DengueWatch: A System for Real-Time Dengue Monitoring and Forecasting in Iloilo Province

A Special Problem Proposal
Presented to
the Faculty of the Division of Physical Sciences and Mathematics
College of Arts and Sciences
University of the Philippines Visayas
Miag-ao, Iloilo

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science by

AMODIA, Kurt Matthew A.
BULAONG, Glen Andrew C.
ELIPAN, Carl Benedict L.

Francis D. DIMZON
Adviser

April 15, 2025

Abstract

Dengue fever remains a significant public health concern in the Philippines, with cases rising dramatically in recent years. Nationwide outbreaks have placed immense strain on healthcare systems, underscoring the need for innovative approaches to surveillance and response. In Iloilo City, this national trend was reflected in a significant surge, with the Iloilo Provincial Health Office reporting 4,585 cases and 10 fatalities as of August 10, 2023—a 319% increase from the previous year's 1,095 cases and one death. This research focused on developing a centralized system for monitoring and forecasting dengue trends in Iloilo City, incorporating graphical visualizations such as heatmaps, trends, and historical graphs. The study explored the application of artificial intelligence (AI) in dengue prediction, utilizing deep learning models. The performance of the Long Short-Term Memory (LSTM) model was compared with traditional statistical methods, including non-seasonal and seasonal Autoregressive Integrated Moving Average (ARIMA) models and the Kalman Filter. Evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Square Error (RMSE) were used to identify the most effective model for integration into the system. Forecasting was based on climate variables such as temperature, rainfall, relative humidity, and previous monthly case counts. The LSTM model emerged as the best performer with an RMSE of 16.15, demonstrating its suitability for time-series predictions, while the Kalman Filter showed the poorest performance with an RMSE of 38.40. By integrating predictive analytics with real-time data visualization, the proposed system aims to support public health agencies, such as the Department of Health (DOH), by providing actionable insights for proactive intervention strategies. This AI-driven solution enhances traditional outbreak reporting systems by enabling timely, data-informed decisions to mitigate the impact of dengue in the region.

Keywords: ARIMA, artificial intelligence, dengue prediction, LSTM, Kalman Filter, deep learning, climate variables, public health, outbreak mitigation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

From 2020 to 2022, dengue cases declined due to reduced surveillance during the COVID-19 pandemic (WHO, 2023), but cases surged in 2023 as restrictions were lifted. This year saw an increase in dengue outbreaks worldwide, with over five million cases and more than 5,000 deaths reported in over 80 countries (Bosano, 2023). Dengue is endemic in the Philippines, leading to longer and more widespread seasonal outbreaks. Globally, dengue infections have increased significantly, posing a major public health challenge. The World Health Organization reported a tenfold rise in cases between 2000 and 2019, with a peak in 2019 when the disease spread across 129 countries (WHO, 2024).

Iloilo City and Province are intensifying efforts to curb the rising dengue cases (Lena, 2024). As of August 10, 2023, the Iloilo Provincial Health Office recorded 4,585 cases and 10 deaths, a 319% increase from last year's 1,095 cases and one death. Governor Arthur Defensor Jr. confirmed that the province has reached the dengue outbreak threshold based on Department of Health (DOH). Local government units (LGUs) have been informed, and the province's disaster management office is on blue alert, indicating disaster mode. (Perla, 2024)

In Iloilo City, 649 dengue cases were recorded this year 2024, with two deaths. Cases cluster in 40 out of 180 barangays, meaning multiple cases are being reported in these areas over several weeks. The city's health officer, Dr. Roland Jay Fortuna, reported high utilization of non-COVID-19 hospital beds, reaching over 76%, prompting concerns about hospital capacity.

This study explores the monitoring and forecasting of dengue outbreaks by analyzing key factors such as temperature, relative humidity, and historical dengue cases, using different models. The findings aim to provide an advanced, AI-driven alternative for dengue prevention and control, targeting agencies like the Department of Health (DOH). By aligning with the national AI Roadmap, particularly in Iloilo City, this research aspires to improve outbreak responses through cutting-edge technology rather than traditional reporting methods.

1.2 Problem Statement

Dengue remains a critical public health challenge worldwide, with cases increasing due to the easing of COVID-19 restrictions and heightened global mobility. While a temporary decline in cases was observed during the pandemic (2020–2022) due to reduced surveillance efforts, 2023 marked a resurgence, with over five million cases and more than 5,000 deaths reported across 80 countries. In dengue-endemic regions like the Philippines, the threat is particularly severe. In Iloilo City and Province, dengue cases rose by 319% as of August 2023, overwhelming local healthcare systems. This surge strained resources, with over 76% of non-COVID-19 hospital beds occupied by dengue patients, highlighting the urgent need for effective predictive tools. The lack of a reliable system to monitor and forecast dengue outbreaks contributes to delayed interventions, exacerbating public health risks and healthcare burdens in the region.

1.3 Research Objectives

1.3.1 General Objective

This study aims to develop an AI-based dengue forecasting and monitoring system for Iloilo City and Province. The researchers will train and compare multiple deep learning models to predict dengue case trends based on climate data and historical dengue cases to help public health officials in possible dengue case outbreaks.

1.3.2 Specific Objectives

Specifically, this study aims to:

1. Gather dengue data from the Iloilo Provincial Health Office and climate data (including temperature, rainfall, wind, and humidity) from online sources. Combine and aggregate these data into a unified dataset to facilitate comprehensive dengue case forecasting.

2. Evaluate deep learning models for predicting dengue cases using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Squared Error (MSE). Compare the performance of these models to determine the most accurate forecasting approach.

3. Develop a web-based analytics dashboard that integrates a predictive model and provides data management system for dengue cases in Iloilo City and the Province.

4. Assess the usability and effectiveness of the analytics dashboard through structured feedback and surveys involving health professionals and policymakers.

1.4 Scope and Limitations of the Research

This study aims to gather dengue data from the Iloilo Provincial Health Office and climate data from online sources such as PAGASA or weatherandclimate.com. These data will be preprocessed, cleaned, and combined into a unified dataset to facilitate comprehensive dengue case forecasting. However, the study is limited by the availability and completeness of historical data. Inconsistent or missing data points may introduce biases and reduce the quality of predictions. Furthermore, the granularity of the data will be in a weekly format.

To evaluate deep learning models for predicting dengue cases, the study will train and compare the performance of various models, using metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). While these models aim to provide accurate forecasts, their performance is heavily influenced by the quality and size of the dataset. Limited or low-quality data may lead to suboptimal predictions. Additionally, the models cannot fully account for external factors such as public health interventions or socio-economic conditions which may impact dengue transmission dynamics.

The study also involves developing a web-based analytics dashboard that integrates predictive models and provides a data management system for dengue cases in Iloilo City and the Province. This dashboard will offer public health officials an interactive interface to visualize dengue trends, input new data, and identify risk areas. However, its usability depends on feedback from stakeholders, which may vary based on their familiarity with analytics tools. Moreover, external factors such as limited internet connectivity or device availability in remote areas may affect the system's adoption and effectiveness. While the dashboard provides valuable insights, it cannot incorporate all factors influencing dengue transmission, emphasizing the need for ongoing validation and refinement.

1.5 Significance of the Research

This study's development of an AI-based dengue forecasting and monitoring system has wide-reaching significance for various stakeholders in Iloilo City:

- Public Health Agencies: Organizations like the Department of Health (DOH) and local health units in Iloilo City and Province stand to benefit greatly from the system. With dengue predictions, we can help these agencies optimize their response strategies and implement targeted prevention measures in high-risk areas before cases escalate.

- Local Government Units (LGUs): LGUs can use the system to support their disaster management and health initiatives by proactively addressing dengue outbreaks. The predictive insights allow for more efficient planning and resource deployment in barangays and communities most vulnerable to outbreaks, improving overall public health outcomes.

- Healthcare Facilities: Hospitals and clinics, which currently face high bed occupancy rates during dengue season will benefit from early outbreak forecasts that can help in managing patient inflow and ensuring adequate hospital capacity.

- Researchers and Policymakers: This AI-driven approach contributes valuable insights for researchers studying infectious disease patterns and policymakers focused on strengthening the national AI Roadmap. The system's data can support broader initiatives for sustainable health infrastructure and inform policy decisions on resource allocation for dengue control.

- Community Members: By reducing the frequency and severity of outbreaks, this study ultimately benefits the community at large. This allows for timely awareness campaigns and community engagement initiatives, empowering residents with knowledge and preventative measures to protect themselves and reduce the spread of dengue.

Chapter 2

Review of Related Literature

2.1 Dengue

Dengue disease is a tropical and subtropical mosquito-borne viral illness and is a major health concern in the Philippines (Bravo, Roque, Brett, Dizon, & L'Azou, 2014). The majority of individuals with dengue experience no symptoms. Fever is the most common symptom, typically 4 to 7 days after being bitten by an infected mosquito (Zhou & Malani, 2024). In recent years, the trend of dengue cases in the Philippines has shown notable fluctuations, with periodic outbreaks occurring every 3 to 5 years, often influenced by climatic and environmental changes. According to the Department of Health (DOH), the number of reported cases has steadily increased over the past decades, attributed to urbanization, population growth, and inadequate vector control measures (World Health Organization (WHO), 2018). Moreover, studies suggest that El Niño and La Niña events have significant effects on dengue incidence, with warmer temperatures and increased rainfall providing favorable breeding conditions for mosquitoes (Watts, Burke, Harrison, Whitmire, & Nisalak, 2020). The study of Carvajal et. al. highlights the temporal pattern of dengue cases in Metropolitan Manila and emphasizes the significance of relative humidity as a key meteorological factor, alongside rainfall and temperature, in influencing this pattern (Carvajal et al., 2018).

2.2 Outbreak Definition

The definition of an outbreak is a critical factor in disease surveillance, as it determines the threshold at which an unusual increase in cases is considered a public health concern. Studies suggest that outbreak thresholds should be context-specific, given the variability in transmission dynamics across different locations (Runge-Ranzinger et al., 2016). Outbreak definitions defined using the Endemic Channel often base thresholds on 2 standard deviations (SD) above the mean number of historic dengue cases. Other studies (Hemisphere, 2015) also used an alert threshold of a 5-year mean plus 2 standard deviations. A study by (Brady, Smith, Scott, & Hay, 2015) discussed that definitions of dengue outbreaks differ significantly across regions and time, making them inconsistent and incomparable.

From the studies above, this research implements an outbreak definition basing the threshold on 2 standard deviations (SD) above the mean number of historic dengue cases. It is important to note that for future outbreak definitions, additional local context like available hospital space etc., must be taken into account for a more effective outbreak definition.

2.3 Existing System: RabDash DC

RabDash, developed by the University of the Philippines Mindanao, is a web-based dashboard for rabies data analytics. It combines predictive modeling with genomic data, enabling local health authorities to optimize interventions and allocate resources more effectively. RabDash's modules include trend visualization, geographic hotspot mapping, and predictive forecasting, utilizing Long Short-Term Memory (LSTM) models for time-series forecasting (RabDashDC, 2024).

For DengueWatch, RabDash serves as a strong inspiration, particularly in its monitoring, historical trend visualization, and forecasting capabilities. These features align well with the needs of dengue control efforts, providing real-time insights into outbreak trends and enabling more effective, data-driven decision-making. RabDash's architecture is relevant to the DengueDash, as dengue outbreaks similarly require time-series forecasting models. By using LSTM, RabDash effectively models trends in outbreak data, which provides a framework for adapting LSTM to dengue forecasting. Research indicates that LSTM models outperform traditional methods, such as ARIMA and MLP, in handling the complexities of time-dependent epidemiological data (Ligue & Ligue, 2022).

2.4 Deep Learning

The study of (Ligue & Ligue, 2022) highlights how data-driven models can help predict dengue outbreaks. The authors compared traditional statistical methods, such as non-seasonal and seasonal autoregressive integrated moving average (ARIMA), and traditional feed-forward network approach using a multilayer perceptron (MLP) model with a deep learning approach using the long short-term memory (LSTM) architecture in their prediction model. They found that the LSTM model performs better in terms of accuracy. The LSTM model achieved a much lower root mean square error (RMSE) compared to both MLP and ARIMA models, proving its ability to capture complex patterns in time-series data (Ligue & Ligue, 2022). This superior performance is attributed to LSTM's capacity to capture complex, time-dependent relationships within the data, such as those between temperature, rainfall, humidity, and mosquito populations, all of which contribute to dengue incidence (Ligue & Ligue, 2022).

2.5 Kalman Filter

The Kalman Filter is another powerful tool for time-series forecasting that can be integrated into our analysis. It provides a recursive solution to estimating the state of a linear dynamic system from a series of noisy measurements. Its application in epidemiological modeling can enhance prediction accuracy by accounting for uncertainties in the data(Li et al., 2022). Studies have shown that Kalman filters are effective in predicting infectious disease outbreaks by refining estimates based on observed data. A study published in Frontiers in Physics utilized the Kalman filter to predict COVID-19 deaths in Ceará, Brazil. They found that the Kalman filter effectively tracked the progression of deaths and cases, providing critical insights for public health decision-making (Ahmadini et al., 2021). Another research article in PLOS ONE focused on tracking the effective reproduction number (R_t) of COVID-19 using a Kalman filter. This method estimated the growth rate of new infections from noisy data, demonstrating that the Kalman filter could maintain accurate estimates even when case reporting was inconsistent(Arroyo-Marioli, Bullano, Kucinskas, & Rondón-Moreno, 2021).

