LIGTAS: LOCALIZED WEATHER MONITORING MANAGEMENT AND EARLY WARNING SYSTEM FOR LAS PIÑAS CITY VIA MACHINE LEARNING

A Project Study Presented to the Faculty of

Electronics Engineering Department

College of Engineering

Technological University of the Philippines

In Partial Fulfillment of the Requirements for the Degree of

Bachelor of Science in Electronics Engineering

Submitted by:

Opeña, Carlyle Erika E.

Pangonotan, Faizah B.

Remigio, Renj Asher Angelo H.

Reyes, Dave C.

Tuvieron, Kerwin G.

Adviser:

Engr. Edmund G. Monilar

August 2023

APPROVAL SHEET

This project study entitled "**LIGTAS: LOCALIZED WEATHER MONITORING MANAGEMENT AND EARLY WARNING SYSTEM FOR LAS PIÑAS CITY VIA MACHINE LEARNING**", has been prepared and submitted by the following proponents:

Opeña, Carlyle Erika E.
Pangonotan, Faizah B.
Remigio, Renj Asher Angelo H.
Reyes, Dave C.
Tuvieron, Kerwin G

In partial fulfillment of the requirements for the degree of **Bachelor of Science in Electronics Engineering is hereby recommended for approval.**

ENGR. EDMUND G. MONILAR Project Adviser	**ENGR. JAY FEL C. QUIJANO** Panel Chair
ENGR. TIMOTHY M. AMADO Panel Member	**ENGR. VILLAMOR M. AMON** Panel Member
ENGR. NILO M. ARAGO Panel Member	**ENGR. EDMON O. FERNANDEZ** Panel Member

Accepted and approved in partial fulfillment of the requirements for the degree of **Bachelor of Science in Electronics Engineering.**

ENGR. TIMOTHY M. AMADO Head, ECE Department	**ENGR. NILO M. ARAGO** Dean, College of Engineering

ABSTRACT

Weather monitoring and forecasting are of the utmost significance in human life, influencing daily activities affected by climate change. With this, the collection of information about weather patterns is vital for making informed decisions in responding to natural phenomena. The LIGTAS weather application system is a weather monitoring and early warning system, focusing on the localized forecasting within Las Piñas City, a recognized flood-prone area in Metro Manila. Long Short-Term Memory (LSTM) Neural Networks were employed as a predictive model, utilizing data collected from sensors that measure parameters such as temperature, relative humidity, wind speed, wind direction, atmospheric pressure, rainfall intensity, and water levels to forecast weather and flood conditions.

Input data were integrated through Arduino Uno, PacketDUINO, and PacketPro. The transmission of these data was facilitated by Long Range Wide Area Network (LoRaWAN) technology, which wirelessly connects sensors to end devices, enabling real-time data collection. The regression metrics results from the trained model were substantial. The model achieved a Mean Squared Error (MSE) of 0.0208 during training and an MSE of 0.4558 in field testing, a Root Mean Squared Error (RMSE) of 0.1 during training and an RMSE of 0.6 in field testing, a Mean absolute error (MAE) of 0.0335 during training and an MAE of 0.4170 in field testing, with an accuracy of 91.1% during training and accuracy of 92.53% in field testing. Therefore, with the results stated above, the LIGTAS application is recommended for use as a localized forecasting tool by Local Government Units (LGUs). LIGTAS will contribute to the mitigation of natural disasters and challenges faced by Local Disaster Risk Reduction and Management Offices (Local DRRMOs).

ACKNOWLEDGEMENT

First and foremost, we would like to express our deepest gratitude to Jesus Christ for His wisdom, strength, support, and guidance throughout this study, enabling us to overcome challenges and pursue our thesis. This research paper would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

To Engr. Edmund G. Monilar, our research adviser, for his exemplary guidance and support, valuable feedback and constant encouragement throughout the duration of this thesis paper. His expertise and insightful feedback were of immense help throughout and have been instrumental in shaping the direction and quality of this study.

To Engr. Nilo M. Arago, Engr. Timothy M. Amado, and ECE faculty, for their exceptional leadership and commitment to strive academic excellence in every thesis paper conducted by the researchers. We are utterly grateful for the shared propitious insights and continuous motivation, which have continuously propelled us forward in our scholarly pursuits.

To the Engineering Department of the Local Government Unit (LGU) of Las Piñas has provided. Their continuous support in providing resources, access to facilities, and technical resources has been indispensable. We are deeply appreciative for their willingness to answer our queries, provided technical advice and means, and offered practical solutions to the challenges encountered during the thesis process have greatly contributed to the success of this study.

To Engr. Romnick U. Cartusiano, the City Engineer of Las Piñas, for his exceptional assistance and support in guiding our team throughout our expedition and lending us valuable help from the engineering department for technical assistance.

To the Packetworx team, especially to these persons, Mr. Rommel Gadil, Mr. lan Gabrielle M. Jimenez, Mr. Lance Kevin Estrella, Mr. Florian S. Agbuya, Ms. Irish Joy Maguddatu, and Ms. Crisanta Paola Catap, we are truly grateful for the outstanding support and assistance provided by Packetworx. We would be delighted to serve as a reference for your services and highly recommend your expertise to others seeking innovative and reliable solutions.

To our dearest family, thank you for encouraging us in all our pursuits and inspiring advice to follow our dreams. We are utterly grateful to our parents for supporting us throughout this study, emotionally and financially, and for their wise counsel and sympathetic ear.

Finally, to our friends, we were not only able to support each other by deliberating over our problems and findings, but also happily by talking about things other than just our papers.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

THE PROBLEM AND ITS SETTING

1.1 Introduction

The Philippines is considered a climates' lodestone and is generally known as one of the world's most disaster-prone countries, highlighting both its tremendous incidence of natural hazard events and significant vulnerability. Natural Hazards such as typhoons, earthquakes, floods, volcanic eruptions, landslides, and fires all have an impact on the country. According to an international report, the Philippines was ranked third out of 173 countries prone to disaster risks [1]. Due to its geography and geology, the country is in the path of tropical cyclones that can be categorized as tropical depressions, tropical storms, severe tropical storms, typhoons, and super typhoons. Along with its geographic location, the country is extremely susceptible to flooding, storm surges, and severe winds [2]. Conversely, the Philippines continues to experience disastrous fires, which are frequently among the worst in history. According to the World Health Organization report, there are an estimated 180,000 global burn deaths every year, which usually occur in low- and middle-income countries [3]. In the Philippines, 96,447 fire incidents have been reported nationwide from 2011 to 2017, causing 1,924 deaths, 5,750 injuries, and ₱31.06 billion in property loss [4].

Natural disasters are inevitable, but applying precautionary measures, and timely, and accurate prediction of both fire and climate extremes aids societies in preparing for and mitigating disasters, as well as reducing losses in infrastructure and productive activities. Early warning systems and forecasts provide foreknowledge, which, when combined with public awareness, education, and preparedness, allows people to act quickly

in response to hazard information, strengthening human safety and reducing human and economic losses from natural disasters. Early warning systems have benefited some countries in reducing the adverse effect of natural disasters, but such systems can be limited in scope. Whilst PAG-ASA's weather forecasting in our country is extensive, restricting it to a wide area or region, such as Metro Manila, Davao Region, MIMAROPA Region, and so on. Extreme local weather events, particularly heavy rainfall limited to specific regions, are becoming more frequent and intense in some cases, but are frequently not recorded by national warning centers [5].

Risk maps provide comprehensive information about flood risks within a city, but their coverage is limited to the city level and lacks granularity at the barangay level. In the context of flood hazard studies conducted in Metro Manila, there is a noticeable absence of accessible flood maps that effectively identify high-risk areas and provide insights into potential flood depths corresponding to various magnitudes of flooding. The need for location data pertaining to flood-prone areas, which can be utilized by government agencies in future flood events, was emphasized by the author. Recognizing the significance of barangays in city activities and their crucial role as frontline defenders, it becomes an utmost need to involve them in government disaster management plans and policies. This inclusion ensures comprehensive and effective strategies for addressing and mitigating flood-related risks. Barangays serve as vital resources for cities in which they contribute to enhancing Philippine government's disaster reduction and mitigation programs. By actively involving barangays, significant improvements can be achieved in these programs, leading to more effective strategies in addressing and mitigating various disasters. Correspondingly, there is inadequate innovation in forecasting in localized areas.

Localized forecasting, specifically for each barangay through a local early warning system that can be operated by local government units to enable communities to respond to warning messages appropriately and in a timely manner, is extremely crucial to circumvent additional damage and losses associated with flooding risks and fire damages. Thus, evacuations can be initiated, and material goods can be protected. In this way, a comprehensive and sustainable contribution to disaster risk management is of utmost need at the local level.

In relation to that, Las Piñas City counts as one of Metro Manila's flood-prone areas. Flood occurrences in Las Piñas are caused by heavy rains, high tides, and waters that come down from Cavite**.** The occurrence of floods within this city is attributed to the tidal movements originating from Manila Bay, leading to the inundation of lands situated up to 0.3 meters above the average sea level. Additionally, measurements have indicated land subsidence of up to 0.33 meters in central Manila [6]. According to Alejandro S. Dela Merced, Officer - in - Charge (OIC) of LPDRRMC or Las Piñas Disaster Risk Reduction Management Council, the city has its own early warning system for weather forecasting that disseminates timely and meaningful warning information of a potential occurrence of risk that enables at-risk individuals, specifically the citizen of the said city to be well-prepared, respond appropriately, and have ample time to minimize potential harm or losses. However, this existing system in the city of Las Piñas is faced with challenges and demands of providing more accurate, timely, and useful forecasts and information, because it is all done through the manual processing by the officers in the LPDRRMC.

In addition, Mr. Dela Merced also mentioned that there are numerous informal settlers residing in the city of Las Piñas. Their houses are mostly built-up with light

materials which are prone to fire. On the local scene, records from the Las Piñas Disaster Risk Reduction Management Council (LPDRRMC) shows that in 2021, there were twenty (20) fire incidents in the city and thirty-one (31) families were affected by this, causing enormous damage to several houses. With that being said, the LPDRRMC is concerned about how they will provide early advisories to alert the residents when there is an occurring fire at a certain barangay and when there is an increase in temperature that is likely to cause fires. Having said that, this project aims to innovate and develop weather forecast information services in localized areas in Las Piñas that will promptly synchronize early warning devices from different barangays for the city to have sufficient time to prepare the citizen and devise a disaster risk mitigation plan, preparedness, and response.

1.2 Background of the Study

Due to the unpredictable variations in climatic and atmospheric conditions, weather forecasting is gaining traction as the most essential field of research. Over the last few decades, researchers have been developing new techniques for training models to achieve accuracy over nonlinear statistical datasets to prevent future disasters and global issues.

Based on the study of Wanjari et al [7], the researchers proposed a system that overcomes the numerous limitations encountered by the Numerical Weather Prediction Models which is used at different weather forecasting agencies. They utilized the concept of recurrent neural network in addition to the Long Short-Term Memory Units to improvise the learning model that helps in minimizing the errors in each epoch and improving the precision of the weather elements being predicted. The use of a recurrent neural network in their study learns from the past historic data in addition to the Long Short-Term Memory Unit (LSTM) algorithm resulted in the design of a new significant proposed system that

produces accurate results with high precision and stores the computed results for a long period of time. Further, they developed a responsive web application that contains the following predicted weather elements such as temperature, pressure, humidity, and cloud cover. The proposed system produced 86.9% accuracy with the prediction of a fortnight on the input of the previous fortnight.

The study [8] developed a web- based application to visualize real - time flood inundations and employ flood early warning systems. An Intelligent Hydroinformatics Information Platform (IHIP) was deployed in the web application to forecast regional floods which is driven by a database. The architecture of the IHIP is composed of five layers which are the data access, data integration, service, functional subsystem, and end - user application layer. To interconnect the layers, modules are used to provide interactions between the layers. The six modules that are used are the Data Connection, Data Query, Data processing, Flood Forecast, Flood Map Display, and Web Server. The Flood Forecast Module utilizes a hybrid model, merging the recurrent nonlinear autoregressive with exogenous inputs (RNARX) and the self-organizing map (SOM), to establish a flood forecasting model. For the Flood Map Display, the IHIP website incorporates the Google Maps API from the Google Maps Server Module, enabling the seamless integration of Google Maps within the platform. The key here in this study is the database and data acquisition in the data access layer. The database is very important in the organization of data and information in building the flood forecasting model. For the acquisition of data, the Quantitative Precipitation Estimation and Segregation Using Multiple Sensors (QPESUMS) system was employed in Taiwan. This system integrates various data sources such as Doppler radar, satellite infrared, and rain gauge, enabling accurate Quantitative

Precipitation Estimations (QPE) for severe weather systems. In addition, the construction of the database involved the utilization of Microsoft SQL Server 2016. Visual C# is employed to extract real-time hydrological data, manage relevant data for the models in real-time, store forecasted data, and facilitate the provision of information for the visual map-based display. Moreover, the online web page display integrates Google Maps and provides KMZ (zipped Keyhole Markup Language) files for download purposes. The ASP.NET Web application was created using C# and data can be accessed from SQL server using ADO.NET. Responsive Web Design (RWD) was also implemented to make the IHIP look great in any device. Lastly, the study was implemented in a regional level, specifically, in Tainan City of southern Taiwan. The stakeholders are the public and authoritative officers.

A weather monitoring and forecasting system is responsible for monitoring and observing fluctuations in temperature, humidity, wind speed, moisture, light intensity, UV radiation, and air quality indicators such as carbon monoxide levels [9]. This advanced system utilizes the Internet of Things (IoT) technology to ensure real-time accessibility and availability of its data across a variety of users. The system uses multiple of sensors such as DHT11 sensor for temperature and humidity, anemometer for wind speed, light dependent resistor for light intensity, ML8511 solar sensor for UV radiation, MQ7 for Carbon monoxide levels in the air, hygrometer for soil moisture, ultrasonic sensor for rainwater level, and raindrop sensor for detecting rainfall or snowfall. The study includes an android application in providing notifications to warn people about sudden and drastic weather changes. Using Node MCU and Ubi dots, the processed data is uploaded and

stored on a web page to serve as a database called Dark Sky and the sensor data is plotted as graphical statistics.

Sankaranarayanan et al. [10] presented an early flood warning system using a Deep Neural Network (DNN) based on temperature and rainfall intensity. The researchers compared DNN to the other machine learning algorithms such as SVM, KNN, and Naive Bayes to determine the best algorithm that provides better accuracy. In this study, determining a suitable dataset which is then divided into training and testing sets, was regarded as a significant step. Also, the corresponding steps must be repeated before deploying the model in real life until a lower bias is observed for a given model. The researchers provide an online flood early warning system portal for the citizens and administrators where the weather observation, weather forecasting, and updating the water level status are displayed to give them ample time to take some precautionary measures.

A research study by Samikwa E. [11] established a system that utilizes rainfall and water levels for its sensors, an ANN for its machine learning algorithm, and an IoT sensor to establish a telemetry system to synchronize the sensor's data directly to the system. It implements the ANN forecasting for flood prediction on a low-power edge device. This means that rather than sending raw sensor data over the Internet for real-time water level prediction, the forecast output is transmitted at appropriate intervals. Furthermore, the proposed system uses low-power devices for IoT sensing and edge computing and ultra-low-power Wireless Sensor Network (WSN) communication between sensing nodes and the computing edge platform, resulting in a low-power system. As a result, the system can be used as a battery powered IoT flood prediction system in which just the edge computing platform has Internet access rather than all sensor nodes. This study emphasizes the

convenient use of IoT in sending the data from the sensors wirelessly and the possibility of applying ANN for flood forecasting implemented on a low-power edge device.

1.3 Research Gap

There is a noticeable research gap regarding the involvement of citizens in citizen science for the purpose of supporting early warning systems. Limited attention has been given to exploring the intersection between citizen science and disaster prevention from a social science standpoint. Additionally, the technological focus has primarily been on developing systems, methodologies, and approaches, rather than adequately understanding citizens' needs, or implementing effective strategies for engaging them.

The existing studies did not provide a localized forecast-based early warning system in a Barangay-Level for the citizens, and authoritarian web-based applications, which citizens can use. This kind of system that integrates web-based applications will give way to involving the citizens in disaster prevention and may provide more accurate forecasting than regional-level forecasting. Also, this gives the local government unit and citizens enough time to take precautionary measures. Lastly, the synchronization of the hydrometeorological sensors is considered a challenge in weather forecasting. Sending and receiving real-time data from the sensors can lessen the tedious work of manually collecting data from these sensors, hence, improving the data gathering process for weather forecasting.

1.4 Research Objectives

1. To design, construct, and incorporate hydrometeorological sensors that will collect data such as temperature, humidity, wind pressure and speed, rainfall intensity, and water level status that will be utilized for analysis and interpretation.
2. To develop a web-based application that provides real-time weather monitoring and flood forecasting notifications.
3. To develop a weather and flood predictive model through Artificial Neural Networks.
4. To test and evaluate the recommended LIGTAS weather system after deployment through User Acceptance Test using Likert Scale.

1.5 Significance of the Study

This study is solely focused on the utilization and integration of the existing weather sensor technology in determining the best approach for predicting the occurrence of hazards in a localized area in the Philippines. The findings of the study would be beneficial to the following sectors. It will contribute to the residents of Brgy. Pulang Lupa Dos, Brgy. Talon Dos, and Brgy. Manuyo Dos, Las Piñas City to enable the community to respond to warning messages appropriately and in a timely manner using the developed web-based application to circumvent additional damage and losses associated with flooding risks and fire damages. Early warning systems is responsible in providing individuals and communities that faces potential hazards with the necessary tools and information to take appropriate and timely actions. Empowering individuals through these systems helps reduce the likelihood of personal injuries, loss of life, and damage to property and the environment.

Weather Monitoring and Early Warning System include detection, analysis, prediction, and warning dissemination to reduce the risk factors of climate-hazards. This study contributes to the Department of Science and Technology's Harmonized National Research and Development Agenda (HNRDA), under Section IV: Industry, Energy and Emerging Technology, Delivery of Social Services, Item One (1): "Environment and Pollution Control", whereas "Instrumentation for early warning, monitoring and rapid assessment - Forecast Based Financing and Weather Based Insurance Mechanism" is stated as one of the priorities for year 2018. Furthermore, this study will also contribute under Section V: Disaster Risk Reduction and Climate Change Adaptation (DRR & CA). The agenda in this section prioritizes research that addresses the four major action themes for Disaster Risk Reduction and Management which are Monitoring and Forecasting, Hazard and Risk Assessment, Warning, and Proper and Timely Response.

In addition, this study will contribute to the Sustainable Development Goals (SDGs). First, Goal Eleven (11): Make cities and human settlements inclusive, safe, resilient, and sustainable. One of the targets of the United Nations under this goal to significantly decrease the number of fatalities, minimize the number of affected individuals, and substantially reduce the direct economic losses in relation to the global gross domestic product, including water-related disasters, which this study also aims to achieve. Furthermore, Goal Thirteen (13): Take urgent action to combat climate change and its impacts. This study develops and integrates an Early Warning System that will monitor weather and detect natural hazards to strengthen resilience and to reduce the adverse effects of natural disasters.

1.6 Scope and Limitations

This research study will focus on developing and integrating an early warning system to forecast local weather and flood levels in a specific city in real-time through a web-based application. This project study will focus on utilizing a telemetry system using the Internet of Things (IoT) to collect data on each sensor and a machine learning algorithm, specifically, Artificial Neural Network, to predict the weather and flood levels in real-time. This research study will be implemented in Las Piñas City with support from the city's Local Disaster Risk Reduction Office (LDRRMO).

Innovated for this, a web-based application acts as the medium for real-time monitoring of data from the sensors and a medium for disseminating early warning messages by the system. The web-based application will be the destination of the forecast of weather and flood levels that will be used by the LPDRRMO, barangay personnel, and the citizens of the chosen barangay. Lastly, the end-user-level web-based application will be the primary tool for early warning messages of the system that the city's residents will use. This is also where the data such as temperature, humidity, weather forecast, and an early warning message about flood level can be viewed interactively.

The system implemented will be deployed in the Barangay Pulang Lupa Dos, Barangay Talon Dos, and Barangay Manuyo Dos of Las Piñas City, since it is one of the flood prone areas in Las Piñas because of its land topography considering its closeness to the river drive. This research study will focus on predicting weather and flood level; hence, other disasters such as earthquakes, hurricanes, tsunamis, and other hazards, are not included. Moreover, a native mobile application will not be developed for the display of real - time weather, predicted flood levels, and notifications for the users. Lastly, this

research study will only train the model for one month because of the limited time of the project.

1.7 Definition of Terms

Artificial Intelligence - Artificial intelligence is a technology that enables a machine to simulate human behavior. Machine learning is a subset of AI which allows a machine to automatically learn from past data without programming explicitly. The goal of AI is to make a smart computer system like humans to solve complex problems.

Artificial Neural Networks (ANNs) - An artificial neural network is a powerful data-driven, self-adaptive, flexible computational tool with the ability to effectively capture the complex, nonlinear underlying characteristics of any physical process.

Early Warning System (EWS) - An early warning system for floods, droughts, heat waves, or storms is a comprehensive system that alerts people when dangerous weather is approaching and provides information on what actions governments, communities, and individuals can take to lessen the effects.

Hydrometeorological Hazards - Process or phenomenon of atmospheric, hydrological or oceanographic nature that may cause loss of life, injury or other health impacts, property damage, loss of livelihoods and services, social and economic disruption, or environmental damage.

LoRaWAN - In short for Long Range Wide Area Network, is a low-power, wide-area network (LPWAN) protocol designed for long-range communication between battery-powered devices and internet-connected applications. It enables efficient, long-range

communication for Internet of Things (IoT) devices in various domains such as smart cities, agriculture, industrial monitoring, and asset tracking.

Predictive Model - Is a data model, based on inferential statistics, that is used to predict the response to a marketing promotion or a certain investment. The predictive model is commonly created by data scientists and uses statistics to predict outcomes.

Time Series Data - A sequence of measurable data taken from the sensors that in chronological order within a certain amount of time period.

Training Model - A dataset that is used to train an ML algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output. The training model is used to run the input data through the algorithm to correlate the processed output against the sample output.

Weather Forecasting - The prediction of the weather through application of the principles of physics, supplemented by a variety of statistical and empirical techniques.

Chapter 2

REVIEW OF RELATED LITERATURE

2.1 Weather Forecasting System

Weather forecasting is an application of using technology and scientific knowledge to predict the state of the atmosphere at a particular place and time. This includes temperature, humidity, cloudiness, rain, and wind speed. Weather changes from time to time. Since the weather cannot be controlled, the best meteorologists can do is to make use of the past and present data and patterns to predict the future weather. Natural disasters have a big impact on our society. Through accurate and timely weather forecasting, a good forecasting model can help in reducing losses and damage. Weather prediction has never been 100 percent accurate, but it has increased over time [12]. The process of predicting weather patterns is complex. It requires the need to analyze and decode large data sets gathered from sensors and weather satellites every day [13]. Weather forecasts still have limitations despite the use of modern technology and improved techniques for weather prediction.

2.1.1 Early Warning System

Early warning system is designed to notify individuals of impending hazardous weather conditions such as floods, droughts, heat waves, or storms. It not only alerts people about the approaching danger but also offers guidance on the appropriate actions that can be taken by governments, communities, and individuals to mitigate the adverse effects of such events. Numerous early warning systems have been created to enhance our ability to predict weather events. By leveraging

sensors, satellites, and computer models, these systems enable weather forecasting and predictions.

An effective real-time flood forecasting system relies on gathering real-time data concerning rainfall forecasts. This data is then fed into a flood forecasting model to accurately predict the severity, timing, extent, and magnitude of potential flooding. The effectiveness of flood forecasting and early warning systems depends on the precision of rainfall forecasts and the performance of predictive hydrological models. These models play a crucial role in flood forecasting by providing essential meteorological inputs. [14].

Table 1. Summary of Previous Studies of Weather Forecasting System

Author	Year	Title	Relevant Findings	Relationship to the Study
Alam, Fakhri Salam, Muhammad Khalil, Nasir Ahmad Khan, Owais Khan, Masaud	2021	Rainfall trend analysis and weather forecast accuracy in selected parts of Khyber Pakhtunkhwa, Pakistan	Prediction of monthly weather through historical data of Rainfall and Temperature	Both studies utilize the historical data of different parameters in predicting and analyzing weather forecast
Naresh, R. K. Narwal, Ekta Narwal, Ekta	2021	Weather Forecasting	A weather forecasting system that uses IoT sensors for weather monitoring and applying machine learning for weather forecasting.	Both studies employ different sensors for weather monitoring. In addition, this study uses a machine learning algorithm for analyzation and prediction based on the data.

2.2 Artificial Neural Network for Weather Prediction

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning algorithms are significant in predictive analytics just like in weather forecasting, which uses current and historical data sets to discover knowledge and predict future occurrences. A computerized system that can process nonlinear weather conditions within a defined domain and make predictions is usually developed using an artificial neural network (ANN). An artificial neural network is a powerful data-driven, self-adaptive, flexible computational tool with the ability to effectively capture the complex, nonlinear underlying characteristics of any physical process.

ANN is a system that processes data and gives output in relation to the input. Several highly non-linear neurons are interconnected to form a network in an artificial neural network. The neural network consists of three layers; these are input, hidden, and output layers. These neurons are connected by links that comprise weight; weights are the connection quality that exists between the neurons in the system [27]. There are other existing algorithms for weather prediction such as SVM, KNN, and Naïve Bayes. However, ANN model accuracy of prediction level is high as compared to other machine learning algorithms [10].

Table 2. The results of prediction accuracy for each algorithm for Weather Forecasting

	SVM	KNN	Naïve Bayes	ANN
Accuracy	85.57%	85.73%	87.01%	91.18%
Prediction	0.88	0.91	0.96	0.95
Sensitivity	0.94	0.90	0.87	0.93

Table 2 shows the results of the accuracy of the prediction. On comparing the accuracy obtained from four different algorithms, it can be clearly seen that the DNN outperforms the other algorithms with an accuracy of 91.18% Technically, an artificial neural network (ANN) that has a lot of layers is a Deep Neural Network (DNN).

2.2.1 Artificial Neural Network for Flood Monitoring

Flood is a natural phenomenon that is caused by heavy rainfall, rapid snowmelt or a storm surge from a tropical cyclone or tsunami in coastal areas. Flood occurs when the volume of water in canals, rivers, and streams overflow and submerges the land. Floods are one of the most commonly occurring disasters and have caused significant damage to life, including agriculture and economy.

For that, it was necessary to take proactive action by warning people even before a flood occurred to save their lives from devastation and to reduce the damage from flood occurrences, a flood monitoring and forecasting system is implemented. An effective real-time flood forecasting system that predicts flood severity, time of onset, extent, and magnitude of flooding is based on a collection

of real-time data related to rainfall forecast to flood forecasting model. ANN model is an efficient algorithm that is used in monitoring and forecasting flash floods [10,29]

Table 3. Summary of Previous Studies of Artificial Neural Network for Weather Prediction

Author	Year	Title	Relevant Findings	Relationship to the Study
Fente, Dires Negash Singh, Dheeraj Kumar	2018	Weather Forecasting Using Artificial Neural Network	From experimental result, it is observed that Long-Short Term Memory neural network gives substantial results with high accuracy among the other weather forecasting techniques	Both studies utilize the Artificial Neural Network algorithm for Weather Forecasting
Sahagun, Mary Anne M. Dela Cruz, Jennifer C. Dela Cruz, Jennifer C.	2017	Wireless Sensor Nodes for Flood Forecasting using Artificial Neural Network	Developed sensor stations to predict the flood water level at medium and high-risk areas. Used raspberry pi microcontroller to link received data and display this data in a website.	Both studies discuss the development of a system by integrating existing sensors. Further, both use a Raspberry Pi to send data to the cloud.
Sankaranarayanan, Suresh et. al,	2020	Flood prediction based on weather parameters using deep learning	DNN provides better accuracy when com-pared with other algorithms such as SVM, KNN and Naïve Bayes with a precision of 89.71%	Both studies utilize an algorithm such as Neural Network to provide better accuracy for prediction.

2.3 Recurrent Neural Network (RNN) for Weather Forecasting

The Recurrent Neural Network (RNN) comes under Artificial Neural Network which is used to perform predictive operation on sequential or time series data. To be prepared for future weather-related disasters, weather prediction models are essential as they provide information on potential upcoming weather conditions. One of the prediction models in use is the RNN. RNN has been used to forecast weather using a variety of atmospheric parameters, such as temperature, pressure, wind speed, and others. In study [15], it investigates deep learning models for weather forecasting using heuristically optimization methods wherein RNN can be applied for weather prediction with an adequate accuracy level. The authors in [16] conducted a comparison of ARIMA and RNN models for forecasting short-term wind speeds. The RNN model outperformed the ARIMA model for wind speed prediction, the MSE and RMSE are roughly 20 to 25% and 11 to 14% smaller in the RNN model compared to the ARIMA model for both datasets. Another studies, ARIMA compared to RNN [17] and SVM compared to RNN [18] as predictive models for predicting rainfall shows that the RNN model outperformed the ARIMA model and SVM since RNN model shows higher accuracy as the RMSE values of the RNN model is lower than the ARIMA model and SVM, and RNN model is more suitable for long term prediction.

Recurrent Neural Networks are considered as a powerful type of neural network due to the presence of their internal memory. Recurrent neural networks (RNN) have the ability to learn sequences with respect to time since they feature an additional recurrent connection to the hidden layers than feed-forward neural networks. In a feed-forward neural network, signals flow in only one direction, from input to output one layer at a time.

In a recurrent neural network, the output of a layer is added to the next input and fed back into the same layer, which is typically the only layer in the entire network. Therefore, unlike feed-forward neural networks, a recurrent neural network can receive a sequence of values as inputs, and it can also produce a sequence of values as output. Hence, RNN is being used as a forecasting prediction model in time series data.

In training an RNN, it uses backpropagation [15] through time and at each time step or loop operation gradient is being calculated. The gradient's purpose is to update the weight value of the neural network. If the weight is too low, the gradient will vanish and the hidden layer next to the input layer would stop learning. On the other hand, if the weight is too high, the gradient will explode [19]. Due to this problem, Long Short-Term Memory (LSTM) and Gate Recurrent Unit (GRU) were developed. To overcome the impact of short-term memory, LSTM and GRU use a gate structure. The information flow across the sequence chain can be regulated by the gate structure. They are widely utilized in natural language processing, speech synthesis, and speech recognition. Comparing the LSTM and GRU, the GRU has two gates (reset and update), whereas LSTM has three (input, forget, and output). This makes GRU save a lot of time without sacrificing performance since the matrix multiplication is reduced [25]. However, GRU does not have an internal memory which makes GRU only suitable for long text and small datasets.

2.3.1 Long Short-Term Memory (LSTM) for Weather Forecasting

The Long Short-Term Memory (LSTM) is a special type of RNN. LSTM was first introduced in 1997 by Sepp Hochreiter and Jurgen Schmidhuber. Despite the power of RNN to learn short-term dependencies, problems occur when capturing a long term set due to the vanishing gradient. To avoid the vanishing gradient

problem in RNNS, LSTM was proposed which is able to capture long term dependencies by utilizing memory cells and gate units [20].

Srivastava et. al [21], proposed the utilization of Long Short-Term Memory (LSTM) neural networks in forecasting the weather condition so that the model can better decide what to retain and what to forget. Based on the result of the experiments, the actual and predicted values are highly correlated, and the authors concluded LSTMs show the capability to be both efficient and accurate in making the predictions.

Salehin et.al [22], proposed a rainfall prediction model using artificial neural network and LSTM techniques. For accurate forecasting, 6 parameters were taken (temperature, dew point, humidity, wind pressure, wind speed, and wind direction). The authors arrived at a 76% accuracy in predicting rainfall using the LSTM and RNN model.

In the study [23], the authors aim to predict climate changes, since climate is a long-term pattern of weather. The LSTM was used in climate forecast, as it stores a large amount of data for a long time compared to Convolution Neural Network (CNN) and Recurrent Neural Network (RNN) techniques.

In a standard RNN cell, the input at the timestamp and the hidden state from the previous time step is passed through the activation layer to obtain a new state. In contrast, the LSTM process is a little more complicated because it requires input from three states, including the current input state, the short-term memory from the previous cell, and lastly the long-term memory. Before passing on the long-term and short-term information to the following cell, these cells use the gates to control the information that will be kept or deleted during loop operation. The gates are called (1) Input Gate,

(2) Forget Gate, and (3) Output Gate. These gates can be thought of as filters that eliminate unwanted, undesired, and irrelevant information.

2.3.1.1 Long Short-Term Memory (LSTM) Gates

The Input Gate consists of two parts, the first part is a sigmoid layer that decides on the information which would be updated, and the second part is a tanh layer that vectorizes the new values as z(t) and it is the input activation function. To create an update to the cell state, these are multiplied. After multiplication, it is added to the previous cell state with the forget gate output to create the next cell state C(t). The Forget Gate takes input from the previous cell state and some new information and outputs a number between 0 and 1. The 0 output will be discarded, whereas the 1 output will be kept. Lastly, The Output Gate is also composed of two parts, first a sigmoid layer which decides which values of the cell state will be outputted and the second is a tanh layer in which the cell state is input and acts as an output activation function. The two are multiplied to produce the final output of the cell at a time. [21]

Table 4. Summary of Previous Studies of Recurrent Neural Network (RNN) for Weather Forecasting

Author	Year	Title	Relevant Findings	Relationship to the Study
Srivastava, A. Anto, S.	2022	Weather Prediction Using LSTM Neural	The LSTM has been effective in correlating the actual and predicted values which concludes that LSTM has high accuracy in making weather predictions.	Both studies utilize the LSTM algorithm for predicting weather.

Salehin, I. Hasan, M. Talha, I. M. Dip, S. T.	2020	An Artificial Intelligence Based Rainfall Prediction Using LSTM and Neural Network	A study which uses LSTM and RNN to predict rainfall. It uses six parameters (temperature, dew point, humidity, wind pressure, wind speed, and wind direction). After analyzing all our data, it got 76% accuracy in our work.	Both studies utilize the use of the LSTM and RNN in time series data for predicting weather.

2.4 Hydro Meteorological Sensors

The basic means of hydrometeorological variables include pressure, wind speed and direction, temperature, and humidity, short and longwave radiation, and rainfall [28]. Weather instrument devices are generally essential specifically in the field of weather forecasting to manipulate the surrounding nature to our needs and benefits. Local and personal weather monitoring commences with the proper sensors, gauges, and equipment, in essence, the vast array of weather instruments. In the long days, people use weather instrument devices to improve a number of industries. As technologies advance and enhance throughout the years, so does the mechanism of weather instrument devices.

2.4.1 Air humidity and Temperature Sensors

A humidity sensor is a device that detects, measures, and reports the relative humidity (RH) of air or the amount of water vapor present in a gas mixture (air) or pure gas [29]. Air humidity is the next most frequently monitored physical element in meteorology. The two measuring methods of air humidity are generally used in digital weather stations today. These are the psychrometric method and the capacity method [30].

2.4.1.1 Psychrometric Method

In meteorology, the psychrometer method serves as the fundamental approach for measuring air humidity. A digital psychrometer, comprising temperature sensors such as PT100 and IST, is used for this purpose. By utilizing the psychrometric difference between the dry and wet-bulb thermometers, vapor pressure and relative humidity can be accurately calculated. [30]. A psychrometer is made up of two thermometers exposed side by side, one with a film of water on its sensing surface (the wet bulb) and the other dry (the dry bulb). Humidity is calculated using the cooling effect of evaporation from a wet surface. Although accurate, this method has several drawbacks for measuring fine-scale humidity, especially when used in remote areas [31].

2.4.1.2 Capacity Method

The capacity method relies on a change in permittivity caused by changes in air humidity. The humidity sensor works on the capacity principle, measuring frequency impulses from a designed oscillator [30]. Synoptic weather stations commonly employ hygrometers which depends on the principles of capacitance change. These hygrometers utilize small transducers that absorb and desorb water vapor, resulting in detectable changes in capacitance. [28]. The advantages of capacitive sensors include low power consumption, good linearity, and wide-range RH detection, but the main disadvantage is a complicated fabrication process [32].

In contrast, a temperature sensor is a type of electronic device that measures the temperature of its surroundings and converts the measured data into electronic data to record, monitor, or signal temperature changes [33].

2.4.2 Anemometer

Most weather stations have an anemometer, which detects patterns and changes in wind behavior. An anemometer measures the wind's pressure and force [34]. It is classified into three types: cup or propeller anemometers, mechanical anemometers, and handheld anemometers. It generally falls under three categories, cup or propeller anemometers, mechanical anemometers, and Handheld anemometers. Among the affordable and extensively used instruments in ecological studies are cup or propeller anemometers. These anemometers utilize rotating cups or a propeller driven by the wind. In the case of propeller anemometers, the device must either be held perpendicular to the wind direction or be mounted on a vane. In automated measurements, such devices often measure both wind direction and speed. Furthermore, mechanical anemometers are still used in meteorological stations, and ultrasonic anemometers are becoming more popular. Ecologists frequently use handheld anemometers for short-term measurements, but they are occasionally equipped with data loggers to measure microclimate [31].

2.4.3 Rain gauge sensor

From the first traditional rain gauge, which was said to be developing in 1418, up until the electronic rain gauge, technology in this field has improved significantly. Whilst discovered in 1418, the rain gauge system was not widely used

until 1441 in Korea, where farmers used it to measure rainfall and potential harvest [35].

Rain gauges measure the amount of precipitation that falls such as rain, snow, sleet, etc. Conversely, they come in a variety of styles. A simple funnel and collection jar will suffice for low-level applications where accuracy is not paramount. As a result, this type of rain gauge necessitates manual monitoring and recording. Tipping bucket rain gauges are ideal for meteorological applications where automated monitoring is preferential. [36].

2.4.4 Water Level Sensor

A device that converts energy into ultrasound, or sound waves above the normal range of human hearing, is known as an ultrasonic transducer. The ultrasonic sensor generates high-frequency sound waves and evaluates the echo that the sensor receives [37]. An ultrasonic sensor is a sensor used to detect the state of an object by using a distance count for the sensor and the object. Its functions are as a converter of sound magnitudes to electrical quantities [38]. The JSN-SR04T sensor, on the other hand, consists of two parts, namely the sensor and the control module. The sensor system has two main components, namely the ultrasonic transmitter and the ultrasonic receiver. The function of the ultrasonic transmitter is to emit ultrasonic waves with a frequency of 40 KHz then the ultrasonic receiver captures the results of the ultrasonic waves that hit the object [39].

2.4.5 Barometer

A barometer is a scientific device that measures atmospheric pressure, also known as barometric pressure. The atmosphere is made up of layers of air that wrap

around the Earth. As gravity pulls it to Earth, that air has weight and presses against everything it touches. This pressure is measured by barometers. Because the density of air is lower at higher altitudes and exerts less pressure, the number of atmospheres decreases. The density of air increases as altitude decreases, and so does the number of atmospheres. To obtain accurate atmospheric pressure readings, barometers must be adjusted for changes in altitude [40].

2.5 Flood prediction parameters

Floods are widespread natural disasters that result in extensive damage to individuals, agriculture, and the economy. A flood is a hydrometeorological event that occurs because the volume of river water or drainage channels exceed its drainage capacity. High-intensity rainfall exacerbates the issue, causing an increase in water volume along the river flow. However, monitoring river water levels can mitigate the impact of floods. One effective method is the installation of floodgates in the river flow. One approach to monitor river water levels involves the installation of floodgates within the river flow [39].

2.5.1 Rainfall Intensity

Rainfall intensity is a more dynamic factor than other factors (like watershed and drainage) as the main causes of flood. Rainfall intensity prediction can be valuable information for flood and drought risk management, as well as for planning outdoor activities. Rainfall exhibits a range of characteristics that have significant implications for planning purposes. Factors such as the quantity of rainfall, its seasonal distribution, and the sizes and intensities of individual storms play a crucial role in various activities. These include budgeting for city drainage design, forecasting seasonal runoff for hydro-electric power or irrigation, and other

related planning initiatives [41]. Rainfall is measured using a rain gauge, which can be either recording or non-recording. Non-recording rain gauges are cylindrical devices that require manual emptying at regular intervals to measure the collected rainwater. They provide measurements solely for the specific interval in which they are emptied, typically spanning 1, 6, or 24 hours. In contrast, recording gauges, also referred to as automatic or autographic gauges, not only measure the quantity of rainfall but also record its timing and duration. The instrument is connected to a recording device, which could be a mechanism that punctures holes in a paper tape or a pen that records the accumulated rainfall on a paper chart wrapped around a drum driven by a clock mechanism [42].

Recent research has been done that implements data mining to determine daily rain potency with naive bayes algorithm. The data consisted of 731 records with several attributes like MAX Temperature, MIN Temperature, Mean Temperature, Med Hr., Max Wind, Mean Wind, SLP Pressure, STN Pressure, Vis, and Prec. The research resulted in 82.51 % accuracy rate and helpful suggestions for future research which provide more dataset so it will produce more accurate results [43]. Another research of Muthmainnah et al. predicted rainfall intensity in Malang City using Naïve Bayes and Laplace Estimator methods. The result of this research proved that the Naïve Bayes and Laplace Estimator method could be used to predict rainfall time series for seven days with a percentage accuracy of 97.74%, 97.74% for precision, 100% for sensitivity and error rate of 2.26% [44].

Table 5. Summary of Previous Studies of Rainfall Intensity

Author	Year	Title	Relevant Findings	Relationship to the Study
Ali, Alwin Khairan, Amal Tempola, Firman Fuad, Achmad	2021	Application Of Naïve Bayes to Predict the Potential of Rain in Ternate City	Utilized Naive Bayes algorithm to determine daily rain potency. Numerous attributes and parameters are considered in the collection of data and yields an accuracy rate of 82.51 % using this algorithm.	Both studies discuss the usage of the algorithm for daily rain parameters and the said attributes will be taken into consideration.
Muthmainnah, Mahmuda Et al.	2018	Time Series Forecast for Rainfall Intensity in Malang City with Naive Bayes Methodology	Used Naïve Bayes and Laplace Estimator methods in predicting rainfall time series and obtained an accuracy of 97.74%, 97.74% for precision, 100% for sensitivity and error rate of 2.26%	Both discussed the utilization of an algorithm for the predictions of rainfall.

2.5.2 Water Level

According to the study of Dswilan S. et al, ultrasonic sensor JSN-SR04T can detect distances in the range of 20 cm - 600 cm. The results of characterization tests have been carried out in accordance with the datasheet. Output voltage values below 20 cm and above 600 cm experience significant changes because they are in

the saturation region [39]. Subsequent research from Andang [45] conducted a test to see the ability of the ultrasonic sensor JSN-SR04T to detect floods based on river water levels. The use of the ultrasonic sensor JSN-SR04T gives more accurate results because it can detect distances up to 6 meters. The JSN-SR04T can also be placed higher and safer in the event of rain because the sensor and control systems are in separate modules. Furthermore, another research conducted by Sulistyowati et al [46], where they utilized ultrasonic sensors and android Smartphone as flood early warning system was conducted in three study sites and showed good results. The level of accuracy of the equipment to detect the water levels was 97.28% for some categories of observation. The AFD application on Smartphone functions appropriately to provide information regarding the changes of water level of the river.

Table 6. Summary of Previous Studies of Water Level

Author	Year	Title	Relevant Findings	Relationship to the Study
Sulistyowati, Riny Sujono, Hari Agus Musthofa, Ahmad Khamdi	2017	Design and field test equipment of river water level detection based on ultrasonic sensor and SMS gateway as flood early warning	Ultrasonic sensors and android Smartphones as flood early warning systems were utilized and produced an accuracy of 97.28% for some categories of observation.	Both discussed the usage of ultrasonic sensors to detect the water level of the flood in a certain area. However, this study is solely focused on developing a web-based application for monitoring and alert systems.

Andang, A. Hiron, N. Chobir, A. Busaeri, N.	2019	Investigation of ultrasonic sensor type JSN-SRT04 performance as flood elevation detection	Ultrasonic sensor JSN-SR04T is used to detect floods based on river water levels. The use of the ultrasonic sensor JSN-SR04T gives more accurate results compared with manual measurement.	Ultrasonic sensor HC-SR04 will be implemented in this study to detect water level on the flood.

2.6 Weather forecasting parameters

Weather forecasting is the use of science and technology to predict the state of the atmosphere at a specific time and location in the future [47]. Weather forecasts have always played an important role for humankind in their everyday life activities. It is significantly imperative in today's smart technological world. A precise forecast model entails plentiful data to attain the most accurate predictions. In general, understanding weather conditions (in other words meteorology) of the atmosphere plays a critical role in our sustainable existence on the Earth surface, including real-time weather analysis for forecasting weather-induced calamities [48].

2.6.1 Temperature and Humidity

In meteorology, the rule in the air temperature is considered, so all other details refer to this ambient temperature, but are only briefly mentioned with temperature in the following. The weather of the day is frequently described using day or night temperatures. It should also be noted that temperature is highly dependent on altitude. This means that the air cools at about 1°C for every 100m of height gain.

As a result, it is critical to understand the measuring station's altitude and the height at which the temperature sensor is fixed to correctly consider the height and thus the temperature difference to other sensors or stations [49].

According to WMO (World Meteorological Organization) [50], weather stations for measuring air temperature are the most commonly used devices with electrical resistance. The temperature is measured in short intervals of a few seconds, and a 5-minute average is calculated. These 5-minute funds are then aggregated to an hourly average at the end of the hour. The measurement interval for non-automated - mostly private - stations can also be one hour or even one day.

2.6.2 Precipitation

Precipitation is moisture fluxes from the atmosphere to the earth’s surface, and it is a key hydrologic variable of the water cycle for meteorology, climatology, and hydrology. The atmosphere contains approximately three quarters of its heat energy from the release of latent heat by precipitation [51]. Precipitation is important in the climate system since it connects atmospheric and land surface processes. Even though precipitation takes many forms, it can be measured in a variety of ways. Precipitation is frequently measured by measuring the volume of water in a tipping bucket sensor. There are several methods for measuring different types of precipitation. This includes radar sensors and pressure sensors which are two of the most common methods. Precipitation is measured by pressure sensors by counting the number and impact of drops hitting a pressure plate. By measuring the duration of the impact, these sensors can usually tell the difference between rain and hail. Radar sensors also measure the amount, intensity, and speed of precipitation.

Correlating quantity and intensity yield volume, and speed is used to estimate precipitation type [52].

2.7 Web Application for Forecasting

The study of Ravaud M. et al is focused on the development of a flood forecasting system and a web-based platform for the city of Dijon. The operational tool is based on INFLUX which is a real-time management system for sanitation network, flood risk management, and preservation of the natural environments. It provides powerful tools to process and access data as well as to link external models. In addition, the INFLUX real-time platform works on top of the pre-existing Supervisory Control and Data Acquisition system (SCADA) that collects and stores measurements performed by the local operator. At each new run, the real-time system connects to the SCADA database to extract measurements of rainfall depth by local rain gauges along with measurements of water depth in Lake Kir and weirs positions [53].

Another study [54] developed a smart weather monitoring system along with the sensors that send the data to a web page and the sensor data is plotted as graphical statistics. The data uploaded to the web page can easily be accessible from anywhere in the world. This interface was created by the researchers consists of an app called MIT app inventor. This is responsible for providing notifications on weather updates and to act as a warning system in case of bad weather. The app will obtain the necessary information through the existing database.

Table 7. Summary of Previous Studies of Web Application for Forecasting

Author	Year	Title	Relevant Findings	Relationship to the Study
Mure-Ravaud, Mathieu Binet, Guillaume Bracq, Michael Perarnaud, Jean Jacques Fradin, Antonin Litrico, Xavier	2016	A web-based tool for operational real-time flood forecasting using data assimilation to update hydraulic states	Created a web-based platform wherein they used INFLUX platform which is a real-time management system for flood risk management. It also provides powerful tools to process and access data as well as to link external models.	Both discussed the execution of a web-based platform for a real-time flood management system and collection of necessary data.
Yashaswi Rahut, Rimsha Afreen, Divya Kamini	2018	Smart weather monitoring and real time alert system using IoT	Developed a web page application responsible for providing notifications on weather updates and to act as a warning system in case of bad weather.	Both discussed the implementation of a web page application that will send warning and early advisory notifications to the affected citizen in the event of adverse weather.

Chapter 3

METHODOLOGY

3.1 Research Design

INPUT	PROCESS	OUTPUT
Knowledge Requirements: • Weather management and disaster risk reduction • Hydrometeorological sensors • Internet of Things • Machine learning algorithms • Front-end and back-end development • HTML, CSS, JavaScript, Node JS, MySQL • MQTT Server **Hardware Requirements:** • Anemometer • Wind Vane • Barometric Pressure Sensor (BMP 180) • Rain Gauge • Temperature and Humidity Sensor • Water Level Sensor (Ultrasonic) • Float Switch • Arduino Uno • PacketDUINO • PacketPro • Solar Panel • Lithium-Ion Battery • Solar Charger Controller • LM2596 DC - DC Buck Converter **Software Requirements:** • Visual Studio Code • Arduino IDE • Node JS • TensorFlow • Python	1. Calibrate and integrate the hydrometeorological sensors using the Arduino Uno/PacketDUINO microcontroller. 2. Gather data from the hydrometeorological sensors and send that data through LoRaWAN and store it in a MySQL database server. 3. Develop and train a weather and flood predictive model using Artificial Neural Networks and deploy the predictive model using FASTAPI. 4. Develop a web application for the display of predicted data that will be used by the LPDRRMC, barangay officials, and citizens of Las Piñas City. This web application will also be used by the citizens of Las Piñas to get notifications about weather and flood forecasts. This web application will also allow the citizens to report disasters such as fire. 5. In the Node.js server application, make HTTP requests to the FAST API application's specific API endpoints or routes to retrieve predicted outputs. 6. Evaluate the functionality, reliability, and efficiency of the developed Localized Weather Monitoring Management and Early Warning System	LIGTAS: Localized Weather Monitoring Management and Early Warning System for Las Piñas City via Machine Learning

Figure 1. IPO (Input-Process-Output) Diagram

This study is designed as descriptive and developmental research. Descriptive in a way because this study will deploy a web application that needs to be evaluated using the User Acceptance Testing (UAT). Furthermore, this study is designed as developmental for it involves innovating weather and flood forecasting locally or in the barangay level for better accuracy. This is in accordance with Flack et al [55], they recommend for the further researchers to develop a method for effective flood warnings. Moreover, they recommend allowing an accessible forecasting system for the users and end-users to reduce the impacts of flooding from heavy rain that can be avoided by taking informed and timely action.

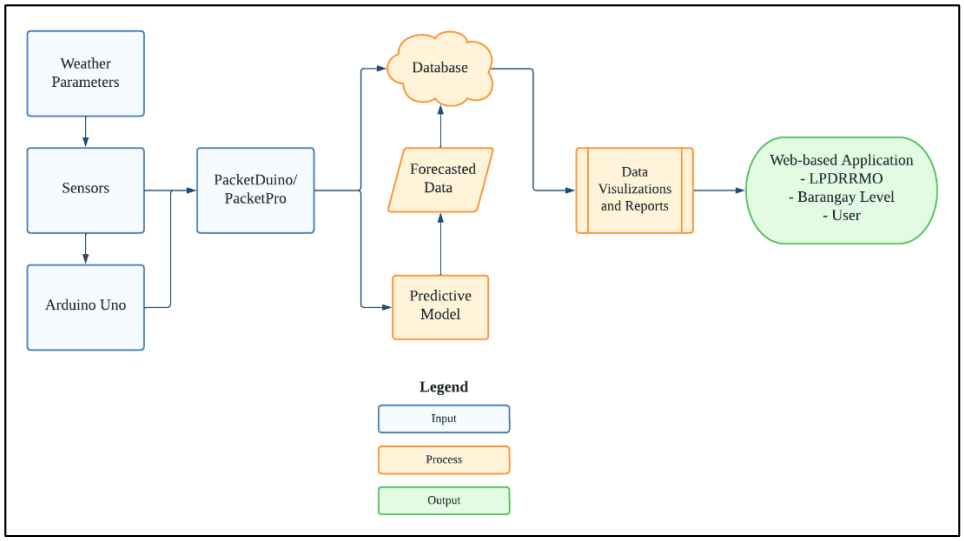


Figure 2. Block Diagram of the LIGTAS System

The proposed system starts with IoT sensing nodes acquiring real - time weather and flood level data that the Arduino Uno and/or PacketDUINO will read. This data will be transmitted by PacketDUINO and/or PacketPro through LoRaWAN technology, which will then be sent to the database. The real - time data will then be displayed to the Web - based Application. Also, the real - time data will be fed to the trained LSTM model that is used to predict weather and flood level. Moreover, this predicted data will be stored in a database and then displayed in the Web - based Application.

