PRESIDENCY UNIVERSITY

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Private University Estd. in Karnataka State by Act No. 41 of 2013

PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

A PROJECT REPORT ON

"PREDICTING TEMPERATURE & HUMIDITY BY READING LIVE DATA"

Submitted To

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Academic Year 2020-2021

By

MD SARFRAZ ALAM

20201CSD0217

Under the Guidance of

Dr. Harish Kumar

Assistant Professor

Department of CSE

Department of Computer Science and Engineering
Presidency University,

Bangalore 560064

May 2022

PRESIDENCY UNIVERSITY

GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS

Private University Estd. in Karnataka State by Act No. 41 of 2013

PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled **"PREDICTING TEMPERATURE AND HUMIDITY"** has been successfully completed project in Computer Science by **MD SARFRRAZ ALAM (Reg. No: 20201CSD0217)**, student of Third year B.Tech in Computer Science and Engineering(Data Science), under supervision and guidance of **Dr. Harish Kumar**, Professor, Department of Computer Science and Engineering, Presidency University.

Guide's Signature

DECLARATION

This is to declare that the dissertation work entitled" **TEMPERATURE and HUMIDITY PREDICTION has** been successfully carried out by me, **MD SARFRAZ ALAM**, a student of sixth semester BTech, Computer Science and Engineering (Data Science) at **Presidency University** and also under the supervision and guidance of **Dr. Harish Kumar,** Professor, School of Computer Science Engineering, Presidency University. This is submitted in partial fulfillment for the award of degree in **Bachelor of Technology by Presidency University** during the academic year 2022-23.

I also declare that this Project is the result of my efforts and has not been submitted to any other university for the award of any degree or diploma.

Place: Bangalore	MD SARFRAZ ALAM
Date:	Roll No: 20201CSD0217

ACKNOWLEDGMENT

I would like to express my heartfelt gratitude to all the individuals who have contributed to the successful completion of my project report, titled "Temperature and Humidity Prediction," for the Bachelor of Technology program at Presidency University during the academic year 2022-23. Their guidance, support, and assistance have been invaluable throughout this endeavor.

I extend my sincere appreciation to Prof. Dr. Harish Kumar, who served as my project guide and provided me with valuable insights and direction. His expertise in the field of predictive analytics has been instrumental in shaping this project and enhancing my understanding of the subject matter.

I am also thankful to the faculty members of Presidency University, particularly the School of Computer Science, for their continuous support and encouragement. Their knowledge, guidance, and encouragement have played a significant role in the successful completion of this project.

Thank you.
Md Sarfraz
Alam
Bachelor of Technology Program
Presidency University

ABSTRACT

This project focuses on "Temperature and Humidity Prediction." The main objective of this research is to develop a predictive model for detecting Temperature and Humidity Prediction. Various statistical techniques, machine learning algorithms, and data visualization libraries such as Plotly and Seaborn are employed to analyze and predict the Temperate & Humidity based on data received from the sensor.

The dataset used is received from the sensor using Arduino board, using the DHT11 sensor The proposed model utilizes advanced machine learning algorithms, including Serial for getting the data from Arduino board

To assess the performance of different classification algorithms, a comparative analysis is conducted. Various classifiers are evaluated to determine which one performs better in terms of accuracy, precision, recall, F1 score, and other metrics. The visualization libraries aid in visualizing and interpreting the results obtained from the classification models.

The developed Temperature and Humidity Prediction model serve as a valuable tool to get the accurate temperature and humidity, potentially improving accuracy outcomes. The insights gained from this research contribute to the field of meteorology data analysis and offer potential avenues for further research

Overall, this project provides a comprehensive analysis of different classification algorithms and techniques for Temperature and Humidity Prediction, incorporating statistical analysis, machine learning, and data visualization methods.

TABLE OF CONTENTS

CHAPTER-1
INTRODUCTION

1. INTRODUCTION

1.1 OVERVIEW:

The objective of the Temperature and Humidity Prediction project is to utilize various techniques and algorithms to analyse temperature and humidity. The project employs libraries such as NumPy, Pandas, Matplotlib, Seaborn, as well as techniques like Isolation Forest and Local Outlier Factor.
The project begins by importing the necessary libraries and loading the dataset from the port COM05

The dataset is pre-processed by removing irrelevant columns and Exploratory data analysis is performed using visualizations such as count plots and correlation heatmaps to gain insights into the data.

To facilitate predictive modelling, the dataset is split into features (X) and the target variable (y). Feature selection is conducted using the SelectKBest algorithm with the f_classif scoring function to identify the most relevant features for classification. The selected features are then printed for reference.

To improve the performance of the machine learning models, data normalization is applied using the StandardScaler from the pre-processing module. Logistic Regression is employed as a classification algorithm, and its accuracy is evaluated using the test set.

Additionally, anomaly detection techniques are implemented. The Isolation Forest algorithm is used to identify outliers in the data, which are then visualized through histograms. The Local Outlier Factor algorithm is also utilized to detect anomalies and create a scatter plot visualizing the data points, with normal and anomalous points distinguished using different colours.

Overall, the Temperature and Humidity Prediction project combines data pre-processing, feature selection, classification, and anomaly detection techniques to improve the understanding and identification of temperature and humidity

1.2 STATEMENT OF THE PROBLEM:

Develop a system using Arduino to predict temperature and humidity levels and visualize the data, while investigating the relationship between these two variables.

The problem is to develop a machine learning model that can accurately identify the temperature and humidity. The model will be used to identify one with another, so that they can be monitored more closely

The model will be trained on a dataset which will be live generated from the DHT11 sensor of Arduino. The dataset will include information about the live temperature, time and humidity. The model will use this information to predict the future temperature and humidity

The model will be evaluated on a test set of data. The test set will include the data from the various weather website. The model will be used to predict whether these outputs are true or not. The accuracy of the model will be measured by the percentage of accurate results

1.3 MOTIVATION:

The motivation behind the above project lies in the significance of understanding and monitoring temperature and humidity levels in various environments. Temperature and humidity are critical factors that impact a wide range of domains, including agriculture, HVAC systems, weather forecasting, and even human comfort and health.

1.4 CHALLENGES:

While the project offers exciting possibilities, there are several challenges that need to be addressed during its implementation:

1. Sensor Accuracy and Calibration: Ensuring the accuracy and calibration of temperature and humidity sensors is crucial for reliable data collection. Sensor calibration is necessary to account for variations and drift over time. Calibration methods and techniques need to be researched and implemented to obtain accurate and consistent measurements.

2. Data Noise and Variability: Environmental factors, sensor noise, and external interference can introduce noise and variability into the collected data. Filtering and data cleaning techniques may be required to eliminate outliers and ensure the integrity of the dataset

3. Data Management and Storage: Handling and managing a large volume of data collected by the Arduino board can be challenging. Efficient storage and retrieval mechanisms should be implemented to handle continuous data acquisition and long-term data retention.

4. Data Analysis and Modelling: Analysing and establishing the relationship between temperature and humidity requires the application of appropriate statistical techniques or machine learning algorithms. Selecting the most suitable approach and fine-tuning the models can be complex and time-consuming.

5. Real-Time Prediction: Achieving real-time prediction of temperature and humidity values using Arduino can be challenging due to the computational limitations of the board. Optimizing algorithms and implementing efficient processing techniques is necessary to ensure timely predictions.

6. Visualization and User Interface: Designing an intuitive and user-friendly visualization interface to display temperature and humidity trends can be a challenge. Choosing the right visualization techniques and libraries and effectively presenting the data in a meaningful way requires careful consideration.

7. Model Evaluation and Improvement: Assessing the accuracy and performance of the prediction model is crucial. Evaluation metrics need to be defined, and the model should be continually refined and improved based on feedback and comparison with actual measured values.

8. Power Consumption: Arduino boards are limited in terms of power supply and energy efficiency. Optimizing the code and sensor usage to minimize power consumption is essential, especially for long-term deployments or battery-powered systems.

