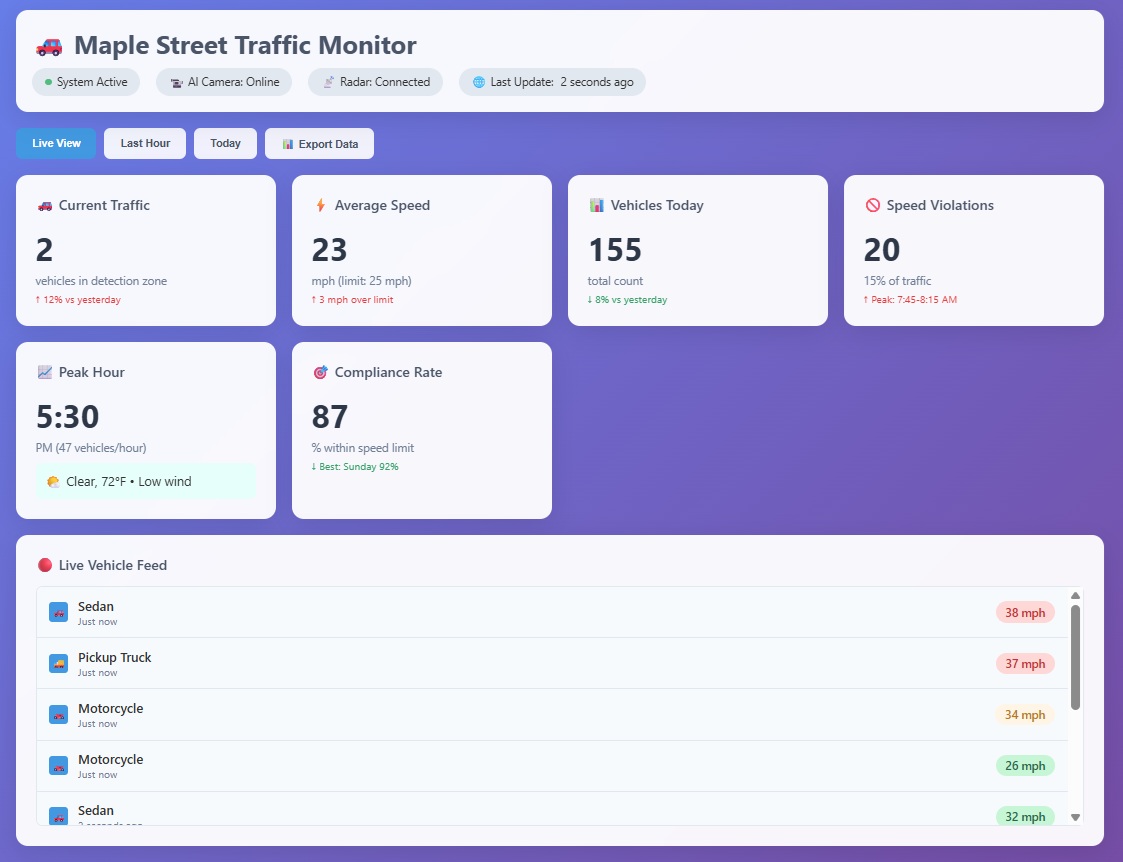
Table of Contents

# User Guide

## Cover Page

**Raspberry Pi 5 Edge ML Traffic Monitoring System**

**User Guide**



System Logo

**Version 1.2**

## Title Page

**Raspberry Pi 5 Edge ML Traffic Monitoring System**

**User Guide**

**Document Version:** 1.2  
**Publication Date:** December 11, 2025  
**Project:** Raspberry Pi 5 Edge ML Traffic Monitoring System  
**Authors:** Documentation Team

## Copyright Page

**Copyright Notice**

© 2025 gcu-merk. All rights reserved.

This document and the software it describes are protected by copyright laws and international treaties. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means without the prior written permission of the copyright holder.

**Trademarks**

* Raspberry Pi is a trademark of Raspberry Pi Ltd.
* Sony IMX500 is a trademark of Sony Corporation
* All other trademarks are the property of their respective owners.

**Disclaimer**

The information in this document is provided “as is” without warranty of any kind. While every effort has been made to ensure accuracy, the authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

**License**

This project is licensed under the terms specified in the LICENSE file in the project repository.

## Preface

### Purpose

This User Guide provides comprehensive instructions for installing, configuring, operating, and maintaining the Raspberry Pi 5 Edge ML Traffic Monitoring System. It is designed to serve both technical operators and non-technical users who need to interact with the system.

**What You Will Learn:**

* How to set up and configure your traffic monitoring system
* How to access and navigate the web dashboard
* How to interpret vehicle detection data and analytics
* How to troubleshoot common problems
* How to maintain your system for optimal performance

### Audience

This guide is intended for:

* **System Operators:** Personnel responsible for installing and maintaining the hardware and software
* **Traffic Analysts:** Users who monitor traffic data and generate reports
* **System Administrators:** IT professionals managing system configuration and cloud integration
* **End Users:** Anyone who needs to access traffic monitoring data through the dashboard
* **Decision Makers:** Managers and officials who review traffic data for planning purposes

**No Advanced Technical Knowledge Required:** This guide is written in plain language with step-by-step instructions that anyone can follow.

### Document Organization

This guide is organized into the following major sections:

1. **General Information:** Overview of what the system does and who can use it
2. **System Summary:** Description of the hardware and software components (simplified)
3. **Getting Started:** How to set up your system for the first time
4. **Using the System:** How to use the dashboard and view traffic data
5. **Troubleshooting:** Solutions to common problems
6. **FAQ:** Answers to frequently asked questions
7. **Help and Contact Details:** How to get additional support
8. **Glossary:** Simple explanations of technical terms

### Prerequisites

**For All Users:** - Ability to use a web browser (Chrome, Firefox, Safari, Edge) - Basic understanding of traffic monitoring concepts (vehicle types, speed limits)

**For Technical Operators Only:** - Familiarity with computer setup and installation procedures - Basic networking knowledge (how to find IP addresses) - Comfort following step-by-step instructions

**What You Don’t Need:** - Programming or coding experience - Advanced computer science knowledge - Experience with Linux or command-line interfaces (unless performing installation)

### Conventions Used in This Guide

**Text Formatting:** - **Bold text** indicates buttons, menu items, or important terms you’ll see on screen - Monospace text indicates code, web addresses, or command-line text - *Italic text* indicates emphasis or new concepts

**Icons and Symbols:** - ⚠️ **WARNING:** Critical information about safety or data loss - ⚡ **CAUTION:** Important information that could affect system operation - 💡 **TIP:** Helpful suggestions to make your work easier - ℹ️ **NOTE:** Additional information or context - ✅ **SUCCESS:** Indication that a step completed correctly - ❌ **ERROR:** Common error message or problem

**Step-by-Step Instructions:** Steps are numbered and include clear actions:

1. **First Action** - Description of what to do
2. **Second Action** - Description of what to do
3. **Third Action** - Description of what to do

**Visual Aids:** - Screenshots show you exactly what to expect on your screen - Diagrams illustrate hardware connections and system architecture - Tables organize information for easy reference - Highlighted areas in images draw attention to important elements

### Accessibility Features

This guide has been designed with accessibility in mind:

**For Users with Low Vision:** - High-contrast text and images - Large, readable fonts (minimum 12pt) - All images include descriptive alt text - Dashboard supports browser zoom (up to 200%) - Screen reader compatible

**For Color-Blind Users:** - Information is not conveyed by color alone - Text labels accompany all color-coded indicators - Patterns and symbols supplement color coding - Dashboard uses color-blind friendly palettes

**For Keyboard-Only Users:** - All dashboard functions accessible via keyboard - Tab navigation supported throughout interface - Keyboard shortcuts documented for common actions

**For Screen Reader Users:** - Proper heading hierarchy for navigation - Descriptive link text (no “click here” links) - Form labels and ARIA attributes included - Tables properly marked up with headers

## Table of Contents

1. [General Information](#Xbaae62f3734f3c051066aa787ac4ee950e5748e)
   * 1.1 [System Features Overview](#Xa09162a9c4c1a80044da12650d76659bb796726)
   * 1.2 [Key Capabilities](#X81bc5fd903e47b6455143bd9b1d7e069a5c6d7e)
   * 1.3 [User Roles](#X680e15bfdcf9aa7e97657b6b7056e5d550898a1)
   * 1.4 [Accessibility & Safety](#X5c9ca8c593e2ee065007c1b27e86cd8fce8e4d4)
2. [System Summary](#X48fb179513b687ef5e343b9e9f94683a0646a6c)
   * 2.1 [Hardware Components](#X006cf43e0343db86308521fbbbd7641b2989ce6)
   * 2.2 [Software Architecture](#Xc18e3df6adf55a7eb5a512d5145541bb4afc23e)
   * 2.3 [System Requirements](#X4eb0dd98b431ab4cc68d4290506bb1aa5a0f30c)
3. [Getting Started](#X94af98a6b7f1e968079e201c06d244c1660524f)
   * 3.1 [Hardware Setup Quick Reference](#Xd3e3b2d1a4754040b521023d1688b21c083f775)
   * 3.2 [Initial Configuration](#Xf2bbeab102b4fccfc33849db6aed40bd6c365ca)
   * 3.3 [Accessing the Dashboard](#X9f7c4419d1dfcf96024a29e2549518bea82dfc3)
4. [Using the System](#Xe165e4a0c2dc4bbcb095dc5507c8652e3fe897e)
   * 4.1 [Web Dashboard](#X44ce56dbac3918736ff29b53a055c96a90ee170)
   * 4.2 [Monitoring Live Traffic](#X70126e67fca83f1d941abd7edadace5bea52d06)
   * 4.3 [Viewing Analytics](#X224c503f86f7dc481ca48e80b3704037b560387)
   * 4.4 [Radar System Monitoring](#X84eebe9f17c646bfb4a54c792410022e0493d93)
   * 4.5 [Mobile App (Optional)](#X23ca44d86ab9e35ee19451c4f130edfa1deb5bb)
   * 4.6 [Common Use Cases](#X37dcdb88f909bc88d38a67588cab5b1daed0e1e)
5. [Troubleshooting](#Xd5b3d8a68c73b756968aad1dd84151376a19c17)
   * 5.1 [Common Issues](#Xf1446222800d252560f89b191a6b6405d06c3f6)
   * 5.2 [Radar Detection Problems](#X4bf4252e063d1ae8a320ea57696e026324652df)
   * 5.3 [Dashboard Issues](#X30aea9eebb5c96c2bee9003d5ce5e448ea15c3c)
   * 5.4 [Network Connectivity](#Xa8f7057cc9fc435ad32912d4f27db2b44e532cc)
   * 5.5 [Performance Monitoring](#Xb908bc06532bc08b01e0b3b7c4d0035b67ca8af)
6. [FAQ](#Xa007fe61f704f9207cb1e3e534f26b44fa98ca5)
   * 6.1 [General Questions](#Xe2d51e91e47e420fca3e23221c062a6196c321e)
   * 6.2 [Technical Questions](#X593612db0d3d51d001f98a643ea4cf0f01e8484)
   * 6.3 [Troubleshooting Questions](#Xbf0afb56dfc76126c22487d506187b431c10ddb)
7. [Help and Contact Details](#X4b9433780e105da9d389633cce35c0356918be5)
   * 7.1 [Support Channels](#X9569bb0611cee461ffa052f521720cd3b1b94e4)
   * 7.2 [Additional Resources](#Xc18dfdd70214c7062a2985418b8a844a18ddf5a)
   * 7.3 [Reporting Issues](#Xafa35ebaa2d7abb0dc6c6ebfc0bd07db926d0d5)
8. [Glossary](#X7aa0aa0821f6383922063c6c8b30c8f600e1bac)

**Related Documentation:**

* [Technical Design Document](./Technical_Design.md)
* [Implementation & Deployment Guide](./Implementation_Deployment.md)
* [Project Management Summary](./Project_Management.md)
* [References & Appendices](./References_Appendices.md)

## 1. General Information

### 1.1 System Features Overview

The Raspberry Pi 5 Edge ML Traffic Monitoring System provides real-time vehicle detection, classification, speed measurement, and traffic analytics at the edge. The system leverages radar-triggered edge AI processing using the Sony IMX500 camera’s built-in neural processing unit for on-camera vehicle classification. This advanced approach provides intelligent, power-efficient traffic monitoring with minimal latency and enhanced accuracy.

**For technical details on the ML/AI workflow and component status, see the** [**ML/AI Workflow and Component Status**](./Technical_Design.md#31-mlai-workflow-and-component-status) **section in the Technical Design Document.**

### 1.2 Key Capabilities

The system offers the following key features:

* **Radar-Triggered Edge AI:** Combines OPS243-C radar detection with IMX500 on-camera AI processing
* **Real-time Vehicle Classification:** 85-95% accuracy for vehicle type identification
* **Ultra-Low Latency:** <350ms from detection to classification
* **Power Efficient:** 4-6W average power consumption
* **Multi-Sensor Fusion:** Correlates radar speed/direction with AI vehicle identification
* **Privacy-First:** All AI processing happens locally on the camera sensor
* **Weather Independent:** Radar detection works in all weather conditions
* **Local & Cloud Dashboards:** Edge UI for live monitoring and Cloud UI for historical analytics
* **REST and WebSocket APIs:** For integration and automation with external systems
* **Data Export & Reporting:** Generate reports and export data for further analysis

### 1.3 User Roles

The system supports different user roles with varying levels of access:

* **Operator:** Installs, configures, and maintains the system on-site. Responsible for hardware setup and basic troubleshooting.
* **Traffic Analyst:** Reviews real-time and historical data, generates reports, and identifies traffic patterns and trends.
* **Administrator:** Manages user access, system configuration, cloud integration, and performs system maintenance tasks.
* **Viewer:** Read-only access to dashboard and reports for monitoring purposes.