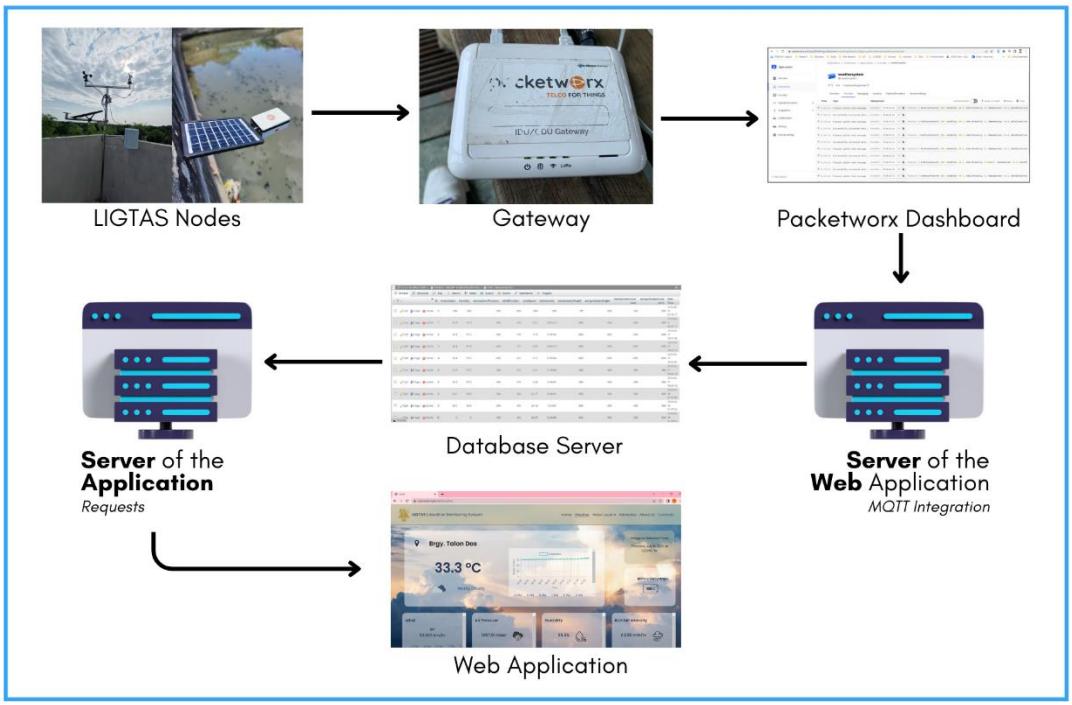


Figure 3. One-line diagram of the system network

The LIGTAS Weather and Flood Monitoring System functions as the publisher of the data, connected to a LoRaWAN gateway, which communicates with the LoRaWAN backend server. The server integrates MQTT for efficient data transfer and acts as the subscriber of the data, interacting with the Database Server for data storage and retrieval. The Application Server facilitates requests to the Database Server for specific data, while the Web Application serves as the user interface, displaying weather information, forecasts, and flood alerts.

3.2 Research Process Flow

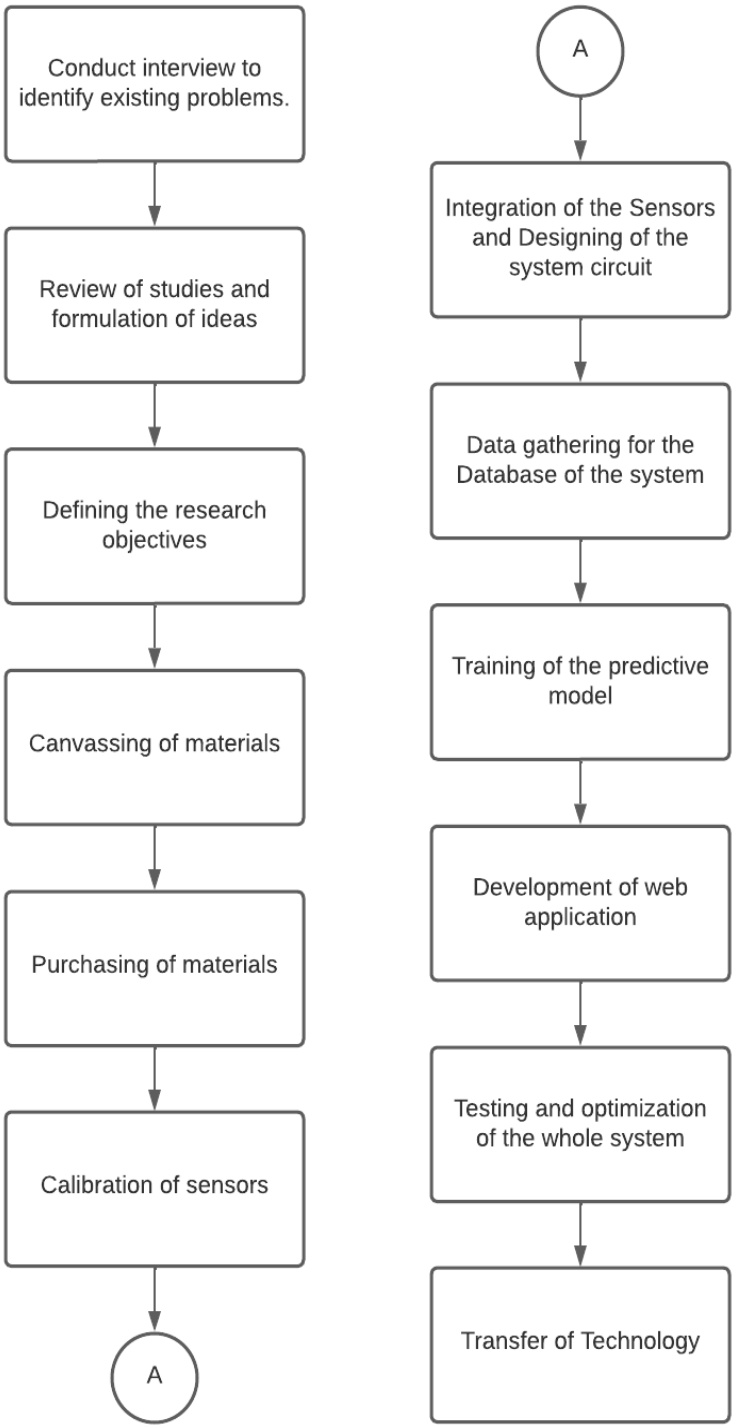


Figure 4. Research Process Flow

3.3 Hardware Development

3.3.1 Circuit Diagram

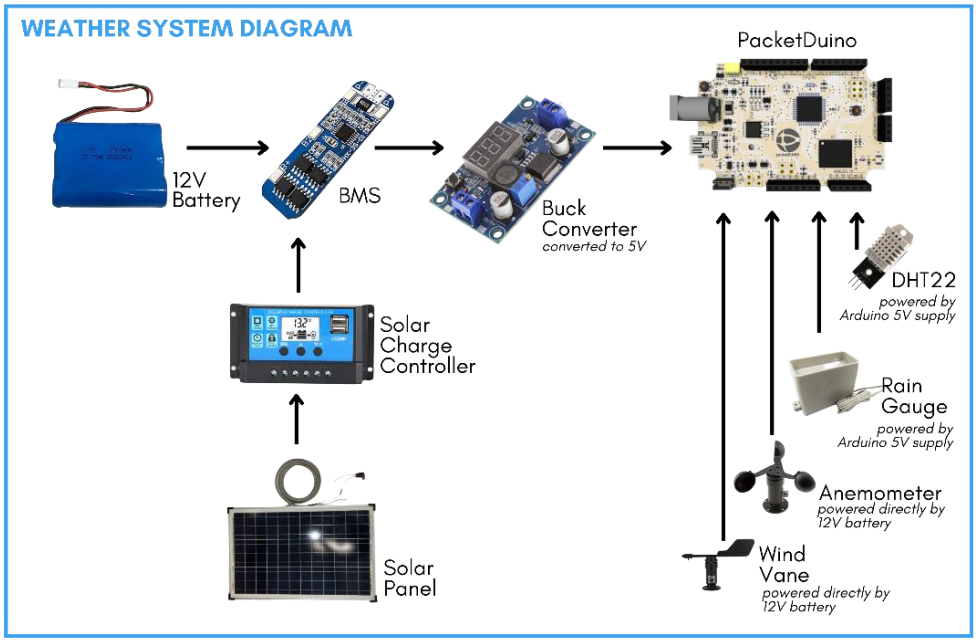


Figure 5. Weather System Diagram

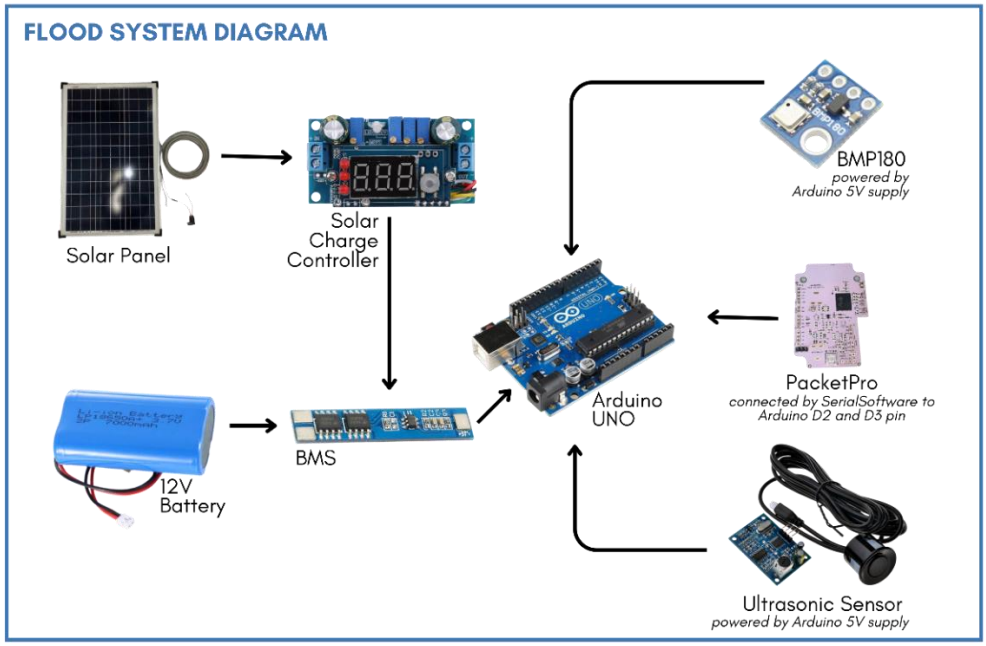


Figure 6. Flood Monitoring System Diagram

3.3.2 Materials and Equipment

The proponents will utilize various sensors in gathering the data that will be used as input variables in prediction. Two types of microcontrollers will be utilized for the Weather Monitoring System and Flood Monitoring System, the Packetduino and Arduino, respectively. These microcontrollers are responsible for reading data in each sensor. LORAWAN technology will be utilized by using Packetduino, an Arduino with built-in LORA technology, and a PacketPro that will be paired with Arduino for it to send the data to the database. As for the power supply of the system, 18650 lithium-ion batteries with BMS will be used. Solar Panels will also be added as a supplement source of electricity supply to save more cost.

3.3.2.1 Hydrometeorological Sensors

- **Anemometer**

  An Anemometer Sensor is a type of three-cup anemometer able to measure wind speed up to 30m/s or 67mph. It was designed to be placed outside and easily measure wind speed. It is made up of a shell, the wind cup, and the circuit module [56].

Figure 7. Anemometer Sensor

- **Barometric Pressure Sensor**

In this study, the BMP 180 is used to measure temperature from -40°C to 85°C and barometric pressure from 300 to 1100 hPa (+9000m to -500m compared to sea level) with an accuracy down to 0.02hPa (0.17m) in advance resolution mode. Both the weather and altitude affect the pressure [58].

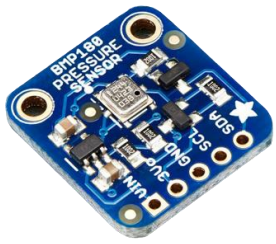


Figure 8. BMP 180 Barometric Pressure Sensor

- **Rain Gauge**

A rain gauge is a meteorological instrument, which will be used to measure the rainfall depth and other precipitation at any given time per unit area. It is usually expressed in mm [59].

Figure 9. Rain Gauge

- **Temperature and Humidity**

In this study, the proponents used a DHT22 sensor to measure the temperature and humidity of a certain area with the Arduino Board. It is much better compared to the DHT11 sensors in terms of precision, accuracy and larger scale of detecting temperature and humidity. However, it is costly and much larger [60].

Figure 10. DHT22 Sensor

- **Water Level (Ultrasonic)**

JSN - SR04T is an ultrasonic distance sensor which uses sonar to determine the distance to an object. This sensor reads from 20cm to 600cm with an accuracy of 2mm. It has a dimension of 23.5 x 20mm, 2.5m long cable. In this study, it will be used to construct systems that accurately detect wave height and water levels at significantly reduced installation and maintenance costs because they use high-frequency soundwaves in determining the distance to a remote object without physically touching it [61].

Figure 11. Ultrasonic Sensor (JSN-SR04T)

- **Float Switch**

  The Water Level Sensor Float Switch is a tool for detecting the liquid level in a tank; it can activate a pump, an indicator, an alarm, or other device. This particular style of horizontally mounted float switch can be changed from a normally-open to a normally-closed switch by simply rotating it 180 degrees. An internal magnet in the device activates a sealed reed relay when the water level reaches the float, which then moves [62].

Figure 12. Float Switch (Side Mount Level Sensor)

3.3.2.2 PacketDUINO

PacketDUINO is a development board for Arduino that incorporates LoRaWAN technology, which enables it to transmit data over long distances. The board is powered by an atmega32u4 microcontroller and features a USB port for programming. One of the key advantages of the PacketDUINO is its ultra-long-range capability, which can reach up to 15km, and its energy efficiency, making it an ideal choice for IoT end nodes. The board also uses Semtech's patented LoRa spread spectrum modulation technique, which is also used by the PacketPro connected to the Arduino by SerialSoftware connect, providing high immunity to signal interference [63].

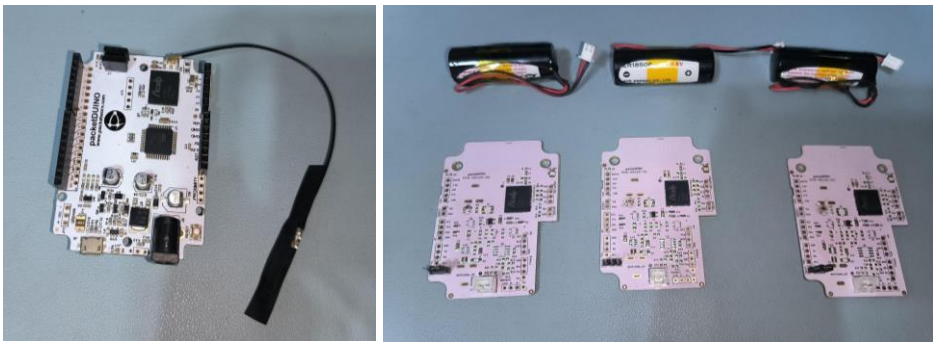


Figure 13. PacketDUINO and PacketPro

3.3.2.3 Arduino Uno

Using the ATmega328 as its foundation, the Arduino Uno is a microcontroller board. The device features 20 digital input/output pins, a 16 MHz resonator, a USB port, a power connector, an in-circuit system programming (ICSP) header, and a reset button. Of these, 6 pins may be used as PWM outputs and 6 as analog inputs. In this study, it will be used to collect the data coming from the sensors and responsible for transmitting the data to the NodeMCU Wi-Fi Module [64]**.**

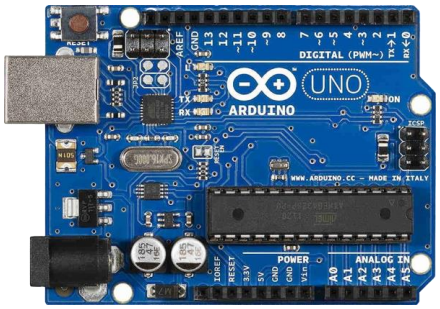


Figure 14. Arduino Uno

3.3.2.4 Buck Converter

DC-DC Buck Converter with a 9V output will be used in this study to power the microcontrollers and sensors since the battery is 12V. The current input and output voltages can be displayed in this specific DC-DC Buck Converter with 7

Segment Display LM2596. With an output voltage that can be changed from 1.25 to 30 volts via a trimpot or trimmer potentiometer, the LM2596 step-down switching regulator has a current capacity of up to 3A [65].

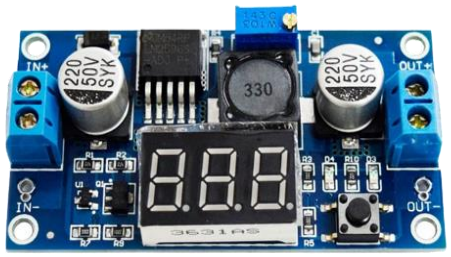


Figure 15. DC-DC Buck Converter with 7 Segment Display LM2596

3.3.2.5 Lithium - Ion Battery

An 18650 battery will be used in this study to supply the required voltage of the microcontrollers and sensors. An 18650 battery is a lithium-ion rechargeable battery. The battery is 18mm in diameter and 65mm in length, hence the numbers "18650" on it. Electronics like computers and torches, as well as electric cars and other high-power applications, frequently employ 18650 batteries. They have a reputation for having a high energy density, a lengthy lifespan, and a relatively low self-discharge rate [66].

Figure 16.18650 Lithium-Ion Battery (3.7V)

3.3.2.6 Solar Panel

A 15W Monocrystalline Solar Panel is used as the source of power in this study where it converts sunlight into electrical energy through photovoltaic effect. It has a maximum power of 15 watts, maximum power voltage of 18V [67].

Figure 17. 15W Monocrystalline Solar Panel

3.3.2.7 Solar Charge Controller

The power entering the battery bank from the solar array is controlled by a solar charge controller. It prevents the deep cycle batteries from being overcharged during the day and from having power run backwards to the solar panels at night, which would drain the batteries. In this study, A PWM solar charge controller will be used. Pulse Width Modulation is the abbreviation for a solar charge controller. These work by connecting the solar panel array and battery bank directly. When the array and battery bank are connected continuously during bulk charging, the array output voltage is "pulled down" to the battery voltage [68].

Figure 18. PWM Solar Charge Controller

3.3.3 Calibration of Hydrometeorological Sensors

3.3.3.1 Anemometer

It is ideally mounted above ground level and a horizontal distance of 10 times the height of the nearest obstruction [69]. In this project, it will be placed in an open area to avoid obstruction.

3.3.3.2 Barometric Pressure Sensor

This sensor can be placed either inside or outside. It is best to stay away from the sunlight and other sources such as heat, drafts, and winds. In this project, the researchers should determine the current atmospheric pressure in the area using a local weather source, and then adjust the barometer to that pressure to get the accurate result [70]. It will be placed in an open area to avoid obstruction.

3.3.3.3 Rain gauge

In the open areas, the proper placement of rain gauges of distances from obstacles are twice as far from the height of obstacles. It is mounted approximately 2 feet off the ground [71]. In this project, it is ideally placed in the open space area that does not have any higher building since it is considered as open space to avoid obstruction.

3.3.3.4 Temperature and Humidity Sensor

An ideal place to be mounted at 1.5 to 2.0 m above the ground. It should not be placed on chimneys, air vents, and air conditioners [71]. In this project, it will be deployed in an open space area with the rain gauge, barometer, and anemometer.

3.3.3.5 Water Level (Ultrasonic)

The proper mounting of an ultrasonic sensor is putting the sensor away from direct sunlight. Establish the sensor's face is perpendicular to the target surface. Also, it should be installed in a position that allows an unobstructed column of air to flow from the sensor to the target. Even at greater distances, a 3–4-foot diameter unobstructed column is required [72]. In this study, it is ideal to be placed under the bridge.

3.3.3.6 Float Switch (Side Mount Level Sensor)

A float switch is made up of an electrical cable and a floating switch. To prevent the depth of the float switch from changing, the electrical wire needs to be fixed in place. It could be put alongside the pipe that is lowered down the river vertically. As the float rises, it may close (turn on) a "Normally Open" circuit, signaling that a specific amount of water has been reached, or it may open (turn off) a "Normally Closed" circuit, signaling that the water level has risen above the threshold level [73].

3.3.4 Sensors and Its Parameter and Unit

Table 8. Table of Sensors and Its Parameter and Unit

Sensor	Parameter	Unit
DHT22 Sensor	Temperature	Celsius (°C)
	Relative Humidity	Percentage (%)
Anemometer	Wind Speed	Meter per second (m/s)
BMP180 - Barometric Pressure Sensor	Atmospheric Pressure	Millibar (mb)
Rain Gauge Sensor	Rainfall	Millimeters per hour (mm/h)
Ultrasonic Sensor (JSN-SR04T)	Water Level	Centimeter (cm)
Float Switch	Water Level	Level (Normal, High, Critical)

3.3.5 Power Budgeting

Table 9. Power requirements of the Weather Station System

Devices	Max. Watt Consumption	Operating Hour/s	Watt Hour	Ideal Backup Storage
PacketDUINO	0.55 W (5min. interval sleep time)	24 Hours	13.2 Wh	2 days
Anemometer	0.14 W		3.36 Wh	
Wind Vane	0.14 W		3.36 Wh	
Total	0.83W		19.92 Wh	

Table 10. Power requirements of the Flood Monitoring System

Devices	Max. Watt Consumption	Operating Hour/s	Watt Hour	Ideal Backup Storage
Arduino UNO with PacketPro	0.6 W (5min. interval sleep time)	24 Hours	14.4 Wh	2 days
Ultrasonic Sensor	0.05 W (5min. interval sleep time)		2.4 Wh	
Total	0.7W		16.8 Wh	

To achieve an ideal backup storage of 2 days, the battery must have a milliampere-hour of 3320 or above for the weather station system, and 4200 or above for the flood monitoring system. Thus, the proponents will use a 3.7V lithium-ion battery connected in series to achieve the 12V (3.7×3 = 11.1V) and 5V requirement (3.7×2 = 7.4V) with a 4500mAh battery to ensure that it can operate it in more than 2 days without sunlight. For the solar panel, the proponents will be using 15Watts solar panel monocrystalline solar panel. The reason is because it can charge the battery with 300 Watts considering a day with 6 hours of sunlight.

Table 11. Power Supply Calculation for the Weather Station System

Devices	Max. Watt Consumption	Watt Hour	11.1V 4500mAh Battery	Theoretical Battery Life	Solar Panel (15W)
Arduino Uno	0.55 W (5min. interval sleep time)	13.2 Wh	49.95 Watts	**60.18 hours**	15 Wh × 6 hrs. = **90 Watts** **Note:** Assuming 6hrs sunlight
Anemometer	0.14 W	3.36 Wh			
Wind Vane	0.14 W	3.36 Wh			
Total	0.83W	19.92 Wh			

Table 12. Power Supply Calculation for the Flood Monitoring System

Devices	Max. Watt Consumption	Watt Hour	7.4V 4500mAh Battery	Theoretical Battery Life	Solar Panel (15W)
Arduino UNO with PacketPro	0.6 W (5min. interval sleep time)	14.4 Wh	33.3 Watts	**47.57 hours**	15 Wh × 6 hrs. = **90 Watts** **Note:** Assuming 6hrs sunlight
Ultrasonic Sensor	0.05 W (5min. interval sleep time)	2.4 Wh			
Total	0.7 W	16.8 Wh			

3.3.6 Integration of Hydrometeorological Sensors and IoT Sensing Nodes

The PacketDUINO for the weather monitoring, and Arduino Uno with PacketPro will be used to establish a link between every sensor. The PacketDUINO and Arduino with PacketPro will act as a node that will send the data from each sensor to one gateway. After the Arduino collects the data, this data will be forwarded through LoRaWAN and will be received by Packetworx database, ThingSpeak. There will be a gateway for all the nodes, which allows the sensor nodes to send the collected data to the Internet, specifically on the database. In general, the PacketDUINO and Arduino UNO with PacketPro will be the primary devices to connect to IoT for monitoring and predicting purposes.

3.3.7 LIGTAS Hardware Model

Figure 19. 3D Model of the LIGTAS (front - view)

Figure 20. 3D Model of the LIGTAS (back- view)

Figure 21. Enclosure for the Microcontroller and Microprocessors

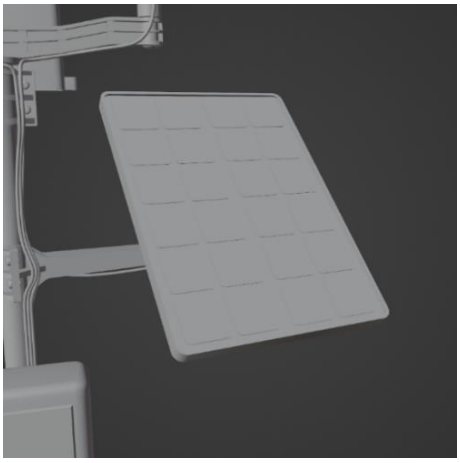


Figure 22. Solar Panel

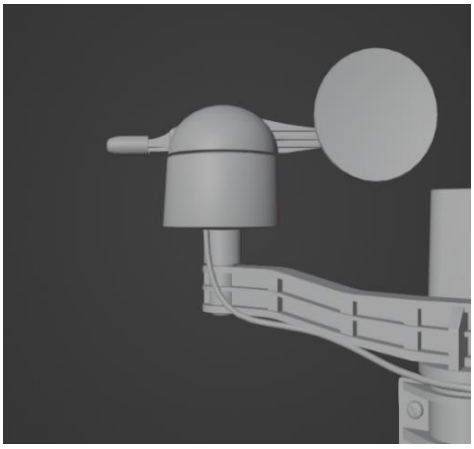


Figure 23. Wind Vane and Enclosure for the Temperature and Humidity Sensor

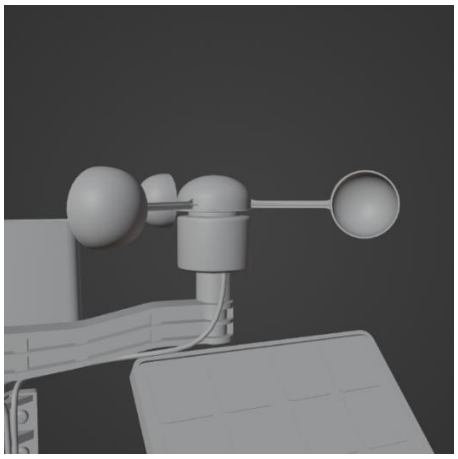


Figure 24. Anemometer and enclosure for Barometer

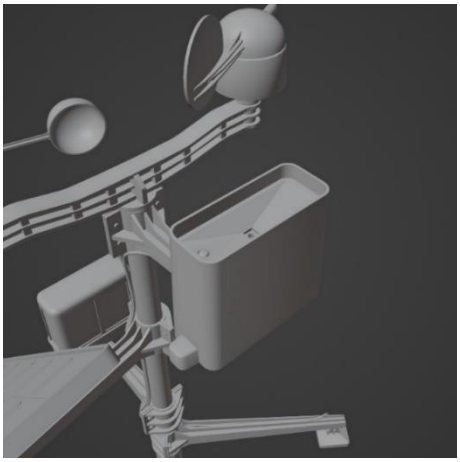


Figure 25. Rain Gauge

Figure 26. Measurement of the Circuit Box of the LIGTAS Model

Figure 27. Measurement of the Pole of the LIGTAS Model

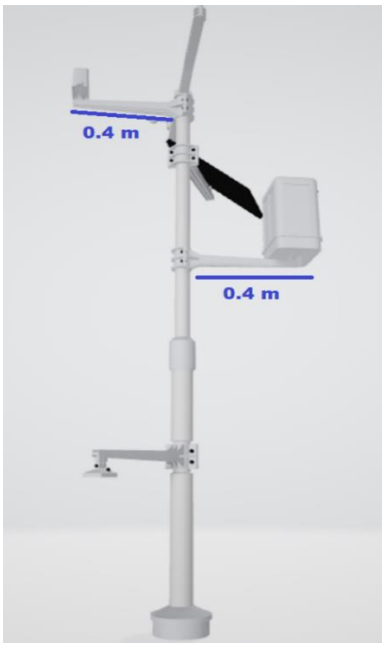


Figure 28. Measurement of the Arm Length of the Circuit Box and Rain Gauge of the LIGTAS Model

Figure 29. Measurement of the Arm Length of the Solar Panel and Ultrasonic Sensor of the LIGTAS Model

Figure 30. Measurement of the Solar Panel of the LIGTAS Model

Figure 31. Measurement of the Top Arm Length for the Wind Vane and Anemometer of the LIGTAS Model

3.4 Software Development

3.4.1 Design of the Web-based Application

The proposed project involves the development of a web application that caters to the needs of the Las Disaster Risk Reduction and Management Council (LPDRRMC), barangay offices, and citizens as end-users. This web application will be accessible on both computers and mobile phones, requiring the construction of a front-end and back-end framework to ensure the optimal user experience. The front-end component is responsible for presenting the user interface and its associated features, while the back-end component is responsible for processing the data and executing the required functionalities.

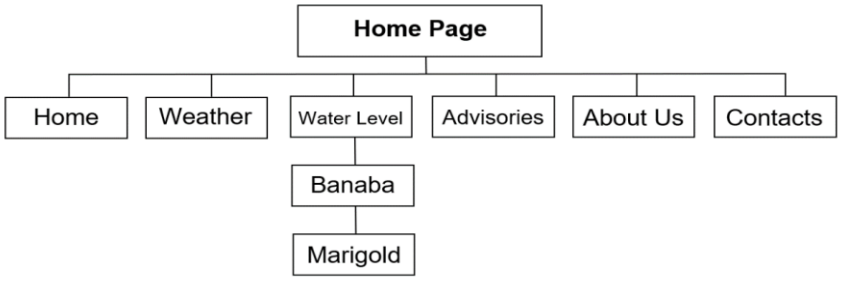


Figure 32. Web Hierarchy of Web Application

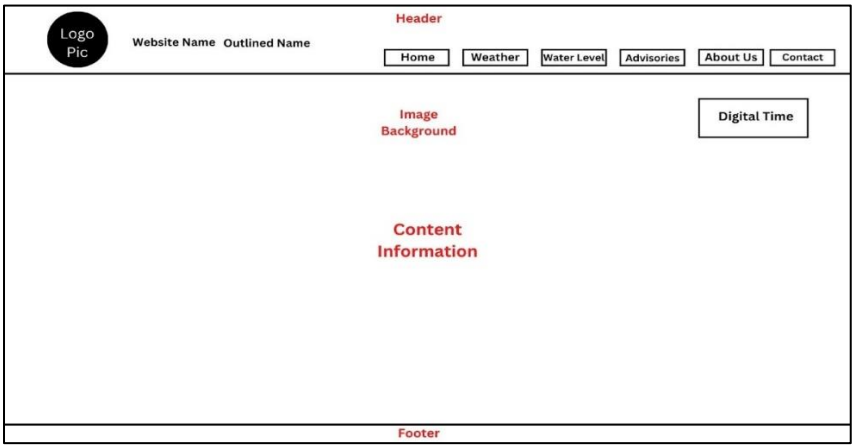


Figure 33. Layout of the Weather Navigation in Web Application

3.4.1.1 Front - end development

The proponents used HTML, CSS, and JavaScript (JS) as the scripting languages for creating the content of the web - application. HTML will handle the structure of contents, CSS will provide the presentation of the contents, and JavaScript (JS) will support the interactivity of the web - application.

3.4.1.2 Back - end development

The back - end of web application was developed using Node.js as the primary programming language. Node.js is a popular choice for building back-end web applications due to its event-driven, non-blocking I/O model, which makes it highly performant and scalable. The back-end of the system is built using Express.js, a web framework for Node.js that allows us to easily define routes and handle HTTP requests, and provides middleware for tasks such as authentication and data validation. Additionally, Socket.Io was utilized to enable real-time, bidirectional communication between the server and client. Several Node.js modules were utilized, including bcrypt for password hashing and validation, and MQTT for the data retrieval in the MQTT Server. After the MQTT Client to the Server-side was connected and subscribed to the MQTT topics, the system was able to retrieve data from MQTT Server. The retrieved data was then processed and stored in a database for weather monitoring, train the machine learning model, and forecasting. To manage dependencies, the proponents utilized the Node.js package manager, npm. Throughout the development process, tools such as Postman were used to test API endpoints. Node.js provided a powerful and flexible platform for building the back-end,

allowing management, and ensuring the application security and scalability. Furthermore, MySQL will serve as the database which will be the one responsible for collecting, storing, processing, and managing the collected data from the sensors and nodes of the system. For the deployment of the model, the proponents opted to use the FASTAPI application, which is known for its suitability in deploying models. FASTAPI is a high-performance web framework for building APIs in Python. It offers features like automatic validation, serialization, and documentation generation, making it an ideal choice for deploying models and creating robust web applications. Its asynchronous capabilities enable efficient handling of concurrent requests, making it suitable for applications with high traffic or real-time requirements [74].

With the development of front - end and back - end of the system, the proponents will be able to develop a web application with the following features:

3.4.1.3 Features

3.4.1.3.1 Real - time Monitoring

This feature lets the user view the temperature, humidity, wind speed, air pressure, rainfall depth, and water level in real - time through the web - application.

3.4.1.3.2 Forecasted weather and flood level

This attribute will help the users decide early to perform precautions and preventive measures as it displays the weather and flood level forecast.

3.4.1.3.3 Two - way communication

Through the "Contact Us" page, users will be able to reach out to the relevant authorities in case of emergencies, such as a fire outbreak. This feature has been incorporated into the proposed software's web application design to facilitate seamless communication between end-users and authorities.

3.5 Artificial Neural Network Analysis

3.5.1 Selection of Dataset of ANN Models

In selecting the proper place for the deployment of the system, Barangay Pulang Lupa Dos, Barangay Talon Dos, and Barangay Manuyo Dos in Las Pinas City is ideal for this study. Following this, Mr. Dela Merced, the Officer-in Charge of LPDRRMC stated that the city has frequently experienced major floods due to typhoons, east monsoon, and high tides. The data that will be used in this study will come from the collected data from the sensors in each time interval. Rainfall Intensity, Temperature, Relative Humidity, Atmospheric Pressure, Wind Speed, Wind Direction, and Water Level will all be used as input variables for both Weather and Flood Forecasting.

3.5.2 Recurrent Neural Networks (RNN)

The Recurrent Neural Network (RNN) which is a variant of Artificial Neural Network (ANN) will be used to perform predictive operation on the time series data that will be collected through the hydrometeorological sensors. The RNN shows promising results as it outperformed other models like the ARIMA and SVM in terms of accuracy, MSE, and RMSE. Moreover, the RNN model is more suitable for long term prediction. Recurrent neural networks (RNN) have the ability to learn sequences with respect to time since they feature an additional recurrent connection to the hidden layers than feed-forward neural networks which makes it a powerful neural network because of its internal memory. Unlike in a feed - forward neural network which only lets signals flow in one direction, in a recurrent neural network, the output of a layer is added to the next input and fed back into the same layer, which is typically the only layer in the entire network. Therefore, a recurrent neural network can receive a sequence of values as inputs, and it can also produce a sequence of values as output. Hence, RNN will be utilized as a forecasting prediction model in time series data.

3.5.2.1 Long Short-Term Memory (LSTM) for Weather Forecasting

The Long Short-Term Memory (LSTM) is a type of RNN. LSTM has demonstrated significant performance in a range of real-world applications due to its capacity to record long-term dependencies.

The LSTM process is a little complicated as it requires input from three states, including the current input state, the short-term memory from the previous cell, and lastly the long-term memory. Before passing on the long-

term and short-term information to the following cell, these cells use the gates to control the information that will be kept or deleted during loop operation. The gates are called Input Gate, Forget Gate, and Output Gate. These gates can be thought of as filters that eliminate unwanted, undesired, and irrelevant information.

3.5.3 Preparation of Dataset for Weather Forecasting and Flood Prediction

As with any data-driven model according to Campolo et al [75], the data used during model setup and calibration have a significant impact on how well neural networks perform. First, the data must be sufficient and specific for the modeling task. Input data must be chosen based on their relevance to the modeling. Also, input data must be as limited as possible to minimize training time and the vulnerability of overfitting. Moreover, since the model that will be used is LSTM, which employs feedback on its signal flow, a function will be created to employ it. The function will chunk the input data into a list with a certain window length which will drop the first value and will be appended by a new value every time an output is predicted by the model.

3.5.4 Discretization of Dataset for Training and Testing Process

The LSTM model will be trained by providing “examples” for the models to learn. The “well-learned” LSTM will then be tested with some unseen data. To accomplish this, the hydrometeorological data obtained were divided into training, validation, and testing data.

The testing method that will be adopted for this research will be by percentage split. The first percentage of the data set will be the train set for the training of the

model, obviously. Meanwhile, the validation set will be used for the evaluation and cross validation of the model in the future. The remaining percentage is the test set which will be then used for testing the model further in the future. Thereafter, interesting patterns representing knowledge will be identified.

3.5.5 Back Propagation Algorithm

The back propagation algorithm will be employed once data discretization is done. It employs supervised learning, which means the model educates itself using the intended results. Each set of input data includes the target output. The neural network model processes the input data using proper activation functions, random weight values using one or more hidden layers in between, and generates the expected output. This targeted output is then evaluated against the expected output using the same input dataset.

In relation, training an RNN uses backpropagation through time and at each time step or loop operation gradient is being calculated. The gradient's purpose is to update the weight value of the neural network. If the weight is too low, the gradient will vanish and the hidden layer next to the input layer would stop learning. On the other hand, if the weight is too high, the gradient will explode. To overcome the impact of short-term memory, LSTM uses a gate structure. The information flow across the sequence chain can be regulated by the gate structure.

3.5.6 Lead Time of the System

Lead time is a crucial factor when it comes to flood forecasting and early warning systems. The lead time is the length of time between the release of the forecast and the occurrence of the phenomena. Weather forecasts need to be of finer

spatial scale and temporal resolution. Thus, lead time has an inverse relationship with forecast accuracy. The flood would have a high accuracy prediction in the later stages since more information can be observed [76]. The LIGTAS weather system would provide a 6-hour lead time to have accurate and reliable forecasts and to warn the communities for timely and responsive preparations to achieve the reduction of negative weather impacts.

3.6 Research Locale

The study will be conducted at Barangay Pulang Lupa Dos, Barangay Talon Dos, and Barangay Manuyo Dos in Las Piñas, Metro Manila. The proponents chose this area as it is one of the most prone to flood in the city of Las Piñas which was also mentioned by the Las Pinas Disaster Risk Reduction Management Council (LPDRRMC) as one of their priority areas when it comes to flood. The proponents will install the weather system at the DRRMO Building in Barangay Talon Dos and the flood system in Barangay Manuyo Dos and Barangay Pulang Lupa Dos in Las Piñas, Metro Manila. The proponents of this study will install the system properly to have proper measurements that generate a higher accuracy.

3.7 User-Acceptance and Field Testing of the LIGTAS system

The proponents of this study will develop usability and field testing of the system to the clients and end-user. It is used to validate the software and hardware of the system whether the solution provided is suitable for their needs before releasing it for broad consumption and transferring of the technology.

3.7.1 User- Acceptance

The web-based application of the system will be evaluated to ensure that it meets the criteria for which it was designed. It will be evaluated by the officials of Las Pinas Disaster Risk Reduction Management Council (LPDRRMC), barangay personnel and residents of Barangay Pulang Lupa Dos, Barangay Manuyo Dos, and Barangay Talon Dos in Las Pinas, Metro Manila. Shown below is the technical evaluation form.

LIGTAS: Localized Weather Monitoring Management and Early Warning System for Las Piñas City via Machine Learning Survey					
Introduction: The proponents will conduct a survey to evaluate the study, entitled **LIGTAS: Localized Weather Monitoring Management and Early Warning System for Las Piñas City via Machine Learning**					
Instruction: Please rate how strongly you agree or disagree with each statement. Check the following scale: (1) Strongly Disagree (2) Disagree (3) Neutral (4) Agree (5) Strongly Agree					
Characteristics	**Rating**				
	1	2	3	4	5
Functionality and Efficiency					
1.The system is more efficient and provides higher accuracy than manual or traditional methods.					
2. The prediction of flood occurrence and weather forecasting is determined through the system.					
3. The system provides real time information.					

4.	The system prevents unauthorized access.					
5.	The system can be operated 24 hours.					
Usability						
6.	The system can be learned easily.					
7.	The application of the system is easy to use.					
8.	The system operates without delay.					
9.	The application of the system is pleasing to the eye.					
Maintainability and Reliability						
10.	The system can be maintained easily.					
11.	The system can restore the data in case of failure.					
12.	The system is capable of handling errors.					
Safety and Portability						
13.	The system is installed properly.					
14.	The deployment of the system does not have harmful effects.					
15.	The system can be adapted to the community.					

Figure 34. Technical Evaluation Form

3.8 Statistical Analysis

In this study, the data to be gathered will be subjected to various statistical analyses to evaluate the accuracy of the trained predictive model. Since the researchers will deal with time-series data, several regression metrics, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE), will be used to

measure forecast accuracy. The MSE is used to evaluate the average squared difference between the predicted and actual values. RMSE, being the square root of MSE, offers a measure of the average disparity between the forecasted and actual values. Lastly, MAE provides the average of the absolute difference between forecasted and ground truth values. Low values for these regression metrics imply a highly accurate predictive model with strong performance. Utilizing these regression metrics will help the researchers in precisely identifying the accuracy of the predictive model. Moreover, the results from these regression metrics will also aid in fine – tuning the predictive model as they show how high or low the errors are in the model's predictive capabilities.

Furthermore, to evaluate the system's overall effectiveness and pinpoint areas for improvement, a survey will be conducted among residents of Barangay Pulang Lupa Dos, Manuyo Dos, and Talon Dos. The survey will employ a Likert scale evaluation form with a five-point rating system. The median of the collected responses will be calculated to analyze the data, providing insight into the system's strengths and weaknesses from the users' perspective.

3.9 Gantt Chart

		Month 1				Month 2				Month 3				Month 4				Month 5				Month 6				Month 7				Month 8				Month 9			
30%	**Hardware Development**																																		
3%	Designating Suitable Site for the Weather Station System																																		
3%	Purchasing of sensors, microprocessors, and solar panels																																		
7%	Calibration of Sensors																																		
4%	Construction of the Weather Station System																																		
3%	Design of Power allocation and requirements for the sytem																																		
10%	Create a program to gather the data from the sensors																																		
35%	**Development of Web-based Application**																																		
15%	Front-end development																																		
17%	Back-end development																																		
3%	Incorporate the developed forecast model to the LIGTAS website.																																		
25%	**Training of Artificial Neural Network**																																		
2%	Identifying the Prediction Model Algorithm																																		
3%	Data Gathering																																		
2%	Data Preparation																																		
2%	Data Discretization																																		
10%	Model Training																																		
4%	Tuning Algorithm																																		
2%	Out-of-sample testing																																		
10%	**User- Acceptance Testing**																																		
5%	Conduct a Survey for the Evaluation of Web Application and usability of the system																																		
5%	Evaluation of the Result																																		

Figure 35. Gantt Chart

Chapter 4

DATA AND RESULTS

4.1 Project Technical Description

The study LIGTAS: Localized Weather Monitoring Management and Early Warning System for Las Piñas City via Machine Learning is focused on monitoring and predicting weather in the local area, specifically Las Piñas City. It is composed of two systems that have the same process of collecting data. For the weather system, it is using DHT22 sensor that gathers temperature and humidity, barometric pressure sensor for atmospheric pressure, anemometer for wind speed, wind vane for wind direction, and rain gauge for rainfall intensity. On the other hand, the flood monitoring system utilizes ultrasonic sensors and float switches to monitor the water level status. This system simultaneously predicts new data for the next few minutes or hour. The system starts at collecting new data that will be transferred into the database. The real-time data will be displayed in the web-application and this data will be used for forecasting. Then the forecasted data will be also displayed in the web-application.

4.2 Project Structural Organization

4.2.1 Installation Plan of the LIGTAS System

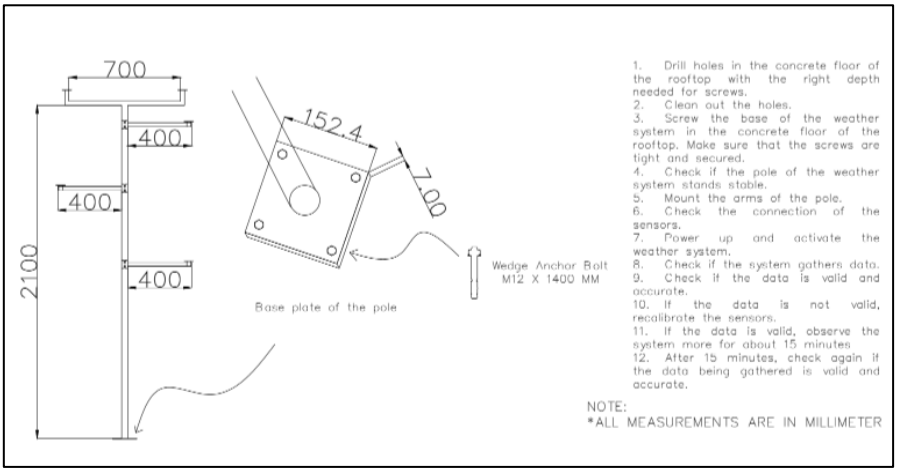


Figure 36. Installation Details of Weather Monitoring System

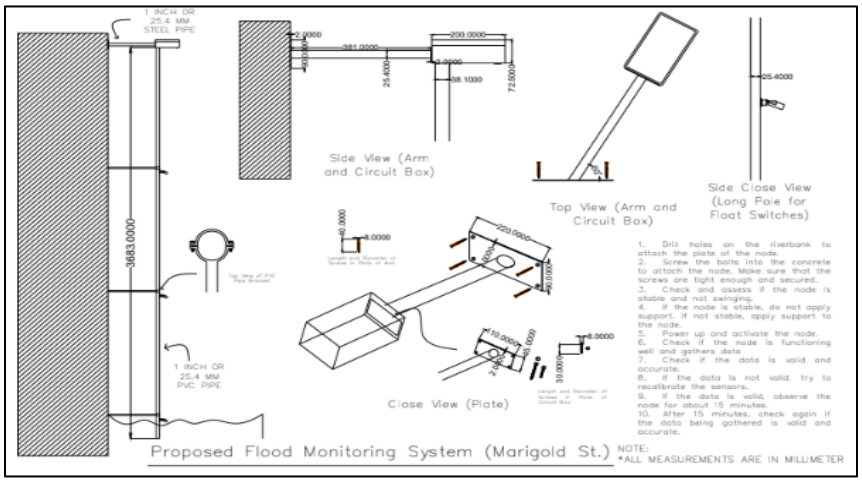


Figure 37. Installation Details of Flood Monitoring System

In Figures 35 and 36, the proposed installation details of the LIGTAS system were designed to meet the criteria to ensure that the systems are working properly. The installation design and details were approved by the engineering office of Las Piñas City to meet the requirements of proper gathering of data that is needed for monitoring and prediction.

4.2.2 LIGTAS System installed in Las Piñas City

Figure 38. LIGTAS Weather Station in DRRMO Building, Brgy. Talon Dos

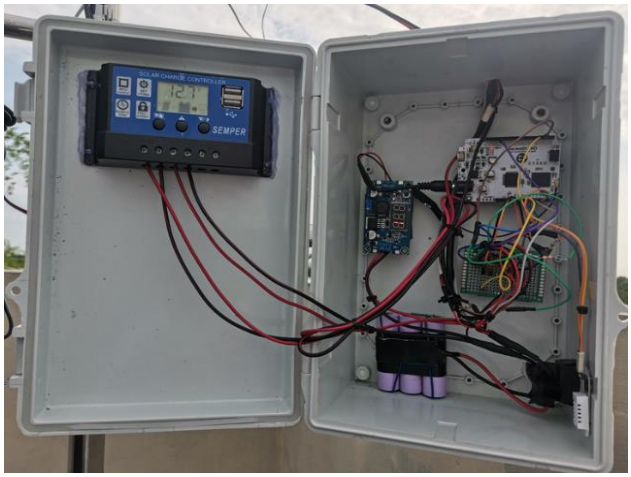


Figure 39. Circuit Box of LIGTAS Weather Station

Figure 40. Flood Monitoring Node in Banaba St., Brgy. Manuyo Dos

Figure 41. Flood Monitoring Node in Marigold Bridge, Brgy. Pulang Lupa Dos

Figure 42. Flood Monitoring Node in Naga Road, Brgy. Pulang Lupa Dos

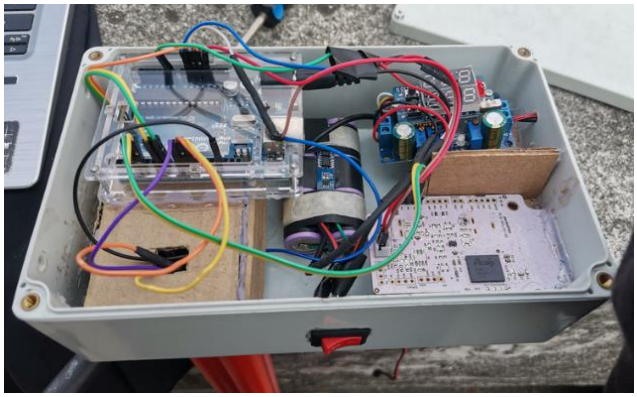


Figure 43. Circuit Box of Flood Monitoring System

4.2.3 Deployment Area

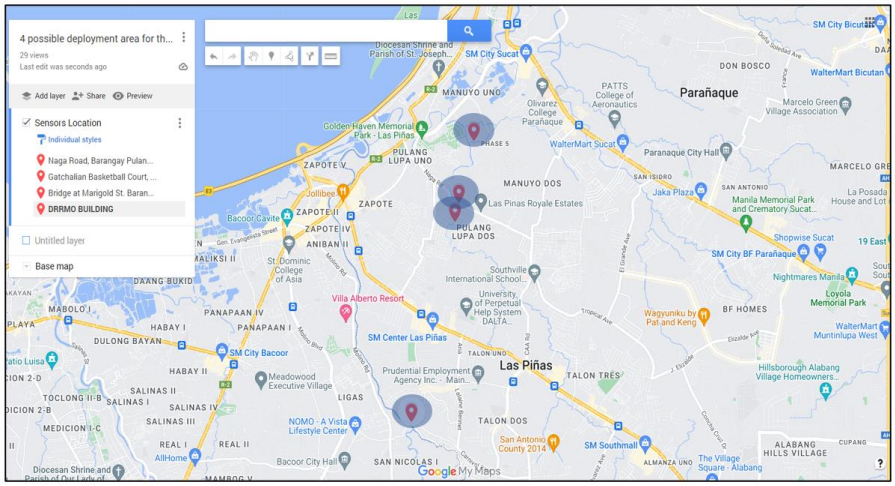


Figure 44. Location of Deployment Areas in Las Piñas City

Figure 44 shows the pinned location in the Google Maps. These pinned locations are where the system installed. The weather Disaster Risk Reduction Management Office (DRRMO) Building, Gatchalian Basketball Court in Banaba, Bridge in Marigold, and Naga Road. The locations are said to be flood prone areas in the Las Piñas City. To scale the coverage of the system, the following are the distances between the areas:

- Banaba to DRRMO Building: **6 km**
- Naga Rd. to Banaba: **1.2 km**
- Naga Rd. to Marigold Bridge: **400 m**
- Marigold Rd. to DRRMO Building: **4.6 km**

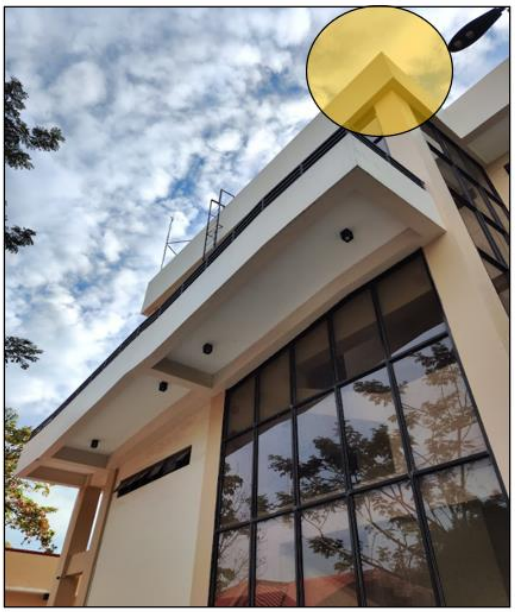


Figure 45. Disaster Risk Reduction Management Office (DRRMO) Building

LOCATION: CXPH+W29, Florlina Dela Cruz, Las Piñas, 1747 Metro Manila

GRID COORDINATES: 14.437289476070028, 120.97759546687232

Figure 46. Gatchalian Basketball Court in Banaba

LOCATION: Banaba Las Piñas, Metro Manila

GRID COORDINATES: 14.4704791, 120.9865827

Figure 47. Marigold Bridge

LOCATION: Bridge at Marigold St. Barangay Pulang Lupa Dos

GRID COORDINATES: 14.46273, 120.98379

Figure 48. Naga Road

LOCATION: Naga Road, Barangay Pulang Lupa Dos, Metro Manila

GRID COORDINATES: 14.46515, 120.98425

4.2.4 Calibration of the Sensors

The sensors are calibrated with the aim of achieving highly accurate readings. This calibration ensures that all the data collected and displayed in the web application is correct and precise. Another reason for this emphasis on accuracy is the reliability of the generated model. The data fed into the model must be both correct and accurate to produce reliable forecasted data. The tables below illustrate the changes in data after the calibration process has been completed.