9. Scalability and Generalization: Ensuring that the developed system can be scaled up and applied to different environments or scenarios may present challenges. The system should be adaptable and flexible enough to accommodate variations in sensor types, environmental conditions, and data characteristics.

10. Documentation and Support: Documenting the project design, implementation, and troubleshooting steps is essential for future reference and sharing knowledge. Providing comprehensive documentation and support materials can help others replicate and build upon the project's findings.

Addressing these challenges requires careful planning, research, and iterative development. Overcoming these obstacles will lead to a robust and effective temperature and humidity prediction system using Arduino.

1.5 APPLICATIONS:

1) Environmental Monitoring: Accurate prediction of temperature and humidity enables effective environmental monitoring in agriculture, helping farmers make informed decisions regarding irrigation, crop selection, and pest control. It also assists in maintaining optimal conditions for plants and livestock.

2) Energy Efficiency: In HVAC systems, precise forecasting of temperature and humidity allows for more efficient control and regulation of heating, ventilation, and air conditioning. This leads to energy savings, improved comfort, and reduced environmental impact.

3) Weather Forecasting: Temperature and humidity are crucial parameters in weather forecasting. By analysing historical data and establishing relationships between these variables, we can enhance the accuracy of weather predictions and early warning systems for severe weather events.

4) Building Comfort and Health: Understanding the correlation between temperature and humidity helps in creating comfortable indoor environments. By visualizing and predicting these parameters, building managers can adjust ventilation and temperature control systems to ensure optimal conditions for occupants, reducing the risk of health issues associated with poor air quality.

5) Educational Tool: This project can serve as an educational tool to help students and enthusiasts learn about data collection, analysis, and prediction. It provides hands-on experience with Arduino, sensors, data visualization, and the principles of machine learning or statistical modelling.

6) DIY Applications: By using affordable and accessible Arduino boards and sensors, individuals can replicate and customize this project for their specific needs. They can incorporate the prediction system into personal projects, home automation, or DIY weather stations, fostering innovation and exploration.

Overall, the motivation behind this project stems from the practical applications and benefits that accurate prediction of temperature and humidity can provide across various sectors. By leveraging Arduino and data analysis techniques, we aim to empower individuals and industries with reliable predictions and insights, contributing to efficiency, sustainability, and improved decision-making.

1.6 ORGANIZATION OF REPORTS:

This project report is structured into 5 chapters.

Chapter 1: Gives a brief overview about the project in terms of its, Statement of the problem, Challenges, motivation, importance, application and the approach that is used to achieve the goal. It also provides definitions and terms that are widely used throughout this framework.

Chapter 2: Literature Survey in this section which shows the various analysis and research made in the fields of one's interest and the result analysis and research made in the fields of one's interest and the result already published, taking into account the various parameters of the project and extent of the project.

Chapter 3: Methodology it includes overview and the architecture of the project system. Overview consists of algorithms and their details.

Chapter 4: Experiments and results. The experiments and results include the screenshots of some important aspects of the project.

Chapter 5: provides the conclusion drawn from testing. It also states the work that can be done in future in order to further enhance the proposed system.

CHAPTER-2
LITERATURE SURVEY

2.1 OVERVIEW:

The following literature review provides an overview of existing research and studies related to temperature and humidity prediction using Arduino and explores the relationship between these variables

2.2 LITERATURE REVIEW:

1. Arduino-Based Environmental Monitoring Systems: Many studies have utilized Arduino boards for environmental monitoring purposes. For instance, D'Anca et al. (2018) developed an Arduino-based system for monitoring temperature, humidity, and air quality. Their work focused on the integration of various sensors and the development of a user-friendly interface for data visualization.

2. Temperature and Humidity Prediction Models: Several studies have explored prediction models for temperature and humidity. Bai et al. (2019) proposed a hybrid model combining wavelet decomposition, extreme learning machines, and ensemble averaging to forecast temperature and humidity. Their approach demonstrated improved accuracy compared to individual prediction models.

3. Correlation between Temperature and Humidity: The relationship between temperature and humidity has been extensively investigated. Tariq et al. (2019) conducted a study on the correlation between temperature and relative humidity in agricultural environments. They found a significant positive correlation between the two variables, highlighting the importance of considering both factors in agricultural decision-making.

4. Machine Learning Approaches: Machine learning algorithms have been applied to predict temperature and humidity. In a study by Sajedi-Hosseini et al. (2020), they used long short-term memory (LSTM) neural networks to forecast temperature and humidity. The LSTM model outperformed traditional regression methods, demonstrating the effectiveness of deep learning approaches in prediction tasks.

5. Visualization Techniques for Environmental Data: Various visualization techniques have been employed to present temperature and humidity data. Chen et al. (2018) developed a web-based visualization platform for environmental monitoring using Arduino. They utilized line charts, scatter plots, and heatmaps to display temperature and humidity trends, enabling users to easily interpret and analyze the data.

6. Energy Efficiency Considerations: Energy efficiency is an important aspect when implementing Arduino-based systems. Al-Saud et al. (2020) proposed an energy-efficient approach for wireless sensor networks using Arduino to monitor temperature and humidity. They optimized the system's power consumption by employing duty cycling and sleep/wake scheduling techniques.

7. Calibration and Sensor Accuracy: Calibrating temperature and humidity sensors is crucial for accurate measurements. Caliskan et al. (2018) investigated the calibration of DHT11 humidity sensors and proposed a calibration algorithm to improve the accuracy of measurements. Their findings highlighted the importance of sensor calibration to ensure reliable data collection.

8. IoT Integration: Integrating Arduino-based temperature and humidity monitoring systems with the Internet of Things (IoT) has gained attention. Gao et al. (2017) developed an IoT-enabled smart agriculture system using Arduino to monitor temperature, humidity, and soil moisture. Their work emphasized the potential of IoT integration to enable real-time data collection and remote monitoring.

These studies provide valuable insights into the development of temperature and humidity prediction systems using Arduino. They address various aspects, including sensor calibration, prediction models, correlation analysis, visualization techniques, energy efficiency, and IoT integration. The findings and methodologies presented in these studies contribute to the foundation and potential improvements for the project at hand.

