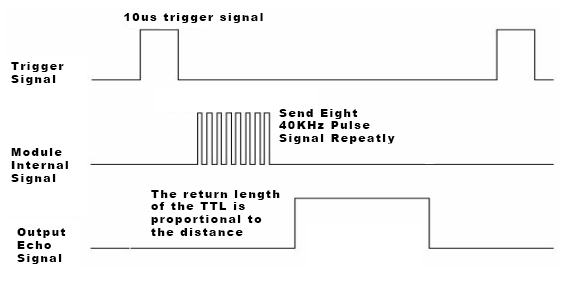
This sensor operates as a transmitter-receiver system for ultrasound waves. [Figure 2.6](file:///C:\Users\Richmond\Downloads\Objectives-of-the-Stud1.docx#fig24a) illustrates the pin configuration of this sensor module with the Arduino board. The power pins represented by the VCC and GND which are attached to the 5 V and GND pins respectively on power pins of Arduino. The other two pins are the trig and the echo which are used to trigger the transmitter to generate 8 ultrasonic pulses and receive the reflected echo from an object within the detected range. For the calculation of the distance of any object, let an object located at 10 cm away from the sensor, it is well-known that the speed of the sound is 340 m/s and to make the calculation more convenient for the case of experimental circuits, the speed is considered as 0.034 cm/µs. so, in order to measure the distance (D), we need to count the elapsed time (t) of wave travelling towards the object and return back, then use it as in equation from Figure 2.5.



**Figure 2.5. Equation of Distance to an Object**



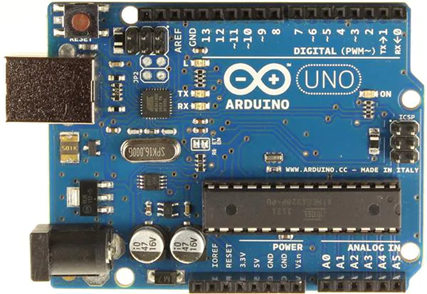
**Figure 2.6.a. Ultrasonic Sensor module connections**



**Figure 2.6.b. Shows Ultrasonic Sensor module operation principle**

**2.7 Arduino Development Board**

Arduino Uno is a microcontroller development board based on the Atmel ATmega328 MCU. The Arduino Uno has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. This Arduino MCU board contains everything needed to support the microcontroller. Simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The Arduino Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter. The Arduino Uno MCU board can be powered via the USB connection or with an external power supply. The power source is selected automatically. Figure 2.7 shows an Arduino Uno microcontroller board.

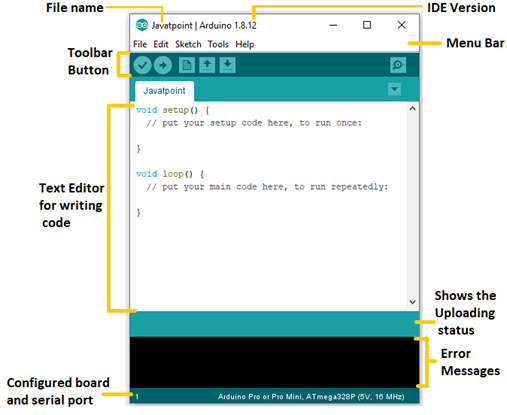


**Figure 2.7. Arduino Uno microcontroller board**

* 1. **Arduino Integrated Development Environment (IDE)**

The Arduino IDE is an easy to install free software, which is used to write and upload code to the Arduino boards. The IDE application can be used for different operating systems such as Windows, Mac OS X, and Linux. It supports the programming languages C and C++. Sketching is the term use for the program or code written in the Arduino IDE.

The Arduino IDE will appear as:



**2.9 Flood Disaster Indicator of Water Level Monitoring System**

The early warning systems for flood management have been developed rapidly with the growth of technologies. These systems help to alert people early with the used of Short Message Service (SMS) via Global System for Mobile Communications (GSM). This paper presents a simple, portable and low cost of early warning system using Arduino board, which is used to control the whole system and GSM shields to send the data. System has been designed and implemented based on two components which is hardware and software. The model determines the water level using float switch sensors, then it analyzes the collected data and determine the type of danger present. The detected level is translated into an alert message and sent to the user. The GSM network is used to connect the overall system units via SMS. (Wan Haszerila Wan Hassan, Aiman Zakwan Jidin, Siti Asma Che Aziz, Norain Rahim 2018).

* 1. **Flood Early Warning Systems Using Machine Learning Techniques: The Case of the Tomebamba Catchment at the Southern Andes of Ecuador**

Worldwide, machine learning (ML) is increasingly being used for developing ﬂood early warning systems (FEWSs). However, previous studies have not focused on establishing a methodology for determining the most efﬁcient ML technique. We assessed FEWSs with three river states, No-alert,Pre-alert and Alert for ﬂooding, for lead times between 1 to 12 h using the most common ML techniques, such as multi-layer perceptron (MLP), logistic regression (LR), K-nearest neighbors (KNN), naive Bayes (NB), and random forest (RF). The Tomebamba catchment in the tropical Andes of Ecuador was selected as a case study. For all lead times, MLP models achieve the highest performance followed by LR, with f1-macro (log-loss) scores of 0.82 (0.09) and 0.46 (0.20) for the 1 h and 12 h cases, respectively. The ranking was highly variable for the remaining ML techniques. According to the g-mean, LR models correctly forecast and show more stability at all states, while the MLP models perform better in the Pre-alert and Alert states. The proposed methodology for selecting the optimal ML technique for a FEWS can be extrapolated to other case studies. Future efforts are recommended to enhance the input data representation and develop communication applications to boost the awareness of society of ﬂoods. (Hydrology, 2021).