A. Testing before the calibration

The following data presents the readings obtained from the system's sensor, as well as the data collected from an external sensor independent of the system.

Table 13. Anemometer Reading before Calibration

Systems Sensor Reading (km/hr)	External Sensor Reading (km/hr)	Percent Difference
15.2	18.7	18.72%
12.6	16.4	23.17%
14.9	18.5	19.46%
11.5	14.8	28.38%
13.8	17.5	21.14%
10.9	13.8	26.49%
12.1	15.4	27.27%
14.5	18.3	20.77%
16.3	21.1	22.27%
13.2	17.0	22.35%

Table 14. Wind Vane Reading before Calibration

Systems Sensor Reading (degrees)	External Sensor Reading (degrees)	Percent Difference
135	156	15.38%
180	216	20.00%
225	257	14.29%
45	54	16.67%
315	368	16.67%
90	108	20.00%
270	324	20.99%
360	432	20.37%
135	162	20.00%
180	216	20.00%

Table 15. Atmospheric Pressure Sensor Reading before Calibration

System Sensor Reading (mb)	External Sensor Reading (mb)	Difference (mb)	Percent Difference
1015.2	1014.4	0.8	0.08%
1019.7	1018.1	1.6	0.16%
1011.4	1013.5	2.1	0.21%
1018.9	1016.8	2.1	0.21%
1014.6	1013.1	1.5	0.15%
1009.8	1009.3	0.5	0.05%
1022.1	1020.7	1.4	0.14%
1016.3	1014.9	1.4	0.14%
1011.7	1009.9	1.8	0.18%
1017.9	1016.4	1.5	0.15%

Table 16. Temperature and Humidity Sensor Reading before Calibration

System Temp (°C)	External Temp (°C)	Temp Percent Difference	System Humidity (%)	External Humidity (%)	Humidity Percent Difference
24.5	24.3	0.82%	53.2	53.4	0.38%
26.0	26.2	0.77%	52.7	52.8	0.19%
25.2	24.9	1.19%	51.9	51.7	0.39%
23.8	23.4	1.69%	51.5	51.3	0.39%
27.1	27.3	0.73%	52.2	52.0	0.38%
24.9	25.1	0.80%	53.5	53.3	0.38%
26.5	26.7	0.75%	52.8	53.0	0.38%
25.4	25.7	1.17%	52.1	51.9	0.39%
23.9	23.6	1.27%	51.7	51.5	0.39%
27.3	27.0	1.11%	52.9	53.1	0.38%

Table 17. Ultrasonic Sensor Reading before Calibration

System Sensor Reading (cm)	Traditional Method Reading (cm)	Percent Difference
207	210	1.43%
209	207	0.96%
208	211	1.42%
210	213	1.41%
207	209	0.96%
209	206	1.45%
208	212	1.89%
210	207	1.45%
207	210	1.43%
209	207	0.96%

Table 18. Testing and setting up the Rain Gauge

Trial Number	Drop per tick of Rain Gauge (ml)	Average drop per tick (ml)	Calculated drop per tick in mm
1	1.6	1.7	0.309091
2	1.8		
3	1.8		
4	1.7		
5	1.6		

The table above presents a comparison between the readings from the system sensors and the external sensors or weather instruments, aiming to assess the accuracy and reliability of the system's data. This comparison is crucial to ensure that the recorded readings are precise and free from errors. The atmospheric pressure sensor, ultrasonic sensor, as well as the

temperature and humidity sensor, exhibit minimal percent differences, indicating that no additional calibration is required. To determine the average drop per tick for the Rain Gauge, multiple trials were conducted. Once the average drop per tick was obtained, the calculated drop per tick was determined by dividing the average drop per tick by the catchment area of 55 sq. cm (5 cm x 11 cm). However, for the anemometer and wind vane, the percentage difference is relatively higher, highlighting the need for hardware and code improvements for enhanced accuracy.

B. Testing after the Calibration

Table 19. Anemometer Reading after Calibration

Systems Sensor Reading (km/hr)	External Sensor Reading (km/hr)	Percent Difference
15.3	15.7	2.55%
18.1	17.6	2.81%
21.7	22.4	3.13%
14.9	14.4	3.47%
19.6	20.2	2.97%
17.9	18.3	2.19%
23.4	24.0	2.50%
13.2	13.6	2.94%
20.3	19.5	4.10%
16.6	16.9	1.77%

Table 20. Wind Vane Reading after Calibration

Systems Sensor Reading (degrees)	External Sensor Reading (degrees)	Percent Difference
165	162	1.85%
198	204	2.94%
234	230	1.72%
110	114	3.51%
285	278	2.52%
143	140	2.10%
56	54	3.57%
247	241	2.43%
317	311	1.90%

After implementing additional hardware modifications, such as adding two resistors to form a voltage divider, and making corresponding changes in the code, the desired percent difference has been successfully achieved. In the case of the anemometer, the percent difference has significantly reduced from approximately 18-26% to a range of 2-5%. Similarly, the wind vane now exhibits a percent difference of 2-4%, a substantial improvement compared to the initial range of 14-20% prior to calibration.

4.3 Graphical User Interface of the LIGTAS Web Application

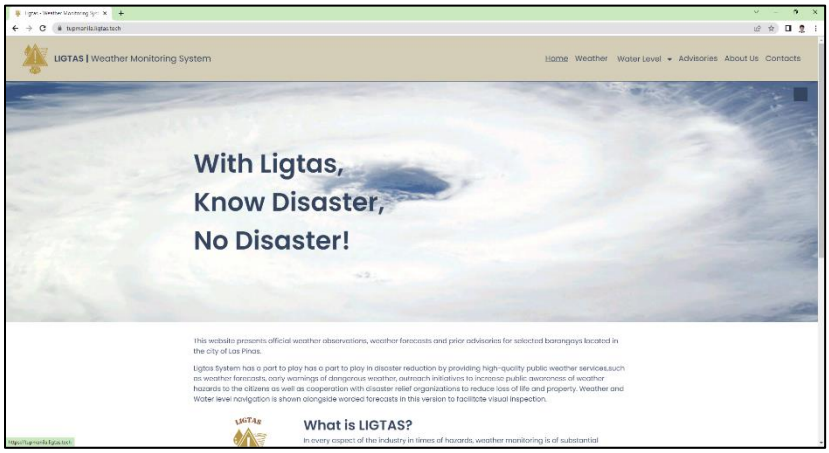


Figure 49. LIGTAS Home Page

In figure 49, the home page is designed to be simple and user-friendly. The logo displayed represents the LIGTAS brand and creates a visual identity for the application. Additionally, it provides an overview of the LIGTAS Weather Monitoring System and informs the users about the purpose and benefits of the system.

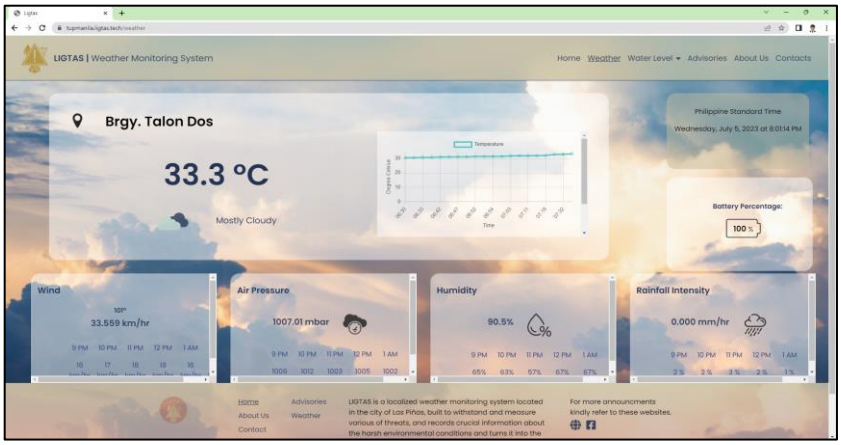


Figure 50. LIGTAS Weather Page

In figure 50, users can effortlessly stay updated on the latest weather conditions in their area. Since the LIGTAS web application provides real-time weather updates such as temperature, humidity, atmospheric pressure, wind speed, wind direction, and rain intensity along with the weather conditions over the next six hours.

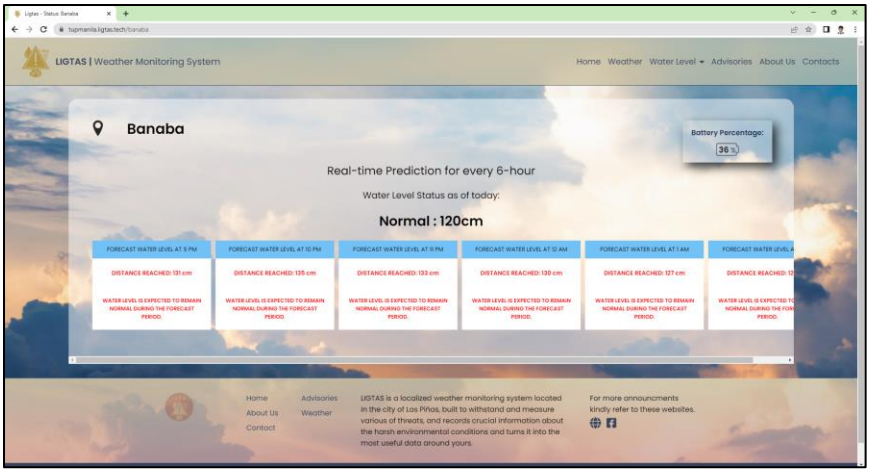


Figure 51. LIGTAS Water Level - Banaba Page

In figure 51, users can effortlessly stay updated on the latest flood conditions in Banaba Street, Phase 7B, Las Piñas City. Since the LIGTAS web application provides real-time water level data and corresponding status indicators, such as low, moderate, or high along with the predicted water level changes over the next six hours.

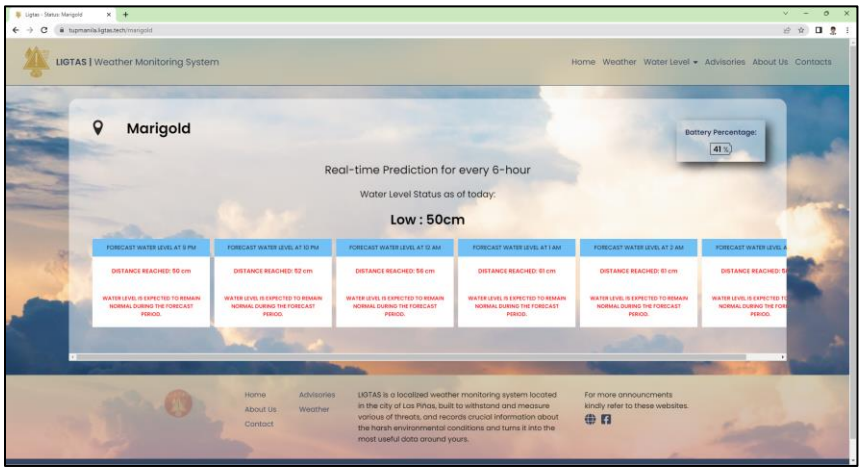


Figure 52. LIGTAS Water Level - Marigold Page

In figure 52, users can effortlessly stay updated on the latest flood conditions in Marigold Bridge, Barangay. Pulang Lupa Dos, Las Piñas City. Since the LIGTAS web application provides real-time water level data and corresponding status indicators, such

as low, moderate, or high along with the predicted water level changes over the next six hours.

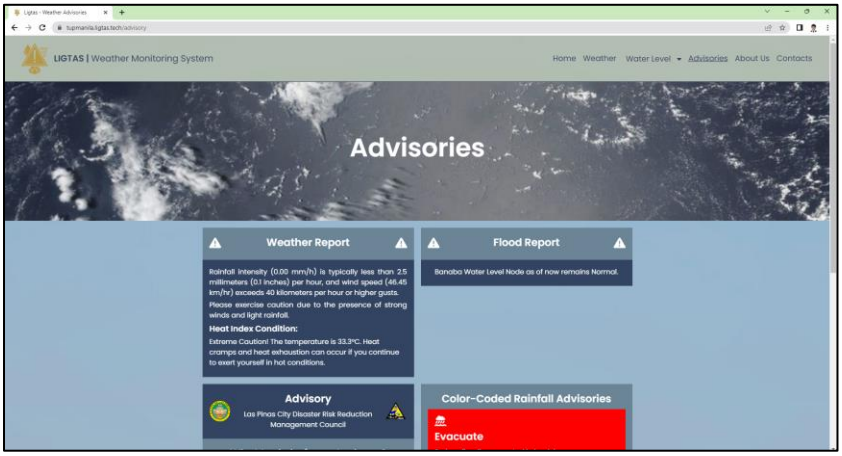


Figure 53. LIGTAS Advisory Page

In Figure 53, to ensure user safety and preparedness, the LIGTAS Advisory Page offers a consolidated view of weather and flood information, along with the recommended actions to take based on the weather and flood conditions. Additionally, color-coded rainfall advisories help users to easily understand the intensity of the rain.

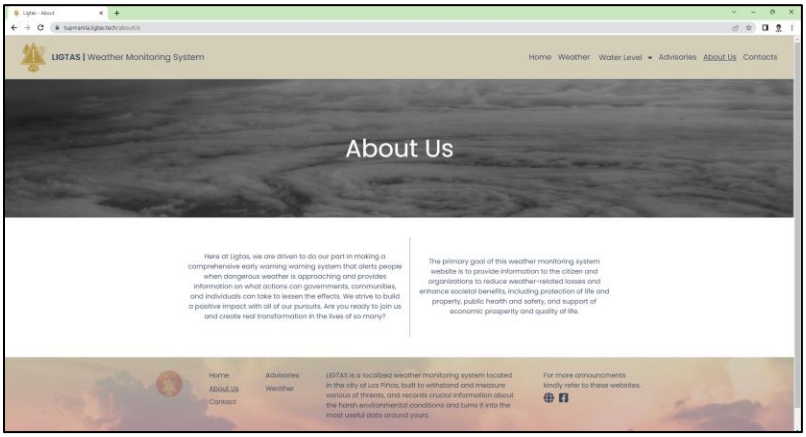


Figure 54. LIGTAS About Us Page

In Figure 54, there is a brief introduction about LIGTAS along with the goal of this system which is to provide weather and flood updates to the citizens and authorities.

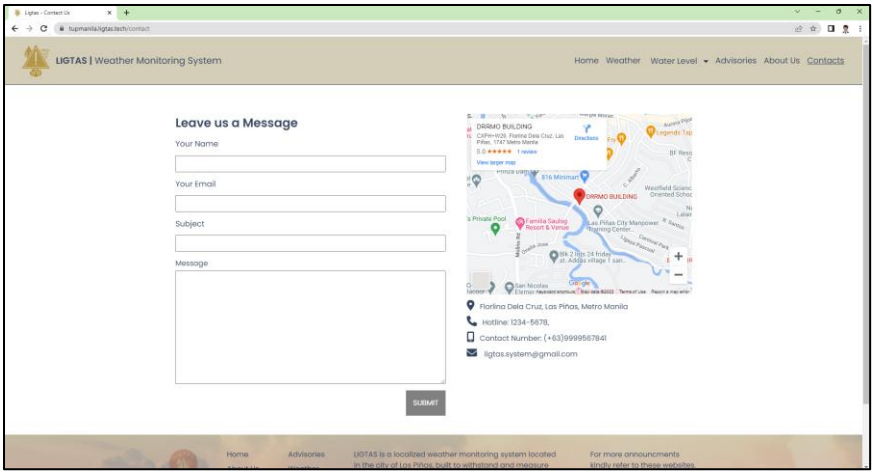


Figure 55. LIGTAS Contact Page

In Figure 55, a user-friendly contact form is provided to users, allowing them to directly communicate with the application's administrators and conveniently submit their queries or messages. Additionally, the LIGTAS Contact Page includes an email and contact number for further assistance. Lastly, a map displaying the location of the weather monitoring system is also included.

4.4 Predictive Model Evaluation on Training and Field Data

To validate the accuracy, effectiveness, and reliability of the system to the officials of Las Pinas Disaster Risk Reduction Management Council (LPDRRMC), barangay personnel and residents of Barangay Pulang Lupa Dos, Barangay Talon Dos, and Barangay Manuyo Dos in Las Pinas, Metro Manila, the proponents proceed to the training of predictive model using neural networks. The training of the predictive model was done using deep LSTM neural networks. The training was set to 40 epochs and stopped at the 12th epoch, with the 7th epoch having the best result, due to early stopping applied in the callbacks. A learning rate of 0.001 was set to reduce the loss in a gradual manner. Stated below are the values of regression metrics in the 7th epoch for the trained predictive model.

Table 21. Training Performance of Predictive Model using LSTM Neural Networks

	Training Set Performance	**Validation Set Performance**	**Test Set Performance**
Mean Squared Error (MSE)	0.0208	0.0177	0.0210
Root Mean Squared Error (RMSE)	0.1000	0.0825	0.0996
Mean absolute error (MAE)	0.0335	0.0325	0.0352
Accuracy	91.1%	92.81%	92.68%

Table 13 shows that the predictive model trained with LSTM Neural Networks yields good results. The regression metrics used, such as Mean absolute error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE), indicate the predictive model's accuracy is exceptionally high when the values are lower.

As can be seen in the table, the MAE, MSE, and RMSE values are all low and close to zero, signifying the accuracy of the model. Moreover, the values of MAE, MSE and RMSE in the training set are always greater than their values in the validation set which indicates that the model did not overfit. Furthermore, in terms of accuracy itself, the trained model has an accuracy of 91.1% in the test set, 92.81% in the validation set, and 92.68% in the test set, which backups the high accuracy shown by the low values of MSE, RMSE, and MAE.

To test the performance of the trained model in further unknown data, field data was collected when all the nodes of the system were working. After which, the data collected are fed into the trained predictive model to test its performance to unknown data.

Table 22. Field Test Performance of Predictive Model using LSTM Neural Networks

	Field Test Performance
Mean Squared Error (MSE)	0.4558
Root Mean Squared Error (RMSE)	0.6008
Mean Absolute Error (MAE)	0.4170
Accuracy	92.53%

As can be seen from the results of field testing, the predictive LSTM model has a MSE of 0.4558, RMSE of 0.6008, and MAE of 0.4170 in the actual field data. The regression metric values are still low and near zero which shows that the predictive model still performed well in the field data. In terms of accuracy, the predictive LSTM model performed 92.53% accuracy in the field which again backs up the high accuracy shown by the MSE, RMSE, and MAE regression metrics.

4.5 Data Results and Interpretation

Prediction results for the weather and flood nodes are presented in tables and figures. All are true values after post processing was performed right after the prediction of data. For the figures, the green line represents the actual weather and flood data while the blue line represents the predicted weather and flood data.

4.5.1 Flood Monitoring System Data Results

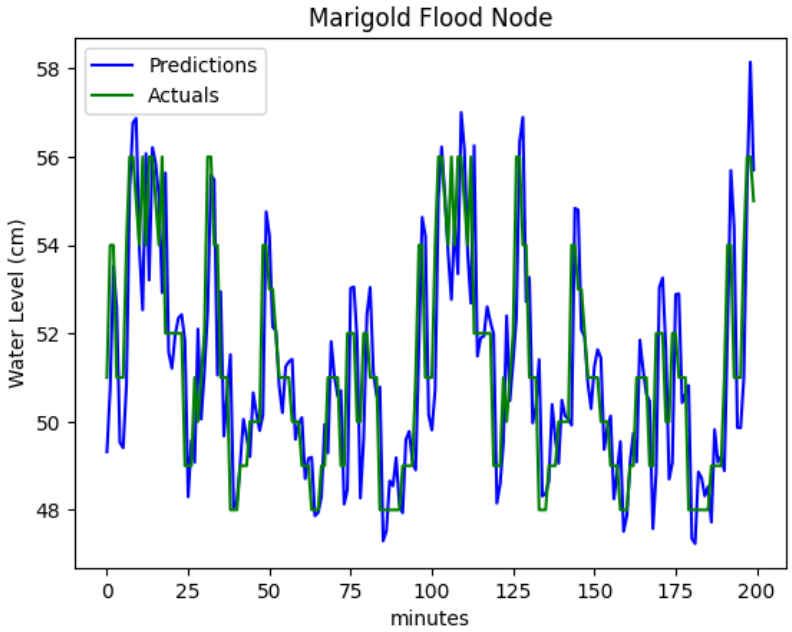


Figure 56. Predicted and Actual Water level Data at Marigold Flood Node

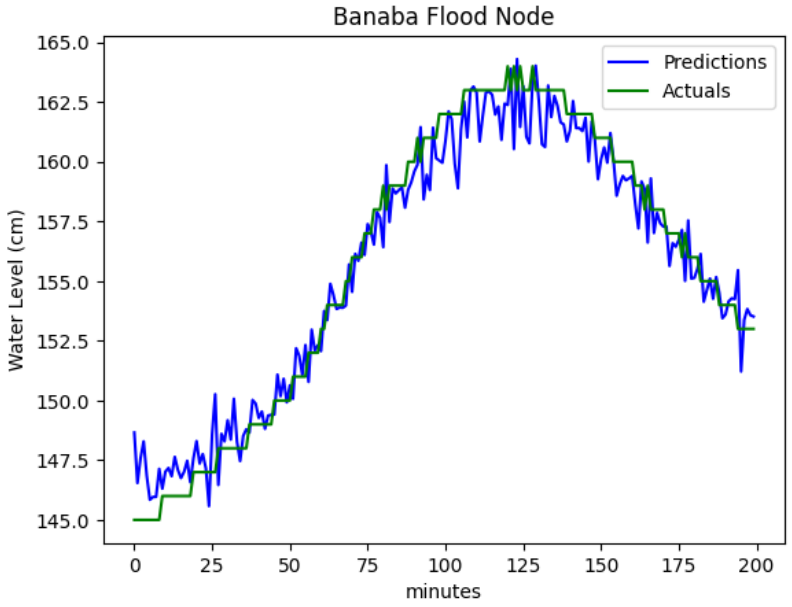


Figure 57. Predicted and Actual Water level Data at Banaba Flood Node

Table 23. Predicted and Actual Water Level Data at Banaba and Marigold

index	Marigold Water Level (cm) Predictions	Marigold Water Level (cm) Actuals	Banaba Water Level (cm) Predictions	Banaba Water (cm) Level Actuals
0	49.316	51	148.6581	145
1	50.7422	54	146.5442	145
2	53.5203	54	147.5795	145
3	52.5902	51	148.2887	145
4	49.5225	51	146.8465	145
5	49.4069	51	145.8419	145
6	50.826	54	145.9598	145
7	55.2738	56	145.9653	145
8	56.7728	56	147.137	145
9	56.8737	55	146.3075	146
10	53.897	54	147.0253	146
11	52.5324	56	147.1748	146
12	56.0772	54	146.8291	146
13	53.2075	56	147.6406	146
14	56.213	56	147.0948	146
15	55.8485	55	146.7517	146
16	55.0941	54	147.0168	146
17	52.9232	56	147.4748	146
18	55.6368	52	146.5768	146
19	51.5709	52	147.5722	147
20	51.208	52	148.3003	147

4.5.2 Weather Monitoring System Data Results

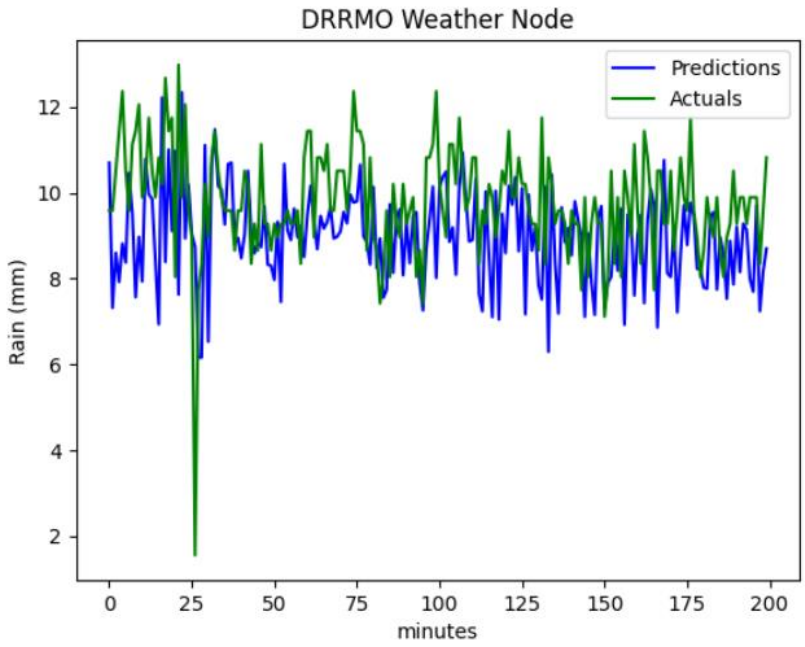


Figure 58. Predicted and Actual Rain Data at DRRMO Building

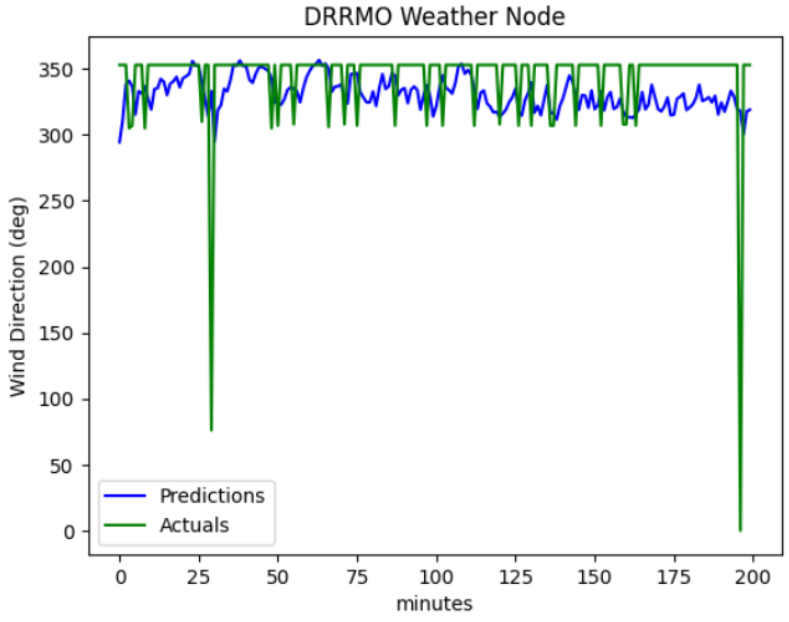


Figure 59. Predicted and Actual Wind Direction Data at DRRMO Building

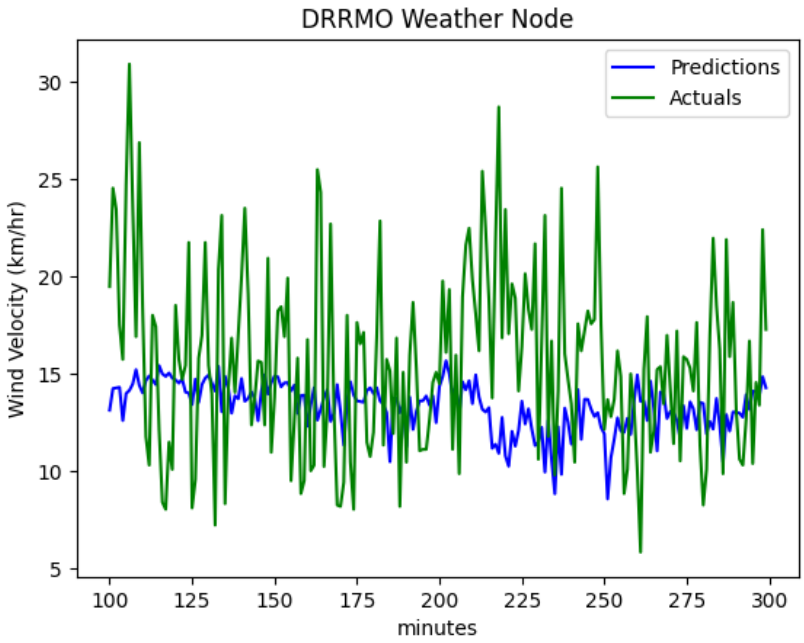


Figure 60. Predicted and Actual Wind Speed Data at DRRMO Building

Table 24. Predictions and Actual Data of Rain, Wind Direction, and Wind Speed at the DRRMO Building

	Rain (mm) Predictions	Rain (mm) Actuals	Wind Direction (deg) Predictions	Wind Direction (deg) Actuals	Wind Speed (km/hr) Predictions	Wind Speed (km/hr) Actuals
0	12.9535	9.5818	268.734	353	13.3616	23.97
1	5.786	9.5818	311.0865	353	14.9002	26.02
2	8.2729	10.5091	298.8483	353	11.642	21.55
3	5.8241	11.4364	294.3339	305	8.5447	26.97
4	6.755	12.3636	304.9753	307	8.9886	18.03
5	5.6468	10.5091	309.0761	353	11.4327	25.51
6	7.9524	9.5818	292.5161	353	12.0926	23.75
7	6.6064	11.1273	308.5473	353	13.374	15.9

8	6.2678	11.4364	290.1719	305	10.7585	18.18
9	7.6822	12.0545	276.0074	353	9.4901	14.73
10	6.4695	9.8909	297.9141	353	12.9059	25.95
11	8.4614	10.2	295.9963	353	13.0186	23.68
12	7.1449	11.7455	309.2271	353	14.0676	21.84
13	8.1083	10.5091	296.2361	353	12.1228	20.16
14	7.2797	9.8909	295.1552	353	11.8716	15.54
15	6.3882	10.8182	304.3644	353	13.4664	26.83
16	10.0476	10.2	284.0237	353	11.6399	24.56
17	6.9299	12.6727	317.5327	353	14.7027	25.43
18	10.031	11.4364	291.2941	353	11.7815	17.96
19	7.8634	11.7455	296.8306	353	12.5103	24.92
20	9.9925	8.0364	277.361	353	10.219	16.86

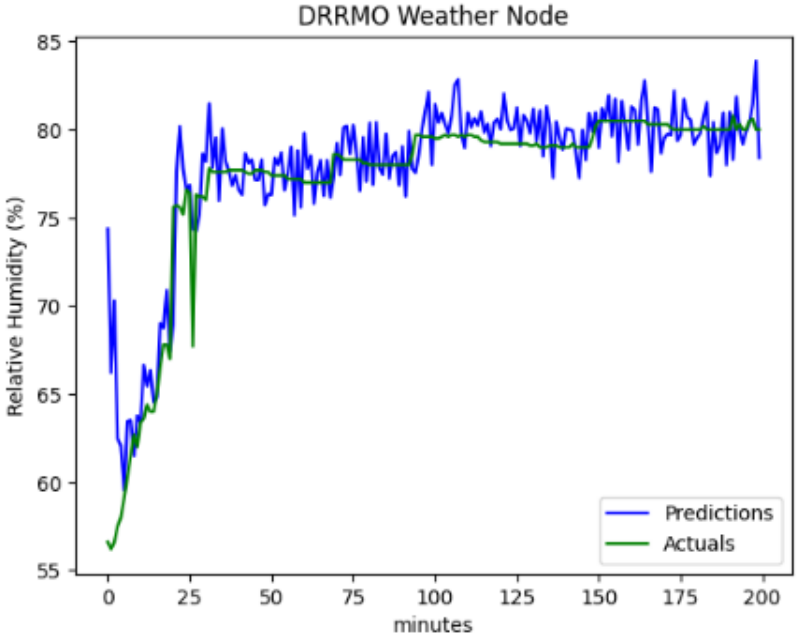


Figure 61. Predicted and Actual Relative Humidity Data at DRRMO Building

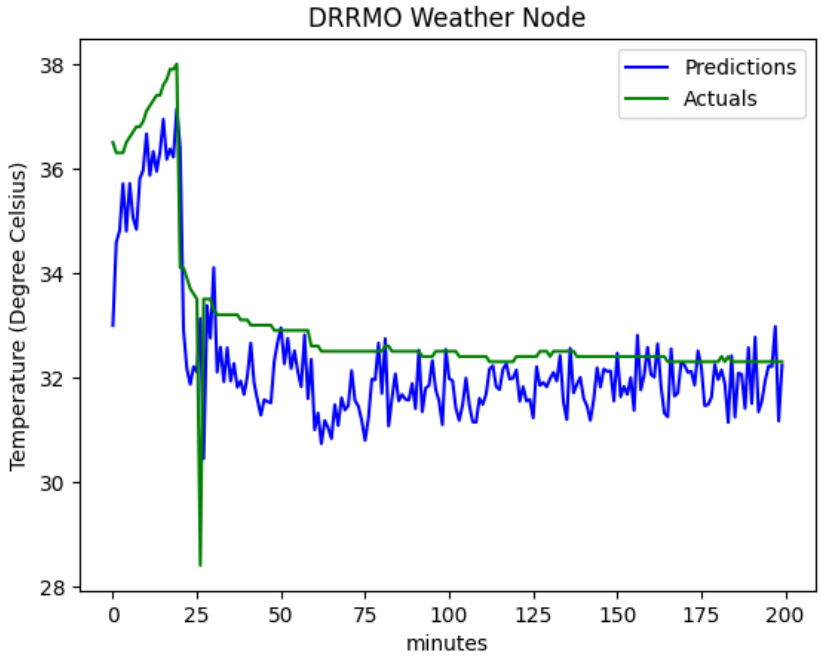


Figure 62. Predicted and Actual Temperature Data at DRRMO Building

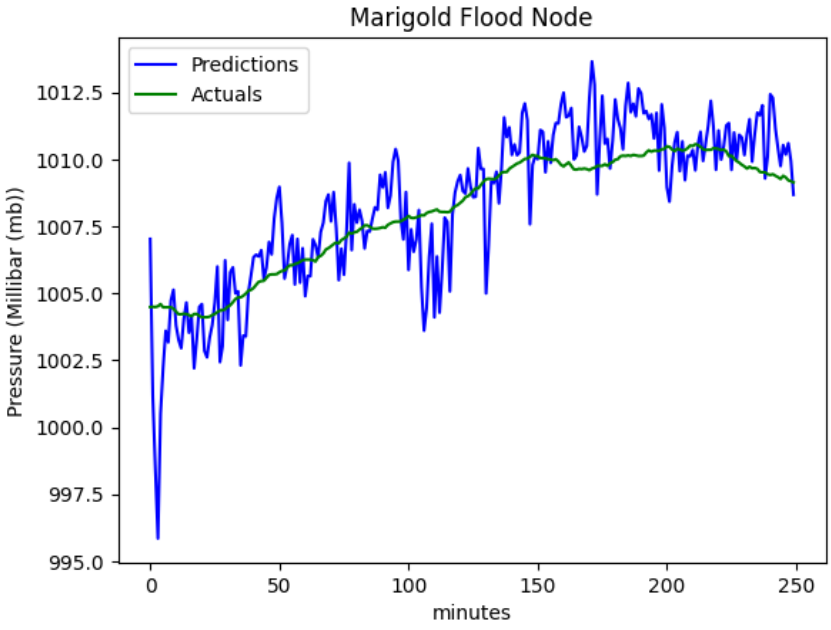


Figure 63. Predicted and Actual Atmospheric Pressure Data at DRRMO Building

Table 25. Predictions and Actual Data of Temperature, Atmospheric Pressure, and Relative Humidity at the DRRMO Building

index	Temperature (degC) Predictions	Temperature (degC) Actuals	Atmospheric Pressure (mbar) Predictions	Atmospheric Pressure (mbar) Actuals	Relative Humidity (%) Predictions	Relative Humidity (%) Actuals
0	32.9928	36.5	1007.0314	1004.49	74.3702	56.6
1	34.5861	36.3	1001.1494	1004.5	66.2041	56.2
2	34.808	36.3	998.4098	1004.49	70.3019	56.6
3	35.7078	36.3	995.8495	1004.52	62.4779	57.5
4	34.8006	36.5	1000.4963	1004.6	62.0625	58
5	35.7124	36.6	1002.2284	1004.48	59.5468	59.1
6	35.056	36.7	1003.6012	1004.48	63.4551	60.3
7	34.8335	36.8	1003.1746	1004.48	63.5152	61.5
8	35.8031	36.8	1004.7279	1004.48	61.4861	62.7
9	35.9669	36.9	1005.1328	1004.43	63.7821	62
10	36.6627	37.1	1003.8034	1004.32	63.3916	63.4
11	35.8675	37.2	1003.2609	1004.22	66.6541	63.6
12	36.3266	37.3	1002.952	1004.22	65.4378	64.4
13	35.9432	37.4	1004.0339	1004.24	66.3556	64
14	36.3156	37.4	1004.6575	1004.23	64.5445	64
15	36.9424	37.6	1003.5344	1004.19	64.7795	65
16	36.1742	37.7	1004.1501	1004.15	69.0134	66.5
17	36.3726	37.9	1002.204	1004.23	68.7386	67.8
18	36.2187	37.9	1003.2797	1004.23	70.8915	67.8

19	37.1417	38	1004.4931	1004.16	67.4342	67
20	36.4078	34.1	1004.5989	1004.11	68.9446	75.6

From the tables and figures presented, the predicted data fits well to the actual data of the weather and flood nodes. The difference between the actual and predicted data is very low, which proves the low values of MSE, RMSE, and MAE. These table and visualization results show that the trained predictive LSTM model performs well in localized forecasting of weather and flood data.

4.6 Project Evaluation

Figure 64. Website Demonstration and Evaluation

Fifty (50) residents of Barangay Manuyo Dos, Pulang Lupa Dos, and Talon Dos participated in a comprehensive survey that the researchers conducted to gather important information on public perceptions and attitudes toward the system and web application. The purpose of the survey, which included a diverse sample of respondents from various age groups and backgrounds, was to find out how much people accepted the system.

4.6.1 Functionality and Efficiency Acceptance Results

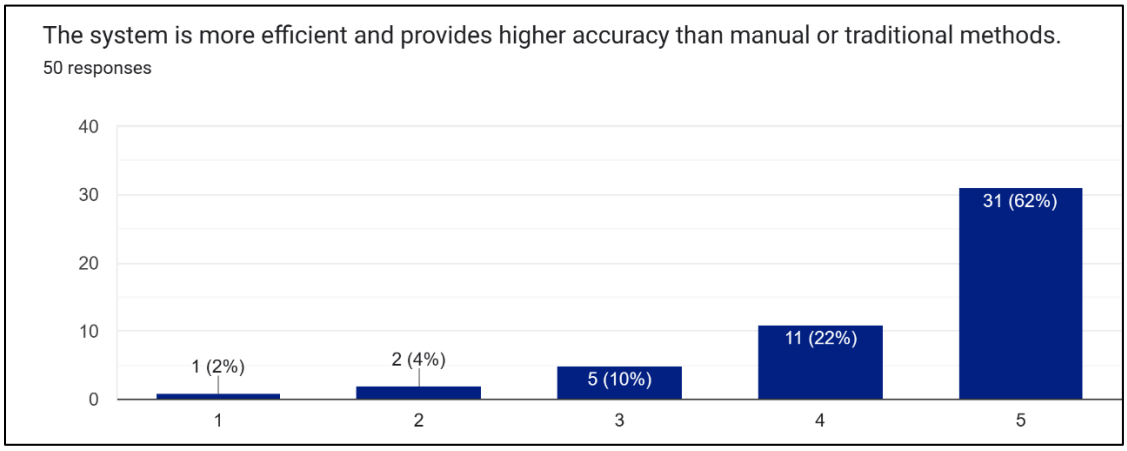


Figure 65. Summary of Answers in Question One of Functionality and Efficiency

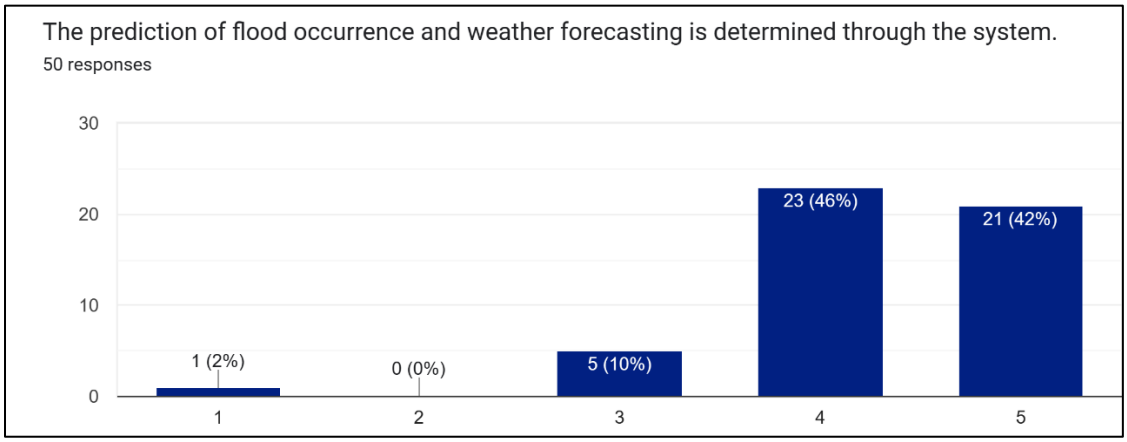


Figure 66. Summary of Answers in Question Two of Functionality and Efficiency

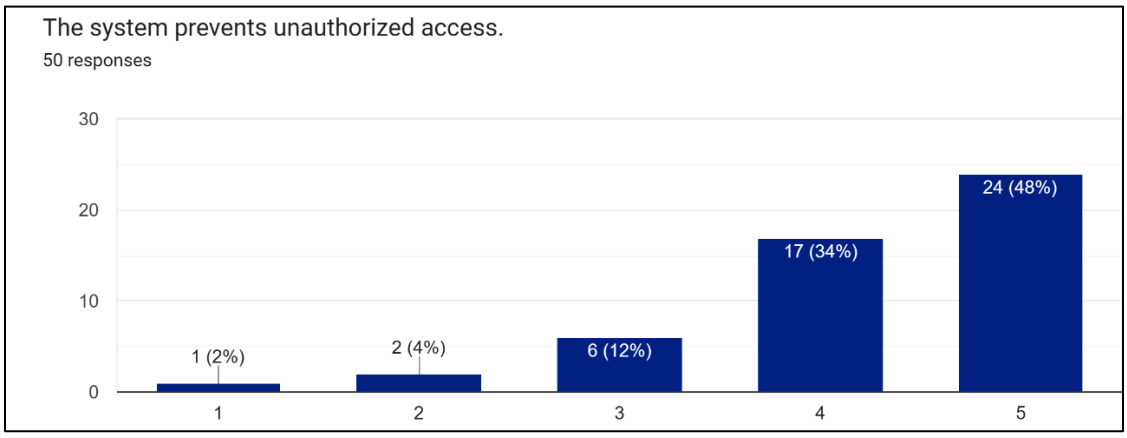


Figure 67. Summary of Answers in Question Three of Functionality and Efficiency

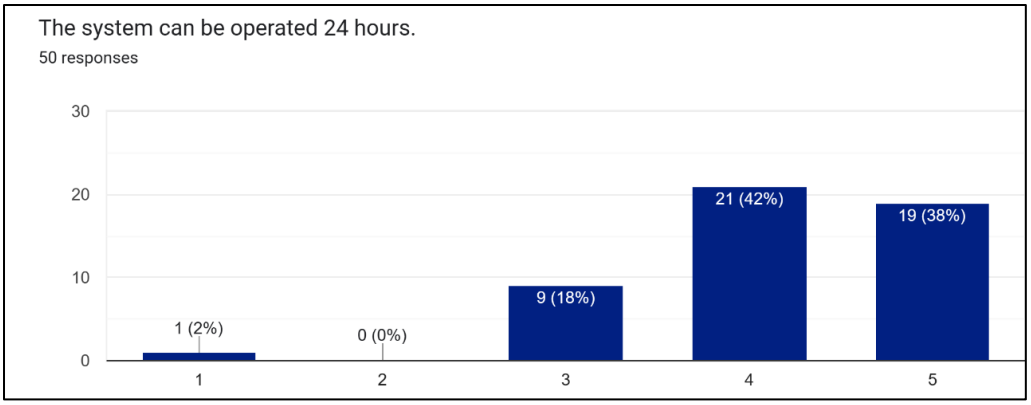


Figure 68. Summary of Answers in Question Four of Functionality and Efficiency

Based on the Likert-scale data collected from the survey conducted on user acceptance of the project in terms of functionality and efficiency, majority of respondents (62%) expressed a high level of satisfaction with the project's accuracy compared to manual or traditional methods. 42% of users strongly agreed and 46% agreed that the system can determine the flood occurrence and weather forecast. Additionally, 48% of the users strongly agreed that the system prevents unauthorized access. Furthermore, 38% of users strongly agreed and 46% agreed that the system can operate within 24 hours. There are negative responses, but most responses have a high percentage of positive which underscores the successful alignment of the project with users' expectations, suggesting that it effectively addressed their functional needs.

4.6.2 Usability Acceptance Results

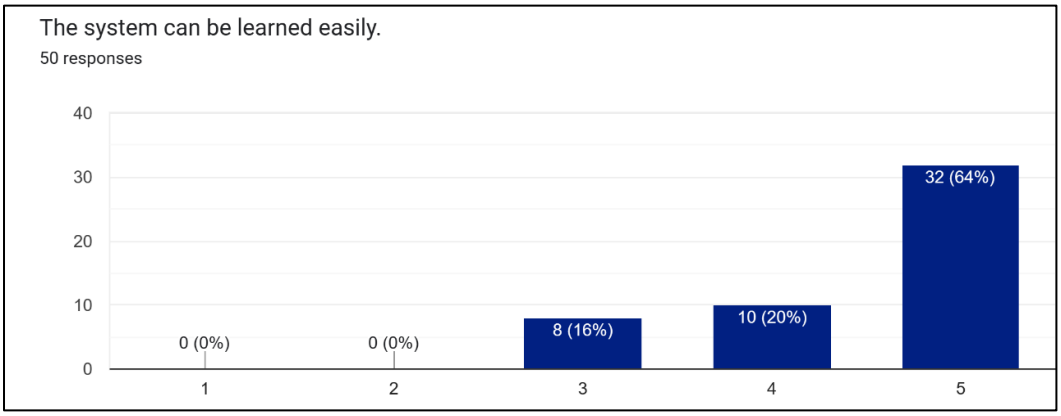


Figure 69. Summary of Answers in Question One of Usability

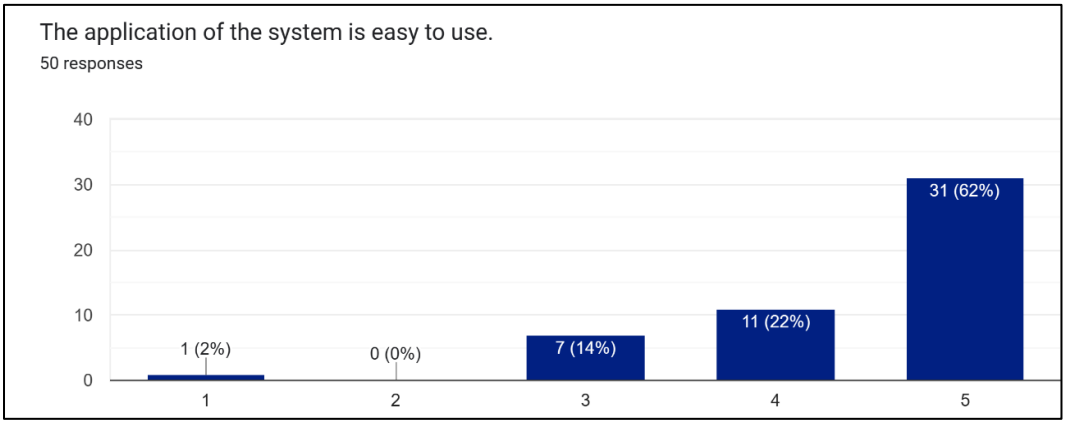


Figure 70. Summary of Answers in Question Two of Usability

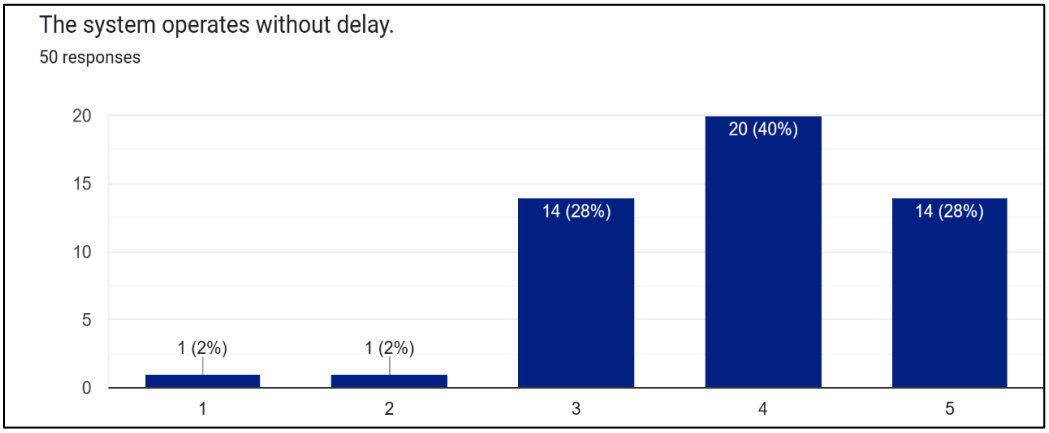


Figure 71. Summary of Answers in Question Three of Usability

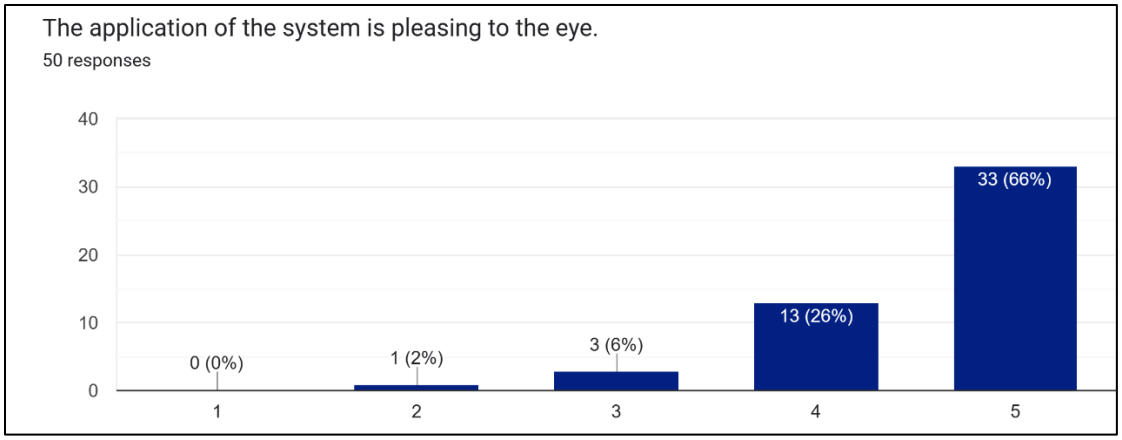


Figure 72. Summary of Answers in Question Four of Usability

Based on the Likert-scale data collected from the survey conducted on user acceptance of the project in terms of usability, the survey results indicated that majority of respondents (64%) reported that they were able to quickly learn how to use the project, suggesting that it had a short learning curve. Additionally, 62% of the users found the web application easy to use. This indicates that users found it effortless to navigate and interact with the project's features. The data also revealed that 40% of users did not strongly affirm the project's responsiveness and the system can operate without delay. Moreover, 66% of respondents expressed satisfaction with the project's design, indicating its ease of navigation and intuitive design. The high percentage of positive responses underscores the successful design and implementation of the project's user interface, suggesting that it effectively facilitated user interactions, resulting in a positive perception of usability and overall user satisfaction.

