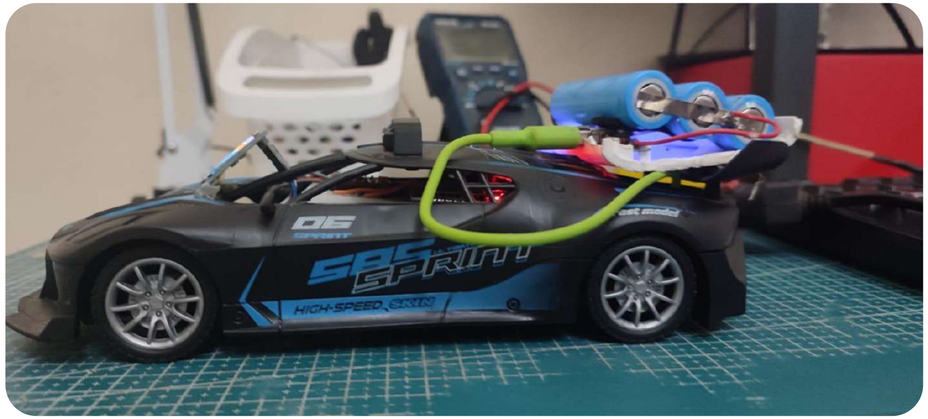
Presented by:

BAARAR Mohamed

Supervised by:

Pr, Aimad Karkouch , **At Ibn Zohr university**

كلية العلوم التطبيقية - أيت ملول
ⵜⴰⵖⵡⴰⵍⵜ ⵏ ⵜⵓⵙⵙⵏⵉⵡⵉⵏ ⵜⵉⵖⵉⵡⵉⵏ – ⴰⵢⵜ ⵎⵍⵍⵓⵍ
FACULTE DES SCIENCES APPLIQUEES – AIT MELLOUL



Internet Of Things

Master SESN M1 SOC 2023-2024

Smart IoT-Based Operator: RC Car WIFI Control System with Real time sensors data and Camera feed.

Plan

- Project Overview
- System Architecture and Technologies
- Protocols and Security Considerations
- Implementation Details
- Demo
- Future Developments and vision

Project Overview

Overview

- Development of a remote-controlled (RC) car using ESP32 microcontrollers, WiFi, and an Xbox 360 controller.
- Integrate it inside an Iot system.
- Leveraging MQTT for communication, sensor-based
- environmental adaptations for speed control, and a camera module for real-time video streaming.
- Operator Receives Data from the Sensor Node governing the Area it is currently at and act upon that data (for now it’s a speed limiter based on weather conditions).