Our study will compare ARIMA, seasonal ARIMA, Kalman Filter, and LSTM models using our own collected dengue case data along with weather data to identify the most effective model for real-time forecasting.

2.6 Weather Data

The relationship between weather patterns and mosquito-borne diseases is inherently nonlinear, meaning that fluctuations in disease cases do not respond proportionally to changes in climate variables(Colón-González, Fezzi, Lake, & Hunter, 2013) Weather data, such as minimum temperature and accumulated rainfall, are strongly linked to dengue case fluctuations, with effects observed after several weeks due to mosquito breeding and virus incubation cycles. Integrating these lagged weather effects into predictive models can improve early warning systems for dengue control(Cheong, Burkart, Leitão, & Lakes, 2013). A study also suggests that weather-based forecasting models using variables like mean temperature and cumulative rainfall can provide early warnings of dengue outbreaks with high sensitivity and specificity, enabling predictions up to 16 weeks in advance(Hii, Zhu, Ng, Ng, & Rocklöv, 2012).

We will utilize weather data, including variables such as temperature, rainfall, and humidity, as inputs for our dengue forecasting model. Given the strong, nonlinear relationship between climate patterns and dengue incidence, these weather variables, along with their lagged effects, are essential for enhancing prediction accuracy and providing timely early warnings for dengue outbreaks.

2.7 Chapter Summary

This chapter reviewed key literature relevant to our study, focusing on existing systems, predictive modeling techniques and the role of weather data in forecasting dengue outbreaks. We examined systems like RabDash DC, which integrates predictive modeling with real-time data to inform public health decisions, providing a foundational structure for our Dengue Watch System. Additionally, deep learning approaches, particularly Long Short-Term Memory (LSTM) networks, were highlighted for their effectiveness in time-series forecasting, while alternative methods such as ARIMA and Kalman Filters were considered for their ability to model complex temporal patterns and handle noisy data.

The literature further underscores the significance of weather variables—such as temperature and rainfall—in forecasting dengue cases. Studies demonstrate that these variables contribute to accurate outbreak prediction models. Leveraging these insights, our study will incorporate both weather data and historical dengue case counts to build a reliable forecasting model.

Chapter 3

Research Methodology

This chapter lists and discusses the specific steps and activities that will be performed to accomplish the project. The discussion covers the activities from pre-proposal to Final SP Writing.

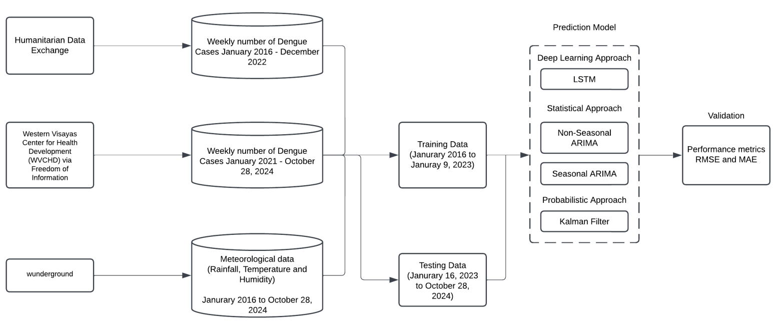

Figure 3.1: Workflow for forecasting the number of weekly dengue cases

This summarizes the workflow for forecasting the number of weekly dengue cases. This workflow focuses on using statistical, deep learning, and probabilistic models to forecast the number of reported dengue cases. The approach involves deploying several models for prediction, including ARIMA and Seasonal ARIMA as statistical approaches, LSTM as a deep learning approach, and the Kalman Filter as a probabilistic approach. These methods are compared with each other to determine the most accurate model.

3.1 Research Activities

3.1.1 Gather Dengue Data and Climate Data to Create a Complete Dataset for Forecasting

Acquisition of Dengue Case Data

The historical dengue case dataset used in this study was obtained from the Humanitarian Data Exchange and the Western Visayas Center for Health Development (WVCHD) via Freedom of Information (FOI) requests. The decision to use weekly intervals was driven by the need for precision and timeliness in capturing fluctuations in dengue cases and weather conditions. Dengue transmission is influenced by short-term changes in weather variables such as rainfall and temperature, which impact mosquito breeding and virus transmission cycles. A weekly granularity allowed the model to better capture these short-term trends, enabling more accurate predictions and responsive public health interventions.

Moreover, using a weekly interval provided more data points for training the models compared to a monthly format. This is particularly critical in time series modeling, where larger datasets help improve the robustness of the model and its ability to generalize to new data. Also, the collection of weather data was done by utilizing web scraping techniques to extract weekly weather data (e.g., rainfall, temperature, and humidity) from Weather Underground (wunderground.com).

Data Fields

- **Time.** Represents the specific year and week corresponding to each entry in the dataset.
- **Rainfall.** Denotes the observed average rainfall, measured in millimeters, for a specific week.
- **Humidity.** Refers to the observed average relative humidity, expressed as a percentage, for a specific week.
- **Max Temperature.** Represents the observed maximum temperature, measured in degrees Celsius, for a specific week.
- **Average Temperature.** Represents the observed average temperature, measured in degrees Celsius, for a specific week.

- **Min Temperature.** Represents the observed minimum temperature, measured in degrees Celsius, for a specific week.
- **Wind.** Represents the observed wind speed, measured in miles per hour (mph), for a specific week.
- **Cases.** Refers to the number of reported dengue cases during a specific week.

Data Integration and Preprocessing

The dengue case data was integrated with the weather data to create a com prehensive dataset, aligning the data based on corresponding timeframes. The dataset undergoed a cleaning process to address any missing values, outliers, and inconsistencies to ensure its accuracy and reliability. To ensure that all features and the target variable were on the same scale, a MinMaxScaler was applied to normalize both the input features (climate data) and the target variable (dengue cases).

Exploratory Data Analysis (EDA)

- Analyzed trends, seasonality, and correlations between dengue cases and weather factors.
- Created visualizations like time series plots and scatterplots to highlight relationships and patterns in the data.

Outbreak Detection

To detect outbreaks, we computed the outbreak threshold value of dengue cases using the formula,

$$\text{Outbreak Threshold Value} = \mu + 2\sigma \tag{3.1}$$

where μ is the historical mean and σ is the standard deviation.

3.1.2 Develop and Evaluate Deep Learning Models for Dengue Case Forecasting

The deep learning models were developed and trained to forecast weekly dengue cases using historical weather data (rainfall, temperature, wind, and humidity) and dengue case counts. The dataset was normalized and divided into training and testing sets, ensuring temporal continuity to avoid data leakage. The methodology for preparing and training the model are outlined below.

Data Preprocessing

The raw dataset included weekly aggregated weather variables (rainfall, temperature, wind, humidity) and dengue case counts. The "Time" column was converted to a datetime format to ensure proper temporal indexing. To standardize the data for training, MinMaxScaler was employed, normalizing the feature values and target variable to a range of 0 to 1. This step ensured that the models could efficiently process the data without being biased by feature scaling differences.

LSTM Model

To prepare the data for LSTM, a sliding window approach was utilized. Sequences of weeks of normalized features were constructed as input, while the dengue case count for the subsequent week was set as the target variable. This approach ensured that the model leveraged temporal dependencies in the data for forecasting.

The LSTM model was designed using the TensorFlow and Keras libraries. The architecture comprised the following layers:

- Input Layer: Accepting sequences of weeks with three features (rainfall, max temperature, and humidity).
- LSTM Layer: A single LSTM layer with 64 units and ReLU activation, capturing temporal dependencies and feature interactions.
- Dense Output Layer: A fully connected layer with a single neuron to predict the dengue cases for the next week.

The model was trained for 100 epochs implementing early stopping with a batch size of 1, enabling fine-grained weight updates. The training dataset consisted of 80% of the sequences, while the remaining 20% was used as the test set to evaluate model performance. Validation loss was monitored during training to assess model generalization.

The training process was conducted using three distinct window sizes (5 weeks, 10 weeks, and 20 weeks) to determine the optimal sequence length of weeks to input into the LSTM model for improved forecasting performance.

After training, predictions on both the training and test datasets were rescaled to their original scale using the inverse transformation of MinMaxScaler. Model performance was evaluated using the mean squared error (MSE), root mean squared error (RMSE) and mean absolute error (MAE).

Hyperparameter Tuning

After identifying the optimal window size, it is saved and used to generate the final data sequences, which are then utilized during hyper-parameter tuning.

To enhance the performance of the LSTM model in predicting dengue cases, Bayesian Optimization was employed using the Keras Tuner library. The tuning process aimed to minimize the validation loss (mean squared error) by adjusting key model hyper-parameters. The search space is summarized below:

LSTM units:

- min value: 32
- max value: 256
- step: 32
- sampling: linear

Learning Rate:

- min value: 0.0001
- max value: 0.01
- step: None
- sampling: log

The tuner was instanstiated with:

- **max trials = 20:** Limiting the search to 20 different configurations
- **executions per trial = 3:** Running each configuration thrice to reduce variance
- **validation split = 0.2:** Reserving 20% of the training data for validation

ARIMA

The ARIMA model was employed to forecast weekly dengue cases using historical weather data (rainfall, max temperature, and humidity) as exogenous variables and historical case counts as the primary dependent variable. The dataset was split into training (80%) and testing (20%) sets. To determine the optimal configuration for the ARIMA model, a grid search was conducted over the following parameter ranges:

- p (autoregressive order): 0 to 3
- d (differencing order): 0 to 2
- q (moving average order): 0 to 3

The combinations of these parameters were evaluated by fitting an ARIMA model for each set of (p, d, q) values. The model's performance was assessed using the mean squared error (MSE) between the predicted and actual dengue cases in the test set. The combination yielding the lowest MSE was selected as the optimal parameter configuration.

The fitted ARIMA model was used to forecast weekly dengue cases for the test dataset. Predictions were directly assigned to the PredictedCases column in the test dataset.

Steps to Create the ARIMA Model:

1. **Data Preprocessing:**Prepare the dataset by handling any missing values and scaling the data if necessary to improve model convergence and stability.
2. **Hyperparameter Tuning:** Use a grid search on potential ARIMA parameters (p, d, q) to identify the configuration that minimizes error. The optimal parameters were found to be **(1, 2, 2)**.

3. **Model Training:**
    - Set the number of iterations to 400 to ensure thorough training and convergence.
    - Train the ARIMA model on 80% of the data and reserve 20% for testing.

Seasonal ARIMA (SARIMA)

1. **Data Preprocessing**
    - Handle missing values through interpolation or imputation.
    - Normalize or standardize features to ensure stable training.
    - Split data into training (80%) and testing (20%) sets while maintaining temporal continuity.
2. **Seasonality Analysis**
    - Perform time series decomposition to examine trend, seasonality, and residual components.
    - Identify seasonality using autocorrelation plots and spectral analysis.
    - A periodicity of **52 weeks** was detected, justifying the use of a seasonal model.
3. **Hyperparameter Tuning**
    - Conduct a grid search to optimize SARIMA parameters $(p, d, q)(P, D, Q)[S]$.
    - Determine optimal configuration for seasonal and non-seasonal components.
    - Verify stationarity through Augmented Dickey-Fuller (ADF) test.
4. **Model Training**
    - Fit the SARIMA model on the training dataset, incorporating exogenous variables such as rainfall, temperature, and humidity.
    - Set a maximum number of iterations to ensure convergence.
    - Monitor model diagnostics (residual analysis) to confirm the absence of autocorrelation in residuals.
5. **Forecasting and Validation**
    - Generate out-of-sample forecasts for future dengue cases.

- Compare predicted values against actual data to assess real-world applicability.
- Visualize results with line plots and confidence intervals.

Kalman Filter:

- Input Variables: The target variable (Cases) was modeled using three regressors: rainfall, max temperature, and humidity.
- Training and Testing Split: The dataset was split into 80% training and 20% testing to evaluate model performance.
- Observation Matrix: The Kalman Filter requires an observation matrix, which was constructed by adding an intercept (column of ones) to the regressors.

The Kalman Filter's EM method was employed for training, iteratively estimating model parameters over 10 iterations. The smooth method was used to compute the smoothed state estimates for the training data. Observation matrices for the test data were constructed similarly, ensuring compatibility with the trained model.

3.2 Kalman Filter

- **Input Variables**: The target variable (Cases) was modeled using three regressors: rainfall, max temperature, and humidity.
- **Training and Testing Split**: The dataset was split into 80% training and 20% testing to evaluate model performance.
- **Observation Matrix**: The Kalman Filter requires an observation matrix, which was constructed by adding an intercept (column of ones) to the regressors.