4.6.3 Maintainability and Reliability Acceptance Results

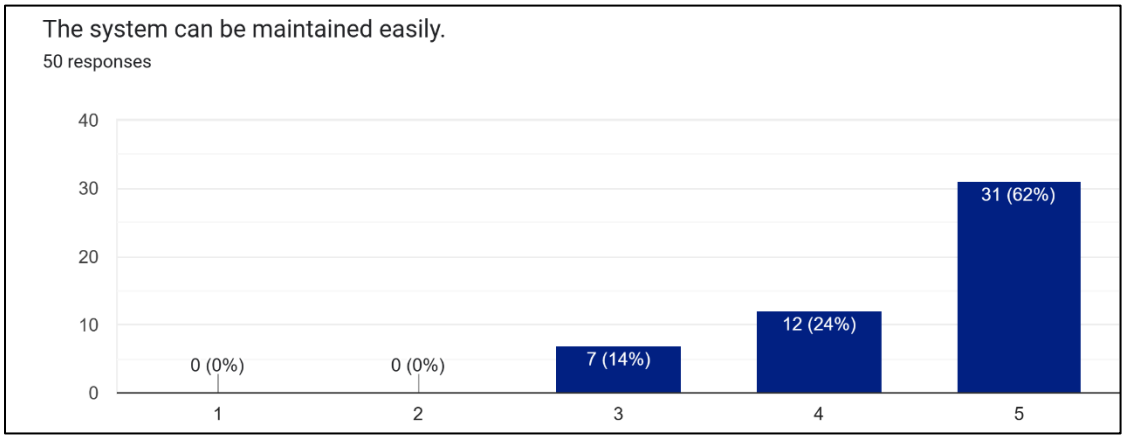


Figure 73. Summary of Answers in Question One of Maintainability and Reliability

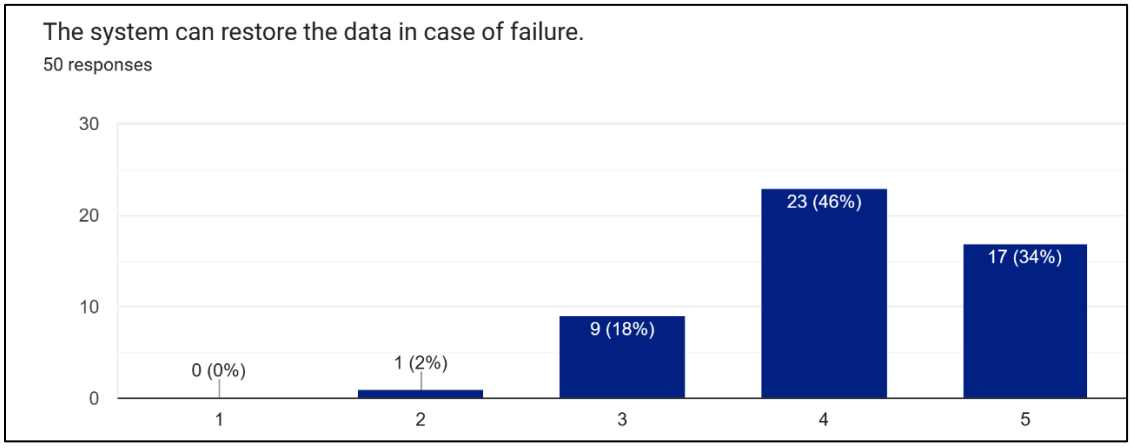


Figure 74. Summary of Answers in Question Two of Maintainability and Reliability

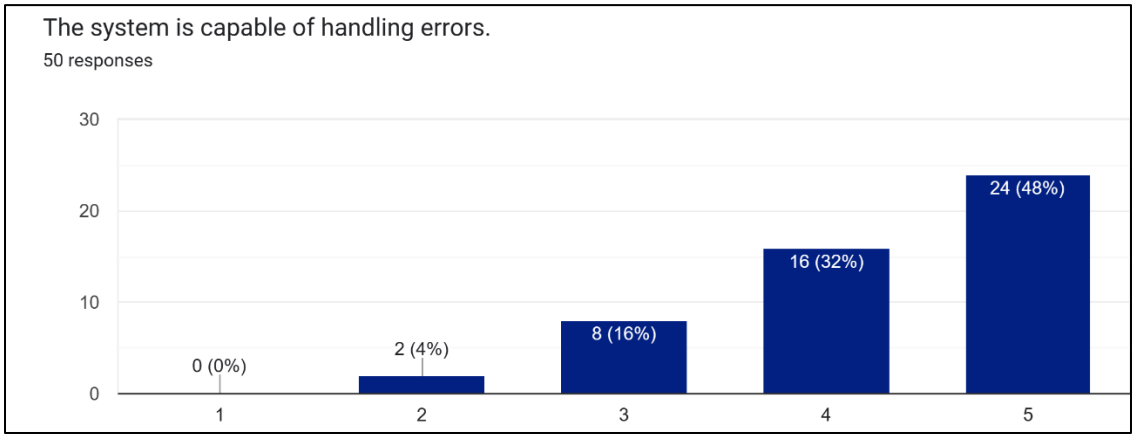


Figure 75. Summary of Answers in Question Three of Maintainability and Reliability

Based on the Likert-scale data collected from the survey conducted on user acceptance of the project in terms of maintainability and reliability showed that majority of respondents (62%) perceived the project to be highly maintainable, highlighting its ability to be easily updated, modified, and repaired. However, 34% of users reported that the system can restore the data in case of failure. Furthermore, 48% of users expressed satisfaction with the project's ability to recover from potential errors or crashes, indicating its robustness and resilience. The positive responses emphasize the successful implementation of effective maintenance practices and the provision of a reliable system, contributing to user confidence, trust, and overall satisfaction with the project.

4.6.4 Safety and Portability Acceptance Result

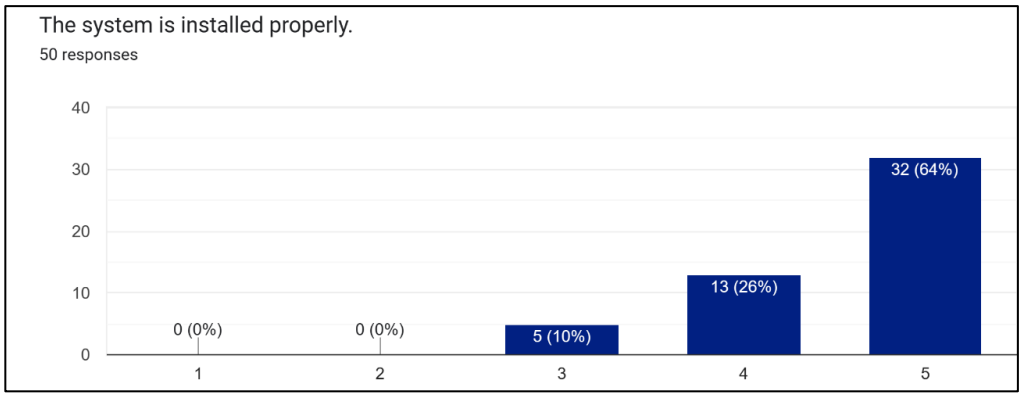


Figure 76. Summary of Answers in Question One of Safety and Portability

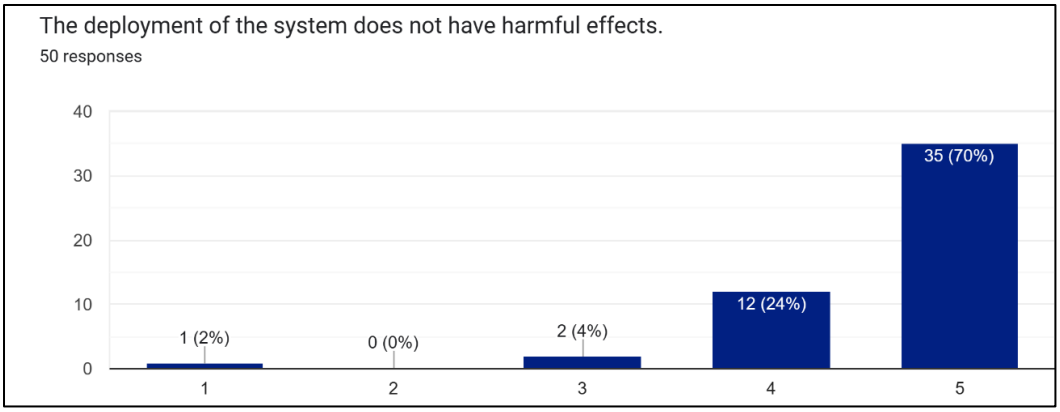


Figure 77. Summary of Answers in Question Two of Safety and Portability

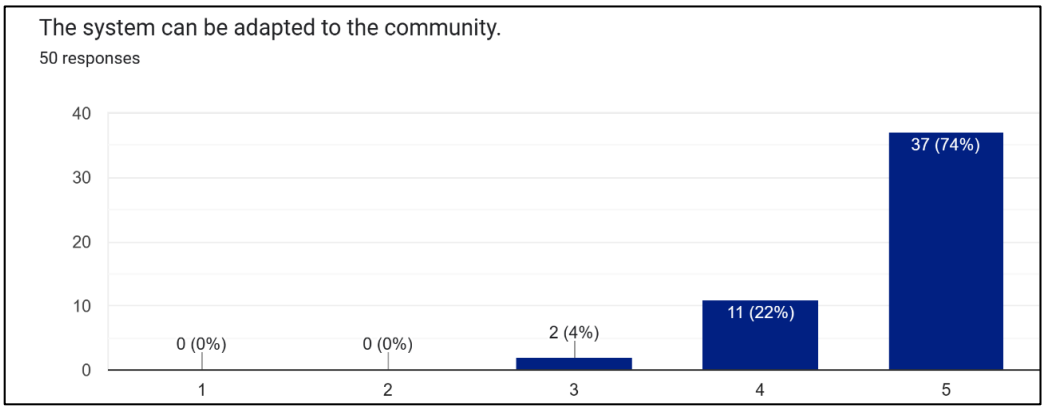


Figure 78. Summary of Answers in Question Three of Safety and Portability

Based on the Likert-scale data collected from the survey conducted on user acceptance of the project in terms of safety and portability showed that 64% of the users expressed satisfaction with the system's installation to the community. Moreover, 70% of users perceived the project to be safe for use, highlighting their confidence that the system does not have harmful effects. Additionally, most users (74%) believe the system can be adapted to the community, indicating that citizens considered the project to be portable and easy to use across a variety of devices. The positive responses highlight the successful implementation of stringent security measures and the provision of a portable solution, contributing to user trust, and overall satisfaction with the project.

Chapter 5

SUMMARY OF FINDINGS, CONCLUSION, AND RECOMMENDATIONS

5.1 Summary of Findings

The findings of this study showed that the predictive LSTM Model performed well both during training and field testing. The LSTM model's performance was evaluated using accuracy and regression metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean absolute error (MAE). During training, the LSTM model achieved an MSE of 0.0208, an RMSE of 0.1, an MAE of 0.0335, and an accuracy of 91.1%. In the field testing, the LSTM model has an MSE of 0.4558, an RMSE of 0.6008, an MAE of 0.4170, and an accuracy of 92.53%. The low values of MSE, RMSE, and MSE, coupled with the high accuracy, suggest that the LSTM performs well in both training and field testing. In addition, tabulation and visualization of actual and predicted weather and flood data were provided to further validate the accuracy and the results of the regression metrics. The tables and graphs indicate that the model fits well with the data, as the difference between the actual and predicted data is minimal. This agreement further corroborates the low values of MSE, RMSE, and MAE. These results substantiate the effectiveness of the LSTM model in providing weather and flood forecasts.

The user acceptance was analyzed from the summary of the Likert-scale data for the LIGTAS system, which it yielded valuable insights into the perceptions of users across various categories, functionality, efficiency, usability, maintainability, reliability, safety, and portability. According to the survey results, 47.5% of the users believe in the system's functionality and efficiency, indicating that the system does not meet most of the users' expectations in terms of functionality. The data also revealed that 55% of the users accepted

the system's usability indicating that the users found the web application easy to use and they are satisfied with the user interface. Moreover, 48% of users accepted the project in terms of maintainability and reliability, indicating that the system can be maintained and can handle errors. Lastly, 69.3% of the users believe in the system's safety and portability, indicating that the users trust the system and it can be adapted to the local community. These findings imply that users found the system to be valuable and beneficial in meeting their needs and expectations in the advent of heavy rainfall and flooding but there are some features that need improvement.

5.2 Conclusions

In conclusion, this research study has successfully developed a weather and flood monitoring system with an early warning system, named LIGTAS (Localized Integrated weather management and GUI Technology Artificial Intelligence System), designed for barangay - level implementation. Moreover, each specific objectives have been successfully achieved, as discussed in the following sections:

1. Hydrometeorological sensors were integrated through Arduino and PacketDUINO to collect data on temperature, relative humidity, atmospheric pressure, wind speed and direction, rainfall intensity, and water level. Additionally, the real – time transmission of data was facilitated by LoRaWAN technology.
2. The development of a web-application that provides real-time weather and monitoring flood forecasting notifications has been successfully accomplished. Through the integration of MQTT technology, the application was able to receive and process data in real-time, ensuring accurate and up-to-date information for users. The web application developed provides a valuable solution for individuals

and communities seeking timely weather and flood updates and proactive measures to minimize the impact of potential weather and flood disasters.

3. A weather and flood predictive model were developed using Long Short - Term Memory (LSTM), a specialized variant of Recurrent Neural Networks (RNNs). LSTM brings the advantage of effectively dealing with long data sequences and it also solves the vanishing gradient problem of traditional RNNs. Moreover, RNN itself is a category within the larger family of Artificial Neural Networks (ANNs). This hierarchy of ANNs, RNNs, and LSTM formed the basis of the predictive model developed. In connection, this developed predictive model showed good results during training and field testing which proves that it is deemed useful in providing weather and flood forecasting at the barangay level. Furthermore, the developed predictive model was deployed in the web - based application via FastAPI to provide a 6-hour weather and flood forecast with 1-hr intervals.
4. Results from the evaluation forms indicate that the LIGTAS system is recommended as an effective and reliable system for mitigating disaster risks and weather hazards. While there is room for improvement, such as addressing occasional delays in the website response, the system's overall effectiveness and user satisfaction remain high. The flaws that have been identified are duly noted and will be addressed to enhance the system's overall effectiveness and user satisfaction. Despite these limitations, the Local Government Unit of Las Piñas, involved in the evaluation process, recognizes the LIGTAS system as an effective instrument for disaster risk reduction, and weather and flood hazard management. They appreciate its future contributions to preparedness, response, and recovery

efforts. Furthermore, the system's user-friendly interface and effective communication channels have garnered praise, acknowledging its role in supporting coordinated actions and decision-making.

These results emphasized the study's relevance in addressing weather and flood management challenges in the Philippines, demonstrating its potential for widespread applicability across other flood-prone regions within the nation. The research not only enriches the current body of knowledge but also offers pragmatic solutions geared towards mitigating flood risks and bolstering flood management strategies.

5.3 Recommendations

The project was successfully implemented and finished. However, the proponents would like to make the following recommendations to further improve the project:

1. Extend the data collection period: Increase the duration of data collection to gather a comprehensive dataset over time, enabling analysis of long-term trends and patterns for more accurate predictions and informed decision-making.
2. Explore additional Neural Networks: Incorporating other neural networks in the predictive model architecture may also help in further increasing the accuracy of the predictive model.
3. Adopt cloud computing platforms: Utilizing cloud computing platforms such as Amazon Web Services (AWS) and Google Cloud Platform (GCP) may also help in providing better scalability and flexibility in deploying the trained predictive model in the web application.
4. Strengthen system security: Prioritize the security of the monitoring system by implementing robust measures such as encryption protocols, regular updates, and

strong authentication mechanisms to protect against unauthorized access and data breaches, ensuring the integrity and confidentiality of collected data.

5. Expand geographical coverage: Broaden the geographic scope of the monitoring system by strategically deploying additional monitoring stations or sensors in key locations to gain a comprehensive understanding of weather patterns and flood dynamics, enhancing prediction, early warning, and response capabilities.
6. Foster community engagement and participation: Actively involve local communities in the monitoring process through awareness campaigns, education programs, or citizen science initiatives, empowering community members to participate in data collection, reporting, and response efforts, integrating local knowledge for improved accuracy and relevance to community needs

REFERENCES

[1] W. Bank, “Harnessing Local Knowledge to Build Resilience,” 2014.

[2] J. P. Lapidez et al., “Identification of storm surge vulnerable areas in the Philippines through the simulation of Typhoon Haiyan-induced storm surge levels over historical storm tracks,” Nat. Hazards Earth Syst. Sci., vol. 15, no. 7, pp. 1473–1481, 2015, doi: 10.5194/nhess-15-1473-2015.

[3] The international bank for reconstruction and development. Natural disasters in the Middle East and North Africa: a regional overview. Washington DC: The World Bank; 2014. Available from: http://www.worldbank.org/en/region/mena/publication/ natural-disaster in the Middle East and North Africa. Accessed 7 June 2022.

[4] Commission on Audit, “BFP’s capability of safeguarding the populace from destructive fires hampered by scarce resources, and gaps in the procurement, construction of fire stations, and maintenance of fire trucks,” Bur. Fire Prot. Mod. Progr., 2018.

[5] “Local flood early warning systems in the Philippines,” Local Flood Early Warning Systems in the Philippines | The Global Initiative on Disaster Risk Management. [Online]. Available: https://www.gidrm.net/en/products/local-flood-early-warning-systems. [Accessed: 11-Jun-2022].

[6] D. B. Zoleta-Nantes, “Flood hazards in Metro Manila: Recognizing commonalities, differences and courses of action,” Soc. Sci. Diliman, vol. 1, no. 1, pp. 60–105, 2000.

[7] N. Agrawal, J. K. Saini, and A. Sharma, “Weather Forecasting : Era of Artificial Intelligence Weather Forecasting : Era of Artificial Intelligence,” no. May, 2020.

[8] L. C. Chang et al., “Building an intelligent hydroinformatics integration platform for regional flood inundation warning systems,” Water (Switzerland), vol. 11, no. 1, 2018, doi: 10.3390/w11010009.

[9] M. Bhagat, A. G. Thakare, and K. A. M. | N. S. M. | P. V. Choudhary, “IOT Based Weather Monitoring and Reporting System Project,” Int. J. Trend Sci. Res. Dev., vol. Volume-3, no. Issue-3, pp. 365–367, 2019, doi: 10.31142/ijtsrd21677.

[10] S. Sankaranarayanan, M. Prabhakar, S. Satish, P. Jain, A. Ramprasad, and A. Krishnan, “Flood prediction based on weather parameters using deep learning,” J. Water Clim. Chang., vol. 11, no. 4, pp. 1766–1783, 2020, doi: 10.2166/wcc.2019.321.

[11] E. Samikwa and E. Samikwa, “Edge Computing Flood Prediction System Using IoT Edge Computing,” 2020.

[12] Alam, F., Salam, M., Khalil, N. A., khan, O., & Khan, M. (2021). Rainfall trend analysis and weather forecast accuracy in selected parts of Khyber Pakhtunkhwa, Pakistan. SN Applied Sciences, 3(5). https://doi.org/10.1007/s42452-021-04457-z

[13] Naresh, R. K., Narwal, E., & Kumar, S. (2021). Weather Forecasting. https://www.researchgate.net/publication/349737259

[14] Wu, R.-S., Sin, Y.-Y., Wang, J.-X., Lin, Y.-W., Wu, H.-C., Sukmara, R. B., Indawati, L., & Hussain, F. (2022). Real-Time Flood Warning System Application. Water, 14(12), 1866. https://doi.org/10.3390/w14121866

[15] Salman, A. G., Kanigoro, B., & Heryadi, Y. (2016). Weather forecasting using deep learning techniques. *ICACSIS 2015 - 2015 International Conference on Advanced Computer Science and Information Systems, Proceedings*, 281–285. https://doi.org/10.1109/ICACSIS.2015.7415154

[16] Shivani, Sandhu, K. S., & Nair, A. R. (2019). *A Comparative Study of ARIMA and RNN for Short Term Wind Speed Forecasting-ARIMA (Auto Regressive Integrated Moving Average), RNN (Recurrent Neural Network), LSTM (long short-term memory) and short-term wind forecasting*.

[17] Ponnamperuma, N., & Rajapakse, L. (2021). Comparison of time series forecast models for rainfall and drought prediction. *MERCon 2021 - 7th International Multidisciplinary Moratuwa Engineering Research Conference, Proceedings*, 626–631. https://doi.org/10.1109/MERCon52712.2021.9525690

[18] Jafri, N. S., Ismail, S., Sadon, A. N., Rahman, N. A., & Shaharuddin, S. M. (2022). Support Vector Machine and Recurrent Neural Network Algorithm for Rainfall Forecasting. *Lecture Notes in Networks and Systems*, *457 LNNS*, 131–139. https://doi.org/10.1007/978-3-031-00828-3_13

[19] Pascanu, R., Mikolov, T., & Bengio, Y. (2012). *On the difficulty of training Recurrent Neural Networks*. http://arxiv.org/abs/1211.5063

[20] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[21] Srivastava, A., & Anto, S. (2022). Weather Prediction Using LSTM Neural Networks. 2022 IEEE 7th International Conference for Convergence in Technology, I2CT 2022. https://doi.org/10.1109/I2CT54291.2022.9824268

[22] Salehin, I., Talha, I. M., Hasan, M., Dip, S. T., Saifuzzaman, M., & Moon, N. N. (2020). An Artificial Intelligence Based Rainfall Prediction Using LSTM and Neural Network. Proceedings of 2020 IEEE International Women in Engineering (WIE)

Conference on Electrical and Computer Engineering, WIECON-ECE 2020, 5–8. https://doi.org/10.1109/WIECON-ECE52138.2020.9398022

[23] Akhila, P., Anjana, R. L. S., & Kavitha, M. (2022). Climate Forecasting:Long Short Term Memory Model using Global Temperature Data. Proceedings - 6th International Conference on Computing Methodologies and Communication, ICCMC 2022, 469–473. https://doi.org/10.1109/ICCMC53470.2022.9753779

[24] Sit, M., Zahit Demiray, B., Xiang, Z., & Sermet, Y. (2020). *A Comprehensive Review of Deep Learning Applications in Hydrology and Water Resources*. https://doi.org/10.31223/osf.io/xs36g

[25] Yang, S., Yu, X., & Zhou, Y. (2020). LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example. *Proceedings - 2020 International Workshop on Electronic Communication and Artificial Intelligence, IWECAI 2020*, 98–101. https://doi.org/10.1109/IWECAI50956.2020.00027

[26] Fente, D. N., & Singh, D. K. (2018). Weather Forecasting Using Artificial Neural Network

[27] Sahagun, M. A. M., dela Cruz, J. C., & Garcia, R. G. (2017). Wireless Sensor Nodes for Flood Forecasting using Artificial Neural Network

[28] K. B. Katsaros, Sensors for mean meteorology, 3rd ed., no. April 2018. Elsevier Ltd., 2019. doi: 10.1016/B978-0-12-409548-9.11210-2.

[29] K. I. Han et al., “Compliment graphene oxide coating on silk fiber surface via electrostatic force for capacitive humidity sensor applications,” Sensors (Switzerland), vol. 17, no. 2, 2017, doi: 10.3390/s17020407.

[30] J. Čimo and B. Šiška, “Design and realization of monitoring system for measuring air temperature and humidity, wind direction and speed,” J. Environ. Eng. Landsc. Manag., vol. 14, no. 3, pp. 127–134, 2006, doi: 10.1080/16486897.2006.9636889.

[31] I. Bramer et al., “Advances in Monitoring and Modelling Climate at Ecologically Relevant Scales,” Adv. Ecol. Res., vol. 58, pp. 101–161, 2018, doi: 10.1016/bs.aecr.2017.12.005.

[32] F. L. Meng, Z. Guo, and X. J. Huang, “Graphene-based hybrids for chemiresistive gas sensors,” TrAC - Trends Anal. Chem., vol. 68, pp. 37–47, 2015, doi: 10.1016/j.trac.2015.02.008.

[33] D. Jost, “What is a temperature sensor?” What is a temperature sensor?, 03-Jul-2019. [Online]. Available: https://www.fierceelectronics.com/sensors/what-a-temperature-sensor. [Accessed: 26-Jun-2022].

[34] J. Liu, “Why is the anemometer important to weather forecasting?” *Sciencing*, 22-Nov-2019. [Online]. Available: https://sciencing.com/info-8701078-anemometer-important-weather-forecasting.html. [Accessed: 26-Jun-2022].

[35] M. M. Rashid, M. R. bin M. Romlay, M. M. Ferdaus, and A. Al-Mamun, “Development of Electronic Rain Gauge System,” Int. J. Electron. Electr. Eng., vol. 3, no. 4, 2014, doi: 10.12720/ijeee.3.4.245-249.

[36] “What weather sensor should I use? meteorological sensors explained,” *Earth Sciences*, 28-Mar-2021. [Online]. Available: https://www.essearth.com/what-weather-sensor/. [Accessed: 26-Jun-2022].

[37] Z. S. M. Odli et al., “Development of portable water level sensor for flood management system,” ARPN J. Eng. Appl. Sci., vol. 11, no. 8, pp. 5352–5357, 2016.

[38] J. N. P. Santoso, T. T. F. Hartini, and A. S. Agoes, “Flood Detection System Using Ultrasonic Sensor with Fuzzy Logic Method,” Proc. 2nd Int. Semin. Sci. Appl. Technol. (ISSAT 2021), vol. 207, no. Issat, pp. 171–176, 2021, doi: 10.2991/aer.k.211106.027.

[39] S. Dswilan, Harmadi, and Marzuki, “Flood monitoring system using ultrasonic sensor SN-SR04T and SIM 900A,” J. Phys. Conf. Ser., vol. 1876, no. 1, 2021, doi: 10.1088/1742-6596/1876/1/012003.

[40] “Barometer,” *National Geographic Society*. [Online]. Available: https://education.nationalgeographic.org/resource/barometer. [Accessed: 26-Jun-2022].

[41] I. G. W. Sena, J. W. Dillak, P. Leunupun, and A. J. Santoso, “Predicting Rainfall Intensity using Naïve Bayes and Information Gain Methods (Case Study: Sleman Regency),” J. Phys. Conf. Ser., vol. 1577, no. 1, 2020, doi: 10.1088/1742-6596/1577/1/012011.

[42] W. Bureau, “Rainfall Intensity-Frequency Regime Part 3-The Middle Atlantic Region,” no. 29.

[43] Ali, A. Khairan, F. Tempola, and A. Fuad, “Application Of Naïve Bayes to Predict the Potential of Rain in Ternate City,” E3S Web Conf., vol. 328, p. 04011, 2021, doi: 10.1051/e3sconf/202132804011.

[44] M. Muthmainnah, M. Ashar, I. M. Wirawan, and T. Widiyaningtyas, “Time Series Forecast for Rainfall Intensity in Malang City with Naive Bayes Methodology,” 3rd Int. Conf. Sustain. Inf. Eng. Technol. SIET 2018 - Proc., pp. 137–141, 2018, doi: 10.1109/SIET.2018.8693171.

[45] A. Andang, N. Hiron, A. Chobir, and N. Busaeri, “Investigation of ultrasonic sensor type JSN-SRT04 performance as flood elevation detection,” IOP Conf. Ser. Mater. Sci. Eng., vol. 550, no. 1, pp. 0–7, 2019, doi: 10.1088/1757-899X/550/1/012018.

[46] R. Sulistyowati, H. A. Sujono, and A. K. Musthofa, “Design and field test equipment of river water level detection based on ultrasonic sensor and SMS gateway as flood early warning,” AIP Conf. Proc., vol. 1855, no. 2017, 2017, doi: 10.1063/1.4985517.

[47] A. K. Shukla, Y. A. Garde, and I. Jain, “Forecast of weather parameters using time series data,” Mausam, vol. 65, no. 4, pp. 509–520, 2014, doi: 10.54302/MAUSAM.V65I4.1185.

[48] M. Wiston and M. KM, “Weather Forecasting: From the Early Weather Wizards to Modern-day Weather Predictions,” J. Climatol. Weather Forecast., vol. 06, no. 02, 2018, doi: 10.4172/2332-2594.1000229.

[49] “What is Temperature?” meteoblue. [Online]. Available: https://content.meteoblue.com/en/specifications/data-sources/measurements/temperature. [Accessed: 26-Jun-2022].

[50] “Home,” *World Meteorological Organization*. [Online]. Available: https://public.wmo.int/en. [Accessed: 26-Jun-2022].

[51] C, Kummerow, W. Barnes, T. Kozu, J. Shiue, and J. Simpson, “The Tropical Rainfall Measuring Mission (TRMM) sensor package,” J. Atmos. Ocean. Technol., vol. 15, no. 3, pp. 809–817, 1998, doi: 10.1175/1520-0426(1998)015<0809:TTRMMT>2.0.CO;2.

[52] K. Rose, “Meteorological sensors,” *Lake Scientist*. [Online]. Available: https://www.lakescientist.com/meteorological-sensors/. [Accessed: 26-Jun-2022].

[53] M. Mure-Ravaud, G. Binet, M. Bracq, J. J. Perarnaud, A. Fradin, and X. Litrico, “A web based tool for operational real-time flood forecasting using data assimilation to update hydraulic states,” Environ. Model. Softw., vol. 84, pp. 35–49, 2016, doi: 10.1016/j.envsoft.2016.06.002.

[54] Y. Rahut, R. Afreen, and D. Kamini, “Smart weather monitoring and real time alert system using IoT,” Int. Res. J. Eng. Technol., vol. 848, pp. 848–854, 2008, [Online]. Available: www.irjet.net

[55] D. L. A. Flack et al., “Recommendations for improving integration in national end-to-end flood forecasting systems: An overview of the FFIR (Flooding From Intense Rainfall) programme,” Water (Switzerland), vol. 11, no. 4, 2019, doi: 10.3390/w11040725.

[56] W. Wael et. al,: “How to measure wind speed using anemometer & arduino,” How To Electronics, 03-Aug-2021. [Online]. Available: https://how2electronics.com/measure-wind-speed-using-anemometer-arduino/. [Accessed: 26-Jun-2022].

[57] “Installing your personal weather station,” Weather Underground. [Online]. Available: https://www.wunderground.com/pws/installation-guide. [Accessed: 26-Jun-2022].

[58] “Last minute engineers,” Last Minute Engineers. [Online]. Available: http://lastminuteengineers.com/. [Accessed: 26-Jun-2022].

[59] "Rain gauge," *Rain Gauge - an overview | ScienceDirect Topics*. [Online]. Available: https://www.sciencedirect.com/topics/engineering/rain-gauge. [Accessed: 26-Jun-2022].

[60] Last Minute Engineers, "Insight into how DHT11 dht22 Sensor Works & interface it with Arduino," *Last Minute Engineers*, 17-Jun-2022. [Online]. Available: https://lastminuteengineers.com/dht11-dht22-arduino-tutorial/. [Accessed: 26-Jun-2022].

[61] "Waterproof JSN-SR04T ultrasonic distance sensor," *Proto*, 26-Jun-2022. [Online]. Available: https://proto-pic.co.uk/product/waterproof-jsn-sr04t-ultrasonic-distance-sensor/. [Accessed: 26-Jun-2022].

[62] "Water level sensor float switch - makerlab electronics," *Makerlab Electronics - Bits and pieces of Hardware and Electronic Stuffs!* [Online]. Available: https://www.makerlab-electronics.com/product/water-level-sensor-float-switch/. [Accessed: 29-Oct-2022].

[63] R. Gadil, "Packetduino Guide," Google Slides, https://docs.google.com/presentation/d/19IorLCkJzk2VKTT691uraq5J18RwGUiahQ9rBn69iZ4/edit#slide=id.g23c3b5bd1a0_0_0 (accessed Jun. 30, 2023).

[64] "Pololu - Arduino Uno R3," *Pololu Robotics & Electronics*. [Online]. Available: https://www.pololu.com/product/2191. [Accessed: 26-Jun-2022].

[65] "DC-DC Buck Converter LM2596S Philippines," Makerlab Electronics, https://makerlab-electronics.com/products/dc-dc-buck-converter-lm2596s (accessed Apr. 30, 2023).

[66] “18650 batteries,” 18650BatteryStore.com, https://www.18650batterystore.com/collections/18650-batteries (accessed Apr. 30, 2023).

[67] “TP300 series - ENF.” [Online]. Available: https://www.enf.com.cn/Product/pdf/Crystalline/5b233d229e43d.pdf. [Accessed: 26-Jun-2022].

[68] “About PWM solar charge controllers,” What is a PWM Charge Controller? EcoDirect, https://www.ecodirect.com/What-is-a-PWM-Charge-Controller-s/144.htm (accessed Jun. 30, 2023).

[69] “Installing your personal weather station,” *Weather Underground*. [Online]. Available: https://www.wunderground.com/pws/installation-guide. [Accessed: 26-Jun-2022].

[70] Admin, W. Says: M. A. says: W. says: P. J. C. says: weal5300 says: J. W. says: A. says: “How to measure wind speed using anemometer & arduino,” *How To Electronics*, 03-Aug-2021. [Online]. Available: https://how2electronics.com/measure-wind-speed-using-anemometer-arduino/. [Accessed: 26-Jun-2022].

[71] “Personal weather station - siting.” [Online]. Available: https://www.weather.gov/media/epz/mesonet/CWOP-Siting.pdf. [Accessed: 26-Jun-2022].

[72] “Ultrasonic sensor mounting tips,” *APG*, 06-Oct-2011. [Online]. Available: https://www.apgsensors.com/about-us/blog/ultrasonic-sensor-mounting-tips. [Accessed: 26-Jun-2022].

[73] “How to wire a float switch,” Tameson. [Online]. Available: https://tameson.com/float-switch-wiring.html. [Accessed: 30-Oct-2022].

[74] “FASTAPI,” FastAPI, https://fastapi.tiangolo.com/ [Accessed: 30-May-2023

[75] M. Campolo, A. Soldati, and P. Andreussi, “Artificial neural network approach to flood forecasting in the River Arno,” Hydrol. Sci. J., vol. 48, no. 3, pp. 381–398, 2003, doi: 10.1623/hysj.48.3.381.45286.

[76] Flood Forecasting and Early Warning in Transboundary River Basins: A Toolkit. (2015)

Appendix A
Bill of Materials

A. Bill of Materials for Weather System

Item Description	Unit Price	Quantity	Amount
BMP180 Barometric Pressure Sensor	40.00	1	40.00
DHT22 Module	200.00	1	200.00
Anemometer	2,750.00	1	2,750.00
Rain Gauge Sensor	1,299.00	1	1,299.00
Weather System Structure	7,000.00	1	7,000.00
Circuit Box	850.00	1	850.00
DC - DC Buck converter (LM2596)	150.00	1	150.00
10 A Controller	400.00	1	400.00
Wind Vane	1,325.00	1	1,325.00
Cable Tie 10"	25.00	1	25.00
Cable Tie 6"	15.00	1	15.00
10cm F/F Wire	50.00	1	50.00
3mm Shr	15.00	1	15.00
15W Solar Panel (Raon)	800.00	1	800.00
Metal Screw 14x2 w/ Tox	7.50	12	90.00
3S 10A BMS PCM	80.00	1	80.00
Switch	55.00	1	55.00
Soldering Lead	25.00	1	25.00
Glue Stick	8.00	3	24.00
3.5mm shrinkable tube	20.00	1	20.00
Glue Stick	9.00	4	36.00
#22 Wire	10.00	1	10.00
100k Resistor	1.00	5	5.00

10k Resistor	1.00	5	5.00
BMP Sensor Male Header	15.00	1	15.00
M-F 20cm	70.00	1	70.00
M-M 20cm	70.00	1	70.00
4x6 PCB	15.00	1	15.00
PacketDUINO	free	1	free
Web Hosting and Domain Subscription	1,314.68		1,314.68
TOTAL			**16,753.68**

B. Bill of Materials for the three Flood Nodes

ITEM DESCRIPTION	UNIT PRICE	QUANTITY	AMOUNT
Waterproof Ultrasonic Sensor	390.00	3	1,170.00
Enclosure Box	400.00	3	1,200.00
Bracket Stainless plus Plate (Structure)	1,500.00	3	4,500.00
Screw for Box to Plate (with washer)	4.00	12	48.00
15W Solar Panel (Online)	600.00	3	1,800.00
5A MPPT w/ Display	529.00	3	1,587.00
2s BMS straight	50.00	3	150.00
Switch mini	35.00	3	105.00
UNO case	45.00	3	135.00
Arduino Uno	650.00	3	1,950.00
PacketPro	free	3	free
TOTAL			**12,645.00**

Appendix B

Evaluation Form

LIGTAS: Localized Weather Monitoring Management and Early Warning System for Las Piñas City via Machine Learning

User Acceptance Form

ligtas.system02@gmail.com Switch account

Not shared

* Indicates required question

Greetings! We, *4th Year Electronics Engineering students* from the *Technological University of the Philippines - Manila*, would like to ask a few minutes of your time to answer our survey questions about our research that aims to Localize weather and flood monitoring and forecasting in the City of Las Piñas. Your answers will greatly help in the evaluation of our research. Thank you very much!

LIGTAS shall not disclose the participants' personal information without their consent. It shall only use this information for data analysis and presentation. Furthermore, LIGTAS shall only retain this information for one academic year. *

These conditions shall then be strictly implemented in compliance with R.A. 10173 or the Philippines' Data Privacy Act of 2012 to protect your right to data privacy.

By checking "I agree," you have read LIGTAS Data Privacy Statement and express your consent for LIGTAS to collect, record, organize, update or modify, retrieve, consult, use, consolidate, block, erase or destruct your data as part of your information.

- ◯ I Agree
- ◯ I Disagree

Next Page 1 of 7 Clear form

Demographic Profile

Name (Optional)

Your answer

Gender *

- ◯ Male
- ◯ Female

Age *

Your answer

Address *

Your answer

Contact Number *

Your answer

Back Next Page 2 of 7 Clear form

Functionality and Efficiency

Instruction: Please rate how strongly you agree or disagree with each statement. Check the following scale:
(1) Strongly Disagree
(2) Disagree
(3) Neither Agree nor Disagree
(4) Agree
(5) Strongly Agree

The system is more efficient and provides higher accuracy than manual or traditional methods. *

1 2 3 4 5

The prediction of flood occurrence and weather forecasting is determined through the system. *

1 2 3 4 5

The system prevents unauthorized access. *

1 2 3 4 5

The system can be operated 24 hours. *

1 2 3 4 5

Back Next Page 3 of 7 Clear form

Usability

Instruction: Please rate how strongly you agree or disagree with each statement. Check the following scale:
(1) Strongly Disagree
(2) Disagree
(3) Neither Agree nor Disagree
(4) Agree
(5) Strongly Agree

The system can be learned easily. *

1 2 3 4 5

The application of the system is easy to use. *

1 2 3 4 5

The system operates without delay. *

1 2 3 4 5

The application of the system is pleasing to the eye. *

1 2 3 4 5

Back Next Page 4 of 7 Clear form

Maintainability and Reliability

Instruction: Please rate how strongly you agree or disagree with each statement. Check the following scale:
(1) Strongly Disagree
(2) Disagree
(3) Neither Agree nor Disagree
(4) Agree
(5) Strongly Agree

The system can be maintained easily. *

1 2 3 4 5

The system can restore the data in case of failure. *

1 2 3 4 5

The system is capable of handling errors. *

1 2 3 4 5

Back Next Page 5 of 7 Clear form

Safety and Portability

Instruction: Please rate how strongly you agree or disagree with each statement. Check the following scale:
(1) Strongly Disagree
(2) Disagree
(3) Neither Agree nor Disagree
(4) Agree
(5) Strongly Agree

The system is installed properly. *

1 2 3 4 5

The deployment of the system does not have harmful effects. *

1 2 3 4 5

The system can be adapted to the community. *

1 2 3 4 5

Back Next Page 6 of 7 Clear form

Timestamp	LIGTAS shall not disclose the participants' personal information without their consent	Name Optional	Gender	Age	Address	Contact Number	The system is more efficient and provides higher accuracy than manual or traditional methods.	The prediction of flood occurrence and weather forecasting is determined through the system.	The system prevents unauthorized access.	The system can be operated 24 hours.	The system can be learned easily.	The application of the system is easy to use.	The system operates without delay.	The application of the system is pleasing to the eye.	The system can be maintained easily.	The system can restore the data in case of failure.	The system is capable of handling errors.	The system is installed properly.	The deployment of the system does not have harmful effects.	The system can be adapted to the community.
6/14/2023 16:05:54	I Agree		Female	45	Las Pinas	09472267364	2	5	5	3	5	5	5	5	5	5	5	5	5	5
6/14/2023 17:16:38	I Agree		Male	15	BLK 6 LOT 1 Topdville Subv. Pulanglupa Dos Las Piñas City	09368647916	5	5	4	4	3	3	3	4	3	3	3	3	1	4
6/14/2023 17:17:09	I Agree	Nicole Malate	Female	15	PULANGLUPA DOS LAS PIÑAS CITY	....	5	4	3	4	5	4	4	5	4	3	3	4	5	5
6/14/2023 17:31:37	I Agree	Mae	Female	21	Vergonville, Pulang Lupa Dos, Las Piñas	-	4	5	5	5	5	5	4	5	5	5	5	5	5	5
6/14/2023 18:37:52	I Agree		Male	57	Phase 7-B Las Piñas	09615213222	3	3	3	3	3	3	3	3	3	3	3	3	3	3
6/14/2023 21:46:56	I Agree		Male	22	Pulang Lupa Dos, Las Pinas	9669975307	4	4	5	5	4	5	5	4	5	4	5	5	4	5
6/15/2023 0:15:29	I Agree		Male	22	Pulang Lupa Dos, Las Pinas	09087625734	5	5	5	4	5	4	5	5	5	4	5	5	4	5
6/15/2023 0:31:25	I Agree	Carlo	Male	21	Pulang Lupa Dos, Las Pinas	09560275799	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6/24/2023 0:38:21	I Agree		Male	23	Las Pinas	09166085001	4	4	4	4	4	4	4	4	4	4	4	4	4	4
6/24/2023 0:39:56	I Agree		Male	23	Las Piñas	09686812363	3	5	5	4	4	5	4	5	3	4	3	4	5	4
6/24/2023 0:41:42	I Agree		Male	23	Las Piñas	9217753133	4	3	3	5	4	4	3	5	4	4	4	5	5	5
6/24/2023 0:44:04	I Agree		Female	34	Las Piñas	09274891467	5	5	4	5	5	5	4	5	5	5	5	5	5	5

6/24/2023 0:44:54	I Agree	John Peter A. Mendoza	Male	23	Las Piñas	09289606587	1	1	1	1	3	1	1	2	3	2	2	3	3	4
6/24/2023 0:45:41	I Agree		Female	26	LAS PIÑAS	09472057673	5	4	4	3	3	4	4	5	4	4	3	3	4	5
6/24/2023 0:46:56	I Agree		Female	23	1232 Balite St. Ilaya Las Piñas City	09058893529	2	4	2	3	3	4	2	4	4	3	4	4	4	4
6/24/2023 0:47:40	I Agree		Female	28	Las Piñas	09933759140	5	4	3	4	4	3	4	5	3	4	5	4	5	5
6/24/2023 0:47:43	I Agree		Male	56	Las Piñas	09127456782	5	5	5	5	3	3	3	5	5	3	3	5	5	5
6/24/2023 0:48:51	I Agree		Female	25	Pulang Lupa Dos, Las Pinas	09278453261	5	3	4	5	4	5	3	4	5	3	4	4	5	5
6/24/2023 1:19:32	I Agree		Male	22	Las Pinas	09282430624	4	4	4	4	5	4	4	4	4	4	4	4	4	4
6/24/2023 2:18:42	I Agree	Danielle	Female	23	Las Piñas	09612529620	4	4	2	3	4	4	4	5	4	5	5	4	5	5
6/24/2023 2:58:33	I Agree		Female	26	LAS PIÑAS	9063445263	4	3	3	3	4	3	3	3	3	4	3	4	4	4
6/24/2023 3:00:02	I Agree		Female	27	LAS PIÑAS	09215841451	3	4	3	3	3	3	4	3	3	4	4	3	4	5
6/24/2023 3:01:57	I Agree		Male	27	LAS PIÑAS	9954765460	4	3	4	4	3	3	3	4	4	4	4	4	4	4
6/24/2023 8:30:19	I Agree		Male	31	Talos Dos, Las Piñas	N/A	4	4	4	5	5	5	5	4	5	4	5	5	5	5
6/24/2023 8:31:18	I Agree		Female	29	Talon Dos, Las Piñas	N/A	5	5	5	5	5	4	5	4	5	4	2	5	5	5
6/24/2023 8:32:20	I Agree		Male	23	Pulang Lupa Dos, Las Piñas	N/A	5	4	5	4	5	5	5	4	5	4	4	5	5	4
6/24/2023 8:33:08	I Agree		Male	26	Pulang Lupa Dos, Las Piñas	N/A	5	4	5	4	5	5	4	4	5	4	4	5	4	4
6/24/2023 8:34:03	I Agree		Female	56	Pulang Lupa Dos, Las Piñas	N/A	5	4	5	4	4	5	4	4	5	4	4	5	4	4
6/24/2023 8:34:52	I Agree		Male	60	Manuyo Dos, Las Piñas	N/A	4	4	4	4	5	4	4	4	4	4	5	5	5	5
6/24/2023 10:23:19	I Agree	Marbe Moreno	Female	34	Las Piñas	09213166752	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6/24/2023 10:31:53	I Agree		Male	33	Las piñas	8736-3241	5	5	5	5	5	5	5	5	5	5	5	5	5	5

6/24/2023 14:37:47	I Agree		Female	25	Las Piñas City	09165448724	5	5	4	5	5	5	5	5	5	5	5	5	5	5
6/24/2023 16:32:38	I Agree		Male	22	Las Pinas	09661559678	3	4	5	3	4	4	3	5	4	3	3	4	4	3
6/24/2023 17:21:28	I Agree		Female	36	Las Piñas City	09190018486	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6/24/2023 17:34:09	I Agree	Val	Female	26	Talon, Las Piñas	09756789215	5	4	5	4	5	5	4	5	5	4	5	4	5	5
6/24/2023 17:35:59	I Agree		Female	25	Pulang Lupa Dos, Las Piñas	09999785671	5	5	4	4	5	5	3	5	4	3	5	5	5	5
6/24/2023 17:37:58	I Agree		Male	25	Talon, Las Piñas	09725336712	5	4	5	5	5	5	3	5	5	4	4	5	5	5
6/25/2023 13:12:02	I Agree		Female	23	Pulang Lupa, Las Piñas	09874651235	5	5	4	4	5	5	3	5	4	5	4	4	5	5
6/25/2023 13:14:32	I Agree	Kyle	Male	37	Las Piñas City, NCR	9998544521	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6/25/2023 13:51:43	I Agree		Male	34	Talon Dos, Las Piñas	9448465568	5	4	4	4	5	5	3	5	5	5	4	5	5	5
6/25/2023 13:57:29	I Agree	Rhiana	Female	18	Pulang Lupa, Las Piñas	N/A	5	5	4	4	5	5	4	5	5	5	5	5	5	5
6/25/2023 14:01:40	I Agree		Male	23	Phase 8 Las Piñas City	N/A	5	4	5	4	5	5	4	5	5	4	5	5	5	5
6/30/2023 9:52:22	I Agree		Female	34	Pulang Lupa, Las Piñas	N/A	3	4	4	3	5	5	3	5	5	4	4	5	5	5
6/30/2023 9:54:09	I Agree		Male	45	Las Piñas	9653752443	5	4	4	4	5	5	4	5	5	4	4	5	5	5
6/30/2023 9:54:59	I Agree		Male	24	Talon Dos, Las Piñas	----	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6/30/2023 9:56:00	I Agree		Female	23	Talon Dos, Las Piñas	92534726352	4	4	4	4	5	5	3	5	5	3	4	5	5	5
6/30/2023 11:19:28	I Agree	Kaye	Female	24	Vergonville, Pulang Lupa Dos, Las Piñas	N/A	5	4	5	4	5	5	4	5	5	5	5	5	5	5
6/30/2023 11:20:53	I Agree	John De Leon	Male	24	Vergonville, Pulang Lupa Dos, Las Piñas	09568312036	5	5	5	5	5	5	4	5	5	5	5	5	5	5
6/30/2023 11:23:51	I Agree	Jane	Female	25	Talon Dos, Las Piñas	09242536423	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6/30/2023 11:25:15	I Agree	Shaina Cruz	Female	23	Pulang Lupa Dos, Las Piñas	09525426444	5	5	5	5	5	5	4	5	5	4	5	5	5	5

