Smart Traffic Management System

CS1005 – Internet of Things Laboratory

&

CMP515 - Internet of Things

Faculty Name: Prof. Gaurav Kumar Gaharwar

Mr. Yash V Jhaveri

SCHOOL OF ENGINEERING & TECHNOLOGY

BACHELOR OF TECHNOLOGY

5th SEMESTER

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Documented By:

Pathan Mohammed Sarhankhan (23000008)

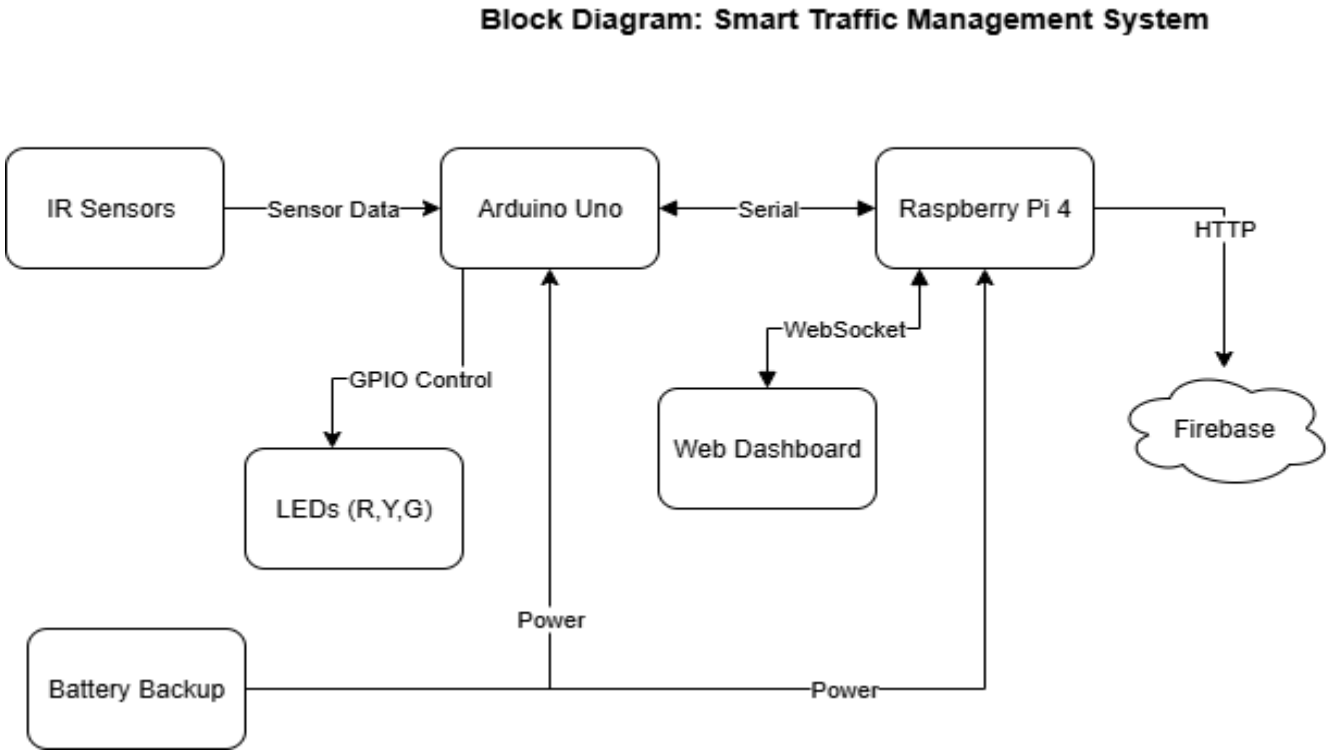
Amin Preet (23000771)

Chauhan Harsh (23000929)