**Chapter 3**

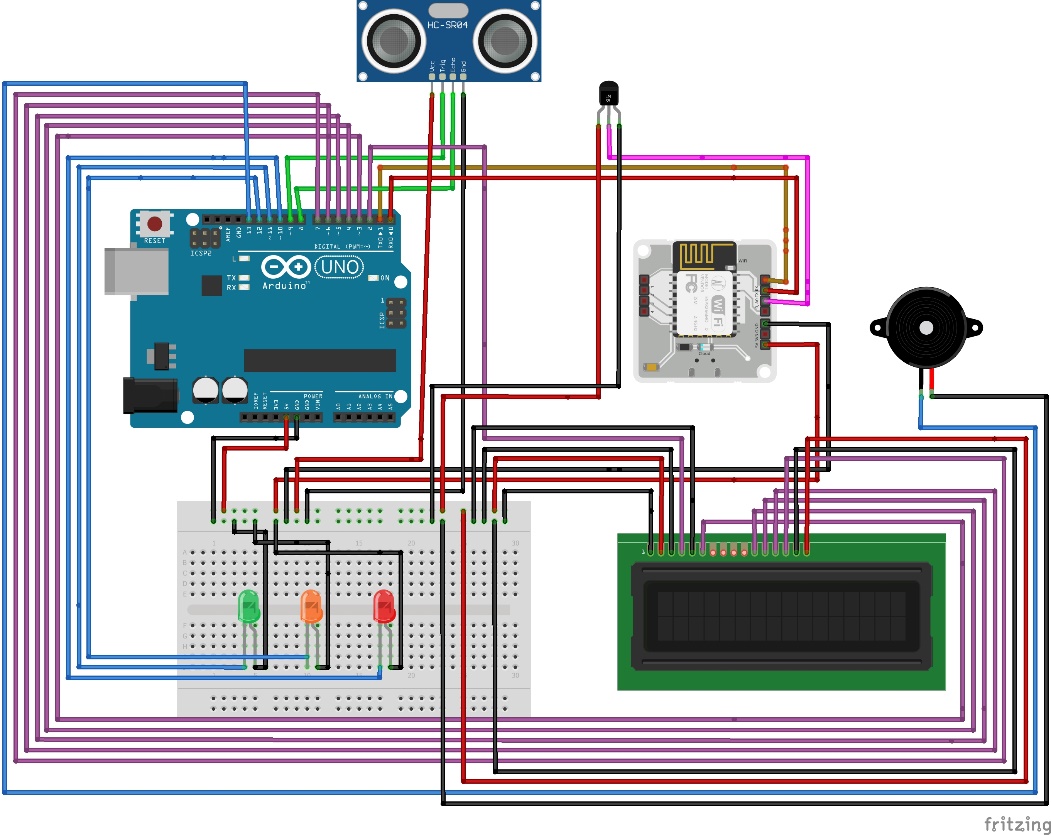
**PROPOSED METHODOLOGY**

The ultimate aim is to build a water level detection using ultrasonic sensor to monitor the rivers in a community and develop a web and SMS application as an early warning system that provides essential information to the local communities and concern agencies.

An SMS approach was used for transmitting data from the monitoring system to the computer server and for sending notification to the concern stakeholders. The SMS application was installed in the computer server to process the received data and make proper action. The application also implements fuzzy logic algorithm for decision making. The inputs of the algorithm are the water level status coming from the two monitoring systems sent through SMS. A threshold value was set in the two-monitoring system as basis for the Arduino to trigger the GSM module to send an SMS to the computer server. Then the developed program installed in the computer server send an SMS notification to the concern stakeholders and uploads an update post in the developed web-based monitoring system. After the development of the prototype, the model had undergone several tests and experimentations to check the effectiveness of the system.

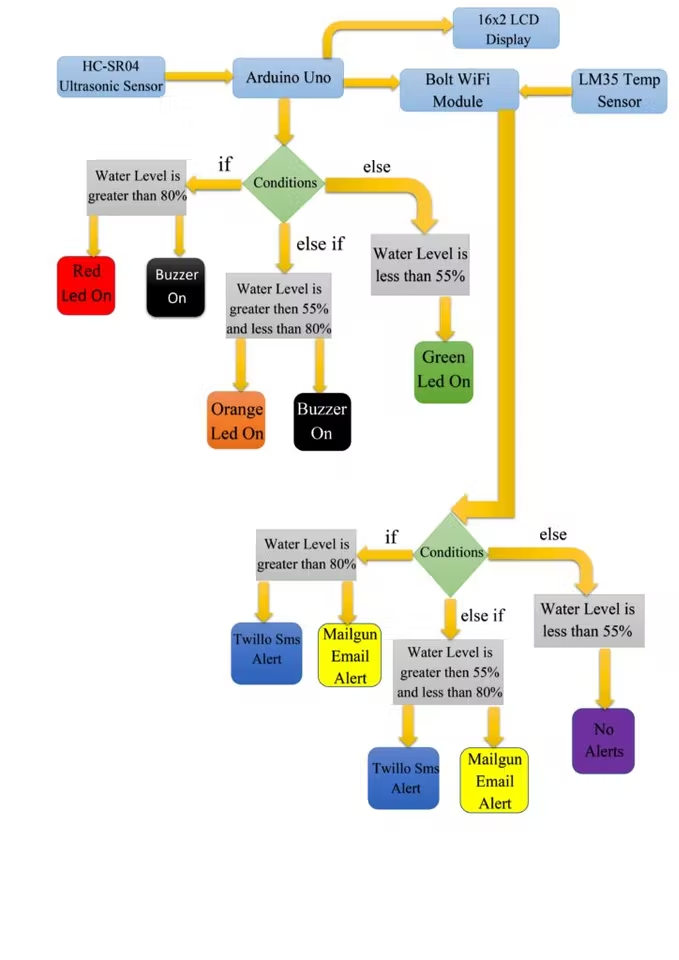
* 1. **System Design**

As everyone is aware, flooding is one of the most common and well-known natural disasters. When the water level in dams, river beds, etc. suddenly rises, a lot of destruction on the surroundings areas. Both our environment and other living things suffer tremendous losses as a result. Therefore, in these circumstances, receiving emergency notifications about the water level situation in various river bed conditions is extremely crucial. The flood monitoring and alerting system will be simulated by the researchers on the Tinker Cad website (<https://www.tinkercad.com/>). The system utilizes Ultrasonic Sensors that will measure distance by using ultrasonic waves. The system will use a Wi-Fi module to connect to network and upload real-time data to a cloud server. If the ultrasonic sensors detect that the water reaches 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.