Appendix C

LIGTAS System Codes

A. Hardware Codes

Code for Weather Monitoring System

```
#include <SoftwareSerial.h>
#include <DHT.h>
#include <Wire.h>

//const byte rxPin = 5;
//const byte txPin = 6;
//SoftwareSerial Serial1 (rxPin, txPin);

//For DHT
#define DHTPIN 5         // Pin connected to DHT sensor
#define DHTTYPE DHT22     // DHT sensor type
DHT dht(DHTPIN, DHTTYPE);     // Initialize DHT sensor

//For Rain Gauge
#define SWITCHPIN 2       // Pin connected to switch
const uint32_t bucketAmount = 309091; //Real value per click:
0.3090909091 to trip tipping-bucket
volatile uint32_t RainCount = 0; // Counter variable for switch clicks
volatile uint32_t dailyRain = 0; //Total clicks for the day
unsigned long lastDebounceTime = 0; // Last time the switch was
debounced
unsigned long debounceDelay = 500; // Debounce delay time in ms

//For anemometer
// Define the pin that the analog signal is connected to
float windS;
int windPin = A0;
// Define the voltage range of the anemometer
float WSvoltageMin = 0.0; // minimum voltage
float WSvoltageMax = 2.0; // maximum voltage
// Define the wind speed range of the anemometer
float windMin = 0.0; // minimum wind speed (m/s)
float windMax = 30; // maximum wind speed (m/s)

//For windvane
// Define the pin that the analog signal is connected to
int windDirPin = A1;
// Define the voltage range of the wind direction sensor
float voltageMin = 0.0; // minimum voltage
float voltageMax = 2.3; // maximum voltage
// Define the wind direction range of the sensor in degrees
float windDirMin = 0.0; // minimum wind direction (degrees)
float windDirMax = 360.0; // maximum wind direction (degrees)
float R1 = 10000.0; // resistance of R1 (10K)
float R2 = 100000.0; // resistance of R2 (100K)

//For battery monitoring
const int voltagePin = A2; // Analog pin connected to the BMS voltage
reading
float R1batt = 100000;
float R2batt = 10000;
const float minVoltage = 9.0; // Minimum battery voltage (in volts)
const float maxVoltage = 12.6; // Maximum battery voltage (in volts)
```

```
//For parameters
uint16_t temp;
uint16_t hum;
uint16_t windSpeed;
uint16_t windDir;
uint16_t percentage;

String appeui = "0000000000000000";
String deveui = "70B3D57ED800114A";
String appkey = "5754FB3BD365F541AE75B08333875E0C";

String Buffer;
bool set_baud_acsip = true;

void setup()
{
  dht.begin();
  pinMode(SWITCHPIN, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(SWITCHPIN), onSwitchClick,
FALLING);
  Serial.begin(9600);
  Serial1.begin(115200);
  delay(5000);
  run_once_acsip();
}

void loop()
{
  static unsigned long lastTime = 0;
  unsigned long currentTime = millis();
  if (currentTime - lastTime >= 60000) {
    dhtReadings();
    anemometer();
    windVane();
    batteryPercent();
    Serial.print("Rain Reading: ");
    Serial.println(RainCount);

    char data[100];
    sprintf(data,"mac tx ucnf 15 %04X%04X%08lX%04X%04X%04X", hum, temp,
(unsigned long)RainCount, windSpeed, windDir, percentage);
    Serial1.print(data);
    Serial.println(data);
    RainCount = 0;
    lastTime = currentTime;
  }
}

void dhtReadings() {
  hum = (dht.readHumidity())*100;
  temp = (dht.readTemperature())*100;
  Serial.print("Humidity: ");
  Serial.print(hum);
  Serial.print("% Temperature: ");
  Serial.println(temp);
}
```

```
void onSwitchClick() {
  unsigned long currentMillis = millis();
  if (currentMillis - lastDebounceTime >= debounceDelay) {
    RainCount+=bucketAmount;
    lastDebounceTime = currentMillis;
  }
}

void anemometer() {
  // Read the analog signal and convert it to a voltage value
  int sensorValue = analogRead(windPin);
  float WSvoltage = sensorValue * (5.0 / 1023.0);

  // Convert the voltage value to a wind speed value
  windS = ((WSvoltage - WSvoltageMin) * (windMax - windMin) /
(WSvoltageMax - WSvoltageMin) + windMin);
  windSpeed = windS*3.6*100;

  // Print the wind speed value to the serial monitor
  Serial.print("Voltage: ");
  Serial.println(WSvoltage);
  Serial.print("Wind Speed: ");
  Serial.print(windSpeed);
  Serial.println(" m/s");
}

void windVane() {
  // Read the analog signal and convert it to a voltage value
  int sensorValue = analogRead(windDirPin);
  float vout = sensorValue * (5.0 / 1023.0);
  float voltage = vout / (R2 / (R1 + R2));

  // Convert the voltage value to a wind direction value in degrees
  windDir = ((voltage - voltageMin) * (windDirMax - windDirMin) /
(voltageMax - voltageMin) + windDirMin);

  // Print the wind direction value to the serial monitor
  Serial.print("Voltage: ");
  Serial.println(voltage);
  Serial.print("Wind Direction: ");
  Serial.print(windDir);
  Serial.println(" degrees");
}

void batteryPercent() {
  int rawValue = analogRead(voltagePin); // Read the raw ADC value

  // Calculate the battery voltage using the voltage divider formula
  float voltageReading = rawValue * (5.0 / 1023.0); // Assuming a 5V
reference voltage
  float batteryVoltage = (voltageReading * (R1batt + R2batt) /
R1batt)*10; // Adjust R1 and R2 values as needed

  // Calculate the percentage
  percentage = ((batteryVoltage - minVoltage) / (maxVoltage -
minVoltage)) * 100;
```

```
  percentage = constrain(percentage, 0, 100); // Ensure the percentage
is within 0-100 range

  Serial.print("Battery Voltage: ");
  Serial.print(batteryVoltage);
  Serial.println(" V");

  Serial.print("Battery Percentage: ");
  Serial.print(percentage);
  Serial.println("%");
}

void run_once_acsip()
{
  String set_deveui = "mac set_deveui ";
  String set_appeui = "mac set_appeui ";
  String set_appskey = "mac set_appkey ";
  String mac_join = "mac join otaa";

  Serial.println("sip reset");
  Serial1.print("sip reset");
  delay(1000);
  Buffer = set_deveui;
  Buffer += deveui;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Buffer = set_appeui;
  Buffer += appeui;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Buffer = set_appskey;
  Buffer += appkey;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Serial.print("Joining...");
  Serial1.print(mac_join);
  delay(1000);
  Serial.print("Sending initial packet");
  Serial1.print("mac tx ucnf 2 deadbeef");      // This is implemented
to test connectivity.
  delay(3000);
  Serial.println("done");
}
```

Code of Water Level Monitoring System for Banaba

```
#include <SoftwareSerial.h>

// For packetPro
const byte rxPin = 2;
const byte txPin = 3;
SoftwareSerial Serial1 (rxPin, txPin);
```

```
//For battery percentage
uint16_t percentage;
int analogInput = 0;
float vout = 0.0;
float vin = 0.0;
float R1 = 100000.0; // resistance of R1 (100K)
float R2 = 10000.0; // resistance of R2 (10K)
int value = 0;

// For Ultrasonic define Trig and Echo pin:
const int trigPin = 6;
const int echoPin = 7;

// Assigning pins for float switches
const int floatSwitch1Pin = 8;
const int floatSwitch2Pin = 9;
const int floatSwitch3Pin = 10;

// Define variables:
long duration;
int distance;
int level;

String appeui = "0000000000000000";
String deveui = "70B3D57ED800122A";
String appkey = "A3C7052A95664977C33EABA7E57EA3CB";

String Buffer;
bool set_baud_acsip = true;

void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);


  pinMode(analogInput, INPUT);

  Serial.begin(9600);
  Serial1.begin(115200);
  delay(5000);
  run_once_acsip();
}

void loop()
{
  static unsigned long lastTime = 0;
  unsigned long currentTime = millis();
  if (currentTime - lastTime >= 60000) { // Wait for 60 seconds
    Ultrasonic();
    floatSwitches();
    batteryMonitoring();
   char data[100];
    sprintf(data,"mac tx ucnf 15 %04X%04X%04X", distance, level,
percentage);
    Serial1.print(data);
    Serial.println(data);
```

```
    lastTime = currentTime;
  }
}

void Ultrasonic() {
  // Clear the trigPin by setting it LOW:
  digitalWrite(trigPin, LOW);

  delayMicroseconds(5);

 // Trigger the sensor by setting the trigPin high for 10 microseconds:
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the echoPin. pulseIn() returns the duration (length of the
pulse) in microseconds:
  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance:
  distance = duration*0.034/2;

  // Print the distance on the Serial Monitor (Ctrl+Shift+M):
  Serial.print("Distance = ");
  Serial.print(distance);
  Serial.println(" cm");
}

void floatSwitches() {
  if (distance <= 93.75){
    Serial.println("Water Level is CRITICAL");
    level = 40;
  }
  else if(distance > 93.75 && distance <= 187.5){
    Serial.println("Water Level is HIGH");
    level = 30;
  }
  else if(distance > 187.5 && distance <= 291.25){
    Serial.println("Water Level is NORMAL");
    level = 20;
  }
  else if(distance > 291.5 && distance <= 375){
    Serial.println("Water Level is LOW");
    level = 10;
  }
  else{
    Serial.println("There is an error");
    level = 99;
  }
}

void batteryMonitoring() {
  // read the value at analog input
  value = analogRead(analogInput);
  vout = (value * 5) / 1000.0;
  vin = vout / (R2 / (R1 + R2));
```

```
  // convert voltage to percentage based on 2S lithium-ion battery
voltage range
  float min_voltage = 6.0; // minimum voltage of a 2S lithium-ion
battery
  float max_voltage = 8.4; // maximum voltage of a 2S lithium-ion
battery
  percentage = ((vin - min_voltage) / (max_voltage - min_voltage)) *
100;
  percentage = (constrain(percentage, 0, 100)); // constrain percentage
to 0-100 range

  Serial.print("Voltage: ");
  Serial.print(vin);
  Serial.print("V, Percentage: ");
  Serial.print(percentage);
  Serial.println("%");
}
void run_once_acsip()
{
  String set_deveui = "mac set_deveui ";
  String set_appeui = "mac set_appeui ";
  String set_appskey = "mac set_appkey ";
  String mac_join = "mac join otaa";

  Serial.println("sip reset");
  Serial1.print("sip reset");
  delay(1000);
  Buffer = set_deveui;
  Buffer += deveui;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Buffer = set_appeui;
  Buffer += appeui;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Buffer = set_appskey;
  Buffer += appkey;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Serial.print("Joining...");
  Serial1.print(mac_join);
  delay(1000);
  Serial.print("Sending initial packet");
  Serial1.print("mac tx ucnf 2 deadbeef");      // This is implemented
to test connectivity.
  delay(3000);
  Serial.print("done");
}
```

Code of Water Level Monitoring System for Marigold Bridge

```
#include <SoftwareSerial.h>
#include <SFE_BMP180.h>
#include <Wire.h>
```

```
// For packetPro
const byte rxPin = 2;
const byte txPin = 3;
SoftwareSerial Serial1 (rxPin, txPin);

// For BMP180
SFE_BMP180 pressure;

//For battery percentage
uint16_t percentage;
int analogInput = 0;
float vout = 0.0;
float vin = 0.0;
float R1 = 100000.0; // resistance of R1 (100K)
float R2 = 10000.0; // resistance of R2 (10K)
int value = 0;

// For Ultrasonic define Trig and Echo pin:
const int trigPin = 6;
const int echoPin = 7;

// Define variables:
long duration;
int distance;
int level;
uint32_t Pressure;

String appeui = "0000000000000000";
String deveui = "70B3D57ED800125B";
String appkey = "0869BE9F6FDC8F378F17DE0180BC4114";

String Buffer;
bool set_baud_acsip = true;

void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  pinMode(analogInput, INPUT);

  Serial.begin(9600);
  Serial1.begin(115200);

  Serial.println("Initialize BMP...");
  delay(1000);

  if (pressure.begin()){
    Serial.println("BMP180 init success");}
  else {
    Serial.println("BMP180 init fail\n\n");
    while(1);}

  delay(5000);
  run_once_acsip();
}
```

```
void loop()
{
  static unsigned long lastTime = 0;
  unsigned long currentTime = millis();
  if (currentTime - lastTime >= 60000) { // Wait for 10 seconds
    Ultrasonic();
    floatSwitches();
    batteryMonitoring();
    bmpReadings();
   char data[100];
    sprintf(data,"mac tx ucnf 15 %04X%04X%04X%06lX", distance, level,
percentage, Pressure);
    Serial1.print(data);
    Serial.println(data);
    lastTime = currentTime;
  }
}

void Ultrasonic() {
  // Clear the trigPin by setting it LOW:
  digitalWrite(trigPin, LOW);

  delayMicroseconds(5);

 // Trigger the sensor by setting the trigPin high for 10 microseconds:
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the echoPin. pulseIn() returns the duration (length of the
pulse) in microseconds:
  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance:
  distance = duration*0.034/2;

  // Print the distance on the Serial Monitor (Ctrl+Shift+M):
  Serial.print("Distance = ");
  Serial.print(distance);
  Serial.println(" cm");
}

void floatSwitches() {
  if (distance <= 93.75){
    Serial.println("Water Level is CRITICAL");
    level = 40;
  }
  else if(distance > 93.75 && distance <= 187.5){
    Serial.println("Water Level is HIGH");
    level = 30;
  }
  else if(distance > 187.5 && distance <= 291.25){
    Serial.println("Water Level is NORMAL");
    level = 20;
  }
  else if(distance > 291.5 && distance <= 375){
```

```
    Serial.println("Water Level is LOW");
    level = 10;
  }
  else{
    Serial.println("There is an error");
    level = 99;
  }
}

void batteryMonitoring() {
  // read the value at analog input
  value = analogRead(analogInput);
  vout = (value * 5) / 1000.0;
  vin = vout / (R2 / (R1 + R2));

  // convert voltage to percentage based on 2S lithium-ion battery
voltage range
  float min_voltage = 6.0; // minimum voltage of a 2S lithium-ion
battery
  float max_voltage = 8.4; // maximum voltage of a 2S lithium-ion
battery
  percentage = ((vin - min_voltage) / (max_voltage - min_voltage)) *
100;
  percentage = (constrain(percentage, 0, 100)); // constrain percentage
to 0-100 range
  Serial.print("Voltage: ");
  Serial.print(vin);
  Serial.print("V, Percentage: ");
  Serial.print(percentage);
  Serial.println("%");
}

void bmpReadings() {
  char status;
  double T,P,p0,a;

  status = pressure.startTemperature();
  if (status != 0)
  {
    // Wait for the measurement to complete:
    delay(status);

    status = pressure.getTemperature(T);
    if (status != 0)
    {
      status = pressure.startPressure(3);
      if (status != 0)
      {
        // Wait for the measurement to complete:
        delay(status);

        status = pressure.getPressure(P,T);
        if (status != 0)
        {
          Serial.print("absolute pressure: ");
          Serial.print(P);
          Serial.println(" mb");
```

```
          Pressure = P*100;
          Serial.println(Pressure);
        }
      }
    }
  }
}

void run_once_acsip()
{
  String set_deveui = "mac set_deveui ";
  String set_appeui = "mac set_appeui ";
  String set_appskey = "mac set_appkey ";
  String mac_join = "mac join otaa";

  Serial.println("sip reset");
  Serial1.print("sip reset");
  delay(1000);
  Buffer = set_deveui;
  Buffer += deveui;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Buffer = set_appeui;
  Buffer += appeui;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Buffer = set_appskey;
  Buffer += appkey;
  Serial.println(Buffer);
  Serial1.print(Buffer);
  delay(500);
  Serial.print("Joining...");
  Serial1.print(mac_join);
  delay(1000);
  Serial.print("Sending initial packet");
  Serial1.print("mac tx ucnf 2 deadbeef");      // This is implemented
to test connectivity.
  delay(3000);
  Serial.print("done");
}
```

Payload Formatter Code for Weather System

```
function decodeUplink(input) {

  payload = input.bytes;
  var data = {};
  data.Humidity = (payload[0] << 8 | payload[1])/100;
  data.Temperature =  (payload[2] << 8 | payload[3])/100;
  data.RainIntensity = (payload[4] << 24 | payload[5] << 16 |
payload[6] << 8 | payload[7])/1000000;
  data.WindSpeed = (payload[8] << 8 | payload[9])/100;
  data.WindDirection = (payload[10] << 8 | payload[11]);
  data.BatteryPercent = (payload[12] << 8 | payload[13]);
```

```
  return {
    data:data
  };
}
```

Payload Formatter Code for Flood System (Banaba)

```
function decodeUplink(input) {

  payload = input.bytes;
  var data = {};
  data.WaterHeight = 375- (payload[0] << 8 | payload[1]);
  data.WaterLevel = (payload[2] << 8 | payload[3]);
  data.BatteryPercent = (payload[4] << 8 | payload[5]);

  return {
    data:data
  };
}
```

Payload Formatter Code for Flood System (Marigold Bridge)

```
function decodeUplink(input) {
  payload = input.bytes;
  var data = {};
  data.WaterHeight = 375 - (payload[0] << 8 | payload[1]);
  data.WaterLevel = (payload[2] << 8 | payload[3]);
  data.BatteryPercent = (payload[4] << 8 | payload[5]);
  data.AtmosphericPressure = (payload[6] << 16 | payload[7] << 8 |
payload[8]) / 100;

  return {
    data: data
  };
}
```

B. Training Model Codes

(Better to be used in Jupyter Notebook or Google Colab)

```
# Import the Libraries

import tensorflow as tf
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import joblib

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import *
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping,
ReduceLROnPlateau
from tensorflow.keras.losses import MeanSquaredError
```

```
from tensorflow.keras.metrics import RootMeanSquaredError
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import load_model

from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score
from sklearn.preprocessing import StandardScaler

# Load Data
df = pd.read_csv(r'dataset_path.csv')

# Data Preparation

# Describe Dataset
df.describe().transpose()
# Removal of outliers (mean imputation)
tp = df['T (degC)']
bad_tp = tp == -9999.00
tp[bad_tp] = df['T (degC)'].mean()

p = df['p (mbar)']
bad_p = p == -9999.00
p[bad_p] = df['p (mbar)'].mean()

rh = df['rh (%)']
bad_rh = rh == -9999.00
rh[bad_rh] = df['rh (%)'].mean()

rain = df['rain (mm)']
bad_rain = rain == -9999.00
rain[bad_rain] = df['rain (mm)'].mean()

# Describe again the Dataset to see if the outliers are really removed.
df.describe().transpose()

# Data Visualization of every feature
df['rain (mm)'].plot()
df['wd (deg)'].plot()
df['wv (m/s)'].plot()
df['rh (%)'].plot()
df['p (mbar)'].plot()
df['Water Level1 (cm)'].plot()
df['Water Level2 (cm)'].plot()

# Get only the required features from the dataset
temp = df['T (degC)']
p_temp_df = pd.concat([df['p (mbar)'], temp], axis=1) #
rh_p_temp_df = pd.concat([df['rh (%)'], p_temp_df], axis=1)
wv_rh_p_temp_df = pd.concat([df['wv (m/s)'], rh_p_temp_df], axis=1)
wd_wv_rh_p_temp_df = pd.concat([df['wd (deg)'], wv_rh_p_temp_df],
axis=1)
rain_wd_wv_rh_p_temp_df = pd.concat([df['rain (mm)'],
wd_wv_rh_p_temp_df], axis=1)
new_df_water1 = pd.concat([df['Water Level1 (cm)'],
rain_wd_wv_rh_p_temp_df], axis=1)
new_df = pd.concat([df['Water Level2 (cm)'], new_df_water1], axis=1)
```

```
# Supervised Learning
def df_to_X_y(df, window_size=8):
  df_as_np = df.to_numpy()
  X = []
  y = []
  for i in range(len(df_as_np) - window_size):
      row = df_as_np[i:i+window_size]
      X.append(row)
      label = df_as_np[i+window_size]
      y.append(label)
  return np.array(X), np.array(y)
# Call the function to make the input a 3d tensor and the output a 2d
tensor
X, y = df_to_X_y(new_df)

# Verify their shapes
X.shape, y.shape


# Data Discretization (split the dataset into training, validation, and
test sets)
X_train, y_train = X[:625000], y[:625000]
X_val, y_val = X[625000:800000], y[625000:800000]
X_test, y_test = X[800000:], y[800000:]
X_train.shape, y_train.shape, X_val.shape, y_val.shape, X_test.shape,
y_test.shape

# Preprocessing

# Initialize StandardScaler
scaler = StandardScaler()

# Fit and transform the X_train
X_train_scaled = scaler.fit_transform(X_train.reshape(-1,
X_train.shape[-1])).reshape(X_train.shape)

# Save the scaler
joblib.dump(scaler, 'scaler.pkl')

# Define the preprocessing function for X
def preprocess(X):
    return scaler.transform(X.reshape(-1, X.shape[-
1])).reshape(X.shape)

# Define the preprocessing function for y (same with preprocessing
function of X, but for training purposes, let's just create another
function for y.)
def preprocess_output(y):
    return scaler.transform(y.reshape(-1, y.shape[-
1])).reshape(y.shape)


# Call the preprocess in the inputs and outputs
X_train = preprocess(X_train)
```

```
X_val = preprocess(X_val)
X_test = preprocess(X_test)

y_train = preprocess_output(y_train)
y_val = preprocess_output(y_val)
y_test = preprocess_output(y_test)

# Verify their shapes
X_train.shape, y_train.shape, X_val.shape, y_val.shape, X_test.shape,
y_test.shape


# Model Training
#define the model
model = Sequential()
model.add(InputLayer(input_shape = (8,8)))
model.add(LSTM(256, return_sequences=True))
model.add(LSTM(128, return_sequences=True))
model.add(LSTM(64, return_sequences=True))
model.add(LSTM(48, return_sequences=True))
model.add(LSTM(32, return_sequences=True))
model.add(LSTM(24, return_sequences=True))
model.add(LSTM(16, return_sequences=True))
model.add(LSTM(8))
model.add(Dense(8, 'relu'))
model.add(Dropout(0.001))
model.add(Dense(8, 'linear')) # Output layer.

model.summary()

# Callbacks during training
model_save = ModelCheckpoint('./model.h5',
                             save_best_only=True,
                             save_weights_only = False,
                             monitor = 'val_loss',
                             mode = 'min',
                             verbose = 1)
early_stop = EarlyStopping(monitor = 'val_loss', min_delta = 0.001,
                           patience = 5, mode = 'min', verbose = 1,
                           restore_best_weights = True)
reduce_lr = ReduceLROnPlateau(monitor = 'val_loss', factor = 0.3,
                              patience = 2, min_delta = 0.001,
                              mode = 'min', verbose = 1)

# Compile Model
model.compile(loss=MeanSquaredError(),
optimizer=Adam(learning_rate=0.001), metrics=[RootMeanSquaredError(),
'accuracy'])


# Train the model using the fit method
model.fit(
        X_train,
        y_train,
        validation_data=(X_val, y_val),
        epochs = 40,
        verbose = 1,
```

```
        callbacks = [model_save, early_stop, reduce_lr])

# Load Model
model = load_model(r'model.h5')
model.summary()

# Test model in the test set
loss = model.evaluate(X_test, y_test)


# Postprocessing

# Load the scaler
scaler = joblib.load('scaler.pkl')

# Postprocessing function
def postprocess(X):
    original_shape = X.shape
    X = X.reshape(-1, original_shape[-1])
    X = scaler.inverse_transform(X)
    return X.reshape(original_shape)

def postprocess_output(y):
    y = scaler.inverse_transform(y)
    return y

# Calculate other regression metrics (MSE, RMSE, MAE) in the test set

# make predictions on the test dataset
predictions = model.predict(X_test)

# postprocess the predictions and the true labels
predictions = postprocess(predictions)
y_test_processed = postprocess_output(y_test)

# calculate MSE, RMSE, and MAE
mse = mean_squared_error(y_test_processed, predictions)
rmse = mean_squared_error(y_test_processed, predictions, squared=False)
mae = mean_absolute_error(y_test_processed, predictions)

# print the results
print("Mean squared error:", mse)
print("Root mean squared error:", rmse)
print("Mean absolute error:", mae)


# Visualize the actual and predicted data

# Plot function for predicted and actual data
def plot_predictions(model, X, y, start=0, end=1000):
    predictions = model.predict(X)
    y = postprocess_output(y)
    predictions = postprocess(predictions)

    df = pd.DataFrame(data={'Water2 Predictions': predictions[:, 0],
                            'Water2 Actuals': y[:, 0],
                            'Water1 Predictions': predictions[:, 1],
```

```
                            'Water1 Actuals': y[:, 1],
                            'Rain Predictions': predictions[:, 2],
                            'Rain Actuals': y[:, 2],
                            'Wind Direction Predictions':
predictions[:, 3],
                            'Wind Direction Actuals': y[:, 3],
                            'Wind Velocity Predictions': predictions[:,
4],
                            'Wind Velocity Actuals':y[:, 4],
                            'Relative Humidity Predictions':
predictions[:, 5],
                            'Relative Humidity Actuals':y[:, 5],
                            'Pressure Predictions': predictions[:, 6],
                            'Pressure Actuals':y[:, 6],
                            'Temperature Predictions': predictions[:,
7],
                            'Temperature Actuals': y[:, 7]
                            })

    df[start:end].plot(figsize=(15,10))
    plt.grid(True)
    return df[start:end]

# Call the plot function for the post processed actual and predicted
data
post_processed_df = plot_predictions(model, X_test, y_test)  #plot the
post processed data
post_processed_df   #the output is terrible, you can't visualize the 8
curves

# Visualize actual and predicted data in each feature

#plot only the water level1 using matplotlib
start, end = 0, 100
plt.plot(post_processed_df['Water1 Predictions'][start:end], color =
'b', label = 'Predictions')
plt.plot(post_processed_df['Water1 Actuals'][start:end], color = 'g',
label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Water level1 (cm)")
plt.legend()

#plot only the water level2 using matplotlib
start, end = 0, 100
plt.plot(post_processed_df['Water2 Predictions'][start:end], color =
'b', label = 'Predictions')
plt.plot(post_processed_df['Water2 Actuals'][start:end], color = 'g',
label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Water level2 (cm)")
plt.legend()

#plot only the rain using matplotlib
start, end = 0, 200
```

```
plt.plot(post_processed_df['Rain Predictions'][start:end], color = 'b',
label = 'Predictions')
plt.plot(post_processed_df['Rain Actuals'][start:end], color = 'g',
label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Rain (mm)")
plt.ylim([-1, 1])
plt.legend()

#plot only the wind direction using matplotlib
start, end = 0, 200
plt.plot(post_processed_df['Wind Direction Predictions'][start:end],
color = 'b', label = 'Predictions')
plt.plot(post_processed_df['Wind Direction Actuals'][start:end], color
= 'g', label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Wind Direction (deg)")
plt.legend()

#plot only the wind velocity using matplotlib
start, end = 0, 200
plt.plot(post_processed_df['Wind Velocity Predictions'][start:end],
color = 'b', label = 'Predictions')
plt.plot(post_processed_df['Wind Velocity Actuals'][start:end], color =
'g', label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Wind Velocity (m/s)")
plt.legend()

#plot only the relative humidity using matplotlib
start, end = 0, 100
plt.plot(post_processed_df['Relative Humidity Predictions'][start:end],
color = 'b', label = 'Predictions')
plt.plot(post_processed_df['Relative Humidity Actuals'][start:end],
color = 'g', label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Relative Humidity (%)")
plt.legend()

#plot only the temp using matplotlib
start, end = 0, 100
plt.plot(post_processed_df['Temperature Predictions'][start:end], color
= 'b', label = 'Predictions')
plt.plot(post_processed_df['Temperature Actuals'][start:end], color =
'g', label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Temperature (Degree Celsius)")
plt.legend()

#plot only the pressure using matplotlib
start, end = 0, 500
```

```
plt.plot(post_processed_df['Pressure Predictions'][start:end], color =
'b', label = 'Predictions')
plt.plot(post_processed_df['Pressure Actuals'][start:end], color = 'g',
label = 'Actuals')

plt.xlabel("minutes")
plt.ylabel("Pressure (Millibar (mb))")
plt.legend()
```

C. Website Codes

Front – End Codes

About Us Page

```
<meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
<link rel="preconnect" href="https://fonts.googleapis.com" />
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
<link
  href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;6
00&display=swap"
  rel="stylesheet"
/>
<title>Ligtas - About</title>
<link rel="stylesheet" href="css/style.css" />
<link
  rel="stylesheet"
  type="text/css"
  href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
/>
<link
  rel="apple-touch-icon"
  sizes="180x180"
  href="src/favicon/apple-touch-icon.png"
/>
<link
  rel="icon"
  type="image/png"
  sizes="32x32"
  href="src/favicon/favicon-32x32.png"
/>
<link
  rel="icon"
  type="image/png"
  sizes="16x16"
  href="src/favicon/favicon-16x16.png"
/>
<link rel="manifest" href="src/favicon/site.webmanifest" />
<meta name="msapplication-TileColor" content="#da532c" />
<meta name="theme-color" content="#ffffff" />
</head>
<body>
<header class="nav-container">
```

```
  <nav class="nav-parent">
    <div class="nav-logo">
      <a href="/">
        <img src="src/img/logo.png" />
        <span><b>LIGTAS |</b> Weather Monitoring System</span>
      </a>
    </div>
    <ul class="nav-links toggle">
      <li><a href="/">Home</a></li>
      <li><a href="weather">Weather</a></li>
      <li class="dropdown">
        <a><button>Water Level</button></a>
        <ul class="dropdown-content toggle">
          <li><a href="banaba">Banaba</a></li>
          <li><a href="marigold">Marigold</a></li>
        </ul>
      </li>
      <li><a href="advisory">Advisories</a></li>
      <li><a href="" style="text-decoration: underline">About
Us</a></li>
      <li><a href="contact">Contacts</a></li>
    </ul>
    <div class="menu-lines">
      <i class="fa-solid fa-bars"></i>
    </div>
  </nav>
</header>
<section class="about-bg">
  <div class="about-header">
    <div class="about-text">About Us</div>
  </div>
</section>
<section class="about-container">
  <div class="about-definition">
    <div class="ligtas-about">
      <p>
        Here at Ligtas, we are driven to do our part in making a
        comprehensive early warning warning system that alerts people
when
        dangerous weather is approaching and provides information on
what
        actions can governments, communities, and individuals can take
to
        lessen the effects. We strive to build a positive impact with
all of
        our pursuits. Are you ready to join us and create real
        transformation in the lives of so many?
      </p>
      <hr class="underline" />
      <p>
        The primary goal of this weather monitoring system website is
to
        provide information to the citizen and organizations to reduce
        weather-related losses and enhance societal benefits, including
        protection of life and property, public health and safety, and
        support of economic prosperity and quality of life.
      </p>
```

```
    </div>
  </div>
</section>
<footer>
  <div class="footer-parent">
    <div class="footer-logo">
      <img src="src/img/footer-logo.png" alt="" />
    </div>
    <div class="footer-links">
      <li><a href="index">Home</a></li>
      <li><a href="" style="text-decoration: underline">About
Us</a></li>
      <li><a href="contact">Contact</a></li>
    </div>
    <div class="footer-links">
      <li><a href="advisory">Advisories</a></li>
      <li><a href="weather">Weather</a></li>
    </div>
    <div class="footer-description">
      LIGTAS is a localized weather monitoring system located in the
city of
      Las Piñas, built to withstand and measure various of threats, and
      records crucial information about the harsh environmental
conditions
      and turns it into the most useful data around yours.
    </div>
    <div class="foot-another">
      <span>For more announcements kindly refer to these
websites.</span>
      <div class="foot-me">
        <li>
          <a href="https://laspinascity.gov.ph/home" target="_blank">
            <i class="fa-solid fa-globe"></i>
          </a>
        </li>
        <li>
          <a href="https://facebook.com/lpdrrmo/" target="_blank">
            <i class="fa-brands fa-square-facebook"></i>
          </a>
        </li>
      </div>
    </div>
  </div>
</footer>
<div class="copyright">© 2023 LIGTAS Weathering System.</div>
<script src="js/main.js"></script>
</body>
</html>
```

Advisory Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
```

```
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin
/>
    <link
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;4
00;600&display=swap"
      rel="stylesheet"
    />
    <title>Ligtas - Weather Advisories</title>
    <link rel="stylesheet" href="css/style.css" />
    <link
      rel="stylesheet"
      type="text/css"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
    />
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href="src/favicon/apple-touch-icon.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="32x32"
      href="src/favicon/favicon-32x32.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="16x16"
      href="src/favicon/favicon-16x16.png"
    />
    <link rel="manifest" href="src/favicon/site.webmanifest" />
    <meta name="msapplication-TileColor" content="#da532c" />
    <meta name="theme-color" content="#ffffff" />
    <script
src="https://cdn.socket.io/4.5.1/socket.io.min.js"></script>
    <script src="/socket.io/socket.io.js"></script>
    <script>
      // Connect to the server using Socket.IO
      const socket = io.connect("https://tupmanila.ligtas.tech");

      // Handle the 'weatherUpdate' event to update the weather
condition, description, heat index condition, temperature, and output
      socket.on(
        "advisoryUpdate",
        (advisoryData, waterLevelBanaba, waterLevelMarigold) => {
          // Update the weather condition and description in the
advisory section
          document.getElementById(
            "advisory-condition-description"
          ).textContent = advisoryData.advisoryConditionDescription;
          // Update the heat index condition and temperature in the
advisory section
```

```
          document.getElementById("heat-index-condition").textContent =
            advisoryData.heatIndexCondition;

          // Update the weather output in the advisory section
          document.getElementById("advisory-output").textContent =
            advisoryData.output;

          // Update the flood banaba output in the advisory section
          document.getElementById("water-level-status-
banaba").textContent =
            advisoryData.banabaLevelDescription;

          // Update the flood banaba output in the advisory section
          document.getElementById("water-level-status-
marigold").textContent =
            advisoryData.marigoldLevelDescription;
        }
      );
    </script>
  </head>
  <body class="advisory-bg">
    <header class="nav-container">
      <nav class="nav-parent">
        <div class="nav-logo">
          <a href="/">
            <img src="src/img/logo.png" />
            <span><b>LIGTAS |</b> Weather Monitoring System</span>
          </a>
        </div>
        <ul class="nav-links toggle">
          <li><a href="/">Home</a></li>
          <li><a href="weather">Weather</a></li>
          <li class="dropdown">
            <a><button>Water Level</button></a>
            <ul class="dropdown-content toggle">
              <li><a href="banaba">Banaba</a></li>
              <li><a href="marigold">Marigold</a></li>
            </ul>
          </li>
          <li><a href="" style="text-decoration:
underline">Advisories</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contacts</a></li>
        </ul>
        <div class="menu-lines">
          <i class="fa-solid fa-bars"></i>
        </div>
      </nav>
    </header>
    <section class="advisory-text-bg">
      <div class="advisory-text">
        <h2>Advisories</h2>
      </div>
    </section>
    <section class="advisory-container">
      <div class="advisory-grid-parent">
        <div class="weather-report height-fix">
```

```
  <div class="header-report" style="background: #6b8597">
    <div class="left">
      <i
        class="fa-solid fa-triangle-exclamation"
        style="font-size: 1.675em"
      ></i>
    </div>
    <div class="text">
      <h2>Weather Report</h2>
    </div>
    <div class="right">
      <i
        class="fa-solid fa-triangle-exclamation"
        style="font-size: 1.675em"
      ></i>
    </div>
  </div>
  <div class="body-report" style="background: #324461">
    <div id="advisory-section">
      <p
        style="text-align: justify"
        id="advisory-condition-description"
      ></p>
      <p style="text-align: justify" id="advisory-output"></p>
      <!-- Display the heat index condition and temperature -->
      <h3>Heat Index Condition:</h3>
      <p id="heat-index-condition"></p>
    </div>
  </div>
</div>
<div class="weather-report height-fix">
  <div class="header-report" style="background: #6b8597">
    <div class="left">
      <i
        class="fa-solid fa-triangle-exclamation"
        style="font-size: 1.675em"
      ></i>
    </div>
    <div class="text">
      <h2>Flood Report</h2>
    </div>
    <div class="right">
      <i
        class="fa-solid fa-triangle-exclamation"
        style="font-size: 1.675em"
      ></i>
    </div>
  </div>
  <div class="body-report" style="background: #324461">
    <div>
      <p id="water-level-status-banaba"></p>
      <p id="water-level-status-marigold"></p>
    </div>
  </div>
</div>
<div class="weather-report height-fix">
  <div class="header-report">
```

```
            <div class="left">
              <img src="src/img/image.png" alt="" />
            </div>
            <center>
              <h2>Advisory</h2>
              <p>Las Pinas City Disaster Risk Reduction Management
Council</p>
            </center>
            <div class="right">
              <img src="src/img/triangle-logo.png" alt="" />
            </div>
          </div>
          <div class="body-report">
            <div class="body-header">
              <h2>What to do before a typhoon?</h2>
            </div>
            <ol>
              <li>Listen to weather updates and advisories.</li>
              <li>
                Keep watch for warnings and plans regarding evacuation
in your
                community.
              </li>
              <li>Check your house's condition and make necessary
repairs.</li>
              <li>
                Prepare a survival kit. (emergency items such as canned
goods,
                water and first aid kits)
              </li>
              <li>
                Keep your survival kit in an area where you can easily
get it in
                case of emergency.
              </li>
              <li>
                Evacuate immediately once asked by the authorities to
do so
              </li>
            </ol>
          </div>
        </div>
        <div class="weather-report">
          <div class="color-coded">
            <center>
              <h2>Color-Coded Rainfall Advisories</h2>
            </center>
            <div class="level">
              <div
                style="
                  background: red;
                  padding: 1em;
                  display: flex;
                  flex-direction: column;
                "
              >
                <i class="fa-solid fa-cloud-showers-water"></i>
```

```
            <span style="font-weight: bold">Evacuate</span>
            <p>Serious flooding expected in low lying areas</p>
          </div>
          <div
            style="
              background: orange;
              padding: 1em;
              color: #000;
              display: flex;
              flex-direction: column;
            "
          >
            <i class="fa-solid fa-cloud-showers-heavy"></i>
            <span style="font-weight: bold">Alert</span>
            <p>Serious flooding expected in low lying areas</p>
          </div>
          <div
            style="
              background: #ffd700;
              color: #000;
              padding: 1em;
              display: flex;
              flex-direction: column;
            "
          >
            <i class="fa-solid fa-cloud"></i>
            <span style="font-weight: bold">Monitor</span>
            <p>Serious flooding expected in low lying areas</p>
          </div>
          <div
            style="
              background: green;
              padding: 1em;
              display: flex;
              flex-direction: column;
            "
          >
            <i class="fa-solid fa-cloud-sun"></i>
            <span style="font-weight: bold">No Alert</span>
            <p>You're safe</p>
          </div>
        </div>
      </div>
    </div>
    <!-- add more -->
  </div>
</section>
<footer>
  <div class="footer-parent">
    <div class="footer-logo">
      <img src="src/img/footer-logo.png" alt="" />
    </div>
    <div class="footer-links">
      <li><a href="/">Home</a></li>
      <li><a href="aboutUs">About Us</a></li>
      <li><a href="contact">Contact</a></li>
    </div>
```

```
        <div class="footer-links">
          <li><a href="" style="text-decoration:
underline">Advisories</a></li>
          <li><a href="weather">Weather</a></li>
        </div>
        <div class="footer-description">
          LIGTAS is a localized weather monitoring system located in
the city of
          Las Piñas, built to withstand and measure various of threats,
and
          records crucial information about the harsh environmental
conditions
          and turns it into the most useful data around yours.
        </div>
        <div class="foot-another">
          <span>For more announcements kindly refer to these
websites.</span>
          <div class="foot-me">
            <li>
              <a href="https://laspinascity.gov.ph/home"
target="_blank">
                <i class="fa-solid fa-globe"></i>
              </a>
            </li>
            <li>
              <a href="https://facebook.com/lpdrrmo/" target="_blank">
                <i class="fa-brands fa-square-facebook"></i>
              </a>
            </li>
          </div>
        </div>
      </div>
    </footer>
    <div class="copyright">© 2023 LIGTAS Weathering System.</div>
    <script src="js/main.js"></script>
  </body>
</html>
```

Banaba Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin
/>
    <link
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;4
00;600&display=swap"
      rel="stylesheet"
    />
    <title>Ligtas - Status: Banaba</title>
    <link rel="stylesheet" href="css/weather.css" />
```

```
    <link
      rel="stylesheet"
      type="text/css"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
    />
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href="src/favicon/apple-touch-icon.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="32x32"
      href="src/favicon/favicon-32x32.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="16x16"
      href="src/favicon/favicon-16x16.png"
    />
    <link rel="manifest" href="src/favicon/site.webmanifest" />
    <meta name="msapplication-TileColor" content="#da532c" />
    <meta name="theme-color" content="#ffffff" />
    <script
src="https://cdn.socket.io/4.5.1/socket.io.min.js"></script>
    <script src="/socket.io/socket.io.js"></script>
    <script>
      document.addEventListener("DOMContentLoaded", function () {
        // Your Socket.IO client code here
        // const socket = io();
        const socket = io.connect("https://tupmanila.ligtas.tech");
        // Listen for 'weatherUpdate' event
        socket.on("banabaFloodUpdate", (banabaFloodData) => {
          console.log("Banaba Flood Data");
          // Update the display or trigger necessary actions using the
received weather data
          const banabaWaterHeightElement =
            document.getElementById("banabaWaterHeight");
          const banabaWaterLevelStatusElement =
document.getElementById(
            "banabaWaterLevelStatus"
          );
          const banabaBatteryPercentageElement =
document.getElementById(
            "banabaBatteryPercentage"
          );

          // Update the HTML elements with the received weather data
          banabaWaterHeightElement.innerHTML =
            banabaFloodData.banabaWaterHeight;
          banabaWaterLevelStatusElement.innerHTML =
            banabaFloodData.banabaWaterLevelStatus;
          banabaBatteryPercentageElement.innerHTML =
            banabaFloodData.banabaBatteryPercentage;
```

```
      });
    });
  </script>
</head>
<body id="body-bg">
  <header class="nav-container">
    <nav class="nav-parent">
      <div class="nav-logo">
        <a href="/">
          <img src="img/logo.png" />
          <span><b>LIGTAS |</b> Weather Monitoring System</span>
        </a>
      </div>
      <ul class="nav-links toggle">
        <li><a href="/">Home</a></li>
        <li><a href="weather">Weather</a></li>
        <li class="dropdown">
          <a class="desktop-toggle">Water Level</a>
          <ul class="dropdown-content drop">
            <li><a href="banaba">Banaba</a></li>
            <li><a href="marigold">Marigold</a></li>
          </ul>
        </li>
        <li><a href="advisory">Advisories</a></li>
        <li><a href="aboutUs">About Us</a></li>
        <li><a href="contact">Contacts</a></li>
      </ul>
      <div class="menu-lines">
        <i class="fa-solid fa-bars"></i>
      </div>
    </nav>
  </header>

  <div class="container">
    <div class="main-content">
      <div class="weather level" id="temp">
        <div class="card">
          <div class="row justify-content">
            <div class="row">
              <img
                src="img/pin-icon.png"
                alt="pin-icon"
                class="small-icon"
              />
              <h1 class="card-label">Banaba</h1>
            </div>
            <div class="level-small-card">
              <p class="blue-txt m-0" style="text-align: center">
                <b>Battery Percentage:</b>
              </p>
              <div class="battery-percentage center relative">
                <div class="center">
                  <img
                    src="img/battery-icon.png"
                    alt="battery-icon"
                    class="battery-icon"
                  />
```

```
                  </div>
                  <p>
                    <span
                      id="banabaBatteryPercentage"
                      style="font-size: 18px; font-weight: bold"
                    >
                    </span>
                    %
                  </p>
                </div>
              </div>
            </div>
            <div>
              <p class="card-header m-0">
                Real-time Prediction for every 6-hour
              </p>
              <p class="card-subheader">Water Level Status as of
today:</p>
              <h1 class="card-bold">
                <span id="banabaWaterLevelStatus"></span> :
                <span id="banabaWaterHeight"></span>cm
              </h1>
            </div>
            <div id="scroll">
              <div class="weather-display-card">
                <div id="banabaWaterHeight-forecast"></div>
              </div>
            </div>
          </div>
        </div>
      </div>

      <footer>
        <div class="footer-parent">
          <div class="footer-logo">
            <img src="img/footer-logo.png" alt="" style="width: 64px" />
          </div>
          <div class="footer-links">
            <li><a href="/">Home</a></li>
            <li><a href="aboutUs">About Us</a></li>
            <li><a href="contact">Contact</a></li>
          </div>
          <div class="footer-links">
            <li><a href="advisory">Advisories</a></li>
            <li><a href="weather">Weather</a></li>
          </div>
          <div class="footer-description">
            LIGTAS is a localized weather monitoring system located in
the city of
            Las Piñas, built to withstand and measure various of threats,
and
            records crucial information about the harsh environmental
conditions
            and turns it into the most useful data around yours.
          </div>
          <div class="foot-another">
```

```
          <span>For more announcements kindly refer to these
websites.</span>
          <div class="foot-me">
            <li>
              <a href="https://laspinascity.gov.ph/home"
target="_blank">
                <i class="fa-solid fa-globe"></i>
              </a>
            </li>
            <li>
              <a href="https://facebook.com/lpdrrmo/" target="_blank">
                <i class="fa-brands fa-square-facebook"></i>
              </a>
            </li>
          </div>
        </div>
      </div>
    </footer>
    <div class="copyright">© 2023 LIGTAS Weathering System.</div>
    <script src="js/main.js"></script>
  <script src="js/deploy.js"></script>
    <script src =”js/banaba.js”></script>
  </body>
</html>
```

Contact Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin
/>
    <link
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;4
00;600&display=swap"
      rel="stylesheet"
    />
    <title>Ligtas - Contact Us</title>
    <link rel="stylesheet" href="css/style.css" />
    <link
      rel="stylesheet"
      type="text/css"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
    />
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href="src/favicon/apple-touch-icon.png"
    />
    <link
      rel="icon"
```

```
      type="image/png"
      sizes="32x32"
      href="src/favicon/favicon-32x32.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="16x16"
      href="src/favicon/favicon-16x16.png"
    />
    <link rel="manifest" href="src/favicon/site.webmanifest" />
    <meta name="msapplication-TileColor" content="#da532c" />
    <meta name="theme-color" content="#ffffff" />
  </head>
  <body>
    <header class="nav-container">
      <nav class="nav-parent">
        <div class="nav-logo">
          <a href="/">
            <img src="src/img/logo.png" />
            <span><b>LIGTAS |</b> Weather Monitoring System</span>
          </a>
        </div>
        <ul class="nav-links toggle">
          <li><a href="/">Home</a></li>
          <li><a href="weather">Weather</a></li>
          <li class="dropdown">
            <a><button>Water Level</button></a>
            <ul class="dropdown-content toggle">
              <li><a href="banaba">Banaba</a></li>
              <li><a href="marigold">Marigold</a></li>
            </ul>
          </li>
          <li><a href="advisory">Advisories</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="" style="text-decoration:
underline">Contacts</a></li>
        </ul>
        <div class="menu-lines">
          <i class="fa-solid fa-bars"></i>
        </div>
      </nav>
    </header>
    <section class="contact-container">
      <div class="contact-parent">
        <div class="contact-list">
          <div class="contact-header">
            <span>Leave us a Message</span>
          </div>
          <form
            action="/cont/send"
            method="post"
            enctype="application/x-www-form-urlencoded"
            class="form-input"
          >
            <div class="form-group">
              <label for="">Your Name</label>
```

```
              <input
                type="text"
                name="name"
                id="name"
                autocomplete="off"
                required
              />
            </div>
            <div class="form-group">
              <label for="">Your Email</label>
              <input
                type="email"
                name="email"
                id="email"
                autocomplete="off"
                required
              />
            </div>
            <div class="form-group">
              <label for="">Subject</label>
              <input
                type="text"
                name="subject"
                id="subject"
                autocomplete="off"
                required
              />
            </div>
            <div class="form-group">
              <p>Message</p>
              <textarea
                id="message"
                name="message"
                rows="10"
                cols="50"
                required
              ></textarea>
            </div>
          </form>
          <div
            style="
              display: flex;
              justify-content: right;
              margin-bottom: 0.657em;
            "
          >
            <button type="submit">Submit</button>
          </div>
        </div>
        <div class="contact-hero">
          <div class="map-responsive">
            <iframe
              src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!
1d3863.82415517467!2d120.9727201771564!3d14.437295345057905!2m3!1f0!2f0
!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x3397d3965390411d%3A0x3d7e272df
6197afb!2sDRRMO%20BUILDING!5e0!3m2!1sen!2sph!4v1683641289540!5m2!1sen!2
sph"
```