The Kalman Filter's EM method was employed for training, iteratively estimating model parameters over 10 iterations. The smooth method was used to compute the smoothed state estimates for the training data. Observation matrices for the test data were constructed similarly, ensuring compatibility with the trained model.

3.3 Kalman Filter Methodology with Matrix Calculations

Measurement Acquisition: Obtain the measurement: (z_k) of the system's state with associated confidence. This measurement matrix provides a noisy observation of the true state.

The dataset was split into training and test sets to evaluate the Kalman Filter's performance and generalizability:

- **Training Set**: 80% of the data was used for training, enabling the Kalman Filter model to capture key patterns.
- **Test Set**: The remaining 20% of the data was reserved for testing.

Prediction Step:

- Predict the next state:

$$\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1} + Bu_k$$

- Update the state covariance:

$$P_{k|k-1} = AP_{k-1|k-1}A^T + Q$$

where Q is the process noise covariance matrix.

Compute Residual: Calculate the residual:

$$y_k = z_k - H\hat{x}_{k|k-1}$$

where H is the observation matrix. This residual represents the new information from the measurement.

Scaling Factor (Kalman Gain):

- Compute the Kalman Gain:

$$K_k = P_{k|k-1}H^T \left(HP_{k|k-1}H^T + R\right)^{-1}$$

where R is the measurement noise covariance matrix.

- The Kalman Gain determines the weight of the measurement relative to the prediction.

State Update:

- Update the state estimate:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k y_k$$

blending the prediction and measurement.

Uncertainty Update:

- Update the state covariance:

$$P_{k|k} = (I - K_k H) P_{k|k-1}$$

where I is the identity matrix.

3.3.1 Integrate the Predictive Model into a Web-Based Data Analytics Dashboard

Dashboard Design and Development

- Design an intuitive, user-friendly web-based dashboard incorporating:
  - Interactive visualizations of yearly dengue case trends.
  - Data input and update forms for dengue and weather data.
  - Map display of dengue cases in each district in Iloilo City

Model Integration and Deployment

- Deploy the best-performing model within the dashboard as a backend service to enable real-time or periodic forecasting.

3.3.2 System Development Framework

The Agile Model is the birthchild of both iterative and incremental approaches in Software Engineering. It aims to be flexible and effective at the same time by being adaptable to change. It's also important to note that small teams looking to construct and develop projects quickly can benefit from this kind of methodology. As the Agile Method focuses on continuous testing, quality assurance is a guarantee since bugs and errors are quickly identified and patched.

3.3.3 Design, Building, Testing, and Integration

Design and Developlment

After brainstorming and researching the most appropriate type of application to accommodate both the prospected users and the proposed solutions, the team has decided to proceed with a web application. Given the time constraints and available resources, we believe this is the most pragmatic and practical move. The next step is to select modern and stable frameworks that align with the fundamental ideas we have learned at the university. The template obtained from WVCHD and Iloilo Provincial Epidemiology and Surveillance Unit was meticulously analyzed to create use cases and develop a preliminary well-structured database that adheres to the requirements needed to produce a quality application. The said use cases serve as the basis of general features. Part by part, these are converted into code, and with the help of selected libraries and packages, it resulted in the desired outcome that may still modified and extended since it is continuously being developed.

Testing and Integration

Each feature will be rigorously user-tested to ensure quality assurance, with particular emphasis on prerequisite features, as development cannot progress properly if these fail. Moreover, integration between each feature serves as a pillar for a cohesive user experience. Presently, we have not been able to use performance metrics to measure the system's performance, as developing and connecting the core features is the utmost priority.

3.4 Development Tools

3.4.1 Software

Github

GitHub is a cloud-based platform that tracks file changes using Git, an open-source version control system (*About GitHub and Git - GitHub Docs*, n.d.). It is used in the project to store the application's source code, manage the system's source version control, and serve as a repository for the Latex files used in the actual research.

Visual Studio Code

Visual Studio Code is a free, lightweight, and cross-platform source code editor developed by Microsoft (*Why Visual Studio Code?*, 2021). As VS Code supports this project's programming and scripting languages, it was chosen as the primary source code editor.

Django

Django is a free and open-sourced Python-based web framework that offers an abstraction to develop and maintain a secure web application. As this research aims to create a well-developed and maintainable application, it is in the best interest to follow an architectural pattern that developers and contributors in the future can understand. Since Django adheres to Model-View-Template (MVT) that promotes a clean codebase by separating data models, business logic, and presentation layers, it became the primary candidate for the application's backbone.

Next.js

A report by Statista (2024) claims that React is the most popular front-end framework among web developers. However, React has limitations that can be a nuisance in rapid software development, which includes routing and performance optimizations. This is where Next.js comes in—a framework built on top of React.

It offers solutions for React's deficiency, making it a rising star in the framework race.

Postman

As the application heavily relies on the Application Programming Interface (API) being thrown by the backend, it is a must to use a development tool that facilitates the development and testing of the API. Postman is a freemium API platform that offers a user-friendly interface to create and manage API requests (*What is Postman? Postman API Platform*, n.d.).

3.4.2 Hardware

The web application is continuously being developed on laptop computers with minimum specifications of an 11th-generation Intel i5 CPU and 16 gigabytes of RAM.

3.4.3 Packages

Django REST Framework

Django Rest Framework (DRF) is a third-party package for Django that provides a comprehensive suite of features to simplify the development of robust and scalable Web APIs (Christie, n.d.). These services include Serialization, Authentication and Permissions, Viewsets and Routers, and a Browsable API .

Leaflet

One of the features of the web application is the ability to map the number of cases using a Choropleth Map. Leaflet is the only free, open-sourced, and most importantly, stable JavaScript package that can do the job. With its ultra-lightweight size, it offers a comprehensive set of features that does not trade off performance and usability (*Leaflet — an open-source JavaScript library for interactive maps*, n.d.).

Chart.js

Another feature of the application is to provide users with informative, approachable data storytelling that is easy for everyone to understand. The transformation of pure data points and statistics into figures such as charts is a big factor. Thus, there is a need for a package that can handle this feature without compromising the performance of the application. Chart.js is a free and open-source JavaScript package that is made to meet this criteria as it supports various types of charts (*Chart.js*, n.d.).

Tailwind CSS

Using plain CSS in production-quality applications can be counterproductive. Therefore, CSS frameworks were developed to promote consistency and accelerate the rapid development of web applications (Joel, 2021). One of these is Tailwind, which offers low-level utility classes that can be applied directly to each HTML element to create a custom design (*Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.*, n.d.). Given the limited timeline for this project, using this framework is a wise choice due to its stability and popularity among developers.

Shadcn

Shadcn offers a collection of open-source UI boilerplate components that can be directly copied and pasted into one's project. With the flexibility of the provided components, Shadcn allows developers to have full control over customization and styling. Since this is built on top of Tailwind CSS and Radix UI, it is supported by most modern frontend frameworks, including Next.js (Shadcn, n.d.).

Zod

Data validation is integral in this web application since it will handle crucial data that will be used for analytical inferences and observations. Since Zod is primarily used for validating and parsing data, it ensures proper communication between the client and the server (Zod, n.d.).

3.5 Calendar of Activities

A Gantt chart showing the schedule of the activities is included below. Each bullet represents approximately one week of activity.

Table 3.1: Timetable of Activities for 2024

Activities	Aug	Sept	Oct	Nov	Dec
Project Initiation and Team Formation	●●				
Literature Review and Data Gathering	●●	●●●●			
Data Cleaning and Feature Selection		●●		●	●
Creating System Dashboard		●●	●●●●	●	
Analysis and Interpretation of Results			●		●
Documentation	●●	●●●●	●●●●	●●●●	●●●●

Table 3.2: Timetable of Activities for 2025

Activities	Jan	Feb	Mar	Apr	May
Create Admin Dashboard	●	●●●			
Integrate the Best Model to the System	●	●●●●			
Extend Features to Accommodate a National Setting		●	●●		
User Testing			●●	●	
System Deployment				●●●	
Documentation	●●	●●●●	●●●●	●●●●	●●●●

Chapter 4

Results and Discussion/System Prototype

4.1 Data Gathering

The data for dengue case prediction was gathered from a variety of reliable sources, enabling a comprehensive dataset spanning from January 2011 to October 2024. This dataset includes 720 rows of data, each containing weekly records of dengue cases along with corresponding meteorological variables, such as rainfall, temperature, and humidity.

1. Dengue Case Data: The primary source of historical dengue cases came from the Humanitarian Data Exchange and the Western Visayas Center for Health Development (WVCHD). The dataset, accessed through Freedom of Information (FOI) requests, provided robust case numbers for the Western Visayas region. The systematic collection of these data points was essential for establishing a reliable baseline for model training and evaluation.

2. Weather Data: Weekly weather data was obtained by web scraping from Weather Underground, allowing access to rainfall, temperature, wind, and humidity levels that correlate with dengue prevalence.

```
data.head()
```

	Time	Rainfall	MaxTemperature	AverageTemperature	MinTemperature	Wind	Humidity	Cases
0	2011-01-03	9.938571	29.444400	25.888890	23.888900	11.39	86.242857	5
1	2011-01-10	8.587143	30.000000	26.705556	24.444444	7.32	88.028571	4
2	2011-01-17	5.338571	30.000000	26.616667	25.000000	7.55	84.028571	2
3	2011-01-24	5.410000	30.555556	26.483333	20.555556	10.67	80.971429	7
4	2011-01-31	2.914286	28.333333	25.283333	18.650000	11.01	74.885714	2

Figure 4.1: Snippet of the Combined Dataset

```
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Time                720 non-null    datetime64[ns]
 1   Rainfall            720 non-null    float64
 2   MaxTemperature      720 non-null    float64
 3   AverageTemperature  720 non-null    float64
 4   MinTemperature      720 non-null    float64
 5   Wind                720 non-null    float64
 6   Humidity            720 non-null    float64
 7   Cases               720 non-null    int64
dtypes: datetime64[ns](1), float64(6), int64(1)
memory usage: 45.1 KB
```

Figure 4.2: Data Contents

4.2 Exploratory Data Analysis

From the summary above, the dataset consists of 720 weekly records with 8 columns:

- **Time.** Weekly timestamps (e.g. ”2011-w1”)
- **Rainfall.** Weekly average rainfall (mm)
- **MaxTemperature, AverageTemperature, MinTemperature.** Weekly temperature data (C)
- **Wind.** Wind speed (m/s)
- **Humidity.** Weekly average humidity (%)
- **Cases.** Reported dengue cases

	Time	Rainfall	MaxTemperature	AverageTemperature	MinTemperature	Wind	Humidity	Cases
count	720	720.000000	720.000000	720.000000	720.000000	720.000000	720.000000	720.000000
mean	2017-12-02 11:22:00	13.957499	32.191142	28.110319	25.038472	6.172417	81.609442	23.744444
min	2011-01-03 00:00:00	0.000000	-14.600000	24.494444	12.222222	1.910000	73.185714	0.000000
25%	2014-06-21 06:00:00	1.270000	31.666667	27.504167	25.000000	4.117500	79.885713	5.000000
50%	2017-12-07 12:00:00	4.318000	32.222222	28.161111	25.000000	5.725000	81.771429	12.000000
75%	2021-05-11 18:00:00	10.414000	32.777778	28.751389	25.555556	7.860000	83.503571	26.000000
max	2024-10-28 00:00:00	445.008000	58.333333	30.916667	32.222222	19.200000	89.571429	319.000000
std	NaN	35.448846	2.616379	0.999800	1.291659	2.446703	2.831674	37.144813

Figure 4.3: Dataset Statistics

From the statistics in figure 4.3, the number of cases ranges from 0 to 319. The average number of dengue cases per week is 23.74, with a median of 12 cases and a standard deviation of 37.14. The distribution is highly skewed, with some weeks experiencing significant number of cases (up to 319 cases). Rainfall shows a wide variation (0 to 445mm), while temperature remains relatively stable, with an average of 28.1 degree celsius. Humidity levels ranges from 73% to 89% with a mean of 81.6%.

