Project Report

Course Title: **Big Data and IoT**

Course Code: **CSE413**

Submitted By:

Jannatul Ferdous

Student ID: 212-15-14735

and

Mahfujur Rahman

Student ID: 212-15-14718

Both Section: 59-D

Department of Computer Science & Engineering

Daffodil International University

Submitted To:

Ms. Dristy Saha

Lecturer, Department of CSE

Faculty of Science and Information Technology

Daffodil International University

Submission Date: **11-06-2024**

Project Title

Weather Monitoring System

Project Description

This project involves creating a weather monitoring system that utilizes various IoT sensors to gather data on temperature, humidity, light intensity, and weather conditions. The data is displayed on an LCD screen and uploaded to ThingSpeak for remote monitoring and analysis. The system can detect, and alert users about rainy conditions using a buzzer.

Equipment

1. **DHT22 Sensor**: Used for measuring temperature and humidity.
2. **Rain Sensor Module**: Used to detect the presence of rain.
3. **LDR (Light Dependent Resistor)**: Used to measure light intensity.
4. **WiFi Module (ESP32)**: Used to connect the system to the internet and send data to ThingSpeak.
5. **Buzzer**: Used to alert users in case of rain.
6. **LiquidCrystal_I2C**: LCD screen used to display the readings.

Environment Setup

IoT Tools and Configuration:

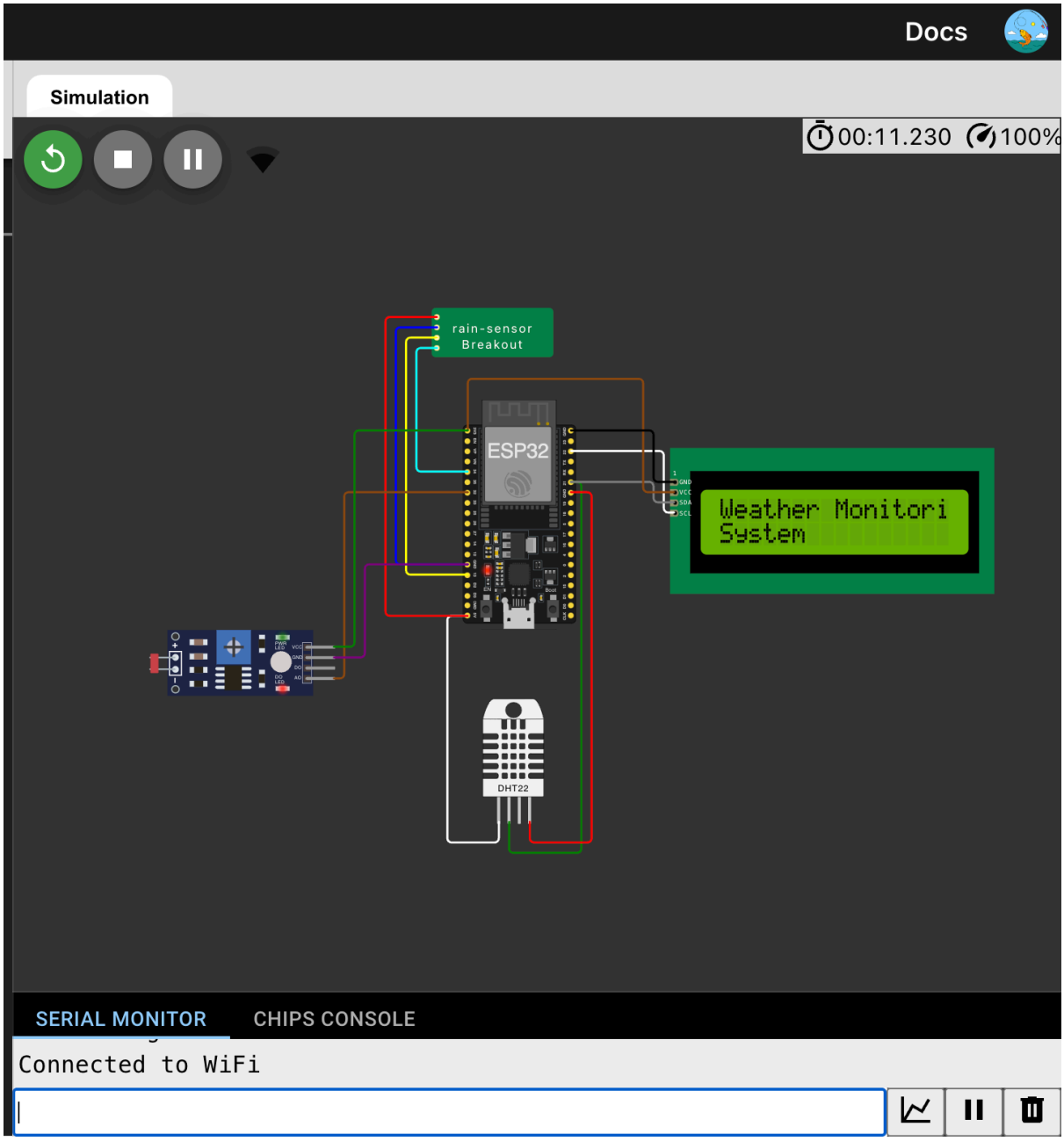
1. **ThingSpeak**: A cloud-based IoT analytics platform aggregating, visualizing, and analyzing live data streams. To set up ThingSpeak:
    - Create a MathWorks account and log into ThingSpeak.
    - Create a new channel note the Channel ID and Write the API Key.
    - Configure fields for temperature, humidity, weather conditions, and light intensity.

