**Premonitor: A Comprehensive Technical and Business Manual**

**Author’s Perspective:** As a serial entrepreneur, embedded systems developer, and AI software architect with deep experience in building sensor-based products, this document consolidates the full vision, engineering blueprint, and commercial roadmap of the Premonitor system. All content has been audited and expanded based on historical design discussions to ensure full technical, behavioral, and architectural coverage. This version is now fully implementation-ready, suitable for development, funding, and deployment.

### 1. **Project Summary and Purpose**

**Premonitor** is an AI-driven, multi-sensor IoT system that proactively prevents environmental damage in high-value and sensitive areas—starting with laboratories and scientific storage.

**Key Goals:** - Prevent water, fire, gas, and chemical-related incidents - Protect laboratory and academic research samples - Enable voice, visual, and remote alerting - Create an autonomous agentic AI layer that anticipates danger

### 2. **Hardware Architecture**

#### A. **Main Premonitor Device**

* **Board**: Raspberry Pi (tested on 1GB RAM)
* **Power**: USB supply or UPS battery bank
* **Enclosure**: Designed for passive airflow, camera visibility, and sensor exposure

**Integrated Sensors:** - **Microphone**: Audio sampling to detect patterns like dripping, hissing, smashing (glass break) - **MQ Gas Sensor**: Detects VOCs, smoke, gas (e.g., methane) - **Thermal Camera**: Scans pixelated heat maps for fire or overheating - **RGB Camera**: Captures snapshots during anomalies - **PIR Motion Sensor (optional)**: Detects movement - **Air Quality Sensor (planned)**: Measures oxygen (O₂), CO₂, and general AQI (e.g., using CCS811, SCD41 or BME688) - **LED Output**: WS2812B or DotStar strip; animated alerts - **Speaker**: Voice and siren feedback - **Reset/Test Buttons**

**Connectivity:** - Wi-Fi (provisioned via local AP + captive portal) - BLE (connects to external stick-on sensor securely)

**LED Behavior (Custom Defined):** - **Red Pulse**: Critical alert (fire, gas, or toxic air) - **Orange Blink**: Warning (vibration/door left open/high CO₂) - **Green Idle**: All clear - **Rainbow Chase**: Power-on test/startup mode - **Blue Slow Pulse**: BLE sync event

#### B. **Stick-On BLE Sensor**

**Purpose**: Mounted externally to fridges/freezers, with internal probe

**Design:** - Main body attaches magnetically or with adhesive to the outside - Probe (housing DHT20 or waterproof analog probe) inserted via gasketed pass-through inside fridge - Total size: approx. 40mm × 50mm × 18mm

**Components:** - ESP32-C3 or nRF52 MCU (BLE + ultra-low-power) - Sensors: - **Temperature & Humidity**: DHT20 - **Vibration Sensor**: SW-420 - **Reed Switch**: Magnetic door detection - **Battery**: 3.7V 500mAh LiPo + supercapacitor - **Energy Recovery**: Wi-Fi RF harvester to prolong life

**Power Calculations:** - BLE sync: ~12mA for 1.5s - Deep sleep: ~10–15µA - Harvested Wi-Fi energy: ~10–50µW

### 3. **Software Architecture (Python Modular System)**

Modules include: - microphone.py: FFT + pattern match - thermal\_camera.py: Heatmap analysis - gas\_sensor.py: Analog level tracking - air\_quality.py: CO₂/O₂ sensor input (planned) - camera.py: Image capture - motion.py: PIR input - ble\_receiver.py: BLE packet processing - alerts.py: Evaluates all conditions - led\_output.py: Visual feedback - speaker\_output.py: Voice warnings - cloud\_sync.py: Uploads alerts/logs - config.py: Editable thresholds and phrases

### 4. **Firmware (BLE Sensor)**

* Language: C/C++ (ESP-IDF or Arduino)
* BLE Secure Connection
* Deep sleep cycles
* Event-driven wake (magnet or vibration)
* Payload: temp, humidity, door, vibration, timestamp

### 5. **Alert System Flow**

**Stage 1:** LED flash, voice, image **Stage 2:** Remote sync via webhook/email/SMS **Stage 3:** “Stabilized” voice + green LED

**Test Mode:** Cycles LED + voice check

### 6. **AI and Machine Learning**

* **Sound**: 1D CNN anomaly detection
* **Thermal**: Classification of patterns
* **Air Quality**: Historical profiling and alerts
* **BLE Sensor**: Behavior profiling per fridge

**Agentic AI Vision:** - Self-mitigating suggestions or actions - Redundant protection logic

### 7. **Deployment Workflow**

* Attach BLE module and insert probe
* Power main device and provision network
* Confirm BLE connection
* Set thresholds and webhook links

**Battery Check:** BLE sensor can issue low-power alert

### 8. **Academic Use**

* Used in CMP9782 project module
* Included A0 poster with system diagrams, photos, alert flow
* Emphasized premitigation in lab storage

### 9. **Business Strategy**

* Markets: Universities, medical storage, homes
* Revenue: Hardware + subscription + licenses
* Expansion: Server rooms, greenhouses, museums

### 10. **Brand Identity**

* **Logo**: Eye inside shield
* **Tagline**: “Detect early. Prevent damage. Protect knowledge.”
* **Interface**: Minimal, LED-based, voice feedback

### 11. **Risks & Mitigations**

* BLE interference → retry + mesh
* Probe fog → waterproofing
* Low battery → supercap + alerts
* Privacy → camera deactivation modes

### 12. **Roadmap**

* Phase 1: Prototypes
* Phase 2: BLE rollout
* Phase 3: AI tuning
* Phase 4: Lab pilots + fundraising

**Prepared by:**  
Alexandru Emanuel Vasile  
Entrepreneur | AI & IoT Systems Architect | Embedded Product Developer