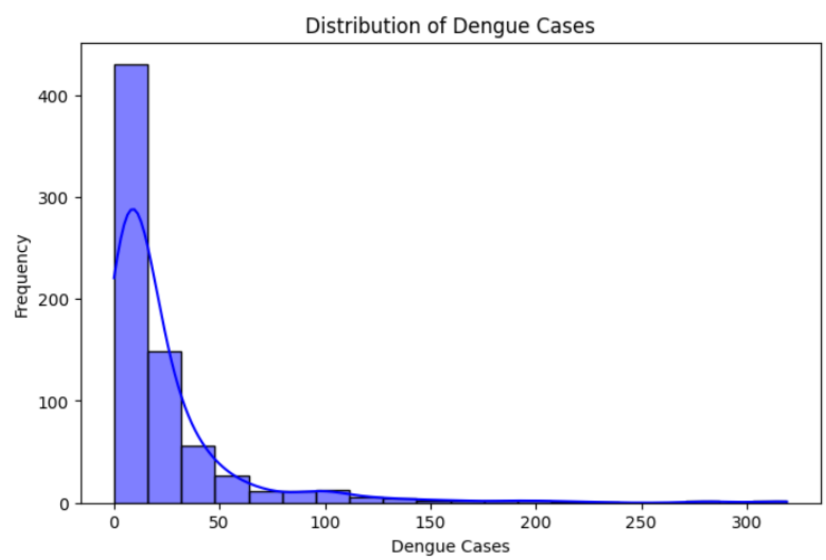

Figure 4.4: Distribution of Dengue Cases

In figure 4.4, a histogram of dengue cases shows a right-skewed distribution, indicating that most weeks experience low case counts, while a few weeks record outbreaks.
To further analyze the distribution, dengue cases were categorized into different intervals (Figure 4.5): 0-5 cases, 6-15 cases, 16-30 cases, 31-100 cases and 101+

cases. The majority of weeks falls within the 0-5 cases and 6-15 cases categories, indicating that most weeks have low dengue cases. Meanwhile, weeks with 101+ cases are rare, suggesting that extreme outbreaks are not frequent.

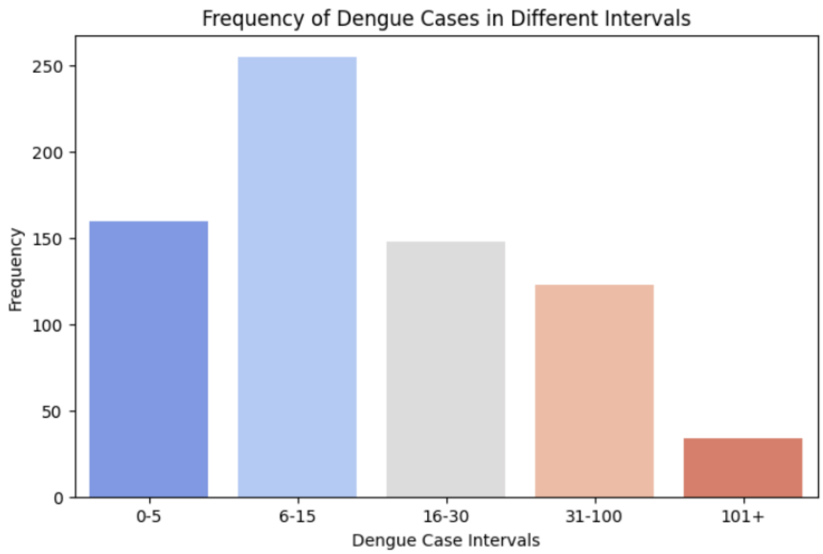

Figure 4.5: Frequency of Dengue Cases in Different Intervals

Figure 4.6 illustrates the trend of weekly dengue cases over time. The data reveals periodic spikes in the number of cases, suggesting a seasonal pattern in dengue cases. Notably, peak cases are observed during certain periods approximately 3 years, potentially aligning with specific climatic conditions such as increased rainfall or temperature changes. This underscores the importance of incorporating climate variables into the forecasting model.

Figure 4.7 shows the ranking of correlation coefficients between dengue cases and selected features, including rainfall, humidity, maximum temperature, average temperature, minimum temperature, and wind speed. Among these, rainfall exhibits the highest positive correlation with dengue cases (correlation coefficient 0.13), indicating that increased rainfall may contribute to higher cases counts. This aligns with existing studies suggesting that stagnant water from heavy rainfall creates breeding grounds for mosquitos. It is followed by humidity (0.10), suggesting that higher humidity levels may enhance mosquito reproduction, leading to more dengue cases. Temperature has a weak to moderate positive correlation with dengue cases, with maximum temperature (0.09) showing a stronger relationship than average and minimum temperature.

Figure 4.8 shows the ranking of correlation coefficients between dengue cases and selected features, with the addition of lagged effects. The analysis reveals no

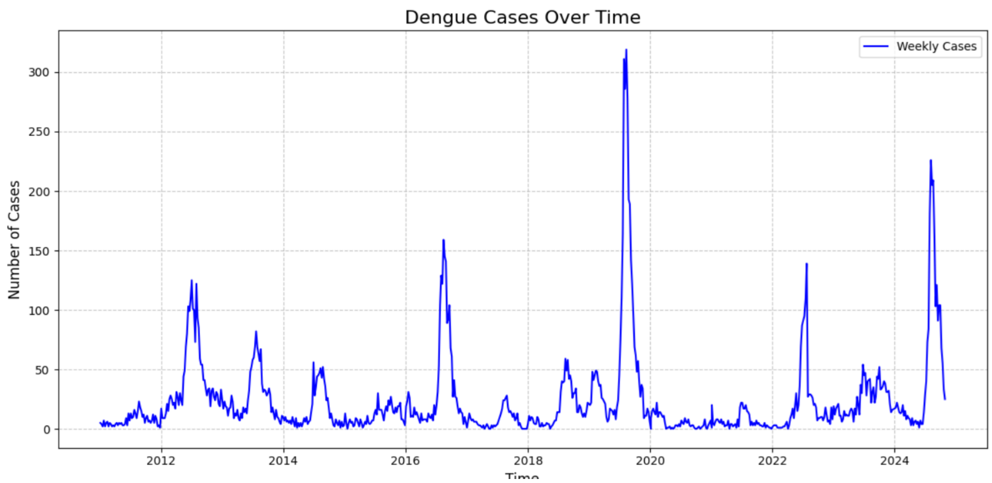

Figure 4.6: Trend of Dengue Cases

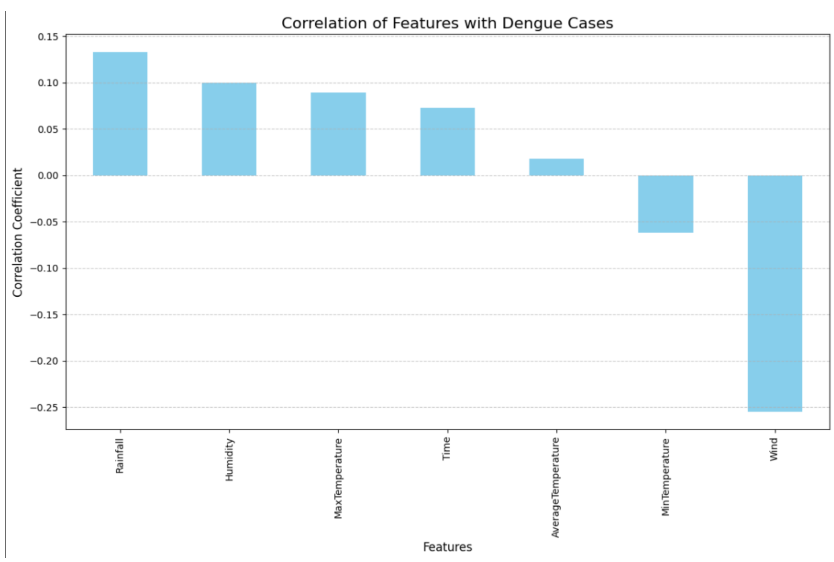

Figure 4.7: Ranking of Correlations

improvement in correlation when lagged variables are compared to direct observations. This suggests that the observed values of rainfall, humidity, and maximum temperature remain the most significant predictors for dengue case forecasting. Overall, the exploratory data analysis highlights the significance of rainfall, humidity, and max temperature variables in dengue case forecasting.

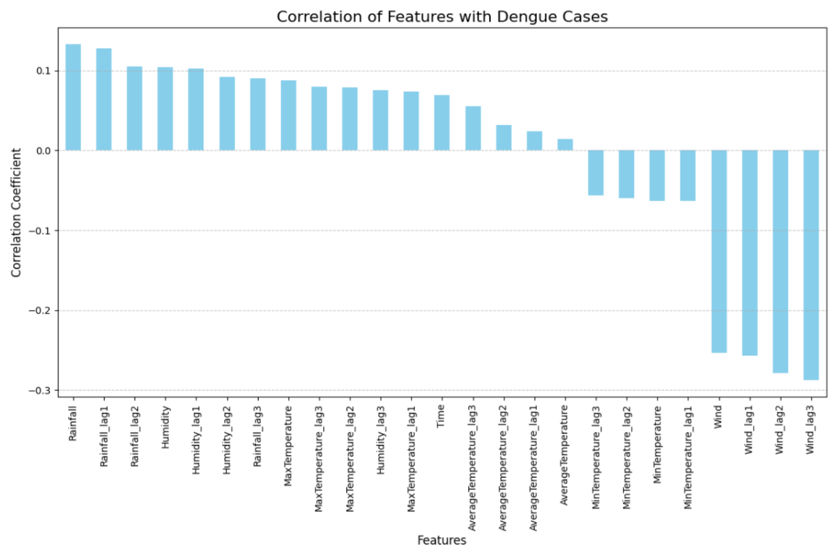

Figure 4.8: Ranking of Correlations (with lagged effects)

4.3 Outbreak Detection

To identify outbreaks, we calculated the outbreak threshold value using the historical mean as the endemic channel. The threshold is determined using the formula:

$$
\begin{aligned}
\text{Outbreak Threshold Value} &= \mu + 2\sigma && (4.1)\\
&= 23.744444 + 2(37.144813) && (4.2)\\
&= 23.744444 + 74.289626 && (4.3)\\
&= 98.03407 && (4.4)
\end{aligned}
$$

where μ is the historical mean and σ is the standard deviation.

This result indicates that dengue cases exceeding 98 in Iloilo City can be considered an outbreak. However, it is important to note that this threshold serves only as a baseline. Additional parameters, such as the number of hospital beds available in the city, must be considered to compute a more effective threshold and develop an appropriate response strategy.

4.4 Model Training

The proposed Dengue Watch system utilized four distinct models to forecast weekly dengue cases: Long Short-Term Memory (LSTM) networks, Autoregressive Integrated Moving Average (ARIMA), Seasonal ARIMA (SARIMA), and Kalman Filter. Each model was trained on a dataset containing 720 weeks of historical dengue cases from 2011 to 2024, with meteorological variables such as max temperature, humidity, and rainfall.

Using SARIMA and LSTM for dengue forecasting requires an adaptive approach due to seasonal changes and long-term trends. Dengue case data is updated every month, and weather data can be extracted manually every week. By continuously monitoring performance, incorporating external factors, and updating the model regularly (preferably monthly or semi-annually), forecasting accuracy can be maintained. If drastic environmental or epidemiological changes occur, more frequent retraining is necessary. This ensures that public health interventions remain proactive, effectively mitigating dengue outbreaks.

To optimize predictive performance, hyperparameter tuning was conducted individually for each model, refining parameters to achieve the most accurate and reliable forecasts. Following training, the models were rigorously evaluated against the dataset using a set of key performance metrics, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE).

The table below provides a summary and comparative analysis of each model's results across these metrics, offering insights into the strengths and limitations of each forecasting technique for dengue case prediction in Iloilo City. The acceptable threshold for Mean Absolute Error (MAE) in forecasting dengue cases for it to be considered accurate can vary depending on the context. However, related studies often serve as benchmarks, with commonly cited acceptable values ranging from 20 to 30. For this study, we have established a threshold of 15 to emphasize the significance of accurate dengue prediction.

Model	MSE	RMSE	MAE
LSTM	**260.93**	**16.15**	**9.30**
Seasonal ARIMA (2, 0, 2) (0, 1,1)	**1109.69**	**33.31**	**18.09**
ARIMA (1, 2, 2)	**1521.48**	**39.01**	**25.80**
Kalman Filter	**1474.82**	**38.40**	**22.34**

Table 4.1: Comparison of Models

4.4.1 LSTM Model

The LSTM model architecture consisted of an input layer, a single LSTM layer with 64 units and ReLU activation, followed by a dense layer with a single output neuron to predict the dengue case count. Key hyperparameters included:

- Window Size: 5, 10, and 20 weeks, representing the time steps used in the sequence data for each prediction.
- Epochs: 100 epochs were used for training, balancing sufficient training time with computational efficiency also implementing early stopping to avoid overfitting.
- Batch Size: 1, allowing the model to process one sequence at a time, which is beneficial for small datasets but increases training time.
- Optimizer: The Adam optimizer was chosen for its adaptive learning capabilities and stability in training. A custom learning rate of 0.001 was set to ensure gradual convergence and minimize risk of overfitting.

The dataset was split into training and test sets to evaluate the model's performance and generalizability:

- **Training Set:** 80% of the data (572 sequences) was used for model training, enabling the LSTM to learn underlying patterns in historical dengue case trends and their relationship with weather variables.
- **Test Set:** The remaining 20% of the data (148 sequences) was reserved for testing

The training process was conducted using three distinct window sizes—5 weeks, 10 weeks, and 20 weeks—to identify the optimal sequence length of weeks for input into the LSTM model, thereby enhancing forecasting performance. The following plots illustrate the performance of the model in predicting dengue cases for each of the specified window sizes.