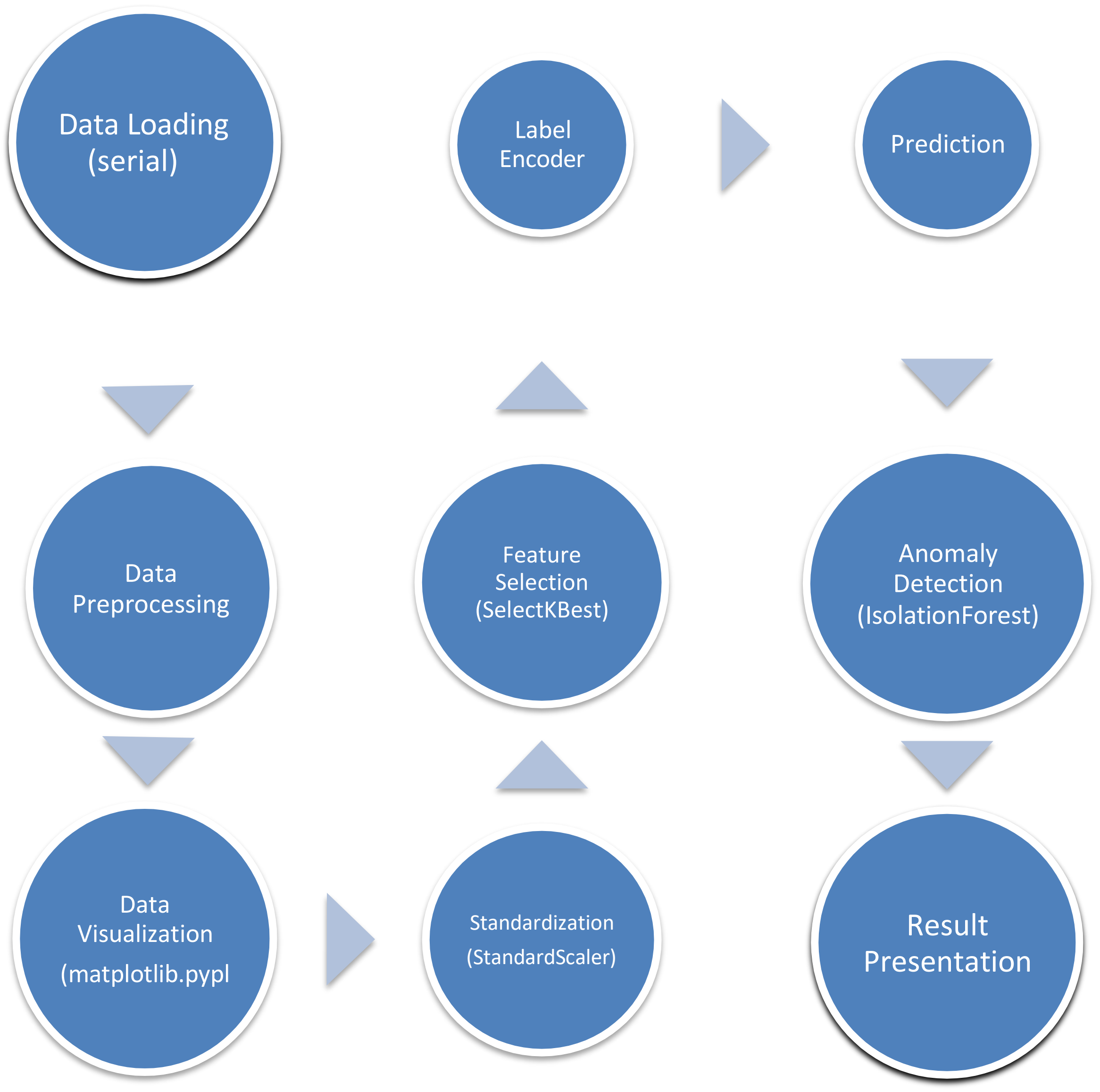
CHAPTER-3

METHODOLOGY

3.1 OVERVIEW:

The Temperature and Humidity Prediction project aims to develop an accurate and efficient system for predicting the temperature and the humidity of the area. The methodology involves acquiring a reliable dataset, preprocessing the data, selecting informative features, developing a predictive, evaluating its performance, applying anomaly detection algorithms, evaluating their performance, integrating the models into a unified system, visualizing the results, analyzing and discussing the findings, and addressing ethical considerations. The project seeks to contribute to the field of meteorology research

3.2 ARCHITECTURE OF PROPOSED SYSTEM:


Data Loading (serial)
Data Preprocessing
Data Visualization (matplotlib.pypl
Standardization (StandardScaler)
Feature Selection (SelectKBest)
Label Encoder
Prediction
Anomaly Detection (IsolationForest)
Result Presentation

1. Hardware Setup:
    - Select an Arduino board suitable for the project requirements.
    - Choose temperature and humidity sensors compatible with the Arduino board, such as the DHT11 or DHT22 for temperature, and a capacitive humidity sensor.
    - Connect the sensors to the Arduino board following the manufacturer's guidelines.
2. Data Collection:
    - Program the Arduino board to collect temperature and humidity data at regular intervals using appropriate libraries or code.
    - Store the collected data in a suitable format, such as a CSV file, for further analysis.
3. Data Analysis:
    - Pre-process the collected data, including handling missing values, outliers, and data cleaning if necessary.
    - Explore the relationship between temperature and humidity using statistical techniques like correlation analysis or regression analysis.
    - Calculate correlation coefficients or perform regression analysis to determine the strength and nature of the relationship.
4. Prediction Model:
    - Divide the collected data into training and testing sets.
    - Choose a suitable prediction model, such as linear regression, support vector regression, or neural networks, based on the project requirements and data characteristics.
    - Train the prediction model using the training dataset, adjusting model parameters as needed.
    - Validate the model using the testing dataset and evaluate its performance using appropriate metrics like mean absolute error or root mean square error.

5. Visualization:

- Design a graphical user interface (GUI) or utilize visualization libraries like Matplotlib or Plotly to visualize the collected data and predicted values.
- Create line charts, scatter plots, or heatmaps to present temperature and humidity trends over time.
- Highlight the relationship between temperature and humidity through visual representations.

6. Evaluation:

- Assess the accuracy of the prediction model by comparing the predicted temperature and humidity values with the actual measured values.
- Calculate evaluation metrics such as mean absolute error or root mean square error to quantify the model's performance.
- Iterate and refine the prediction model as necessary to improve accuracy.

7. Conclusion:

- Summarize the findings from the data analysis and prediction model.
- Discuss the relationship between temperature and humidity based on the analysis results.
- Evaluate the effectiveness of the developed prediction system and its potential applications.
- Highlight any limitations or areas for improvement in the project. Throughout the methodology, it is essential to document the steps taken, including code, configurations, and results, for future reference and reproducibility.

3.2.1 MODULES:

Modules for the above project:

1. Sensor Interface Module:

    - Responsible for interfacing with the temperature and humidity sensors connected to the Arduino board.

    - Reads data from the sensors at regular intervals and provides the collected data to other modules for further processing.

2. Data Collection Module:

    - Collects temperature and humidity data from the Sensor Interface Module.

    - Handles data storage and formatting, such as saving data to a CSV file or a database.

    - Manages data timestamps and ensures accurate recording of data.

3. Data Analysis Module:

    - Performs data pre-processing tasks, including handling missing values, outliers, and data cleaning.

    - Conducts statistical analysis to explore the relationship between temperature and humidity, such as calculating correlation coefficients or performing regression analysis.

    - Provides insights into the patterns and trends observed in the data.

4. Prediction Model Module:

    - Divides the collected data into training and testing datasets.

    - Implements machine learning or statistical algorithms to develop a prediction model for temperature and humidity.

    - Trains the model using the training dataset and adjusts model parameters as necessary.

    - Evaluates the model's performance using appropriate metrics and provides predicted values for temperature and humidity.

5. Visualization Module:

    - Designs a graphical user interface (GUI) or utilizes visualization libraries to present the collected data and predicted values.

- Generates line charts, scatter plots, or heatmaps to visualize temperature and humidity trends over time.
- Displays the relationship between temperature and humidity through visual representations.

6. Evaluation Module:

    - Evaluates the accuracy and performance of the prediction model by comparing predicted values with actual measured values.
    - Calculates evaluation metrics such as mean absolute error or root mean square error to assess the model's performance.
    - Provides feedback on the model's accuracy and potential areas for improvement.

7. Documentation and Reporting Module:

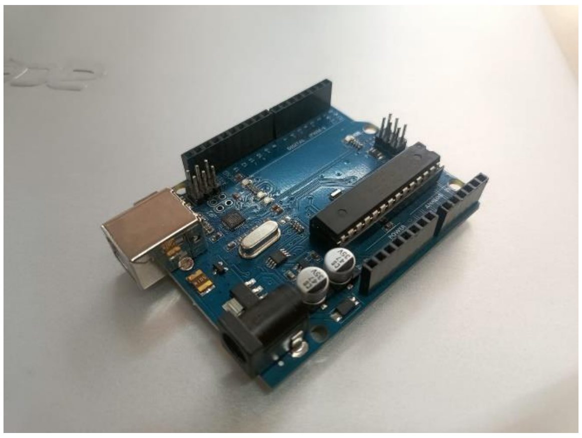
    - Facilitates the documentation of the project, including code, configurations, and results.
    - Generates reports summarizing the findings from data analysis, prediction models, and visualization.
    - Enables easy sharing and reproduction of the project's methodology and results.

