FGCU

Senior Design Project

An Open-Source Framework for Managing Low-Cost Sensors

Ali Irmak Ozdagli, PhD[2]

Assistant Professor of Civil Engineering

US Infrastructure

- American Society of Civil Engineers (ASCE) publishes reports periodically.
- The infrastructure is poor and at risk
- Approaching the end of their service life
  - Significant deterioration
  - Condition and capacity issues
- Additional pressure due to emerging natural and man-made hazards
- Required funding by 2029 for US:
  $5,937 billion
- If funding not met by 2039:
  $10 trillion in losses to GDP

I-35W Mississippi River Bridge

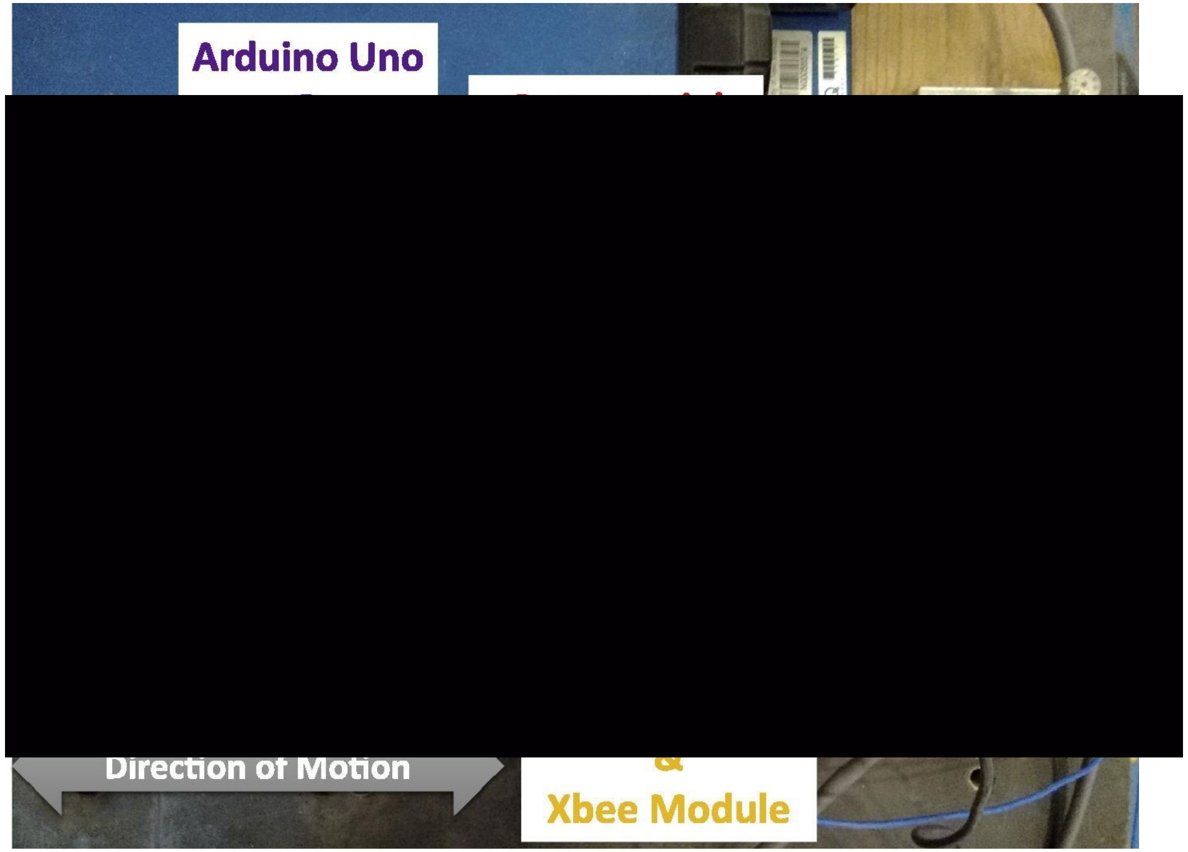
I-40 Mississippi River Bridge

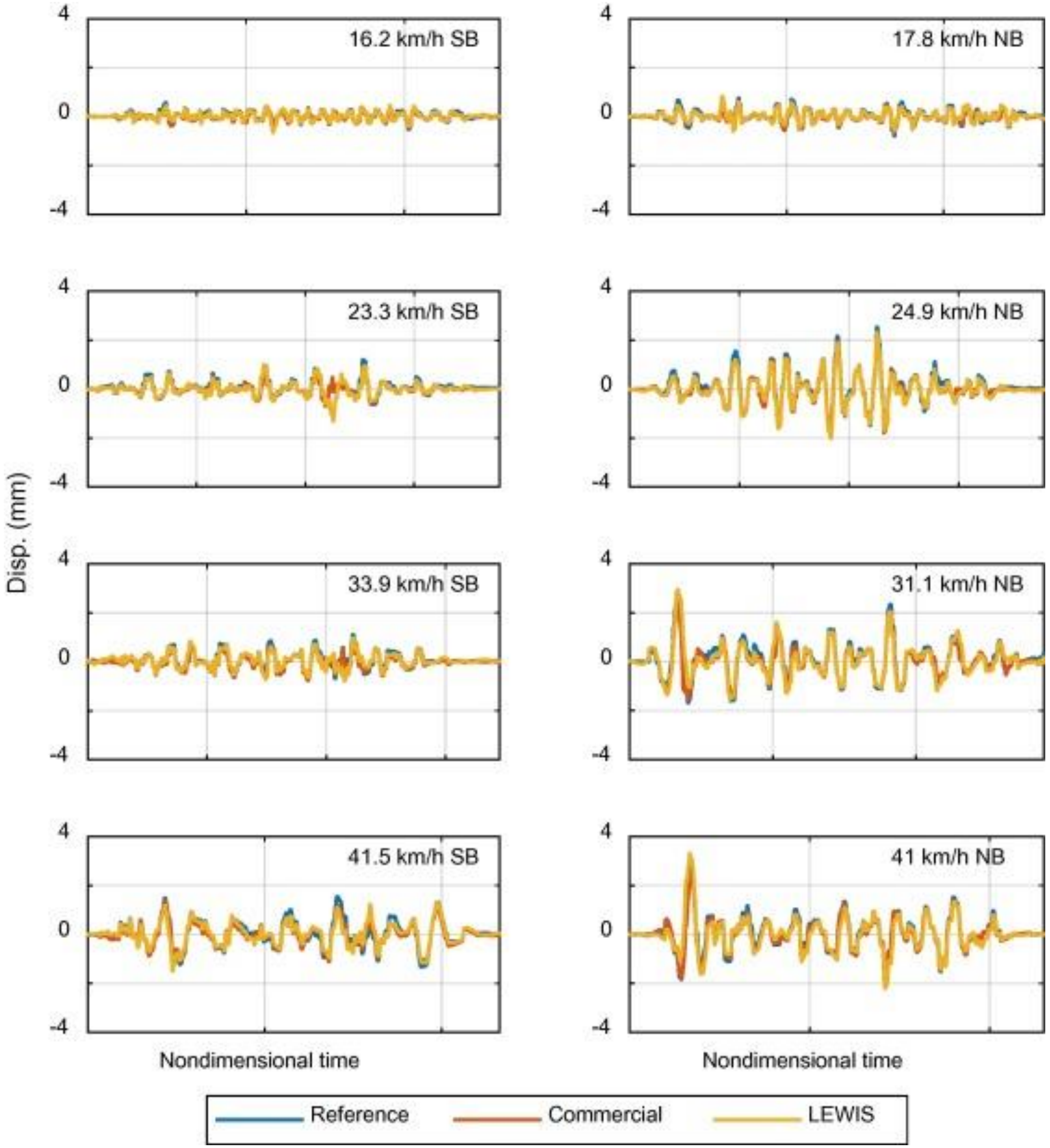
Fern Hollow Bridge

Decision Making through Structural Health Monitoring

- Prioritization of operations of maintenance, repair, and replacement (MRR)
- Qualitative (inspection) and quantitative (sensing) assessment
- SHM: Process of damage detection and characterization of structures