****

**Figure 3.1. Pictorial Circuit Diagram of the Finished System**

* 1. **Workflow of the Project**

****

**Figure 3.2. Project Workflow**

* 1. **System Components**

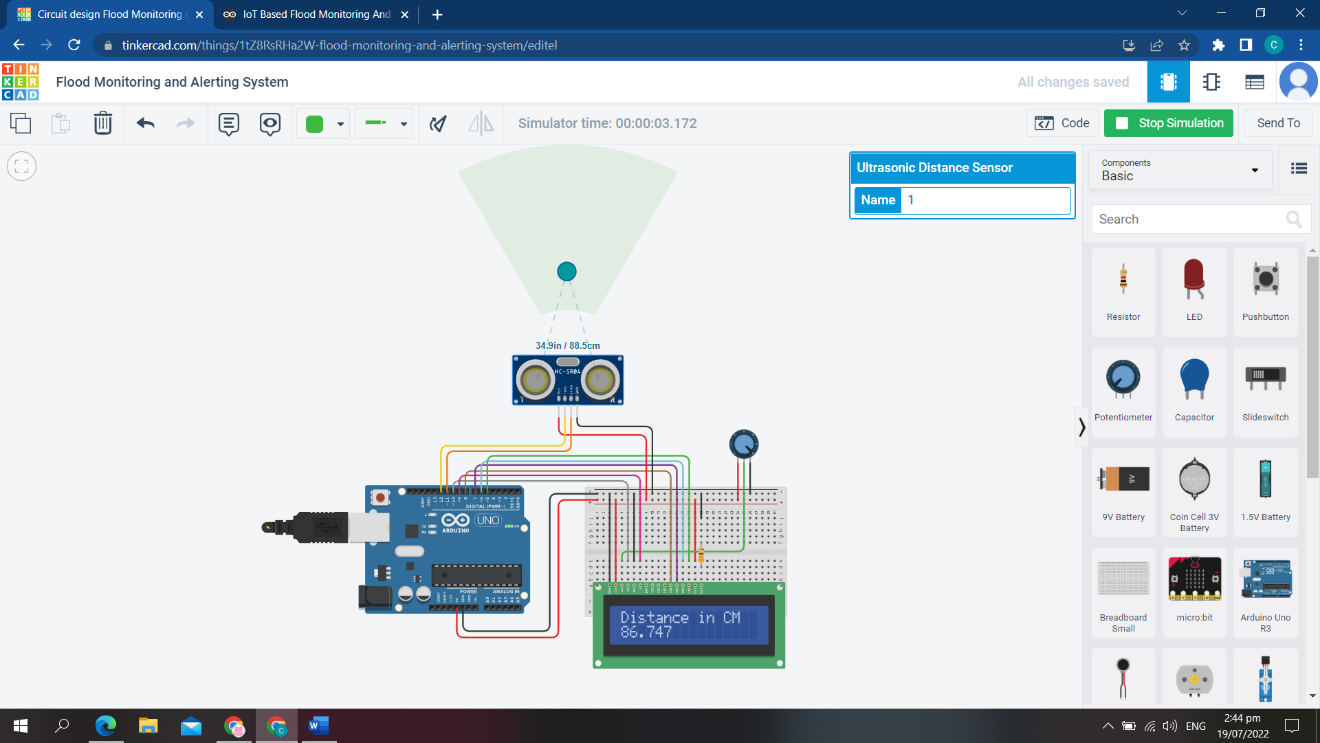
This part of the chapter will discuss what components will be use for the flood monitoring and alerting system. Since some components or electronic parts are not available in Tinker Cad website, the system components portion of this chapter will be divided into two namely: system components for flood monitoring and alerting system; and system components for tinker simulation.

* + 1. **System Components for Flood Monitoring and Alerting System**

|  |  |
| --- | --- |
| **Component** | **Quantity** |
| Bolt IOT Bolt Wi-Fi Module | 1 |
| Arduino Uno | 1 |
| Breadboard | 1 |
| 5 mm LED Green | 1 |
| 5 mm LED Red | 1 |
| 5 mm LED Orange | 1 |
| Buzzer | 1 |
| 16x2 LCD | 1 |
| Temperature sensor | 1 |
| Ultrasonic sensor | 1 |
| Male/female jumper wires | 15 |
| Male/male jumper wires | 10 |
| Female/female jumper wires | 5 |
| 9V battery | 1 |
| USB A-to-B cable | 1 |

* + 1. **System Components for Tinker Cad Simulation**

|  |  |
| --- | --- |
| **Component** | **Quantity** |
| Arduino Uno R3 | 1 |
| Ultrasonic Distance Sensor | 1 |
| LCD 16x2 | 1 |
| Breadboard | 1 |
| 330-ohm resistor | 1 |
| Potentiometer | 1 |

****

**Figure 3.3. Tinker Cad Simulation**

* 1. **Procedure**

This section will give an insight on how to set-up the hardware components for the flood monitoring and alerting system.

Step 1: Connecting 5v and GND of Arduino to the Breadboard for power connection to other components.

Step 2: Connecting LED’s

For Green LED:

* VCC of Green Colour LED to Digital Pin ‘10’ of the Arduino.
* GND of Green Colour LED to the GND of Arduino.

For Orange LED:

* VCC of Orange Colour LED to Digital Pin ‘11’ of the Arduino.
* GND of Orange Colour LED to the GND of Arduino.

For Red LED:

* VCC of Red Colour LED to Digital Pin ‘12’ of the Arduino.
* GND of Red Colour LED to the GND of Arduino.

Step 3: Connecting Buzzer

* VCC of Buzzer to Digital Pin ‘13’ of the Arduino.
* GND of Buzzer to the GND of Arduino.

Step 4: Connecting HC-SR04 Ultrasonic Sensor

* VCC of Ultrasonic Sensor to 5v of Arduino.
* GND of Ultrasonic Sensor to GND of Arduino.
* Echo of Ultrasonic Sensor to Digital Pin ‘8’ of Arduino.
* Trig of Ultrasonic Sensor to Digital Pin ‘9’ of Arduino.

Step 5: Connecting Bolt WiFi Module

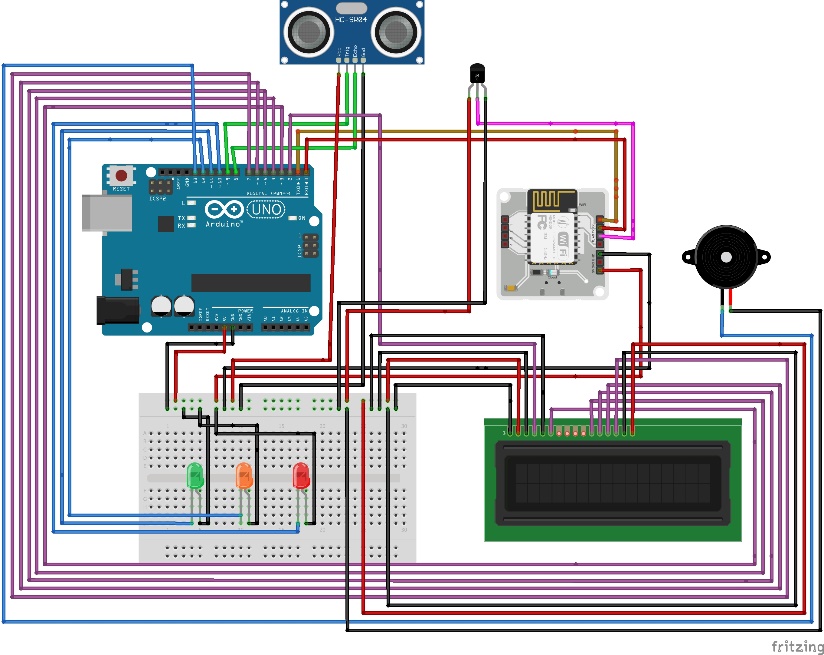
* 5v of Bolt WiFi Module to 5v of Arduino.
* GND of Bolt WiFi Module to GND of Arduino.
* TX of Bolt WiFi Module to RX of Arduino.
* RX of Bolt WiFi Module to TX of Arduino.