### 1.4 Accessibility & Safety

#### Accessibility Features

This system and documentation have been designed to be accessible to all users:

**For Users with Low Vision:**

* High-contrast text and images throughout the documentation and dashboard
* Large, readable fonts (minimum 12pt in documentation, 14px in dashboard)
* All images include descriptive alt text for screen readers
* Dashboard supports browser zoom up to 200% without breaking layout
* Adjustable font sizes in dashboard settings

**For Color-Blind Users:**

* Information is never conveyed by color alone
* Text labels accompany all color-coded indicators
* Patterns and symbols supplement color coding
* Dashboard uses color-blind friendly palettes (tested with deuteranopia and protanopia simulators)
* Status indicators use shapes in addition to colors (circles, squares, triangles)

**For Keyboard-Only Users:**

* All dashboard functions accessible via keyboard navigation
* Tab key moves through all interactive elements in logical order
* Enter and Space keys activate buttons and links
* Escape key closes menus and dialogs
* Keyboard shortcuts available for common actions (press **?** in dashboard to view)

**For Screen Reader Users:**

* Proper heading hierarchy for easy navigation
* Descriptive link text (no “click here” or “read more” links)
* Form labels and ARIA attributes properly implemented
* Tables properly marked up with headers and captions
* Real-time updates announced to screen readers

#### Safety Information

⚠️ **IMPORTANT SAFETY WARNINGS**

Please read and follow these safety guidelines to prevent injury and equipment damage:

**Electrical Safety:**

1. **Use Only Approved Power Supplies**
   * Use only the official Raspberry Pi 5.1V 5A power supply
   * Or use a certified PoE+ (Power over Ethernet) HAT
   * ⚡ **DANGER:** Using incorrect power supplies can cause fire or electric shock
2. **Keep Electronics Dry**
   * Never expose the Raspberry Pi or camera to water or moisture
   * Use only IP65-rated weatherproof enclosures for outdoor installations
   * Ensure all cable entry points are properly sealed
   * ⚡ **DANGER:** Water can cause electric shock and permanent equipment damage
3. **Proper Grounding**
   * Ensure all equipment is properly grounded according to local electrical codes
   * Use surge protectors for outdoor installations
   * Install lightning protection if required by local codes

**Installation Safety:**

1. **Power Off Before Installation**
   * Always disconnect power before connecting or disconnecting components
   * Wait 30 seconds after disconnecting power before touching components
   * ⚡ **DANGER:** Components may remain energized even after power is disconnected
2. **Work at Safe Heights**
   * Use proper ladders or lifts when installing equipment above ground level
   * Wear appropriate personal protective equipment (hard hat, gloves, safety glasses)
   * Have a second person assist with elevated installations
   * ⚡ **DANGER:** Falls from height can cause serious injury or death
3. **Traffic Safety**
   * If installing near roadways, follow all traffic safety regulations
   * Use proper signage, cones, and barriers to protect work area
   * Wear high-visibility clothing
   * Follow local traffic control requirements
   * ⚡ **DANGER:** Working near traffic can result in serious injury or death

**Radar Safety:**

1. **Avoid Prolonged Exposure**
   * The OPS243-C radar operates at 24.125 GHz
   * While low-power, avoid placing your head or body directly in front of radar beam for extended periods
   * Maintain at least 1 foot (30 cm) distance from radar sensor during operation
   * ℹ️ **NOTE:** Radar power level is very low and complies with FCC regulations
2. **Medical Device Interference**
   * Consult your doctor if you have a pacemaker or other implanted medical device
   * Maintain distance from radar during operation if you have medical concerns

**Data Privacy and Legal Compliance:**

1. **Respect Privacy Laws**
   * Ensure your installation complies with local privacy and surveillance laws
   * Post appropriate signage if required by local regulations
   * Do not capture images or video of private property without permission
   * Configure data retention policies according to legal requirements
2. **Secure Your Data**
   * Change default passwords immediately after installation
   * Use strong, unique passwords for all accounts
   * Enable HTTPS/TLS encryption for remote access
   * Keep software updated with security patches
3. **Obtain Required Permits**
   * Check with local authorities for required permits before installation
   * Comply with all zoning and installation regulations
   * Obtain permission before installing on property you don’t own

**Environmental Considerations:**

1. **Operating Temperature**
   * Operate only within temperature range: 0°C to 50°C (32°F to 122°F)
   * Use outdoor-rated enclosures with proper ventilation
   * Avoid direct sunlight on enclosure (use shade or reflective coating)
2. **Storage Temperature**
   * Store equipment at -20°C to 70°C (-4°F to 158°F)
   * Protect from moisture, dust, and extreme conditions
3. **Humidity**
   * Operate only in 10% to 90% relative humidity (non-condensing)
   * Use desiccants in outdoor enclosures if necessary

**Maintenance Safety:**

1. **Regular Inspections**
   * Inspect power cables for damage monthly
   * Check weatherproof seals for deterioration
   * Verify mounting hardware is secure
   * Look for signs of overheating or damage
2. **Safe Cleaning**
   * Power off before cleaning any components
   * Use only dry or slightly damp microfiber cloth for camera lens
   * Never use solvents, alcohol, or abrasive cleaners
   * Allow equipment to dry completely before powering on

**Emergency Procedures:**

**If You Smell Burning or See Smoke:**

1. Immediately disconnect power at the source
2. Do not touch hot components
3. Use appropriate fire extinguisher if fire is present (Class C for electrical fires)
4. Evacuate area and call emergency services if fire cannot be controlled
5. Do not attempt to salvage equipment until safe to do so

**If Equipment Gets Wet:**

1. Immediately disconnect power
2. Do not attempt to power on wet equipment
3. Allow equipment to dry completely (24-48 hours)
4. Have equipment inspected by qualified technician before use
5. Replace equipment if extensive water damage occurred

**If Someone Receives Electric Shock:**

1. Do not touch the person if still in contact with power source
2. Disconnect power immediately
3. Call emergency services (911 or local emergency number)
4. Begin CPR if trained and necessary
5. Stay with person until help arrives

**Emergency Contact Information:**

* **Emergency Services:** 911 (or local emergency number)
* **Electrical Safety:** Contact licensed electrician
* **Product Support:** See [Help and Contact Details](#X4b9433780e105da9d389633cce35c0356918be5) section
* **Poison Control:** 1-800-222-1222 (US)

**Regulatory Compliance:**

This equipment must be installed and operated in compliance with:

* FCC Part 15 (Radio Frequency Devices) - United States
* CE Marking requirements - European Union
* Local electrical codes and standards
* Privacy and surveillance regulations
* Traffic monitoring and enforcement laws

⚠️ **Failure to comply with these safety requirements may result in injury, death, equipment damage, legal liability, or regulatory violations.**

💡 **When in Doubt:** If you are unsure about any aspect of installation or operation, consult a qualified professional. Safety is always the top priority.

## 2. System Summary

### 2.1 Hardware Components

The system consists of the following hardware components:

#### Core Components

| Component | Model/Specification | Purpose |
| --- | --- | --- |
| **Single Board Computer** | Raspberry Pi 5 (8GB RAM) | Main processing unit |
| **AI Camera** | Sony IMX500 (12MP) | On-chip AI vehicle classification |
| **Radar Sensor** | OPS243-C FMCW Radar | Speed and direction detection (200m range) |
| **External Storage** | Samsung T7 Shield 2TB SSD | Local data storage via USB 3.2 |
| **Power Supply** | Official 5.1V 5A PSU or PoE+ HAT | System power |

#### Connectivity

* **CSI-2 Ribbon Cable:** Connects IMX500 camera to Raspberry Pi
* **GPIO Pins:** Radar sensor connections (UART + GPIO)
* **Ethernet/WiFi:** Network connectivity
* **USB 3.2:** External SSD connection

#### Optional Components

* **Weatherproof Enclosure:** IP65-rated for outdoor installation
* **Mounting Hardware:** Brackets and poles for positioning
* **UPS (Uninterruptible Power Supply):** Backup power during outages

### 2.2 Software Architecture

The system uses a containerized microservices architecture with the following key components:

#### Services

The system uses a containerized microservices architecture with Docker Compose:

* **Redis:** In-memory data store for real-time event streaming and caching
* **Traffic Monitor (API Gateway):** RESTful API and health monitoring endpoints (port 5000)
* **Radar Service:** Interfaces with OPS243-C radar sensor via UART (/dev/ttyAMA0)
* **Data Maintenance:** Automated cleanup of old data and storage management
* **Airport Weather:** Fetches weather data from nearby airports via API
* **DHT22 Weather:** Reads local temperature and humidity from DHT22 sensor (GPIO pin 4)
* **IMX500 Camera Service:** Runs as host service (not containerized) for direct hardware access

ℹ️ **NOTE:** The IMX500 AI camera service runs directly on the host (Raspberry Pi) rather than in a Docker container because it requires direct access to the camera hardware (/dev/video\* devices). This is a Raspberry Pi 5-specific requirement.

#### Technology Stack

* **Operating System:** Raspberry Pi OS (64-bit)
* **Containerization:** Docker and Docker Compose
* **Programming Languages:** Python 3.x, JavaScript
* **Database:** SQLite, Redis
* **Web Framework:** Flask (Python), HTML5/CSS3/JavaScript
* **APIs:** REST, WebSocket (Socket.IO)

### 2.3 System Requirements

#### Minimum Hardware Requirements

* Raspberry Pi 5 with 4GB RAM (8GB recommended)
* 32GB microSD card for OS (64GB+ recommended)
* External USB SSD (256GB minimum, 1TB+ recommended for extended storage)
* Compatible IMX500 camera module
* OPS243-C radar sensor with cables
* 5.1V 5A power supply or PoE+ HAT

#### Network Requirements

* Ethernet or WiFi connectivity (1 Mbps minimum for cloud features)
* Static IP address or DHCP reservation (recommended)
* Open ports: 5000 (HTTP), 8080 (API), 443 (HTTPS optional)
* Optional: Tailscale VPN for secure remote access

#### Software Requirements

* Raspberry Pi OS (64-bit, Bullseye or newer)
* Docker Engine 20.x or newer
* Docker Compose v2.x or newer
* Python 3.9 or newer
* Git for version control

#### Environmental Requirements

* **Operating Temperature:** 0°C to 50°C (outdoor enclosure)
* **Storage Temperature:** -20°C to 70°C
* **Humidity:** 10% to 90% non-condensing
* **Installation:** Outdoor-rated IP65 enclosure for harsh environments

## 3. Getting Started

### 3.1 Hardware Setup Quick Reference

This section provides a quick reference for the hardware connections. For detailed installation instructions, see the [Implementation & Deployment Guide](./Implementation_Deployment.md).