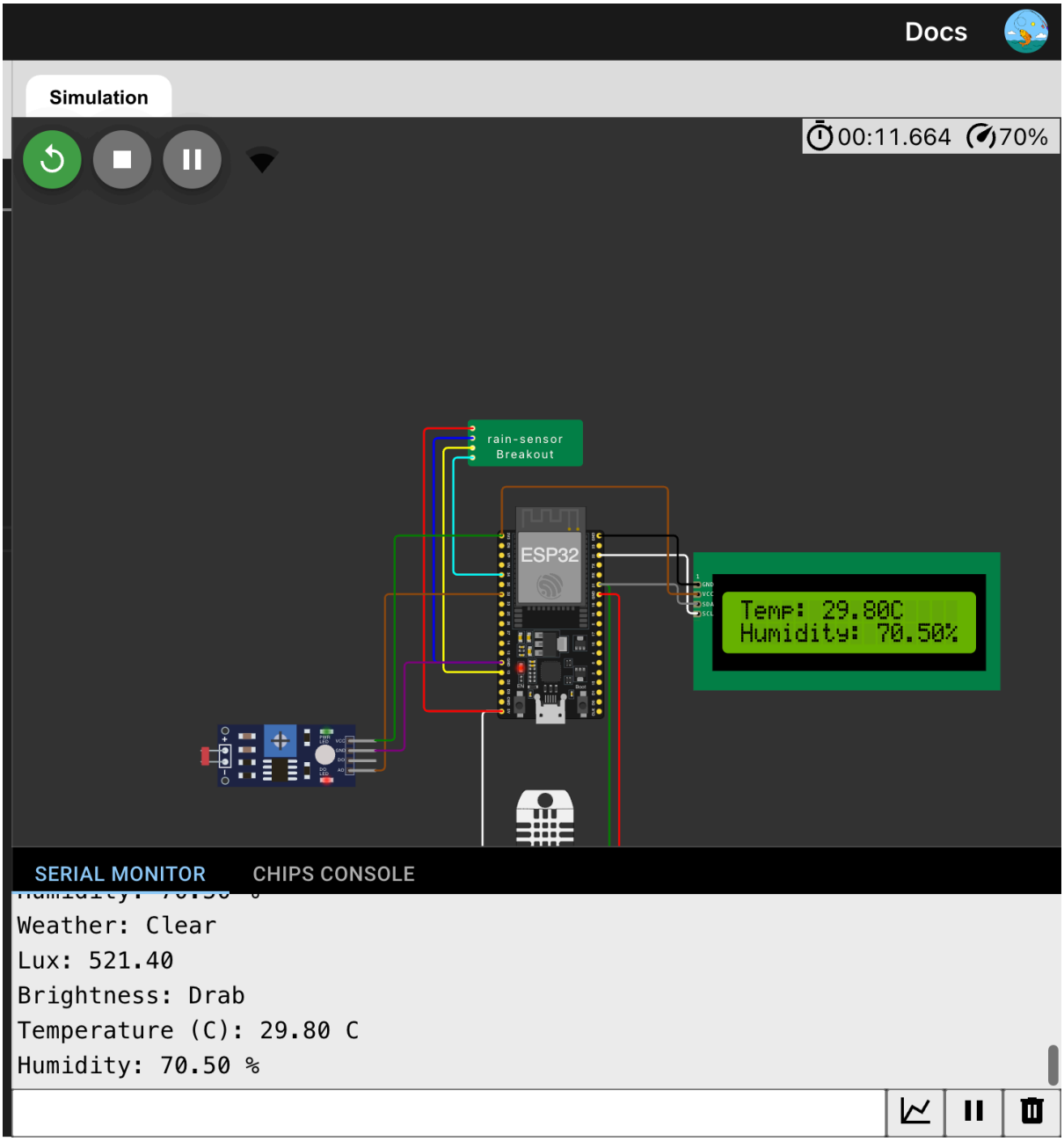
2. **Arduino IDE**: An open-source electronics platform based on easy-to-use hardware and software. Install the necessary libraries:
    - WiFi.h: For WiFi connectivity.
    - ThingSpeak.h: For communication with ThingSpeak.
    - DHT.h: For interfacing with the DHT22 sensor.
    - LiquidCrystal_I2C.h: For the LCD screen.
3. **Libraries Installation**: Install the required libraries via Arduino Library Manager:
    - DHT Sensor Library
    - ThingSpeak Library
    - LiquidCrystal I2C Library

