NOISE POLLUTION MONITORING

Phase 4 Submission Document

Team Members:

MAHALAKSHMI S

PETCHISHRI S

MATHIVATHANI T

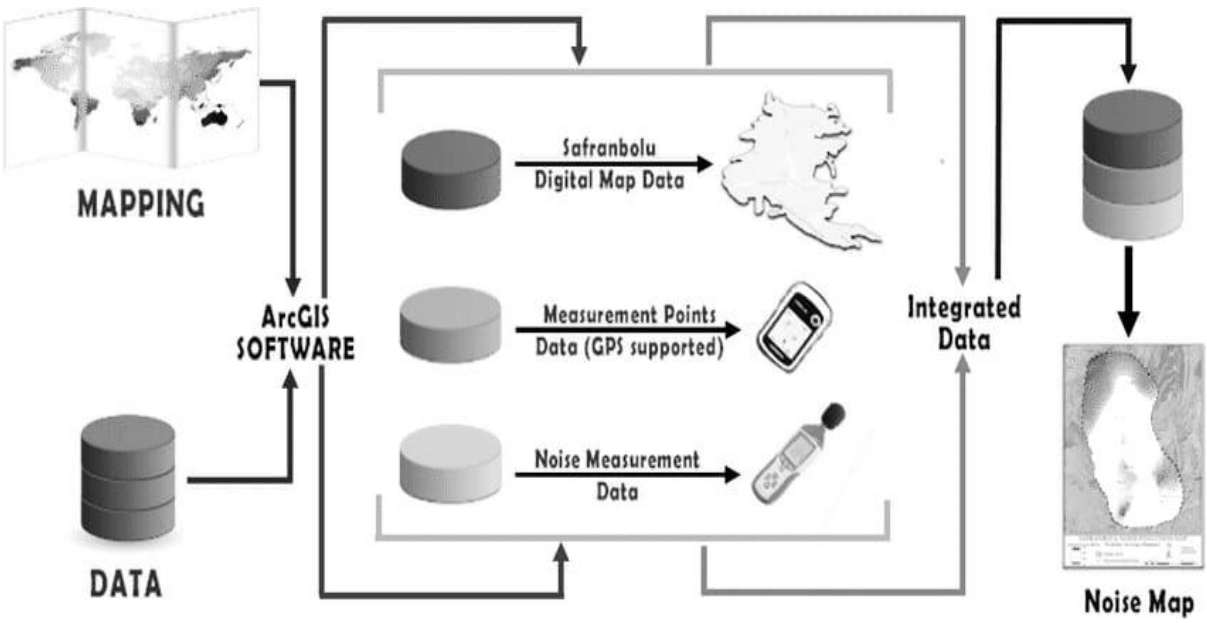
DEVATHARSHINI S

KANIKA B

Project: Noise Pollution Monitoring

Phase 4: Development Part 2

Introduction:



- Noise pollution is a growing concern in urban and industrial areas, impacting the quality of life and public health. It arises from various sources such as traffic, industrial activities, construction, and even recreational events. To effectively manage and mitigate noise pollution, it's crucial to have real-time monitoring systems in place.
- Internet of Things (IoT) technology has emerged as a powerful tool for addressing this issue by providing continuous and data-driven insights into noise pollution levels. IoT in noise pollution monitoring involves the use of interconnected sensors, data analytics, and cloud computing to track and analyse environmental noise.

Noise Pollution Monitoring Information in Mobile app:

Hardware Components:

1.Noise Sensors:

- Choose appropriate noise sensors such as microphones or sound level meters.
- You can use MEMS (Micro-Electro-Mechanical Systems) microphones for accurate noise measurement.

2.Microcontroller:

- You'll need a microcontroller to interface with the noise sensor and transmit data to the cloud.
- Popular choices include Arduino, Raspberry Pi, or specialized IoT development boards like ESP8266 or ESP32.

3.Connectivity:

- You'll need a Wi-Fi or cellular module to connect the microcontroller to the internet.
- Wi-Fi is a common choice for indoor applications.

4.Power Supply:

- Ensure a reliable power source for the hardware, whether it's a battery or a power outlet, depending on your application.