System Design

1. Block Diagram

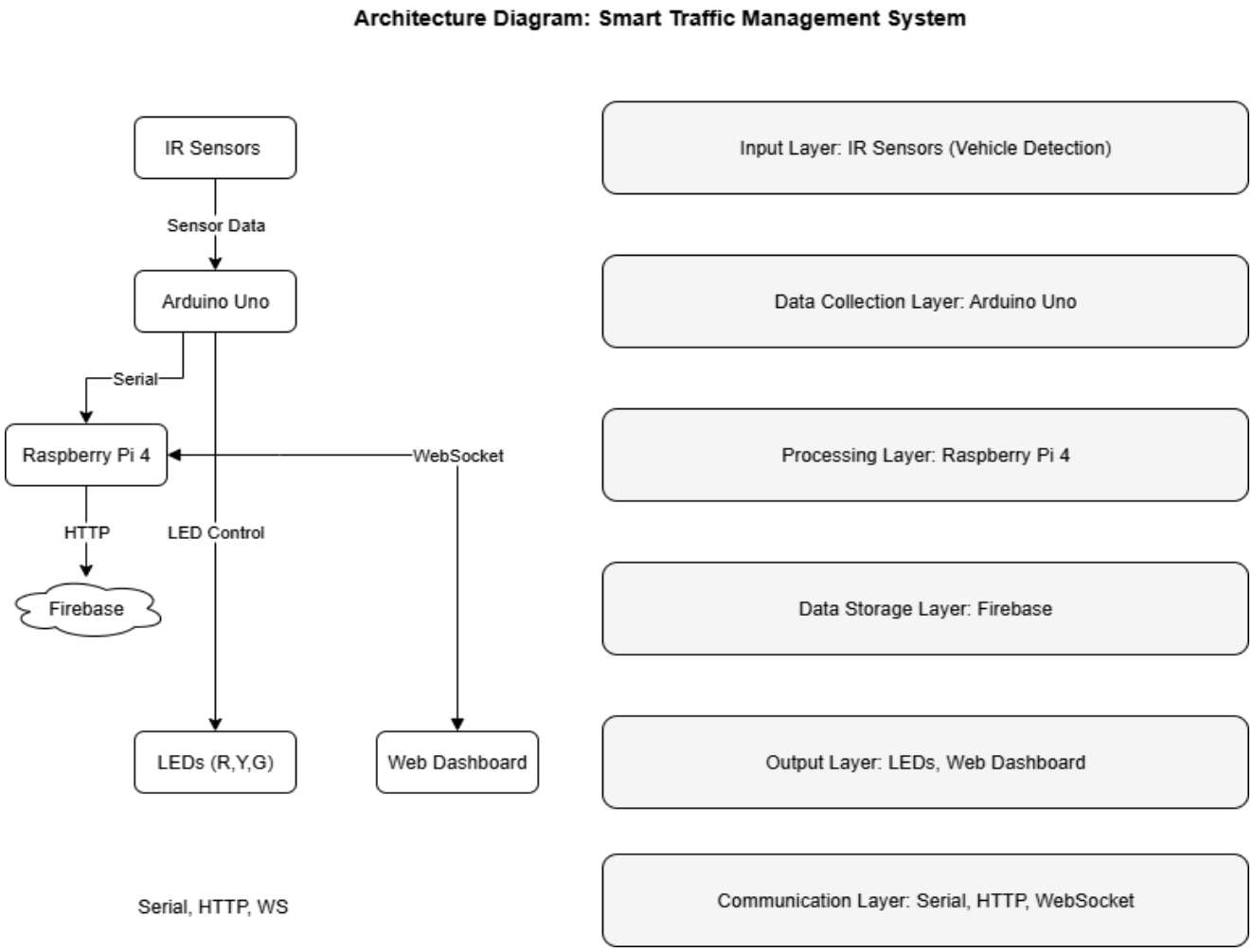
The block diagram shows the high-level components and their interactions.