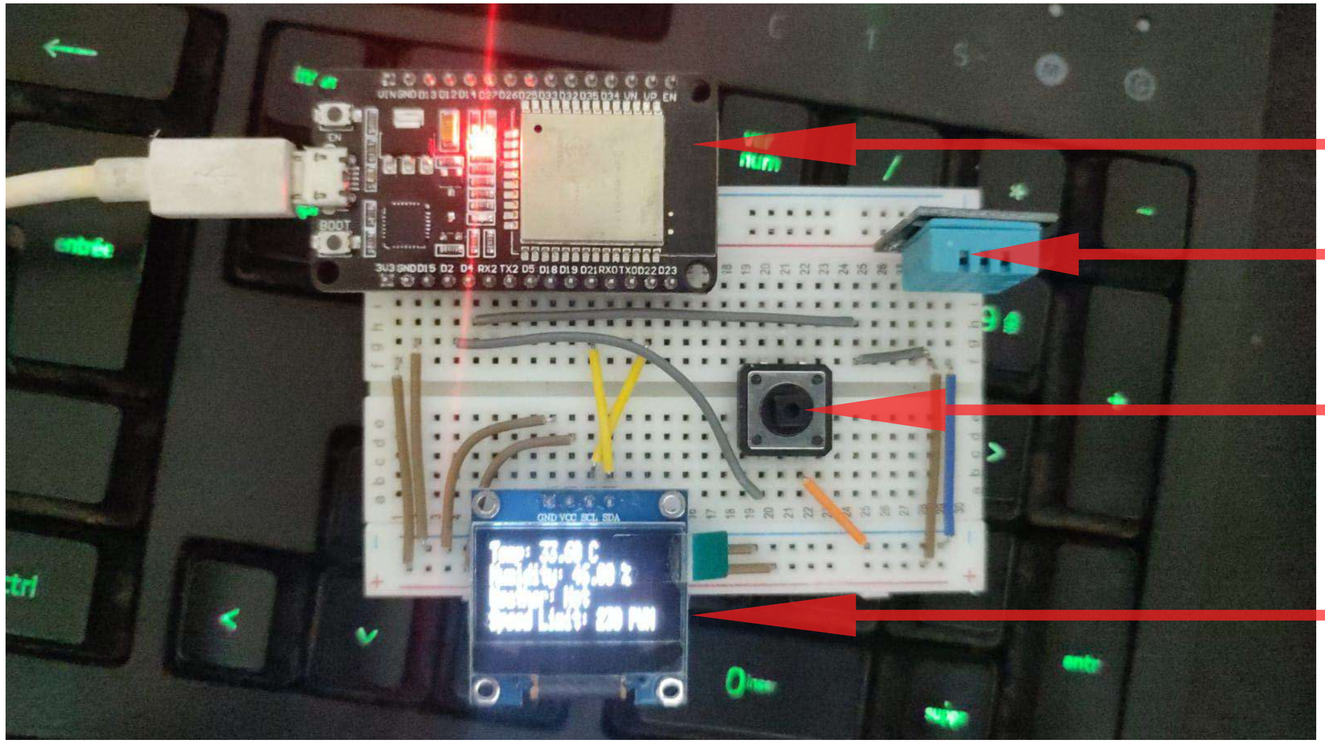
Inspiration Vehicule

System Architecture

Main Components:

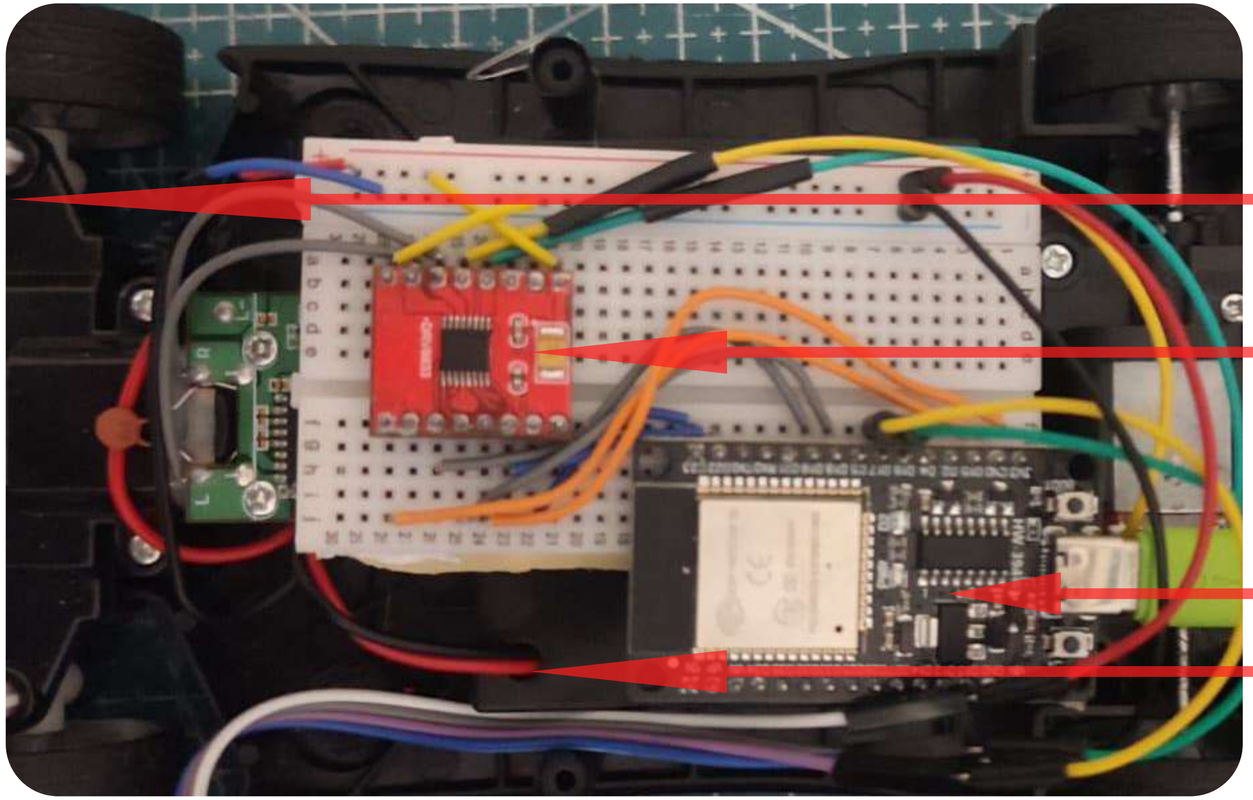
- **Sensor Node (ESP32):** Measures temperature and humidity, classifies weather conditions.
- **Control Module (ESP32):** Receives control commands from the MQTT broker and drives the RC car’s motors.
- **Camera Module (ESP32):** Streams video from the car.
- **Control Server (PC):** Interfaces with an Xbox 360 controller to send control commands via MQTT.