Platform and Code Configuration

1. **WiFi Configuration**: Enter the SSID and password of the WiFi network in the code.
2. **ThingSpeak Configuration**: Enter the channel number and Write the API Key.
3. **Pin Configuration**: Define the pins for the sensors and output devices (e.g., buzzer, LCD).

Circuit Diagram





Data Collection Procedure

The data collection process involves the following steps:

1. **Initialization**: The system initializes the sensors and establishes a WiFi connection. It also initializes the LCD screen and clears any previous data.
2. **Reading Sensor Data**:
    - **Temperature and Humidity**: The DHT22 sensor reads the temperature and humidity.
    - **Rain Detection**: The rain sensor checks for the presence of rain.

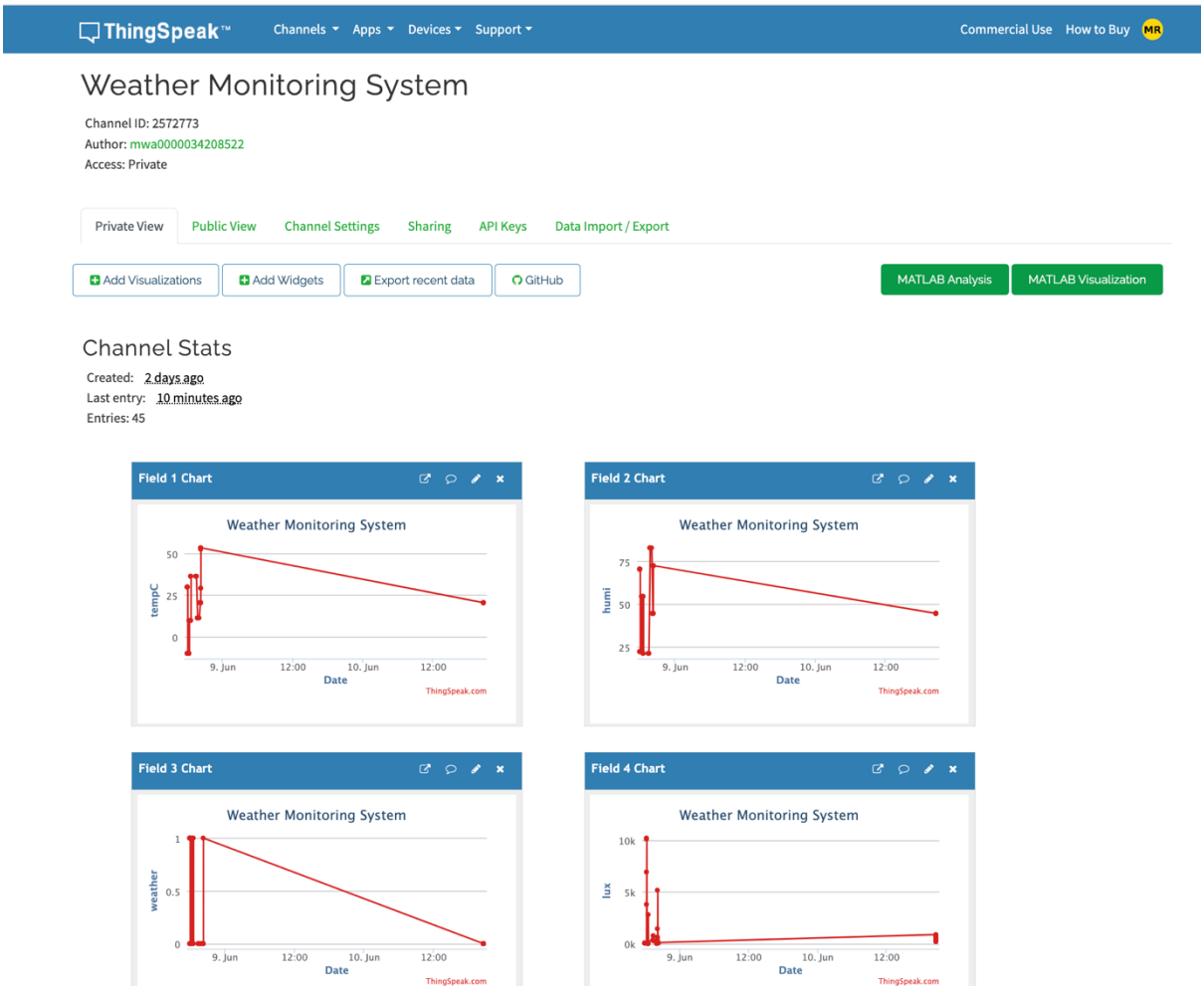
   - **Light Intensity**: The LDR sensor measures the ambient light intensity and calculates the lux value.
3. **Data Display**: The collected data is sequentially displayed on the LCD screen, showing temperature, humidity, weather conditions, and light intensity.
4. **Weather Condition and Alert**: The system determines the weather condition (clear or rainy) based on the rain sensor's state. If it is rainy, the buzzer is activated.
5. **Data Transmission**: The collected data is sent to ThingSpeak for remote monitoring. This includes:
   - Temperature
   - Humidity
   - Weather condition (encoded as 0 for clear and 1 for rainy)
   - Light intensity (lux value)
   - Light category based on the lux value
6. **Looping**: The system waits for a defined interval before repeating the process to continuously monitor the environment.

Sending Data to ThingSpeak

1. **Channel Setup**: We have created a channel on ThingSpeak. The myChannelNumber is the unique identifier for this channel.
2. **API Key**: The myWriteAPIKey is a unique key that authorizes your code to send data to your channel.

Visualizing Data on ThingSpeak

1. **Login to ThingSpeak**: Go to ThingSpeak and log in to our account.
2. **Navigate to Your Channel**: Go to Channels > My Channels and select your channel.
3. **Channel View**: You will see the data you sent plotted in real time. Each field will have its own graph.
4. **Customize Visualization**: You can customize the graphs, add widgets, and use MATLAB visualizations for more advanced analysis.



Exporting Data to CSV

1. **Export Data**: On our channel view, click the "Export" button (usually found under the "Data Import/Export" tab).
2. **Choose Format**: Select the time range and format (CSV).
3. **Download**: Click "Download" to get our data as a CSV file.