- **Input**: 12 IR sensors (3 per lane) detect vehicles.
- **Processing**: Arduino Uno processes sensor data; Raspberry Pi 4 handles traffic logic and cloud communication.
- **Output**: 12 LEDs (R, Y, G per direction) simulate traffic lights; web dashboard provides monitoring/control.
- **Communication**: Serial between Arduino and Raspberry Pi; HTTP/WebSocket for cloud/dashboard.
- **Power**: Battery backup for reliability.

2. Architecture Diagram

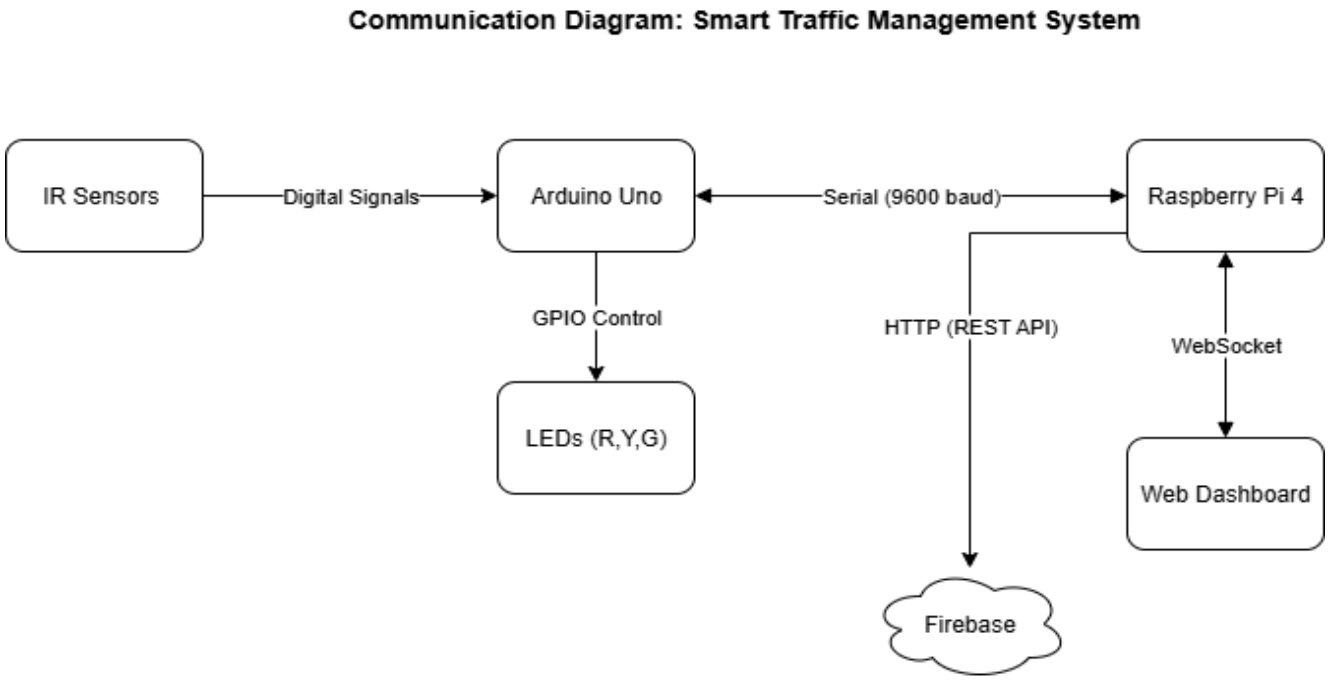
The system architecture is layered as follows:



- **Input Layer**: 12 IR sensors detect vehicle presence and density in four lanes.
- **Data Collection Layer**: Arduino Uno reads sensor data.
- **Processing Layer**: Arduino Uno controls LEDs via shift registers; Raspberry Pi 4 manages logic and cloud.
- **Data Storage Layer**: Firebase stores traffic logs.
- **Output Layer**: 12 LEDs indicate traffic states; React.js dashboard for monitoring/control.
- **Communication Layer**: Serial, HTTP, and WebSocket protocols.