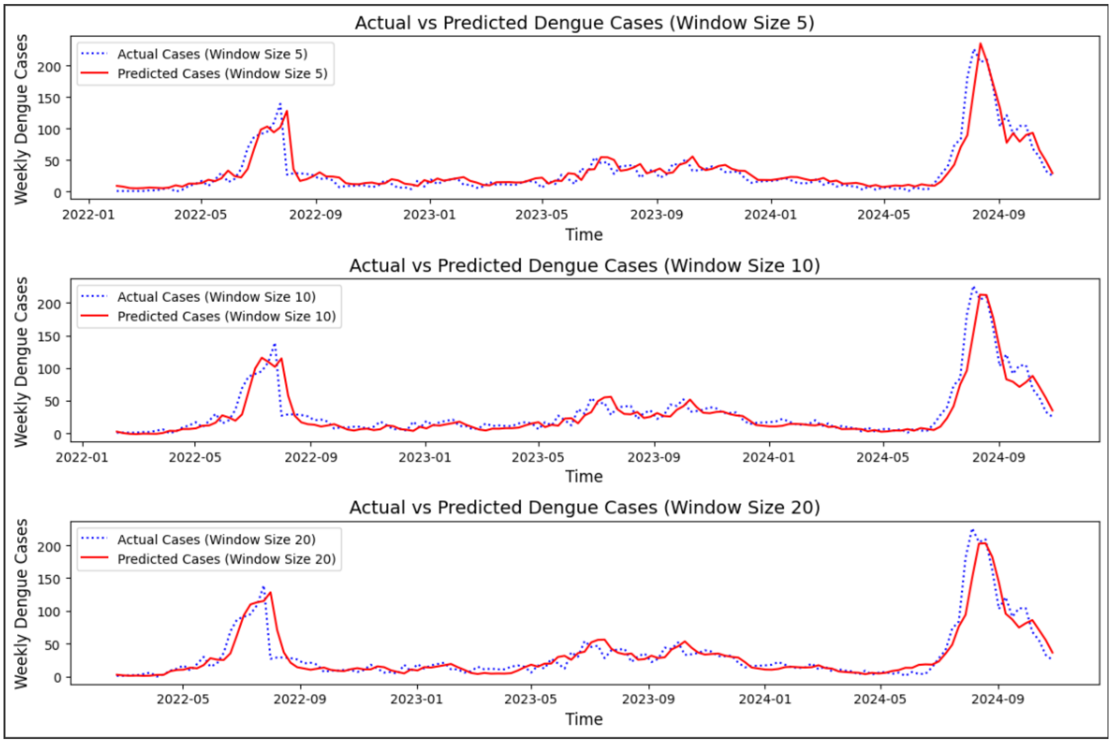

Figure 4.9: Comparison of Window Sizes

The evaluation metrics included Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and R^2 Score, which assess the accuracy of the model's predictions.

Window Size	**MSE**	**RMSE**	**MAE**	**R^2**
5	**274.70**	**16.57**	**9.57**	**0.84**
10	**260.93**	**16.15**	**9.30**	**0.85**
20	**297.11**	**17.24**	**9.84**	**0.83**

Table 4.2: Comparison of Window Sizes

The results indicate that a window size of 10 weeks provides the most accurate predictions, as evidenced by the lowest MSE (260.93) and RMSE (16.15) values. Although the 10-week window size yields the lowest MAE (9.30), the 5-week window follows closely with 9.57, while the 20-week window is slightly higher at 9.84. These differences are relatively small, especially between the 5- and 10-week windows, indicating that the average prediction error remains fairly consistent across different window sizes.

Furthermore, the R^2 score of 0.85 for the 10-week window indicates that 85%

of the variability in the target variable (cases) is explained by the independent variables (the inputs) in the model, making it a reliable configuration overall. In contrast, the 5-week and 20-week windows yield R^2 scores of 0.84 and 0.83, respectively, reflecting marginally lower explanatory power.

This suggests that using a 10-week sequence length effectively balances the model's ability to capture temporal dependencies with predictive accuracy, without unnecessarily increasing model complexity or introducing additional noise from longer sequences.

Hyperparameter Tuning

Using the 10-week sequence length identified as the optimal window size in preliminary experiments, the dataset was reshaped accordingly and served as the input for hyperparameter tuning. The tuning process was conducted using the *Bayesian Optimization* approach provided by the `Keras Tuner` library, targeting the minimization of validation loss (Mean Squared Error). The key hyperparameters explored during the tuning were:

- **LSTM units:** 96
- **Learning Rate:** 0.006

Although the tuning process successfully identified a configuration that minimized the validation loss during training, it did not result in improved performance on the test set. In fact, the model's evaluation metrics slightly declined when compared to the baseline model trained with manually selected hyperparameters.

Model	**MSE**	**RMSE**	**MAE**	**R^2**
Before tuning	**260.93**	**16.15**	**9.30**	**0.85**
After tuning	**317.70**	**17.82**	**10.42**	**0.81**

Table 4.3: Comparison of Model Performance Before and After Tuning (Using window size = 10)

This outcome suggests that the tuned model may have overfitted the validation split, a common occurrence when working with relatively small datasets. It is also possible that the default or manually chosen configuration was already close to optimal in terms of generalization. Furthermore, although the tuning search space was reasonably defined, it may have excluded other more effective hyperparameter combinations. These results emphasize the importance of critically evaluating tuning results and underscore that automated hyperparameter optimization does not always guarantee better model performance on unseen data.

Training and Testing Data Division for ARIMA and Seasonal Arima

Both models utilized an **80%-20% split** to evaluate generalizability:

- **Training Set**: 80% of the data was used for training, allowing the models to learn underlying patterns in the dataset.
- **Test Set**: 20% of the data was reserved for testing, providing an unbiased assessment of the models' performance on unseen data.

4.4.2 ARIMA Model

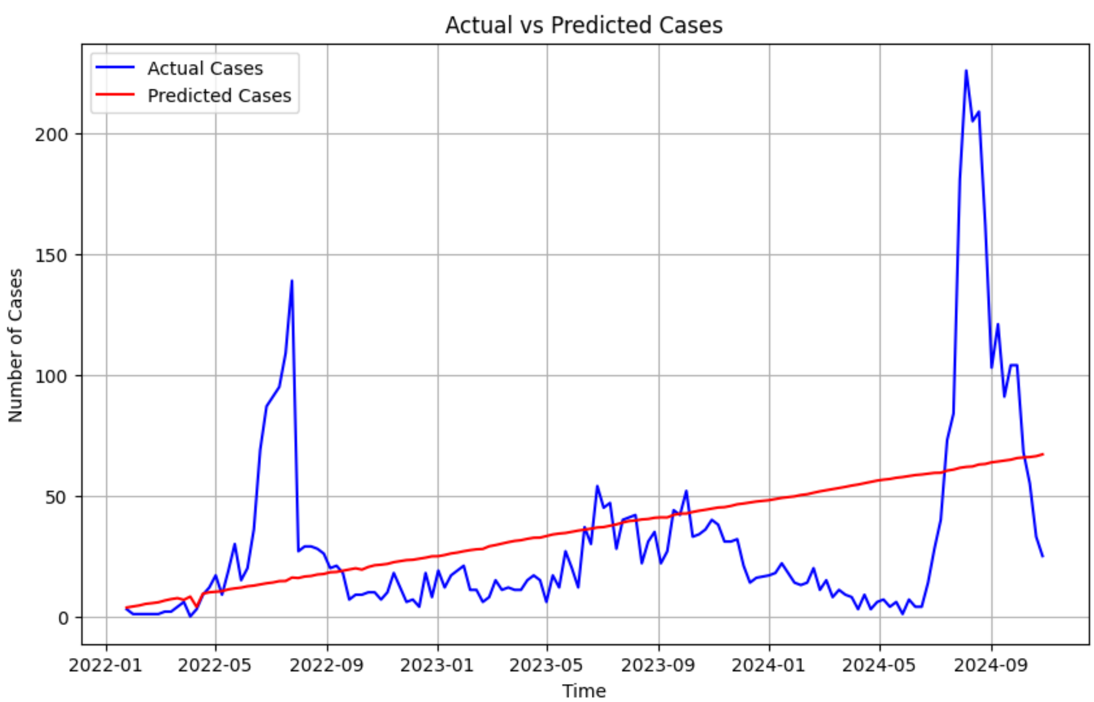

Figure 4.10: ARIMA Prediction Results for Test Set

The ARIMA model was developed to capture non-seasonal trends in the data. To determine the best model configuration, grid search was used to explore various combinations of ARIMA parameters, ultimately selecting **ARIMA(1, 2, 2)**. The model was iteratively refined over **400 iterations** to ensure convergence to an optimal solution.

The model's performance was assessed using regression metrics to evaluate its forecasting capability:

- Mean Squared Error (MSE): Quantifies average squared prediction error.
- Root Mean Squared Error (RMSE): Measures average prediction error on the data's original scale.
- Mean Absolute Error (MAE): Measures the average magnitude of the absolute errors between the predicted and actual values.

The ARIMA model yielded the following error metrics:

- **MSE (Mean Squared Error)**: 1521.48
- **RMSE (Root Mean Squared Error)**: 39.01
- **MAE (Mean Absolute Error)**: 25.80

4.4.3 Seasonal ARIMA (SARIMA) Model

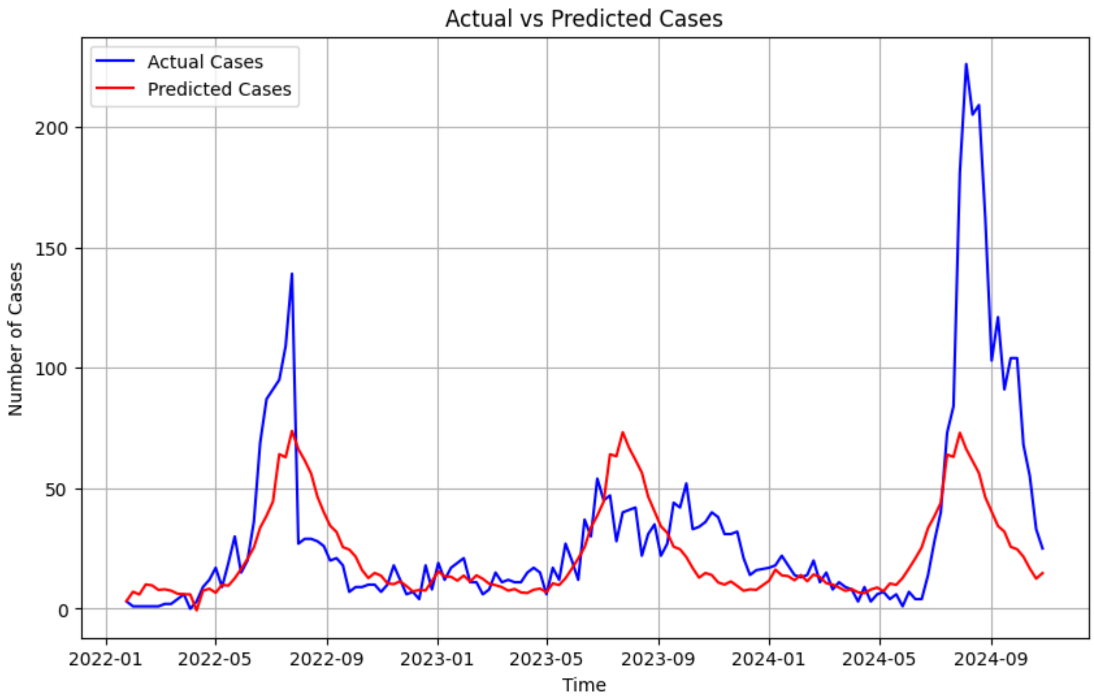

Figure 4.11: Seasonal ARIMA Prediction Results for Test Set

This model incorporates seasonal parameters, which were tuned using grid search to find the best configuration: **SARIMA(2, 0, 2)(0, 1, 1)[52]**. As with ARIMA, **400 iterations** were applied to ensure a robust fit.

The model's performance was assessed using regression metrics to evaluate its forecasting capability:

- Mean Squared Error (MSE): Quantifies average squared prediction error.
- Root Mean Squared Error (RMSE): Measures average prediction error on the data's original scale.
- Mean Absolute Error (MAE): Measures the average magnitude of the absolute errors between the predicted and actual values.

The SARIMA model yielded the following error metrics:

- **MSE**: 1109.69

- **RMSE**: 33.31
- **MAE**: 18.09

The SARIMA model outperformed the ARIMA model in terms of lower MSE and RMSE values, indicating its effectiveness in capturing the seasonal patterns in the data.