System Architecture



Sensor Node Prototype v1

System Architecture



I2C display (not visible here)

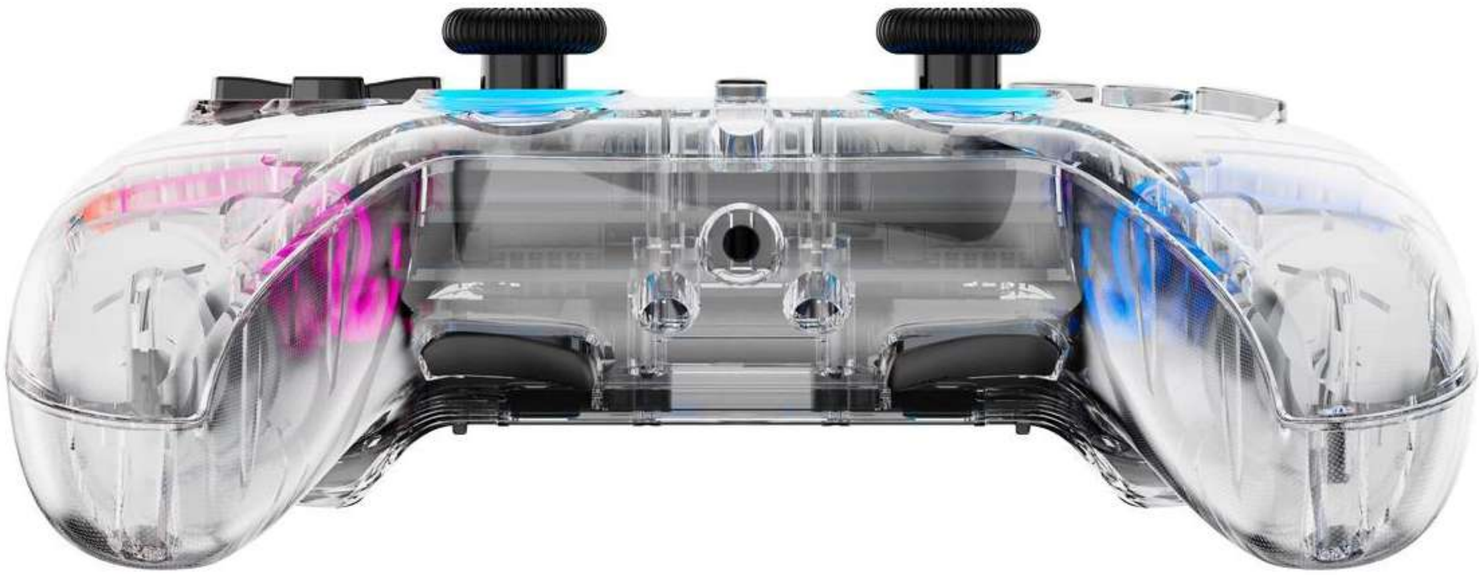
DHT11 sensor (temp & humidity)
Motor Driver (control both motors)

ESP32 micro-controller

Dedicated battery for DC motors (under)