Machine Learning Implementation

In the final stages of the Weather Monitoring System project, we implemented a machine learning model to predict light intensity based on temperature and humidity data. This section provides a detailed explanation of the steps involved in the machine learning part of the project.

Data Loading and Preparation

The first step in the machine learning process was to load the collected data. The data, stored in a CSV file, was imported into a Pandas DataFrame for analysis and preprocessing. This was done using the pd.read_csv function.

After loading the data, we inspected it to understand its structure and contents. This involved checking for missing values and irrelevant columns that needed to be dropped. Specifically, the 'created_at' and 'entry_id' columns were removed as they were irrelevant to our analysis.

Data Cleaning and Preprocessing

Next, we addressed missing values in the dataset. Since missing values can affect the performance of machine learning models, it was crucial to handle them appropriately. We chose to drop rows with missing values.

With a clean dataset, we proceeded to define the features and the target variable. The features included temperature and humidity, while the target variable was the light intensity (lux).

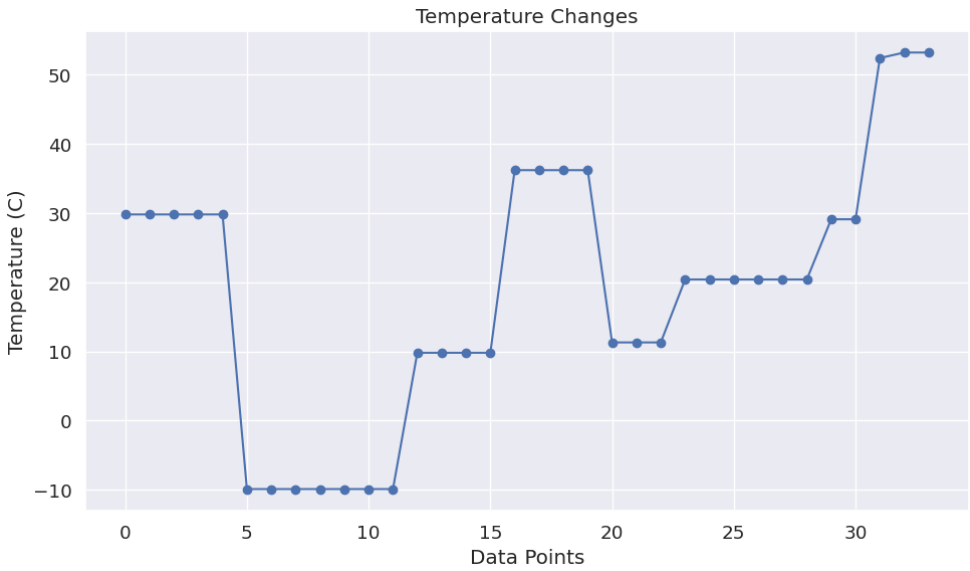
Encoding

We found that ightCategory column has categorical value. That’s why we have to encode it. We perform one-hot encoding on the categorical column 'ightCategory' within the DataFrame df. First, it computes the frequency of each category using value_counts(). Then, using pd.get_dummies(), it creates binary indicator variables (dummy variables) for each category, ensuring they are integers (dtype=int). These dummy variables are concatenated back into a new DataFrame merged alongside the original df using pd.concat(). Finally, the original categorical column 'ightCategory' is dropped from merged to form the final DataFrame. This process is crucial for transforming categorical data into a numerical format suitable for machine

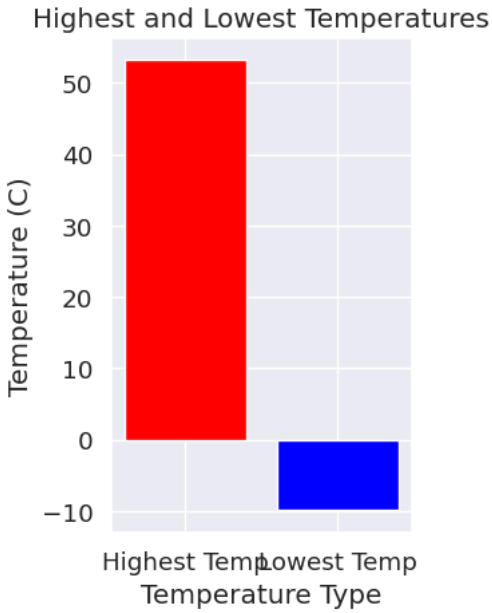
learning models, preserving categorical distinctions and enhancing model interpretability and performance.