Prior Work


Arduino Uno
[redacted]
Direction of Motion
Xbee Module


16.2 km/h SB
17.8 km/h NB
23.3 km/h SB
24.9 km/h NB
33.9 km/h SB
31.1 km/h NB
41.5 km/h SB
41 km/h NB
Disp. (mm)
Nondimensional time
Nondimensional time
Reference
Commercial
LEWIS

Low-Cost Sensing in other industries

Wind Turbine Monitoring

Flood Warning Monitoring

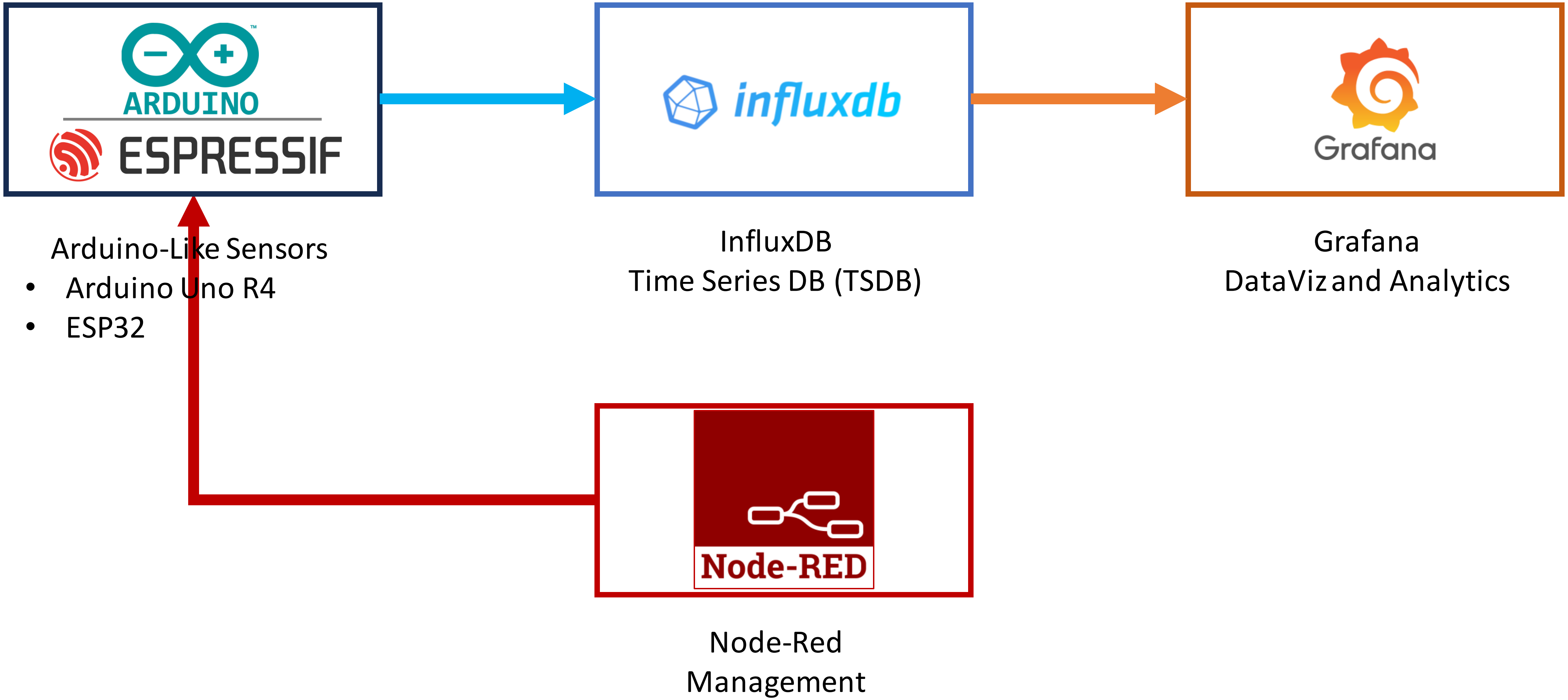
Air Quality Monitoring

Project Overview and Goals – A Modest Proposal

- A need for managing more than one sensor
  - Hardware: Low-cost Sensor
  - Backend: Data storage
  - Frontend: Data visualization
  - Sensor Management
  - **Open-Source: FGCU's Visibility**