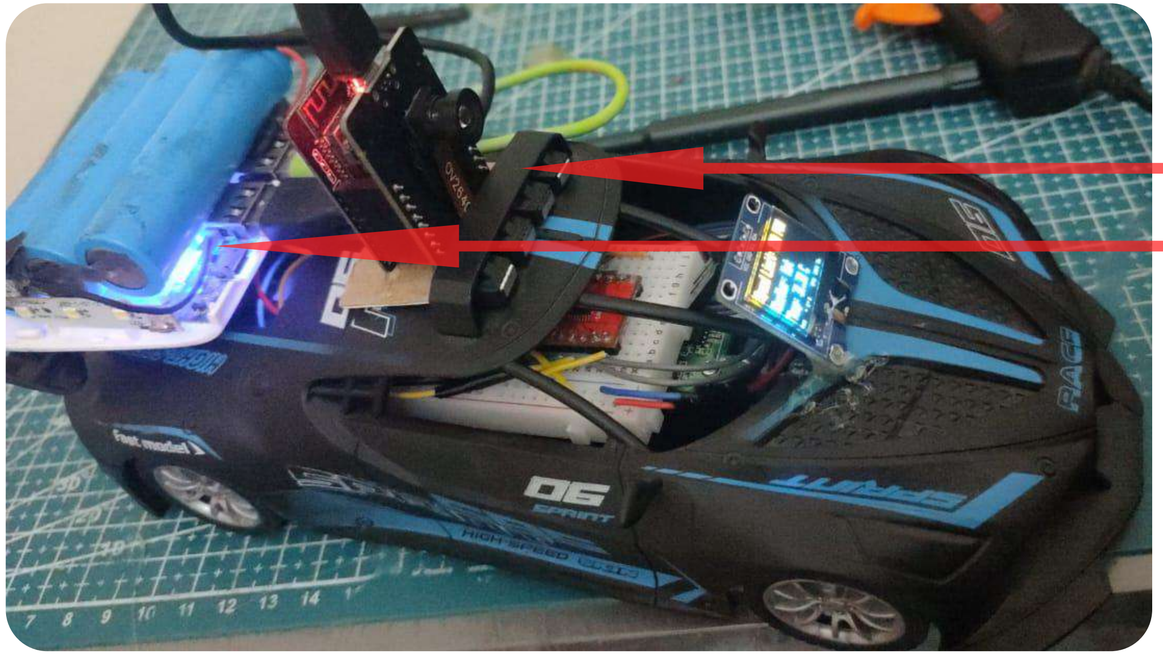
Car control system v1

System Architecture



PC controller

System Architecture

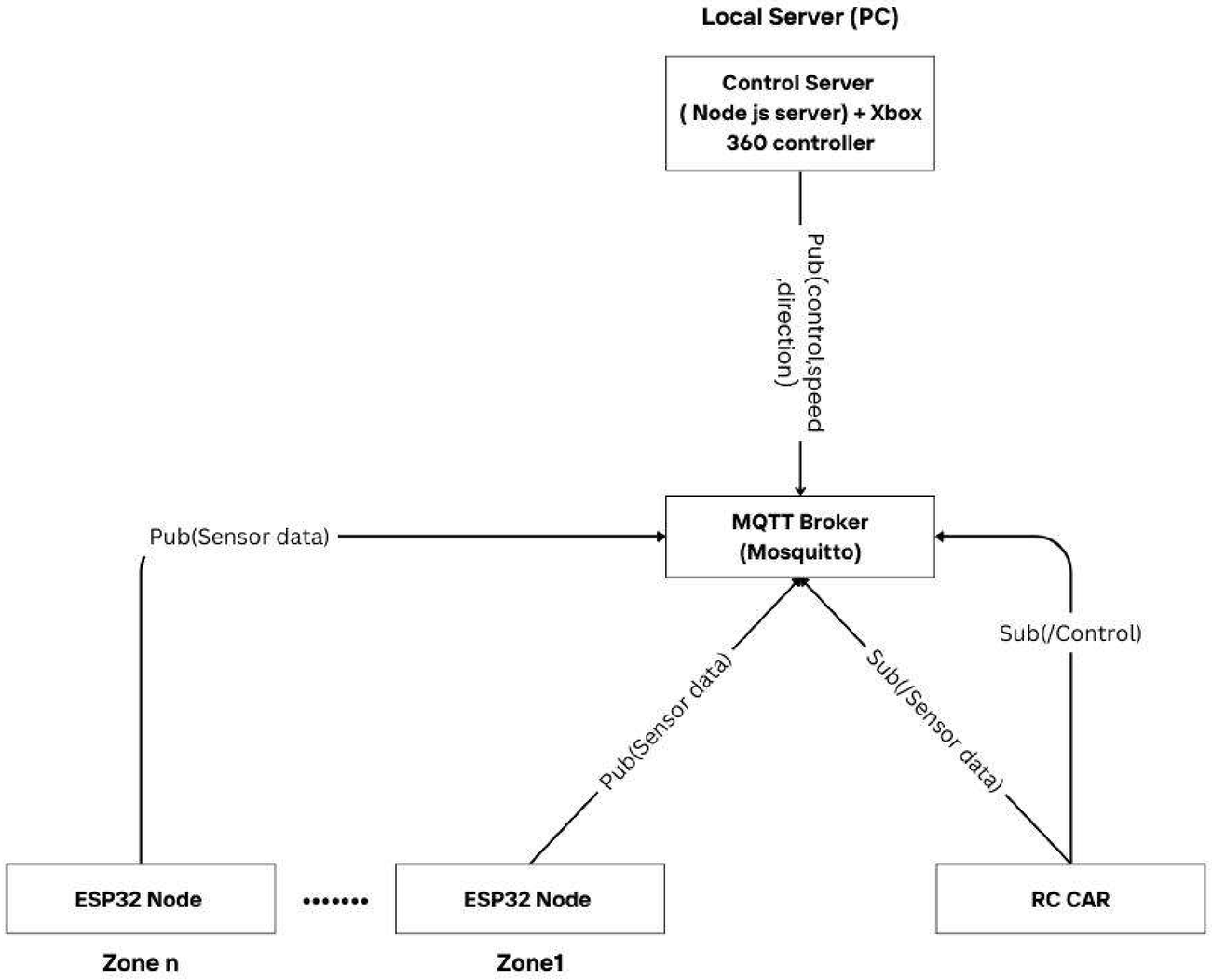


RC Operator v1

ESP32 cam module

Dedicated battery to power both ESP32s with USB PORTS

Data flow


Local Server (PC)
Control Server
(Node js server) + Xbox
360 controller
Pub(control,speed
,direction)
MQTT Broker
(Mosquitto)
Pub(Sensor data)
Pub(Sensor data)
Sub(/Sensor data)
Sub(/Control)
ESP32 Node
.....
ESP32 Node
RC CAR
Zone n
Zone1

Components and Technologies

- **ESP32 Microcontroller:**
  - Low-cost, low-power microcontroller with integrated Wi-Fi and Bluetooth.
  - Supports multiple GPIO pins and various sensors and peripherals.
- **MQTT (Message Queuing Telemetry Transport):**
  - Lightweight messaging protocol for low-bandwidth, high-latency networks.
  - Uses a publish/subscribe model, ideal for IoT applications.
- **Sensors and Actuators:**
  - DHT11 Sensor: Measures temperature and humidity.
  - Motor Driver: Controls the motors of the RC car.
  - I2C Displays: Display information such as speed and weather conditions.
- **Video Streaming:**
  - ESP32 camera module captures and streams video in real-time.
- **Xbox Controller:**
  - Provides an intuitive interface for controlling the RC car via PC.

Protocols and Security Considerations

Protocols and Security Considerations

WiFi Communication:

- Connects ESP32 modules to the local network.