Data Visualization:

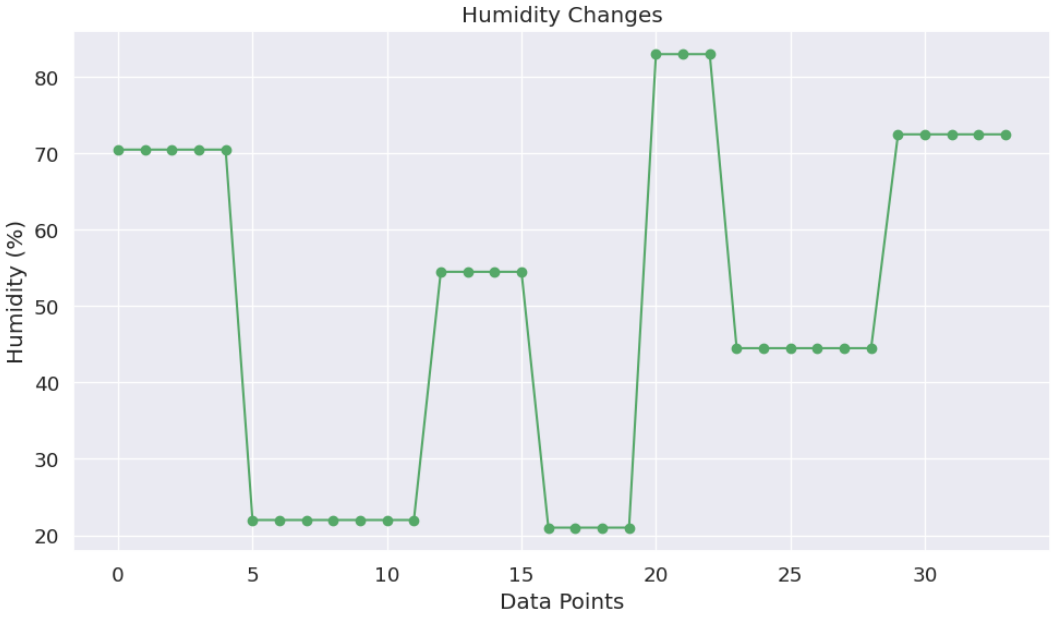
We visualized the overall temperature and humidity ups and down and also found the highest temperature and humidity and the lowest temperature and humidity by graphs.



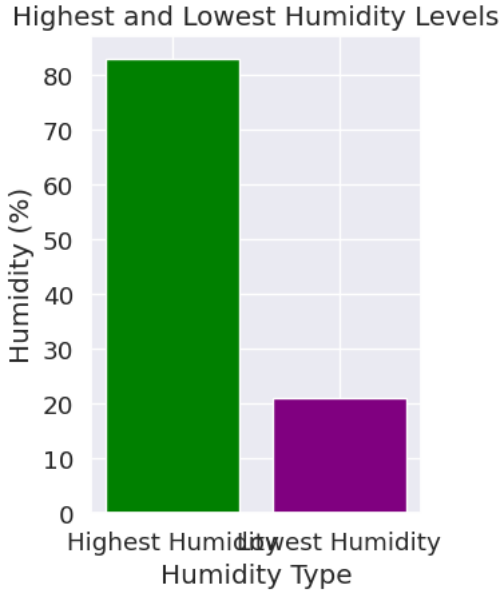
The graph depicts temperature changes across sequential data points, shown as a line plot with blue lines connecting each data point marked by circles. The x-axis represents sequential data points, while the y-axis shows temperature in Celsius. Titled "Temperature Changes," the plot is clear and well-labeled, aiding in visualizing the temperature trends over the dataset.