Step 6: Connecting LM35 Temperature Sensor

* VCC of LM35 to 5v of Bolt WiFi Module.
* Output Pin of LM35 to Pin ‘A0’ of Bolt WiFi Module.
* GND of LM35 to GND of Bolt WiFi Module.

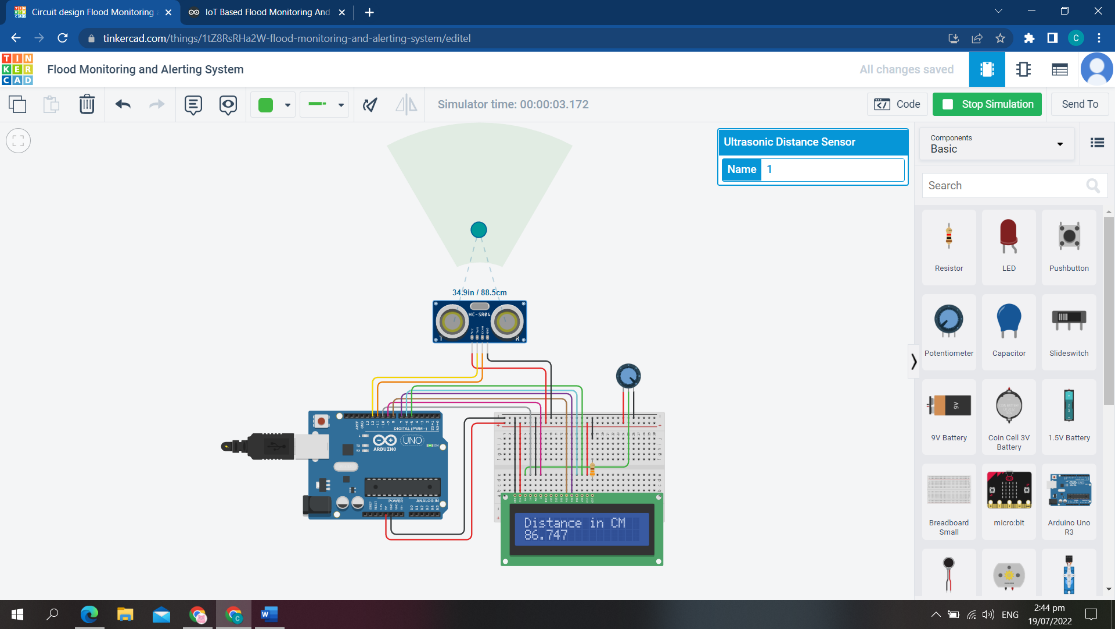
Step 7: Connecting 16x2 LCD Display

* Pin 1, 3, 5, 16 of 16x2 LCD to GND of Arduino.
* Pin 2, 15 of 16x2 LCD to 5v of Arduino.
* Pin 4 of 16x2 LCD to Digital Pin ‘2’ of Arduino.
* Pin 6 of 16x2 LCD to Digital Pin ‘3’ of Arduino.
* Pin 11 of 16x2 LCD to Digital Pin ‘4’ of Arduino.
* Pin 12 of 16x2 LCD to Digital Pin ‘5’ of Arduino.
* Pin 13 of 16x2 LCD to Digital Pin ‘6’ of Arduino.
* Pin 14 of 16x2 LCD to Digital Pin ‘7’ of Arduino.

****

**Figure 3.4. Actual Wiring of Components for Flood Monitoring and Alerting System**

Note that in the Tinker CAD website, some components are not available for the simulation so the researchers simulate only the working principle on how the ultrasonic sensors detect the distance/level of water. Figure 3.5 shows the simulation.