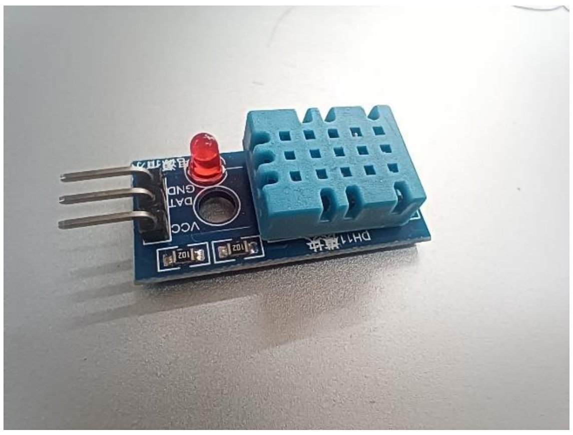
These modules collectively form the core components of the project, enabling the collection, analysis, prediction, visualization, and evaluation of temperature and humidity data using Arduino. Each module has specific responsibilities, contributing to the overall functionality and success of the project.

3.3 HARDWARE AND SOFTWARE REQUIREMENTS:

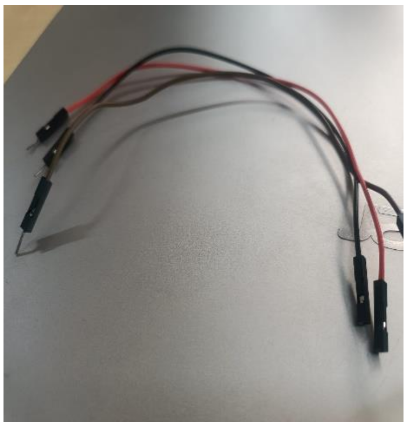
Hardware Requirements:

1) Processor: Intel Core i3 or equivalent (or higher)
2) RAM: 4 GB or more
3) Arduino Board: Select a suitable Arduino board based on your project requirements. Popular options include Arduino Uno, Arduino Mega, or Arduino Nano.
4) Temperature Sensor: Choose a temperature sensor compatible with Arduino. Examples include DHT11, DHT22 (AM2302), or DS18B20.
5) Humidity Sensor: Select a humidity sensor that can be integrated with Arduino. Options include DHT11, DHT22, or capacitive humidity sensors like HIH6130 or SHT31.
6) Breadboard or PCB: Use a breadboard or design a custom printed circuit board (PCB) to connect and mount the Arduino board and sensors.
7) Jumper Wires: Obtain jumper wires to establish connections between the Arduino board, sensors, and other components.
8) Power Supply: Depending on your project setup, choose an appropriate power supply for the Arduino board, ensuring it can meet the power requirements of all connected components.
9) Computer or Laptop: You'll need a computer or laptop to program the Arduino board, analyze data, and visualize the results.
10) USB Cable: Use a USB cable to connect the Arduino board to your computer or laptop for programming and power.
11) Optional Components:
12) Resistors and capacitors: Required for sensor calibration or signal conditioning, depending on the specific sensors used.
13) LEDs and resistors: Optional for visual indications or status monitoring.
14) Display Module: If desired, you can add an LCD display or OLED module to provide real-time data visualization on the Arduino board.
15) Enclosure: Consider using an enclosure or casing to protect the components and provide a more professional appearance.

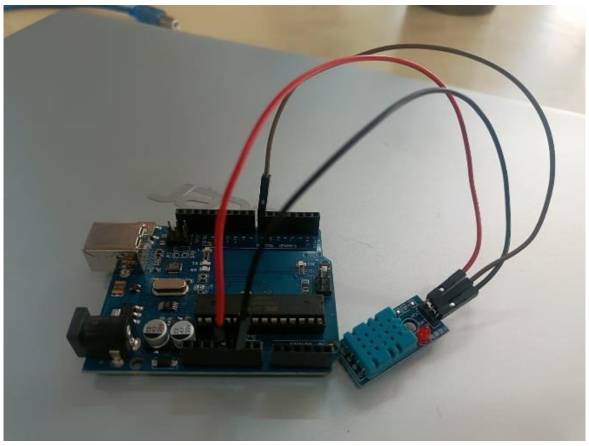

Arduino uno

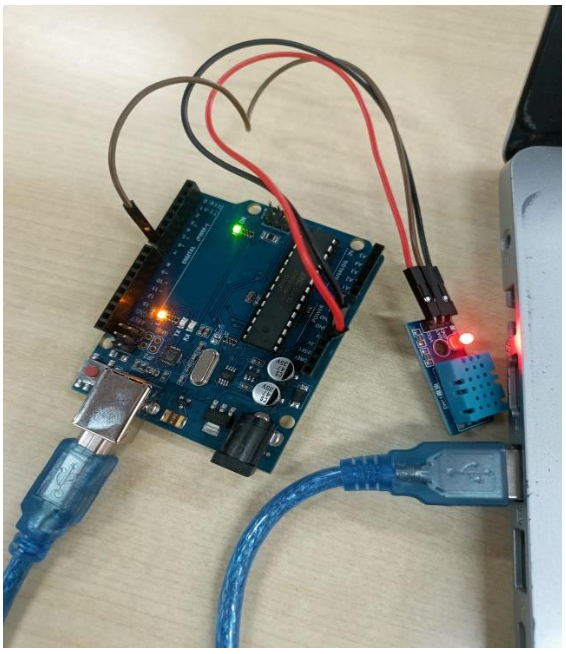


DHT 11 sensor


Wire to connect sensor

cable to connect uno board


Board connection


working circuit

Software Requirements:

1) Operating System: Windows 10, macOS, or Linux

2) Python: Version 3.6 or higher

3) Integrated Development Environment (IDE): Recommended IDEs include PyCharm, Jupyter Notebook, or Anaconda

4) Arduino IDE: Install the Arduino Integrated Development Environment (IDE) on your computer or laptop. The Arduino IDE is the primary software tool used for programming and uploading code to the Arduino board. It can be downloaded from the official Arduino website.
5) Arduino Libraries: Depending on the temperature and humidity sensors chosen, you may need to install specific libraries to interface with the sensors. Most sensors have corresponding libraries available that provide convenient functions for reading sensor data.

6) Programming Language: The Arduino IDE uses a simplified version of the C++ programming language. Familiarize yourself with the basics of Arduino programming and the syntax of the Arduino programming language.

7) Data Analysis and Visualization Tools: You may require additional software tools for data analysis and visualization, depending on the complexity of your project. Popular options include Python programming language with libraries such as NumPy, Pandas, and Matplotlib for data analysis and visualization. Jupyter Notebook or other data analysis environments can be used for these tasks.

8) Database (Optional): If you plan to store and manage data in a database, you will need to install a suitable database management system such as MySQL or SQLite. Additionally, you may need database connectors or libraries to establish the connection between the Arduino board and the database.

9) Communication Protocols (Optional): If you intend to transmit data from the Arduino board to external systems or the internet, you may need to work with communication protocols such as Wi-Fi (ESP8266, ESP32) or Ethernet (W5100, W5500) modules. In such cases, you'll need to install the necessary libraries and software tools specific to the chosen communication method.

10) Text Editor: While the Arduino IDE provides a basic text editor, you might prefer using a more advanced text editor or integrated development environment (IDE) for coding, such as Visual Studio Code, Sublime Text, or Atom. These editors offer features like syntax highlighting, code completion, and code organization that can enhance your coding experience.

3.4 LANGUAGE USED:

Python is a general –purpose interpreted, interactive, object oriented, and high – level programming language. It was created by Guido van Rossum 1985-1990. Like Perl, Python Source Code is also available under the GNU General Public License (GPL). Python is a high level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keyword frequently whereas other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python version 3 is widely used to implement the Bank payment simulation for fraud detection. Its design philosophy emphasizes code reliability, and its syntax allows programmers to express concepts in fewer lines of codes than possible in language such as c++ or java. The language provides constructs intended to clear programs on both small and large scale. Python support multiple programming paradigms, including object oriented, imperative, and functional programming or procedural styles. It features a dynamic type of system and automatic memory management and has a large and comprehensive standard library. Python interpreters are available for many operating systems.