3. Communication Diagram

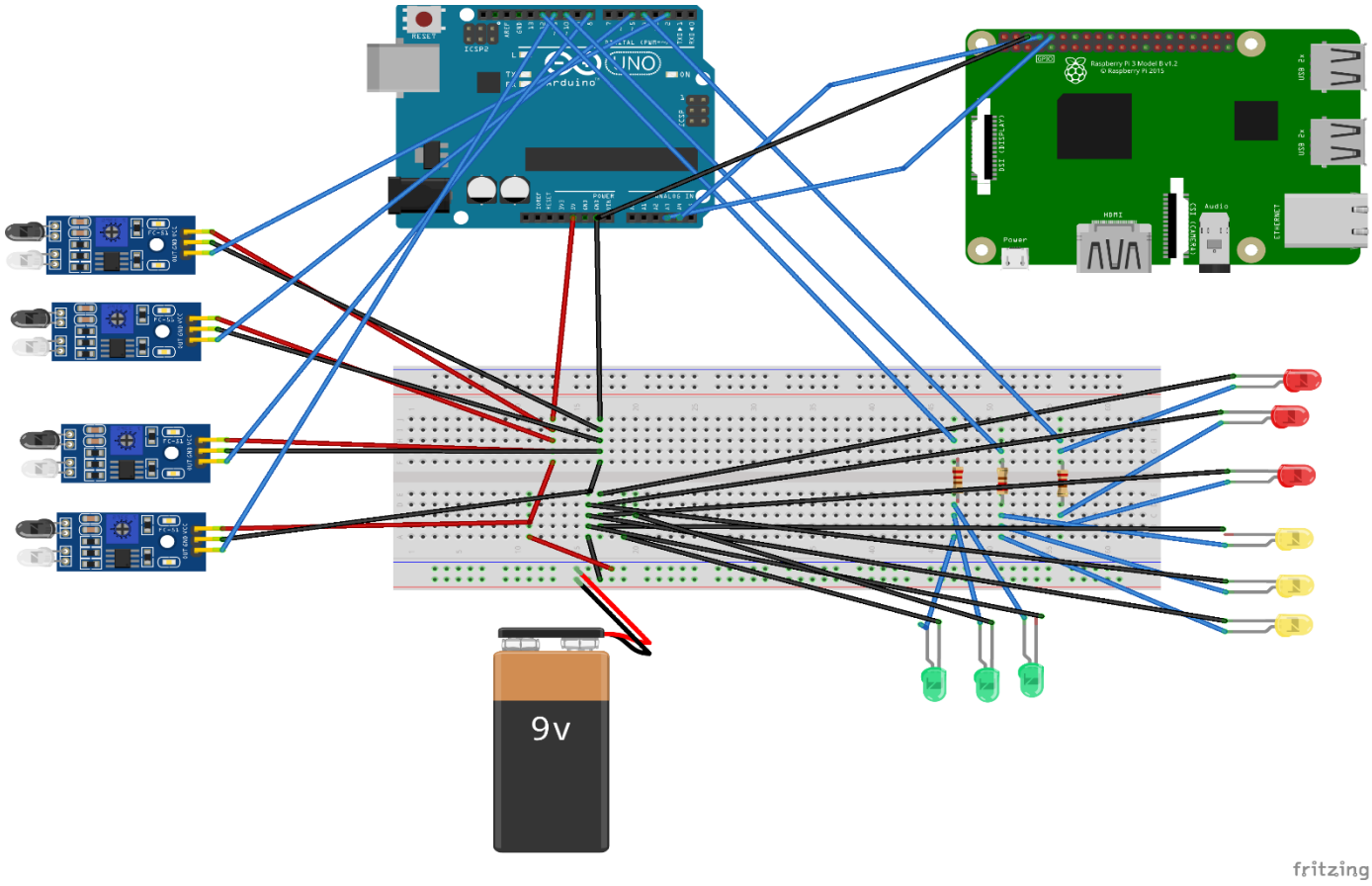
The communication flow is depicted below:



- **Devices**: 12 IR sensors, Arduino Uno, Raspberry Pi 4, 12 LEDs (via shift registers), Firebase, web dashboard.
- **Protocol**: Serial (UART) between Arduino and Raspberry Pi; HTTP for Firebase; WebSocket for dashboard.
- **Flow**: Sensors send data to Arduino, which controls LEDs and sends density logs to Raspberry Pi. Raspberry Pi logs to Firebase and communicates with the dashboard.

4. Circuit Diagram

The circuit connections are shown below:



- **Arduino Uno**: Connects to 12 IR sensors (D2–D13) and shift registers for LEDs (A0–A2).
- **IR Sensors**: 12 TCRT5000 modules (3 per lane), VCC to 5V, GND to GND, OUT to D2–D13.
- **LEDs**: 12 LEDs (4 red, 4 yellow, 4 green) with 220Ω resistors, connected to two chained 74HC595 shift registers.
- **Shift Registers**: Data to A0 (14), Clock to A1 (15), Latch to A2 (16); outputs to LEDs.
- **Raspberry Pi 4**: Connected to Arduino via SoftwareSerial (TX A3/17, RX A4/18) for serial communication.
- **Power**: 5V USB for Arduino/Raspberry Pi; 9V battery backup.

5. Sensors/Actuators/Processing Elements

1. **IR Sensors (TCRT5000) x12 (3 per lane)**
    - **Specifications**: Reflective optical sensor, 3.3–5V, 940nm wavelength, 20mA.
    - **Data Range**: Digital output (HIGH: no vehicle, LOW: vehicle present).
    - **Constraints**: Detection range ~1–25mm; sensitive to ambient light.
    - **Role**: Detects vehicles in each of the four lanes; 3 sensors per lane measure density (e.g., 0-3 vehicles).
2. **LEDs (Red x4, Yellow x4, Green x4)**
    - **Specifications**: 3–5V, 20mA, 220Ω resistors.
    - **Data Range**: Binary ON/OFF via shift register outputs.
    - **Constraints**: Limited visibility in sunlight; requires resistors.
    - **Role**: Simulates traffic lights for each of the four directions at the intersection.
3. **Arduino Uno**
    - **Specifications**: ATmega328P, 5V, 16MHz, 14 digital pins + 6 analog (used as digital), 32KB flash.
    - **Data Range**: Reads 12 sensor inputs, controls shift registers for LEDs, sends serial data.
    - **Constraints**: Limited pins (using analog as digital and SoftwareSerial); no built-in Wi-Fi.
    - **Role**: Processes sensor data, calculates lane density, controls traffic lights via shift registers.
4. **Raspberry Pi 4**
    - **Specifications**: 8GB RAM, Quad-core Cortex-A72, GPIO, Wi-Fi, 5V.
    - **Data Range**: Receives serial data, sends HTTP requests, hosts dashboard.
    - **Constraints**: Requires cooling; higher power consumption.

- **Role**: Manages cloud connectivity, data logging, and web interface.

6. **Data Storage**

- **Format**: JSON objects in Firebase Realtime Database.
  - Example: { "lane": "Lane1", "density": 2, "timestamp": "2025-08-24T18:51:00Z", "light_state": "Green" }
- **APIs for Data Exchange**:
  - **Firebase REST API**: Raspberry Pi sends HTTP POST requests for logging.
  - **Serial Communication**: Arduino sends data to Raspberry Pi via SoftwareSerial UART (9600 baud).
  - **React.js Dashboard**: Uses Firebase SDK for data retrieval; WebSocket for control.