****

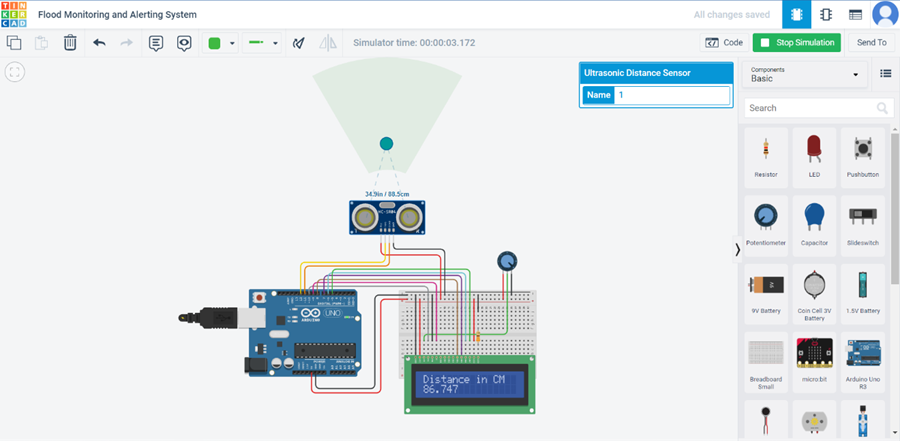
**Figure 3.5. Tinker Cad Simulation**

**Reference List**

* Allen, M., Antwi-Agyei, P., Aragon-Durand, F., Babiker, M., Bertoldi, P., Bind, M., et al., 2019. Technical Summary: Global warming of 1.5◦ C. An IPCC Special Report on the impacts of global warming of 1.5◦ C above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty.
* Alves, 2022. Global number of deaths caused by floods 1960-2020 <https://www.statista.com/statistics/1293207/global-number-of-deaths-due-to-flood/>
* “Arduino Development Boards”, [Online]. Available: <https://www.mouser.ph/applications/open-source-arduino/>
* “Arduino IDE”, [Online]. Available: <https://www.javatpoint.com/arduino-ide>
* Arduino – Temperature Sensor. Tutorials Point. Retrieved from [https://www.tutorialspoint.com/arduino/arduino\_temperature\_sensor.htm#](https://www.tutorialspoint.com/arduino/arduino_temperature_sensor.htm)
* EG Projects. (n.d.) LM35 Temperature Sensor Pin Out, Interfacing guide, Circuit Construction and working Principals. Retrieved from https://engineersgarage.com/lm35-description-and-working-principal/
* Golding, B.W., 2009. Long lead time flood warnings: reality or fantasy? Meteorol. Appl. 16 (1), 3–12.
* Hydrology 2021,8, 183. Flood Early Warning Systems Using Machine Learning Techniques: The Case of the Tomebamba Catchment at the Southern Andes of Ecuador https://doi.org/10.3390/hydrology8040183
* Kellens, W., Terpstra, T., De Maeyer, P., 2013. Perception and communication of flood risks: a systematic review of empirical research. Risk Anal. 33 (1), 24–49.
* Louis, 2018. WORKING PRINCIPLE OF ARDUINO AND USING ITAS A TOOL FOR STUDY AND RESEARCH <https://www.researchgate.net/publication/326316390_Working_Principle_of_Arduino_and_Using_it_as_a_Tool_for_Study_and_Research>
* Office for the Coordination of Humanitarian Affairs Services, 2020. <https://reliefweb.int/report/philippines/philippines-typhoon-vamco-ulysses-worst-flooding-decades-cagayan-valley-region-16>
* O’Sullivan, J.J., Bradford, R.A., Bonaiuto, M., De Dominicis, S., Rotko, P., Aaltonen, J., et al., 2012. Enhancing flood resilience through improved risk communications. Nat. Hazards Earth Syst. Sci. 12 (7), 2271–2282. [https://doi.org/10.5194/nhess-12- 2271-2012](https://doi.org/10.5194/nhess-12-%202271-2012).
* Project Hub. <https://create.arduino.cc/projecthub/vaibhavshanbhag467/iot-based-flood-monitoring-and-alerting-system-e03cff>
* ŠKULÉTY, E., PIVARČIOVÁ, E., KARRACH, L., (2018). The Comparing of the Selected Temperature Sensors Compatible with The Arduino Platform. Management System in Production Engineering. 26(3), 168-171
* The World Bank Group, 2021**.** <https://climateknowledgeportal.worldbank.org/country/philippines/vulnerability>
* UNISDR, 2018. UNISDR Annual Report 2017 United Nations Office for Disaster Risk Reduction. Geneva, Switzerland. Retrieved from <https://www.undrr.org/publication/unisdr-annual-report-2017>.
* Wan Haszerila Wan Hassan, Aiman Zakwan Jidin, Siti Asma Che Aziz, Norain Rahim Centre for Telecommunication Research and Innovation, Fakulti Teknologi Kejuruteraan Elektrik Elektronik (FTKEE), Universiti Teknikal Malaysia Melaka (UTeM), Malaysia
* WMO, 2013. Flood forecasting and early warning. World Meteorological Organization. Geneva, Switzerland. Retrieved from <https://library.wmo.int/doc\_num.php?explnum\_id=4269>.
* Ye Zheng, Yazhou Xie, Xuejiao Long, A comprehensive review of Bayesian statistics in natural hazards engineering, Natural Hazards, 10.1007/s11069-021-04729-2, 108, 1, (63-91), (2021).