#### OPS243-C Radar Sensor Pinout

| OPS243 Pin | Function | Wire Color | RPi Physical Pin | RPi GPIO | Notes |
| --- | --- | --- | --- | --- | --- |
| Pin 3 | Host Interrupt | Orange | Pin 16 | GPIO23 | Real-time detection |
| Pin 4 | Reset | Yellow | Pin 18 | GPIO24 | Software reset |
| Pin 6 | UART RxD | Green | Pin 8 | GPIO14 (TXD) | Commands to radar |
| Pin 7 | UART TxD | White | Pin 10 | GPIO15 (RXD) | Data from radar |
| Pin 9 | 5V Power | Red | Pin 4 | 5V Power | 150mA typical |
| Pin 10 | Ground | Black | Pin 6 | Ground | Common ground |
| **Pin 1** | **Low Alert** | **Blue** | **Pin 29** | **GPIO5** | **Speed/range alerts** |
| **Pin 2** | **High Alert** | **Purple** | **Pin 31** | **GPIO6** | **Speed/range alerts** |

### Key Hardware Features

* **IMX500 AI Camera**: 12MP sensor with on-chip AI processing (CSI-2 ribbon cable)
* **OPS243-C Radar**: 24.125 GHz FMCW radar with 200m range (GPIO + UART)
* **External SSD**: Samsung T7 Shield 2TB for local storage (USB 3.2)
* **Power**: 5.1V 5A official PSU or PoE+ HAT

### Connection Verification

# Test camera  
libcamera-hello --preview=none --timeout=5000  
  
# Test radar UART  
sudo cat /dev/ttyACM0  
  
# Check GPIO status  
gpio readall

For complete installation instructions, hardware assembly, and troubleshooting, see:

* [Technical Design Document - Hardware Design](./Technical_Design.md#3-hardware-design)
* [Implementation & Deployment Guide - Hardware Assembly](./Implementation_Deployment.md#22-hardware-assembly)

### 3.2 Initial Configuration

After completing the hardware setup, follow these steps for initial system configuration:

#### Step 1: Boot and Access the Raspberry Pi

1. Insert the microSD card with Raspberry Pi OS installed
2. Connect power to boot the system
3. Wait 2-3 minutes for initial boot
4. Connect via SSH or direct keyboard/monitor

# SSH connection (replace with your Pi's IP address on local network)  
ssh pi@raspberrypi.local  
  
# Or use direct IP if hostname doesn't work  
# Find IP with: hostname -I

#### Step 2: Install Required Software

# Update system packages  
sudo apt update && sudo apt upgrade -y  
  
# Install Docker  
curl -fsSL https://get.docker.com -o get-docker.sh  
sudo sh get-docker.sh  
  
# Add user to docker group  
sudo usermod -aG docker $USER  
  
# Install Docker Compose  
sudo apt install docker-compose-plugin -y  
  
# Verify installations  
docker --version  
docker compose version

#### Step 3: Clone Project Repository

# Clone the project repository  
git clone https://github.com/gcu-merk/CST\_590\_Computer\_Science\_Capstone\_Project.git  
cd CST\_590\_Computer\_Science\_Capstone\_Project

#### Step 4: Configure Environment

# Copy environment template  
cp .env.example .env  
  
# Edit configuration (use nano or your preferred editor)  
nano .env  
  
# Set required variables:  
# - RADAR\_PORT=/dev/ttyAMA0  
# - STORAGE\_PATH=/mnt/storage  
# - API\_PORT=5000

#### Step 5: Start Services

# Start all services using Docker Compose  
docker compose up -d  
  
# Verify services are running  
docker compose ps  
  
# Check service health  
docker compose logs

#### Step 6: Verify Installation

# Test camera connection  
libcamera-hello --preview=none --timeout=5000  
  
# Test radar UART  
sudo cat /dev/ttyAMA0  
  
# Check GPIO status  
gpio readall  
  
# Test API endpoint (on Pi itself)  
curl http://localhost:5000/api/health/system

💡 **Tip:** If you encounter permission errors with GPIO or serial ports, ensure your user is in the gpio, dialout, and video groups.

### 3.3 Accessing the Dashboard

Once the system is configured and running, you can access the web dashboard:

#### Public Access (Recommended)

The easiest way to access the dashboard is through GitHub Pages:

1. **Open any web browser**
2. **Navigate to the public dashboard:**

* https://gcu-merk.github.io/CST\_590\_Computer\_Science\_Capstone\_Project/

1. **The dashboard automatically connects to the API** through Tailscale VPN

💡 **Note:** The GitHub Pages dashboard is publicly accessible, but the API requires Tailscale VPN for security.

#### Local Network Access (Alternative)

If accessing from the same local network as the Raspberry Pi:

1. **Find your Raspberry Pi’s IP address:**

* hostname -I

1. **Access the traffic-monitor API directly:**

* http://[PI\_IP\_ADDRESS]:5000

#### Remote Access via Tailscale

For secure remote access to the API backend:

1. **Install Tailscale on Raspberry Pi** (if not already installed):

* curl -fsSL https://tailscale.com/install.sh | sh  
  sudo tailscale up

1. **Install Tailscale on your client device** (laptop, phone, etc.)
2. **Access the API through Tailscale:**

* https://edge-traffic-monitoring.taild46447.ts.net/api

💡 **Tip:** The GitHub Pages dashboard already uses the Tailscale API, so you only need Tailscale on your device if accessing the API directly.

## 4. Using the System

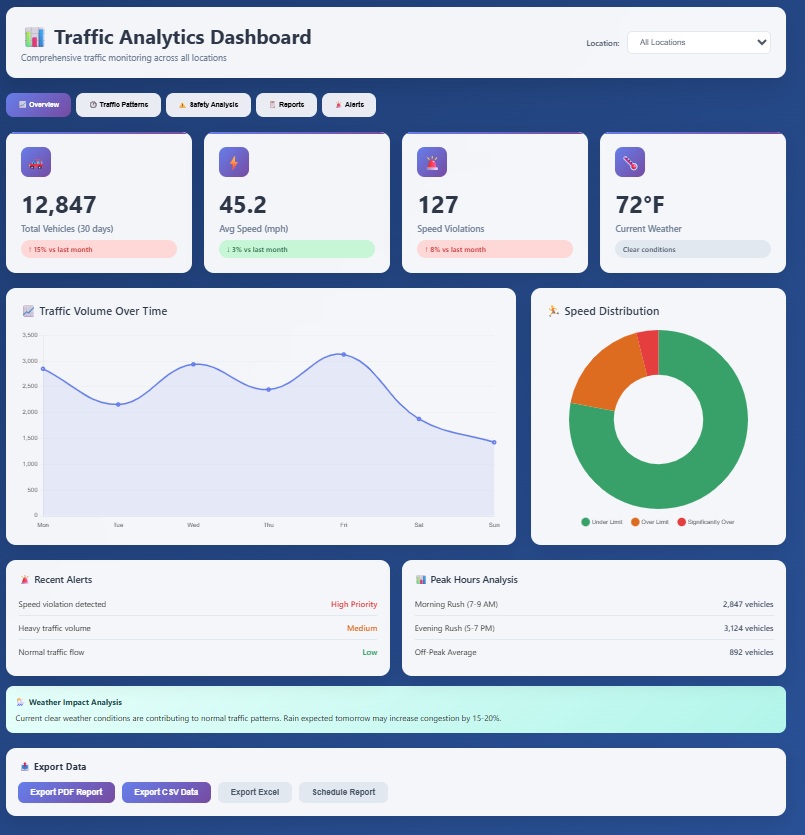
### 4.1 Web Dashboard

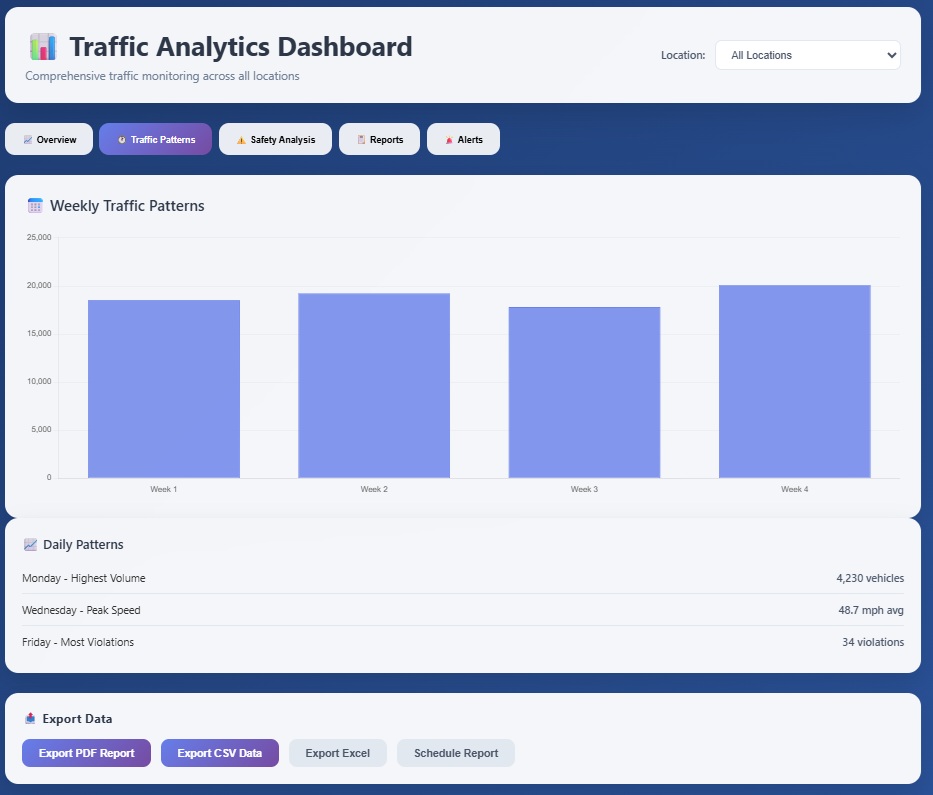
The Edge UI (local web dashboard) provides real-time monitoring, configuration, and analytics for the traffic monitoring system. The Cloud UI (optional) extends these features with historical data and remote management.

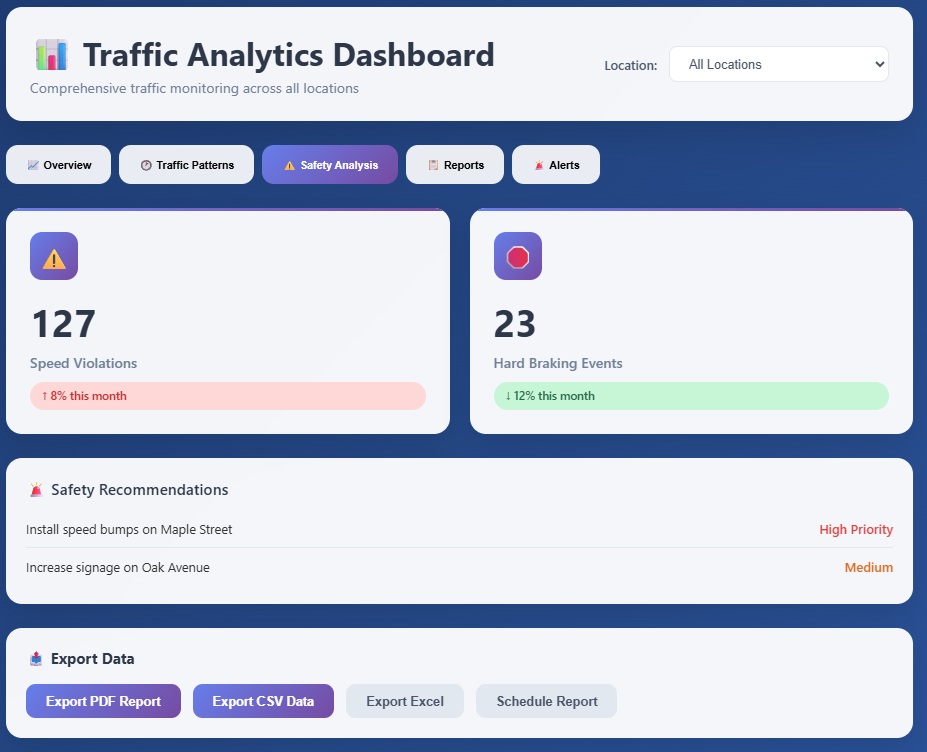
### Layout & Navigation

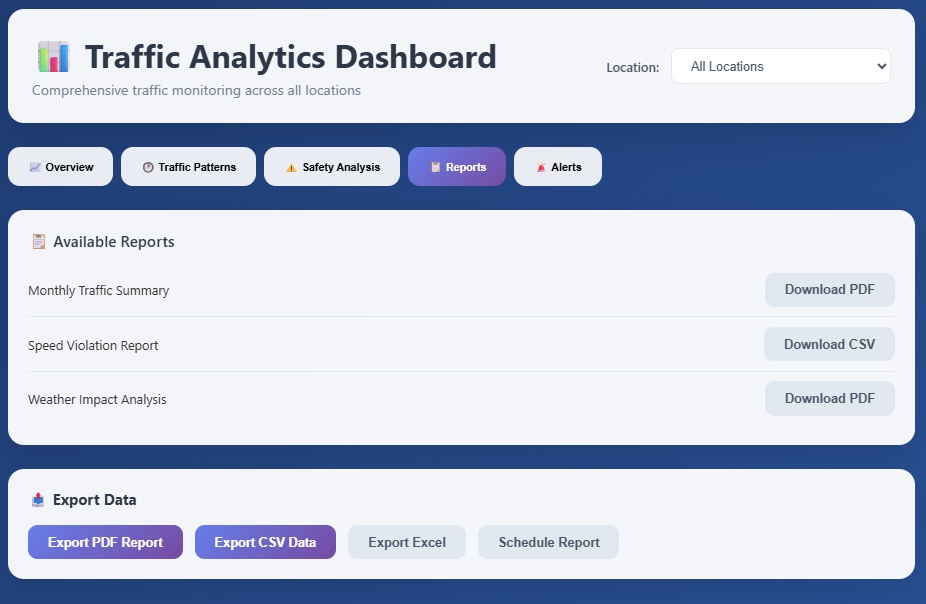
* **Header:** Displays system name, current status, and navigation links (e.g., Live View, Analytics, Settings)
* **Live View Panel:**
  + Real-time video feed from the AI camera (if enabled)
  + Overlay of detected vehicles and bounding boxes
  + Speed readings from radar sensor
* **Event Table:**
  + List of recent vehicle events (timestamp, type, speed, location)
  + Filter/search by date, type, or speed
* **Analytics Panel:**
  + Charts for vehicle counts, speed distributions, and violation rates
  + Historical trends and anomaly alerts
* **System Status:**
  + Health indicators for camera, radar, storage, and network
  + Log messages and error notifications