This plot uses a horizontal bar chart to visually compare the highest (red bar) and lowest (blue bar) temperatures. It labels each bar with "Highest Temp" and "Lowest Temp" on the x-axis, showing temperatures in Celsius on the y-axis. The title "Highest and Lowest Temperatures" summarizes the focus of the graph, highlighting extreme temperature values effectively.



This plot shows changes in humidity over sequential data points, using green markers and lines. The x-axis represents data points, while the y-axis measures humidity in percentage (%). The title "Humidity Changes" summarizes the plot's focus, which visualizes how humidity fluctuates across the dataset.



This subplot uses horizontal bars to display the highest (green) and lowest (purple) humidity levels. Each bar is labeled as "Highest Humidity" and "Lowest Humidity" on the x-axis, measuring humidity in percentage (%) on the y-axis. Titled "Highest and Lowest Humidity Levels," the graph provides a straightforward comparison of extreme humidity values.

Data Splitting

To evaluate the performance of our machine learning model, we split the data into training and testing sets. This was done using the train_test_split function from Scikit-learn, with 80% of the data used for training and 20% for testing.

Model Training

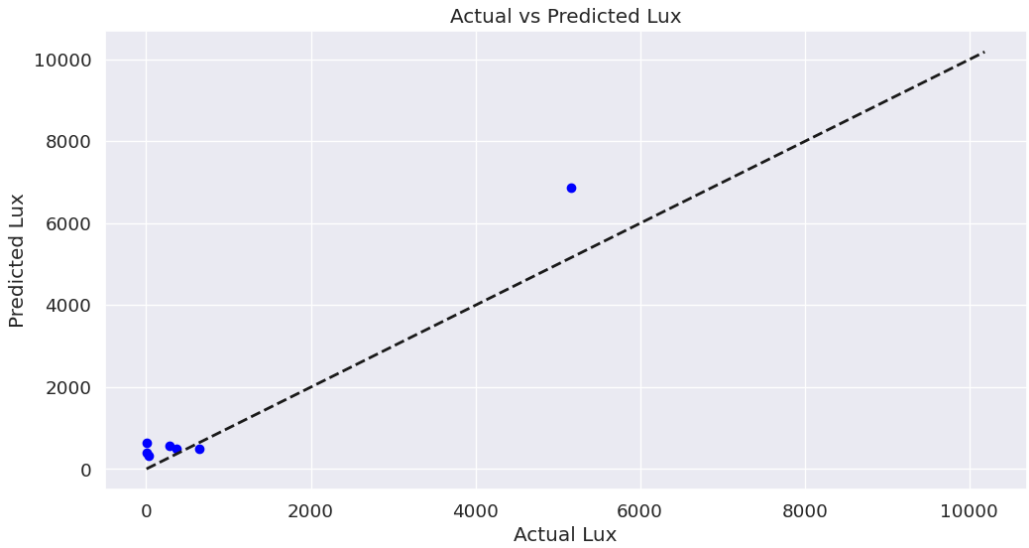
We chose Linear Regression as our machine learning model due to its simplicity and effectiveness for regression tasks. The model was instantiated and trained using the training data.

Model Evaluation

After training the model, we evaluated its performance on the test data. The predictions were generated using the prediction method, and the performance was measured using Mean Squared Error (MSE) and R-squared (R2) metrics. The results showed how well the model performed on both the training and testing sets, giving us an indication of its accuracy and generalization capability.

Visualization of Results

To gain further insights into the model's performance, we visualized the relationship between the actual and predicted lux values. Scatter plots were used to compare the predictions with the actual values for both the training and testing sets.