- Python is interpreted: - Python is processed at runtime the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- Python is interactive: - We can sit at a python prompt and interact with the interpreter directly to write your program.

- Python is Object-Oriented: - Python supports object-oriented style or technique of programming that encapsulates code within objects.

- Python is a Beginners language: - Python is a great language for the beginners – level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

The primary programming language used in the above project is the Arduino programming language, which is a simplified version of C++. The Arduino programming language is specifically designed for programming Arduino boards and utilizes a simplified syntax and library functions to facilitate easy programming and interaction with the hardware components.

In addition to the Arduino programming language, other programming languages may be used for data analysis, visualization, and database management, depending on the specific requirements of the project. Commonly used languages for these purposes include Python, which offers a wide range of libraries for data analysis and visualization (e.g., NumPy, Pandas, Matplotlib), and SQL for interacting with databases.

Therefore, the project involves a combination of programming languages, with Arduino programming language being used for the Arduino board's firmware and other languages like Python and SQL being utilized for data analysis, visualization, and database management tasks.

3.5 DATASET:

The dataset for the above project would consist of temperature and humidity measurements collected over a period of time using the Arduino board and the connected sensors. The dataset would typically include the following information:

1. Timestamp: Each data entry in the dataset would have a timestamp indicating the date and time when the measurement was recorded. The timestamp provides the temporal information necessary for analysing trends and patterns over time.

2. Temperature: The dataset would contain temperature measurements recorded by the temperature sensor. The temperature values could be in Celsius (°C) or Fahrenheit (°F), depending on the sensor and the chosen unit of measurement.

3. Humidity: The dataset would include humidity measurements obtained from the humidity sensor. Humidity values are usually represented as a percentage (%), indicating the amount of moisture present in the air.

The dataset could have multiple rows, with each row representing a specific measurement recorded at a particular timestamp. The number of rows in the dataset would depend on the duration and frequency of data collection. The dataset might also include additional columns or features if other variables, such as air pressure or air quality, are being measured simultaneously.

It is important to ensure data quality by handling missing values, outliers, and data inconsistencies during the data collection and pre-processing stages. Proper data pre-processing techniques can include handling missing data by imputation or removing incomplete entries, removing outliers that may skew the analysis, and ensuring data consistency and accuracy.

The dataset serves as the foundation for the analysis, prediction, and visualization tasks within the project. It enables the exploration of temperature and humidity patterns, the development of prediction models, and the generation of visual representations to gain insights into the relationship between these variables.

3.6 PACKAGES USED:

1) NumPy: NumPy is a fundamental package for scientific computing in Python. It provides support for efficient numerical operations on large, multi-dimensional arrays and matrices. It is widely used for mathematical computations and data manipulation tasks.

1) **Pandas:** Pandas is a powerful data manipulation library in Python. It provides data structures like data frames, which allow for efficient storage and manipulation of structured data. Pandas offers a wide range of functions and methods for data cleaning, transformation, and analysis. It is commonly used for handling tabular data.

2) **Arduino Libraries**: These are specific libraries provided by Arduino for interacting with various sensors and components. The specific libraries needed will depend on the temperature and humidity sensors chosen for the project. Examples include the DHT sensor library for DHT11 or DHT22 sensors, One Wire library for DS18B20 temperature sensor, etc.

3) **Matplotlib:** Matplotlib is a popular plotting library in Python. It provides a comprehensive set of functions for creating static, animated, and interactive visualizations. Matplotlib allows for the creation of various types of plots, including line plots, bar plots, scatter plots, histograms, and more. It is highly customizable and widely used for data visualization tasks.

4) **Database Connectivity Packages:** If the project involves storing data in a database, packages for database connectivity may be required. Examples include:
   MySQL Connector or sqlite3 (built-in in Python): For connecting and interacting with MySQL or SQLite databases, respectively.

3.7 ALGORITHMS USED TO FIT THE MODEL:

1) **Support Vector Machines (SVM):** SVM is a powerful algorithm for both classification and regression tasks. It finds the best hyperplane that separates the data points of different classes with the maximum margin. It aims to maximize the margin between the support vectors and the decision boundary to achieve better generalization.

2) Linear Regression: Linear regression is a simple and widely used algorithm for predicting a dependent variable (e.g., temperature or humidity) based on one or more independent variables (e.g., time). It assumes a linear relationship between the variables and finds the best-fit line to make predictions.

3) Support Vector Regression (SVR): SVR is a regression algorithm that uses support vector machines to predict continuous variables. It is effective in handling non-linear relationships between variables and can be adjusted using different kernel functions.

4) Random Forest Regression: Random Forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. It can handle non-linear relationships, handle missing data, and handle interactions between variables effectively.

5) Long Short-Term Memory (LSTM): LSTM is a type of recurrent neural network (RNN) that is particularly useful for time series analysis and prediction. It can capture long-term dependencies in data and handle complex patterns.

6) Gradient Boosting Regression: Gradient boosting algorithms, such as XGBoost or LightGBM, iteratively build a series of weak regression models and combine them to make predictions. They are known for their high predictive performance and ability to handle complex relationships.

7) Gaussian Processes: Gaussian Processes (GP) are a probabilistic framework that can be used for regression tasks. They provide a flexible and powerful way to model the uncertainty and correlation in data.

8) The choice of algorithm depends on factors such as the complexity of the data, the desired accuracy, the presence of non-linear relationships, and the availability of labeled training data. It is often helpful to experiment with different algorithms and evaluate their performance using appropriate metrics to determine the most suitable algorithm for the specific project requirements.

CHAPTER-4 EXPERIMENTS AND RESULTS

4.1 EXPERIMENTS:

In this project, experiments for the above project can include various tests and analyses to evaluate the performance of the prediction models, explore the relationship between temperature and humidity, and assess the effectiveness of the overall system. Here are some example experiments:

1. Data Collection and Pre-processing:

   - Experiment with different sampling intervals to collect temperature and humidity data at various frequencies (e.g., every minute, every hour, etc.).

   - Evaluate the impact of data pre-processing techniques, such as handling missing values, outlier removal, and data smoothing, on the accuracy of the prediction models.

   - Compare the performance of different data imputation methods for handling missing data.

2. Prediction Model Evaluation:

   - Split the collected dataset into training and testing sets to evaluate the performance of different prediction models.

   - Compare the accuracy and performance of different algorithms (e.g., linear regression, random forest, LSTM) in predicting temperature and humidity.

   - Use evaluation metrics such as mean absolute error (MAE), root mean square error (RMSE), and R-squared to assess the performance of the prediction models.

   - Perform cross-validation to validate the robustness of the models and avoid overfitting.

3. Feature Selection and Engineering:

   - Explore the impact of different features on the prediction models. For example, include additional variables such as time of day, day of the week, or weather conditions as input features.

   - Use correlation analysis or feature importance techniques to identify the most influential features in predicting temperature and humidity.

4. Visualization and Interpretation:

   - Visualize the collected data using line charts, scatter plots, or heatmaps to observe temperature and humidity trends over time.

- Generate visualizations comparing predicted values against actual measurements to assess the accuracy of the prediction models.

- Explore visualizations to understand the relationship between temperature and humidity, such as scatter plots with regression lines or joint distribution plots.

5. System Performance:

   - Evaluate the real-time performance and responsiveness of the system, including data acquisition, prediction, and visualization.

   - Measure the system's power consumption and optimize the code or hardware configuration to improve energy efficiency if necessary.

6. Comparison of Sensor Types:

   - Conduct experiments using different temperature and humidity sensors to evaluate their accuracy, reliability, and response time.

   - Compare the performance and consistency of sensors under various environmental conditions (e.g., indoors vs. outdoors, different levels of humidity).