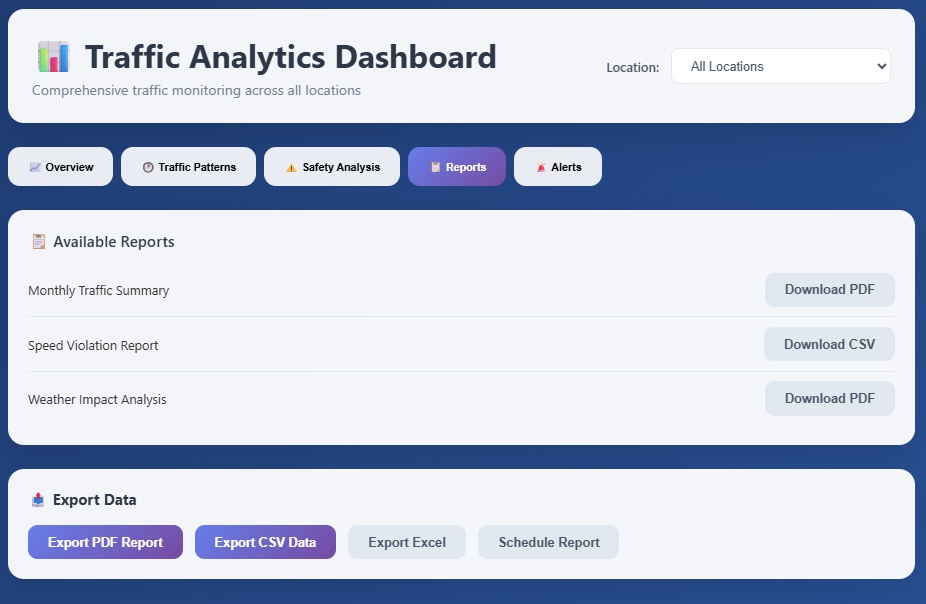
#### Dashboard Screenshots

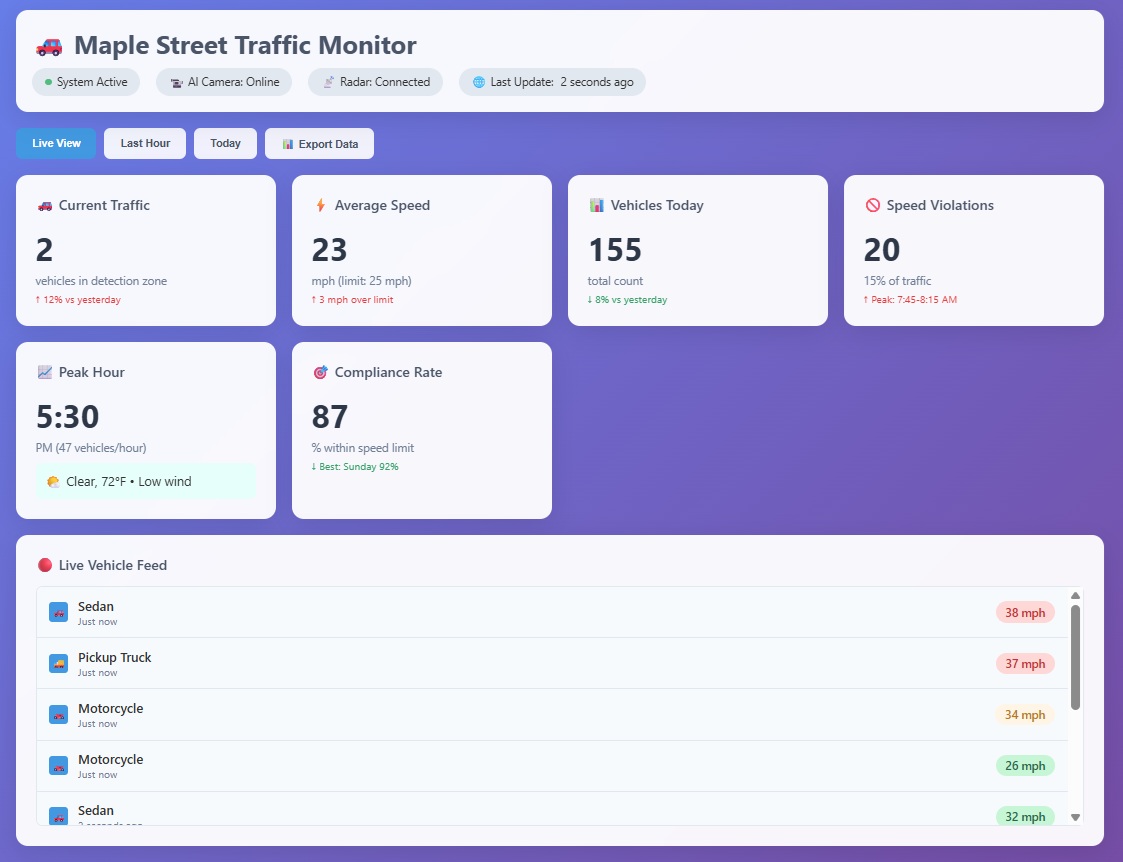
 *Figure 1: Cloud Dashboard - Main Analytics View*

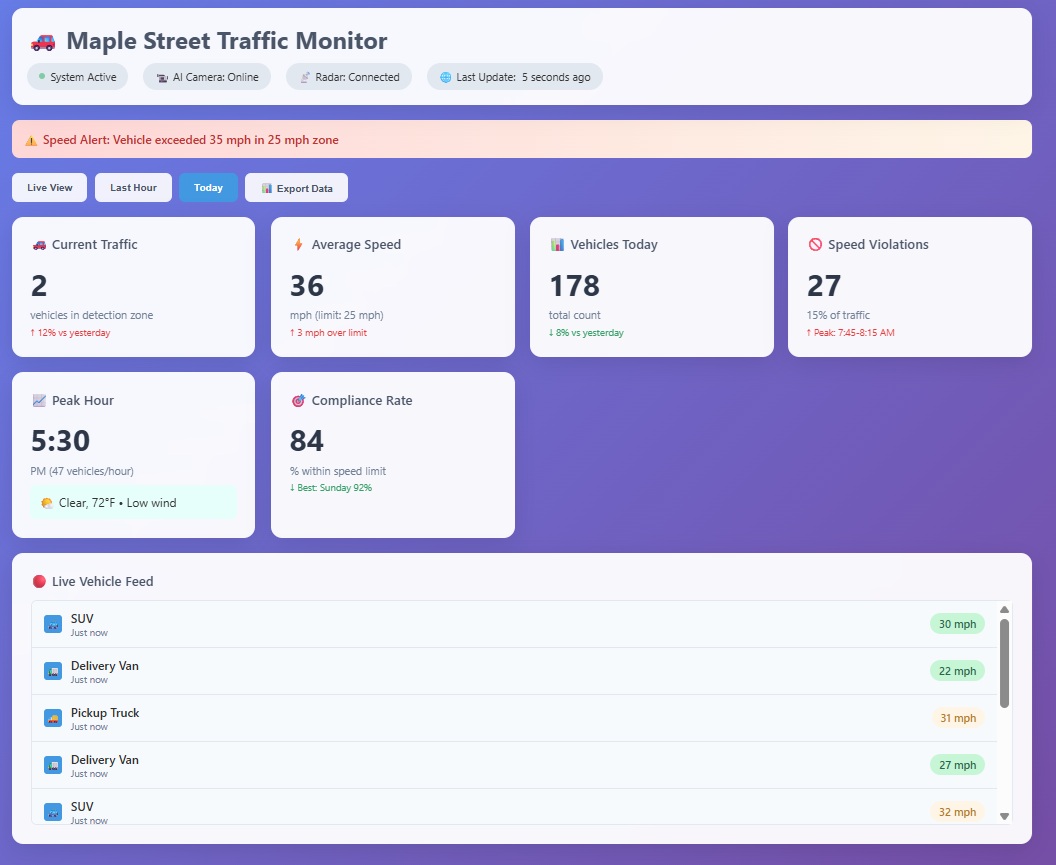
 *Figure 2: Cloud Dashboard - Event Table*

 *Figure 3: Cloud Dashboard - System Status*

 *Figure 4: Cloud Dashboard - Settings Panel*

 *Figure 5: Cloud Dashboard - Historical Trends*

 *Figure 6: Local Edge UI - Live View*

 *Figure 7: Local Edge UI - Analytics Panel*

## 4. Using the System

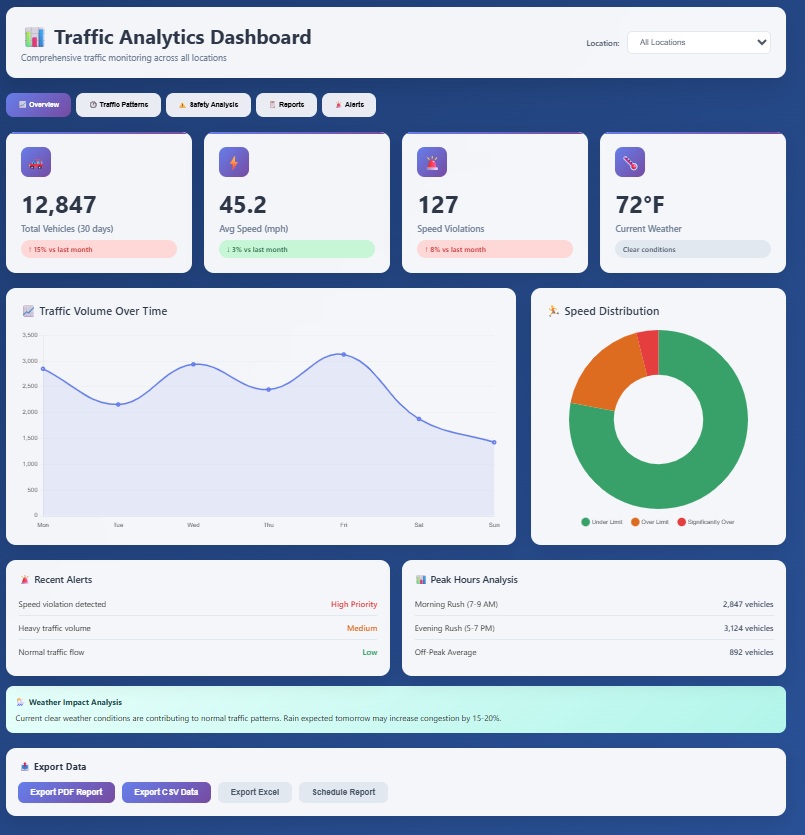
### 4.1 Web Dashboard

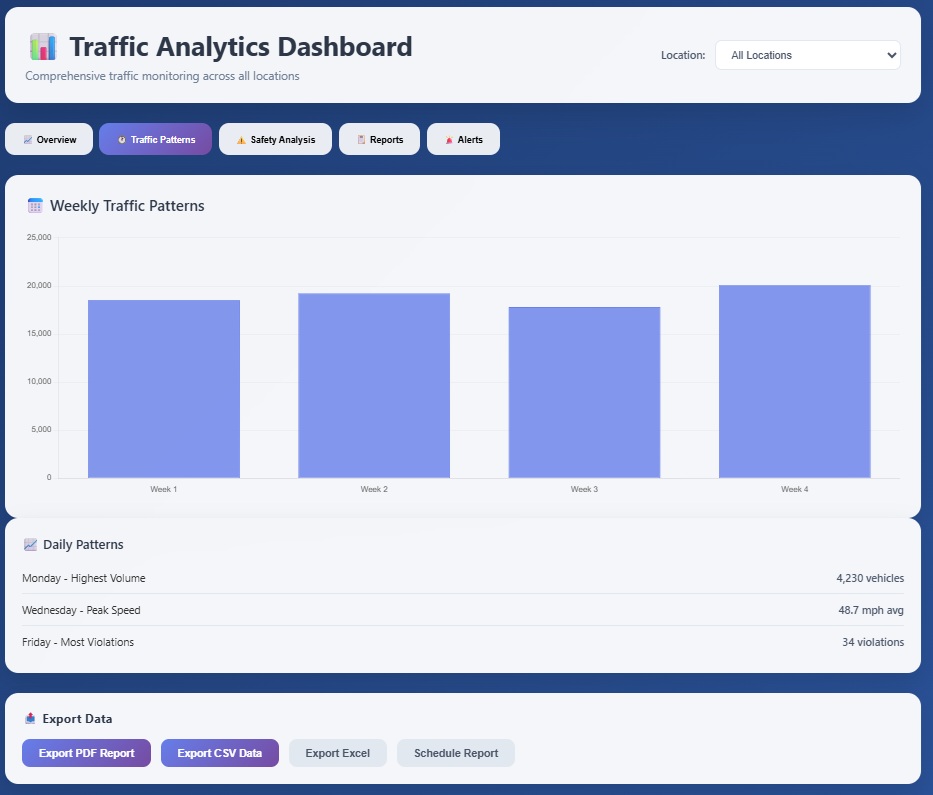
The Edge UI (local web dashboard) provides real-time monitoring, configuration, and analytics for the traffic monitoring system. The Cloud UI (optional) extends these features with historical data and remote management.

