Meteor Camera Alerting System – Technical Documentation

1. INTRODUCTION

This document describes the complete Meteor Camera Alerting System used to monitor distributed Raspberry Pi–based meteor camera stations. The system detects operational failures and sends automated email alerts via Mailjet, while providing a secure unsubscribe mechanism protected by Cloudflare.

The system is designed to be resilient, restart-safe, auditable, and configurable without code changes.

2. SYSTEM OVERVIEW

The system consists of several interacting components:

- Meteor camera stations publishing status data via MQTT
- A central MQTT broker (HiveMQ Cloud)
- The camera_monitor.py service
- A SQLite database (stations.db)
- HTML email templates
- Mailjet email delivery
- unsubscribe.py web service
- Cloudflare DNS and HTTPS termination

All timestamps are handled in UTC and all alert decisions are persistent across restarts.

3. DATA FLOW

1. Cameras publish MQTT messages under topics such as:
   meteorcams/<station>/camerastatus
   meteorcams/<station>/lastboot

2. camera_monitor.py subscribes to the MQTT broker.

3. Messages update in-memory state and trigger scenario logic.

4. Alerts are evaluated using both time-based logic and persistent alert_state.

5. Emails are sent via Mailjet if conditions are met.

6. Unsubscribe requests are handled securely via unsubscribe.py.

4. MQTT CONFIGURATION

The MQTT broker is HiveMQ Cloud using TLS on port 8883.

Key configuration settings:

- TLS enabled automatically for port 8883
- Authentication using username/password
- Wildcard subscription to meteorcams/#

Debug logging can display raw MQTT traffic when logging level is DEBUG.

5. DATABASE STRUCTURE

stations.db contains two key tables.

5.1 stations table

Fields:

• station (text, primary identifier)
• email (recipient address)
• unsubscribed (0/1)
• reboot (0/1)

5.2 alert_state table
Tracks alert escalation:
• station
• scenario
• first_sent_utc
• last_sent_utc
• send_count

This table ensures reminders and auto-unsubscribe persist across restarts.

6. ALERT SCENARIOS

Three independent alert scenarios are implemented.

6.1 Camera Status Down
Triggered when camerastatus=0 for longer than timeout_minutes.

6.2 Pi and Camera Silent
Triggered when no MQTT messages are received for silence_timeout_minutes.

6.3 Has Not Rebooted
Triggered when lastboot timestamp exceeds reboot_threshold_hours.

Each scenario:
• Sends an initial email
• Sends reminders every 24 hours (configurable)
• Auto-unsubscribes after a configurable number of repeats

7. EMAIL DELIVERY (MAILJET)

Mailjet is used for reliable email delivery.

Features:
- HTML email templates
- Optional BCC recipients
- Central sender identity
- Detailed error logging

Emails are only sent if:
- Station exists in database
- Station is not unsubscribed
- Template file exists

Templates are stored locally and loaded at runtime.

8. UNSUBSCRIBE SERVICE

unsubscribe.py is a Flask-based web service.

Features:
- HMAC-signed unsubscribe links
- Protection against forged requests
- Confirmation step before unsubscribe
- Updates stations.db directly

Cloudflare provides HTTPS termination and DNS routing.

9. LOGGING AND DEBUGGING

Logging is controlled via config.ini:

[logging]
level = DEBUG | INFO | WARNING | ERROR

Debug mode enables:
• MQTT RX logging
• TLS connection details
• Scenario evaluation traces

Logs are written to systemd journal for reliability.

10. CONFIGURATION FILE (config.ini)

All system behavior is controlled by config.ini.

Sections:
• [mqtt]
• [monitor]
• [mailjet]
• [logging]

This allows tuning without modifying Python code.

11. SYSTEMD SERVICE

camera_monitor.py runs as a systemd service.

Benefits:
• Automatic startup
• Restart on failure
• Centralized logging
• Resource isolation

The service runs continuously and is designed to recover cleanly.