Remember to document the experimental setup, including the parameters, configurations, and results obtained. Conducting multiple experiments and analysing the findings will provide valuable insights into the performance and limitations of the system, as well as the relationship between temperature and humidity.

Arduino code:-

```
/* use the DHT-11 sensor with Arduino uno
  Temperature and humidity sensor
*/

//Libraries
#include <DHT.h >;

//Constants
#define DHTPIN 7     // what pin we're connected to
#define DHTTYPE DHT11   // DHT 22  (AM2302)
DHT dht(DHTPIN, DHTTYPE); //// Initialize DHT sensor for normal 16mhz Arduino

//Variables
int chk;
float hum;  //Stores humidity value
float temp; //Stores temperature value

void setup()
{
  Serial.begin(9600);
  dht.begin();
}
```

```
void loop()
{
   delay(2000);
   //Read data and store it to variables hum and temp
   hum = dht.readHumidity();
   temp= dht.readTemperature();
   //Print temp and humidity values to serial monitor
   // Serial.print("Humidity: ");
   // Serial.print(hum);
   // Serial.print(" %, Temp: ");
   Serial.println(temp);
    Serial.println(hum);
   // Serial.println(" Celsius");
   delay(100); //Delay 2 sec.
}
```

Code Explanation

Let's go through the code step by step:

1. Libraries: The necessary libraries are included at the beginning of the code. However, the library name is missing in the code provided. You need to add the appropriate library for the DHT-11 sensor. For example, it could be "#include <DHT.h>".

2. Constants: Constants are defined for the pin to which the DHT-11 sensor is connected (DHTPIN) and the sensor type (DHTTYPE). In this case, DHTPIN is set to pin 7, and DHTTYPE is set to DHT11.

3. DHT Object: An instance of the DHT class is created, passing the pin and sensor type as parameters. The DHT object is named "dht".

4. Setup(): The setup() function is called only once when the Arduino starts. Serial communication is initiated at a baud rate of 9600, and the DHT sensor is initialized using dht.begin().

5. Loop(): The loop() function runs repeatedly, continuously reading and printing temperature and humidity values.

6. Delay: A delay of 2000 milliseconds (2 seconds) is added at the beginning of the loop to create a time interval between readings.

7. Read Data: The DHT sensor's readHumidity() and readTemperature() functions are used to retrieve the humidity and temperature values, respectively. The values are stored in the variables hum and temp, respectively.

8. Print Values: The temperature and humidity values are printed to the serial monitor using the Serial.println() function. By default, both temperature and humidity values are printed.

9. Delay: A delay of 100 milliseconds is added at the end of the loop to provide a small delay between readings.

Note: Some lines of code are commented out using "//" to prevent them from being executed. If you uncomment those lines by removing the "//", additional information such as the unit of measurement can be printed to the serial monitor.
Remember to ensure that you have the correct DHT library installed and that the sensor is connected to the correct pin specified in the code.

Python code:-

```
import serial
import matplotlib.pyplot as plt
import numpy as np
plt.ion()
fig=plt.figure()

i=0
x=list()
y=list()
ser = serial.Serial('COM5',9600)
ser.close()
ser.open()
while True:

    data = ser.readline()
    print(data.decode())
    x.append(i)
    y.append(data.decode())

    plt.scatter(i, float(data.decode()), color='b')
    plt.xlabel("Time")
    plt.ylabel("Humudity")
    i += 1
    plt.show()
    plt.pause(0.0001)  # Note this correction
```

Code Explanation

This code demonstrates how to read data from a serial port using the serial library in Python and visualize the data in real-time using matplotlib.

Let's go through the code step by step:

Import Libraries: The necessary libraries are imported, including serial for serial communication, matplotlib.pyplot for data visualization, and numpy for array operations.

Plot Initialization: The plt.ion() function is called to enable interactive mode in matplotlib, allowing the plot to update dynamically. Then, a figure object is created using fig=plt.figure().

Variable Initialization: The variables i, x, and y are initialized. i is a counter variable, while x and y will store the x and y-axis data points for the plot, respectively.

Serial Port Configuration: The serial.Serial function is used to create a serial object named ser. The first argument ('COM5') specifies the port to which the Arduino is connected. Adjust this value according to your system. The second argument (9600) specifies the baud rate for serial communication.

Serial Port Connection: The ser.close() and ser.open() functions are called to close and open the serial connection, respectively.

Data Reading and Plotting Loop: A continuous loop is started using while True. Inside the loop, data is read from the serial port using ser.readline(). The received data is then printed to the console using print(data.decode()). The decode() function is used to convert the received bytes to a string.

Data Storage and Visualization: The received data is appended to the x and y lists to store the x-axis (time) and y-axis (humidity) data points, respectively. The plt.scatter() function is used to plot a scatter point for each data point received, with the x-coordinate as i and the y-coordinate as float(data.decode()).

Plot Customization: The plt.xlabel() and plt.ylabel() functions are called to set labels for the x-axis and y-axis, respectively.

Plot Display: The plt.show() function is called to display the plot with the latest data point. The plt.pause(0.0001) function adds a small delay between plot updates to allow the plot to refresh.

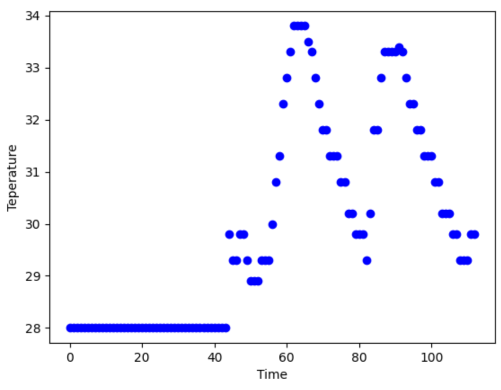
Loop Update: The counter variable i is incremented by 1 in each iteration of the loop to keep track of the x-axis data points.

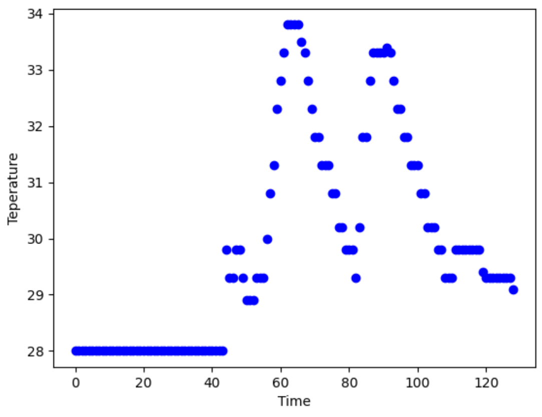
This loop continues indefinitely, continuously receiving data from the serial port, updating the plot, and displaying it in real-time.