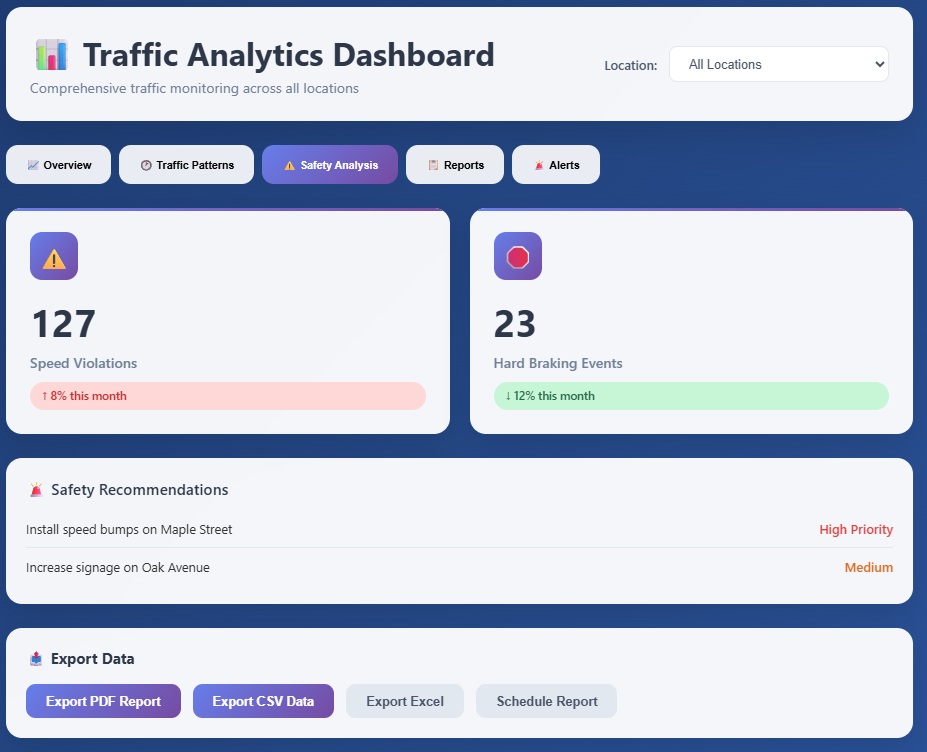
#### Dashboard Layout & Navigation

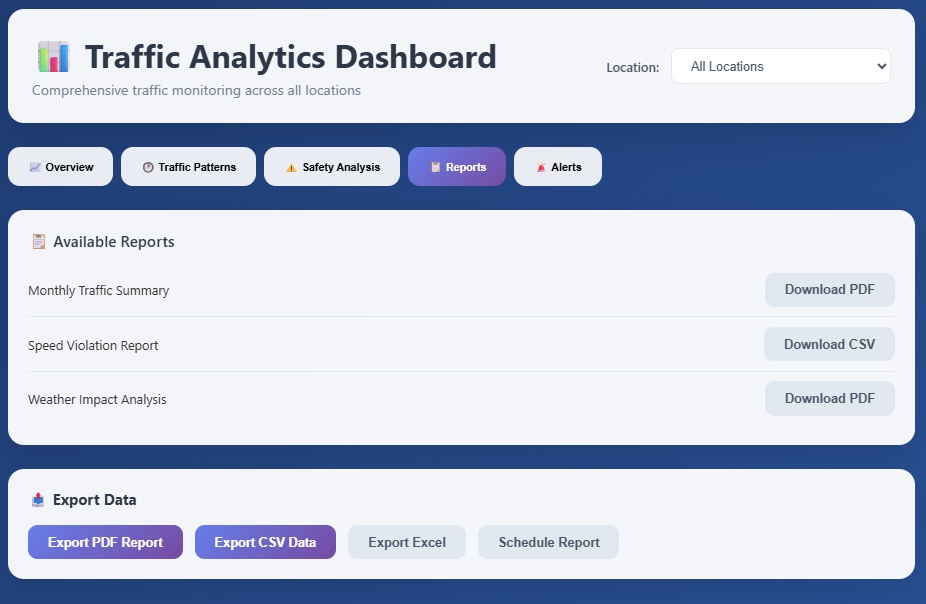
* **Header:** Displays system name, current status, and navigation links (e.g., Live View, Analytics, Settings)
* **Live View Panel:**
  + Real-time video feed from the AI camera (if enabled)
  + Overlay of detected vehicles and bounding boxes
  + Speed readings from radar sensor
* **Event Table:**
  + List of recent vehicle events (timestamp, type, speed, location)
  + Filter/search by date, type, or speed
* **Analytics Panel:**
  + Charts for vehicle counts, speed distributions, and violation rates
  + Historical trends and anomaly alerts
* **System Status:**
  + Health indicators for camera, radar, storage, and network
  + Log messages and error notifications

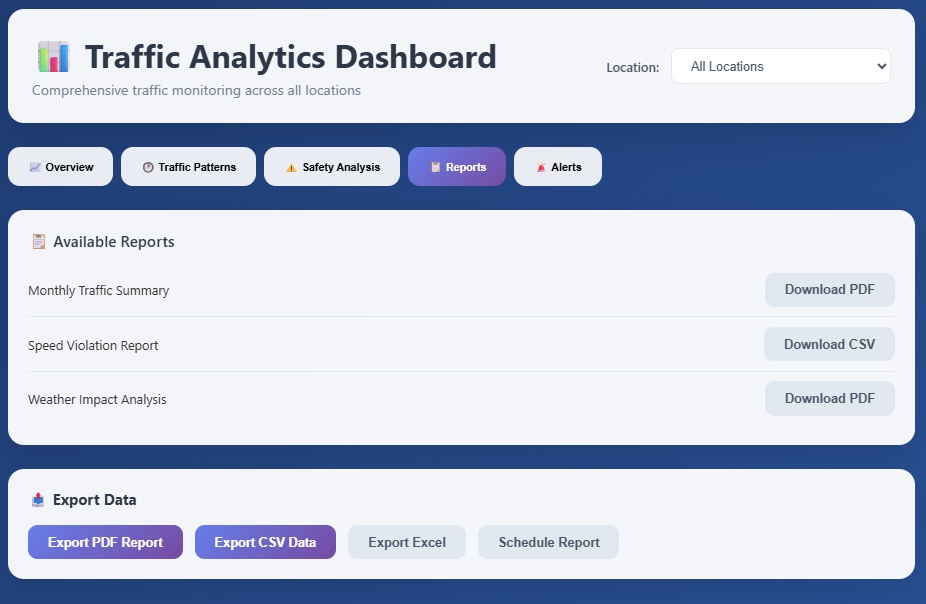
#### Dashboard Screenshots

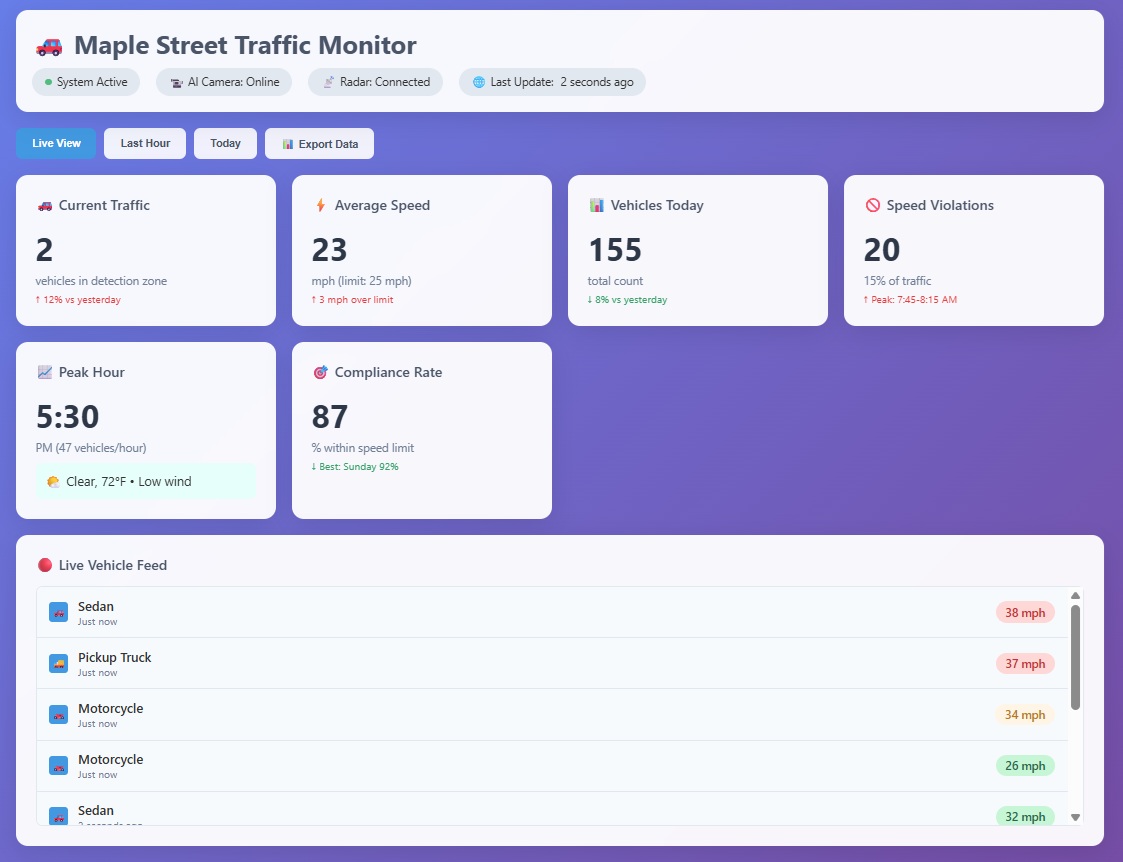
 *Figure 1: Cloud Dashboard - Main Analytics View*

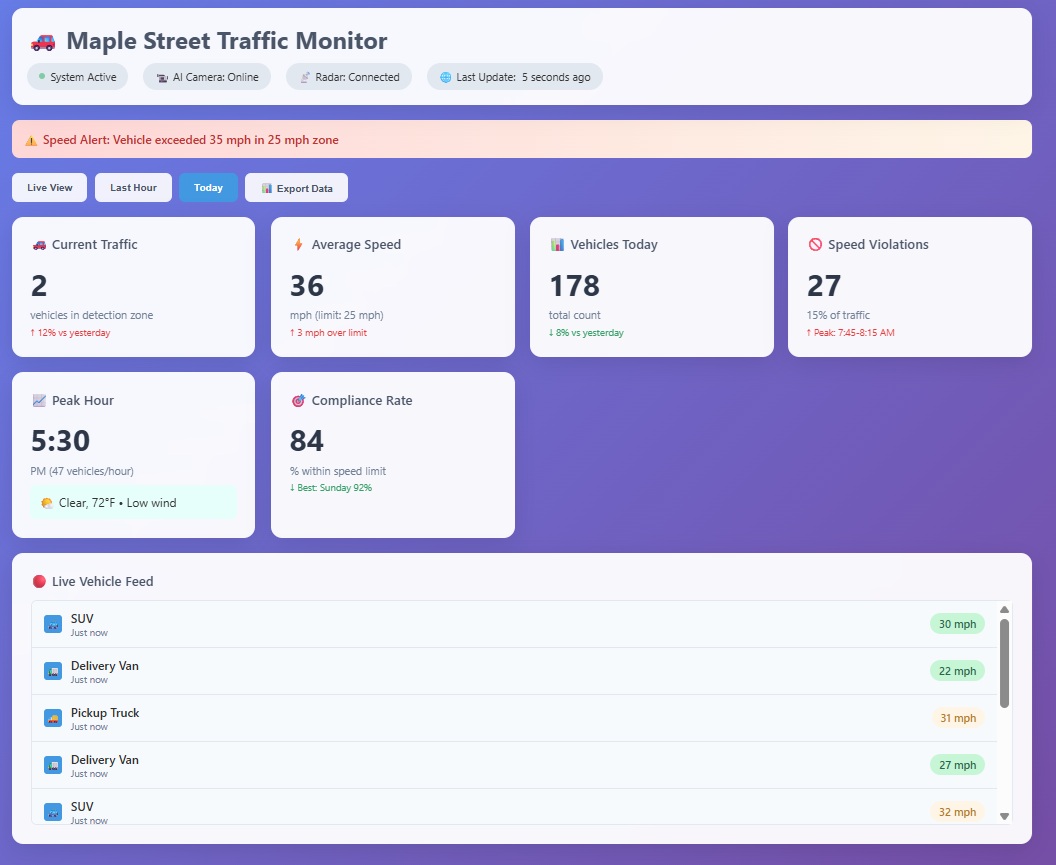
 *Figure 2: Cloud Dashboard - Event Table*

 *Figure 3: Cloud Dashboard - System Status*

 *Figure 4: Cloud Dashboard - Settings Panel*

 *Figure 5: Cloud Dashboard - Historical Trends*

 *Figure 6: Local Edge UI - Live View*

 *Figure 7: Local Edge UI - Analytics Panel*

### 4.2 Monitoring Live Traffic

This section explains how to view real-time vehicle detections as they happen.

#### Step-by-Step: Viewing Real-Time Data

**Step 1: Open the Dashboard**

1. Open your web browser (Chrome, Firefox, Safari, or Edge)
2. Type the dashboard address in the URL bar:
   * **If using GitHub Pages (recommended):** https://gcu-merk.github.io/CST\_590\_Computer\_Science\_Capstone\_Project/
   * **If using Tailscale direct access:** https://edge-traffic-monitoring.taild46447.ts.net/
   * **If on local network:** http://[PI\_IP\_ADDRESS]:5000 (requires direct access to Raspberry Pi)
3. Press **Enter**

💡 **TIP:** The GitHub Pages dashboard is the recommended interface and works from anywhere with internet access. Bookmark this page for quick access!