12. SECURITY CONSIDERATIONS

Security is enforced through:
- TLS-encrypted MQTT
- HMAC-signed unsubscribe links
- Cloudflare HTTPS
- No secrets stored in templates
- Environment variable for unsubscribe secret

13. MAINTENANCE AND OPERATIONS

Recommended practices:
- Regular database backups
- Monitoring unsubscribe counts
- Testing templates after edits
- Verifying Mailjet sender domain
- Periodic DEBUG logging during upgrades

14. EXTENSIBILITY

The system is designed for expansion:
- Additional alert scenarios
- Per-station thresholds
- Daily summary emails

• Web-based admin console
• SMS or push notification integration

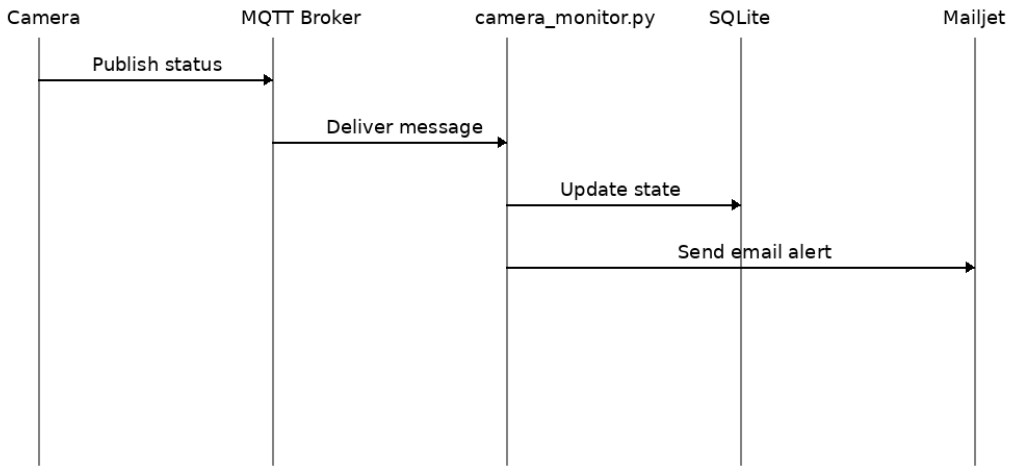
15. CONCLUSION

This alerting system provides a robust, auditable, and scalable solution for monitoring distributed meteor camera infrastructure. Its emphasis on persistence, clarity, and operational safety makes it suitable for long-term unattended operation.

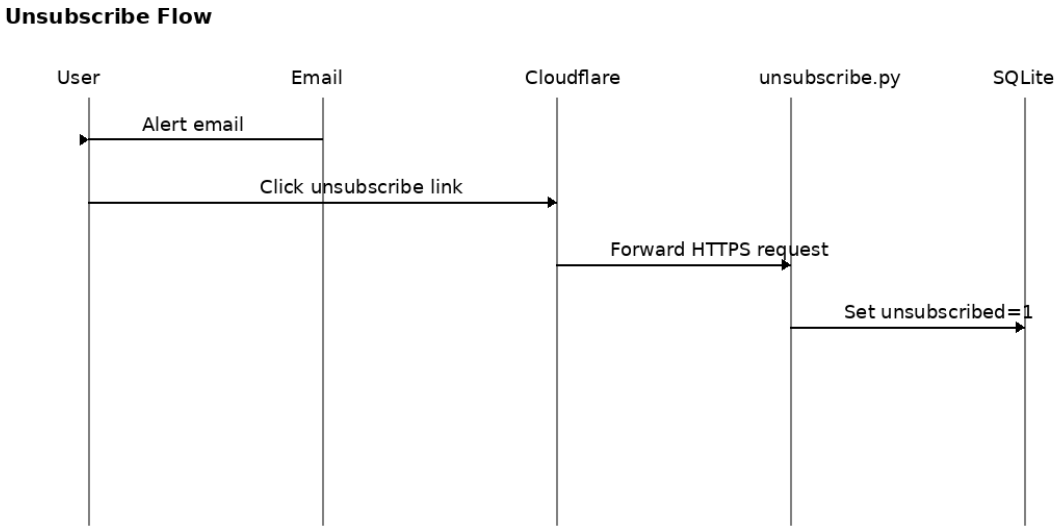
Appendix A – Sequence Diagrams

A.1 MQTT Alert Detection

MQTT Alert Detection Flow



A.2 Unsubscribe Process



Appendix B – Operational Screenshots

The following screenshots should be captured during operation and inserted here:

- systemctl status camera-monitor
- journalctl -u camera-monitor.service showing DEBUG output
- Example Mailjet delivered email
- Unsubscribe confirmation page via Cloudflare

These screenshots document correct system behaviour and are useful for audits and handover.