JAI SHRIRAM ENGINEERING COLLEGE
TIRUPPUR – 638 660
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai
Recognized by UGC & Accredited by NAAC and NBA (CSE and ECE)

Department of Electronics And Communication Engineering

IBM - Naan Mudhalvan

Internet of Things Group 3

Phase 3 – Development part 1

Title : Noise Pollution Monitoring

NAME : Ranganayaki C

NM ID. : AU711221106027

YEAR : III

Development of my project with Requirements technology wise:

- ## AI&DS :

AI&Data Science plays a crucial role in noise pollution monitoring.Let us the discuss in the following topics

Data Collection:

Start by collecting data from IoT devices or sensors. This could be temperature, humidity, motion, or any other relevant data.

Data Cleaning:

Remove duplicates, handle missing values, and ensure data quality. This might involve using Python libraries like pandas for cleaning.

Data Visualization:

Create visualizations to better understand the data. Tools like Matplotlib or Seaborn can help here.

Data Transformation:

Depending on the analysis, transform data if necessary, for example, converting timestamps to the appropriate format.

Feature Engineering:

Create new features from existing data, like calculating averages, min, max, or adding context to the data.

Exploratory Data Analysis (EDA):

Perform EDA to identify trends, outliers, and correlations within the dataset.

Data Export:

Once the data is preprocessed, save it to a format that can be easily shared and analyzed, like CSV or JSON.

CODING:

```
Import random
Import time
Threshold = 70
Def measure_noise_level():
 Noise_level = random.randint(50, 100)
    Return noise_level
Def monitor_noise_pollution():
    While True:
        Current_noise_level = measure_noise_level
        Print(f”Current Noise Level: {current_noise_level} dB”)
If current_noise_level > threshold:
            Print(“Noise pollution alert! Noise level exceeds the threshold.”)
Time.sleep(30)
If __name__ == “__main__”:
    Print(“Noise Pollution Monitoring System”)
    Monitor_noise_pollution()
```

- **DAC**:

Data Loading: Start by importing your dataset into IBM Cognos. Ensure the data source is connected and properly configured.

Data Preprocessing:

Clean and transform the data as needed. You might filter out irrelevant data, handle missing values, and create calculated fields.

Data Exploration:

Explore the dataset using IBM Cognos tools to get a sense of the data’s structure and characteristics.

Report Creation:

Use the report creation features in Cognos to design visually appealing and informative reports. You can create various types of reports, such as tables, charts, and dashboards.

Data Visualization:

Leverage Cognos for data visualization. Customize charts and graphs to represent your data effectively. Choose the right visualization type based on the nature of your data.

Analysis:

Utilize Cognos for data analysis. You can perform calculations, aggregations, and filtering within the tool.

Dashboard Creation:

Assemble multiple visualizations and reports into interactive dashboards for a holistic view of the data.

Interactive Features:

Implement interactive features such as drill-through actions, filters, and prompts to allow users to explore the data further.

Documentation:

Create a comprehensive document that explains your approach, data insights, and the significance of your findings.

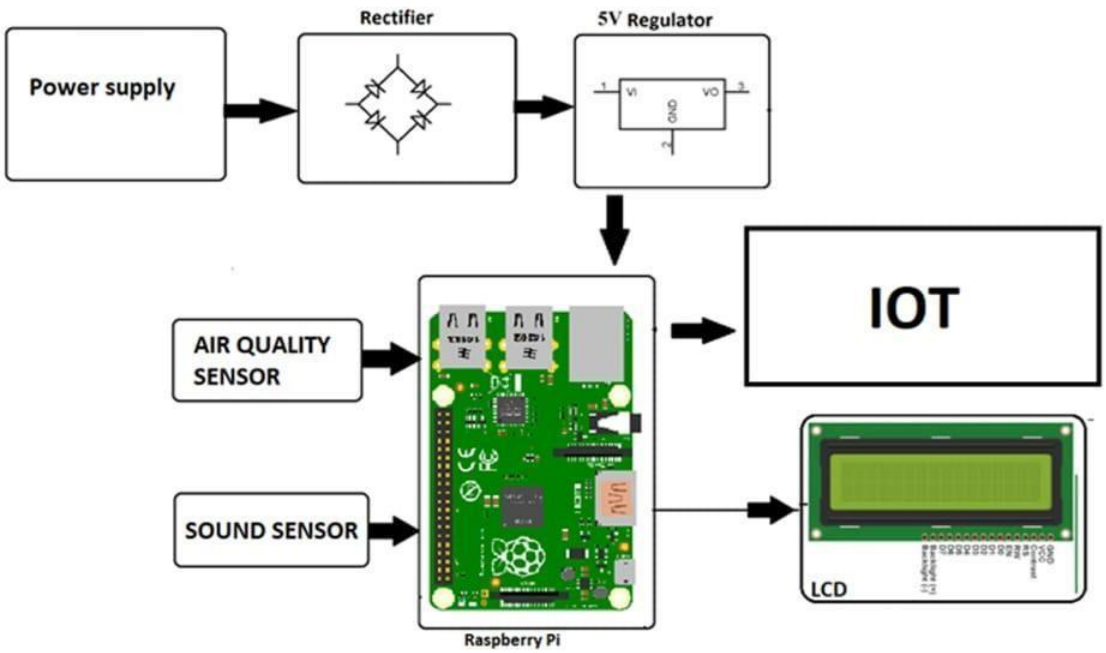
Sharing:

Share your reports and dashboards with others using IBM Cognos collaboration and sharing features.

Assessment:

Submit the document and visualizations for assessment, ensuring it covers all aspects of the project, from data loading to insights.

Block Diagram:



- **IOT:**

IoT Device Selection:

Choose appropriate IoT devices based on the project requirements, such as sensors, actuators, or microcontrollers.

IoT Device Configuration:

Configure IoT devices, including setting up network connectivity (Wi-Fi, cellular, etc.) and ensuring they can communicate with each other or a central hub.

Sensors Integration:

Connect sensors to the IoT devices to gather relevant data. Ensure proper wiring and sensor calibration.

Python Script Development:

Write Python scripts to program the IoT devices. This script should include data collection, data processing, and communication protocols (e.g., MQTT) as per project requirements.

Data Collection:

Implement code to collect data from connected sensors. Ensure data accuracy and consistency.

Data Processing:

Include data processing and analysis within the script, such as filtering, aggregation, or realtime computations.

Communication Setup:

Establish communication protocols to transmit data to a central server or cloud platform for further analysis and storage.

Security Measures:

Implement security measures to protect data and the IoT network from unauthorized access.

Error Handling:

Include error handling in the script to address potential issues that may arise during device operation.

Testing and Validation:

Thoroughly test IoT devices and the Python script in a real-world environment to ensure they function as expected.

Documentation:

Create detailed documentation that covers the hardware setup, code explanation, and configuration procedures.

Report Generation:

Generate a project report that summarizes the deployment, script development, and testing results.

Assessment:

Share the project document and findings with your team or instructors for assessment. This could be done in the form of a report or presentation

- **CAD**:

Project Setup:

Begin by setting up your project environment in IBM Cloud Foundry, including account provisioning, space creation, and organization configuration.

Application Development:

Develop the necessary applications or services tailored to your project requirements. Utilize relevant programming languages and frameworks supported by CF.

Code Version Control:

Implement a version control system (e.g., Git) to manage and track changes in your application codebase.

Application Testing:

Rigorously test your applications to ensure they perform as expected. This may involve unit testing, integration testing, and user acceptance testing.

Deployment:

Deploy your applications on IBM Cloud Foundry. Ensure that they are correctly configured, and any dependencies are resolved.

Scaling and Resource Management:

Leverage CF's scaling capabilities to manage resources and application instances dynamically based on usage and demand.

Service Integration:

Integrate with various cloud services, like databases, messaging services, or third-party APIs, as required by your project.

Monitoring and Logging:

Implement monitoring and logging mechanisms to track application performance, diagnose issues, and gather usage insights.

Security Measures:

Address security concerns by implementing authentication, authorization, and data encryption in your applications.

Documentation Creation:

Develop comprehensive project documentation that includes architecture diagrams, code explanations, configuration details, and deployment instructions.

Testing Results:

Describe the outcomes of your testing, including any challenges faced and how they were resolved.

Report Generation:

Create a well-structured project report summarizing the entire project lifecycle, from setup to deployment and testing.

Assessment Sharing:

Share the project document with your team or assessors for evaluation. This can be done in the form of a written report or a presentation.