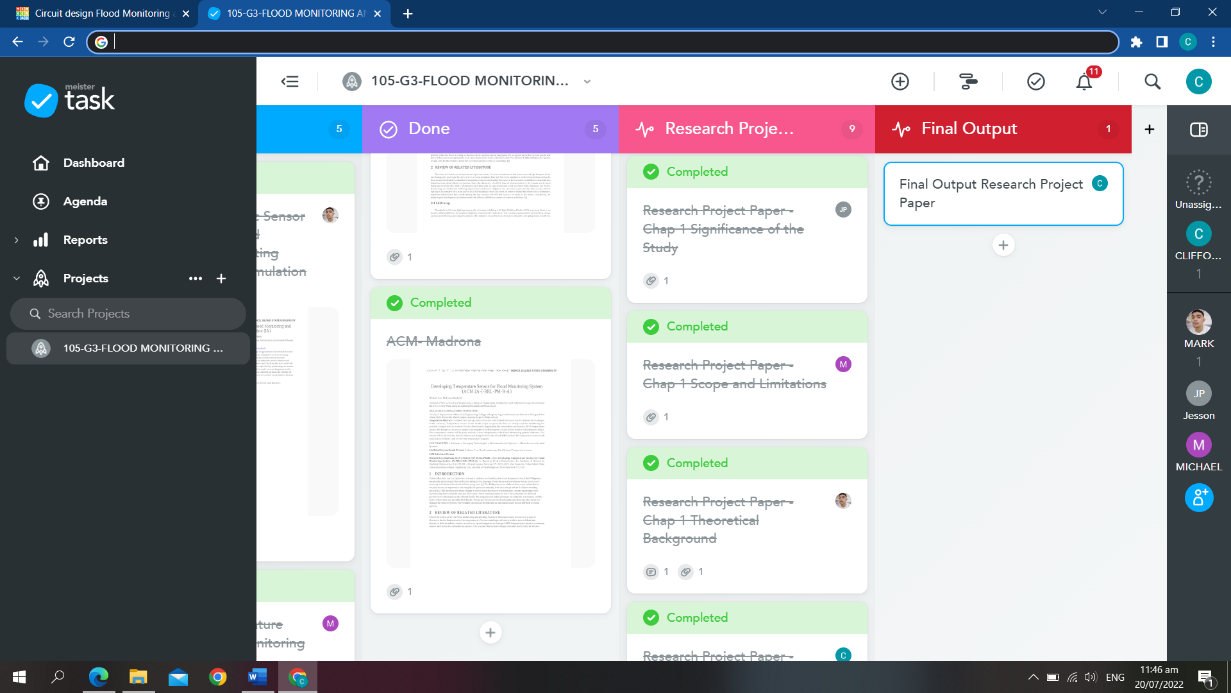
**APPENDICES**

**A.1 GROUP TINKERCAD SIMULATION**

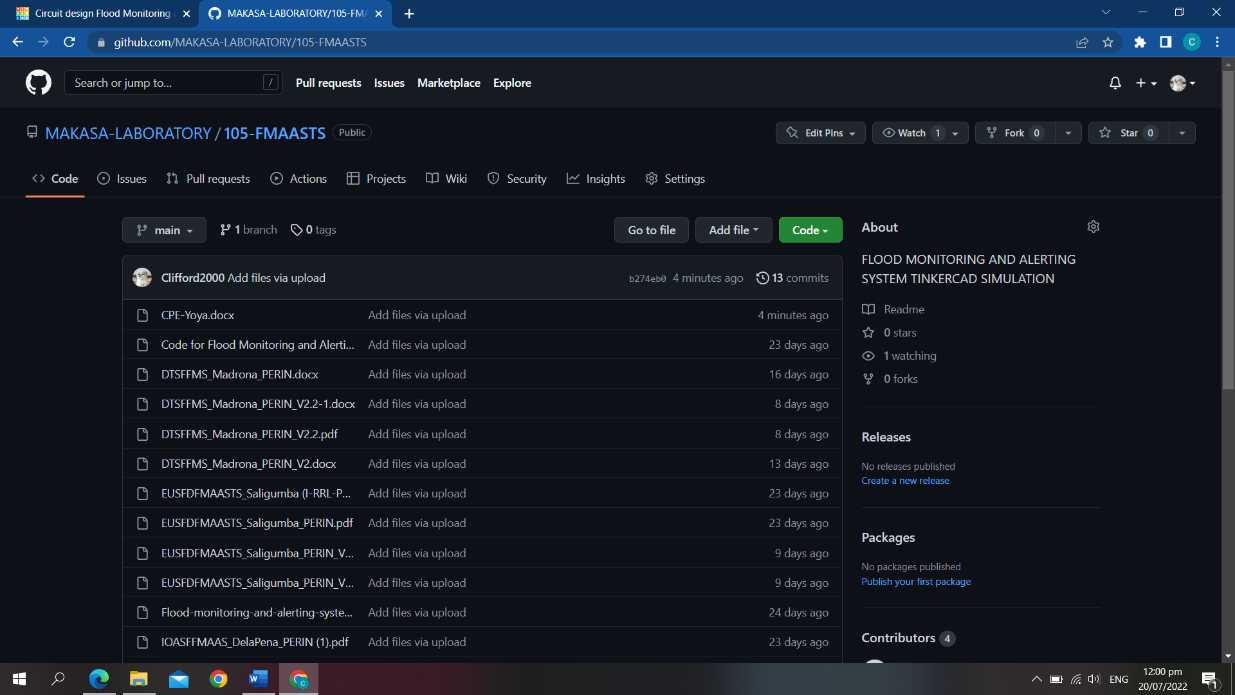
<https://www.tinkercad.com/things/1tZ8RsRHa2W-flood-monitoring-and-alerting-system/editel?sharecode=7SeK3XzfXnQtMMkclFVAJcOt5VIRolLs1hXndxljGp0>

**A.2 GROUP MEISTERTASK**

<https://www.meistertask.com/app/project/gM9jmKqQ/105-g3-flood-monitoring-and-alering-system-tinkercad-simulation>

****

**A.3 GITHUB GROUP CONTRIBUTIONS**

<https://github.com/MAKASA-LABORATORY/105-FMAASTS>

**A.4 SOURCE CODE FOR TINKERCAD SIMULATION**

Note that the researchers only focus on the principle on how the ultrasonic sensors will measure distance of the water because some electronic components are not available in the website.

#include <LiquidCrystal.h>

LiquidCrystal lcd = LiquidCrystal(10,9,8,7,6,5); // Create an LCD object. Parameters: (RS, E, D4, D5, D6, D7):

const int trigPin = 12;

const int echoPin = 11;

float time, distance;

void setup()

{

lcd.begin(16, 2); // Specify the LCD's number of columns and rows. Change to (20, 4) for a 20x4 LCD

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

Serial.begin(9600);

}

void loop()

{

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

time = pulseIn(echoPin, HIGH);

distance = (time\*.0343)/2;

// For Serial Monitor

Serial.print("Distance:CM ");

Serial.println(distance);

// For LCD Display

lcd.setCursor(0,0);

lcd.print("Distance in CM");

lcd.setCursor(0,1);

lcd.print(distance);

}

**A.4.2 SOURCE CODE FOR ACTUAL SYSTEM**

**A.4.2.1 Code for Arduino**

//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);

}

**A.4.2.2 Code for conf.py**

#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'

**A.4.2.3 Code for main.py**

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:

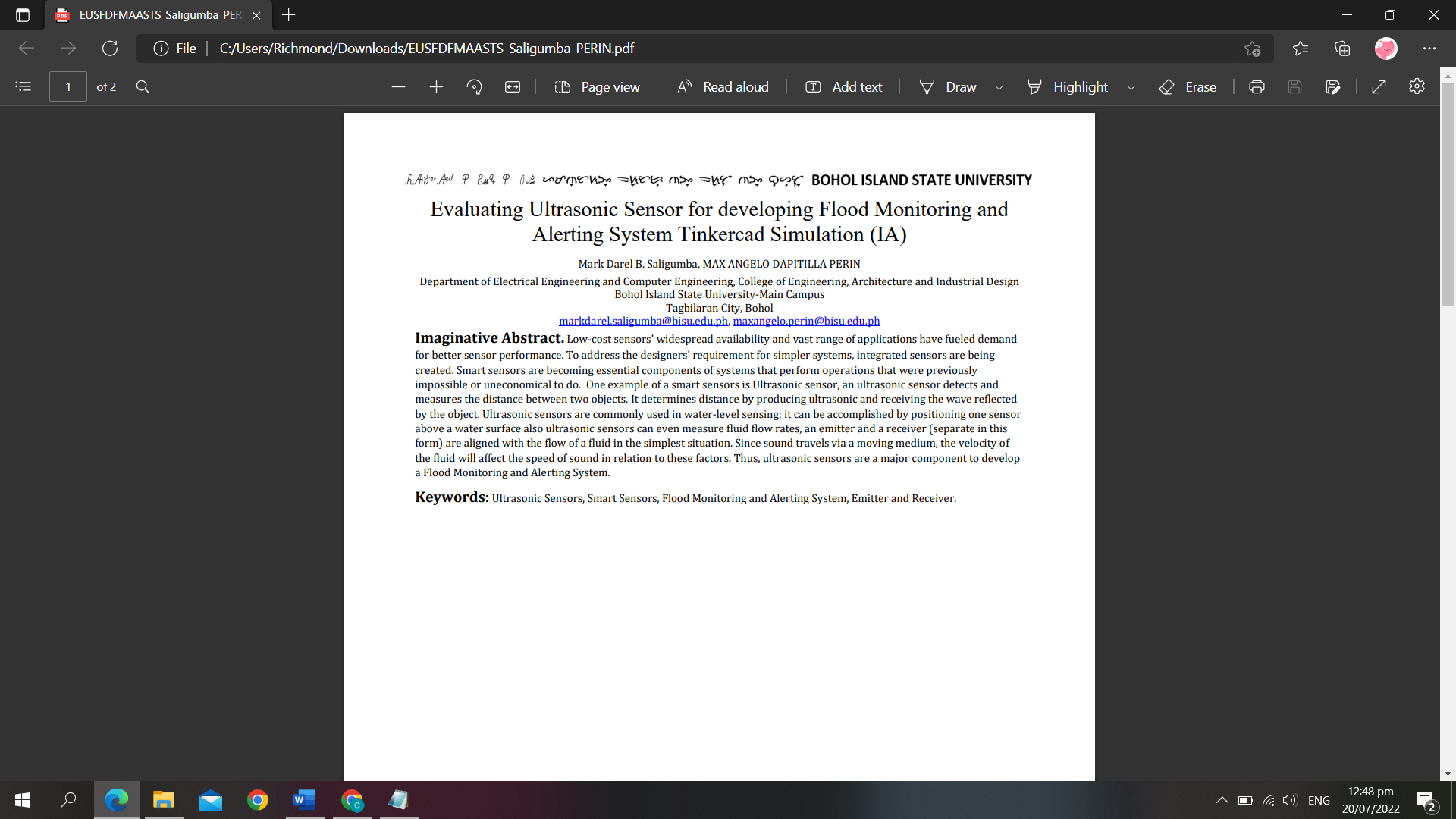
print ("Error occured: Below are the details")