**Step 2: Navigate to Live View**

1. Look for the navigation menu at the top of the page
2. Click on the **“Live View”** tab or button
3. Wait a few seconds for the live feed to load

✅ **SUCCESS:** You should now see the live camera feed and a list of recent vehicle detections

**Step 3: Observe Vehicle Detections**

When a vehicle is detected, you’ll see:

1. **Visual Indicators:**
   * A colored box (bounding box) appears around the vehicle in the video feed
   * The box color or label shows the vehicle type:
     + 🚗 **Blue box** = Car / Passenger vehicle
     + 🚚 **Green box** = Truck / Large vehicle
     + 🏍️ **Yellow box** = Motorcycle
     + 🚌 **Red box** = Bus
2. **Speed Display:**
   * A number appears next to the vehicle showing its speed
   * Example: “**35 mph**” or “**56 km/h**”
3. **Event Entry:**
   * A new row appears in the event table below the video
   * Shows timestamp, vehicle type, speed, and direction

#### Understanding the Event Table

Each detection creates a new row in the table with the following information:

| Column | What It Means | Example |
| --- | --- | --- |
| **Timestamp** | Date and time the vehicle was detected | 2025-12-20 14:32:15 |
| **Vehicle Type** | What kind of vehicle it is | Car, Truck, Motorcycle, Bus |
| **Speed** | How fast the vehicle was traveling | 35 mph, 56 km/h |
| **Direction** | Which way the vehicle was going | Inbound, Outbound, North, South |
| **Confidence** | How sure the AI is about the vehicle type | 95% (higher is better) |
| **Source** | Which sensor detected it | Camera, Radar, or Fusion (both) |

**What Do the Confidence Scores Mean?**

* **90-100%** = Very confident ✅ (Highly reliable)
* **75-89%** = Confident ⚠️ (Generally reliable)
* **60-74%** = Somewhat confident ℹ️ (Less reliable, verify if important)
* **Below 60%** = Low confidence ❌ (May be inaccurate)

#### Filtering and Searching Events

**To Filter by Date/Time:**

1. Click on the **Date Picker** at the top of the event table
2. Select your desired date range (e.g., “Today”, “Last 7 Days”, or custom dates)
3. Click **Apply**
4. The table refreshes to show only events from that time period

**To Filter by Vehicle Type:**

1. Click on the **Vehicle Type** dropdown menu
2. Check the boxes for the types you want to see:
   * ☑️ Cars
   * ☑️ Trucks
   * ☐ Motorcycles (uncheck to hide)
   * ☑️ Buses
3. Click **Apply** or click outside the menu
4. The table updates to show only selected vehicle types

**To Search for Specific Speeds:**

1. Find the **Speed Filter** section
2. Enter minimum speed (e.g., **40** mph) to see only vehicles going at least that fast
3. Enter maximum speed (e.g., **60** mph) to see vehicles below that speed
4. Click **Apply**
5. Useful for finding speeding violations!

**To Export Data:**

1. After filtering your data, click the **Export** button
2. Choose your format:
   * **CSV** - Opens in Excel or Google Sheets
   * **PDF** - Printable report format
   * **JSON** - For technical integration
3. The file downloads to your computer
4. Use this for reports or record-keeping

💡 **TIP:** Export data at the end of each day for your records!

### 4.3 Viewing Analytics

Analytics help you understand traffic patterns over time. This section shows you how to view charts, interpret data, and generate reports.

#### Step-by-Step: Accessing Analytics

**Step 1: Navigate to Analytics**

1. From the dashboard, click on the **“Analytics”** tab in the main navigation
2. Wait for the charts to load (2-3 seconds)
3. You’ll see several different charts and graphs

**Step 2: Select Your Time Period**

1. Look for the **Time Range Selector** at the top of the page
2. Click the dropdown menu to see options:
   * Today
   * Yesterday
   * Last 7 Days
   * Last 30 Days
   * Custom Range (pick specific dates)
3. Choose your desired time period
4. Charts automatically update to show data for that period

💡 **TIP:** Use “Last 7 Days” to see weekly traffic patterns and identify busy times!

#### Understanding the Analytics Charts

**Chart 1: Vehicle Count Over Time**

**What It Shows:** Total number of vehicles detected each hour or day

**How to Read It:** - **X-axis (horizontal):** Time (hours or days) - **Y-axis (vertical):** Number of vehicles - **Bars:** Height shows vehicle count - taller bars = more traffic

**What to Look For:** - **Peak hours:** The tallest bars show your busiest traffic times - **Trends:** Are bars getting taller over time? Traffic is increasing! - **Patterns:** Do certain days show consistent patterns?

**Example Interpretation:**

Monday 8:00 AM - Bar height: 120 vehicles  
Monday 2:00 PM - Bar height: 45 vehicles  
Monday 5:30 PM - Bar height: 135 vehicles  
  
Interpretation: Morning and evening rush hours have the most traffic

**Chart 2: Vehicle Type Distribution**

**What It Shows:** Breakdown of vehicle types detected (cars, trucks, buses, motorcycles)

**How to Read It:** - **Pie chart** or **bar chart** showing percentages - **Different colors** for each vehicle type: - 🔵 Blue = Cars (usually 70-85%) - 🟢 Green = Trucks (usually 10-20%) - 🟡 Yellow = Motorcycles (usually 2-5%) - 🔴 Red = Buses (usually 1-3%)

**What to Look For:** - **Dominant vehicle type:** Which type is most common? - **Unusual patterns:** Sudden increase in trucks might indicate construction - **Changes over time:** Compare different time periods

**Example Interpretation:**

Today: 78% Cars, 15% Trucks, 5% Motorcycles, 2% Buses  
Last Month: 82% Cars, 12% Trucks, 4% Motorcycles, 2% Buses  
  
Interpretation: Slight increase in truck traffic this month

**Chart 3: Speed Distribution**

**What It Shows:** How fast vehicles are traveling - helps identify speeding

**How to Read It:** - **Histogram** (bars grouped by speed ranges) - **X-axis:** Speed ranges (e.g., 0-25 mph, 25-35 mph, 35-45 mph) - **Y-axis:** Number of vehicles in each range - **Red line:** Speed limit (if configured)

**What to Look For:** - **Bars to the right of red line:** Vehicles exceeding speed limit (speeding) - **Tallest bar:** Most common speed range - **Spread:** Are speeds concentrated or spread out?

**Example Interpretation:**

Speed Limit: 35 mph  
Vehicles under 35 mph: 450 (75%)  
Vehicles over 35 mph: 150 (25%)  
  
Interpretation: 1 in 4 vehicles are speeding

**Chart 4: Traffic Flow by Direction**

**What It Shows:** Traffic volume separated by direction (inbound vs. outbound)

**How to Read It:** - **Two lines** (one for each direction) - **Blue line:** Traffic flowing one direction (e.g., Northbound) - **Orange line:** Traffic flowing opposite direction (e.g., Southbound)

**What to Look For:** - **Balanced lines:** Equal traffic both directions - **Imbalanced lines:** One direction much busier (common during rush hour) - **Crossover points:** Times when traffic direction switches

**Example Interpretation:**

7:00 AM: Inbound = 80 vehicles, Outbound = 25 vehicles  
5:00 PM: Inbound = 30 vehicles, Outbound = 95 vehicles  
  
Interpretation: Morning commute into town, evening commute out of town

#### Step-by-Step: Generating Reports

Reports allow you to save and share traffic data.

**Step 1: Choose Report Type**

1. Click the **“Generate Report”** button in the Analytics section
2. Select from predefined templates:
   * **Daily Summary:** Traffic data for one day
   * **Weekly Report:** 7-day traffic patterns
   * **Monthly Report:** Full month analysis
   * **Custom Report:** Choose your own parameters

**Step 2: Configure Report Settings**

1. **Set Date Range:**
   * Click the calendar icon
   * Select start date and end date
   * Click **Apply**
2. **Choose What to Include:**
   * ☑️ Vehicle counts
   * ☑️ Speed violations
   * ☑️ Vehicle type breakdown
   * ☑️ Charts and graphs
   * ☑️ Peak hour analysis
3. **Add Filters (Optional):**
   * Include only specific vehicle types
   * Show only speeding violations
   * Filter by time of day

**Step 3: Generate and Download**

1. Click the **“Generate Report”** button
2. Wait for processing (5-30 seconds depending on data volume)
3. Preview the report on screen
4. Choose your download format:
   * **PDF** - Best for printing and presentations
   * **Excel/CSV** - Best for further analysis in spreadsheets
   * **HTML** - Best for viewing in web browser
5. Click **“Download”**
6. File saves to your Downloads folder

✅ **SUCCESS:** Your report is ready! Check your Downloads folder.

💡 **TIP:** Generate weekly reports every Monday to track trends over time!

#### Common Analysis Tasks

**Task 1: Identify Peak Traffic Hours**

1. Go to Analytics → Vehicle Count Over Time chart
2. Set time range to “Last 7 Days”
3. Look for the tallest bars - these are peak hours
4. Note the times for scheduling road work or enforcement

**Task 2: Check Compliance with Speed Limits**

1. Go to Analytics → Speed Distribution chart
2. Note the red line (speed limit)
3. Count bars to the right of the line (speeders)
4. Calculate percentage: (speeders ÷ total vehicles) × 100

**Task 3: Compare Traffic Patterns Week-Over-Week**

1. Generate a report for “Last 7 Days”
2. Generate another report for “Previous 7 Days”
3. Compare vehicle counts between the two periods
4. Look for increases/decreases and identify causes

**Task 4: Monitor Specific Vehicle Types**

1. Use vehicle type filter to show only trucks
2. Set date range to last 30 days
3. Look for unusual increases
4. Could indicate construction, delivery routes, or detours

### 4.4 Radar System Monitoring

This section explains how to monitor the radar system operation and verify that vehicle detections are working correctly.

#### Real-Time Radar Monitoring

**Checking Radar Service Status:**

To verify the radar service is running properly:

# Connect to your Pi via SSH  
ssh pi@raspberrypi.local  
  
# Check if radar-service container is running  
docker ps | grep radar-service  
  
# Expected output: radar-service container with "healthy" status

**Monitoring Live Radar Detections:**

View real-time vehicle detections as they happen:

# Follow radar service logs in real-time  
docker logs radar-service -f  
  
# Example output when a vehicle passes:  
# Vehicle detected: 23.9 mph (magnitude: unknown)  
# LOW SPEED ALERT: 23.9 mph

**Checking Recent Detections in Redis:**

View the last few vehicle detections stored in Redis:

# Check last 5 radar detections  
docker exec redis redis-cli XREVRANGE radar\_data + - COUNT 5  
  
# Check latest detection data in Redis  
docker exec redis redis-cli KEYS \*

#### System Health Dashboard

**Accessing the Health Dashboard:**

1. **Public access (recommended):**
   * GitHub Pages: <https://gcu-merk.github.io/CST_590_Computer_Science_Capstone_Project/>
2. **Remote secure access with Tailscale:**
   * Tailscale VPN: https://edge-traffic-monitoring.taild46447.ts.net/
   * (Requires Tailscale VPN connection to your network)
3. **Dashboard sections to check:**
   * **System Status:** Shows if all services are running
   * **Recent Detections:** Displays latest vehicle detections with timestamps
   * **Radar Health:** Indicates radar sensor connectivity and data quality

**Health Check Endpoints:**

Test system health programmatically:

# Basic health check  
curl https://edge-traffic-monitoring.taild46447.ts.net/health  
  
# Detailed system health (includes all services)  
curl https://edge-traffic-monitoring.taild46447.ts.net/api/health/system  
  
# Expected response for healthy system:  
{  
 "status": "healthy",  
 "services": {  
 "redis": "running",  
 "traffic-monitor": "running",  
 "radar-service": "running",  
 "data-maintenance": "running",  
 "airport-weather": "running",  
 "dht22-weather": "running"  
 },  
 "camera\_host\_service": "imx500 (runs on host, not containerized)"  
}

### 4.5 Mobile App (Optional)

If a mobile app is implemented, it provides remote access to real-time and historical traffic data, system status, and configuration options. The app uses the same REST and WebSocket APIs as the dashboard.

#### Key Screens

* **Login:** Secure authentication for users
* **Live Events:** Real-time list of detected vehicles, speeds, and locations
* **Analytics:** Charts for traffic volume, speed, and violations
* **System Status:** Health indicators for camera, radar, and connectivity
* **Settings:** User preferences and notification options

#### Typical Workflows

1. **Login and Authentication:**
   * User logs in with credentials (or SSO if enabled)
2. **Monitor Live Events:**
   * View real-time vehicle detections and speed data
   * Tap on an event for more details
3. **Review Analytics:**
   * Access historical charts and reports
   * Filter by date, location, or event type
4. **Check System Status:**
   * Monitor health of edge devices and receive alerts
5. **Adjust Settings:**
   * Set notification preferences or update user profile

**Note:** The mobile app is optional and may be implemented as a progressive web app (PWA) for cross-platform compatibility.