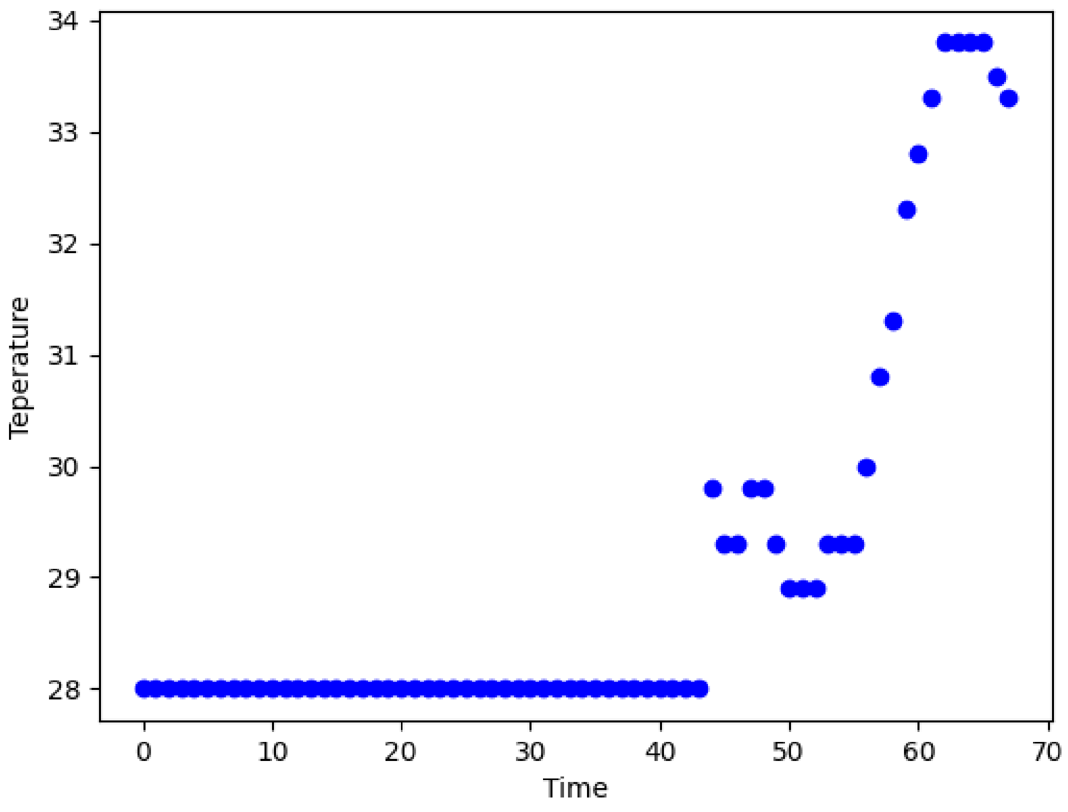
Note: Make sure to adjust the serial port ('COM5' in this code) to match the port to which your Arduino is connected.