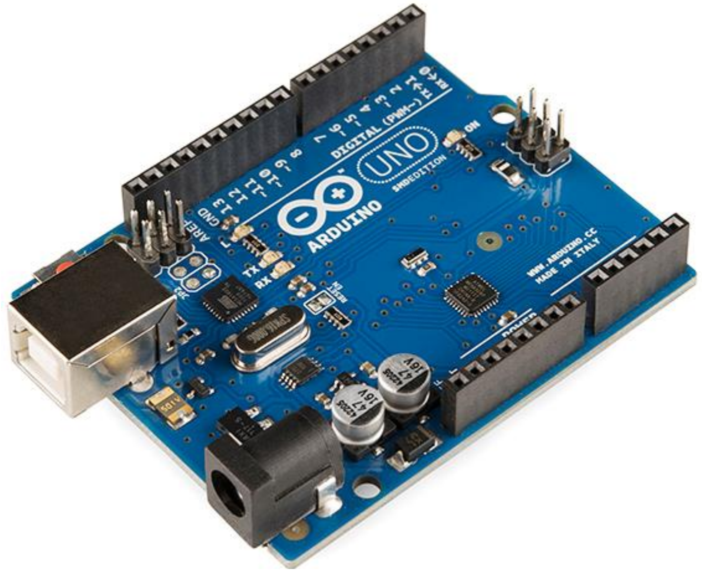
- Develop an open-source framework for managing low-cost sensors using Arduino, InfluxDB, Grafana, Node-RED

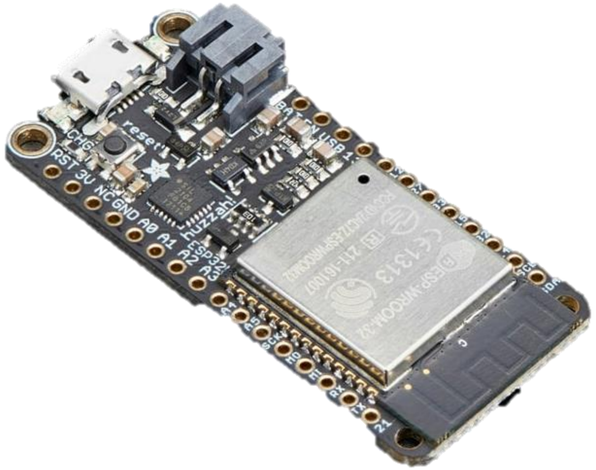
Tech Stack


ARDUINO
ESPRESSIF
influxdb
Grafana
Node-RED
Arduino-Like Sensors
• Arduino Uno R4
• ESP32
InfluxDB
Time Series DB (TSDB)
Grafana
DataViz and Analytics
Node-Red
Management

Tech Stack

- Arduino: **Open-source** hardware and software
- ESP32: Low-cost, low-power SoC microcontroller