### 4.6 Common Use Cases

#### Use Case 1: Live Traffic Monitoring

1. Operator opens the Edge UI dashboard in a browser
2. Observes real-time video feed and vehicle detections with speed overlays
3. Checks system status indicators for camera, radar, and storage health
4. Reviews the event table for recent detections and violations

#### Use Case 2: Reviewing Historical Analytics

1. Traffic analyst logs into the Cloud UI dashboard
2. Navigates to the Analytics panel to view charts of vehicle counts and speed distributions over the past week
3. Filters data by date, location, or vehicle type to identify trends or anomalies
4. Downloads a report for further analysis or presentation

#### Use Case 3: Investigating a Speed Violation

1. Operator receives an alert for a speeding event
2. Clicks on the event in the dashboard to view details (timestamp, speed, vehicle type, location)
3. Reviews associated video and radar data for verification
4. Exports event data for record-keeping or enforcement

#### Use Case 4: System Maintenance

1. Administrator logs into the dashboard and checks system health logs
2. Notices a warning about low storage space
3. Initiates a backup and clears old data as needed
4. Updates system software and dependencies via the maintenance panel

## 5. Troubleshooting

### 5.1 Common Issues

## 5. Troubleshooting

### 5.1 Common Issues

#### Dashboard Shows “No Data”

**Problem:** Dashboard displays “No data” message when you expect to see vehicle detections.

**Solutions:**

1. **Verify service status:**

* docker compose ps  
  # All services should show "running" status

1. **Check camera and radar connections:**

* # Test camera  
  libcamera-hello --preview=none --timeout=5000  
    
  # Test radar  
  sudo cat /dev/ttyAMA0

1. **Check database accessibility:**

* # Verify database file exists  
  ls -la /mnt/storage/traffic.db  
    
  # Check database service logs  
  docker logs database-persistence --tail 20

1. **Review system logs for errors:**

* # Check all service logs  
  docker compose logs --tail=50

#### Vehicle Detections Seem Inaccurate

**Problem:** System is detecting vehicles, but classifications or speeds seem incorrect.

**Solutions:**

1. **Clean the camera lens:**
   * Power off the system
   * Gently clean the IMX500 camera lens with a microfiber cloth
   * Ensure no obstructions block the camera view
2. **Adjust camera angle:**
   * Ensure camera has clear view of roadway
   * Optimal angle is 20-30 degrees from horizontal
   * Avoid backlighting or direct sunlight
3. **Check lighting conditions:**
   * System works best in daylight or good artificial lighting
   * Add external lighting for nighttime operation if needed
4. **Verify radar alignment:**
   * Radar should point directly at traffic flow
   * Check for obstructions blocking radar beam
   * Ensure radar is securely mounted
5. **Review confidence thresholds:**

* # Check current detection settings  
  cat config/detection\_settings.yaml  
    
  # Lower thresholds may increase detections but reduce accuracy

#### System Running Slowly or Unresponsive

**Problem:** Dashboard is slow to load or system is unresponsive.

**Solutions:**

1. **Check system resources:**

* # Check CPU and memory usage  
  top  
    
  # Check disk space  
  df -h

1. **Restart services:**

* # Restart all services  
  docker compose restart  
    
  # Or restart individual service  
  docker compose restart radar-service

1. **Clear old data:**

* # Archive old logs  
  mv /mnt/storage/logs/\* /mnt/storage/archive/  
    
  # Clean up old database records (keep last 30 days)  
  python3 scripts/cleanup\_old\_data.py --days 30

1. **Check for runaway processes:**

* docker stats

### 5.2 Radar Detection Problems

#### Radar Service Not Detecting Vehicles

**Problem:** No vehicle detections appearing in radar logs.

**Solutions:**

1. **Check UART connection:**

* # Verify radar device is accessible  
  ls -la /dev/ttyAMA0  
    
  # Should show device with proper permissions (crw-rw----)

1. **Add user to dialout group:**

* sudo usermod -aG dialout $USER  
  # Log out and back in for changes to take effect

1. **Verify radar service logs for errors:**

* docker logs radar-service --tail 20  
    
  # Look for Redis connection errors or UART issues

1. **Test radar manually:**
   * Safely walk or drive past the radar sensor
   * Check logs immediately: docker logs radar-service -f
   * You should see speed readings
2. **Check radar power:**
   * Verify 5V power is connected to radar sensor
   * Check for loose connections
   * Measure voltage at radar pins if possible

#### Inconsistent Speed Readings

**Problem:** Radar reports erratic or obviously incorrect speeds.

**Solutions:**

1. **Calibrate radar orientation:**
   * Radar works best when aimed directly at traffic flow
   * Angle sensor to minimize ground clutter
   * Secure mounting to prevent vibration
2. **Check for interference:**
   * Move away from other radar sources
   * Ensure no metal objects in radar path
   * Check for EMI from nearby equipment
3. **Verify configuration:**

* # Check radar configuration  
  docker exec radar-service cat /config/radar\_config.yaml

1. **Test with known speed reference:**
   * Use GPS speedometer or radar gun
   * Compare readings at various speeds
   * Contact support if consistent offset detected

### 5.3 Dashboard Issues

#### Cannot Access Dashboard

**Problem:** Browser cannot connect to dashboard URL.

**Solutions:**

1. **Try the GitHub Pages dashboard first:**

* https://gcu-merk.github.io/CST\_590\_Computer\_Science\_Capstone\_Project/

1. **If accessing via Tailscale, verify VPN connection:**

* # Check Tailscale status  
  tailscale status  
    
  # Verify connection to the edge device  
  curl https://edge-traffic-monitoring.taild46447.ts.net/health

1. **For local network access, verify Raspberry Pi is reachable:**

* # On the Pi, check IP address  
  hostname -I  
    
  # Ping the Pi from your computer (replace with actual IP)  
  ping raspberrypi.local

1. **Check firewall settings:**

* # On the Pi, check if port 5000 is listening  
  sudo netstat -tulpn | grep 5000  
    
  # Allow port through firewall if needed  
  sudo ufw allow 5000/tcp

1. **Verify API service is running:**

* docker ps | grep api-gateway  
  docker logs api-gateway --tail 20

1. **Try different browser:**
   * Clear browser cache and cookies
   * Try incognito/private browsing mode
   * Test with different browser (Chrome, Firefox, Edge)

#### Dashboard Shows Old Data

**Problem:** Dashboard displays outdated information.

**Solutions:**

1. **Force browser refresh:**
   * Press Ctrl+F5 (Windows/Linux) or Cmd+Shift+R (Mac)
   * This clears browser cache
2. **Check WebSocket connection:**
   * Open browser developer console (F12)
   * Look for WebSocket connection errors
   * Verify real-time updates are working
3. **Verify data services:**

* # Check traffic-monitor service (API gateway)  
  docker logs traffic-monitor --tail 20  
    
  # Verify recent data in Redis  
  docker exec redis redis-cli XREVRANGE radar\_data + - COUNT 5

1. **Restart API gateway:**

* docker compose restart api-gateway

### 5.4 Network Connectivity

#### Cannot Connect Remotely via Tailscale

**Problem:** Unable to access dashboard using Tailscale IP.

**Solutions:**

1. **Verify Tailscale is running:**

* # On Raspberry Pi  
  sudo tailscale status  
    
  # Should show "online" status

1. **Check Tailscale IP:**

* tailscale ip -4  
  # Use this IP to access dashboard

1. **Restart Tailscale:**

* sudo tailscale down  
  sudo tailscale up

1. **Check ACL rules:**
   * Log into Tailscale admin console
   * Verify device has proper access permissions
   * Check for any blocklisted ports

#### Slow Network Performance

**Problem:** Dashboard loads slowly or data updates are delayed.

**Solutions:**

1. **Test network speed:**

* # Install speedtest-cli if needed  
  sudo apt install speedtest-cli  
    
  # Run speed test  
  speedtest-cli

1. **Check local network:**
   * Restart router/switch
   * Test with wired Ethernet connection
   * Move closer to WiFi access point
2. **Reduce data polling frequency:**
   * Edit dashboard settings
   * Increase refresh interval from 1s to 5s
   * Reduces network traffic
3. **Optimize database queries:**

* # Vacuum database to improve performance  
  docker exec database-persistence sqlite3 /data/traffic.db "VACUUM;"

### 5.5 Performance Monitoring

#### Monitoring System Performance

**Key Metrics to Monitor:**

* **Detection Rate:** Number of vehicles detected per hour
* **Response Time:** Time from radar detection to dashboard update (<350ms target)
* **Storage Usage:** Disk space used by logs and database
* **Service Uptime:** How long services have been running without restart
* **CPU/Memory Usage:** System resource consumption

#### Performance Monitoring Commands

# Check service uptime  
docker ps --format "table {{.Names}}\t{{.Status}}"  
  
# Monitor storage usage  
du -h /mnt/storage/logs/  
df -h /mnt/storage  
  
# Check Redis memory usage  
docker exec redis redis-cli INFO memory  
  
# View system resources  
htop

#### Maintenance Tasks

**Daily Checks:**

* Verify dashboard is accessible
* Check recent detections show reasonable vehicle counts
* Monitor system disk space usage
* Review error logs for issues

**Weekly Maintenance:**

* Review radar service logs for errors
* Check that detection data is being properly stored in Redis
* Verify backup systems (if configured)
* Test remote access capabilities

**Monthly Tasks:**

* Update system software: sudo apt update && sudo apt upgrade
* Review and archive old log files
* Test emergency procedures and failover systems
* Verify all sensors and hardware connections

## 6. FAQ

### 6.1 General Questions

## 6. FAQ

### 6.1 General Questions

**Q: What types of vehicles can the system detect?**

A: The AI camera can detect and classify cars, trucks, motorcycles, and buses. Detection accuracy is highest for standard passenger vehicles (85-95%) and decreases slightly for unusual vehicle types or motorcycles due to their smaller size.

**Q: How accurate is the speed measurement?**

A: The OPS243-C radar sensor provides ±2 mph accuracy for vehicles traveling up to 60 mph under normal conditions. Accuracy may decrease in adverse weather or with very small vehicles (motorcycles, bicycles).

**Q: Can the system work in all weather conditions?**

A: Yes, the system is designed for outdoor use with an IP65-rated enclosure. The radar sensor works reliably in all weather conditions including rain, snow, and fog. Camera-based AI classification may have reduced accuracy in very poor visibility but the radar will continue to detect vehicles and measure speed.

**Q: How much data storage is required?**

A: With default settings, the system generates approximately 1-2 GB of data per day, including logs, detection events, and periodic image captures. The recommended 1TB external SSD provides several months to over a year of storage, depending on traffic volume and retention settings.

**Q: Is the system compliant with privacy regulations?**

A: Yes. All AI processing occurs on the IMX500 camera sensor itself, and the system does not store continuous video footage by default. Only metadata (timestamps, vehicle types, speeds) is logged. The system can be configured to capture snapshots for verification purposes, with appropriate retention policies to comply with local privacy laws.

**Q: Can I integrate this system with existing traffic management systems?**

A: Yes. The system provides REST and WebSocket APIs that can be integrated with external traffic management platforms. See the Technical Design Document for API specifications and integration examples.

### 6.2 Technical Questions

**Q: What network connectivity is required?**

A: The system requires an internet connection for cloud synchronization and remote access. Local operation (Edge UI) works without internet. Minimum bandwidth requirement is 1 Mbps for cloud features. For optimal performance, 5-10 Mbps is recommended, especially if capturing and uploading snapshots.

**Q: Can I access the system remotely?**

A: Yes. The system supports secure remote access using Tailscale, a mesh VPN that creates a private network between your devices. Tailscale allows you to SSH or access the dashboard from anywhere, as if you were on the same local network. See the Implementation & Deployment Guide for setup instructions.

**Q: How do I connect to the Raspberry Pi using Tailscale?**

A: The system uses Tailscale for secure remote access. After installing Tailscale on both your Raspberry Pi and your client device (laptop/desktop), you can access the system:

**Option 1: Web Dashboard (Recommended)**

https://gcu-merk.github.io/CST\_590\_Computer\_Science\_Capstone\_Project/

The GitHub Pages dashboard will automatically connect to your Tailscale-protected API.

**Option 2: Direct API Access**

# SSH to Raspberry Pi  
ssh merk@edge-traffic-monitoring.taild46447.ts.net  
  
# Or access API directly  
https://edge-traffic-monitoring.taild46447.ts.net/api

Ensure Tailscale is running and connected on both devices. For more details, see the Implementation & Deployment Guide.

**Q: How do I calibrate the speed measurement?**

A: The OPS243-C radar sensor is factory-calibrated and typically requires no additional calibration. However, you can verify accuracy by comparing readings with a GPS speedometer or radar gun. If you notice consistent offsets, check the radar orientation and mounting. Contact support if recalibration is needed.

**Q: What happens if the power goes out?**

A: The system will automatically resume normal operation when power is restored. All services are configured to start on boot. If you have a UPS (Uninterruptible Power Supply), the Raspberry Pi can continue operating for several hours during outages, depending on UPS capacity.

