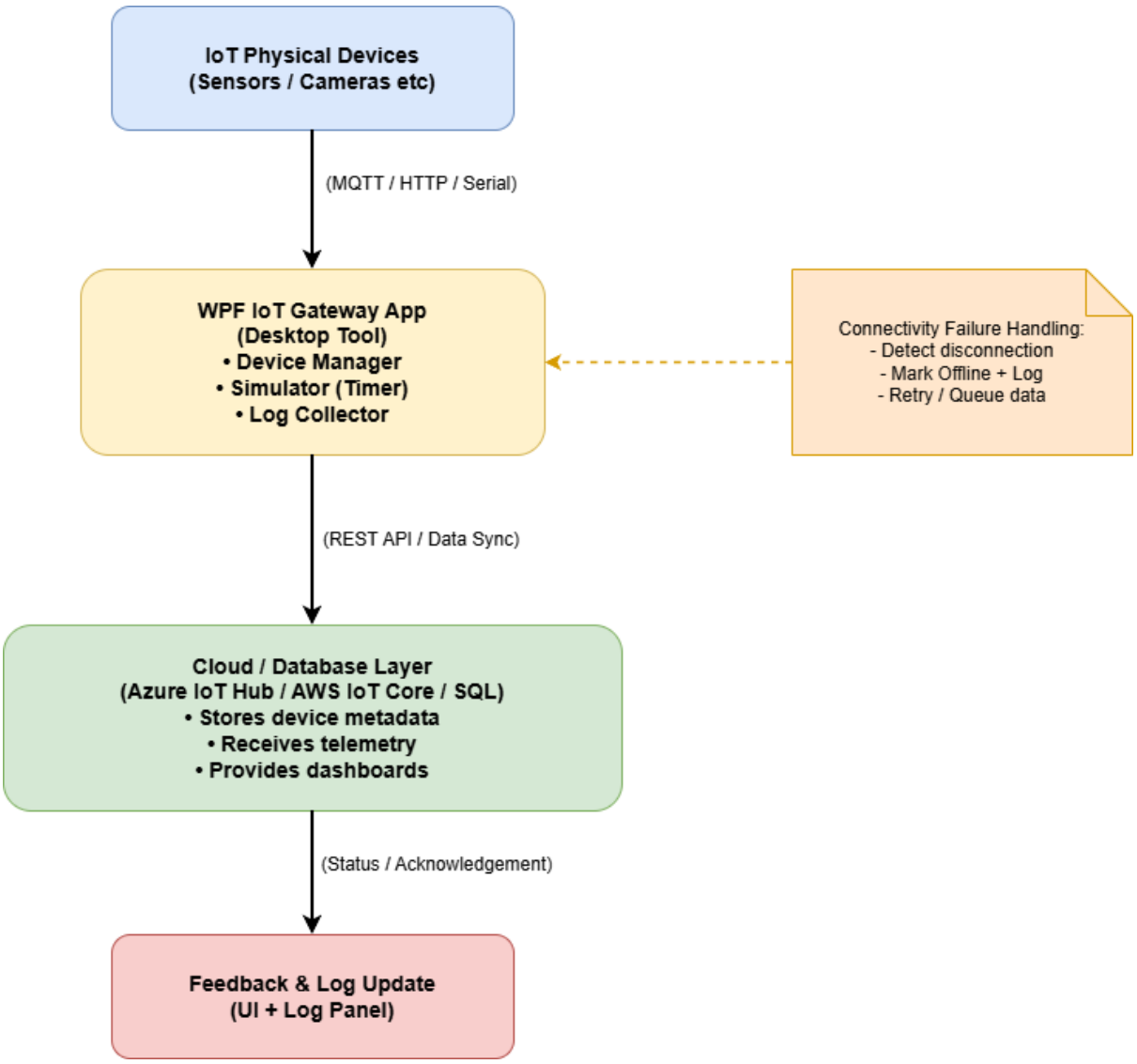
System Architecture

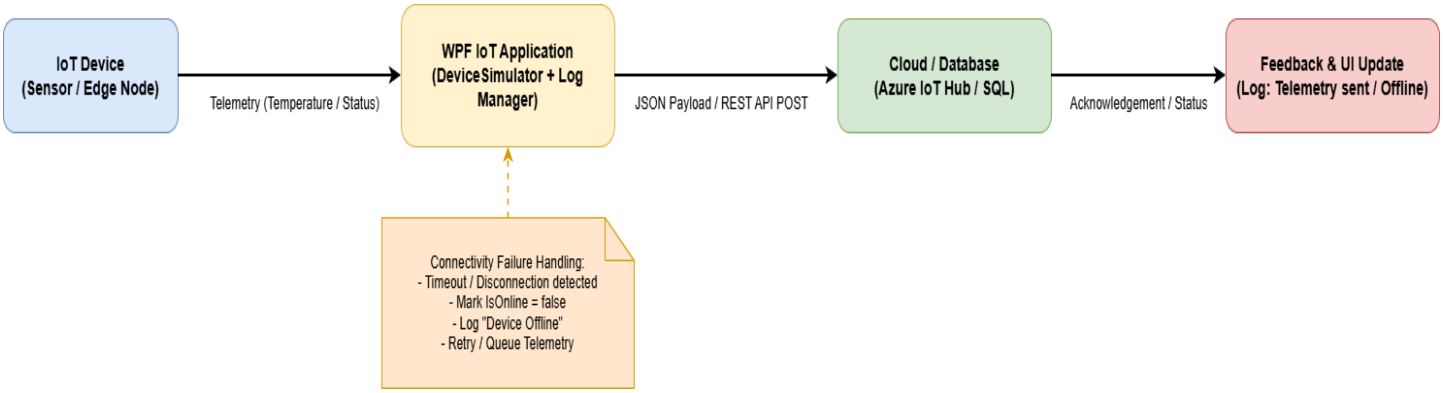


The IoT Device Manager system follows a layered architecture connecting simulated IoT devices to a cloud or database layer via a WPF desktop application.

- **Device Layer:** Physical or simulated IoT nodes (sensors, cameras) produce telemetry data periodically.
- **Gateway Application Layer:** The WPF app acts as a gateway and manager. In this project, DeviceSimulator.cs emulates telemetry generation and device connectivity using timer events. The MainViewModel processes incoming telemetry, updates device status, and logs events in real time.
- **Cloud / Database Layer:** Stores metadata, telemetry readings, and system logs. In a real-world deployment, this would correspond to Azure IoT Hub, AWS IoT Core, or a SQL-based backend.
- **Feedback Layer:** The application UI updates logs and device panels based on connection status or telemetry received, giving immediate feedback to the user.

Connectivity failure is handled through disconnection detection, device status marking (IsOnline = false), logging, and simulated retry logic.

Data Flow



Data Flow Description

1. **IoT Device → WPF App:** Sends telemetry data (temperature, humidity, status) periodically.
2. **WPF App → Cloud:** The app transmits the data as JSON payloads (REST API POST or MQTT Publish).
3. **Cloud → App Feedback:** The cloud responds with acknowledgment or error codes.
4. **App → UI Feedback:** The WPF UI updates logs (e.g., “Telemetry sent” or “Device Offline”) in real time.
5. **Connectivity Handling:** If communication fails, the system logs the disconnection, sets the device offline, queues unsent data, and retries transmission after a delay.