- Integrated 802.11b/g/n
- Classic BT and BLE
- 8 MB Flash
- Deep sleep: 70 uA

ARDUINO

ESPRESSIF

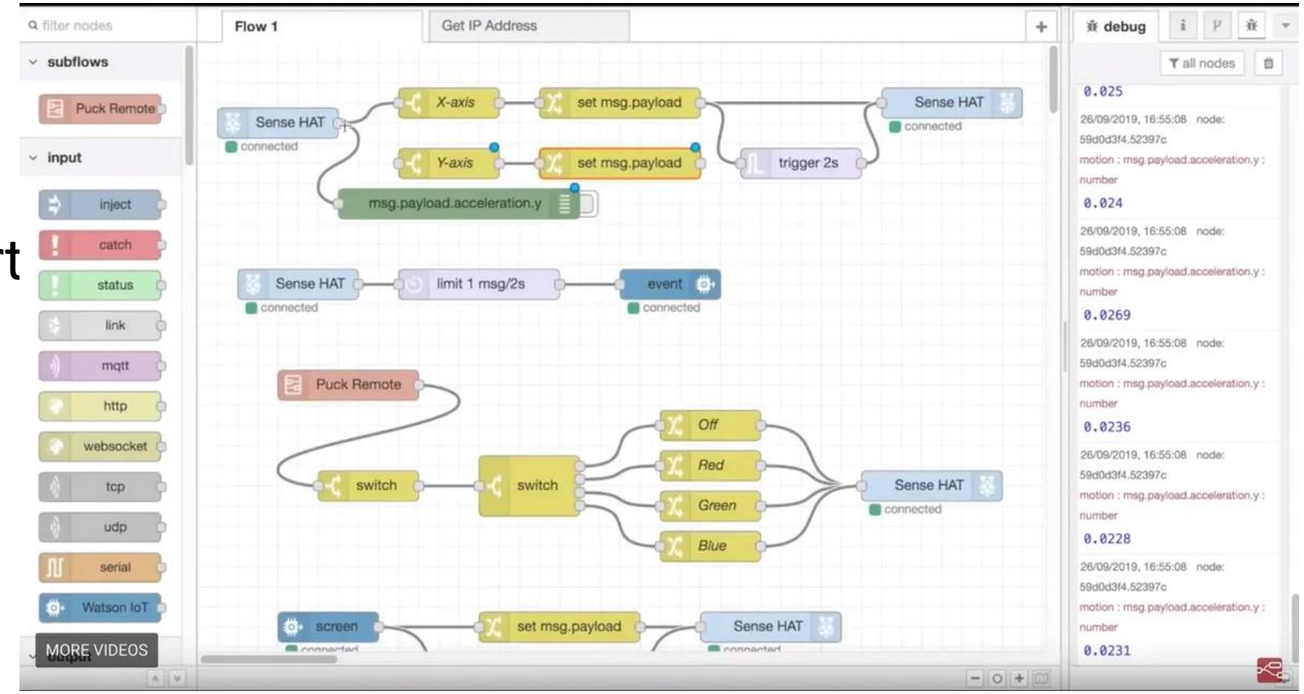
Tech Stack

- InfluxDB
  - **Open-source** time series database
  - NoSQL-like
  - Built specifically for handling **time-stamped** measurements
  - Timestamps can be second, millisecond, microsecond, or nanosecond precision
- Grafana
  - **Open-source** analytics and interactive visualization
  - Web browser-based

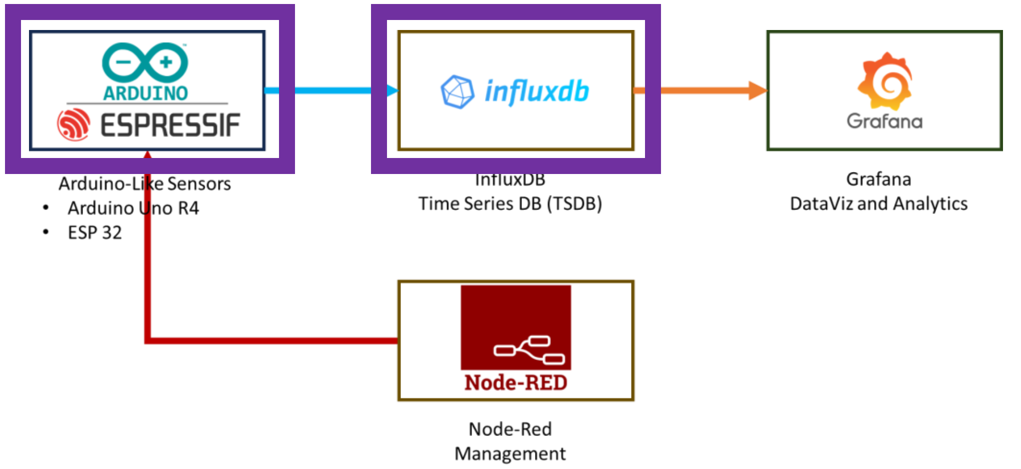
Tech Stack

- Node-RED
  - **Open-source**
  - Flow-based, low-code visual development tool
  - Web browser-based
  - Developed for wiring IOT devices
  - Utilizes MQTT