- Enables communication between ESP32 modules and the MQTT broker.

MQTT Protocol:

- Publishes sensor data from the sensor module to the MQTT broker.
- Publishes control data (speed, direction) from the control server to the MQTT broker.
- Subscribes to control commands from the control server to the control module.
- Subscribes to sensor data updates by the control module for speed and direction adjustments.

SSL/TLS Security:

- Ensures secure communication between the MQTT broker and clients using SSL/TLS.
- Certificates:
  - CA Certificate: Used by clients to verify the broker’s identity.
  - Client Certificate: Used by the broker to verify the client’s identity (not used in this basic setup).
- MQTT authentication using username and password.
- Broker configuration to allow only authorized clients to publish or subscribe to specific topics.

Implementation Details

Protocols and Security Considerations

SetupMosquitto Broker:

- An open-source MQTT broker used to route messages between clients.
- Setup on a local machine (IP: localhost, Port: 8883).

Sensor Module:

- Connect DHT11 sensor and OLED display to ESP32.
- Publish sensor data to the broker at regular intervals.

Control Module:

- Connect motor driver and OLED display to ESP32.
- Subscribe to control commands and sensor data updates.
- Adjust motor speed and direction based on received commands and data from sensor nodes.

Camera Module:

- ESP32 camera captures and streams video to a specified endpoint.

