

Smart Lock IoT System

An Enterprise-Grade Hybrid Architecture

Name

Suriyabandara S.M.M.

Ravimindu P.H.G.

Keembiyage J.S.

Index

210626L

220530V

220319H

The Challenge: Bridging Physical and Digital Security



The Problem

- ❖ Physical keys have no digital logs or audit trail.
- ❖ Keys are easily lost, duplicated, or stolen.
- ❖ No remote access or real-time monitoring capabilities.
- ❖ No immediate alerts for unauthorized access attempts.



Our Solution

- ❖ Keypad-based authentication with digital event logging.
- ❖ Real-time tamper detection via vibration sensors.
- ❖ Cloud-hosted dashboard for remote monitoring and control.
- ❖ Enterprise-grade security with Azure IoT Hub integration.

Core Objectives: Building a Secure and Scalable IoT Solution

1

Secure Hardware Prototype

Develop a reliable smart lock system using Arduino UNO and Raspberry Pi with real-time control and safety mechanisms.

2

Real-Time Web Dashboard

Implement a cloud-hosted Node-RED dashboard for live monitoring, activity logging, and remote lock/unlock control.

3

Proactive Alerting

Integrate tamper detection via vibration sensors and failed authentication alerts with instant notifications.

4

Full-Stack Integration

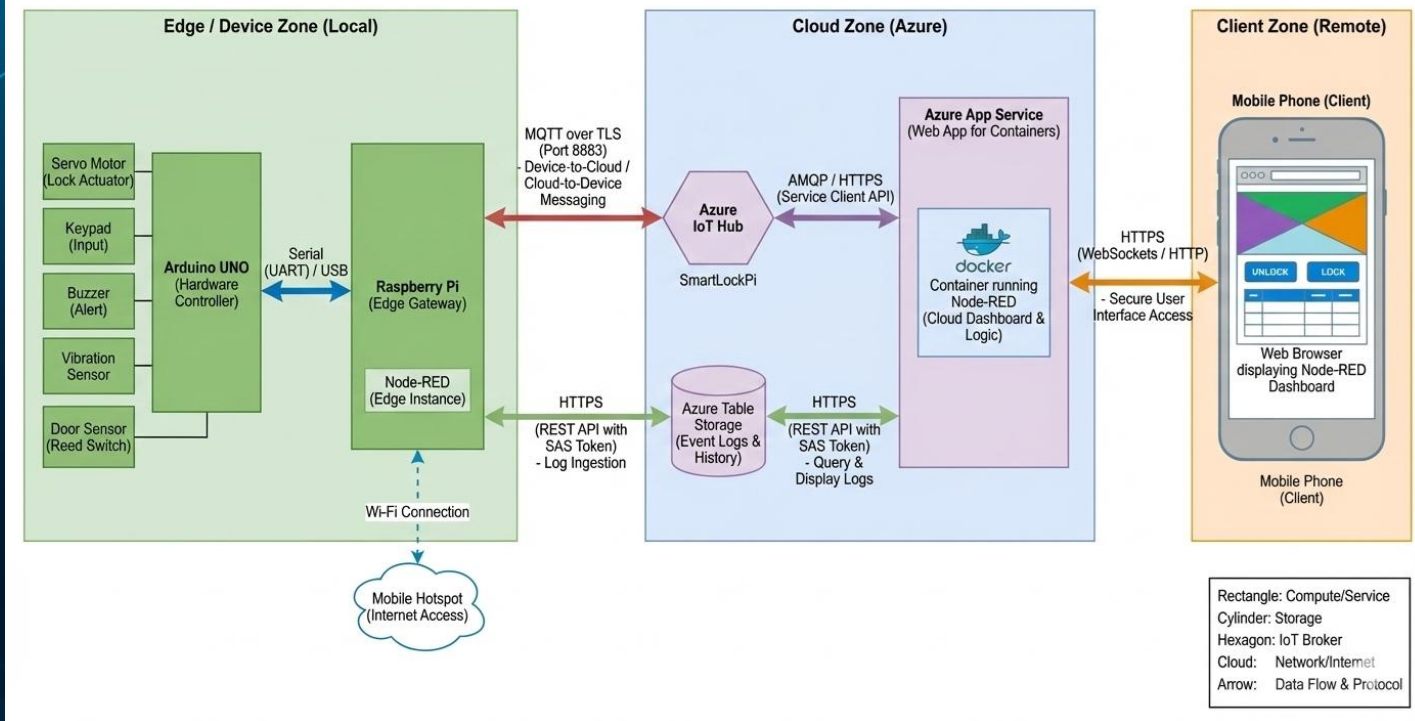
Demonstrate seamless integration of edge devices, cloud services (Azure IoT Hub), and data persistence (Table Storage).

Project Scope

This project demonstrates a complete IoT solution spanning hardware control, edge computing, cloud integration, and secure data management. It validates enterprise-grade architectural patterns for IoT systems.

3-Layer Architecture

Smart Lock System Architecture - Detailed Diagram



🔒 Safe Edge Gateway Pattern

RPI isolates the Arduino from direct internet access, handling TLS and MQTT. The Arduino stays focused on real-time hardware control, keeping lock logic protected even if cloud connectivity is lost.