```
              style="border: 0"
              allowfullscreen=""
              loading="lazy"
              width="500"
              height="400"
              referrerpolicy="no-referrer-when-downgrade"
            >
            </iframe>
          </div>
          <div class="contact-details" style="list-style-type: none">
            <li>Florlina Dela Cruz, Las Piñas, Metro Manila</li>
            <li>Hotline: 1234-5678,</li>
            <li>Contact Number: (+63)9999567841</li>
            <li>
              <a href="mailto:ligtas.system@gmail.com" style="color:
#35445f">
                ligtas.system@gmail.com
              </a>
            </li>
          </div>
        </div>
      </div>
    </section>
    <footer>
        </div>
        <div class="footer-description">
          LIGTAS is a localized weather monitoring system located in
the city of
          Las Piñas, built to withstand and measure various of threats,
and
          records crucial information about the harsh environmental
conditions
          and turns it into the most useful data around yours.
        </div>
        <div class="foot-another">
          <span>For more announcements kindly refer to these
websites.</span>
          <div class="foot-me">
            <li>
              <a href="https://laspinascity.gov.ph/home"
target="_blank">
                <i class="fa-solid fa-globe"></i>
              </a>
            </li>
            <li>
              <a href="https://facebook.com/lpdrrmo/" target="_blank">
                <i class="fa-brands fa-square-facebook"></i>
              </a>
            </li>
          </div>
        </div>
      </div>
    </footer>
    <div class="copyright">© 2023 LIGTAS Weathering System.</div>
    <script src="js/main.js"></script>
  </body>
</html>
```

Index Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin
/>
    <link
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;4
00;600&display=swap"
      rel="stylesheet"
    />
    <title>Ligtas - Weather Monitoring System</title>
    <link rel="stylesheet" href="css/style.css" />
    <link
      rel="stylesheet"
      type="text/css"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
    />
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href="src/favicon/apple-touch-icon.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="32x32"
      href="src/favicon/favicon-32x32.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="16x16"
      href="src/favicon/favicon-16x16.png"
    />
    <link rel="manifest" href="src/favicon/site.webmanifest" />
    <meta name="msapplication-TileColor" content="#da532c" />
    <meta name="theme-color" content="#ffffff" />
  </head>
  <body>
    <header class="nav-container">
      <nav class="nav-parent">
        <div class="nav-logo">
          <a href="">
            <img src="src/img/logo.png" />
            <span><b>LIGTAS |</b> Weather Monitoring System</span>
          </a>
        </div>
        <ul class="nav-links toggle">
```

```
          <li><a href="" style="text-decoration:
underline">Home</a></li>
          <li><a href="weather">Weather</a></li>
          <li class="dropdown">
            <a><button>Water Level</button></a>
            <ul class="dropdown-content toggle">
              <li><a href="banaba">Banaba</a></li>
              <li><a href="marigold">Marigold</a></li>
            </ul>
          </li>
          <li><a href="advisory">Advisories</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contacts</a></li>
        </ul>
        <div class="menu-lines">
          <i class="fa-solid fa-bars"></i>
        </div>
      </nav>
    </header>
    <section class="intro-bg">
      <div class="intro-parent">
        <div class="introduction">
          <span>With Ligtas, Know Disaster, No Disaster!</span>
        </div>
        <div id="time" class="record-time"></div>
      </div>
    </section>
    <section class="intro-description">
      <div class="description-parent">
        <article class="description">
          <figcaption>
            This website presents official weather observations,
weather
            forecasts and prior advisories for selected barangays
located in the
            city of Las Pinas.
          </figcaption>
          <figcaption>
            Ligtas System has a part to play has a part to play in
disaster
            reduction by providing high-quality public weather
services,such as
            weather forecasts, early warnings of dangerous weather,
outreach
            initiatives to increase public awareness of weather hazards
to the
            citizens as well as cooperation with disaster relief
organizations
            to reduce loss of life and property. Weather and Water
level
            navigation is shown alongside worded forecasts in this
version to
            facilitate visual inspection.
          </figcaption>
        </article>
      </div>
      <div class="definition">
```

```
        <div class="ligtas-info order1">
          <span>What is LIGTAS?</span>
          <p>
            In every aspect of the industry in times of hazards,
weather
            monitoring is of substantial importance. LIGTAS provides a
useful
            weather information, data analysis, and forecasting to the
local
            communities in the city of Las Piñas for preparedness to
reduce the
            severity and impact of a disaster such as heat wave,
tornado,
            thunderstorm, and flash flood.
          </p>
        </div>
        <div class="ligtas-hero order2">
          <img src="src/img/home-logo.png" alt="" style="width: 165px"
/>
        </div>
      </div>
      <div class="definition">
        <div class="ligtas-info order3">
          <span>About Us</span>
          <div class="inline-pota">
            <p>
              Here at Ligtas, we are driven to do our part in making a
              comprehensive early warning warning system that alerts
people when
              dangerous weather is approaching and provides information
on what
              actions can governments, communities, and individuals can
take to
              lessen the effects. We strive to build a positive impact
with all
              of our pursuits. Are you ready to join us and create real
              transformation in the lives of so many?
            </p>
            <hr class="underline" />
            <p>
              The primary goal of this weather monitoring system
website is to
              provide information to the citizen and organizations to
reduce
              weather-related losses and enhance societal benefits,
including
              protection of life and property, public health and
safety, and
              support of economic prosperity and quality of life.
            </p>
          </div>
        </div>
        <div class="ligtas-hero order4">
          <img src="src/img/home-img-about.png" alt="" style="width:
820px" />
        </div>
      </div>
```

```
      <div class="ligtas-tagline">
        <h2>With Ligtas, lagi kang EFAS!</h2>
      </div>
    </section>
    <footer>
      <div class="footer-parent">
        <div class="footer-logo">
          <img src="src/img/footer-logo.png" alt="" />
        </div>
        <div class="footer-links">
          <li><a href="" style="text-decoration:
underline">Home</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contact</a></li>
        </div>
        <div class="footer-links">
          <li><a href="advisory">Advisories</a></li>
          <li><a href="weather">Weather</a></li>
        </div>
        <div class="footer-description">
          LIGTAS is a localized weather monitoring system located in
the city of
          Las Piñas, built to withstand and measure various of threats,
and
          records crucial information about the harsh environmental
conditions
          and turns it into the most useful data around yours.
        </div>
        <div class="foot-another">
          <span>For more announcements kindly refer to these
websites.</span>
          <div class="foot-me">
            <li>
              <a href="https://laspinascity.gov.ph/home"
target="_blank">
                <i class="fa-solid fa-globe"></i>
              </a>
            </li>
            <li>
              <a href="https://facebook.com/lpdrrmo/" target="_blank">
                <i class="fa-brands fa-square-facebook"></i>
              </a>
            </li>
          </div>
        </div>
      </div>
    </footer>
    <div class="copyright">© 2023 LIGTAS Weathering System.</div>
    <script src="js/main.js"></script>
    <script src="js/notification.js"></script>
  </body>
</html>
```

Marigold Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin
/>
    <link
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;4
00;600&display=swap"
      rel="stylesheet"
    />
    <title>Ligtas - Status: Marigold</title>
    <link rel="stylesheet" href="css/weather.css" />
    <link
      rel="stylesheet"
      type="text/css"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
    />
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href="src/favicon/apple-touch-icon.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="32x32"
      href="src/favicon/favicon-32x32.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="16x16"
      href="src/favicon/favicon-16x16.png"
    />
    <link rel="manifest" href="src/favicon/site.webmanifest" />
    <meta name="msapplication-TileColor" content="#da532c" />
    <meta name="theme-color" content="#ffffff" />
    <script
src="https://cdn.socket.io/4.5.1/socket.io.min.js"></script>
    <script src="/socket.io/socket.io.js"></script>
    <script>
      document.addEventListener("DOMContentLoaded", function () {
        // Your Socket.IO client code here
        // const socket = io();
        const socket = io.connect("https://tupmanila.ligtas.tech");
        // Listen for 'weatherUpdate' event
        socket.on("marigoldFloodUpdate", (marigoldFloodData) => {
          console.log("Marigold Flood Data");
          // Update the display or trigger necessary actions using the
received weather data
          const marigoldWaterHeightElement = document.getElementById(
```

```
            "marigoldWaterHeight"
          );
          const marigoldWaterLevelStatusElement =
document.getElementById(
            "marigoldWaterLevelStatus"
          );
          const marigoldBatteryPercentageElement =
document.getElementById(
            "marigoldBatteryPercentage"
          );

          // Update the HTML elements with the received weather data
          marigoldWaterHeightElement.innerHTML =
            marigoldFloodData.marigoldWaterHeight;
          marigoldWaterLevelStatusElement.innerHTML =
            marigoldFloodData.marigoldWaterLevelStatus;
          marigoldBatteryPercentageElement.innerHTML =
            marigoldFloodData.marigoldBatteryPercentage;
        });
      });
    </script>
  </head>
  <body id="body-bg">
    <header class="nav-container">
      <nav class="nav-parent">
        <div class="nav-logo">
          <a href="/">
            <img src="img/logo.png" />
            <span><b>LIGTAS |</b> Weather Monitoring System</span>
          </a>
        </div>
        <ul class="nav-links toggle">
          <li><a href="/">Home</a></li>
          <li><a href="weather">Weather</a></li>
          <li class="dropdown">
            <a class="desktop-toggle">Water Level</a>
            <ul class="dropdown-content drop">
              <li><a href="banaba">Banaba</a></li>
              <li><a href="marigold">Marigold</a></li>
            </ul>
          </li>
          <li><a href="advisory">Advisories</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contacts</a></li>
        </ul>
        <div class="menu-lines">
          <i class="fa-solid fa-bars"></i>
        </div>
      </nav>
    </header>

    <div class="container">
      <div class="main-content">
        <div class="weather level" id="temp">
          <div class="card">
            <div class="row justify-content">
              <div class="row">
```

```
                <img
                  src="img/pin-icon.png"
                  alt="pin-icon"
                  class="small-icon"
                />
                <h1 class="card-label">Marigold</h1>
              </div>
              <div class="level-small-card">
                <p class="blue-txt m-0" style="text-align: center">
                  <b>Battery Percentage:</b>
                </p>
                <div class="battery-percentage center relative">
                  <div class="center">
                    <img
                      src="img/battery-icon.png"
                      alt="battery-icon"
                      class="battery-icon"
                    />
                  </div>
                  <p>
                    <span
                      id="marigoldBatteryPercentage"
                      style="font-size: 18px; font-weight: bold"
                    >
                    </span>
                    %
                  </p>
                </div>
              </div>
            </div>
            <div>
              <p class="card-header m-0">
                Real-time Prediction for every 6-hour
              </p>
              <p class="card-subheader">Water Level Status as of
today:</p>
              <h1 class="card-bold">
                <span id="marigoldWaterLevelStatus"></span> :
                <span id="marigoldWaterHeight"></span>cm
              </h1>
            </div>
            <div id="scroll">
              <div class="weather-display-card">
                <div id="marigoldWaterHeight-forecast"></div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>


    <footer>
      <div class="footer-parent">
        <div class="footer-logo">
          <img src="img/footer-logo.png" alt="" style="width: 64px" />
        </div>
```

```
        <div class="footer-links">
          <li><a href="/">Home</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contact</a></li>
        </div>
        <div class="footer-links">
          <li><a href="advisory">Advisories</a></li>
          <li><a href="weather">Weather</a></li>
        </div>
        <div class="footer-description">
          LIGTAS is a localized weather monitoring system located in
the city of
          Las Piñas, built to withstand and measure various of threats,
and
          records crucial information about the harsh environmental
conditions
          and turns it into the most useful data around yours.
        </div>
        <div class="foot-another">
          <span>For more announcements kindly refer to these
websites.</span>
          <div class="foot-me">
            <li>
              <a href="https://laspinascity.gov.ph/home"
target="_blank">
                <i class="fa-solid fa-globe"></i>
              </a>
            </li>
            <li>
              <a href="https://facebook.com/lpdrrmo/" target="_blank">
                <i class="fa-brands fa-square-facebook"></i>
              </a>
            </li>
          </div>
        </div>
      </div>
    </footer>
    <div class="copyright">© 2023 LIGTAS Weathering System.</div>
    <script src="js/main.js"></script>
  <script src="js/deploy.js"></script>
    <script src=”js/marigold.js”></script>
  </body>
</html>
```

Weather Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin
/>
    <link
```

```
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;4
00;600&display=swap"
      rel="stylesheet"
    />
    <link
      rel="stylesheet"
      type="text/css"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css"
    />
    <title>Ligtas</title>
    <link rel="stylesheet" href="css/weather.css" />
    <script
src="https://cdn.socket.io/4.5.1/socket.io.min.js"></script>
    <script src="/socket.io/socket.io.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script>
      document.addEventListener("DOMContentLoaded", function () {
        // Your Socket.IO client code here
        // const socket = io();
        const socket = io.connect("https://tupmanila.ligtas.tech");
        // Listen for 'weatherUpdate' event
        socket.on("weatherUpdate", (weatherData) => {
          console.log("WeatherData");
          // Update the display or trigger necessary actions using the
received weather data
          const temperatureElement =
document.getElementById("temperature");
          const humidityElement = document.getElementById("humidity");
          const atmosphericPressureElement = document.getElementById(
            "atmosphericPressure"
          );
          const windSpeedElement =
document.getElementById("windSpeed");
          const windDirectionElement =
document.getElementById("windDirection");
          const rainIntensityElement =
document.getElementById("rainIntensity");
          const batteryPercentageElement =
            document.getElementById("batteryPercentage");

          // Update the HTML elements with the received weather data
          temperatureElement.innerHTML = weatherData.temp;
          humidityElement.innerHTML = weatherData.humid;
          atmosphericPressureElement.innerHTML =
            weatherData.atmosphericPressure;
          windSpeedElement.innerHTML =
weatherData.windSpeed.toFixed(3);
          windDirectionElement.innerHTML = weatherData.windDirection;
          rainIntensityElement.innerHTML =
weatherData.rainIntensity.toFixed(3);
          batteryPercentageElement.innerHTML =
            weatherData.weatherBatteryPercentage;
        });
      });
    </script>
  </head>
```

```
  <body id="body-bg">
    <header class="nav-container">
      <nav class="nav-parent">
        <div class="nav-logo">
          <a href="">
            <img src="img/logo.png" />
            <span><b>LIGTAS |</b> Weather Monitoring System</span>
          </a>
        </div>
        <ul class="nav-links toggle">
          <li><a href="/">Home</a></li>
          <li>
            <a href="weather" style="text-decoration:
underline">Weather</a>
          </li>
          <li class="dropdown">
            <a class="desktop-toggle">Water Level</a>
            <ul class="dropdown-content drop">
              <li><a href="banaba">Banaba</a></li>
              <li><a href="marigold">Marigold</a></li>
            </ul>
          </li>
          <li><a href="advisory">Advisories</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contacts</a></li>
        </ul>
        <div class="menu-lines">
          <i class="fa-solid fa-bars"></i>
        </div>
      </nav>
    </header>

    <div class="container" id="weatherData">
      <div class="main-content">
        <div class="row justify-content-center">
          <div class="col-80">
            <div class="weather level" id="temp">
              <div class="card card-80">
                <div class="justify-content">
                  <div class="row">
                    <img
                      src="img/pin-icon.png"
                      alt="pin-icon"
                      class="small-icon"
                    />
                    <h1 class="card-label">Brgy. Talon Dos</h1>
                  </div>
                  <div class="row">
                    <div class="col-60">
                      <h1 class="temp-text">
                        <span id="temperature"></span> °C
                      </h1>
                      <div class="row justify-content-center">
                        <img
                          src="img/weather-icon.png"
                          alt="weather-icon"
```

```
                        class="medium-icon"
                      />
                      <p class="pl-30 weather-lbl">Mostly Cloudy</p>
                    </div>
                  </div>
                  <div class="col-40">
                    <div class="level-graph-card">
                      <canvas id="myChart"></canvas>
                      <div id="scroll" class="scroll-small">
                        <div class="weather-display-card">
                          <div id="temperature-forecast">°C</div>
                        </div>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
            <div class="col-20">
              <div class="yellow-card">
                <p>Philippine Standard Time</p>
                <p id="time"></p>
                <p id="todaysTime"></p>
                <p id="date"></p>
              </div>
              <div class="xsmall-card">
                <p class="blue-txt m-0" style="text-align: center">
                  <b>Battery Percentage:</b>
                </p>
                <div
                  class="battery-percentage battery-percentage-weather
center relative"
                >
                  <div class="center">
                    <img
                      src="img/battery-icon.png"
                      alt="battery-icon"
                      class="battery-icon-big"
                    />
                  </div>
                  <p class="battery-percentage-weather-p">
                    <span
                      id="batteryPercentage"
                      style="font-size: 18px; font-weight: bold"
                    >
                    </span>
                    %
                  </p>
                </div>
              </div>
            </div>
          </div>
        </div>
        <div class="main-content2">
          <div class="row justify-content-center">
```

```
          <div class="col-25">
            <div class="xsmall-card-type">
              <p>Wind</p>
              <div>
                <h4>
                  <span style="font-size: 15px"
id="windDirection"></span>°
                </h4>
                <p style="color: #2d3c5b; margin-top: 3px">
                  <span id="windSpeed"></span> km/hr
                </p>
              </div>
              <div id="scroll" class="scroll-xsmall">
                <div class="weather-display-card">
                  <div id="windSpeed-forecast">km/hr</div>
                </div>
              </div>
            </div>
          </div>
          <div class="col-25">
            <div class="xsmall-card-type">
              <p>Air Pressure</p>
              <div class="row justify-content-center">
                <p style="margin-top: 25px; color: #2d3c5b">
                  <span id="atmosphericPressure"></span> mbar
                </p>
                <img
                  style="margin-top: 18px"
                  src="img/air-pressure-icon-sml.png"
                  alt="icon"
                  class="m-small-icon"
                />
              </div>
              <div id="scroll" class="scroll-xsmall">
                <div class="weather-display-card">
                  <div id="pressure-forecast">mbar</div>
                </div>
              </div>
            </div>
          </div>
          <div class="col-25">
            <div class="xsmall-card-type">
              <p>Humidity</p>
              <div class="row justify-content-center">
                <p style="margin-top: 25px; color: #2d3c5b">
                  <span id="humidity"></span>%
                </p>
                <img
                  style="margin-top: 20px"
                  src="img/water-icon-sml.png"
                  alt="icon"
                  class="m-small-icon"
                />
              </div>
              <div id="scroll" class="scroll-xsmall">
                <div class="weather-display-card">
                  <div id="humidity-forecast">%</div>
```

```
              </div>
            </div>
          </div>
        </div>
        <div class="col-25">
          <div class="xsmall-card-type">
            <p>Rainfall Intensity</p>
            <div class="row justify-content-center">
              <p style="margin-top: 25px; color: #2d3c5b">
                <span id="rainIntensity"></span> mm/hr
              </p>
              <img
                style="margin-top: 20px"
                src="img/rain-intensity-sml.png"
                alt="icon"
                class="m-small-icon"
              />
            </div>
            <div id="scroll" class="scroll-xsmall">
              <div class="weather-display-card">
                <div id="rainfall-forecast">mm/hr</div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>

    <footer>
      <div class="footer-parent">
        <div class="footer-logo">
          <img src="img/footer-logo.png" style="width: 64px" />
        </div>
        <div class="footer-links">
          <li><a href="/" style="text-decoration:
underline">Home</a></li>
          <li><a href="aboutUs">About Us</a></li>
          <li><a href="contact">Contact</a></li>
        </div>
        <div class="footer-links">
          <li><a href="advisory">Advisories</a></li>
          <li><a href="weather">Weather</a></li>
        </div>
        <div class="footer-description">
          LIGTAS is a localized weather monitoring system located in
the city of
          Las Piñas, built to withstand and measure various of threats,
and
          records crucial information about the harsh environmental
conditions
          and turns it into the most useful data around yours.
        </div>
        <div class="foot-another">
          <span>For more announcements kindly refer to these
websites.</span>
          <div class="foot-me">
```

```
            <li>
              <a href="https://laspinascity.gov.ph/home"
target="_blank">
                <i class="fa-solid fa-globe"></i>
              </a>
            </li>
            <li>
              <a href="https://facebook.com/lpdrrmo/" target="_blank">
                <i class="fa-brands fa-square-facebook"></i>
              </a>
            </li>
          </div>
        </div>
      </div>
    </footer>
    <div class="copyright">© 2023 LIGTAS Weathering System.</div>
    <script src="js/main.js"></script>
    <script src="js/chart.js"></script>
  <script src="js/deploy.js"></script>
    <script src =”js/weather.js”></script>
  </body>
</html>
```

deploy.js

```
// Function to display the forecasted data
function displayForecastedData(data) {
  const forecastDataElement = document.getElementById('forecast-data');

  data.forEach(forecast => {
    const row = document.createElement('tr');
    row.innerHTML = `
      <td>${forecast.time}</td>
      <td>${forecast.temperature}</td>
      <td>${forecast.humidity}</td>
      <td>${forecast.windSpeed}</td>
      <td>${forecast.windDirection}</td>
      <td>${forecast.rainIntensity}</td>
      <td>${forecast.banabawaterLevel}</td>
      <td>${forecast.marigoldwaterLevel}</td>
      <td>${forecast.airPressure}</td>
    `;
    forecastDataElement.appendChild(row);
  });
}

// Make a request to the Node.js server to retrieve the forecasted data
fetch('/getForecastedData)
  .then(response => response.json())
  .then(data => displayForecastedData(data))
  .catch(error => console.error('Error retrieving forecasted data:',
error));
```

main.js

```
function displayDateTime() {
```

```
    const currentDate = new Date();
    const options = {
        weekday: 'long',
        year: 'numeric',
        month: 'long',
        day: 'numeric',
        hour: 'numeric',
        minute: 'numeric',
        second: 'numeric',
        hour12: true
    };
    const dateTimeString = currentDate.toLocaleString('en-US',
options);
    const milliseconds = currentDate.getSeconds();
    localStorage.setItem("dateTime", dateTimeString);

    document.getElementById("time").innerHTML = dateTimeString;
}

window.onload = function() {
    const storedDateTime = localStorage.getItem("dateTime");
    if (storedDateTime) {
        document.getElementById("time").innerHTML = storedDateTime;
    } else {
        displayDateTime();
    }

    setInterval(displayDateTime, 1); // fetch data offline
};

//Toggle method
const toggleMenu = document.querySelector(".menu-lines i"),
      navLinks = document.querySelector(".nav-links"),
      toggleDropdown = document.querySelector(".desktop-toggle"),
      dropdown = document.querySelector(".dropdown-content")

toggleMenu.addEventListener("click", () => {
    if (toggleMenu.classList.toggle("fa-xmark")) {
        toggleMenu.classList.remove("fa-bars")
        document.body.style.overflow = "hidden"
        navLinks.classList.remove("toggle")
    } else {
        toggleMenu.classList.add("fa-bars")
        document.body.style.overflow = "auto"
        navLinks.classList.add("toggle")
    }
})

toggleDropdown.addEventListener("click", () => {
    dropdown.classList.toggle("drop")
})
```

notification.js

```
window.onload = async () => {
  try {
```

```
    if (Notification.permission === "default") {
      const permission = await Notification.requestPermission();
      if (permission === "granted") {
        await registerWorker();
      } else {
        alert("Please allow notifications.");
      }
    } else if (Notification.permission === "granted") {
      await registerWorker();
    } else {
      alert("Please allow notifications.");
    }
  } catch (error) {
    console.error(error);
    alert("Failed to request notification permission.");
  }
};

async function registerWorker() {
  const publicVapidKey =
"BAkhoNbHSlRRZIoR2994m6wRcnzf4h8C0OmocGd7SQ6CU3XofZsiLJ2Q5MN0_uY_wcf5l7
Dg9SM2PufVsk87xzY";

  if ("serviceWorker" in navigator) {
    try {
      const registration = await
navigator.serviceWorker.register("js/service-worker.js");

      const subscription = await registration.pushManager.subscribe({
        userVisibleOnly: true,
        applicationServerKey: urlBase64ToUint8Array(publicVapidKey),
      });

      await sendSubscriptionToServer(subscription);
    } catch (error) {
      console.error("Failed to register service worker:", error);
      alert("Failed to register service worker.");
    }
  } else {
    console.error("Service workers are not supported in this
browser.");
    alert("Service workers are not supported in this browser.");
  }
}

function urlBase64ToUint8Array(base64String) {
  const padding = "=".repeat((4 - (base64String.length % 4)) % 4);
  const base64 = (base64String + padding)
    .replace(/-/g, "+")
    .replace(/_/g, "/");

  const rawData = window.atob(base64);
  const outputArray = new Uint8Array(rawData.length);

  for (let i = 0; i < rawData.length; ++i) {
    outputArray[i] = rawData.charCodeAt(i);
  }
```

```
  return outputArray;
}

async function sendSubscriptionToServer(subscription) {
  try {
    const response = await fetch("/subscribe", {
      method: "POST",
      body: JSON.stringify(subscription),
      headers: { "Content-Type": "application/json" },
    });

    if (response.ok) {
      console.log("Subscription details sent to the server.");
    } else {
      throw new Error(`Failed to send subscription details to the
server. Status: ${response.status}`);
    }
  } catch (error) {
    console.error("Failed to send subscription details to the server:",
error);
    alert("Failed to send subscription details to the server.");
  }
}



// Listen for push notifications
navigator.serviceWorker.addEventListener("message", (event) => {
  const { title, body } = event.data;

  // Display the notification
  if (Notification.permission === "granted") {
    new Notification(title, { body });
  }
});
```

service-worker.js

```
self.addEventListener('push', event => {
  const options = {
    body: 'This is the body of the notification',
    icon: 'img/logo.png',
    // Add any other options you want to customize the notification
  };

  event.waitUntil(
    self.registration.showNotification('Notification Title', options)
  );
});
```

Back – End Codes

Controllers

1. *For Advisory Page*

```
const asyncHandler = require("express-async-handler");
const connection = require('../database');

const getAdvisory = asyncHandler(async (io, socket) => {
  const sendAdvisoryUpdate = async () => {
    try {
      // Fetch weather parameters
      const advisoryQuery = "SELECT * FROM weatherparameters
ORDER BY id DESC LIMIT 1";
      const advisoryRows = await new Promise((resolve, reject) =>
{
        connection.query(advisoryQuery, function (err,
advisoryRows, fields) {
          if (err) {
            reject(err);
            return;
          }
          resolve(advisoryRows);
        });
      });

      // Fetch water level for banaba
      const waterLevelQueryBanaba = "SELECT * FROM banabanodes
ORDER BY id DESC LIMIT 1";
      const waterLevelRowsBanaba = await new Promise((resolve,
reject) => {
        connection.query(waterLevelQueryBanaba, function (err,
waterLevelRowsBanaba, fields) {
          if (err) {
            reject(err);
            return;
          }
          resolve(waterLevelRowsBanaba);
        });
      });

      // Fetch water level for marigold
      const waterLevelQueryMarigold = "SELECT * FROM
marigoldnodes ORDER BY id DESC LIMIT 1";
      const waterLevelRowsMarigold= await new Promise((resolve,
reject) => {
        connection.query(waterLevelQueryMarigold, function (err,
waterLevelRowsMarigold, fields) {
          if (err) {
            reject(err);
            return;
          }
          resolve(waterLevelRowsMarigold);
        });
      });
```

```
      if (advisoryRows.length > 0 ||
waterLevelRowsMarigold.length > 0 || waterLevelRowsBanaba.length
>0) {
        const mostRecentAdvisoryMeasure = advisoryRows[0];
        const mostRecentWaterLevelBanaba =
waterLevelRowsBanaba[0];
        const mostRecentWaterLevelMarigold =
waterLevelRowsMarigold[0];

        let sumRainIntensity = 0;
        let sumWindSpeed = 0;

        for (let i = 0; i < advisoryRows.length; i++) {
          sumRainIntensity += advisoryRows[i].rainIntensity;
          sumWindSpeed += advisoryRows[i].windSpeed;
        }

        const averageWindSpeed = sumWindSpeed /
advisoryRows.length;

        // Weather parameters
        const advisoryData = {
          temp: mostRecentAdvisoryMeasure.temperature,
          humid: mostRecentAdvisoryMeasure.humidity,
          windSpeed: averageWindSpeed,
          windDirection: mostRecentAdvisoryMeasure.windDirection,
          rainIntensity: sumRainIntensity,
          weatherBatteryPercentage:
mostRecentAdvisoryMeasure.BatteryPercentage,
          readingWeatherTime:
mostRecentAdvisoryMeasure.readingTime
        };
        // Water level
        const waterLevelBanaba = {
          flood:mostRecentWaterLevelBanaba.waterLevel,
          readingFloodTime:mostRecentWaterLevelBanaba.readingTime
        };
        // Water level
        const waterLevelMarigold = {
          flood:mostRecentWaterLevelMarigold.waterLevel,
          readingFloodTime:mostRecentWaterLevelMarigold.readingTi
me
        };

        // Determine advisory condition based on rainfall
intensity and wind speed
        let advisoryCondition = "";
        let advisoryConditionDescription = "";
        let output = "";

        if (advisoryData.rainIntensity === -999 &&
advisoryData.windSpeed === -999) {
            // Unknown error
            advisoryCondition = "Unknown Error";
            advisoryConditionDescription = `No data is
available`;
```

```
            output = "Weather information is currently
unavailable. Please visit the Pagasa website for the latest
weather updates.";
          } else if (advisoryData.rainIntensity < 2.5 &&
advisoryData.windSpeed < 20) {
            // Light Condition //////////////////////
            advisoryCondition = "Light";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is typically less
than 2.5 millimeters (0.1 inches) per hour, and wind speed
(${advisoryData.windSpeed} km/hr) is typically less than 20
kilometers per hour.`;
            output = "Enjoy the pleasant weather!The rain is very
light, and the wind is gentle.";
          } else if (advisoryData.rainIntensity >= 2.5 &&
advisoryData.rainIntensity <= 7.6 && advisoryData.windSpeed >= 20
&& advisoryData.windSpeed <= 40) {
            // Moderate Condition
////////////////////////////////////
            advisoryCondition = "Moderate";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is generally
between 2.5 to 7.6 millimeters (0.1 to 0.3 inches) per hour, and
wind speed (${advisoryData.windSpeed} km/hr) is generally between
20 to 40 kilometers per hour.`;
            output = "Take necessary precautions. The rain is
moderate, and the wind is picking up speed.";
          } else if (advisoryData.rainIntensity >= 2.5 &&
advisoryData.rainIntensity <= 7.6 && advisoryData.windSpeed < 20)
{
            // Moderate Rainfall with Light Wind Condition
////////////////////////////////////////
            advisoryCondition = "Moderate Rainfall with Light
Wind ";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is generally
between 2.5 to 7.6 millimeters (0.1 to 0.3 inches) per hour, and
wind speed (${advisoryData.windSpeed} km/hr) is typically less
than 20 kilometers per hour, indicating moderate rainfall with
light wind conditions.`;
            output = "Take precautions for moderate rainfall. The
wind is gentle.";
          } else if (advisoryData.rainIntensity < 2.5 &&
advisoryData.windSpeed >= 20 && advisoryData.windSpeed <= 40) {
            // Moderate Wind with Light Rain Condition
            advisoryCondition = "Moderate Wind with Light Rain";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is typically less
than 2.5 millimeters (0.1 inches) per hour, and wind speed
(${advisoryData.windSpeed} km/hr) is generally between 20 to 40
kilometers per hour, indicating moderate wind with light rainfall
conditions.`;
            output = "Take precautions for moderate wind. The
rain is very light.";
          } else if (advisoryData.rainIntensity > 7.6 &&
advisoryData.windSpeed >= 20 && advisoryData.windSpeed <= 40) {
            // Heavy Rain and Moderate Wind Condition
```

```
            advisoryCondition = "Heavy Rain and Moderate Wind ";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is usually more
than 7.6 millimeters (0.3 inches) per hour, and wind speed
(${advisoryData.windSpeed} km/hr)generally between 20 to 40
kilometers per hour, indicating heavy rain with moderate wind
conditions.`;
            output = "Stay safe and seek shelter. The rain is
heavy, and moderate winds mostly persist .";
          } else if (advisoryData.rainIntensity >= 2.5 &&
advisoryData.rainIntensity <= 7.6 && advisoryData.windSpeed > 40)
{
            // Moderate Rain and Strong Wind Condition
            advisoryCondition = "Moderate Rain and Strong Wind";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h)  is generally
between 2.5 to 7.6 millimeters (0.1 to 0.3 inches) per hour, and
wind speed (${advisoryData.windSpeed} km/hr) exceeds 40
kilometers per hour or higher gusts.`;
            output = "Stay safe and seek shelter. The rain is
moderate, and the wind is strong.";
          } else if (advisoryData.rainIntensity >= 7.6 &&
advisoryData.windSpeed > 40) {
            // Heavy Rain and Strong Wind Condition
            advisoryCondition = "Heavy Rain and Strong Wind";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is usually more
than 7.6 millimeters (0.3 inches) per hour, and wind speed
(${advisoryData.windSpeed} km/hr) exceeds 40 kilometers per hour
or higher gusts.`;
            output = "Stay safe and seek shelter. The rain is
heavy, and the wind is strong.";
          } else if (advisoryData.rainIntensity >= 7.6 &&
advisoryData.windSpeed < 20) {
            // Heavy Rain and Light Wind Condition
            advisoryCondition = "Heavy Rain and Light Wind";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h)  is usually more
than 7.6 millimeters (0.3 inches) per hour, and wind speed
(${advisoryData.windSpeed} km/hr) is typically less than 20
kilometers per hour.`;
            output = "Please exercise caution due to the presence
of heavy rain and light wind.";
          } else if (advisoryData.rainIntensity < 2.5  &&
advisoryData.windSpeed > 40) {
            // Light Rain and Strong Wind Condition
            advisoryCondition = "Light Rain and Strong Wind";
            advisoryConditionDescription = `Rainfall intensity
(${advisoryData.rainIntensity.toFixed(2)} mm/h) is typically less
than 2.5 millimeters (0.1 inches) per hour, and wind speed
(${advisoryData.windSpeed} km/hr) exceeds 40 kilometers per hour
or higher gusts.`;
            output = "Please exercise caution due to the presence
of strong winds and light rainfall.";
          } else {
            advisoryCondition = "Unknown Error";
```

```
            advisoryConditionDescription = `No data is
available`;
            output = "Weather information is currently
unavailable. Please visit the Pagasa website for the latest
weather updates.";
          }

        // Add the advisory condition and description to the
advisoryData object
        advisoryData.advisoryCondition = advisoryCondition;
        advisoryData.advisoryConditionDescription =
advisoryConditionDescription;

         // Determine heat index condition based on temperature
        let heatIndexCondition = "";
        if (advisoryData.temp >= 27 && advisoryData.temp <= 32) {
          heatIndexCondition = `Caution! The temperature is
${advisoryData.temp}째C. It is important to be aware that
prolonged exposure to heat and engaging in continuous activity
can lead to fatigue.`;
        } else if (advisoryData.temp >= 33 && advisoryData.temp
<= 41) {
          heatIndexCondition = `Extreme Caution! The temperature
is ${advisoryData.temp}째C. Heat cramps and heat exhaustion can
occur if you continue to exert yourself in hot conditions.`;
        } else if (advisoryData.temp >= 42 && advisoryData.temp
<= 51) {
          heatIndexCondition = `Danger! The temperature is
${advisoryData.temp}째C. Please be cautious as heat cramps and
heat exhaustion are highly likely if you persist in your
activities without appropriate rest and cooling measures.`;
        } else if (advisoryData.temp >= 52) {
          heatIndexCondition = `Extreme Danger! The temperature
${advisoryData.temp}째C. Urgent action is necessary to avoid heat
stroke, a severe condition that can result from ongoing exposure
to high temperatures.`;
        } else{
            console.log("No temp found");
        }

         // Water level advisory conditions for Banaba
        let banabaLevelDescription = "";

        const currentHour = new Date().getHours();

        if (waterLevelBanaba.flood === -999) {
          banabaLevelDescription = "Flood information is
currently unavailable.";
        } else if (waterLevelBanaba.flood === 10) {
          banabaLevelDescription = `Banaba Water Level Node as of
now remains Low.`;
        } else if (waterLevelBanaba.flood === 20) {
          banabaLevelDescription = `Banaba Water Level Node as of
now remains Normal.`;
        } else if (waterLevelBanaba.flood === 30) {
```

```
          banabaLevelDescription = `Banaba Water Level Node as of
now remains High.`;
        } else if (waterLevelBanaba.flood === 40) {
          banabaLevelDescription = `Banaba Water Level Node as of
now remains Critical.`;
        } else {
          console.log("No Flood Data found");
        }

          // Water level advisory conditions for Banaba
          let marigoldLevelDescription = "";

          if (waterLevelMarigold.flood === -999 ){
            marigoldLevelDescription = "Flood information is
currently unavailable.";
          } else if(waterLevelMarigold.flood === 10 ) {
            marigoldLevelDescription = `Marigold Water Level Node
as of now remains Low.`;
          } else if (waterLevelMarigold.flood === 20) {
            marigoldLevelDescription = `Marigold Water Level Node
as of now remains Normal.`;
          } else if (waterLevelMarigold.flood === 30) {
            marigoldLevelDescription = `Marigold Water Level Node
as of now remains High.`;
          } else if (waterLevelMarigold.flood === 40) {
            marigoldLevelDescription = `Marigold Water Level Node
as of now remains Critical.`;
          } else{
            console.log("No Flood Data found");
          }

        if (socket && socket.emit) {
            socket.emit('advisoryUpdate', {
                advisoryData,
                waterLevelBanaba,
                waterLevelMarigold,
                advisoryCondition,
                advisoryConditionDescription,
                heatIndexCondition,
                banabaLevelDescription,
                marigoldLevelDescription,
                output
              });
            }
      } else {
        console.log("No weather data found");
      }
    } catch (err) {
      console.log("An error occurred with the query:", err);
      if (socket && socket.emit) {
        socket.emit('advisoryError', 'Error executing query');
      }
    }
  };

  // Call the function immediately to send the initial weather
data and output
```

```
  sendAdvisoryUpdate();

  // Schedule the function to be called every 5 minutes
  setInterval(sendAdvisoryUpdate, 60000);
});

module.exports = { getAdvisory };
```

2. *For Banaba Page*

```
const asyncHandler = require("express-async-handler");
const connection = require('../database');

const getFloodQueryBanaba = asyncHandler(async (io, socket) => {
  const sendFloodBanabaUpdate = async () => {
    try {
      const floodQueryBanaba  = "SELECT * FROM banabanodes ORDER
BY id DESC LIMIT 1";
      const floodRows = await new Promise((resolve, reject) => {
        connection.query(floodQueryBanaba , function (err,
floodRows, fields) {
          if (err) {
            reject(err);
            return;
          }
          resolve(floodRows);
        });
      });

      if (floodRows.length > 0) {
        const mostRecentFloodMeasure = floodRows[0];

        function getStatus(num) {
            if (num === -999.0) {
              return 'No Data Measured';
            } else if (num <= 93.75) {
              return 'Low';
            } else if (num > 93.75 && num <= 187.5) {
              return 'Normal';
            } else if (num > 187.5 && num <= 281.25 ) {
              return 'High';
            } else if (num > 281.25) {
              return 'Critical';
            } else {
              return 'Error';
            }
          }

        const banabaFloodData = {
            banabaBatteryPercentage:
mostRecentFloodMeasure.batteryPercentage,
            banabaWaterHeight:
mostRecentFloodMeasure.waterHeight,
            banabaWaterLevelStatus:
getStatus(mostRecentFloodMeasure.waterHeight),
            readingFloodTime: mostRecentFloodMeasure.readingTime
        };
```

```
        if (socket && socket.emit) {
          socket.emit('banabaFloodUpdate',
banabaFloodData);   /// Flood Update needs to be change
        }
      } else {
        console.log("No Banaba flood data found");
      }
    } catch (err) {
      console.log("An error occurred with the query:", err);
      if (socket && socket.emit) {
        socket.emit('floodError', 'Error executing query');
      }
    }
  };

  // Call the function immediately to send the initial weather
data
  sendFloodBanabaUpdate();

  // Schedule the function to be called every 60 seconds
  setInterval(sendFloodBanabaUpdate, 60000);
});

module.exports = { getFloodQueryBanaba };
```

3. *For Contact Page*

```
const nodemailer =require('nodemailer');
// create reusable transporter object using the default SMTP
transport
const send = async (req, res) => {
    const from = req.body.name;
    const message = req.body.message;
    const subject = req.body.subject;
    const email =req.body.email;

    const output = `
    <p>You have a new contact request</p>
    <h3>Contact Details</h3>
    <ul>
      Name of the Sender: ${from}
      Email of the Sender: ${email}
      Subject of the Message: ${subject}
    </ul>
    <h3>Message of the Sender</h3>
    <p>${message}</p>
  `;

  let transporter = nodemailer.createTransport({
    host: '',
    port:,
    secure: false, // true for 465, false for other ports
    auth: {
        user: '', // generated ethereal user
        pass: ''  // generated ethereal password
```

```
    },
    tls:{
      rejectUnauthorized:false
    },
    pool: true, // enable connection pooling
    maxConnections: 10, // maximum number of connections to keep
in the pool
    maxMessages: 100 // maximum number of messages to send
through a single connection
  });

  // setup email data with unicode symbols
  let mailOptions = {
      from: '"LIGTAS Website" <info@ligtas.tech>', // sender
address
      to: 'ligtas.system@gmail.com', // list of receivers
      subject: 'New Message', // Subject line
      html: output // html body
  };

  // send mail with defined transport object
  try {
    let info = await transporter.sendMail(mailOptions);
    console.log('Message sent: %s', info.messageId);
    res.redirect('/contact'); // redirect back to contact page
  } catch (error) {
    console.log(error);
    res.status(500).send('Error sending email');
  }
}

module.exports ={send};
```

4. *For Marigold Page*

```
const asyncHandler = require("express-async-handler");
const connection = require('../database');

const getFloodQueryMarigold = asyncHandler(async (io, socket) =>
{
  const sendFloodMarigoldUpdate = async () => {
    try {
      const floodQueryMarigold  = "SELECT * FROM marigoldnodes
ORDER BY id DESC LIMIT 1";
      const floodRows = await new Promise((resolve, reject) => {
        connection.query(floodQueryMarigold , function (err,
floodRows, fields) {
          if (err) {
            reject(err);
            return;
          }
          resolve(floodRows);
        });
      });

      if (floodRows.length > 0) {
```

```
            const mostRecentFloodMeasure = floodRows[0];

            function getStatus(num) {
              if (num === -999.0) {
                return 'No Data Measured';
              } else if (num <= 93.75) {
                return 'Low';
              } else if (num > 93.75 && num <= 187.5) {
                return 'Normal';
              } else if (num > 187.5 && num <= 281.25) {
                return 'High';
              } else if (num > 281.25) {
                return 'Critical';
              } else {
                return 'Error';
              }
            }

            const marigoldFloodData = {
                marigoldBatteryPercentage:
mostRecentFloodMeasure.batteryPercentage,
                marigoldWaterHeight:
mostRecentFloodMeasure.waterHeight,
                marigoldWaterLevelStatus:
getStatus(mostRecentFloodMeasure.waterHeight),
                readingFloodTime: mostRecentFloodMeasure.readingTime
            };

            if (socket && socket.emit) {
              socket.emit('marigoldFloodUpdate',
marigoldFloodData);   /// Flood Update needs to be change
            }
          } else {
            console.log("No Marigold flood data found");
          }
        } catch (err) {
          console.log("An error occurred with the query:", err);
          if (socket && socket.emit) {
            socket.emit('floodError', 'Error executing query');
          }
        }
      };

      // Call the function immediately to send the initial weather
data
      sendFloodMarigoldUpdate();

      // Schedule the function to be called every 60 seconds
      setInterval(sendFloodMarigoldUpdate, 60000);
    });

    module.exports = { getFloodQueryMarigold };
```

5. *For Weather Page*

```
    const asyncHandler = require("express-async-handler");
```

```
const connection = require('../database');

const getWeather = asyncHandler(async (io, socket) => {
  const sendWeatherUpdate = async () => {
    try {
      const weatherQuery = "SELECT * FROM weatherparameters ORDER
BY id DESC LIMIT 60";
      const weatherRows = await new Promise((resolve, reject) =>
{
        connection.query(weatherQuery, function (err,
weatherRows, fields) {
          if (err) {
            reject(err);
            return;
          }
          resolve(weatherRows);
        });
      });

      if (weatherRows.length > 0) {
        let sumRainIntensity = 0;
        let sumWindSpeed = 0;

        for (let i = 0; i < weatherRows.length; i++) {
          sumRainIntensity += weatherRows[i].rainIntensity;
          sumWindSpeed += weatherRows[i].windSpeed;
        }

        const averageWindSpeed = sumWindSpeed /
weatherRows.length;

        const mostRecentWeatherMeasure = weatherRows[0];
        const weatherData = {
          temp: mostRecentWeatherMeasure.temperature,
          humid: mostRecentWeatherMeasure.humidity,
          atmosphericPressure:
mostRecentWeatherMeasure.atmosphericPressure,
          windSpeed: averageWindSpeed,
          windDirection: mostRecentWeatherMeasure.windDirection,
          rainIntensity: sumRainIntensity,
          weatherBatteryPercentage:
mostRecentWeatherMeasure.BatteryPercentage,
          readingWeatherTime:
mostRecentWeatherMeasure.readingTime,
        };

        // Retrieve atmospheric pressure data from a separate
database
        const pressureQuery = "SELECT * FROM atmospheric ORDER BY
id DESC LIMIT 1";
        const pressureRows = await new Promise((resolve, reject)
=> {
          // Replace `pressureConnection` with your separate
database connection
          connection.query(pressureQuery, function (err,
pressureRows, fields) {
            if (err) {
```

```
                reject(err);
                return;
              }
              resolve(pressureRows);
            });
          });

          if (pressureRows.length > 0) {
            const mostRecentPressureMeasure = pressureRows[0];
            weatherData.atmosphericPressure =
mostRecentPressureMeasure.atmosphericPressure;
          } else {
            console.log("No atmospheric pressure data found");
          }

          if (socket && socket.emit) {
            socket.emit('weatherUpdate', weatherData);
          }
        } else {
          console.log("No weather data found");
        }
      } catch (err) {
        console.log("An error occurred with the query:", err);
        if (socket && socket.emit) {
          socket.emit('weatherError', 'Error executing query');
        }
      }
    };
    // Call the function immediately to send the initial weather
data
    sendWeatherUpdate();

    // Schedule the function to be called every 60 seconds
    setInterval(sendWeatherUpdate, 60000);
});


module.exports = { getWeather };
```

Routes

1. For Advisory Page

```
const express = require("express");
const router = express.Router();
const advisoryController = require("../controllers/advisory");

router.get("/", (req, res) => {
  // Call the weatherController method and pass any required data
  res.render("advisory"); // Render the weather.ejs template
});

module.exports = router;
```

2. For Banaba Page

```
const express = require("express");
```

```
const router = express.Router();
const banabaFloodController = require("../controllers/banaba");

router.get("/", (req, res) => {
  // Call the marigoldFloodController method and pass any
required data
  res.render("banaba"); // Render the weather.ejs template
});

module.exports = router;
```

3. For Contact Page

```
const {send} = require ("../controllers/cont");
const express = require('express');
const router = express.Router();

router.post('/send', send);

module.exports= router;
```

4. For Marigold Page

```
const express = require("express");
const router = express.Router();
const marigoldFloodController =
require("../controllers/marigold");

router.get("/", (req, res) => {
  // Call the marigoldFloodController method and pass any
required data
  res.render("marigold"); // Render the weather.ejs template
});

module.exports = router;
```

5. For Pages

```
const express = require('express');
const router = express.Router();

//route for index page
router.get('/', (req,res)=>{
    res.render('index') ;
  });

//route for about us page
router.get('/aboutUs', (req,res)=>{
    res.render('aboutUs');
  });

module.exports= router;
```

6. For Weather Page

```
const express = require("express");
const router = express.Router();
const weatherController = require("../controllers/weather");
```

```
      router.get("/", (req, res) => {
        // Call the weatherController method and pass any required data
        res.render("weather"); // Render the weather.ejs template
      });

      module.exports = router;
```

For model deployment,
main.py

```
# Import the Libraries
import mysql.connector
import tensorflow as tf
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pickle
import joblib
import json

from datetime import datetime, timedelta
from fastapi import FastAPI
from pydantic import BaseModel
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import *
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping,
ReduceLROnPlateau
from tensorflow.keras.losses import MeanSquaredError
from tensorflow.keras.metrics import RootMeanSquaredError
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import load_model

from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import mean_absolute_error, r2_score,
accuracy_score

# Create an app object
app = FastAPI()

# Create an index route
@app.get('/')
def index():
    return "Welcome to the Ligtas Prediction API!"


# Loading the saved model and scaler for preprocessing
ligtas_model = load_model('8var2_model.h5')
scaler = joblib.load('scaler.pkl')

# Connect to MySQL
cnx = mysql.connector.connect(
    host='localhost',
    user='ligtcatb_LIGTAS',
    password='Ligtas2023',
    database='ligtas_weatherFloodSystem'
)
```

```
# Create a cursor
cursor = cnx.cursor()

# Define the function to retrieve the latest data from MySQL
def retrieve_latest_data():
    query = "SELECT temperature, humidity, airPressure, windSpeed,
windDirection, rainIntensity, banabawaterHeight, marigoldwaterHeight
FROM ligtasparameters ORDER BY id DESC LIMIT 60"
    cursor.execute(query)
    rows = cursor.fetchall()
    return rows

# Prepare the input data from the retrieved rows
input_data = []
for row in retrieve_latest_data():
    input_data.append(list(row))

# Convert the input data to NumPy array
input_data = np.array(input_data)

# Adjust the index of the array and name it input_features
input_features = input_data[:, [6, 7, 5, 4, 3, 1, 2, 0]].tolist()

# Define the function to convert array to X and y
def array_to_X_y(data_array, window_size=8):
    X = []
    y = []
    for i in range(len(data_array) - window_size):
        row = data_array[i:i+window_size]
        X.append(row)
        label = data_array[i+window_size]
        y.append(label)
    return np.array(X), np.array(y)

# Define the preprocessing function
def preprocess(X):
    return scaler.transform(X.reshape(-1, X.shape[-
1])).reshape(X.shape)

# Define the post-processing function
def postprocess(X):
    original_shape = X.shape
    X = X.reshape(-1, original_shape[-1])
    X = scaler.inverse_transform(X)
    return X.reshape(original_shape)

# Define the function to generate predictions
def generate_prediction(input_list, model, scaler, steps=6,
interval=1):
    predictions = []
    for _ in range(steps):
        input_array = np.array(input_list).reshape(1, -1)
        X, _ = array_to_X_y(input_array)
        X = preprocess(X)
        prediction = model.predict(X)
        postprocessed = postprocess(prediction)
```

```
        predictions.append(postprocessed)

        # Update the input_list for the next prediction
        for _ in range(interval):
            input_list.append(postprocessed[0])
            input_list.pop(0)

    return predictions

# Define the function to update the input features every 1 hour
def update_input_features():
    while True:
        input_data = []
        for row in retrieve_latest_data():
            input_data.append(list(row))
        input_data = np.array(input_data)
        input_features = input_data[:, [6, 7, 5, 4, 3, 1, 2,
0]].tolist()
        time.sleep(3600)  # Wait for 1 hour before updating the input
features

# Start a separate thread to update the input features
import threading
update_thread = threading.Thread(target=update_input_features)
update_thread.start()

# Define the endpoint to retrieve the predictions
@app.get('/ligtas_prediction')
def ligtas_pred():
    predictions = generate_prediction(
        input_features, ligtas_model, scaler, steps=6, interval=1)
    result = [pred[0].tolist() for pred in predictions]
    return {"predictions": result}


if __name__ == "__main__":

    uvicorn.run(main:app, host="https:tupmanila.ligtas.tech", port=443)
```

Create a Database Table

For atmospheric,

```
CREATE TABLE atmospheric(
  id INT AUTO_INCREMENT PRIMARY KEY,
  atmosphericPressure FLOAT DEFAULT -999.0,
  readingTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

For Banaba nodes,

```
CREATE TABLE banabanodes(
  id INT AUTO_INCREMENT PRIMARY KEY,
  waterHeight FLOAT DEFAULT -999.0,
  waterLevel FLOAT DEFAULT -999.0,
  batteryPercentage FLOAT DEFAULT -999.0,
  readingTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

For LIGTAS parameters,

```
CREATE TABLE ligtasparameters (
  id INT AUTO_INCREMENT PRIMARY KEY,
  temperature FLOAT DEFAULT -999.0,
  humidity FLOAT DEFAULT -999.0,
  windSpeed FLOAT DEFAULT -999.0,
  atmosphericPressure FLOAT DEFAULT -999.0,
  windDirection FLOAT DEFAULT -999.0,
  rainIntensity FLOAT DEFAULT -999.0,
  banabawaterHeight (cm) FLOAT DEFAULT -999.0,
  marigoldwaterHeight (cm)FLOAT DEFAULT -999.0,
  Date TimeTIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

For Marigold nodes,

```
CREATE TABLE marigoldnodes(
  id INT AUTO_INCREMENT PRIMARY KEY,
  waterHeight FLOAT DEFAULT -999.0,
  waterLevel FLOAT DEFAULT -999.0,
  batteryPercentage FLOAT DEFAULT -999.0,
  readingTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

For Users,

```
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(128),
  email VARCHAR(128),
  password VARCHAR(64)
);
```

For Weather parameters,

```
CREATE TABLE weatherparameters (
  id INT AUTO_INCREMENT PRIMARY KEY,
  temperature FLOAT DEFAULT -999.0,
  humidity FLOAT DEFAULT -999.0,
  windSpeed FLOAT DEFAULT -999.0,
  windDirection FLOAT DEFAULT -999.0,
  rainIntensity FLOAT DEFAULT -999.0,
  BatteryPercentage FLOAT DEFAULT -999.0,
  readingTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

database.js

```
const mysql = require('mysql');

// to create connection
const connection = mysql.createConnection({
  host: '',
  user: '',
  password: ‘’,
  database: ''
});

// connect to database
connection.connect(function(err) {
```

```
  if (err) {
      console.log(err.code);
      console.log(err.fatal);
  }
});
module.exports = connection;
```

Server-side of the Node JS Application