4.4.4 Kalman Filter Model

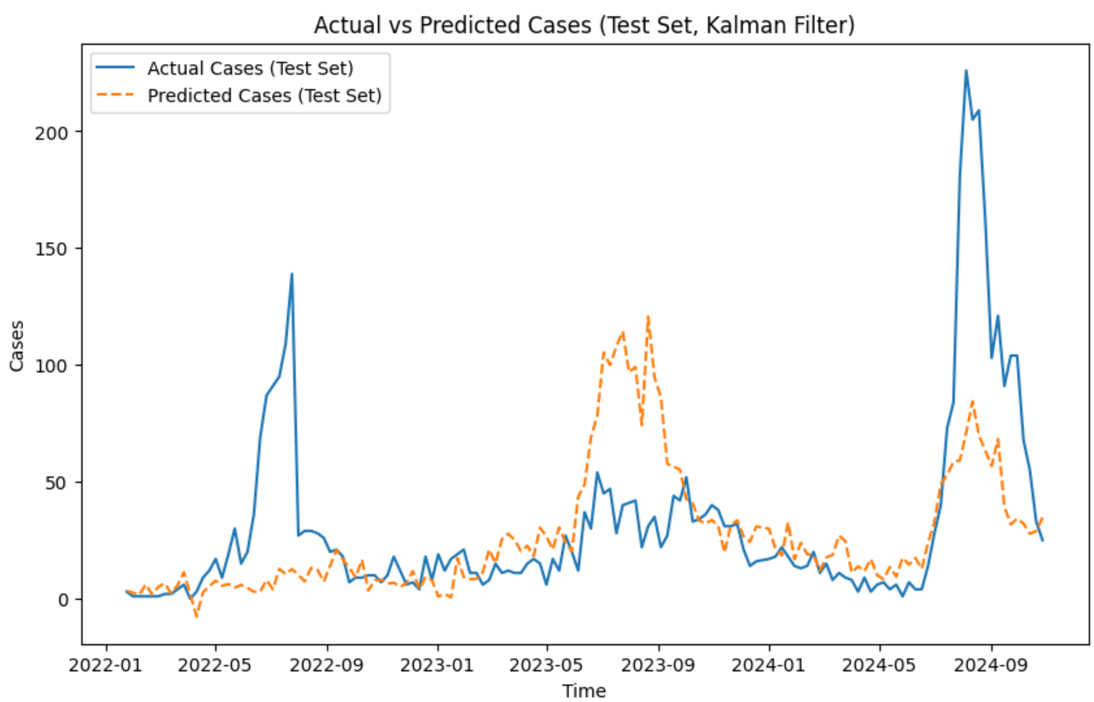

Figure 4.12: Kalman Filter Prediction Results for Test Set

Model Evaluation: Upon testing, the Kalman Filter produced the following error metrics:

$$\text{MSE} = 1474.82, \quad \text{RMSE} = 38.40, \quad \text{MAE} = 22.34$$

These results indicate the model's performance in predicting dengue cases, where lower errors suggest a better fit to the observed data.

4.5 Preliminary System Requirements

4.5.1 Backend Requirements

Database Structure Design

Determining how data flows and how it would be structured is crucial in creating the system as it defines how extendible and flexible it would be for future features and updates. Thus, creating a comprehensive map of data ensures proper normalization that eliminates data redundancy and improves data integrity. Figure 4.13 depicts the designed database schema that showcases the relationship between the application's entities.

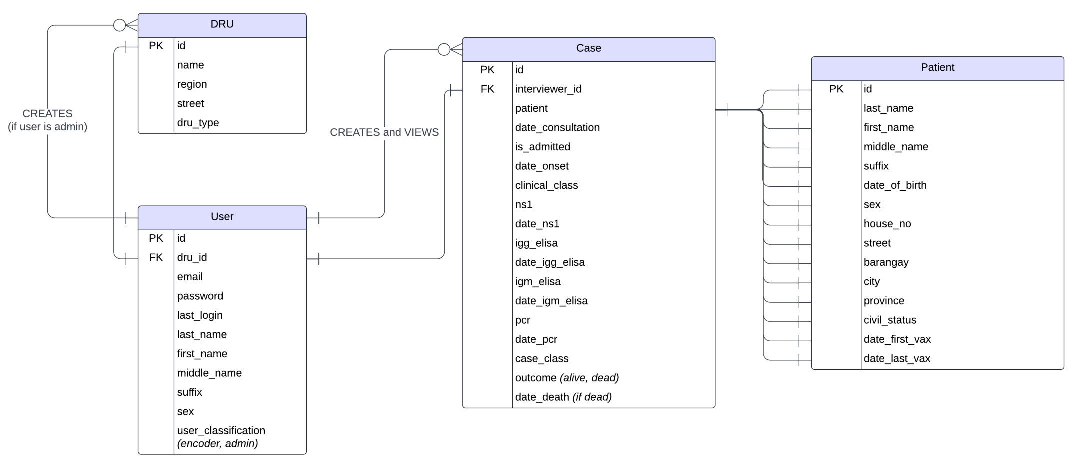

Figure 4.13: Entity-Relationship Database Schema Hybrid Diagram for DengueDash Database Structure

4.5.2 User Interface Requirements

Admin Interface

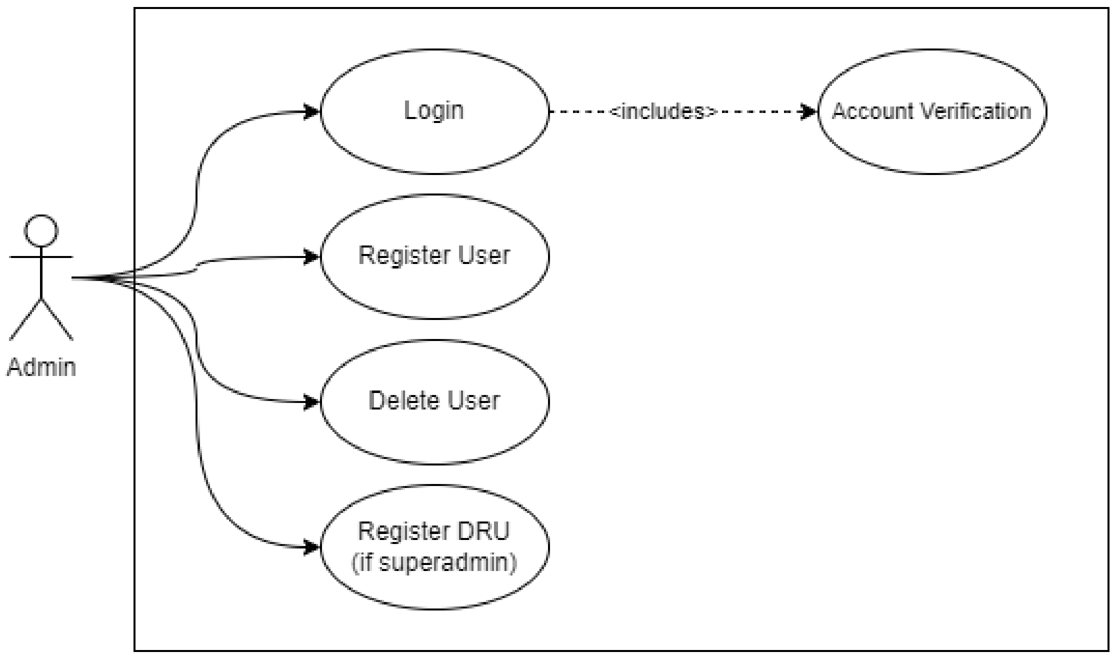

Figure 4.14: Use Case Diagram for Admin

Figure 4.14 shows the possible tasks that the admin can do in the application. To protect the integrity of data, only the admins can register and delete accounts. Both account creation and deletion will be done within the application.

Encoder Interface

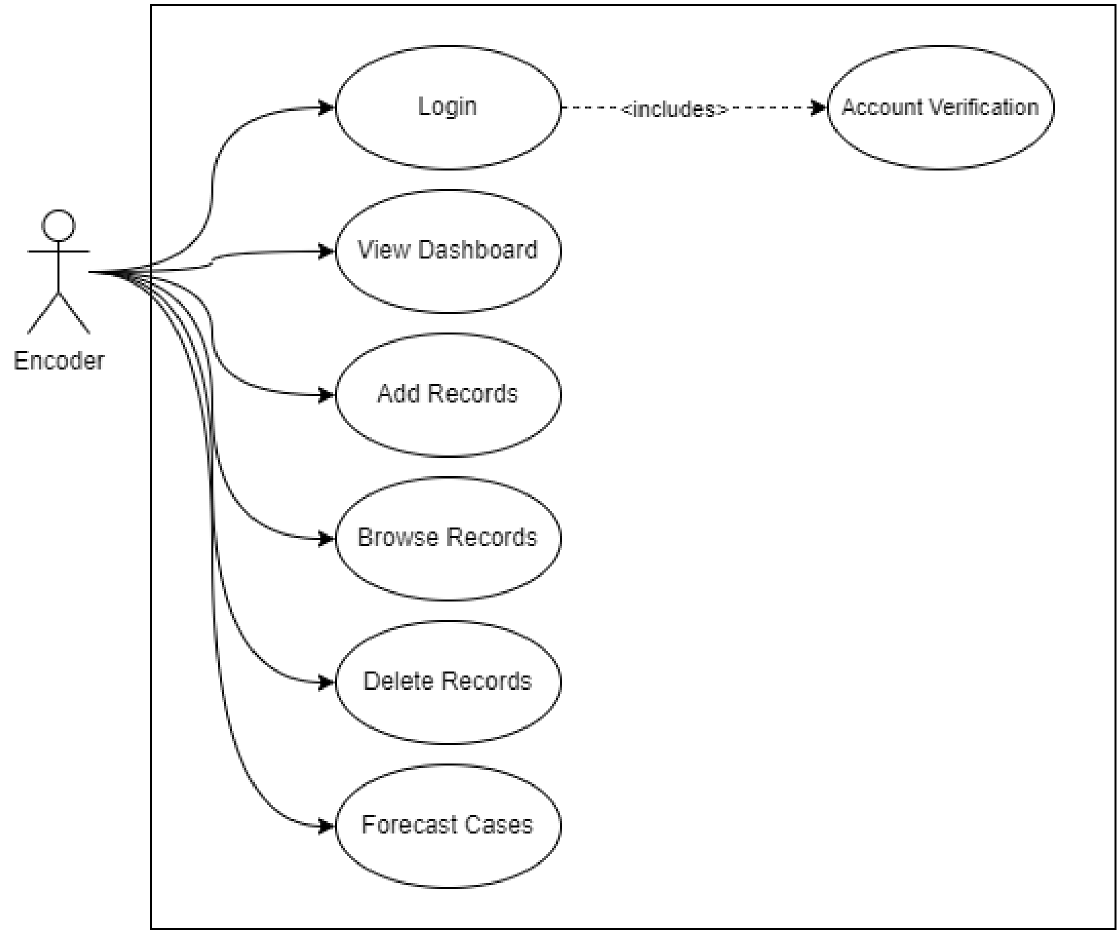

Figure 4.15: Use Case Diagram for Encoder

Figure 4.15, on the other hand, illustrates the use cases for the system's primary users. Since only the admin accounts can register a user, the registration process is not part of it. Instead, the main features, which include reporting and viewing records, are the only permitted actions for this type of user. The said processes can be done in the application by filling out a form with details required for each dengue case. As data is entered, it will be consolidated for model training and used for further forecasting of dengue cases.

4.5.3 Security and Validation Requirements

Password Encryption

Storing passwords as plain text in the database is a disgrace and a mortal sin in production. It is important to implement precautionary methods such as hashing and salting, followed by encryption with a strong algorithm, to prevent bad actors from using the accounts for malicious transactions. By default, Django generates a unique random salt for each password and encrypts it with Password-Based Key Derivation Function 2 (PBKDF2) with a SHA256 hash function. Utilizing these techniques ensures that in the event of a data breach, cracking these passwords would be time-consuming and useless for the attackers.

Authentication

DengueWatch utilizes JSON Web Tokens (JWT) to authenticate the user. Since the mechanism operates in a stateless manner, tokens are served only after a successful login, eliminating the need for the server to keep a record of the token, which is vulnerable to session hijacking. In addition, these tokens are signed with a secret key, ensuring they have not been tampered with.

Data Validation

Both the backend and frontend should validate the input from the user to preserve data integrity. Thus, Zod is implemented in the latter to help catch invalid inputs from the user. By doing this, the user can only send proper requests to the server which streamlines the total workflow. On the other hand, Django has also a built-in validator that checks the data type and ensures that the input matches the expected format on the server side. These validation processes ensure that only valid and properly formatted data is accepted, which reduces the risk of errors and ensures consistency across the web application.

