

Project Proposal

TransitGuard: Public Transport Violation Monitoring & Automated Priority Response System

Client: Ministry of Transport – Sri Lanka (Internship Project)

Prepared by: [Your Name] | [Intern ID / University]

Date: 10 February 2026

1. Executive Summary

Public transport complaints in Sri Lanka are currently reported through unstructured WhatsApp messages, making it hard to track, prioritize, and respond quickly—especially for high-risk violations like drunk driving or reckless driving. **TransitGuard** proposes a web-based complaint and case-management system that uses **QR codes (WhatsApp + web link)** to capture reports in a structured format and an **automated priority algorithm** to trigger **fast response** through a real-time Admin Command Center. The system includes geo-tagging, digital evidence uploads, ticket tracking, habitual offender detection, and route-level analytics to support enforcement and long-term improvements.

2. Background & Problem Statement

Current Situation

- Complaints arrive via WhatsApp in free-text form.
- No consistent structure (missing bus number, route, time, location, evidence).
- Admin teams manually sort messages → delays and missed urgent cases.
- No reliable tracking or status updates for passengers.
- No analytics to identify high-risk routes/buses.

Core Problem

A lack of structured data and prioritization prevents **timely action** on serious violations and weakens accountability.

3. Project Objectives

1. Provide a **QR-based reporting mechanism** that passengers can access instantly.
2. Capture complaints in a **structured format** (bus number, route, category, time, GPS location, evidence).
3. Automatically classify incidents by **severity and priority** for rapid response.

4. Enable an Admin Command Center to manage case workflow (Investigating → Warning → Fine → Suspension).
5. Flag repeated offenders and generate route-level insights for targeted enforcement.
6. Improve public trust through a **Reference ID** and status tracking.

4. Project Scope

In Scope

- Passenger Reporting Portal (mobile-first)
- QR Code access (WhatsApp deep-link + Web Form link)
- Evidence upload (photo/video)
- Priority-based dashboard (live feed)
- Case workflow + action templates
- Habitual offender rule (3 reports within 7 days)
- Route analytics charts
- Role-based admin access

Out of Scope (Phase 2 / Future Enhancements)

- Full integration with national licensing/permit systems
- Automated fine payment processing
- AI-based video/image analysis (optional later)
- Mobile app (web-first approach recommended)

5. Stakeholders & Users

- **Passengers (Public Users):** Submit incidents and track status
- **Moderators (Depot/Regional Officers):** Verify, investigate, issue warnings
- **Super Admin (Ministry/Head Office):** Manage users, templates, policies, escalations
- **Law Enforcement / Depot Teams:** Receive alerts for priority incidents

6. Proposed Solution Overview

TransitGuard consists of two major modules:

Module A: Passenger Reporting Portal (Front-End)

- Accessed via **QR code scan**
- Web form with:
 - Auto-captured date/time

- Bus registration no (mandatory), route no
 - Violation category (weighted priority)
 - Optional description
 - **Geo-tagging via browser GPS**
 - Evidence upload (photo/video)
 - Generates **Ticket Tracking ID** (e.g., REQ-2026-000105)
- Confirmation shown immediately and (optional) sent via WhatsApp channel

Module B: Admin Command Center (Back-End)

- Real-time “Live Action Feed” sorted by severity:
 - **Red (Priority 1)** – immediate action
 - **Orange (Priority 2)** – urgent follow-up
 - **Green (Priority 3)** – standard handling
- Incident workflow:
 - Investigating → Warning Sent → Fine Imposed → License Suspended
- “Habitual Offender” flag:
 - Same bus reported **3 times within 7 days**
- One-click action templates:
 - Standard warning letter/SMS/WhatsApp message
- Route analytics dashboard:
 - Violations by route, by category, by bus

7. Functional Requirements

7.1 Passenger Reporting Portal

FR-01: QR/URL access

FR-02: Auto-capture date/time

FR-03: Mandatory bus registration number, optional route number

FR-04: Violation selection (weighted categories)

FR-05: Browser-based GPS capture (with user consent)

FR-06: Upload evidence (images/videos)

FR-07: Generate unique tracking reference ID

FR-08: Status check page using reference ID

7.2 Admin Command Center

FR-09: Secure login and role-based access

FR-10: Live incident feed with severity labeling (Red/Orange/Green)

FR-11: Filter/search by bus number, route, date, category, status

FR-12: Update case status through defined workflow

FR-13: Habitual offender detection and flagging

FR-14: Action template generator (letters/SMS/WhatsApp)

FR-15: Analytics charts and export (CSV/PDF)

8. Priority-Based Algorithm (Fast Response Logic)

Severity Weights (Initial Model)

Category	Weight	Priority	Level	Dashboard Color	Target Response
Drunk Driving	100	P1		Red	Immediate
Reckless Driving	90	P1		Red	Immediate
Overloading	60	P2		Orange	Same day
Rude Behavior	30	P3		Green	Within 48 hrs

Priority Score (Example Logic)

Priority Score = **Category Weight** + **Evidence Bonus** + **Repeat Offender Bonus**

- Evidence Bonus: +10 if media attached
- Repeat Bonus: +20 if bus has ≥ 2 reports in last 7 days

This ensures the system automatically pushes critical incidents to the top and supports **rapid escalation**.