```
const express = require("express");
const mysql = require('mysql');
const http = require('http');
const mqtt = require('mqtt');
const webPush = require('web-push');
const errorHandler = require("./middleware/errorHandler");
const app = express();
const session = require("express-session");
const path = require("path");
const ejs = require("ejs");
const cookieParser = require("cookie-parser");
const bodyParser = require("body-parser");
const flash = require('express-flash');
const server = http.createServer(app);
const io = require('socket.io')(server);
const weatherController = require("./controllers/weather");
const weatherRoutes = require('./routes/weather');
const marigoldController = require("./controllers/marigold");
const marigoldRoutes = require('./routes/marigold');
const banabaController = require("./controllers/banaba");
const banabaRoutes = require('./routes/banaba');
const contRoutes = require('./routes/cont');
const connection = require('./database');
const advisoryController = require("./controllers/advisory");
const advisoryRoutes = require('./routes/advisory');
const axios = require('axios');

const port = 443;

const publicDirectoryPath = path.join(__dirname, "public");
app.use(express.static(publicDirectoryPath));
// Customize the server to serve the folder
app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());
app.use(cookieParser());
app.use(flash());

// Parse the URL-encoded bodies (as sent by HTML forms)
app.use(express.urlencoded({ extended: false }));
// Parse JSON bodies (as sent by API clients). Provide a parser which
helps to pass the data stream from the client to the server
app.use(express.json());
// app.use as middleware
app.use(errorHandler);
app.use(
  session({
    secret: "webslesson",
    resave: true,
    saveUninitialized: true,
```

```
  })
);

io.on('connection', (socket) => {
  weatherController.getWeather(io, socket);
  marigoldController.getFloodQueryMarigold(io, socket);
  banabaController.getFloodQueryBanaba(io, socket);
  advisoryController.getAdvisory(io, socket);

  socket.on("connect_timeout", (timeout) => {
    console.error("Socket.IO connection timeout:", timeout);
  });

});
// Make the 'io' object accessible to the routes
app.set("socketio", io);

//setting view engine to ejs
app.set("view engine", "ejs");
app.set('views',path.join(__dirname+'/views/'))

//Define Routes
app.use("/", require("./routes/pages"));
// Set up your routes and controllers here
app.use('/weather', weatherRoutes);
app.use("/banaba", banabaRoutes);
app.use("/marigold", marigoldRoutes);
app.use("/cont", contRoutes);
app.use("/advisory",advisoryRoutes);

// Emit weather updates every 60 seconds (for demonstration purposes)
setInterval(async () => {
  const weatherData = await weatherController.getWeather();
  io.emit('weatherUpdate', weatherData);

  const marigoldFloodData = await
marigoldController.getFloodQueryMarigold();
  io.emit('marigoldFloodUpdate', marigoldFloodData);

  const banabaFloodData = await banabaController.getFloodQueryBanaba();
  io.emit('banabaFloodUpdate', banabaFloodData);

}, 60000);

// Emit weather updates every 5 minutes
setInterval(async () => {
  const advisoryData = await advisoryController.getAdvisory();
  io.emit('advisoryUpdate', advisoryData);

}, 60000);

app.get('/data', (req, res) => {
  // Query the database for the data
  connection.query("SELECT temperature, DATE_FORMAT(readingTime,
'%h:%i') AS readingTime FROM weatherparameters ORDER BY id DESC LIMIT
20", (err, results) => {
    if (err) {
```

```
      console.error('Error querying database:', err);
      res.status(500).json({ error: 'Failed to fetch data from the
database' });
      return;
    }

    // Extract the necessary data from the query results
    const xlabels = results.map((row) => row.readingTime).reverse();
    const temperatures = results.map((row) =>
row.temperature).reverse();

    // Send the data as a JSON response
    res.json({ xlabels, temperatures });
  });
});

// to create the mqtt client and connection
const client =
mqtt.connect('mqtt://au1.cloud.thethings.industries:1883', {
  username: '',
  password: ''
});

client.subscribeToTopics = () => {
    const topics = ['v3/ligtas-
system@packetworx/devices/weathersystem/up',
                    'v3/ligtas-system@packetworx/devices/floodnode-
02/up',
                    'v3/ligtas-system@packetworx/devices/floodnode-
03/up'];
    client.on('connect', () => {
      console.log('Connected to the MQTT Broker')
      client.subscribe(topics, () => {
        console.log(`Subscribe to topic '${topics}'`)
      })
    })

    client.on('message', (topics, message) => {
      console.log(`Received message from topic ${topics}`);
      if (topics === 'v3/ligtas-
system@packetworx/devices/weathersystem/up') {
        // perform logic for weather system topic
        const data = JSON.parse(message.toString());
        // Extract the decoded payload from the `data` object
        const decodedPayload = data.uplink_message.decoded_payload;
        // Assigning the decoded payload
        const humidity = decodedPayload.Humidity;
        const temperature = decodedPayload.Temperature;
        const windDirection = decodedPayload.WindDirection;
        const windSpeed = decodedPayload.WindSpeed;
        const rainIntensity = decodedPayload.RainIntensity;
        const batteryPercentage = decodedPayload.BatteryPercent;
        // Insert the values from the weather system data into the
weatherparameters database
        const weatherDataSql = `INSERT INTO weatherparameters
(humidity, temperature, windDirection, windSpeed, rainIntensity,
BatteryPercentage) VALUES (?, ?, ?, ?, ?, ?)`;
```

```
        const weatherDataValues = [humidity, temperature,
windDirection, windSpeed, rainIntensity, batteryPercentage];
        // Insert the values from the weather system data into the
ligtasparameters database
        const dataSql = `INSERT INTO ligtasparameters (humidity,
temperature, windDirection, windSpeed, rainIntensity) VALUES (?, ?, ?,
?, ?)`;
        const dataValues = [humidity, temperature, windDirection,
windSpeed, rainIntensity];
        connection.query(weatherDataSql, weatherDataValues, (error,
result) => {
          if (error) {
            console.error('Error inserting data into MySQL database: ',
error);
          } else {
            // execute second SQL statement to insert data into
wind_data table
            connection.query(dataSql, dataValues, (error, result) => {
              if (error) {
                console.error('Error inserting data into MySQL
database: ', error);
              } else {
                console.log('Data inserted successfully into MySQL
database');
              }
            });
          }
        });

      } else if (topics === 'v3/ligtas-
system@packetworx/devices/floodnode-02/up') {
        // perform logic for floodnode-02 topic banaba
        const data = JSON.parse(message.toString());
         // Extract the decoded payload from the `data` object
        const decodedPayload = data.uplink_message.decoded_payload;
        // Assigning the decoded payload
        const batteryPercentage = decodedPayload.BatteryPercent;
        const waterHeight = decodedPayload.WaterHeight;
        const waterLevel = decodedPayload.WaterLevel;
        // Insert the values from the flood node data into the
floodnodes database
        const banabaDataSql = `INSERT INTO banabanodes
(batteryPercentage, waterHeight, waterLevel) VALUES (?, ?, ?)`;
        const banabaDataValues = [batteryPercentage, waterHeight,
waterLevel];
         // Insert the values from the flood node 2 into the
ligtasparameters database
        const dataSql = `INSERT INTO ligtasparameters
(banabaWaterHeight) VALUES (?)`;
        const dataValues = [waterHeight];
        connection.query(banabaDataSql, banabaDataValues, (error,
result) => {
          if (error) {
            console.error('Error inserting data into MySQL database: ',
error);
          } else {
            connection.query(dataSql, dataValues, (error, result) => {
```

```
              if (error) {
                console.error('Error inserting data into MySQL
database: ', error);
              } else {
                console.log('Data inserted successfully into MySQL
database');
              }
            });
          }
        });
      } else if (topics === 'v3/ligtas-
system@packetworx/devices/floodnode-03/up') {
        // perform logic for floodnode-03 topic marigold
        const data = JSON.parse(message.toString());
        // Extract the decoded payload from the `data` object
        const decodedPayload = data.uplink_message.decoded_payload;
        // Assigning the decoded payload
        const batteryPercentage = decodedPayload.BatteryPercent;
        const atmosphericPressure = decodedPayload.AtmosphericPressure;
        const waterHeight = decodedPayload.WaterHeight;
        const waterLevel = decodedPayload.WaterLevel;
        // Insert the values from the flood node data into the
floodnodes database
        const marigoldDataSql = `INSERT INTO marigoldnodes
(batteryPercentage, waterHeight, waterLevel) VALUES (?, ?, ?)`;
        const marigoldDataValues = [batteryPercentage, waterHeight,
waterLevel];
        // Insert the values from the AtmosphericPressure into the
weatherparameters database
        const windDataSql = `INSERT INTO
atmospheric(atmosphericPressure) VALUES (?)`;
        const windDataValues = [atmosphericPressure];
        // Insert the values from the flood node 3 into the
ligtasparameters database
        const dataSql = `INSERT INTO ligtasparameters
(atmosphericPressure,marigoldWaterHeight) VALUES (?,?)`;
        const dataValues = [atmosphericPressure, waterHeight];
        connection.query(marigoldDataSql, marigoldDataValues, (error,
result) => {
          if (error) {
            console.error('Error inserting data into MySQL database: ',
error);
          } else {
          connection.query(windDataSql, windDataValues, (error, result)
=> {
            if (error) {
              console.error('Error inserting data into MySQL database:
', error);
            } else {
            connection.query(dataSql, dataValues, (error, result) => {
              if (error) {
                console.error('Error inserting data into MySQL
database: ', error);
              } else {
                console.log('Data inserted successfully into MySQL
database');
              }
```

```
            });
            }
          });
          }
        });
      };
    });
  };
client.subscribeToTopics();

// Function to retrieve the forecasted data
async function getForecastedData() {
  try {
    const response = await
axios.get('http://model.ligtas.tech/ligtas_prediction');
    const data = response.data;

    // Process the data
    const forecasts = data.predictions;
    const currentTime = new Date();
    const forecastedData = [];

    for (let i = 0; i < forecasts.length; i++) {
      const time = new Date(currentTime.getTime() + (i + 1) * 60 * 60 *
1000); // Add 1 hour for each forecast
      const forecast = forecasts[i];
      const forecastData = {
        time: time.toLocaleString(), // Convert the time to a readable
format
        temperature: forecast[0],
        humidity: forecast[1],
        windSpeed: forecast[2],
        windDirection: forecast[3],
        rainIntensity: forecast[4],
        banabawaterLevel: forecast[5],
        marigoldwaterLevel: forecast[6],
        airPressure: forecast[7]
      };
      forecastedData.push(forecastData);
    }

    // Use the forecastedData in your Node.js application as needed
    console.log(forecastedData);
  } catch (error) {
    console.error('Error retrieving forecasted data:', error);
  }
}

// Call the function to get the forecasted data
getForecastedData();

server.listen(port, () => {
  console.log(`Server running on ${port}`);
});
```

package.json

```
{
  "name": "final-website",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js",
    "dev": "nodemon server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@tensorflow/tfjs": "^4.6.0",
    "axios": "^1.4.0",
    "bcrypt": "^5.1.0",
    "body-parser": "^1.20.2",
    "chart.js": "^4.3.0",
    "cookie-parser": "^1.4.6",
    "cors": "^2.8.5",
    "dotenv": "^16.0.3",
    "ejs": "^3.1.9",
    "express": "^4.18.2",
    "express-async-handler": "^1.2.0",
    "express-flash": "^0.0.2",
    "express-session": "^1.17.3",
    "http": "^0.0.1-security",
    "jsonwebtoken": "^9.0.0",
    "mqtt": "^4.3.7",
    "mysql": "^2.18.1",
    "nodemailer": "^6.9.2",
    "nodemon": "^2.0.22",
    "socket.io": "^4.6.1",
    "socket.io-client": "^4.6.1",
    "web-push": "^3.6.1"
  }
}
```

Appendix D
Deployment Data

id	T (degC)	rh (%)	p (mbar)	wd (deg)	wv (km/hr)	rain (mm)	banabawaterHeight (cm)	marigoldwaterHeight (cm)	Date Time
1	28.4	67.8	1004.08	232	0.21	2.47273	144	48	5/18/2023 4:02
2	28.4	67.7	1004.12	310	3.73	2.16364	144	48	5/18/2023 4:02
3	28.4	67.8	1004.23	311	0.29	2.47273	144	48	5/18/2023 4:03
4	28.4	67.8	1004.28	231	0.21	2.16364	144	48	5/18/2023 4:03
5	28.4	67.7	1004.24	232	0.21	3.70909	144	48	5/18/2023 4:04
6	28.4	67.8	1004.33	310	0.29	3.09091	144	48	5/18/2023 4:05
8	36.7	56.8	1004.39	353	21.77	8.96364	144	48	5/16/2023 13:47
9	36.5	56.6	1004.42	353	24.12	10.5091	144	49	5/16/2023 13:47
10	36.5	56.6	1004.49	353	23.97	9.58182	144	49	5/16/2023 13:48
11	36.3	56.2	1004.5	353	26.02	9.58182	144	49	5/16/2023 13:48
12	36.3	56.6	1004.49	353	21.55	10.5091	145	49	5/16/2023 13:49
13	36.3	57.5	1004.52	305	26.97	11.4364	145	51	5/16/2023 13:49
14	36.5	58	1004.6	307	18.03	12.3636	145	54	5/16/2023 13:50
15	36.6	59.1	1004.48	353	25.51	10.5091	145	54	5/16/2023 13:50
16	36.7	60.3	1004.48	353	23.75	9.58182	145	51	5/16/2023 13:51
17	36.8	61.5	1004.48	353	15.9	11.1273	145	51	5/16/2023 13:51
18	36.8	62.7	1004.48	305	18.18	11.4364	145	51	5/16/2023 13:52
19	36.9	62	1004.43	353	14.73	12.0545	145	54	5/16/2023 13:52
20	37.1	63.4	1004.32	353	25.95	9.89091	145	56	5/16/2023 13:53
21	37.2	63.6	1004.22	353	23.68	10.2	145	56	5/16/2023 13:53
22	37.3	64.4	1004.22	353	21.84	11.7455	146	55	5/16/2023 13:54
23	37.4	64	1004.24	353	20.16	10.5091	146	54	5/16/2023 13:54
24	37.4	64	1004.23	353	15.54	9.89091	146	56	5/16/2023 13:55
25	37.6	65	1004.19	353	26.83	10.8182	146	54	5/16/2023 13:55
26	37.7	66.5	1004.15	353	24.56	10.2	146	56	5/16/2023 13:56
27	37.9	67.8	1004.23	353	25.43	12.6727	146	56	5/16/2023 13:56
28	37.9	67.8	1004.23	353	17.96	11.4364	146	55	5/16/2023 13:57

29	38	67	1004.16	353	24.92	11.7455	146	54	5/16/2023 13:57
30	34.1	75.6	1004.11	353	16.86	8.03637	146	56	5/16/2023 15:26
31	34.1	75.7	1004.11	353	23.46	12.9818	146	52	5/16/2023 15:30
32	33.9	75.6	1004.11	353	28.37	9.89091	147	52	5/16/2023 15:31
33	33.7	75.2	1004.13	353	21.7	12.0545	147	52	5/16/2023 15:34
34	33.6	76.6	1004.19	353	22.87	9.58182	147	52	5/16/2023 15:37
35	33.5	76.4	1004.23	353	14.73	8.96364	147	52	5/16/2023 15:42
36	28.4	67.7	1004.3	310	0.36	1.54545	147	52	5/18/2023 4:05
37	33.5	76.3	1004.36	353	13.34	7.72727	147	49	5/16/2023 15:43
38	33.5	76.2	1004.37	353	25.21	8.34546	147	49	5/16/2023 15:43
39	33.5	76.2	1004.41	76	9.38	10.2	147	49	5/16/2023 15:45
40	33.3	76	1004.5	353	17.52	8.96364	148	51	5/16/2023 15:46
41	33.2	77.8	1004.56	353	17.44	10.8182	148	50	5/16/2023 15:50
42	33.2	77.6	1004.67	353	22.43	11.4364	148	51	5/16/2023 15:51
43	33.2	77.6	1004.79	353	16.49	10.5091	148	52	5/16/2023 15:51
44	33.2	77.6	1004.85	353	26.39	9.89091	148	56	5/16/2023 15:52
45	33.2	77.6	1004.85	353	29.39	9.58182	148	56	5/16/2023 15:52
46	33.2	77.6	1004.91	353	26.31	9.58182	148	54	5/16/2023 15:53
47	33.2	77.7	1004.99	353	22.36	9.58182	148	54	5/16/2023 15:53
48	33.1	77.7	1005.1	353	21.48	8.65455	148	51	5/16/2023 15:54
49	33.1	77.7	1005.13	353	18.84	9.58182	148	51	5/16/2023 15:54
50	33.1	77.7	1005.2	353	16.27	9.58182	149	51	5/16/2023 15:55
51	33	77.7	1005.32	353	17.96	10.5091	149	48	5/16/2023 15:55
52	33	77.6	1005.43	353	19.86	10.2	149	48	5/16/2023 15:56
53	33	77.5	1005.46	353	26.09	8.34546	149	48	5/16/2023 15:56
54	33	77.5	1005.47	353	22.43	9.27273	149	49	5/16/2023 15:57
55	33	77.7	1005.58	353	21.18	8.65455	149	49	5/16/2023 15:57
56	33	77.7	1005.69	353	20.6	11.1273	149	49	5/16/2023 15:58

57	33	77.7	1005.71	353	15.76	9.27273	149	50	5/16/2023 15:58
58	32.9	77.6	1005.71	305	11.58	9.27273	150	50	5/16/2023 15:59
59	32.9	77.6	1005.72	353	10.41	8.65455	150	50	5/16/2023 15:59
60	32.9	77.4	1005.79	307	17	9.27273	150	50	5/16/2023 16:00
61	32.9	77.4	1005.84	353	16.86	8.96364	150	54	5/16/2023 16:00
62	32.9	77.4	1005.85	353	21.92	9.27273	150	54	5/16/2023 16:01
63	32.9	77.4	1005.96	353	17.96	9.27273	150	53	5/16/2023 16:01
64	32.9	77.4	1006.05	353	17.52	9.58182	151	53	5/16/2023 16:02
65	32.9	77.2	1006.05	308	27.56	9.27273	151	52	5/16/2023 16:02
66	32.9	77.2	1006.1	353	15.32	9.27273	151	51	5/16/2023 16:03
67	32.9	77.2	1006.12	353	23.46	9.58182	151	51	5/16/2023 16:03
68	32.9	77.2	1006.18	353	20.6	8.34546	151	51	5/16/2023 16:04
69	32.6	77.2	1006.24	353	27.49	10.8182	152	51	5/16/2023 16:09
70	32.6	77	1006.26	353	23.46	11.4364	152	50	5/16/2023 16:10
71	32.6	77	1006.26	353	24.85	11.4364	152	50	5/16/2023 16:10
72	32.5	77	1006.26	353	22.36	8.96364	152	50	5/16/2023 16:11
73	32.5	77	1006.24	353	19.57	10.8182	153	49	5/16/2023 16:11
74	32.5	77	1006.19	353	24.12	10.8182	153	49	5/16/2023 16:12
75	32.5	77	1006.36	353	18.32	10.5091	154	49	5/16/2023 16:12
76	32.5	77	1006.4	306	17.88	11.1273	154	48	5/16/2023 16:13
77	32.5	77	1006.49	353	15.98	9.58182	154	48	5/16/2023 16:13
78	32.5	77	1006.64	353	16.27	9.58182	154	48	5/16/2023 16:14
79	32.5	78.6	1006.69	353	19.42	10.5091	154	49	5/16/2023 16:15
80	32.5	78.6	1006.76	353	12.82	10.5091	154	49	5/16/2023 16:15
81	32.5	78.5	1006.84	308	17.44	10.5091	155	51	5/16/2023 16:16
82	32.5	78.3	1006.9	353	32.69	9.58182	155	51	5/16/2023 16:16
83	32.5	78.3	1006.87	353	24.04	10.5091	156	51	5/16/2023 16:17
84	32.5	78.3	1006.99	353	24.63	12.3636	156	51	5/16/2023 16:17

85	32.5	78.3	1007.05	307	19.28	11.4364	156	49	5/16/2023 16:18
86	32.5	78.3	1007.11	353	10.33	11.4364	156	49	5/16/2023 16:18
87	32.5	78.3	1007.27	353	12.17	11.1273	157	52	5/16/2023 16:19
88	32.5	78.1	1007.23	353	11.43	8.65455	157	52	5/16/2023 16:19
89	32.5	78.1	1007.28	353	18.62	10.8182	157	52	5/16/2023 16:20
90	32.5	78	1007.3	353	11.29	8.96364	158	50	5/16/2023 16:20
91	32.6	78	1007.38	353	21.48	8.65455	158	50	5/16/2023 16:21
92	32.6	78	1007.45	353	26.24	7.41818	158	52	5/16/2023 16:21
93	32.5	78	1007.54	353	15.61	8.03637	159	52	5/16/2023 16:22
94	32.5	78	1007.54	353	22.06	9.58182	158	51	5/16/2023 16:22
95	32.5	78	1007.49	353	21.84	8.03637	159	51	5/16/2023 16:23
96	32.5	78	1007.42	353	18.76	10.2	159	51	5/16/2023 16:23
97	32.5	78	1007.41	307	18.69	9.27273	159	48	5/16/2023 16:24
98	32.5	78	1007.42	353	15.1	8.96364	159	48	5/16/2023 16:25
99	32.5	78	1007.44	353	18.47	10.2	159	48	5/16/2023 16:25
100	32.5	78	1007.47	353	11.58	9.27273	159	48	5/16/2023 16:26
101	32.5	78	1007.45	353	21.04	9.58182	160	48	5/16/2023 16:26
102	32.4	78	1007.56	353	17.81	9.89091	160	48	5/16/2023 16:27
103	32.4	78.6	1007.61	353	14	8.03637	160	48	5/16/2023 16:27
104	32.4	79.7	1007.66	353	9.6	8.65455	161	49	5/16/2023 16:28
105	32.4	79.7	1007.68	353	16.78	7.41818	160	49	5/16/2023 16:28
106	32.5	79.6	1007.68	353	21.84	10.8182	161	49	5/16/2023 16:29
107	32.5	79.6	1007.69	307	27.63	10.8182	161	49	5/16/2023 16:29
108	32.5	79.6	1007.75	353	8.35	11.1273	161	51	5/16/2023 16:30
109	32.5	79.6	1007.8	353	16.2	12.3636	161	54	5/16/2023 16:30
110	32.5	79.5	1007.89	353	19.5	10.2	161	54	5/16/2023 16:31
111	32.5	79.5	1007.82	353	24.56	9.58182	162	51	5/16/2023 16:31
112	32.5	79.6	1007.81	307	23.46	8.96364	162	51	5/16/2023 16:32

113	32.4	79.7	1007.84	353	17.52	11.1273	162	51	5/16/2023 16:32
114	32.4	79.6	1007.87	353	15.76	11.1273	162	54	5/16/2023 16:33
115	32.4	79.7	1007.92	353	24.78	10.2	162	56	5/16/2023 16:33
116	32.4	79.7	1007.91	353	30.93	11.7455	162	56	5/16/2023 16:34
117	32.4	79.6	1008	353	23.68	10.5091	162	55	5/16/2023 16:34
118	32.4	79.7	1008.04	353	16.93	9.58182	162	54	5/16/2023 16:35
119	32.4	79.6	1008.06	353	26.9	9.89091	163	56	5/16/2023 16:35
120	32.4	79.7	1008.1	353	18.47	10.8182	163	54	5/16/2023 16:36
121	32.4	79.7	1008.13	353	11.8	10.8182	163	56	5/16/2023 16:36
122	32.3	79.6	1008.05	307	10.33	8.34546	163	56	5/16/2023 16:37
123	32.3	79.6	1008.04	353	18.03	9.58182	163	55	5/16/2023 16:37
124	32.3	79.4	1008.04	353	17.44	9.27273	163	54	5/16/2023 16:38
125	32.3	79.3	1008.03	353	12.17	10.2	163	56	5/16/2023 16:38
126	32.3	79.3	1008.05	353	8.43	9.89091	163	52	5/16/2023 16:39
127	32.3	79.3	1008.17	353	8.06	8.96364	163	52	5/16/2023 16:39
128	32.3	79.3	1008.25	353	11.51	9.58182	163	52	5/16/2023 16:40
129	32.3	79.3	1008.3	353	10.11	10.5091	163	52	5/16/2023 16:40
130	32.4	79.2	1008.38	308	18.54	10.2	163	52	5/16/2023 16:41
131	32.4	79.2	1008.46	353	15.76	11.4364	163	52	5/16/2023 16:41
132	32.4	79.2	1008.59	353	14.73	9.89091	163	49	5/16/2023 16:42
133	32.4	79.2	1008.63	353	15.46	9.89091	164	49	5/16/2023 16:42
134	32.4	79.2	1008.71	353	21.77	10.8182	163	49	5/16/2023 16:43
135	32.4	79.2	1008.84	353	8.13	10.2	164	51	5/16/2023 16:43
136	32.4	79.2	1008.92	307	9.6	10.2	163	50	5/16/2023 16:44
137	32.5	79.2	1008.92	353	15.83	9.58182	164	51	5/16/2023 16:44
138	32.5	79.2	1009.05	353	17	9.27273	163	52	5/16/2023 16:45
139	32.5	79.1	1009.17	353	21.77	9.27273	163	56	5/16/2023 16:45
140	32.4	79.1	1009.26	307	15.46	8.65455	163	56	5/16/2023 16:46

141	32.5	79.2	1009.28	353	14	11.7455	164	54	5/16/2023 16:46
142	32.5	79	1009.27	353	7.25	8.34546	163	54	5/16/2023 16:47
143	32.5	79	1009.26	353	20.38	10.8182	163	51	5/16/2023 16:47
144	32.5	79	1009.3	353	23.16	10.2	163	51	5/16/2023 16:48
145	32.5	79.1	1009.4	353	8.35	9.27273	163	51	5/16/2023 16:48
146	32.5	79.1	1009.51	307	14.29	9.58182	163	48	5/16/2023 16:49
147	32.5	79.1	1009.53	307	16.86	9.58182	163	48	5/16/2023 16:49
148	32.4	79	1009.53	353	14.07	9.27273	163	48	5/16/2023 16:50
149	32.4	79	1009.65	353	17.08	8.34546	163	49	5/16/2023 16:50
150	32.4	79	1009.72	353	20.01	9.58182	163	49	5/16/2023 16:51
151	32.4	79	1009.78	353	23.53	9.27273	163	49	5/16/2023 16:51
152	32.4	79.2	1009.85	353	19.13	9.27273	162	50	5/16/2023 16:52
153	32.4	79	1009.88	353	12.39	7.72727	162	50	5/16/2023 16:52
154	32.4	79	1009.95	307	13.78	9.89091	162	50	5/16/2023 16:53
155	32.4	79	1010.02	353	15.68	8.03637	162	50	5/16/2023 16:53
156	32.4	79	1010.03	353	15.61	9.27273	162	54	5/16/2023 16:54
157	32.4	79	1010.1	353	12.39	9.89091	162	54	5/16/2023 16:54
158	32.4	79.5	1010.17	353	20.96	9.27273	162	53	5/16/2023 16:55
159	32.4	80.2	1010.13	353	10.99	9.27273	162	53	5/16/2023 16:55
160	32.4	80.5	1010.13	353	13.78	7.10909	162	52	5/16/2023 16:56
161	32.4	80.5	1010.05	353	18.25	8.03637	161	51	5/16/2023 16:56
162	32.4	80.5	1010.05	307	18.47	10.5091	161	51	5/16/2023 16:57
163	32.4	80.5	1010.04	353	16.93	8.34546	161	51	5/16/2023 16:57
164	32.4	80.5	1010.01	353	19.94	9.89091	161	51	5/16/2023 16:58
165	32.4	80.5	1010	353	9.53	8.03637	161	50	5/16/2023 16:58
166	32.4	80.5	1009.97	353	12.6	10.5091	161	50	5/16/2023 16:59
167	32.4	80.5	1010.01	353	15.83	9.89091	160	50	5/16/2023 16:59
168	32.4	80.5	1009.92	353	8.87	8.96364	160	49	5/16/2023 17:00

169	32.4	80.5	1009.8	308	9.53	11.1273	160	49	5/16/2023 17:00
170	32.4	80.5	1009.74	308	16.78	9.58182	160	49	5/16/2023 17:01
171	32.4	80.5	1009.82	353	10.04	8.34546	160	48	5/16/2023 17:01
172	32.4	80.5	1009.89	353	10.33	11.4364	160	48	5/16/2023 17:02
173	32.4	80.5	1009.76	307	25.51	10.8182	160	48	5/16/2023 17:02
174	32.4	80.5	1009.63	353	24.34	9.89091	159	49	5/16/2023 17:03
175	32.3	80.3	1009.62	353	10.26	7.72727	159	49	5/16/2023 17:03
176	32.3	80.3	1009.65	353	13.19	10.5091	159	51	5/16/2023 17:04
177	32.3	80.3	1009.64	353	22.72	10.5091	158	51	5/16/2023 17:04
178	32.3	80.3	1009.66	353	13.26	9.27273	159	51	5/16/2023 17:05
179	32.3	80.3	1009.64	353	8.28	9.27273	158	51	5/16/2023 17:05
180	32.3	80.3	1009.6	353	8.21	10.5091	158	49	5/16/2023 17:06
181	32.3	80.3	1009.66	353	9.45	8.65455	158	49	5/16/2023 17:06
182	32.3	80.1	1009.69	353	18.03	9.58182	158	52	5/16/2023 17:07
183	32.3	80	1009.76	353	10.48	10.8182	158	52	5/16/2023 17:07
184	32.3	80	1009.76	353	8.06	9.89091	157	52	5/16/2023 17:08
185	32.3	80	1009.76	353	17.66	9.58182	157	50	5/16/2023 17:08
186	32.3	80	1009.82	353	16.56	11.7455	157	50	5/16/2023 17:09
187	32.3	80	1009.76	353	17.15	9.58182	157	52	5/16/2023 17:09
188	32.3	80	1009.86	353	11.51	8.96364	157	52	5/16/2023 17:10
189	32.3	80	1009.9	353	10.77	8.03637	156	51	5/16/2023 17:10
190	32.3	80	1009.99	353	12.02	8.65455	157	51	5/16/2023 17:11
191	32.4	80	1010.01	353	17	9.89091	156	51	5/16/2023 17:11
192	32.3	80.2	1010.12	353	22.87	9.27273	156	48	5/16/2023 17:12
193	32.4	80	1010.14	353	11.36	8.96364	156	48	5/16/2023 17:12
194	32.3	80	1010.13	353	15.76	9.89091	156	48	5/16/2023 17:13
195	32.3	80	1010.15	353	15.17	8.65455	155	48	5/16/2023 17:13
196	32.3	80	1010.13	353	11.95	8.03637	155	48	5/16/2023 17:14

197	32.3	80	1010.17	353	16.86	8.96364	155	48	5/16/2023 17:14
198	32.3	80	1010.15	353	8.21	9.27273	155	48	5/16/2023 17:15
199	32.3	80	1010.13	353	15.1	10.5091	155	49	5/16/2023 17:15
200	32.3	80	1010.13	353	10.48	9.27273	155	49	5/16/2023 17:16
201	32.3	80.8	1010.14	353	15.98	9.89091	154	49	5/16/2023 17:16
202	32.3	80	1010.26	353	18.69	9.89091	154	49	5/16/2023 17:17
203	32.3	80.3	1010.34	353	15.32	9.27273	154	51	5/16/2023 17:17
204	32.3	80	1010.29	353	11.07	9.89091	154	54	5/16/2023 17:18
205	32.3	80	1010.33	353	11.14	9.89091	154	54	5/16/2023 17:18
206	32.3	80.5	1010.33	0	11.14	9.89091	154	51	5/16/2023 17:19
207	32.3	80.6	1010.32	353	12.68	8.34546	153	51	5/16/2023 17:19
208	32.3	80	1010.38	353	14.58	9.58182	153	51	5/16/2023 17:20
209	32.3	80	1010.39	353	15.1	10.8182	153	54	5/16/2023 17:20
210	32.3	81.6	1010.49	353	14.51	8.34546	153	56	5/16/2023 17:21
211	32.3	80.5	1010.47	353	19.79	7.41818	153	56	5/16/2023 17:21
212	32.3	80	1010.37	353	16.12	9.58182	153	55	5/16/2023 17:22
213	32.3	80.4	1010.31	353	19.35	8.96364	152	54	5/16/2023 17:22
214	32.3	80	1010.4	353	11.14	11.4364	152	56	5/16/2023 17:23
215	32.3	80	1010.37	353	15.98	10.2	152	54	5/16/2023 17:23
216	32.3	80	1010.33	308	9.89	10.5091	152	56	5/16/2023 17:24
217	32.3	80	1010.35	353	18.98	9.58182	152	56	5/16/2023 17:24
218	32.2	80	1010.42	353	21.62	9.58182	152	55	5/16/2023 17:25
219	32.2	80	1010.52	307	22.5	10.5091	151	54	5/16/2023 17:25
220	32.1	80	1010.5	353	19.86	8.34546	151	56	5/16/2023 17:26
221	32.2	80	1010.57	353	18.1	7.72727	151	52	5/16/2023 17:26
222	32.1	80	1010.54	353	16.2	10.2	151	52	5/16/2023 17:27
223	32.1	80	1010.47	353	25.43	9.89091	151	52	5/16/2023 17:27
224	32.1	80	1010.38	353	22.5	10.2	151	52	5/16/2023 17:28

225	32.1	80	1010.35	306	18.84	11.7455	151	52	5/16/2023 17:28
226	32	80	1010.36	353	13.78	10.2	150	52	5/16/2023 17:29
227	32	80	1010.36	353	21.84	11.1273	150	49	5/16/2023 17:29
228	32	80	1010.42	353	28.73	11.7455	150	49	5/16/2023 17:30
229	32	80	1010.41	353	16.86	11.4364	150	49	5/16/2023 17:30
230	31.9	81.6	1010.39	353	23.46	10.8182	150	51	5/16/2023 17:31
231	31.9	81.6	1010.35	353	17.08	10.2	149	50	5/16/2023 17:31
232	31.9	81.6	1010.37	307	19.64	9.89091	149	51	5/16/2023 17:32
233	31.9	81.6	1010.3	307	18.91	9.89091	149	52	5/16/2023 17:32
234	31.9	81.6	1010.12	353	14.14	10.2	149	56	5/16/2023 17:33
235	31.9	81.4	1010.14	353	16.42	11.1273	149	56	5/16/2023 17:33
236	31.9	81.3	1010.03	307	20.16	10.8182	149	54	5/16/2023 17:34
237	31.9	81.3	1009.94	353	18.32	11.1273	148	54	5/16/2023 17:34
238	31.8	81.3	1009.88	353	17.3	10.8182	148	51	5/16/2023 17:35
239	31.8	81.3	1009.8	353	21.7	11.1273	148	51	5/16/2023 17:35
240	31.8	81.3	1009.74	353	10.63	10.8182	148	51	5/16/2023 17:36
241	31.8	81.3	1009.66	353	16.64	11.1273	148	48	5/16/2023 17:36
242	31.8	81.3	1009.73	353	23.16	11.4364	148	48	5/16/2023 17:37
243	31.8	81.3	1009.67	353	11.8	12.3636	148	48	5/16/2023 17:37
244	31.8	81.3	1009.68	353	16.71	11.1273	147	49	5/16/2023 17:38
245	31.7	81.2	1009.65	353	9.82	9.89091	147	49	5/16/2023 17:38
246	31.7	81	1009.53	353	14.14	10.8182	147	49	5/16/2023 17:39
247	31.7	81	1009.52	353	24.56	9.89091	147	50	5/16/2023 17:39
248	31.8	81	1009.52	353	16.05	10.5091	147	50	5/16/2023 17:40
249	31.8	81	1009.45	353	14.73	7.41818	147	50	5/16/2023 17:40
250	31.8	81	1009.42	353	13.48	8.96364	147	50	5/16/2023 17:41
251	31.8	81	1009.45	353	10.48	7.72727	146	54	5/16/2023 17:41
252	31.8	81	1009.39	353	17.59	11.1273	146	54	5/16/2023 17:42

253	31.8	81	1009.34	353	16.2	10.2	144	53	5/16/2023 17:42
254	31.8	81	1009.27	353	17.22	7.10909	144	53	5/16/2023 17:43
255	31.8	81	1009.39	353	18.25	8.03637	144	52	5/16/2023 17:43
256	31.8	81	1009.33	353	17.59	9.58182	144	51	5/16/2023 17:44
257	31.8	81	1009.22	353	17.81	9.89091	144	51	5/16/2023 17:44
258	31.8	81	1009.19	353	25.65	10.8182	144	51	5/16/2023 17:45
259	31.8	81	1009.14	353	17.52	9.89091	144	51	5/16/2023 17:45
260	31.8	81	1009.11	353	12.17	10.5091	144	50	5/16/2023 17:46
261	31.7	81	1009.08	353	13.7	9.58182	144	50	5/16/2023 17:46
262	31.7	81	1008.88	353	12.82	10.5091	144	50	5/16/2023 17:47
263	31.8	81	1008.73	353	13.7	8.96364	145	49	5/16/2023 17:47
264	31.8	82.3	1008.62	353	16.2	10.8182	145	49	5/16/2023 17:48
265	31.7	82.6	1008.45	353	14.88	9.89091	145	49	5/16/2023 17:48
266	31.8	82.6	1008.4	353	8.87	8.96364	145	48	5/16/2023 17:49
267	31.8	82.6	1008.39	353	10.11	9.58182	145	48	5/16/2023 17:49
268	31.8	82.6	1008.4	308	15.02	8.65455	145	48	5/16/2023 17:50
269	31.8	82.6	1008.32	353	13.34	7.10909	145	49	5/16/2023 17:50
270	31.8	82.6	1008.22	353	10.04	8.65455	145	49	5/16/2023 17:51
271	31.8	82.6	1008.32	308	5.86	8.34546	145	51	5/16/2023 17:51
272	31.8	81.6	1008.4	353	15.24	8.96364	145	51	5/16/2023 17:52
273	31.8	81	1008.41	353	17.96	9.58182	146	51	5/16/2023 17:52
274	31.8	81	1008.31	353	10.99	9.89091	146	51	5/16/2023 17:53
275	31.7	81	1008.19	353	12.09	9.58182	146	49	5/16/2023 17:53
276	31.7	81.1	1008.09	353	15.24	8.65455	146	49	5/16/2023 17:54
277	31.7	81.3	1007.96	353	15.39	9.27273	146	52	5/16/2023 17:54
278	31.7	81.3	1007.77	353	13.48	9.89091	146	52	5/16/2023 17:55
279	31.7	81.3	1007.91	353	17	9.27273	146	52	5/16/2023 17:55
280	31.6	81.3	1007.9	353	14.29	9.58182	146	50	5/16/2023 17:56

281	31.6	81.3	1007.93	353	11.43	9.27273	146	50	5/16/2023 17:56
282	31.6	81.6	1007.98	353	17.22	11.1273	146	52	5/16/2023 17:57
283	31.6	81.6	1007.83	353	10.55	9.89091	147	52	5/16/2023 17:57
284	31.6	81.6	1007.84	353	15.9	9.58182	147	51	5/16/2023 17:58
285	31.6	81.6	1007.67	353	15.76	9.27273	147	51	5/16/2023 17:58
286	31.6	81.6	1007.69	353	15.32	8.34546	147	51	5/16/2023 17:59
287	31.6	81.6	1007.63	353	14.14	9.89091	147	48	5/16/2023 17:59
288	31.5	81.6	1007.37	353	17.66	8.96364	147	48	5/16/2023 18:00
289	31.5	81.6	1007.26	353	11.36	8.65455	147	48	5/16/2023 18:00
290	31.5	81.6	1007.19	353	8.28	9.27273	147	48	5/16/2023 18:01
291	31.5	81.5	1007.43	306	10.11	8.96364	148	48	5/16/2023 18:01
292	31.5	81.6	1007.62	307	17	10.5091	148	48	5/16/2023 18:02
293	31.5	81.6	1007.67	353	21.99	9.27273	148	48	5/16/2023 18:02
294	31.5	81.6	1007.61	353	18.54	10.5091	148	48	5/16/2023 18:03
295	31.5	81.6	1007.51	353	16.27	9.89091	148	49	5/16/2023 18:03
296	31.5	81.6	1007.35	0	9.89	9.58182	148	49	5/16/2023 18:04
297	31.5	81.6	1007.2	353	21.92	8.65455	148	49	5/16/2023 18:04
298	31.5	81.6	1007.19	353	15.9	11.1273	148	49	5/16/2023 18:05
299	31.4	81.3	1007.52	353	18.69	8.03637	148	51	5/16/2023 18:05
300	31.4	81.3	1007.44	308	13.12	9.27273	148	54	5/16/2023 18:06
301	31.4	81.3	1007.29	353	10.63	9.89091	149	54	5/16/2023 18:06
302	31.4	81.3	1007.16	353	10.33	8.96364	149	51	5/16/2023 18:07
303	31.3	81.5	1006.9	353	12.68	8.65455	149	51	5/16/2023 18:07
304	31.3	81.6	1006.77	353	16.71	7.41818	149	51	5/16/2023 18:08
305	31.3	81.6	1006.68	353	10.41	10.2	149	54	5/16/2023 18:08
306	31.3	81.6	1006.57	353	14.58	9.89091	149	56	5/16/2023 18:09
307	31.3	81.6	1006.47	353	13.41	9.27273	149	56	5/16/2023 18:09
308	31.3	81.6	1006.29	353	22.43	8.96364	149	55	5/16/2023 18:10

309	31.2	81.6	1006.13	353	17.3	10.5091	150	54	5/16/2023 18:10
310	31.2	81	1005.9	353	10.19	11.4364	150	56	5/16/2023 18:11
311	31.2	80.3	1005.69	353	8.13	7.10909	150	54	5/16/2023 18:11
312	31.2	80.4	1005.55	353	10.55	9.89091	150	56	5/16/2023 18:12
313	31.2	80.4	1005.41	353	8.79	8.65455	150	56	5/16/2023 18:12
314	31.2	80	1005.43	353	7.47	9.89091	150	55	5/16/2023 18:13
315	31.2	80	1005.4	353	6.74	8.34546	151	54	5/16/2023 18:13
316	31.1	80	1005.35	353	10.7	7.10909	151	56	5/16/2023 18:14
317	31.1	80	1005.33	353	10.55	7.41818	151	52	5/16/2023 18:14
318	31.1	80	1005.21	353	8.28	8.03637	151	52	5/16/2023 18:15
319	31.1	80	1005.19	353	9.89	10.5091	151	52	5/16/2023 18:15
320	31.1	80	1005.14	309	7.99	10.2	152	52	5/16/2023 18:16
321	31.1	80	1005.1	353	10.04	9.58182	152	52	5/16/2023 18:16
322	31.1	80	1004.99	353	8.5	9.89091	152	52	5/16/2023 18:17
323	31.1	80	1004.97	353	9.75	8.96364	152	49	5/16/2023 18:17
324	31.1	80	1004.98	353	9.23	9.58182	153	49	5/16/2023 18:18
325	31.1	80	1005.08	353	0.21	9.58182	153	49	5/16/2023 18:18
326	28.4	67.7	1004.99	232	0.21	2.78182	154	51	5/18/2023 4:06
327	28.4	67.7	1004.96	155	0.21	3.09091	154	50	5/18/2023 4:07
328	28.4	67.7	1004.86	233	0.21	2.78182	154	51	5/18/2023 4:08
329	28.4	67.7	1004.84	232	0.21	3.4	154	52	5/18/2023 4:08
330	28.4	67.7	1004.79	155	0.36	1.85455	154	56	5/18/2023 4:09
331	28.4	67.7	1004.73	154	0.21	4.32727	154	56	5/18/2023 4:09
332	28.4	67.7	1004.63	231	0.21	2.78182	155	54	5/18/2023 4:10
333	28.3	67.7	1004.61	153	0.21	3.4	155	54	5/18/2023 4:10
334	28.3	67.7	1004.6	155	0.21	3.70909	156	51	5/18/2023 4:11
335	28.3	67.7	1004.57	233	0.29	4.01818	156	51	5/18/2023 4:11
336	28.3	67.7	1004.51	232	0.29	3.09091	156	51	5/18/2023 4:12

337	28.3	67.7	1004.46	155	2.78	3.70909	156	48	5/18/2023 4:12
338	28.3	67.7	1004.42	154	6.59	3.4	157	48	5/18/2023 4:13
339	28.3	67.7	1004.43	155	6.08	1.54545	157	48	5/18/2023 4:13
340	28.3	67.7	1004.51	232	5.27	2.78182	157	49	5/18/2023 4:14
341	28.3	67.6	1004.48	155	0.65	2.47273	158	49	5/18/2023 4:14
342	28.3	67.6	1004.47	155	5.93	4.32727	158	49	5/18/2023 4:15
343	28.3	67.6	1004.37	232	5.27	2.47273	158	50	5/18/2023 4:15
344	28.3	67.6	1004.3	155	3.59	2.16364	159	50	5/18/2023 4:16
345	28.3	67.6	1004.36	230	7.62	2.78182	158	50	5/18/2023 4:16
346	28.3	67.6	1004.31	155	5.57	1.54545	159	50	5/18/2023 4:17
347	28.3	67.6	1004.17	232	6.96	1.23636	159	54	5/18/2023 4:17
348	28.2	67.6	1004.22	155	6.15	2.47273	159	54	5/18/2023 4:18
349	28.2	67.6	1004.13	232	7.4	1.54545	159	53	5/18/2023 4:18
350	28.2	67.6	1004.24	232	6.74	2.78182	159	53	5/18/2023 4:19
351	28.2	67.6	1004.38	155	7.47	0.927273	159	52	5/18/2023 4:19
352	28.2	67.5	1004.27	78	4.17	1.54545	160	51	5/18/2023 4:20
353	28.2	67.6	1004.28	233	0.29	2.16364	160	51	5/18/2023 4:20
354	28.2	67.6	1004.36	232	0.21	2.16364	160	51	5/18/2023 4:21
355	28.2	67.6	1004.38	233	0.21	1.85455	161	51	5/18/2023 4:21
356	28.2	67.6	1004.47	310	6.89	4.01818	160	50	5/18/2023 4:22
357	28.2	67.6	1004.48	311	6.67	2.78182	161	50	5/18/2023 4:22
358	28.2	67.6	1004.5	155	1.97	2.78182	161	50	5/18/2023 4:23
359	28.2	67.6	1004.55	310	2.71	2.16364	161	49	5/18/2023 4:23
360	28.2	67.6	1004.5	230	7.4	2.16364	161	49	5/18/2023 4:24
361	28.2	67.6	1004.49	232	4.83	4.63636	161	49	5/18/2023 4:25
362	28.2	67.5	1004.43	233	2.85	4.94546	162	48	5/18/2023 4:26
363	28.2	67.5	1004.49	233	0.21	2.78182	162	48	5/18/2023 4:26
364	28.2	67.6	1004.48	310	0.29	3.09091	162	48	5/18/2023 4:27

365	28.2	67.6	1004.53	233	1.53	1.23636	162	49	5/18/2023 4:27
366	28.2	67.6	1004.5	232	0.29	0.618182	162	49	5/18/2023 4:28
367	28.2	67.5	1004.5	232	0.21	1.54545	162	51	5/18/2023 4:28
368	28.2	67.5	1004.47	233	0.07	2.47273	162	51	5/18/2023 4:29
369	28.2	67.5	1004.5	232	0.21	3.4	162	51	5/18/2023 4:30
370	28.2	67.6	1004.6	233	0.21	3.4	163	51	5/18/2023 4:30
371	28.2	67.5	1004.61	309	0.29	2.78182	163	49	5/18/2023 4:31
372	28.2	67.6	1004.72	310	0.29	4.63636	163	49	5/18/2023 4:31
373	28.2	67.6	1004.65	308	0.21	2.16364	163	52	5/18/2023 4:32
374	28.2	67.6	1004.56	311	0.21	3.09091	163	52	5/18/2023 4:32
375	28.1	67.5	1004.47	155	0.43	3.4	163	52	5/18/2023 4:33
376	28.1	67.5	1004.48	155	0.21	2.47273	163	50	5/18/2023 4:33
377	28.2	67.6	1004.55	232	0.29	1.85455	163	50	5/18/2023 4:34
378	28.2	67.6	1004.55	232	0.21	4.01818	163	52	5/18/2023 4:34
379	28.2	67.6	1004.56	233	0.29	3.4	163	52	5/18/2023 4:35
380	28.2	67.6	1004.61	308	0.29	3.09091	163	51	5/18/2023 4:35
381	28.2	67.6	1004.59	310	4.54	2.16364	163	51	5/18/2023 4:36
382	28.2	67.6	1004.6	232	0.07	3.09091	163	51	5/18/2023 4:36
383	28.2	67.6	1004.56	232	6.08	3.09091	163	48	5/18/2023 4:37
384	28.1	67.5	1004.49	230	0.29	3.70909	164	48	5/18/2023 4:37
385	28.2	67.6	1004.52	310	8.5	3.4	163	48	5/18/2023 4:38
386	28.1	67.5	1004.58	310	3.07	3.09091	164	48	5/18/2023 4:38
387	28.2	67.6	1004.5	310	6.67	5.87273	163	48	5/18/2023 4:39
388	28.1	67.5	1004.48	310	0.14	4.01818	164	48	5/18/2023 4:39
389	28.2	67.6	1004.48	233	0.07	4.63636	163	48	5/18/2023 4:40
390	28.1	67.5	1004.48	310	0.29	2.47273	163	48	5/18/2023 4:40
391	28.1	67.5	1004.47	233	0.21	3.4	163	49	5/18/2023 4:41
392	28.1	67.5	1004.45	232	0.21	5.56364	164	49	5/18/2023 4:41

393	28.1	67.5	1004.5	232	3.81	3.4	163	49	5/18/2023 4:42
394	28.1	67.5	1004.51	155	3.81	3.70909	163	49	5/18/2023 4:42
395	28.1	67.5	1004.48	155	0.21	4.63636	163	51	5/18/2023 4:43
396	28.1	67.5	1004.48	232	5.57	3.09091	163	54	5/18/2023 4:44
397	28.1	67.4	1004.5	232	0.29	3.09091	163	54	5/18/2023 4:44
398	28.1	67.4	1004.55	232	0.21	4.32727	163	51	5/18/2023 4:45
399	28.1	67.5	1004.5	232	0.21	5.56364	163	51	5/18/2023 4:45
400	28.1	67.5	1004.48	155	0.21	2.47273	163	51	5/18/2023 4:46
401	28.1	67.5	1004.49	155	5.42	2.16364	163	54	5/18/2023 4:46
402	28.1	67.4	1004.49	233	6.67	3.09091	163	56	5/18/2023 4:47
403	28.1	67.4	1004.45	155	3.29	0.927273	162	56	5/18/2023 4:47
404	28.1	67.4	1004.31	154	7.18	1.54545	162	55	5/18/2023 4:48
405	28.1	67.4	1004.29	153	5.27	0.618182	162	54	5/18/2023 4:48
406	28.1	67.4	1004.23	232	2.27	2.16364	162	56	5/18/2023 4:49
407	28.1	67.4	1004.24	310	0.29	2.47273	162	54	5/18/2023 4:49
408	28.1	67.4	1004.31	230	5.2	3.09091	162	56	5/18/2023 4:50
409	28.2	67.4	1004.31	233	4.91	2.78182	162	56	5/18/2023 4:50
410	28.2	67.4	1004.28	232	3.51	2.78182	162	55	5/18/2023 4:51
411	28.2	67.4	1004.29	230	0.21	3.09091	162	54	5/18/2023 4:52
412	28.2	67.4	1004.27	232	0.43	3.09091	161	56	5/18/2023 4:52
413	28.2	67.4	1004.33	353	0.29	2.78182	161	52	5/18/2023 4:53
414	28.2	67.4	1004.25	353	0.29	2.16364	161	52	5/18/2023 4:53
415	28.2	67.4	1004.28	353	0.29	1.85455	161	52	5/18/2023 4:54
416	28.2	67.4	1004.34	353	0.29	1.23636	161	52	5/18/2023 4:54
417	28.2	67.4	1004.24	232	0.29	1.54545	161	52	5/18/2023 4:55
418	28.2	67.4	1004.27	233	0.21	2.47273	160	52	5/18/2023 4:55
419	28.2	67.4	1004.36	232	0.21	1.54545	160	49	5/18/2023 4:56
420	28.2	67.4	1004.46	233	0.21	1.85455	160	49	5/18/2023 4:56

421	28.3	67.5	1004.44	232	0.21	1.23636	160	49	5/18/2023 4:57
422	28.3	67.5	1004.37	233	0.21	2.78182	160	51	5/18/2023 4:57
423	28.3	67.5	1004.37	232	0.14	1.85455	160	50	5/18/2023 4:58
424	28.3	67.5	1004.43	155	0.43	3.09091	160	51	5/18/2023 4:58
425	28.3	67.5	1004.37	155	0.29	2.16364	159	52	5/18/2023 4:59
426	28.3	67.5	1004.32	153	0.21	2.16364	159	56	5/18/2023 4:59
427	28.3	67.5	1004.29	155	0.21	2.16364	159	56	5/18/2023 5:00
428	28.3	67.5	1004.24	155	0.21	3.09091	158	54	5/18/2023 5:00
429	28.3	67.5	1004.21	155	0.21	2.16364	159	54	5/18/2023 5:01
430	28.3	67.5	1004.22	153	0.21	1.54545	158	51	5/18/2023 5:01
431	28.3	67.4	1004.2	153	0.21	2.16364	158	51	5/18/2023 5:02
432	28.3	67.4	1004.25	153	0.21	1.85455	158	51	5/18/2023 5:02
433	28.3	67.4	1004.33	155	0.29	4.01818	158	48	5/18/2023 5:03
434	28.3	67.4	1004.4	155	0.21	3.09091	158	48	5/18/2023 5:03
435	28.3	67.4	1004.47	155	0.21	4.01818	157	48	5/18/2023 5:04
436	28.3	67.4	1004.44	154	0.21	3.70909	157	49	5/18/2023 5:04
437	28.3	67.4	1004.37	154	0.21	3.4	157	49	5/18/2023 5:05
438	28.3	67.4	1004.36	232	6.96	2.78182	157	49	5/18/2023 5:05
439	28.3	67.4	1004.41	230	6.52	1.54545	157	50	5/18/2023 5:06
440	28.3	67.4	1004.52	155	0.51	2.47273	156	50	5/18/2023 5:06
441	28.4	67.3	1004.5	232	0.14	3.09091	157	50	5/18/2023 5:15
442	28.3	67.3	1004.56	232	0.21	4.01818	156	50	5/18/2023 5:15
443	28.3	67.3	1004.58	233	0.21	4.01818	156	54	5/18/2023 5:16
444	28.3	67.3	1004.53	232	0.21	3.09091	156	54	5/18/2023 5:16
445	28.3	67.3	1004.5	232	0.21	4.94546	156	53	5/18/2023 5:17
446	28.3	67.3	1004.44	155	0.21	4.63636	155	53	5/18/2023 5:17
447	28.4	67.3	1004.35	310	0.21	4.01818	155	52	5/18/2023 5:18
448	28.3	67.3	1004.26	233	5.05	4.63636	155	51	5/18/2023 5:18

449	28.3	67.3	1004.11	311	8.21	6.18182	155	51	5/18/2023 5:19
450	28.3	67.3	1003.98	232	3.66	2.78182	155	51	5/18/2023 5:19
451	28.4	67.3	1003.9	155	4.91	3.70909	155	51	5/18/2023 5:20
452	28.3	67.3	1003.73	154	0.21	3.70909	154	50	5/18/2023 5:20
453	28.4	67.3	1003.62	154	4.98	2.78182	154	50	5/18/2023 5:21
454	28.3	67.3	1003.62	233	4.1	1.85455	154	50	5/18/2023 5:21
455	28.4	67.3	1003.56	233	8.35	2.78182	154	49	5/18/2023 5:22
456	28.4	67.3	1003.49	155	5.79	2.16364	154	49	5/18/2023 5:22
457	28.4	67.3	1003.5	231	0.21	4.01818	154	49	5/18/2023 5:23
458	28.4	67.2	1003.47	155	0.87	1.85455	153	48	5/18/2023 5:23
459	28.3	67.2	1003.3	155	0.21	2.78182	153	48	5/18/2023 5:24
460	28.3	67.2	1003.25	155	0.21	2.16364	153	48	5/18/2023 5:24
461	28.4	67.2	1003.19	155	2.71	2.78182	153	49	5/18/2023 5:25
462	28.3	67.2	1003.03	155	2.71	4.63636	153	49	5/18/2023 5:25
463	28.3	67.2	1003	232	4.83	4.01818	153	51	5/18/2023 5:26
464	28.3	67.2	1002.98	155	5.49	3.4	152	51	5/18/2023 5:26
465	28.4	67.2	1002.98	154	6.08	4.01818	152	51	5/18/2023 5:27
466	28.4	67.2	1002.89	155	4.69	3.70909	152	51	5/18/2023 5:27
467	28.3	67.1	1002.9	155	8.28	2.78182	152	49	5/18/2023 5:28
468	28.3	67.1	1002.8	155	6.37	4.01818	152	49	5/18/2023 5:28
469	28.3	67.1	1002.79	232	6.59	2.47273	152	52	5/18/2023 5:29
470	28.3	67.1	1002.78	155	5.86	0.927273	151	52	5/18/2023 5:29
471	28.3	67	1002.67	231	6.37	0.927273	151	52	5/18/2023 5:30
472	28.3	67	1002.63	232	5.71	1.85455	151	50	5/18/2023 5:30
473	28.3	67	1002.64	155	6.52	1.54545	151	50	5/18/2023 5:31
474	28.3	67	1002.61	310	5.13	2.16364	151	52	5/18/2023 5:31
475	28.3	67	1002.63	155	6.15	1.85455	151	52	5/18/2023 5:32
476	28.3	67	1002.64	232	6.59	2.78182	151	51	5/18/2023 5:32

477	28.3	67	1002.62	233	3.29	3.4	150	51	5/18/2023 5:33
478	28.3	67	1002.62	155	5.27	4.94546	150	51	5/18/2023 5:33
479	28.3	67	1002.69	154	6.96	4.94546	150	48	5/18/2023 5:34
480	28.3	67	1002.64	154	6.45	5.25455	150	48	5/18/2023 5:34
481	28.3	67	1002.56	232	7.03	3.4	150	48	5/18/2023 5:35
482	28.3	67	1002.45	155	2.71	2.78182	149	48	5/18/2023 5:35
483	28.3	67	1002.35	353	6.23	2.16364	149	48	5/18/2023 5:36
484	28.3	67	1002.26	353	5.35	3.70909	149	48	5/18/2023 5:36
485	28.3	67	1002.26	353	7.55	1.85455	149	48	5/18/2023 5:37
486	28.3	67	1002.27	353	7.25	2.16364	149	48	5/18/2023 5:37
487	28.3	67	1002.22	353	10.41	2.47273	149	49	5/18/2023 5:38
488	28.3	67	1002.22	353	14.29	1.23636	148	49	5/18/2023 5:38
489	28.3	67	1002.12	353	13.85	6.8	148	49	5/18/2023 5:39
490	28.4	67.3	1002.04	353	11.43	4.01818	148	49	5/18/2023 5:39
491	28.4	68.6	1002.1	353	10.99	5.87273	148	51	5/18/2023 5:40
492	28.4	68.9	1002.08	353	9.89	3.4	148	54	5/18/2023 5:40
493	28.4	68.9	1002.15	0	7.03	3.09091	148	54	5/18/2023 5:41
494	28.4	68.2	1002.19	353	6.67	2.47273	148	51	5/18/2023 5:41
495	31.2	66	1002.26	353	7.62	4.01818	147	51	5/18/2023 6:58
496	31.2	66.1	1002.32	353	10.63	6.49091	147	51	5/18/2023 6:59
497	31.2	66.1	1002.39	353	12.46	6.18182	147	54	5/18/2023 6:59
498	31.2	66.1	1002.5	353	10.99	5.56364	147	56	5/18/2023 7:00
499	31.2	66.1	1002.4	353	9.82	3.70909	147	56	5/18/2023 7:00
500	31.2	66.1	1002.47	353	10.55	3.70909	147	55	5/18/2023 7:01
501	31.2	66	1002.5	353	9.38	5.25455	147	54	5/18/2023 7:01

Appendix E

LIGTAS User Manual

Overview

LIGTAS is a system designed for monitoring and predicting weather conditions and flood levels at the barangay level. It utilizes LoRaWAN technology to transmit sensor data and displays this data through a web application. Additionally, it employs predictive modeling techniques to forecast weather patterns and flood levels for the upcoming hours. The weather monitoring hardware of the system incorporates an array of sensors including anemometer, wind vane, rain gauge, atmospheric pressure sensor, and temperature and humidity sensor. These sensors work collectively to acquire the necessary parameters for accurate weather predictions. As for the flood level monitoring hardware, an ultrasonic sensor is utilized to measure the height of water. LIGTAS effectively enables the continuous monitoring and prediction of weather conditions and flood levels, allowing timely notifications to be disseminated to the community's citizens in the event of an existing or imminent disaster. Its primary objective revolves around enhancing the awareness and preparedness of individuals within the community, empowering them to effectively respond to such calamities.

LIGTAS Web Application Features

The LIGTAS web application encompasses a range of features that serve as a Weather and Flood Monitoring System. The outlined features of the application include:

1. Real-Time Data Monitoring and Analysis - The application provides users with instantaneous and continuously updated information, ensuring they have access to

the most recent and relevant data, enabling them to make informed decisions based on real-time conditions.

2. Battery Percentage Display - The authority can conveniently monitor their system's battery level at a glance, allowing them to provide additional backup power based on the system's power consumption and ensure they have sufficient battery life for their needs.
3. Event Reporting - With the event reporting feature, users can quickly and easily document and communicate important events or incidents, enabling efficient reporting and facilitating timely response and resolution.
4. Advanced Forecasting and Prediction - The application utilizes advanced forecasting techniques to predict future conditions or events based on historical data and real-time data collected by the sensor system. This feature provides users with valuable insights into what can be expected in the coming days, enabling them to plan and make informed decisions accordingly.
5. System Notifications - The application delivers timely notifications to users who have granted permission for receiving notifications on their devices, thereby keeping them informed about system updates, essential alerts, or significant messages. This ensures that they remain updated and can promptly take the necessary actions as required.
6. Advisory Report Generation and Analysis - By generating comprehensive advisory reports, the application provides users with valuable insights, recommendations, and guidance based on analyzed data, helping them understand trends, mitigate risks, and make informed decisions for better outcomes.

Hardware User Manual

1. For the weather system, check the inside of the circuit box to ensure that the battery has sufficient charge.
2. Check the sensors outside the box such as anemometer, wind vane, and rain gauge. Ensure that it is not obstructed by anything and it works properly.
3. For the flood system, check the inside of the circuit box to ensure that the battery has sufficient charge.
4. Check the ultrasonic sensor and ensure that there are no obstructions below it to obtain accurate readings.
5. After checking, turn on both the weather and flood monitoring system.

Website User Manual

Homepage

1. Click 

2. Homepage displays all about the LIGTAS website. It exhibits the sole purpose of the weather and flood monitoring system for the citizens of Las Piñas City.

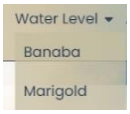
Weather Page

1. Click 

2. Hover the scroll bar to see the next 6-hour lead-time forecasting data in every one hour of the weather parameters of temperature and humidity, air pressure, wind speed and direction, and rainfall intensity.

3. Weather Page displays the real-time forecasted data of each weather parameter (a graph of temperature, humidity, wind speed and direction, air pressure, and rainfall intensity) in Brgy. Talon Dos where the weather monitoring is deployed.
4. Further, in the upper right, exhibits the current time and date in Philippine Standard Time. Beneath, is the percentage charge of the solar panel installed in the weather monitoring system.

Water Level

1. Click the water level 

to see the other area where the flood monitoring system is deployed. Under water level, is the banaba and marigold flood node.
2. Hover the scroll bar to see the next 6-hour lead-time forecasting data in every one hour of the water level height in every node.
3. Water Level exhibits a detailed water level status, such as height and description status for each barangay (Banaba and Marigold).
4. In the upper right, displays the current percentage charge of the solar panel.

Advisories

1. Click 

to see the summarized weather report and a flood report based on the forecasted data on the website.
2. Advisories show a table of precautionary measures, disaster preparedness, and color coded-rainfall advisories of a typhoon.

About Us

1. Click 

2. About Us exhibit information about the Ligtas system. It talks about the sole purpose of a weather and flood monitoring system and how it will make an impact to the citizen of Las Piñas.

Contact

1. Click Contacts
2. Contact page contains a form that will serve as a communication means between the barangay official, LGUs and the citizens for the feedback, reports, and concerns.

Appendix F

LIGTAS
Duplication Manual

MANUAL ON HOW TO DUPLICATE THE PROTOTYPE

INTRODUCTION

This manual provides comprehensive instructions on how to duplicate the prototype LIGTAS System developed by the proponents of this study. An individual may be able to build a replica of this prototype by following the subsequent steps carefully and may ensure consistency and reliability in the manufacturing process.

SAFETY PRECAUTIONS:

- **Verify Equipment Compatibility:** Before duplicating the LIGTAS system, ensure that all the equipment and components being used are compatible with each other. Refer to the system specifications and manufacturer's guidelines to ensure proper compatibility.
- **Workspace Organization:** Maintain a clean and organized workspace to minimize the risk of accidents and promote efficient work processes. Keep cables, tools, and equipment properly stored when not in use to prevent tripping hazards and potential damage.
- **Electrical Safety:** Adhere to proper electrical safety practices throughout the duplication process. Make sure to disconnect the power source and follow all necessary precautions when working with electrical components. Avoid working on the system in wet conditions to prevent electrical hazards.
- **Check Electrical Faulting:** Before initiating the duplication process, thoroughly inspect all electrical connections and components for any signs of faulting, such as

exposed wires or damaged insulation. Repair or replace any faulty components to prevent electrical hazards and ensure safe operation.

- **Emergency Preparedness:** Prioritize emergency preparedness by having safety measures in place. Maintain accessible fire extinguishers and first aid kits in the vicinity of the workspace. Familiarize yourself and your team with emergency protocols and evacuation procedures in case of unforeseen events.
- **Review the application requirements**: Verify that the web or cloud hosting provider satisfies the requirements specified by the web application.

MATERIALS AND EQUIPMENT

The following are the materials and equipment needed to recreate the prototype.

Materials:

See the Annex A: Bill of Materials for the complete list of materials.

See the Annex C: LIGTAS System Codes for the complete codes of the web application.

Tools and Equipment:

For Circuitry:

- Micro-USB to USB cable - for uploading the code to PACKETDUINO
- USB-B to USB cable - for uploading the code to Arduino UNO
- Multimeter
- Breadboard
- Wire Cutter
- Soldering Iron

For Hardware:

- Pliers
- Wrenches
- Hammer
- Screwdriver
- Impact drill

For Software:

- Visual Studio Code
- XAMPP Server
- phpMyAdmin
- Namecheap web hosting provider
- Node.js and Python

PROCEDURE

Step-by-step Procedure:

For the Hardware and Software part for Monitoring:

Uploading Code to PacketDUINO for Weather Monitoring System:

1. Install then open the Arduino IDE application. To setup the PacketDUINO with the Arduino IDE, go to the preference in the options above and paste the link to the additional boards manager text box: “https://raw.githubusercontent.com/Packetworx-IoT/packetDUINO/main/package_packetworx_index.json”

2. In the tools option above, select boards, and choose board manager. Type “packetworx” in the text box and search, you should see “Packetworx AVR Boards”. Once installed, you should be able to see the “PacketDUINO 32u4” and select it in the boards available in Packetworx Boards.
3. For the weather station system, open the “Weather_System_Code.ino” in the Arduino IDE and modify the **appeui**, **deveui**, and **appkey** according to the specification of the end device you register then upload it into the PacketDUINO. (See the Setting up the End Devices for further instruction)
4. Ensure to modify the code based on the specification of sensors and instruments since it varies depending on the specification sheet of the sensor.

Uploading Code to Arduino UNO for Flood Monitoring System:

1. Open the Arduino IDE application. In the tools option above, select boards and choose “Arduino UNO” in the boards available in Arduino Boards.
2. For the water level monitoring system, open the “Flood_Node_Code.ino” or copy the code in Annex C in the Arduino IDE and modify the **appeui**, **deveui**, and **appkey** according to the specification of the end device you register. Before uploading the code, make sure to change the boards into Arduino UNO first then proceed to uploading the code. (See the Setting up the End Devices for further instruction)
3. Ensure to modify the code based on the specification of sensors and instruments since it varies depending on the specification sheet of the sensor.

For the Hardware Part:

Modify the hardware of the system based on the figures shown in Appendix A.

1. For the weather system, modify the system by attaching the arm for each of the sensors to the pole.
2. Ensure that each weather sensor is fitted and meet the standards for the placement to get accurate readings.
3. Calibrate each sensor first before securely installing it to each arm.
4. Once done, proceed to tightly secure the installation of each sensor and follow the connection of the sensor to the microcontroller based on the Weather System Diagram in *Chapter 3.3.1 Circuit Diagram*.
5. Secure a good position for the solar panel for better charging. Once connected to the solar charge controller, power on the system by turning on the switch on the circuit box.
6. For the flood monitoring system, modify the system by installing an arm to a sturdy wall and make sure it aligns at the bodies of water or river.
7. Follow the connection of the flood system based on the Weather System Diagram based in *Chapter 3.3.1 Circuit Diagram.*
8. Once done, place the system in a secure and water-proof box, then mount it into the pole that is attached to the wall.
9. Secure a good position for the solar panel for better charging. Once connected to the solar charge controller, power on the system by turning on the switch on the box.

Setting Up the End Devices

1. Go and create an account at the Packetworx ThingStack Cloud "https://packetworx.eu1.cloud.thethings.industries/oauth/login".
2. Once you have secured an account, go to the console.
3. Type "packetworx" as the Tenant ID and select "Australia 1" as the existing clusters.
4. Go to applications and click "Create application". Fill necessary information such as Application ID.
5. Register all the end devices or nodes that are available.
6. In registering the end device, select *enter end device specifics manually* and choose *Europe 863-870 MHz* for the frequency plan. The LoRaWAN version should be *LoRaWAN Specification 1.0.1*. Type any JoinEUI as long as you can remember it (can be 00 00 00 00 00 00 00 00). Generate DevEUI, AppKey, and enter the End device ID that is related to the node you will be registering.
7. Once done, you will now see the end device you just registered and once you click it, the **AppEUI, DevEUI,** and **AppKey** will now be available.

Setting up the Payload formatters in The ThingStack Cloud:

1. Choose the end-device you want to modify the payload formatters in the list of end-devices.

2. Once you choose and open an end-device, select Payload Formatters and edit the *Formatter Code* in Uplink.
3. Copy then modify the Payload Formatter Code in Annex C.
4. For the weather system, you may or may not modify the code based on the sensors you will be using to monitor the weather.
5. For the flood system, modify the code based on the distance of the system to the bottom of the bodies of water.
6. After modifying the code, click *Save changes* and check the payload changes in the Live data.

For Predictive Model

Shown here are the steps on how to duplicate the predictive model developed. Snippets of the codes are shown here. For the whole code of the predictive model and how it was deployed, refer to Appendix C at the Training Model Codes and main.py.

1. Gather dataset

Gather dataset from free dataset sites such as Kaggle, and Google Dataset Search. Datasets are usually in .csv format. If it is not in that format, convert the dataset to .csv. Datasets can also be requested from DOST - PAGASA. The links for these free dataset websites are enumerated below:

a) Kaggle [https://www.kaggle.com/datasets]

b) Google Dataset Search [https://datasetsearch.research.google.com/]

c) DOST - PAGASA [https://www.pagasa.dost.gov.ph/climate/climate-data]

2. Install necessary libraries

Using **pip install**, install all the necessary libraries such as numpy, pandas, matplotlib, statsmodels, scikit - learn, and tensorflow. Show below is the syntax in using pip install.