**Q: How do I back up my data?**

A: You can back up data using several methods:

1. **Manual backup:**

* # Copy database to backup location  
  cp /mnt/storage/traffic.db /path/to/backup/

1. **Automated backup script:**

* # Run backup script (if configured)  
  ./scripts/backup\_data.sh

1. **Cloud synchronization:** Enable cloud sync in settings to automatically upload data to cloud storage.

**Q: Can I adjust detection sensitivity or thresholds?**

A: Yes, detection thresholds and sensitivity can be adjusted in the configuration files. This allows you to fine-tune the system for your specific deployment:

* **AI Confidence Threshold:** Minimum confidence score for vehicle classification (default: 0.70)
* **Speed Threshold:** Minimum/maximum speed for logging (e.g., ignore stationary objects)
* **ROI (Region of Interest):** Define specific areas of the camera frame to monitor

See the Technical Design Document for detailed configuration parameters.

**Q: How many vehicles can the system handle simultaneously?**

A: The system can reliably process 3-5 vehicles simultaneously in the camera’s field of view. The IMX500’s on-chip AI can handle multiple detections in a single frame, and the radar can track multiple targets. Performance depends on vehicle spacing and speed.

### 6.3 Troubleshooting Questions

**Q: The dashboard shows “No data” - what should I check?**

A:

1. Verify the camera and radar connections
2. Check that all services are running: docker compose ps
3. Ensure the database is accessible: ls -la /mnt/storage/traffic.db
4. Look for error messages in the system logs: docker compose logs --tail=50

See the [Troubleshooting](#Xd5b3d8a68c73b756968aad1dd84151376a19c17) section for detailed solutions.

**Q: How do I update the system software?**

A: Software updates should be performed carefully to avoid disrupting service:

1. **Backup your data first:**

* ./scripts/backup\_data.sh

1. **Pull latest code:**

* git pull origin main

1. **Rebuild containers:**

* docker compose down  
  docker compose build  
  docker compose up -d

1. **Verify services:**

* docker compose ps

Always backup your data before performing updates. See the Implementation & Deployment Guide for detailed update procedures.

**Q: Why are my vehicle classifications inaccurate?**

A: Inaccurate classifications can result from several factors:

1. **Poor lighting:** Ensure adequate lighting for camera
2. **Camera angle:** Optimize camera position for side or front/rear vehicle views
3. **Dirty lens:** Clean camera lens with microfiber cloth
4. **Wrong ROI:** Adjust region of interest to focus on road area
5. **Low confidence threshold:** Increase threshold to filter low-confidence detections

See [Vehicle Detections Seem Inaccurate](#vehicle-detections-seem-inaccurate) for detailed troubleshooting steps.

**Q: How do I reset the system to factory defaults?**

A: To reset the system:

1. **Backup important data first**
2. **Stop all services:**

* docker compose down

1. **Remove data volumes:**

* docker volume rm $(docker volume ls -q)

1. **Restore default configuration:**

* cp config/default\_config.yaml config/config.yaml

1. **Restart services:**

* docker compose up -d

⚠️ **Warning:** This will delete all historical data and reset all settings.

## 7. Help and Contact Details

### 7.1 Support Channels

If you have questions, encounter issues, or need assistance with the Raspberry Pi 5 Edge ML Traffic Monitoring System, please use the following support channels:

#### Documentation Resources

* **Technical Design Document:** [Technical\_Design.md](./Technical_Design.md) - Detailed technical specifications and architecture
* **Implementation & Deployment Guide:** [Implementation\_Deployment.md](./Implementation_Deployment.md) - Installation and deployment instructions
* **Project Management Summary:** [Project\_Management.md](./Project_Management.md) - Project overview and planning documents
* **References & Appendices:** [References\_Appendices.md](./References_Appendices.md) - Additional references and resources

#### Online Resources

* **GitHub Repository:** [CST\_590\_Computer\_Science\_Capstone\_Project](https://github.com/gcu-merk/CST_590_Computer_Science_Capstone_Project)
  + View source code, report issues, and contribute to the project
  + Check the Wiki for additional documentation and tutorials
  + Review closed issues for solutions to common problems
* **Community Forum:** Join discussions with other users and developers (link TBD)
* **Video Tutorials:** Step-by-step video guides for installation and configuration (link TBD)

#### Direct Support

* **Email Support:** Contact the project team at support@example.com (replace with actual support email)
* **Issue Tracking:** Report bugs or request features via GitHub Issues
* **Commercial Support:** For enterprise deployments or custom development, contact the project maintainers

### 7.2 Additional Resources

#### Hardware Documentation

* **Raspberry Pi 5 Documentation:** [raspberrypi.com/documentation](https://www.raspberrypi.com/documentation/)
* **Sony IMX500 Camera:** See Technical Design Document for specifications
* **OPS243-C Radar Sensor:** [OmniPreSense Documentation](https://omnipresense.com/product/ops243-doppler-radar-sensor/)

#### Software and Tools

* **Docker Documentation:** [docs.docker.com](https://docs.docker.com/)
* **Python Documentation:** [python.org/doc](https://www.python.org/doc/)
* **Tailscale VPN:** [tailscale.com/kb](https://tailscale.com/kb/)

#### Learning Resources

* **Edge AI and ML:** Resources on edge machine learning and computer vision
* **Traffic Monitoring Best Practices:** Guidelines for traffic monitoring systems
* **IoT and Sensor Networks:** Background on IoT system design

### 7.3 Reporting Issues

When contacting support or reporting an issue, please provide the following information to help us assist you more effectively:

#### Required Information

1. **System Version:**

* cat version.txt  
  # Or check git commit hash  
  git rev-parse HEAD

1. **Hardware Configuration:**
   * Raspberry Pi model (Pi 5 8GB, etc.)
   * Camera model (IMX500)
   * Radar sensor (OPS243-C)
   * Storage device type and capacity
2. **Issue Description:**
   * Clear description of the problem
   * What you expected to happen
   * What actually happened
   * When the issue started
3. **Error Messages and Logs:**

* # Collect service logs  
  docker compose logs > system\_logs.txt  
    
  # Check system logs  
  journalctl -xe > system\_journal.txt  
    
  # Include these files with your issue report

1. **Steps to Reproduce:**
   * Detailed steps to reproduce the problem
   * Frequency (always, intermittent, rare)
   * Any recent changes or updates
2. **Screenshots:**
   * If applicable, include screenshots of dashboard or error messages
   * Use browser developer console (F12) to capture JavaScript errors

#### Bug Report Template

\*\*System Version:\*\* [e.g., v1.2, commit abc123]  
\*\*Hardware:\*\* [e.g., Raspberry Pi 5 8GB, IMX500, OPS243-C]  
\*\*OS:\*\* [e.g., Raspberry Pi OS 64-bit, Bullseye]  
  
\*\*Description:\*\*  
[Clear description of the issue]  
  
\*\*Steps to Reproduce:\*\*  
1. [First step]  
2. [Second step]  
3. [And so on...]  
  
\*\*Expected Behavior:\*\*  
[What you expected to happen]  
  
\*\*Actual Behavior:\*\*  
[What actually happened]  
  
\*\*Error Messages:\*\*  
[Paste any error messages or logs]  
  
\*\*Additional Context:\*\*  
[Any other relevant information]

## 8. Glossary

### Core System Terms

**AI Camera (IMX500):** Sony 12MP camera sensor with built-in neural processing unit that performs on-chip vehicle classification

**API Gateway:** Service that provides REST and WebSocket endpoints for external system integration; in this system, the traffic-monitor service acts as the API gateway

**Cloud UI:** Optional cloud-based dashboard for historical analytics and remote management; this system uses GitHub Pages for the public dashboard

**Data Integration:** The process of combining data from multiple sensors (IMX500 camera and OPS243-C radar) into a unified stream for analysis and storage in Redis

**Detection Event:** A recorded instance of a vehicle detection, including timestamp, vehicle type, speed, and location

**Edge AI:** Artificial intelligence processing performed locally on the device (camera sensor) rather than in the cloud

**Edge UI:** Local web dashboard for real-time monitoring and system configuration

**FMCW Radar:** Frequency-Modulated Continuous Wave radar technology used in the OPS243-C sensor

**Fusion:** The process of combining data from multiple sensors (radar + camera) to create more accurate detections

**GPIO:** General Purpose Input/Output pins on the Raspberry Pi used to interface with sensors

**IMX500:** Sony image sensor with integrated AI processing capabilities

**OPS243-C:** 24.125 GHz Doppler radar sensor for vehicle speed and direction detection

**Redis:** In-memory data store used for real-time event streaming between services

**ROI (Region of Interest):** Specific area of the camera frame designated for vehicle monitoring

**SQLite:** Embedded database used for persistent storage of detection events

**Tailscale:** Mesh VPN service for secure remote access to the system

**UART:** Universal Asynchronous Receiver-Transmitter serial communication protocol used for radar sensor

**WebSocket:** Communication protocol for real-time bidirectional data exchange between server and client

### Technical Terms

**Bounding Box:** Rectangle drawn around detected vehicles in the camera frame

**Classification:** Process of identifying vehicle type (car, truck, motorcycle, bus) using AI

**Confidence Score:** AI certainty level (0-100%) in vehicle classification accuracy

**Container:** Isolated environment for running services using Docker

**Detection Threshold:** Minimum confidence or signal strength required to register a vehicle detection

**Docker Compose:** Tool for defining and running multi-container Docker applications

**Microservices:** Software architecture where system is composed of small, independent services

**Neural Network:** AI model used by IMX500 camera to classify vehicle types

**PoE+ (Power over Ethernet Plus):** Technology that provides power and network connectivity over a single cable

**SSH (Secure Shell):** Protocol for secure remote command-line access to the Raspberry Pi

**UPS (Uninterruptible Power Supply):** Backup battery system to maintain power during outages

**YOLO (You Only Look Once):** Real-time object detection algorithm often used in traffic monitoring

### Traffic Monitoring Terms

**Direction:** Direction of vehicle travel (typically inbound/outbound, northbound/southbound, etc.)

**Speed Violation:** Vehicle traveling faster than the configured speed limit

**Traffic Analytics:** Statistical analysis of traffic patterns, volumes, and trends

**Traffic Volume:** Number of vehicles detected within a specific time period

**Vehicle Classification:** Categorization of vehicles by type (passenger car, truck, motorcycle, bus)

### Abbreviations

* **AI:** Artificial Intelligence
* **API:** Application Programming Interface
* **CSI:** Camera Serial Interface
* **CSV:** Comma-Separated Values
* **DHCP:** Dynamic Host Configuration Protocol
* **EMI:** Electromagnetic Interference
* **GPIO:** General Purpose Input/Output
* **HTTP:** Hypertext Transfer Protocol
* **HTTPS:** HTTP Secure
* **IP:** Internet Protocol (also: Ingress Protection for weatherproofing)
* **JSON:** JavaScript Object Notation
* **LED:** Light-Emitting Diode
* **ML:** Machine Learning
* **NPU:** Neural Processing Unit
* **OS:** Operating System
* **PDF:** Portable Document Format
* **PWA:** Progressive Web App
* **REST:** Representational State Transfer
* **ROI:** Region of Interest
* **SSD:** Solid-State Drive
* **SSH:** Secure Shell
* **SSL/TLS:** Secure Sockets Layer / Transport Layer Security
* **UART:** Universal Asynchronous Receiver-Transmitter
* **UI:** User Interface
* **UPS:** Uninterruptible Power Supply
* **URL:** Uniform Resource Locator
* **USB:** Universal Serial Bus
* **VPN:** Virtual Private Network
* **WiFi:** Wireless Fidelity

## Appendices

### A. Future Enhancements

* **Mobile App Expansion:** Native iOS and Android apps with push notifications
* **User Customization:** Advanced user roles, permissions, and dashboard customization
* **Accessibility Improvements:** Enhanced UI accessibility with screen reader support
* **Automated Alerts:** Flexible, user-defined alerting and notification options
* **Multi-Site Management:** Support for managing multiple installations from a single dashboard
* **Advanced Analytics:** Machine learning for traffic prediction and anomaly detection
* **License Plate Recognition:** Optional LPR integration for enforcement applications (subject to privacy regulations)

### B. Related Documentation

For additional information, please refer to:

* **Technical Design Document:** Detailed system architecture, hardware specifications, and software design
* **Implementation & Deployment Guide:** Step-by-step installation and configuration instructions
* **Project Management Summary:** Project planning, timeline, and deliverables
* **References & Appendices:** Additional resources, references, and supplementary information

### C. Document Revision History

| Version | Date | Author | Changes |
| --- | --- | --- | --- |
| 1.0 | December 11, 2025 | Documentation Team | Initial release |
| 1.1 | December 15, 2025 | Documentation Team | Added troubleshooting sections |
| 1.2 | December 20, 2025 | Documentation Team | Restructured with standard user guide format |

### D. Acknowledgments

This project was developed as part of the CST 590 Computer Science Capstone Project. Special thanks to:

* Grand Canyon University Computer Science Department
* Raspberry Pi Foundation for hardware platform
* Sony Corporation for IMX500 AI camera technology
* OmniPreSense for OPS243-C radar sensor
* Open source community for Docker, Python, and related tools

**End of User Guide**

For questions or feedback about this documentation, please contact the project team or open an issue on GitHub.