(Message Queue Telemetry Transport



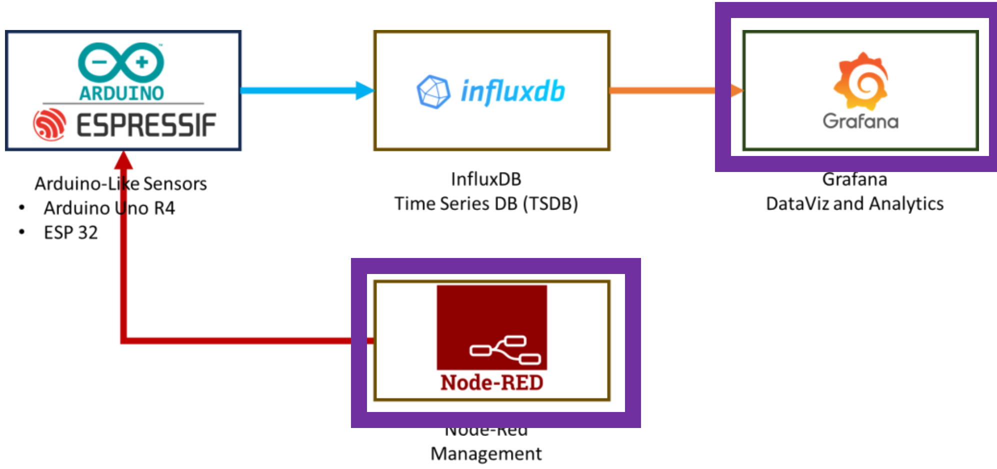
Project Objectives



1) Program ESP32 (Hardware)
   a. Generate Slow-rate data (humidity, temperature)
   b. Generate Fast-rate data (wind speed, acceleration)
   c. Connect to Internet over WiFi using ESP32
   d. Persistent location
   e. Timestamp precisely using GPS

2) Set up InfluxDB (Backend)
   a. AWS or local server
   b. Send slow-rate sensor data to DB
   c. Store sensor data persistently
   d. Send fast-rate sensor data to DB