4.5.4 Testing Process

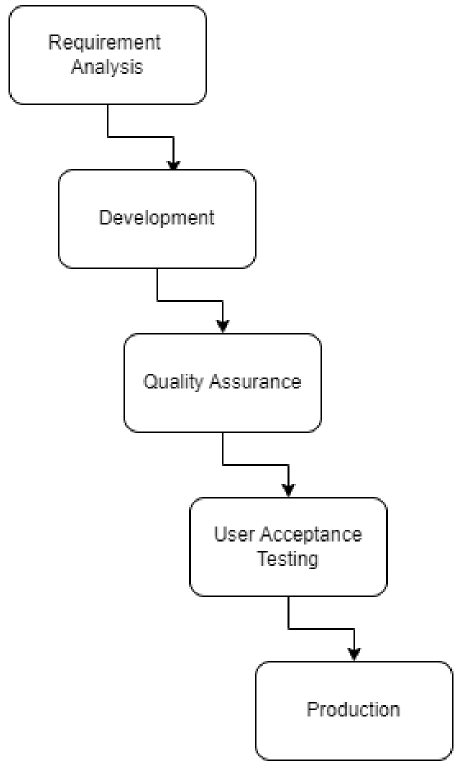

Figure 4.16: Testing Process for DengueWatch

As the system requirements and functionalities have been mentioned above, it is important to implement testing to validate the system's performance and efficacy. Since dengue reports include confidential information, anonymized historical dengue reports were used to train the model and create the foundational architecture of the system. By using functional tests, data validation and visualization can be ensured for further continual improvements. Security testing is also important as it is needed to safeguard confidential information when the system is deployed. It includes proper authentication, permission views, and mitigating common injection attacks. Finally, a user acceptance test from the prospected users, in this case, the Iloilo City Epidemiology and Surveillance Unit, is crucial to assess its performance and user experience. It enables the developers to confirm if the system meets the needs of the problem, and once confirmed, it will be deployed and further evaluated to ensure stability and reliability in live operation.

4.6 System Prototype

4.6.1 Guest Interface

The Guest Interface is intended for all visitors of the web application. It shows the related statistics for dengue cases in a particular area and time. As the system is still in its testing phase, the data converted into charts shown in Figure 4.17 are generated from Python's Faker library.

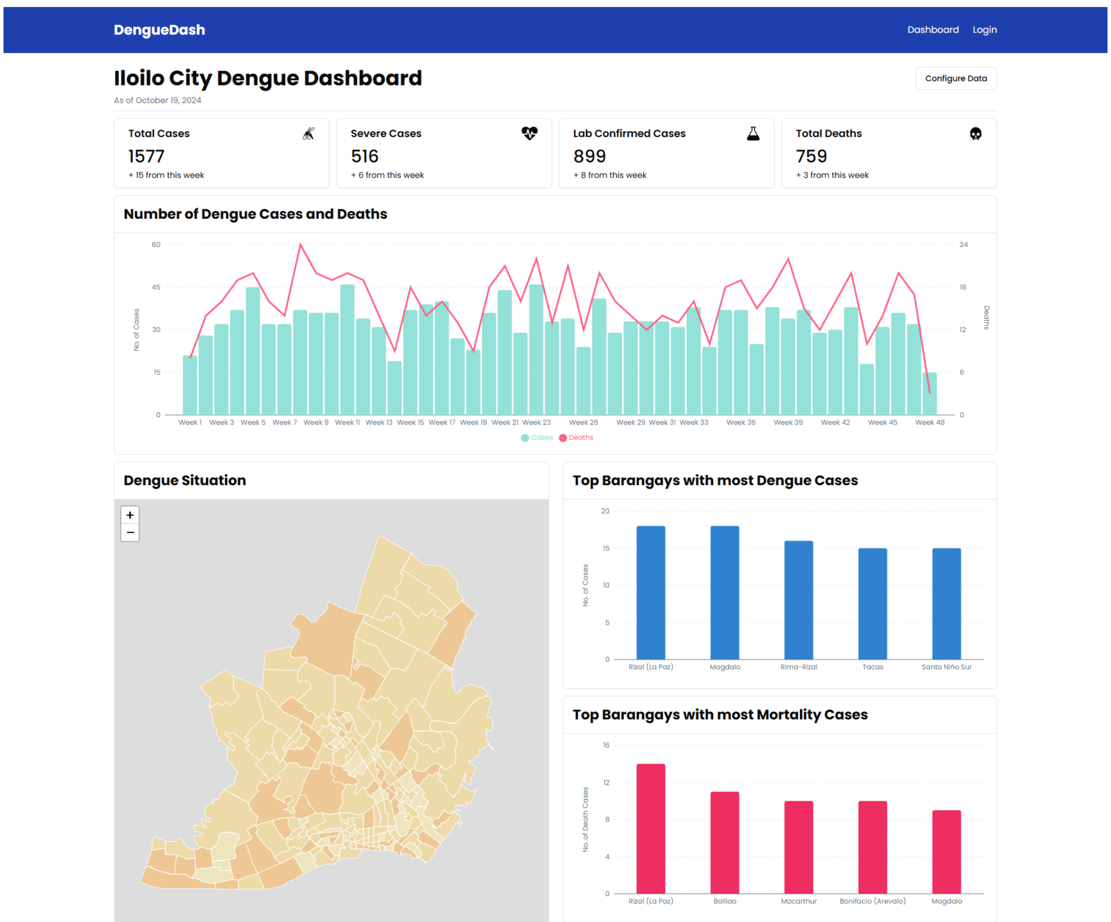

Figure 4.17: Dashboard for Guests

4.6.2 Personnel Interface

User Authentication, and Login

To protect the data's integrity in production, it has been decided that the registration process will not be visible. Instead, an admin must register a user using a different interface. As of the moment, registering a user is done using API via Postman. In the login process, the system implements HTTP-only cookies that contains the JSON Web Tokens (JWT) to protect against XSS attacks. After proper credentials have been provided, it will redirect to the user's home page.

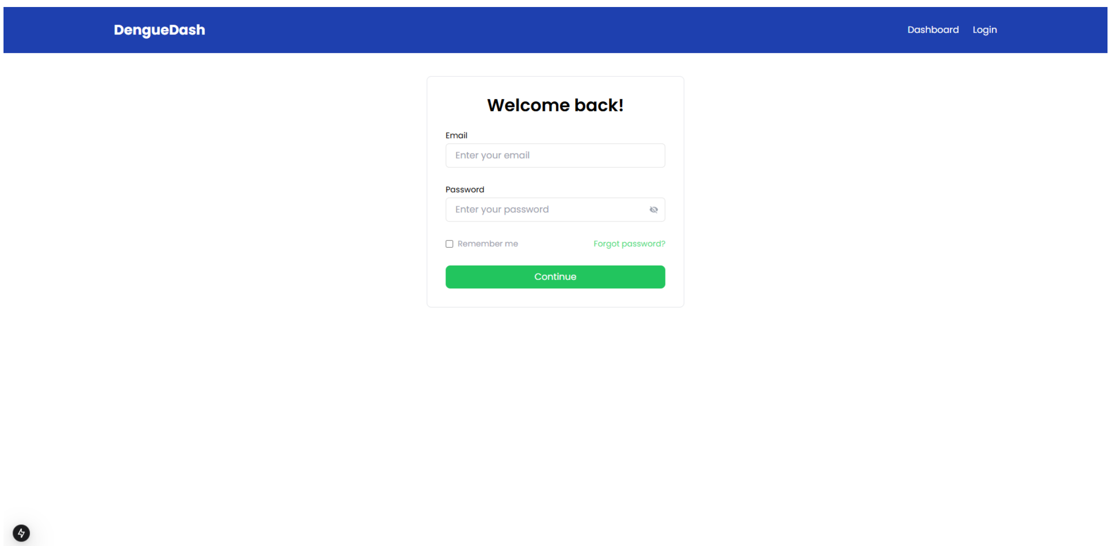

Figure 4.18: Login Page for Users

Encoder's View

Figures 4.19 and 4.20 show the digitized counterpart of the form obtained from the Iloilo Provincial Epidemiology and Surveillance Unit. As the system aims to support expandability for future features, some fields were modified to accommodate more detailed input. It is worth noting that all of the included fields adhere to the latest Philippine Integrated Disease and Surveillance Response (PIDSR) Dengue Forms, which the referenced form was based on. By doing this, it is assumed that the targeted users will have a familiarity when deployed on a national scale. On a further note, the case form includes the patient's basic information, dengue vaccination status, consultation details, laboratory results, and the outcome.

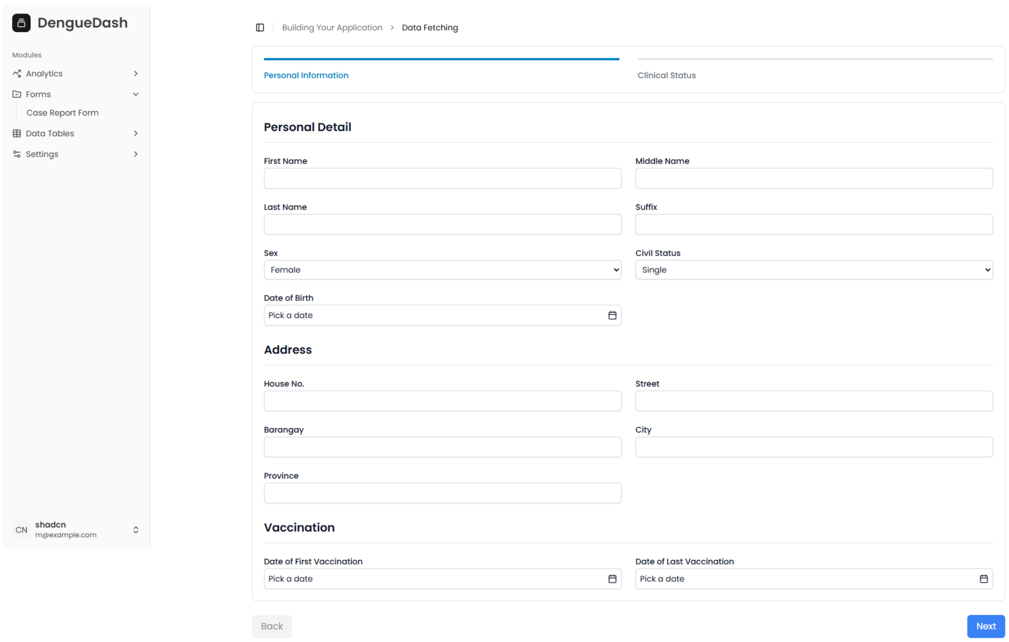

Figure 4.19: First Part of Case Report Form

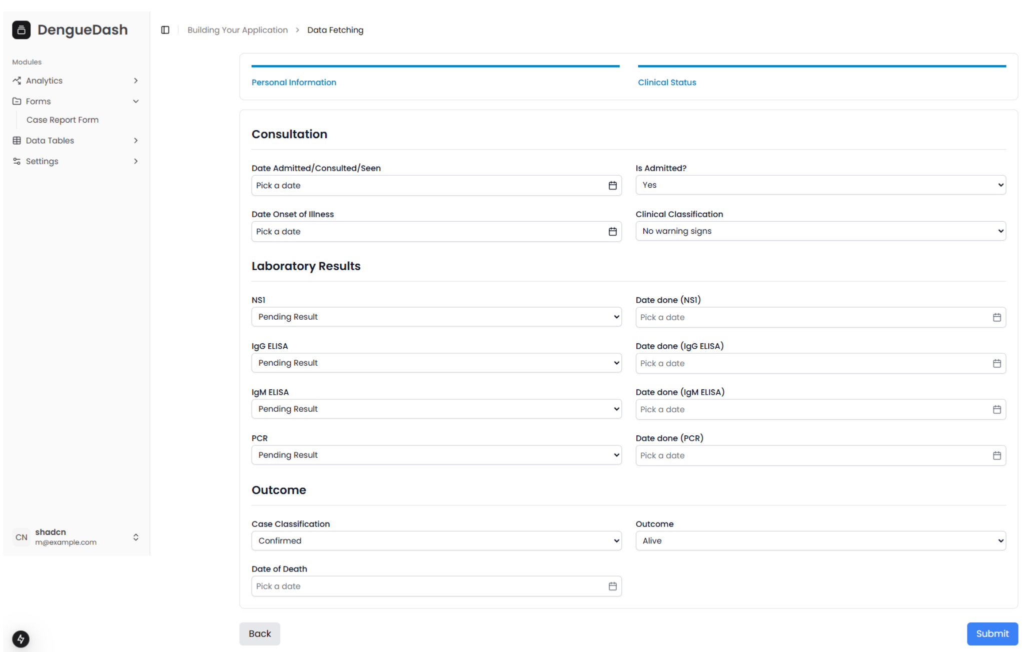

Figure 4.20: Second Part of Case Report Form

Once the data generated from the case report form is validated, it will be assigned as a new case and can be accessed through the Dengue Reports page, as shown in Figure 4.21. The said page displays basic information about the patient related to a specific case, including their name, address, date of consultation, and clinical and case classifications. It is also worth noting that it only shows cases the user is permitted to view. For example, in a local Disease Reporting Unit (DRU) setting, the user can only access records that came from the same DRU. On the other hand, in a consolidated surveillance unit such as a regional and provincial quarter, its users can view all the records that came from all the DRUs that report to them. Moving forward, Figure 4.22 shows the detailed case report of the patient on a particular consultation date.