Control Server (PC):

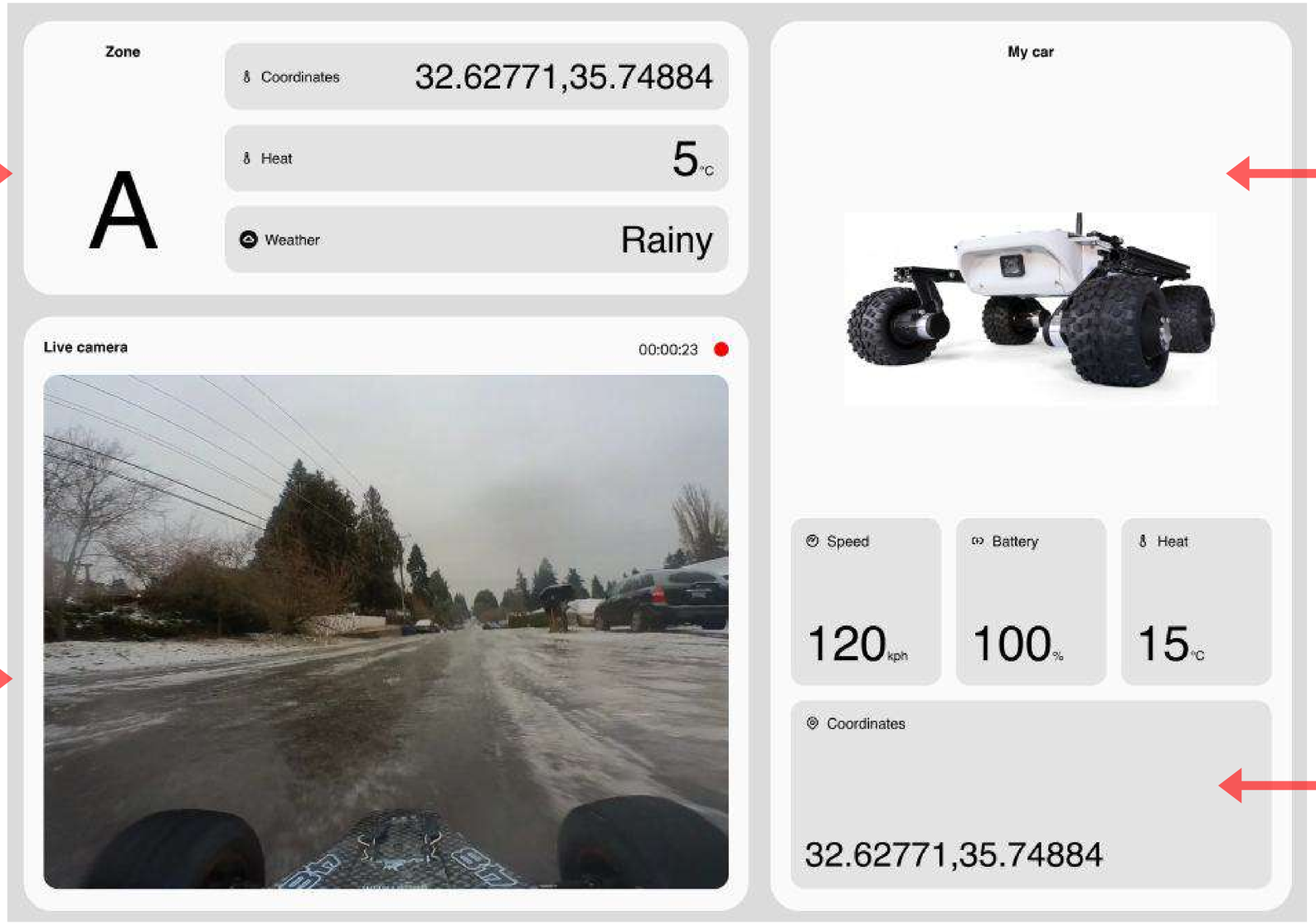
- Interfaces with an Xbox controller to read inputs using Node.js and node-hid.
- Publishes control commands based on user input to the broker.