print (e)

time.sleep(15)

**A.5 Authors**

**Author 1 (Evaluating Ultrasonic Sensor for developing Flood Monitoring and**

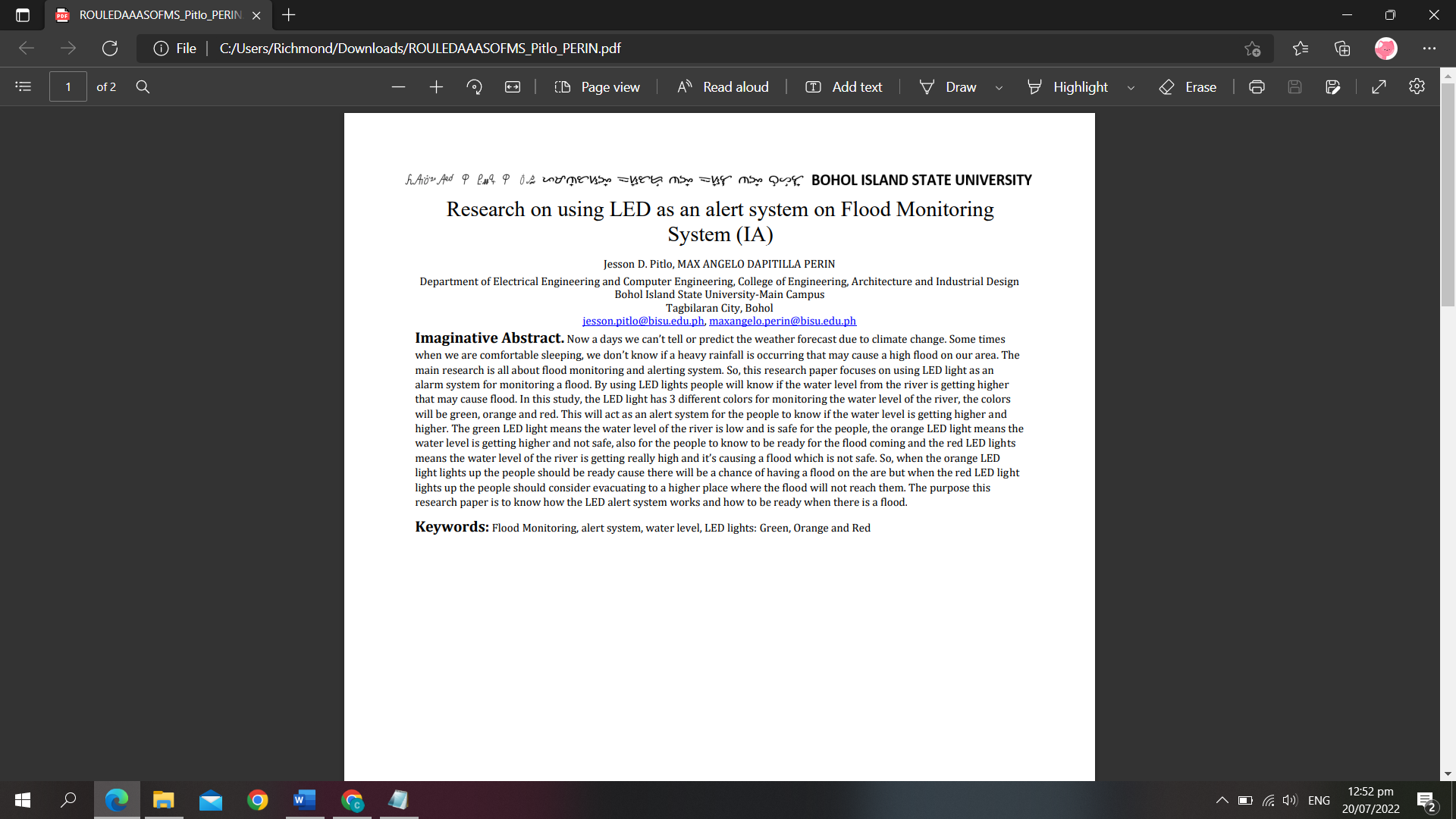
**Alerting System Tinker CAD Simulation)**