Project Objectives

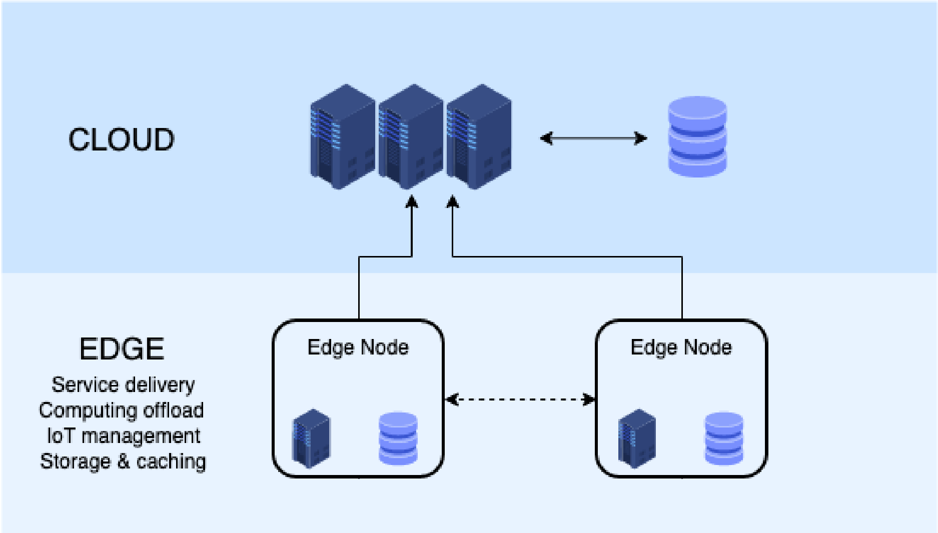


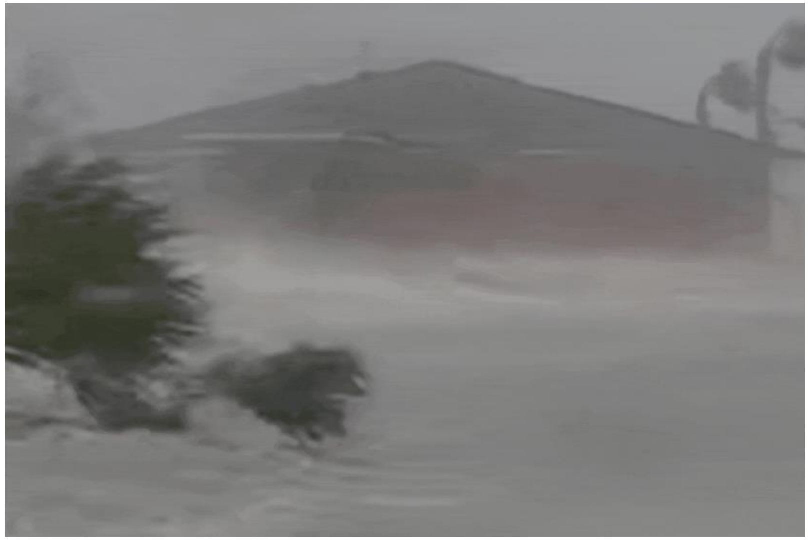
3) Set up Grafana (Frontend)
   a. AWS or local server
   b. Connect Grafana to InfluxDB
   c. Slow-rate data viz
   d. *Fast-rate data viz*
   e. *Research on real-time data visualization*

4) *Set up Node-Red (Management)*
   a. *AWS or local server*
   b. *Understand MQTT model*
   c. *Remote start/stop, sensor quality monitoring*

Additional Components

- Edge Computing
  - Running ML model on the microcontroller by **Andy Holm**
  - Analyze data on edge real-time
  - Advantage of sending only essential data to reduce bandwidth usage
  - Compress and optimize data transmission
- Potential Implementation
  - Hurricane/Storm Probe (idea by **Andy Holm**)
  - Collect real-time data (wind, acceleration, humidity)
  - Early and accurate storm data collection for improved prediction and preparedness
- Power Management





Challenges

1) Hardware Integration Complexity
2) Data Transmission and Connectivity
3) Data Storage and Management Optimization
4) Data Visualization Responsiveness
5) Power Management

Project Timeline

			2023				2024				
Phase	**Milestone**	**Task**	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May
Phase 1 Preparation and Setup	1	Acquire necessary hardware, including ESP32 devices. Establish a development environment for ESP32 programming.									
	2	Begin programming ESP32 devices for basic functionality. Research and choose appropriate sensors for data generation.									
Phase 2 Sensor Development and Preliminary Data Generation	3	Continue programming ESP32 devices for slow-rate data (humidity, temperature) generation. *Develop code for precise timestamping using GPS.*									
	4	*Program ESP32 devices for fast-rate data (wind speed, acceleration) generation.* Ensure data is transmitted to the internet over WiFi.									
	5	Set up InfluxDB on the chosen platform (AWS or local server). Implement the database schema for storing slow-rate and fast-rate sensor data. Configure ESP32 devices to send data to InfluxDB.									
Phase 3 Data Visualization Research and Implementation	6	Set up Grafana on the chosen platform (AWS or local server). Connect Grafana to InfluxDB. Create initial slow-rate and fast-rate data visualizations.									
	7	Research and experiment with real-time data visualization techniques and tools. Begin implementing data visualization features in Grafana.									
Phase 4 System Optimization and Feature Development	8	Set up Node-RED on the chosen platform (AWS or local server). Gain a deep understanding of the MQTT model for sensor communication.									
	9	Implement remote start/stop functionality for ESP32 devices via Node-RED. Develop sensor quality monitoring features within Node-RED.									
	10	Refine and optimize all components of the system. Conduct thorough testing and debugging.									
Phase 5 Final Presentation and Project Closure	11	Deliver the final presentation. Demonstrate the system's functionality and capabilities. Discuss the project's outcomes, challenges, and lessons learned.									