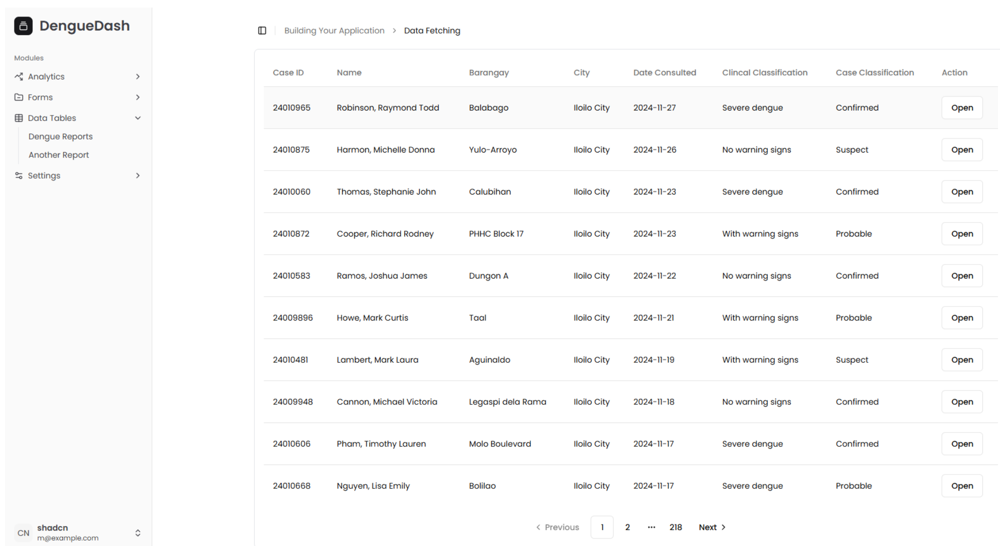

Figure 4.21: Dengue Reports

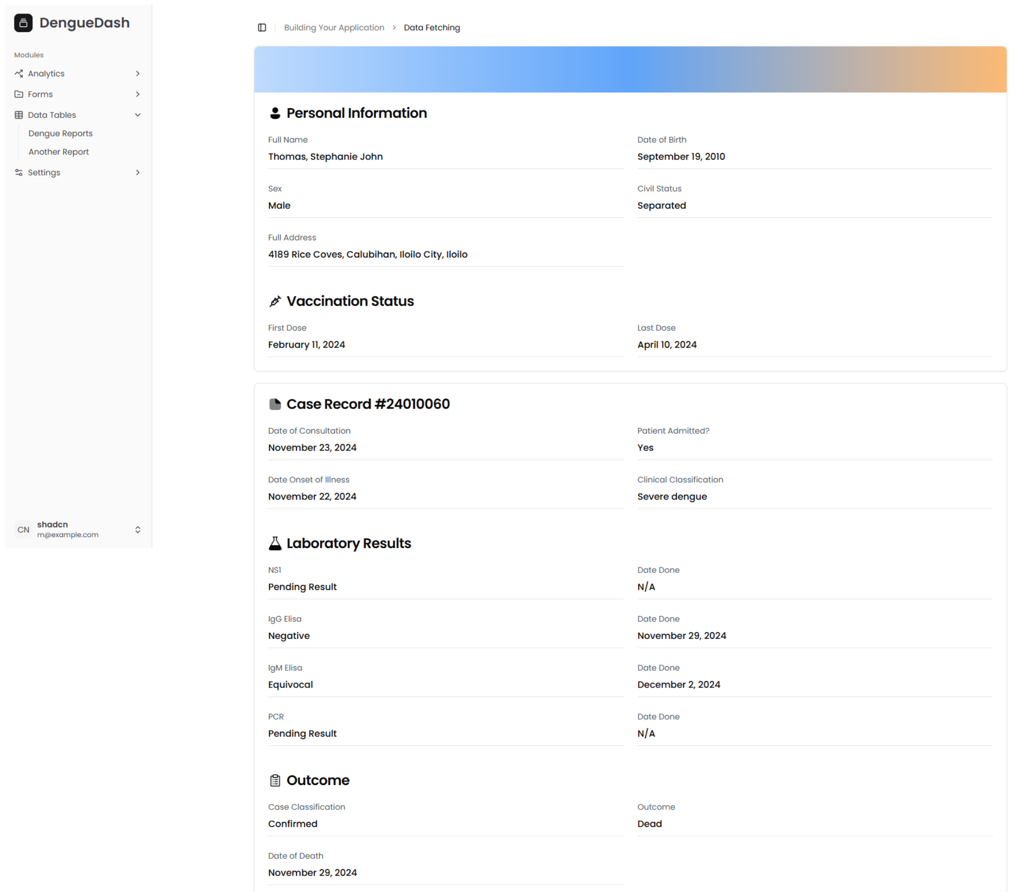

Figure 4.22: Detailed Case Report

References

About GitHub and Git - GitHub Docs. (n.d.). Retrieved from https://docs.github.com/en/get-started/start-your-journey/about-github-and-git

Ahmadini, A. A. H., Naeem, M., Aamir, M., Dewan, R., Alshqaq, S. S. A., & Mashwani, W. K. (2021). Analysis and forecast of the number of deaths, recovered cases, and confirmed cases from covid-19 for the top four affected countries using kalman filter. *Frontiers in Physics*, *9*, 629320.

Arroyo-Marioli, F., Bullano, F., Kucinskas, S., & Rondón-Moreno, C. (2021). Tracking r of covid-19: A new real-time estimation using the kalman filter. *PloS one*, *16*(1), e0244474.

Bosano, R. (2023). *Who: Ph most affected by dengue in western pacific.* Retrieved Use the date of access, from https://news.abs-cbn.com/spotlight/12/22/23/who-ph-most-affected-by-dengue-in-western-pacific

Brady, O. J., Smith, D. L., Scott, T. W., & Hay, S. I. (2015). Dengue disease outbreak definitions are implicitly variable. *Epidemics*, *11*, 92–102.

Bravo, L., Roque, V. G., Brett, J., Dizon, R., & L'Azou, M. (2014). Epidemiology of dengue disease in the philippines (2000–2011): a systematic literature review. *PLoS neglected tropical diseases*, *8*(11), e3027.

Carvajal, T. M., Viacrusis, K. M., Hernandez, L. F. T., Ho, H. T., Amalin, D. M., & Watanabe, K. (2018). Machine learning methods reveal the temporal pattern of dengue incidence using meteorological factors in metropolitan manila, philippines. *BMC infectious diseases*, *18*, 1–15.

Chart.js. (n.d.). Retrieved from https://www.chartjs.org/

Cheong, Y. L., Burkart, K., Leitão, P. J., & Lakes, T. (2013). Assessing weather effects on dengue disease in malaysia. *International journal of environmental research and public health*, *10*(12), 6319–6334.

Christie, T. (n.d.). *Home - Django REST framework.* Retrieved from https://www.django-rest-framework.org/

Colón-González, F. J., Fezzi, C., Lake, I. R., & Hunter, P. R. (2013). The effects of weather and climate change on dengue. *PLoS neglected tropical diseases*, *7*(11), e2503.

Hemisphere, N. (2015). Update on the dengue situation in the western pacific region. *Update*.

Hii, Y. L., Zhu, H., Ng, N., Ng, L. C., & Rocklöv, J. (2012). Forecast of dengue incidence using temperature and rainfall. *PLoS neglected tropical diseases*, *6*(11), e1908.

Joel, C. (2021, 10). *6 reasons to use Tailwind over traditional CSS.* Retrieved from https://dev.to/charliejoel/6-reasons-to-use-tailwind-over-traditional-css-1nc3

Leaflet — an open-source JavaScript library for interactive maps. (n.d.). Retrieved from https://leafletjs.com/

Lena, P. (2024). *Iloilo beefs up efforts amid hike in dengue cases.* Retrieved Use the date of access, from https://www.pna.gov.ph/articles/1231208

Li, X., Feng, S., Hou, N., Li, H., Zhang, S., Jian, Z., & Zi, Q. (2022). Applications of kalman filtering in time series prediction. In *International conference on intelligent robotics and applications* (pp. 520–531).

Ligue, K. D. B., & Ligue, K. J. B. (2022). Deep learning approach to forecasting dengue cases in davao city using long short-term memory (lstm). *Philippine Journal of Science*, *151*(3).

Perla. (2024). *Iloilo beefs up efforts amid hike in dengue cases.* Retrieved Use the date of access, from https://www.pna.gov.ph/articles/1231208

RabDashDC. (2024). *Rabdash dc.* Retrieved Use the date of access, from https://rabdash.com

Runge-Ranzinger, S., Kroeger, A., Olliaro, P., McCall, P. J., Sánchez Tejeda, G., Lloyd, L. S., … Coelho, G. (2016). Dengue contingency planning: from research to policy and practice. *PLoS neglected tropical diseases*, *10*(9), e0004916.

Shadcn. (n.d.). *Introduction.* Retrieved from https://ui.shadcn.com/docs

Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. (n.d.). Retrieved from https://tailwindcss.com/

Watts, D. M., Burke, D. S., Harrison, B. A., Whitmire, R. E., & Nisalak, A. (2020). Effect of temperature on the transmission of dengue virus by aedes aegypti. *The American Journal of Tropical Medicine and Hygiene*, *36*(1), 143–152.

What is Postman? Postman API Platform. (n.d.). Retrieved from https://www.postman.com/product/what-is-postman/

WHO. (2023). *Dengue - global situation.* Retrieved Use the date of access, from https://www.who.int/emergencies/disease-outbreak-news/item/2023-DON498

WHO. (2024). *Dengue and severe dengue.* Retrieved Use the date of access, from https://www.who.int/news-room/fact-sheets/detail/dengue-and-severe-dengue

Why Visual Studio Code? (2021, 11). Retrieved from https://code

.visualstudio.com/docs/editor/whyvscode

World Health Organization (WHO). (2018). Dengue and severe dengue in the philippines. *WHO Dengue Factsheet*. (Available at: https://www.who.int)

Zhou, S., & Malani, P. (2024). What is dengue? *Jama*, *332*(10), 850–850.

Zod. (n.d.). *TypeScript-first schema validation with static type inference.* Retrieved from https://zod.dev/?id=introduction

Appendix A

Appendix Title

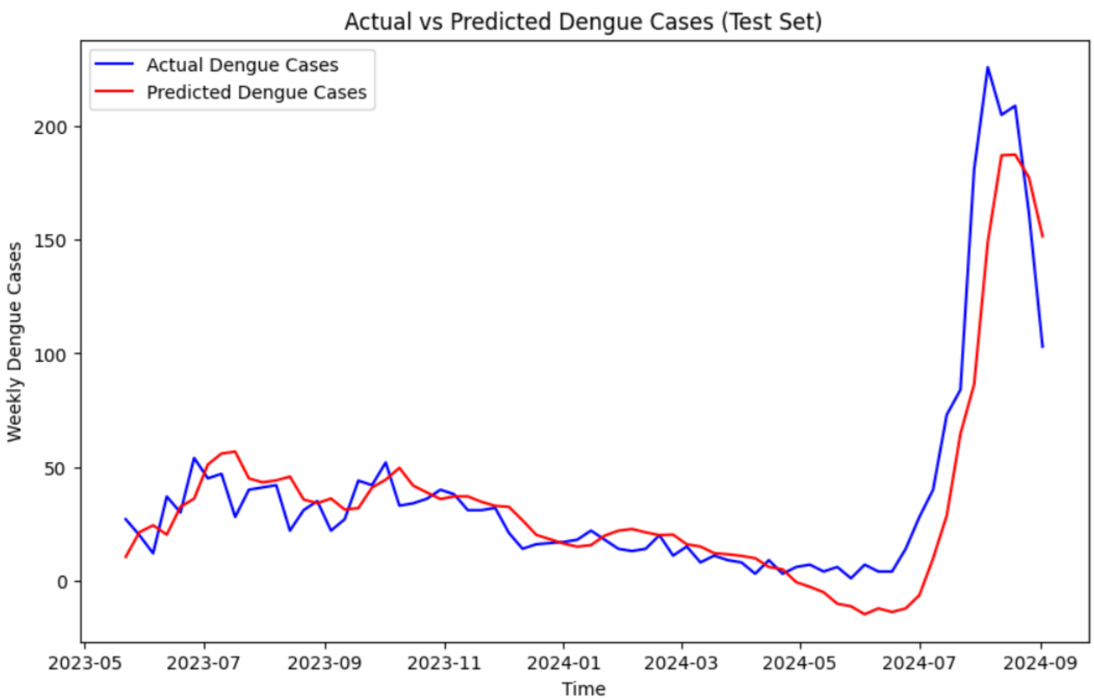

Figure A.1: LSTM Prediction Results for Test Set

Appendix B

Resource Persons

Mr. Firstname1 Lastname1
Role1
Affiliation1
`emailaddr1@domain.com`

Ms. Firstname2 Lastname2
Role2
Affiliation2
`emailaddr2@domain.net`

....