9. QR Code + WhatsApp Complaint Channel Design

To satisfy “QR code for WhatsApp” while keeping reporting structured:

Option A (Recommended): Dual-Channel QR (WhatsApp + Web Form)

- **QR 1:** Opens WhatsApp chat to official number with a prefilled message
 - Example: “I want to report a violation. Bus: [busNo]. Route: [route]. Please send reporting link.”
- **QR 2 (or button inside WhatsApp reply):** Sends user to the web form to capture GPS + evidence + category properly.

Why recommended: WhatsApp alone is unstructured; the web form ensures completeness while WhatsApp remains the entry channel.

Option B (Advanced / Phase 2): WhatsApp Business API Bot

- Passenger sends complaint in WhatsApp
- Bot asks guided questions (bus no, route, category) and creates a ticket automatically
- Bot returns **Reference ID** and status link

Note: This requires WhatsApp Business Platform approval and webhook integration.

10. System Architecture (High Level)

Architecture Style: MVC + layered services (scalable and maintainable)

Components

- **Web Front-End:** ASP.NET Core MVC views + Bootstrap 5 + JS (geo-location)
 - **Backend Services:** C# services for ticketing, prioritization, notifications, analytics
 - **Real-time Updates:** SignalR (for “Live Action Feed” without refreshing)
 - **Database:** Microsoft SQL Server
 - **Evidence Storage:** Azure Blob Storage (recommended) or secure object storage
 - **Hosting:** Render (prototype) → Azure (government-grade production)
-

11. Data Model (Summary)

Key tables/entities:

- **Incidents** (ticket ID, bus no, route, category, priority score, GPS, status, timestamps)
 - **EvidenceFiles** (ticket ID, file path, type, hash, upload time)
 - **Buses** (bus no, owner/operator, depot, flags)
 - **BusReportsHistory** (bus no, incident IDs, weekly count)
 - **Users** (admins/moderators, roles)
 - **ActionsLog** (status changes, template messages, who did what, when)
-

12. Non-Functional Requirements

- **Scalability:** MVC + services; supports thousands of concurrent reports
 - **Availability:** target 99.9% uptime with cloud hosting + monitoring
 - **Security:**
 - Role-based access (Super Admin vs Moderator)
 - Parameterized queries / ORM to prevent SQL injection
 - Input validation + file upload restrictions
 - Audit logs for admin activity
 - **Privacy:**
 - Collect minimal personal data
 - GPS only with consent
 - Secure storage for evidence and access control
-

13. Implementation Plan (Suggested Timeline – 8 Weeks)

Week	Activities	Deliverables
1	Requirements + UI wireframes + DB design	SRS + Wireframes + ERD
2	Build Passenger Portal (form + validation)	Working reporting UI
3	Evidence upload + GPS capture + ticket ID	End-to-end submission
4	Admin login + dashboard + filters	Admin Command Center v1
5	Priority algorithm + SignalR live feed	Severity sorting + real-time
6	Workflow + templates + habitual offender	Case handling complete
7	Analytics charts + exports + testing	Reports + QA results
8	Deployment + documentation + training	Final deployed system + user guide

14. Testing & Quality Assurance

- **Unit Tests:** services (priority scoring, offender detection)
- **Integration Tests:** submission → dashboard → workflow updates
- **Security Testing:** SQL injection checks, upload validation, auth testing
- **User Acceptance Testing (UAT):** ministry staff pilot on sample routes
- **Performance Testing:** load test for concurrent submissions

15. Risks & Mitigation

Risk	Impact	Mitigation
WhatsApp API approval delays	Delays bot automation	Start with WhatsApp deep-link + web form; add API later
Users deny GPS permission	Less accurate location	Allow manual landmark input as fallback
False/malicious reports	Noise in system	Add moderation workflow + repeat offender logic + evidence weighting
Large video uploads	Storage/cost issues	Limit size, compress, store on blob, enforce file type rules
Connectivity issues on buses	Form submission failures	Simple UI, retry mechanism, minimal data first

16. Expected Outcomes & KPIs

- Reduced response time for high-risk cases (P1)
- Increased completeness of complaints (bus no + route + location + evidence)
- Automatic identification of habitual offenders
- Route-level risk heatmap for enforcement planning
- Better transparency through ticket tracking

17. Conclusion

TransitGuard modernizes public transport complaint handling by combining **QR-based access, structured digital reporting**, and an **automated priority response system**. The solution supports fast intervention for dangerous violations, improves accountability through workflows and offender history, and provides analytics for strategic enforcement—aligned with Ministry of Transport operational needs and public safety objectives