```
pip install numpy
pip install pandas
```

3. Import the necessary libraries

Use the **from** and **import** function to import the installed libraries like shown below.

```
import tensorflow as tf
import joblib
from tensorflow.keras.models import Sequential
from tensorflow.keras.metrics import RootMeanSquaredError
from sklearn.preprocessing import StandardScaler
```

4. Prepare dataset

Load the dataset in your workspace (e.g., VS Code) using the **.read_csv** method of pandas.

```
df = pd.read_csv(r'dataset\path.csv')
```

5. Remove and replace outliers

First, describe the dataset using the **.describe** method to see if there are outliers or missing values indicated by -9999.00 at the minimum column.

```
df.describe().transpose()
```

Replace the outliers with the mean of each row of the dataset so that the model will not read erroneous values during training. The code below can be replicated to other columns of the dataset just by replacing the column name. In this case, the ‘T (degC)’. Of course, the variables should also be replaced accordingly.

```
tp = df['T (degC)']
bad_tp = tp == -9999.00
tp[bad_tp] = df['T (degC)'].mean()
```

6. Develop supervised learning code

The code shown below will turn the input (X) of the data frame (dataset) into a 3D tensor and the output (y) into a 2D tensor. This is necessary because an LSTM neural network requires a 3D tensor as its input during training. Also, this code applies the backpropagation algorithm. A specific window size or batch size will be set here to define the number of data inputs during training which will be used to predict the output per batch. After which, the first input will be dropped from the first batch and the latest output will be appended to the next batch of input. This makes the output during training depend on the previous inputs and new inputs. Thus, applying the concept of backpropagation.

```
def df_to_X_y(df, window_size=8):
  df_as_np = df.to_numpy()
  X = []
  y = []
  for i in range(len(df_as_np) - window_size):
     row = df_as_np[i:i+window_size]
```

```
        X.append(row)
        label = df_as_np[i+window_size]
        y.append(label)
    return np.array(X), np.array(y)
```

7. Split the dataset into train, validation, and test set

This is common in machine learning in order for us to have separate sets of datasets for training, validation, and testing. The train set will be used for training and the validation set will be used during training in order to know if the model overfits or not. If the loss in the train set is greater than the loss in the validation set, then the model did not overfit. Otherwise, the model overfitted and is needed to be retuned and retrained. Shown below is the code for splitting the dataset into **train (70%)**, **validation (20%)**, and **test (10%)** sets.

```
X_train, y_train = X[:625000], y[:625000]
X_val, y_val = X[625000:800000], y[625000:800000]
X_test, y_test = X[800000:], y[800000:]
```

8. Preprocess the dataset

Since this weather and flood monitoring model is a multivariate model, the values of every feature are very different from each other which may cause loss value. As a solution, this preprocessing function will put all of the features in the same scale in order for the predictive model to train and converge faster. Refer to Appendix C to visualize how a standardscaler was used to preprocess the train set and save the preprocessed set as a scaler. Shown below is how the preprocess function was called to preprocess the train, validation, and test sets.

```
X_train = preprocess(X_train)

X_val = preprocess(X_val)

X_test = preprocess(X_test)
```

9. Define the model architecture, apply dropout to avoid overfitting.

A deep LSTM neural network was defined here in order to acquire a high accuracy since the dataset is large. In order to avoid overfitting, a dropout layer was added before the output layer. Shown here is a sample model architecture, showing the input layer, hidden layers, dropout layer, and output layer. Refer to Appendix C for the full architecture of the model.

```
model = Sequential()

model.add(InputLayer(input_shape = (8,8)))

model.add(LSTM(256, return_sequences=True))

model.add(Dense(8, 'relu'))

model.add(Dropout(0.001))

model.add(Dense(8, 'linear'))
```

10. Define the callbacks to be used in training

The callbacks defined here are the **model_save**, **early_stop**, and **reduce_lr**. Shown below is the callback **model_save** from the name itself, saves the model during training. Specifically, save_weights_only was set to False in order to save the whole architecture of the model during training. This is very useful especially when you want to reload your model again since a one liner code is enough in this case. Unlike when you set it to True, you will need to compile the model again after you reload it.

```
model_save = ModelCheckpoint('./8var_2_2_model.h5',
                             save_best_only=True,
                             save_weights_only = False,
                             monitor = 'val_loss',
                             mode = 'min',
                             verbose = 1)
```

Next callback shown is the **early_stop**. This basically stops the training of the model when it doesn't learn properly or improve anymore during training.

```
early_stop = EarlyStopping(monitor = 'val_loss', min_delta = 0.001,
                           patience = 5, mode = 'min', verbose = 1,
                           restore_best_weights = True)
```

Last callback shown is the **reduce_lr**. This reduces the learning rate during training when the model reaches a plateau. This helps the model to decrease the loss faster during training.

```
reduce_lr = ReduceLROnPlateau(monitor = 'val_loss', factor = 0.3,
                              patience = 2, min_delta = 0.001,
                              mode = 'min', verbose = 1)
```

11. Compile the model

This is a necessary process to configure the model first before training. Added here are the **loss**, **optimizer**, and **metrics** parameters which are used to know the performance of the model during training in the train and validations sets. The syntax for compiling a model is shown below.

```
model.compile(loss=MeanSquaredError(),
optimizer=Adam(learning_rate=0.001),
metrics=[RootMeanSquaredError(), 'accuracy'])
```

12. Train the model

Shown below is how the **model.fit** is used to train the model in the specified number of epochs. The parameters set here are the **X_train**, **y_train**, **epochs**, and **verbose**. Aside from that, **callbacks** were also called here in order to perform the callbacks defined earlier during training. Moreover, validation data (X_val and y_val) are called here to evaluate the model in the unknown data (validation set).

```
model.fit(
    X_train,
    y_train,
    validation_data=(X_val, y_val),
    epochs = 40,
    verbose = 1,
    callbacks = [model_save, early_stop, reduce_lr])
```

13. Evaluate the model in the test set

This is necessary to further test the trained model in unknown data. The same metrics used earlier during training, with addition of the Mean Absolute Error (MAE) will be used to evaluate the model in the test set. Show here is how evaluation is done in the test set.

```
# make predictions on the test dataset
predictions = model.predict(X_test)
```

```
# calculate MSE, RMSE, and MAE
mse = mean_squared_error(y_test, predictions)
rmse = mean_squared_error(y_test, predictions, squared=False)
mae = mean_absolute_error(y_test, predictions)
# print the results
print("Mean squared error:", mse)
print("Root mean squared error:", rmse)
print("Mean absolute error:", mae)
```

14. If the model does not perform well, fine - tune the model

Fine tuning the model can be done in the model architecture itself or during preprocessing. Here are some solutions for fine tuning:

A. Add or remove some LSTM layers.

If the dataset is not that large, a shallow LSTM model is just fine. However, if the dataset is large, then a deep LSTM model is required. Utilize the **model.add** that was shown earlier in step 9 in adding an LSTM layer. In removing a LSTM layer, **simply delete** the line of code for the specific layer needed to be deleted.

B. Increase or decrease the number of LSTM units per layer.

```
model.add(LSTM(256, return_sequences=True))
```

Shown above is a sample LSTM layer. The LSTM units in this layer is ‘256’. This can be modified in order for the model to fit better.

C. Apply regularizer in every layer.

model.add(LSTM(256, return_sequences =True,

kernel_regularizer = regularizers.l2(0.01)))

Shown above is how a regularizer was applied in an LSTM layer.

D. Increase the rate of the dropout layer.

model.add(Dropout(0.001))

Shown above is a sample dropout layer. The rate of the dropout layer shown is 0.001. This can be modified from 0 to 1.

E. Change the window size applied in the supervised learning code.

This can be easily done by replacing the **window_size** value in the **df_to_X_y** function defined earlier in step 6.

15. Evaluate the model in the field data set.

After the sensors gather a large amount of data, load it in a separate jupyter notebook file then test the model on it. Treat the field data set like the test set in evaluating the model. Duplicate the process in step 13.

16. If the model does not perform well, fine - tune the model using transfer learning.

This can be done by freezing the weights of the trained model (except the output layer), then retraining it with new layers and with the field data set. It is very important to note here that the **field dataset** must be the one loaded here since retraining of model will be done with the field dataset. The process from loading the **field dataset**, splitting of data, and preprocessing up to here must be done again. Shown below is how a pre-trained model (the model that was trained earlier) was loaded, how the pre-trained model layers' weights are frozen, and how new layers are added to be trained for transfer learning.

```
#load model
pretrained_model = load_model(r'path\of\pre-trained\model.h5')
# Freeze the weights of the pre-trained layers
for layer in pretrained_model.layers[:-1]:
    layer.trainable = False
# Replace the output layer with a new layer appropriate for the new dataset
pretrained_model.layers[-2].trainable = True # Set last layer as trainable
pretrained_model.add(Reshape((8, 1)))
pretrained_model.add(LSTM(64, return_sequences=True, name = '64_LSTM'))
pretrained_model.add(LSTM(32, return_sequences=True, name = '32_LSTM'))
pretrained_model.add(LSTM(16, return_sequences=True, name = '16_LSTM'))
pretrained_model.add(LSTM(8, name = '11_LSTM'))
pretrained_model.add(Dense(8, activation='relu', name = '11_Dense'))
pretrained_model.add(Dropout(0.1, name = 'Dropout_tune'))
pretrained_model.add(Dense(8, activation='linear', name = '11_output_Dense'))
```

This code is not shown in Appendix C as transfer learning was not further applied in the study. After defining the model architecture with added layers, retraining of the model can be done again using the model.fit.

17. If the model finally performs well, deploy the model on the website.

Integrate the model to the website through FastAPI. The code for this is shown in the main.py in Appendix C. Also, refer to the **number 7** in **Setting up the Web Application** in this duplication manual for properly deploying the model on the website.

For the Web Application:

Setting up the web application:

1. Install Visual Studio Code and open it. Additionally, install all the required libraries such as the Express framework, Socket.IO, MySQL, MQTT, etc. using the **'npm install'** command. For python, install the required libraries such as tensorflow, keras, sci-kit learn, etc. using the **'pip install'** command.
2. Design the front-end user interface using HTML, CSS, and JavaScript. Create pages for the Home, Weather, Water Level (Banaba and Marigold), Advisory, About Us, and Contact.
3. To enable real-time updates for the Notification, Weather and water level of Banaba and Marigold pages, establish a WebSocket connection to the server at "**https://tupmanila.ligtas.tech**".
4. Set up a database file named database.js and ensure that the database configuration is correct.
5. Create a Node.js backend server named '**server.js'** to handle incoming requests. Import the necessary libraries and folders, integrate MQTT, and

make HTTP requests to the FAST API application at **“model.ligtas.tech”** to retrieve predicted outputs.

6. To integrate MQTT, assign the web application as the subscriber and subscribe to specific topics such as *weathersystem, floodnode-02, and floodnode-03*. The format for subscribing is as follows: *'v3/{application name}@{domain}/devices/{topic}/up'*.
7. Create a Python backend server named **‘main.py’** using FAST API to deploy the model. Make sure to assign the correct paths for the model **‘8var2_model.h5’** file and **‘scalar.pkl’** file.
8. To maintain code organization, create a "controllers" folder to handle incoming requests and return responses to clients, and a "routes" folder to determine how the application responds to client requests.

For the database of the Web Application:

1. Create a database named **"ligtcatb_weatherFloodSystem"**. Under the "ligtcatb_weatherFloodSystem " database, create MySQL tables to store and retrieve data for *atmospheric readings, banabanodes, ligtasparameters, marigoldnodes, users, and weather parameters.* This data will be displayed in the web application and used for model predictions.
2. For the "atmospheric" table, create the following fields: "id" as the primary key, "atmosphericPressure" as a Float, and "readingTime" as the current timestamp.

3. For the "banabanodes" table, create the following fields: "id" as the primary key, "banabawaterHeight" and "banabawaterLevel" as Floats, and "readingTime" as the current timestamp.
4. For the "ligtasparameters" table, create the following fields: "id" as the primary key, "temperature", "humidity", "atmosphericPressure", "windSpeed", "windDirection", "rainIntensity", "banabawaterHeight", and "marigoldwaterHeight" as Floats, and "readingTime" as the current timestamp.
5. For the "marigoldnodes" table, create the following fields: "id" as the primary key, "marigoldwaterHeight" and "marigoldwaterLevel" as Floats, and "readingTime" as the current timestamp.
6. For the "users" table, create the following fields: "id" as the primary key, "name", "email", and "hashed password" as varchars.
7. For the "weatherparameters" table, create the following fields: "id" as the primary key, "temperature", "humidity", "atmosphericPressure", "windSpeed", "windDirection", "rainIntensity" as Floats, and "readingTime" as the current timestamp.
8. Assign a default value of -999.0 for all fields in the database tables.

Uploading Code to Namecheap Web Hosting Provider:

1. Create subdomains such as **model.ligtas.tech** and **tupmanila.ligtas.tech** for the FASTAPI and Node.js applications, respectively. This will automatically create folders named according to the subdomains.
2. Upload all the code files for the FASTAPI application to the model.ligtas.tech folder and the Node.js application to the tupmanila.ligtas.tech folder.
3. Ensure that the database configuration is properly set to establish a connection between the web application and the database server.
4. Set up application start-up files for the Node.js and Python FASTAPI applications. Use **server.js** for the Node.js application and **main.py** for the Python application. These start-up files will be responsible for initializing and running the respective applications.
5. Run npm install and pip install to install all the required libraries for the application.
6. The application will start automatically.
7. Finally, verify if the web application is secure. If not, contact the web hosting provider to ensure the website's security.

TROUBLESHOOTING

- In case that any of the sensors are not working, check the connection if it is connected properly. Do a thorough connection inspection especially in the connection of the sensor to the microcontroller (PacketDUINO/Arduino).

- Make sure that there is a gateway nearby, it may be indoor/outdoor to ensure that there is a connection that will be established.
- If the rain gauge is not functioning, check the connection as it is prone to error. Ensure that it will have sufficient supply from the Arduino(5V).

MAINTENANCE

- Monitor the battery consumption from time to time for the system to continuously operate properly.
- Make sure that the circuit box is tightly secure and water-proof to prevent the components from being damaged when it rains, or high-flood occurs.
- Ensure that the gateway is always available and within reach by the weather or flood node.

Appendix G

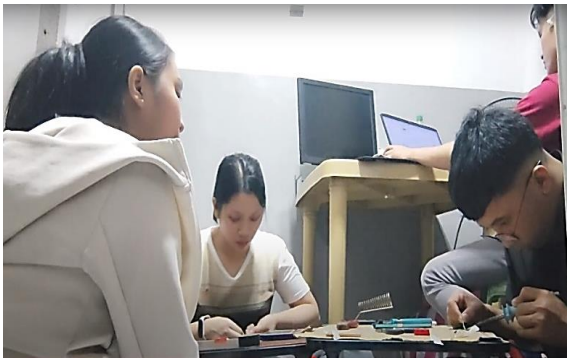
Project Documentation

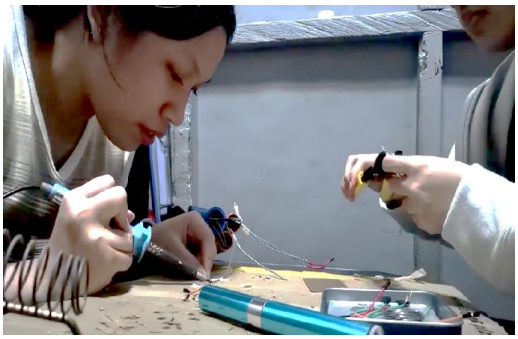
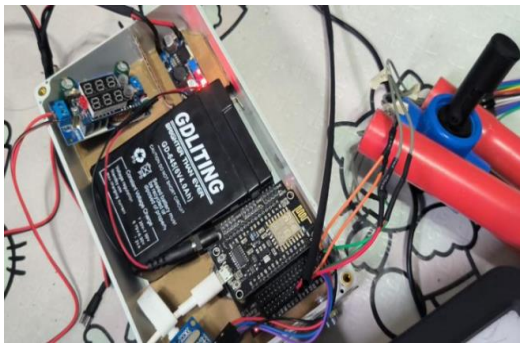
PROJECT PROPOSAL IN LAS PIÑAS CITY

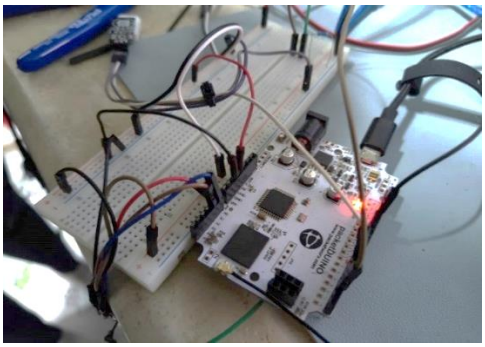
SITE SURVEY

SITE MEASURE

PROJECT CONSTRUCTION





PROJECT INSTALLATION (Weather Station System)


PROJECT INSTALLATION (Flood Monitoring System)

CALIBRATION OF WEATHER SYSTEM AND FLOOD NODES

THESIS DEFENSE

Appendix H
Incident Report

May 13, 2023

Subject: Incident Report - Theft of Flood System and Float Switch Sensor

Dear Panels,

I am writing to formally report a significant incident that transpired at our premises on May 13, 2023. Regrettably, it has come to our attention that our flood system, accompanied by the integral float switch sensor, has been subject to theft. The following are the detailed accounts of the incident:

1. Incident Details:

- Date and Time: May 13, 2023 - 08:00 AM
- Location: Naga Road (Flood System), Banaba Street, and Marigold Bridge (Support of the Float Switch)
- Description: Upon conducting a routine inspection, it was discovered that the flood system and the accompanying float switch sensor were inexplicably missing. The circumstances suggest that an unauthorized individual gained illicit access to our premises and proceeded to remove the aforementioned equipment.

2. Equipment Description:

- Flood System: It is pertinent to note that our flood system encompasses state-of-the-art technology and possesses a range of specific functionalities aimed at detecting and managing potential flooding incidents.
- Float Switch Sensor: The float switch sensor, an integral component of our flood system, serves the vital purpose of accurately detecting water levels and facilitating the appropriate response to such situations.

3. Impact and Mitigation:

- Loss of Functionality: The theft of our flood system and the accompanying float switch sensor significantly compromises our capacity to promptly detect and respond to potential flooding incidents with utmost efficacy.
- Mitigation Plan: We are actively engaging in securing a replacement flood system to expeditiously reinstate the full functionality of our flood detection system.

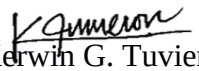
4. Action Taken:

- Incident Report: This formal incident report has been meticulously compiled to comprehensively document the precise details of the theft and initiate the necessary actions required to address this grave matter.
- Internal Communication: All pertinent departments and personnel have been duly informed of this incident, with a specific emphasis on enhancing security awareness and vigilance throughout our organization.

We have attached a file to this report that serves as proof of the installation and functionality of our flood system. This document demonstrates our commitment to mitigating flood risks and ensuring the safety and security of our premises and the surrounding community.

Should you require any supplementary information or should any further developments arise in relation to this incident, please do not hesitate to ask me directly. Your prompt attention and decisive actions in this matter are greatly appreciated.

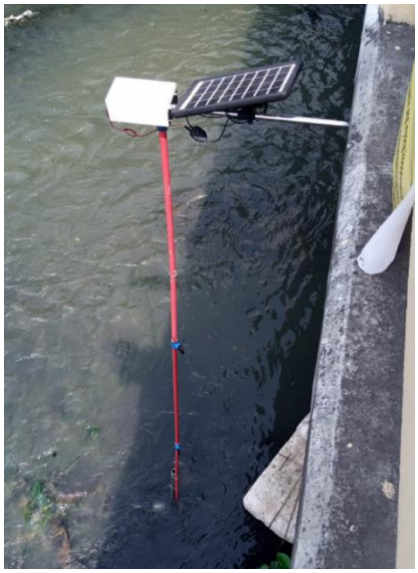
Yours sincerely,



Kerwin G. Tuvieron

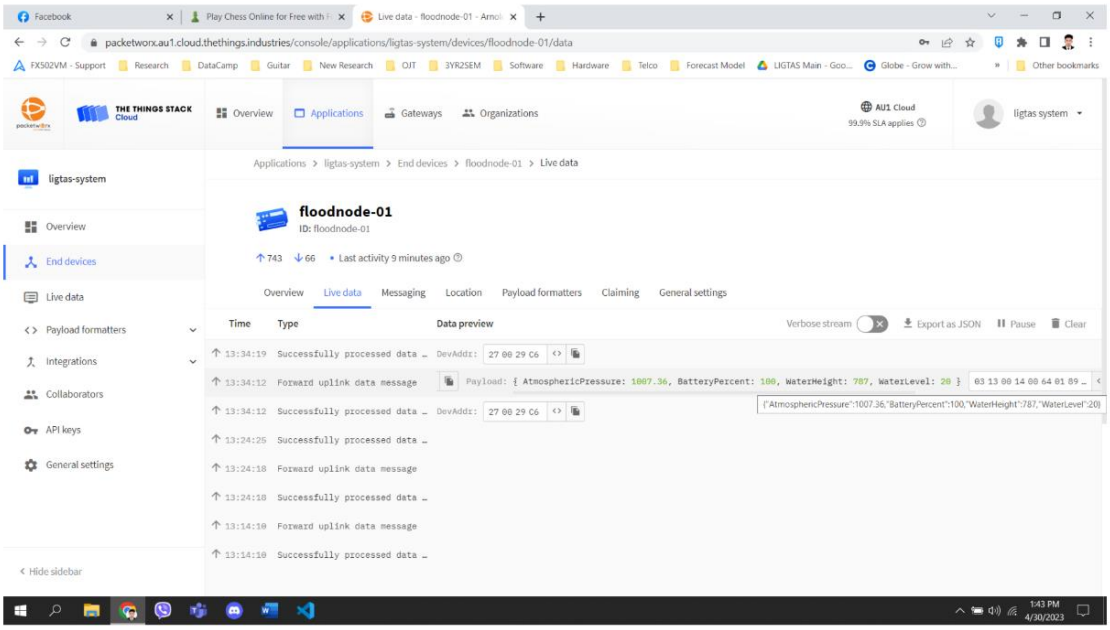
LIGTAS Group Representative

PROOF OF THE INSTALLATION OF THE LIGTAS FLOOD SYSTEM



Flood System at Naga Road, Pulang Lupa Dos was installed successfully

Documentation for the Installation of Flood Monitoring Node at Naga Road, Pulang Lupa Dos with the assistance of the Electrical Engineering Department

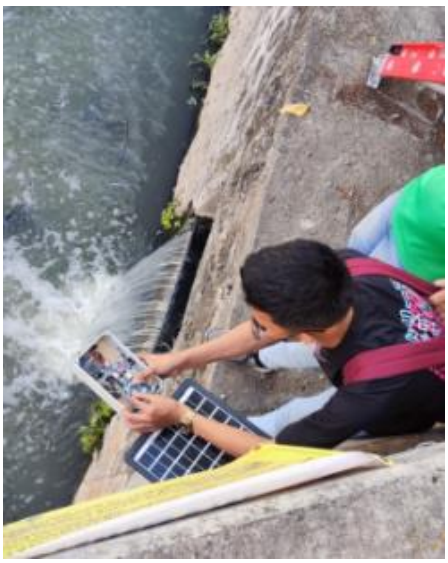


The Flood System at Naga Road was functioning properly, allowing it to successfully send data to The Things Stack dashboard

Flood Monitoring Node at Banaba Street, Phase-7B was installed successfully

Documentation for the Installation of Monitoring Node at Banaba Street, Phase-7B with the assistance of the Electrical Engineering Department

Flood Monitoring Node at Banaba Street, Phase-7B after the incident



Documentation for the Installation of Flood Monitoring Node at Marigold Bridge, Pulang Lupa Dos with the assistance of the Electrical Engineering Department

Flood Monitoring Node at Marigold Bridge, Pulang Lupa Dos after the incident

Appendix I

Proponent's Information

CARLYLE ERIKA E. OPEÑA

RESEARCHER

PERSONAL PROFILE

Detail-oriented and passionate graduating student of Bachelor of Science in Electronics Engineering. With knowledge and a good understanding of communications and emerging technologies.

EDUCATIONAL

Technological University of the Philippines – Manila

2018-2023

BS Electronics Engineering

Quezon City University

2016-2018

Science, Technology, Engineering, and Mathematics

Sampaguita High School

2012-2016

Junior High School

AFFILIATIONS

Organization of Electronics Engineering Students (OECES)

University Organization | Member | 2018 - 2023

Institute of Electronics Engineers of the Philippines (IECEP)

Manila Student Chapter | Board of Director | 2020 - 2021
University Organization | Secretary | 2020 - 2021

CHARACTER REFERENCE

Engr. Edmund Monilar

ECE Faculty, ECE Department, Technological University of the Philippines - Manila

09159453407
edmund.monilar@tup.edu.ph

CONTACT ME AT

09458846611

openacarlyle@gmail.com

B209 L92 St. Martha, Deca Homes Subd, Loma de Gato, Marilao, Bulacan

SKILLS SUMMARY

- Knowledge in Programming Languages: HTML, CSS, Python & MATLAB
- Proficient in Microsoft Office: Word, PowerPoint, Excel, and OneNote
- Circuit design and analysis
- Graphic Design
- Attention to Detail
- Good Organizational Skills

AWARDS RECEIVED

- Finalist in the packetHACKS 2023 Hackathon Competition
- Most Outstanding Board of Director, IECEP - Manila Student Chapter (2020 - 2021)

FAIZAH B. PANGONOTAN

RESEARCHER

PERSONAL PROFILE

An ambitious, dedicated, and hardworking graduating electronics engineering student, possess a deep understanding of circuits and systems, translating theory into practical solutions.

EDUCATIONAL

Technological University of the Philippines – Manila

2019-2023

BS Electronics Engineering

La Consolacion College Manila

2017-2019

Science, Technology, Engineering, and Mathematics

Ramon Magsaysay High School

2013-2017

Junior High School

AFFILIATIONS

Organization of Electronics Engineering Students (OECES)

University Organization | Member | 2019 - 2023

Institute of Electronics Engineers of the Philippines (IECEP)

University Organization | Member | 2019 - 2023

CHARACTER REFERENCE

Engr. Edmund Monilar

ECE Faculty, ECE Department, Technological University of the Philippines - Manila

09159453407

edmund.monilar@tup.edu.ph

CONTACT ME AT

09933759001

bejarfaizah@gmail.com

1337 A. Kalimbas St. Sta Cruz Manila

SKILLS SUMMARY

- Project Management
- Software Development
- Front End Coding
- Computer Literacy
- Strong Communications Skills
- Problem-Solving

AWARDS RECEIVED

- Finalist in the packetHACKS 2023 Hackathon Competition
- Fortinet Training | Network Security Associate 1,2,3

RENJ ASHER ANGELO H. REMIGIO

RESEARCHER

PERSONAL PROFILE

An Electronics Engineering Major with experience in designing Microwave Links and basic Network Configurations. Proficient in training regression predictive models with Keras and Tensorflow and deploying them on website. Experienced in sensor calibrations and integration with LoRaWAN technology.

EDUCATIONAL

Technological University of the Philippines – Manila

2019-2023

BS Electronics Engineering

STI College Fairview

2017-2019

Science, Technology, Engineering, and Mathematics

Caloocan City Business High School

2013-2017

Junior High School

AFFILIATIONS

Organization of Electronics Engineering Students (OECES)

University Organization | Member | 2019 - 2023

Institute of Electronics Engineers of the Philippines (IECEP)

University Organization | Member | 2019 - 2023

CHARACTER REFERENCE

Engr. Edmund Monilar

ECE Faculty, ECE Department, Technological University of the Philippines - Manila

09159453407

edmund.monilar@tup.edu.ph

CONTACT ME AT

09164554196

remigio.renjasherangelo@gmail.com

City of San Jose del monte, Bulacan

SKILLS SUMMARY

- Python, Arduino, MATLAB
- Microwave Link Design and Network Configurations
- Embedded Systems and LoRaWAN Integration
- Machine Learning
- Fast Learner

AWARDS RECEIVED

- President's Lister
- Dean's Lister
- 4th Place STI Tagisan ng Talino Cluster Level (2019)
- Finalist in the packetHACKS 2023 Hackathon Competition

DAVE C. REYES

RESEARCHER

PERSONAL PROFILE

I am seeking a challenging and growth-oriented internship where I can leverage my skills and expertise to make a meaningful contribution to the team. I am highly motivated to work in a fiercely competitive environment where I can gain valuable experience and develop my skills alongside talented professionals.

EDUCATIONAL

Technological University of the Philippines – Manila

2019-2023

BS Electronics Engineering

San Jose Academy

2017-2019

Science, Technology, Engineering, and Mathematics

Governor Andres Pascual College

2013-2017

Junior High School

AFFILIATIONS

Google Developer Student Club Chapter

University Organization | Member | 2021 - 2023

Organization of Electronics Engineering Students

University Organization | Member | 2019 - 2023

Institute of Electronics Engineers of the Philippines

University Organization | Member | 2019 - 2023

CHARACTER REFERENCE

Engr. Edmund Monilar

ECE Faculty, ECE Department, Technological University of the Philippines - Manila

09159453407

edmund.monilar@tup.edu.ph

CONTACT ME AT

09308859036

dave.reyes@tup.edu.ph

107 Quintos Street, Barangay San Jose, Navotas City

SKILLS SUMMARY

- Has a satisfactory critical thinking, decision-making and problem-solving skills
- Intermediate mastery in Python, CSS, HTML Programming
- Intermediate mastery in Arduino Programming
- Detail – oriented and fast learner
- Knowledgeable in using MS Word, MS PowerPoint, and MS Excel

AWARDS RECEIVED

- Dean's Lister
- Finalist in the packetHACKS 2023 Hackathon Competition

KERWIN G. TUVIERON

RESEARCHER

PERSONAL PROFILE

A hardworking and motivated fresh graduate with a BS in Electronics Engineering from Technological University of the Philippines. Seeking opportunities in a company that offers internships in Electronics, Communications, and Programming -related roles to expand my skills and gain valuable real-world experience.

EDUCATIONAL

Technological University of the Philippines – Manila

2019-2023

BS Electronics Engineering

Makati High School

2017-2019

Science, Technology, Engineering, and Mathematics

Makati High School

2013-2017

Junior High School

AFFILIATIONS

Google Developer Student Club Chapter

University Organization | Member | 2021 - 2023

Organization of Electronics Engineering Students

University Organization | Member | 2019 - 2023

Institute of Electronics Engineers of the Philippines

University Organization | Member | 2019 - 2023

CHARACTER REFERENCE

Engr. Edmund Monilar

ECE Faculty, ECE Department, Technological University of the Philippines - Manila

09159453407

edmund.monilar@tup.edu.ph

CONTACT ME AT

09666641832

tuvieronkerwin30@gmail.com

199 J.P. Rizal St. Brgy. Tejeros, Makati City

SKILLS SUMMARY

- MQTT Integration
- Knowledgeable in Python, CSS, HTML Programming
- Intermediate mastery in Tableau
- Intermediate mastery in Node.js, MySQL
- Eagerness to Learn
- Attention to Detail

AWARDS RECEIVED

- Dean's Lister
- Finalist in the packetHACKS 2023 Hackathon Competition