Software and Firmware:

1.Microcontroller Programming:

- Write firmware for the microcontroller to read data from the noise sensor and send it to the cloud.
- For example, if you're using an ESP8266 or ESP32, you can program it using Arduino IDE or Micro Python.

2.Data Transmission:

- Set up communication protocols like MQTT or HTTP to send data from the microcontroller to the cloud server.
- You'll need to create an IoT account and obtain the necessary credentials.

Mobile App Development:

1.Front-end Development:

- Create a mobile app (iOS and Android) to allow users to access the noise data.
- You can use native development (e.g., Swift for iOS and Java/Kotlin for Android) or cross-platform frameworks (e.g., React Native, Flutter).

2.App Features:

- Implement features such as real-time noise level monitoring, historical data display, customizable alerts, and user authentication.

3.API Integration:

- Develop APIs or use existing APIs to fetch data from the cloud platform.
- Integrate this data into your mobile app for display and user interaction.

4.User Interface (UI):

- Design a user-friendly interface with charts, graphs, and settings for users to customize their monitoring preferences.

IOS and Android Mobile apps:

Fig Sound Level Sensor to Control Noise Pollution

1. Define Requirements:

- Determine the specific features and functionality you want in your noise pollution monitoring web app, such as real-time data visualization, historical data analysis, and user accounts.

2. Choose a Web Development Framework:

- You can use popular web development frameworks like React, Angular, or Vue.js for building the user interface and managing the application's front-end.

3. Create a Responsive Design:

- Develop a responsive user interface that works well on both mobile devices (iOS and Android) and desktop browsers. Utilize CSS frameworks like Bootstrap or Material-UI for a mobile-friendly design.

4. Develop the Front-End:

- Write the front-end code to create the user interface. This should include features like real-time noise level display, interactive charts, and user-friendly controls.

5. Data Visualization:

- Use JavaScript libraries like Chart.js or D3.js to create interactive charts for displaying noise level data.

Real Time Noise Level Data:

1.Connect to the Internet:

- Use the IoT device's connectivity options (Wi-Fi, Ethernet, cellular, Long Range, etc.) to connect to the internet. You'll need a network connection to transmit the data to a central server or cloud platform.

2.Data Transmission:

- Send the processed noise level data to a central server or cloud platform for storage and further analysis. You can use MQTT, HTTP, or other suitable protocols to transmit the data securely.

3.Cloud Platform Integration:

- Store the incoming noise level data in a cloud-based database or storage system, such as Amazon Web Services (AWS), Microsoft Azure, Google Cloud, or a custom server. Cloud platforms often provide data analytics and visualization tools for further analysis.

4.Real-Time Monitoring:

- Set up a real-time monitoring system that can continuously display the noise levels. This could involve a web dashboard, mobile app, or other user interface that provide instant access to the data.

HTML using Noise Pollution Monitoring:

```
<!DOCTYPE html>
<html>
<head>
<title>Noise Pollution Monitoring</title>
<script>
// JavaScript code for updating noise data
function updateNoiseData() {
// Simulated data for demonstration (replace with real data)
let noiseLevel = Math.random() * 100;
document.getElementById("noise-level").innerHTML=
noiseLevel.toFixed(2) + " dB";
}
// Periodically update noise data (e.g., every 5 seconds)
```

```
setInterval(updateNoiseData, 5000);
</script>
</head>
<body>
<h1>Noise Pollution Monitoring</h1>
<h2>Historical Noise Data</h2>
Add charts, graphs, or a table to display historical noise data here
&amp;copy; 2023 Noise Monitoring IoT System
</footer>
</body>
</html>
```

Output:

```
Noise Monitoring IoT System
Update Noise Data:5000db
Replace Real Data:100db
Historical Noise Data:200db
```

Conclusion:

- IoT systems rely on a network of sensors and devices to collect data. For noise monitoring, microphones and sound level sensors are used, while for pollution monitoring, various sensors like gas detectors, particulate matter sensors.
- In conclusion, IoT-based noise and pollution monitoring systems play a crucial role in creating smarter and more sustainable cities.

Thank you