Demo

User Interface v1 concept

Current Sensor node Live data

Cam Live feed



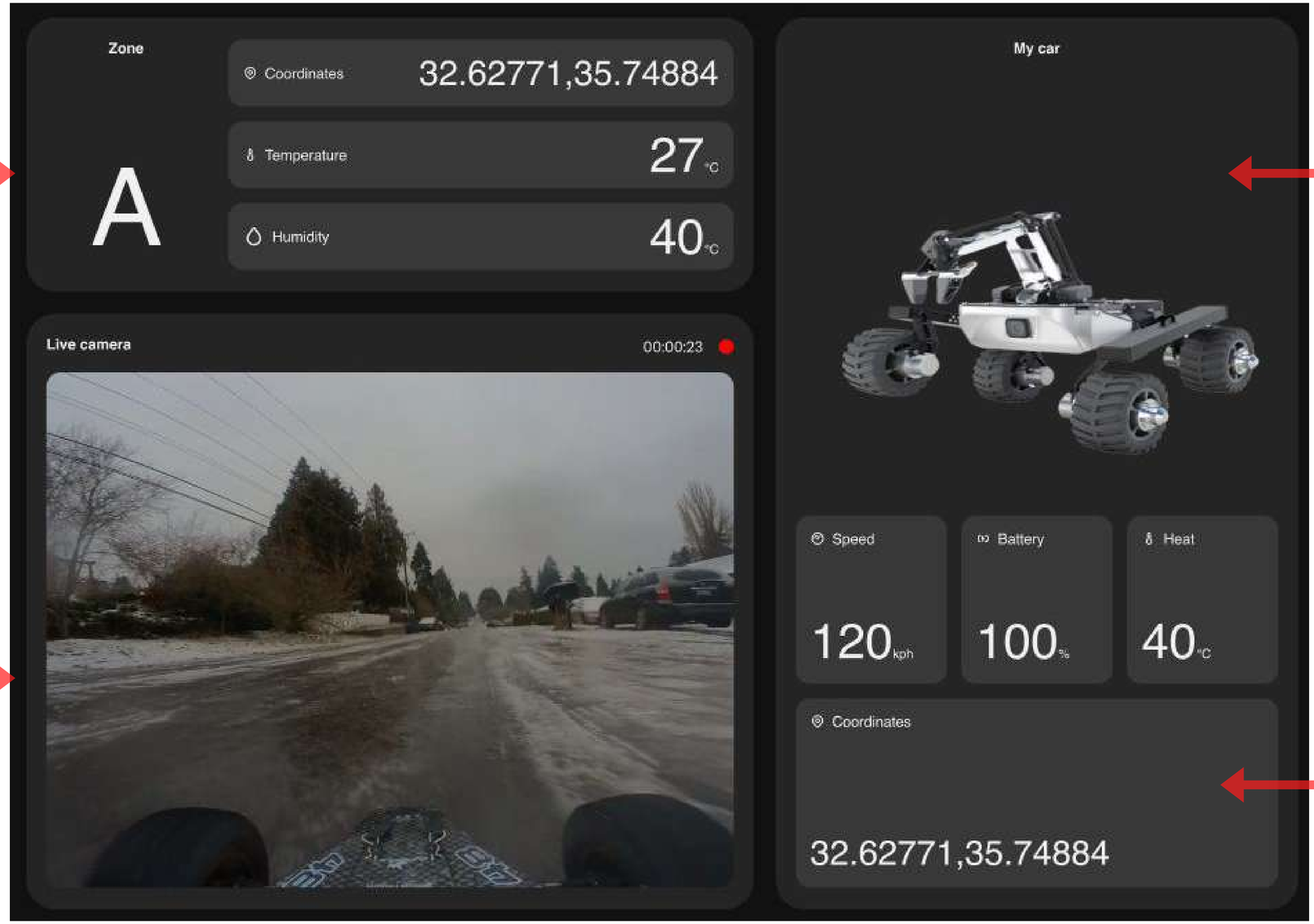
Selected Operator

Operator Live Data

User Interface v1 Dark

Current Sensor node Live data

Cam Live feed



Selected Operator

Operator Live Data

Future Development and Global Vision

Protocols and Security Considerations

Features:

- Add SOS feature for distress signals between operators and sensor nodes too.
- Register historical data and logs from all components.
- Migrate from localhost to cloud services.
- Develop the user interface.
- Integrate obstacle detection using sensors and CV/ML.
- Implement autonomous navigation algorithms (Pathing, Auto nivagation, save Path ...).
- Enhance security measures.
- Extend control features and improve user interface on the control server. (Experiment with solution Radio + Wifi)

Thanks for your attention