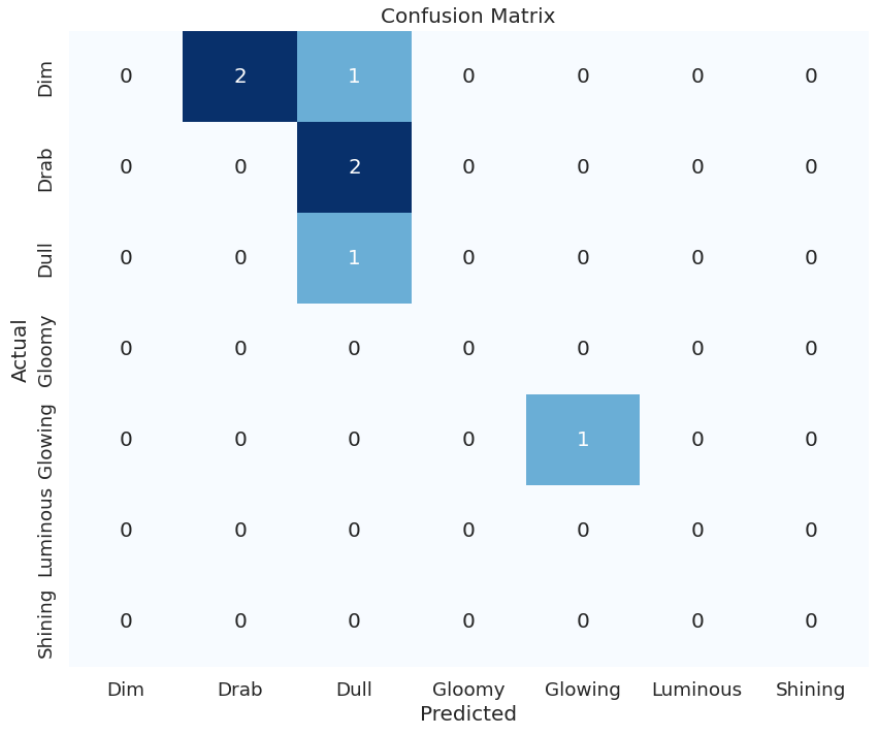
These plots helped us visualize how closely the predicted values matched the actual values, indicating the accuracy of our model.

Categorization of Light Intensity

To provide more meaningful insights, we categorized the predicted lux values into different light intensity categories. A custom function was defined to map lux values to categories such as "Dim," "Drab," "Dull," "Gloomy," "Glowing," and "Luminous." Thus, we transformed the continuous lux values into categorical values, making it easier to interpret the light intensity levels.

Confusion Matrix

Finally, we evaluated the accuracy of the categorized predictions using a confusion matrix. The confusion matrix compared the actual and predicted categories, providing a detailed view of the model's performance in categorizing light intensity.



The heatmap of the confusion matrix visually represented the model's accuracy in predicting each category, highlighting the areas where the model performed well and where it needed improvement.

Conclusion

The Weather Monitoring System successfully collects and analyzes environmental data using various IoT sensors. The data is transmitted to the ThingSpeak platform for real-time monitoring and visualization. The system provides valuable insights into temperature, humidity, weather conditions, and light intensity. The implementation demonstrates the effective use of IoT technologies for environmental monitoring and data analysis. The machine learning implementation in the Weather Monitoring System provided valuable insights into the relationship between temperature, humidity, and light intensity. The Linear Regression model demonstrated a reasonable performance, accurately predicting the lux values and categorizing the light intensity levels. This enhanced the overall functionality of the Weather Monitoring System, enabling it to provide more detailed and meaningful environmental data.

Future Work

- **Expand Sensor Array**: Include additional sensors for measuring other environmental parameters such as air quality, pressure, and wind speed.
- **Enhanced Data Analysis**: Implement advanced data analysis techniques such as time series analysis and machine learning algorithms for better predictions and insights.
- **Mobile Application**: Develop a mobile application to provide users with real-time notifications and access to historical data.
- **Energy Efficiency**: Optimize the system for lower power consumption to enable long-term deployment in remote areas.

References

- **ThingSpeak**: ThingSpeak Documentation
- **DHT22 Sensor**: DHT22 Datasheet
- **ESP32**: ESP32 Documentation
- **Pandas**: Pandas Documentation
- **Matplotlib**: Matplotlib Documentation
- **Seaborn**: Seaborn Documentation

-Thank you-