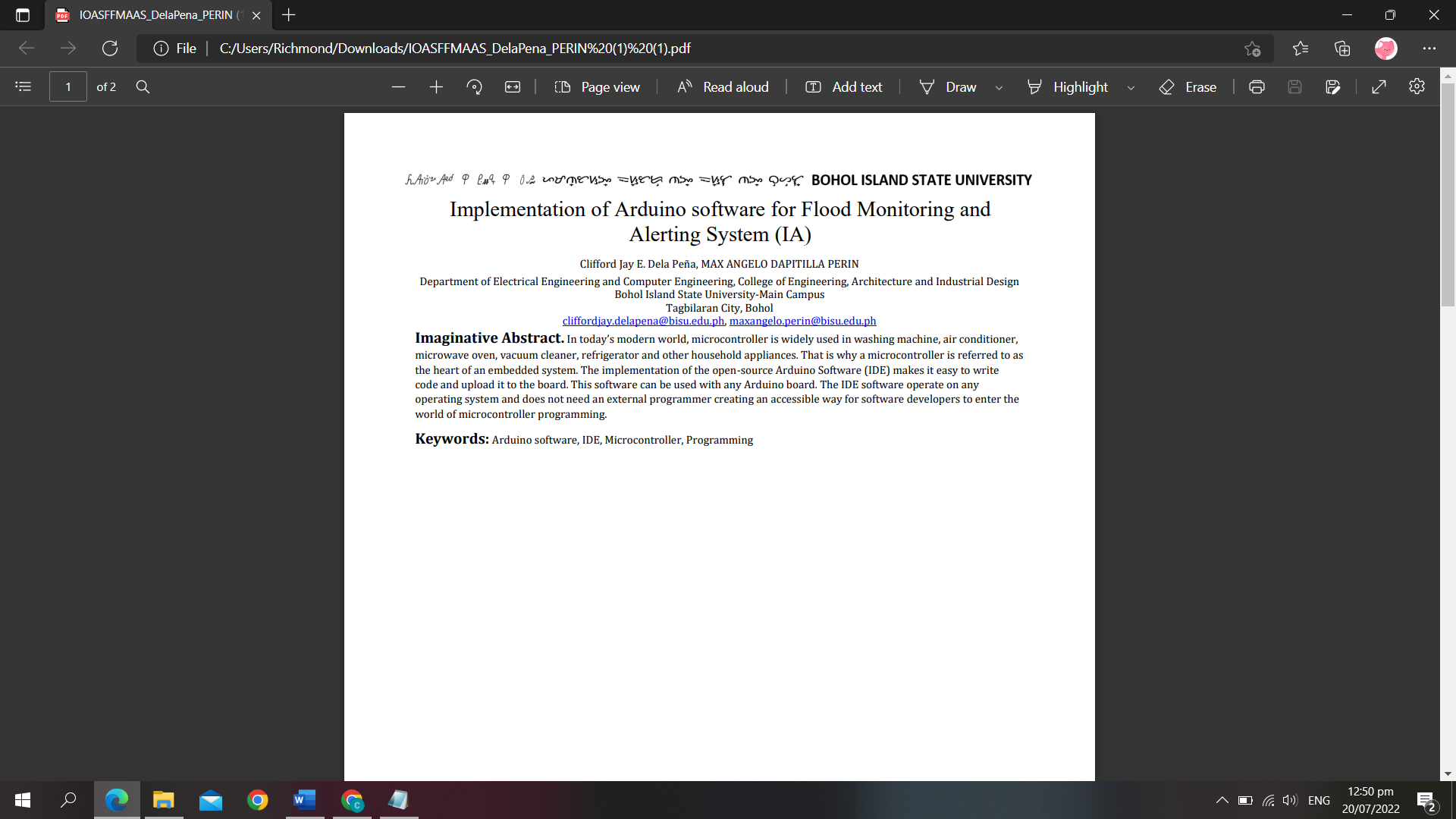
**Author 2 (Implementation of Arduino software for Flood Monitoring and**

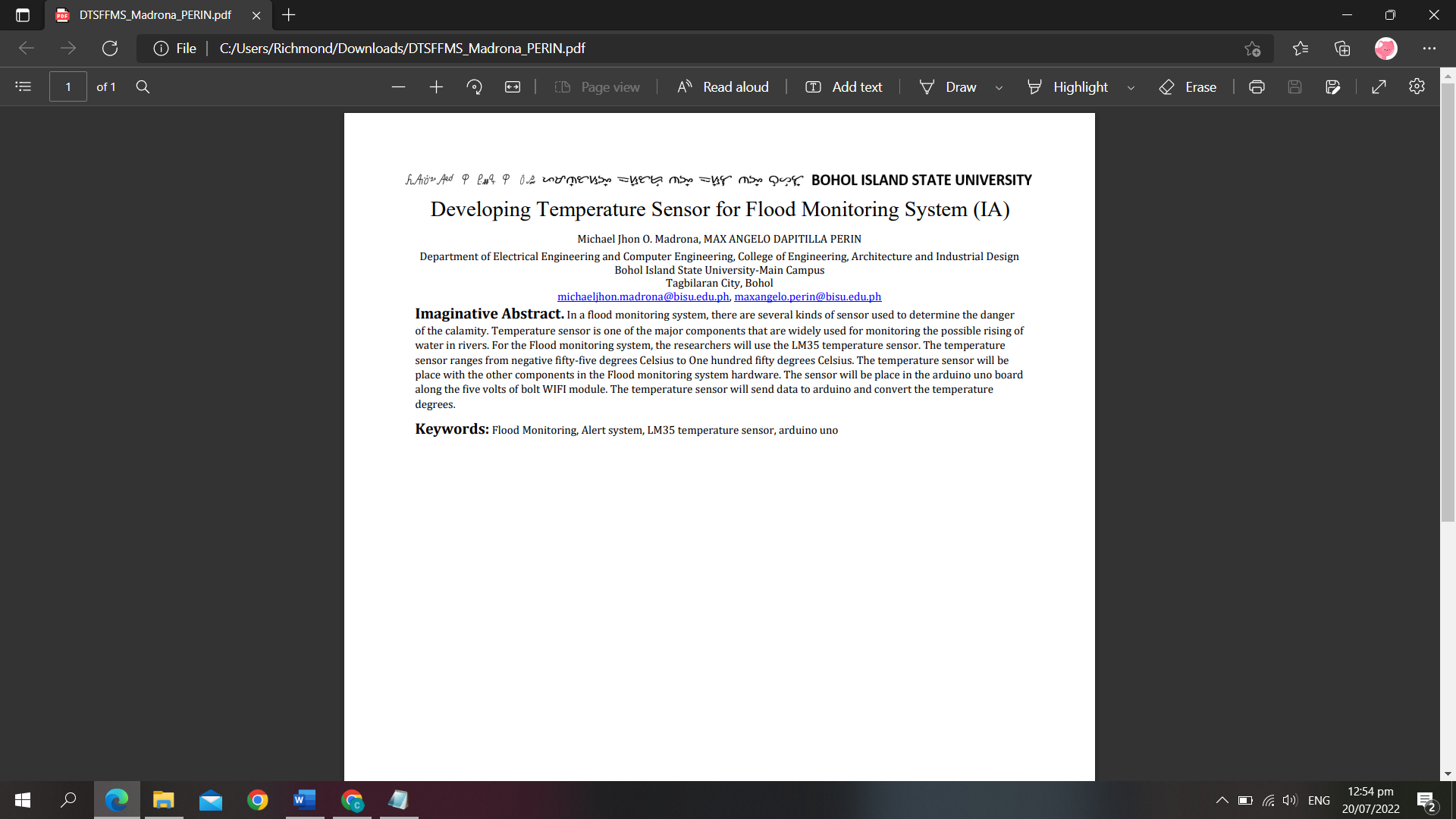
**Alerting System)**

**Author 3 (Research on using LED as an alert system on Flood Monitoring**

**System)**

****

****

**Author 4 (Developing Temperature Sensor for Flood Monitoring System)**