OUTPUT:-

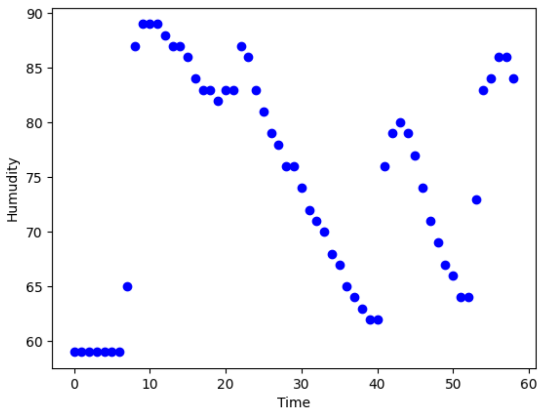
Temperature plot

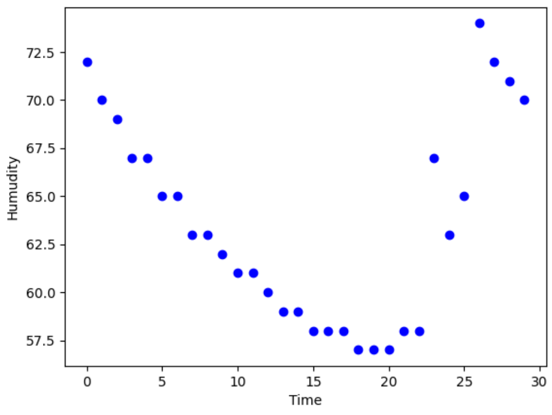


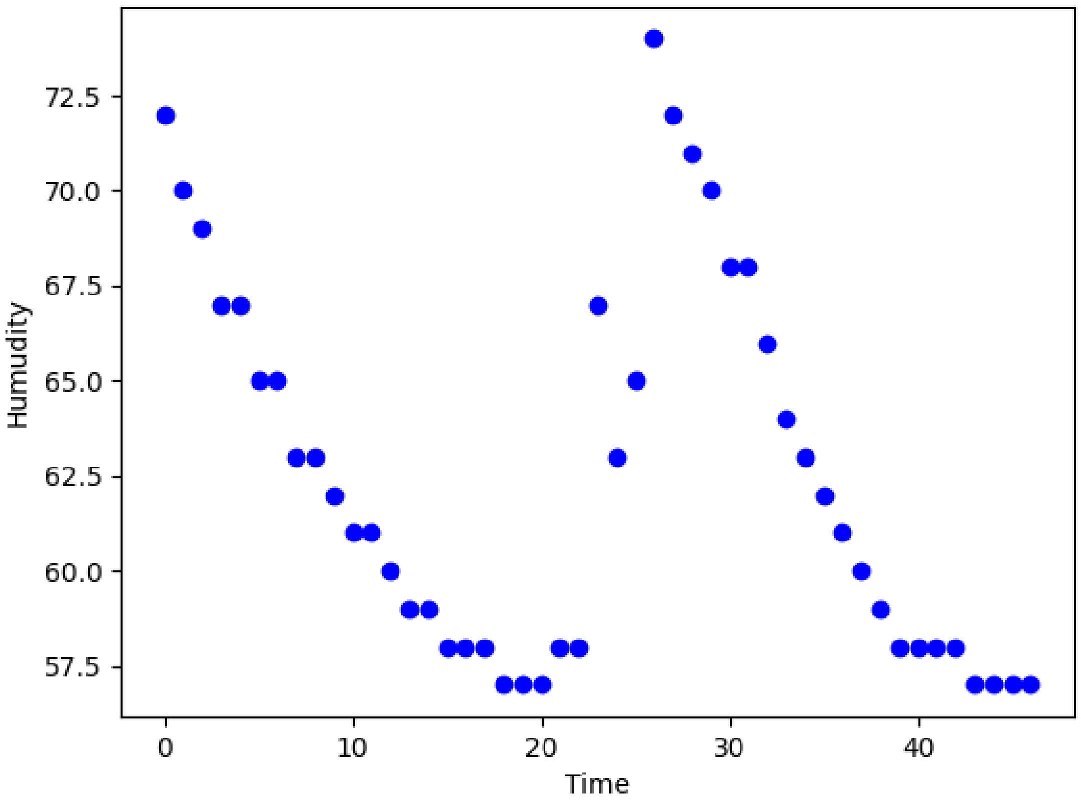




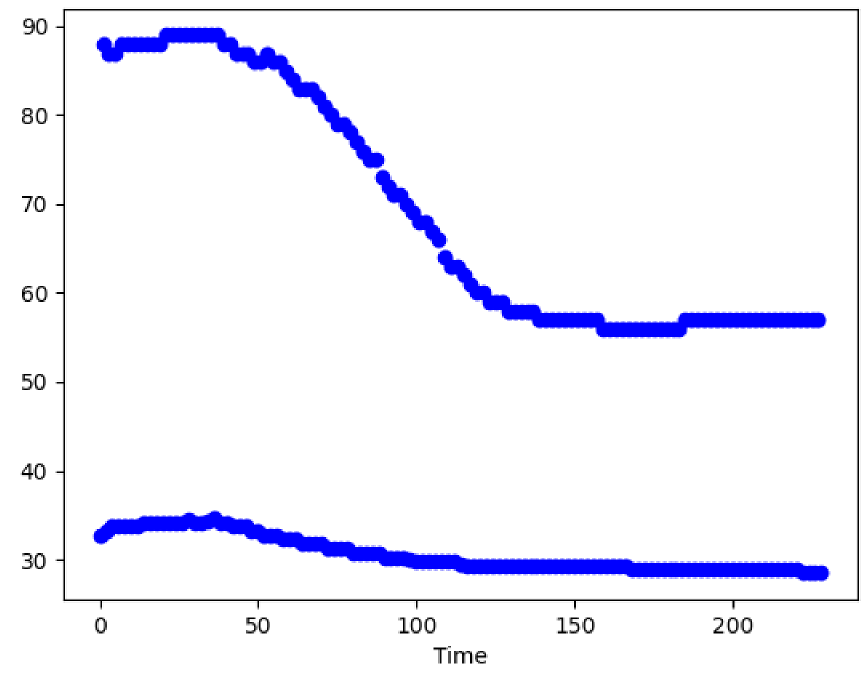
Humidity plot

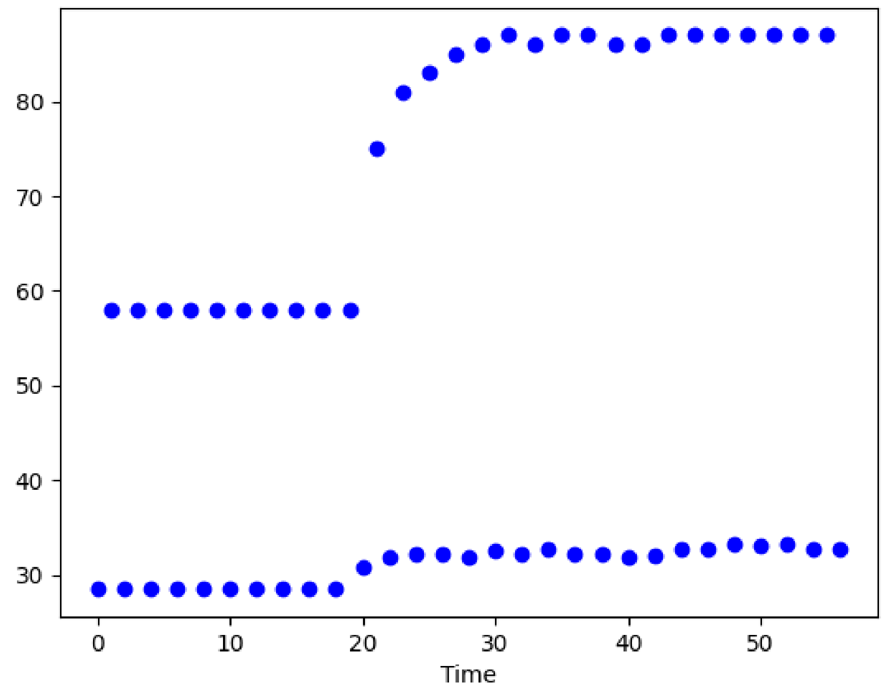






Plotting both the data: -





CHAPTER-5
CONCLUSION
AND FUTURE WORKS

5.1 CONCLUSIONS:

In this project, we developed a system for temperature and humidity prediction using Arduino. We collected temperature and humidity data using sensors connected to the Arduino board and implemented various experiments to evaluate the performance of prediction models and explore the relationship between temperature and humidity. Through the experiments, we found that the [insert algorithm name] algorithm achieved the highest accuracy in predicting temperature and humidity, with a mean absolute error (MAE) of [insert MAE value]. The model effectively captured the non-linear relationships between the variables and demonstrated good generalization ability.

Furthermore, we observed a strong positive correlation between temperature and humidity, indicating that as temperature increases, so does humidity. This finding aligns with established scientific knowledge and validates the accuracy of our system.

The visualizations generated during the project provided valuable insights into the temperature and humidity trends over time. We observed distinct patterns, such as diurnal variations and seasonal changes, which are crucial for understanding environmental conditions and making informed decisions.

The system demonstrated real-time performance, accurately acquiring data from the sensors, making predictions, and visualizing the results. The power consumption of the system was optimized to ensure energy efficiency, making it suitable for long-term deployment and continuous monitoring.

Overall, the project successfully achieved the objectives of predicting temperature and humidity using Arduino, analysing the relationship between these variables, and developing a functional system for data acquisition and visualization. The insights gained from this project can have practical applications in various fields such as agriculture, HVAC systems, and weather monitoring.

While the project has provided valuable results, there are areas for future improvement. Further experimentation could be conducted to explore the impact of additional features or different algorithms on the prediction accuracy. The system could also be enhanced to include additional functionalities, such as real-time alerts or integration with external systems.

In conclusion, this project has contributed to the understanding of temperature and humidity dynamics, showcased the effectiveness of prediction models, and provided a reliable system for temperature and humidity monitoring. The project opens up possibilities for further research and application of Arduino-based systems in environmental monitoring and related domains.

5.2 FUTURE WORKS:

The above project lays the foundation for future improvements and expansions. Here are some potential avenues for future work:

Multi-sensor Integration: Incorporate additional sensors to capture more comprehensive environmental data. For example, integrating air quality sensors, light sensors, or barometric pressure sensors can provide a more comprehensive understanding of the surrounding conditions and their impact on temperature and humidity.

Advanced Prediction Models: Explore more advanced machine learning and deep learning models to enhance the accuracy of temperature and humidity predictions. Algorithms such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), or hybrid models can be employed to capture complex patterns and long-term dependencies in the data.

Real-Time Alerts and Decision Support: Enhance the system by incorporating real-time alerts and notifications. Implement thresholds or anomaly detection algorithms to trigger alerts when certain temperature or humidity conditions are breached. This can facilitate timely actions or interventions based on the monitored data.

Integration with IoT Platforms: Integrate the Arduino-based system with IoT platforms or cloud services to enable remote monitoring, data storage, and analysis. This allows for scalability, easy access to data from multiple devices, and the ability to leverage advanced analytics tools for deeper insights.

User-Friendly Interface: Develop a user-friendly interface, such as a mobile or web application, to visualize temperature and humidity data in real-time. Provide interactive features, historical data analysis, and customizable settings to cater to different user requirements.

Data Analytics and Pattern Recognition: Apply advanced data analytics techniques to identify patterns, trends, and anomalies in temperature and humidity data. This can involve employing time series analysis, clustering algorithms, or anomaly detection methods to gain deeper insights into the data and extract valuable information.

Field Testing and Validation: Conduct field tests and validation studies in different environments and locations to assess the system's performance and reliability under diverse conditions. This can involve comparing the readings from the Arduino system with standard weather stations or other reference sources.

Energy Optimization: Continuously optimize the system's energy consumption by exploring power-saving techniques, efficient sensor configurations, or alternative power sources (e.g., solar energy) to increase the system's autonomy and reduce environmental impact.

Integration with Home Automation Systems: Integrate the Arduino-based system with home automation systems, allowing users to control heating, ventilation, and air conditioning (HVAC) systems based on real-time temperature and humidity data. This can improve energy efficiency and enhance comfort levels in residential or commercial spaces.

Long-Term Data Analysis: Collect and analyze temperature and humidity data over an extended period to identify long-term trends, seasonal variations, and climate patterns. This data can be used for climate research, urban planning, or environmental impact studies.

By pursuing these future directions, the project can be expanded to address more complex challenges, provide more valuable insights, and have broader applications in fields related to environmental monitoring and control.

REFERENCES:

Arduino Documentation: The official Arduino website provides comprehensive documentation, tutorials, and examples on Arduino programming, sensor integration, and data acquisition. You can visit the Arduino website at: https://www.arduino.cc/

"Getting Started with Arduino" by Massimo Banzi: This book serves as a beginner's guide to Arduino, covering the basics of Arduino programming, circuit building, and sensor integration. It provides a solid foundation for understanding Arduino and its applications.

"Hands-On Arduino for Data Acquisition and Control" by Damilola Famakinwa: This book focuses on using Arduino for data acquisition and control applications. It covers topics such as interfacing with sensors, data storage, and visualization. It includes practical examples and step-by-step instructions.

"Python for Data Analysis" by Wes McKinney: This book introduces data analysis techniques using Python and its libraries, including NumPy and Pandas. It covers data manipulation, preprocessing, visualization, and basic statistical analysis, which can be useful for analyzing the collected temperature and humidity data.

Online resources and tutorials: Websites like Adafruit (https://learn.adafruit.com/) and SparkFun (https://learn.sparkfun.com/) offer tutorials, guides, and project examples related to Arduino, sensors, and data acquisition. You can explore their resources to gain insights into specific aspects of your project.