Layer 1: Device Zone - The Physical Edge (Circuit)

Controller

Arduino UNO

Hardware Components

Keypad (4x4)

Vibration Sensor (SW-420)

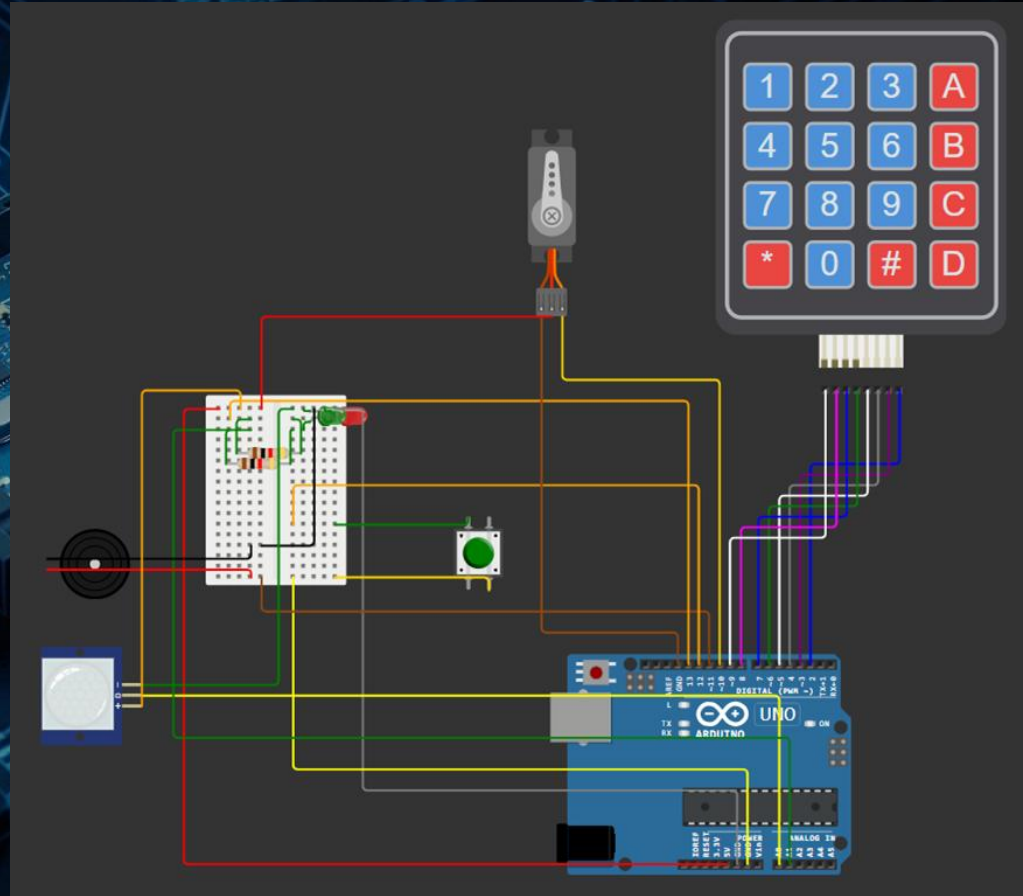
Push Button as the Door Sensor

Servo Motor

Buzzer

IoT Principle: Constrained Device

The Arduino cannot handle heavy TLS or MQTT; the RPi gateway manages network complexity while the Arduino focuses on deterministic hardware control.



Layer 1: Device Zone - The Protocol Converter (RPI Logic)

Primary Role

Protocol Converter & Edge Gateway

The Raspberry Pi bridges the gap between the constrained Arduino device and the cloud, handling protocol conversions and security operations.

Input

Serial Data (JSON)

/dev/ttyACM0

From Arduino UNO

Processing

Hot Path: Convert data to MQTT format

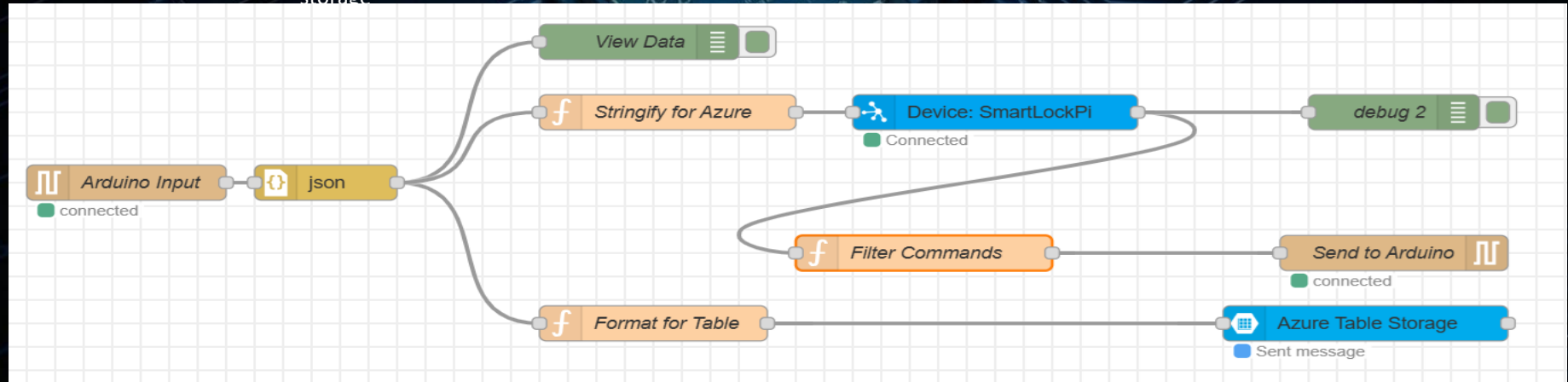
Cold Path: Formats data for REST API insertion into the table

Output

To Cloud: MQTT over TLS(Port 8883) -> Azure IoT Hub

To Table Storage: HTTPS REST API -> Azure Table Storage

storage



Layer 2: Cloud Zone - Azure IoT Hub & Azure Table Storage

Azure IoT Hub: The Secure Broker

Role: Acts as the central message broker for secure, bi-directional communication between the Edge Gateway and the Cloud.

Device Identity: Manages a Device Identity Registry to ensure only authorized devices (like SmartLockPi) can connect, preventing spoofing attacks.

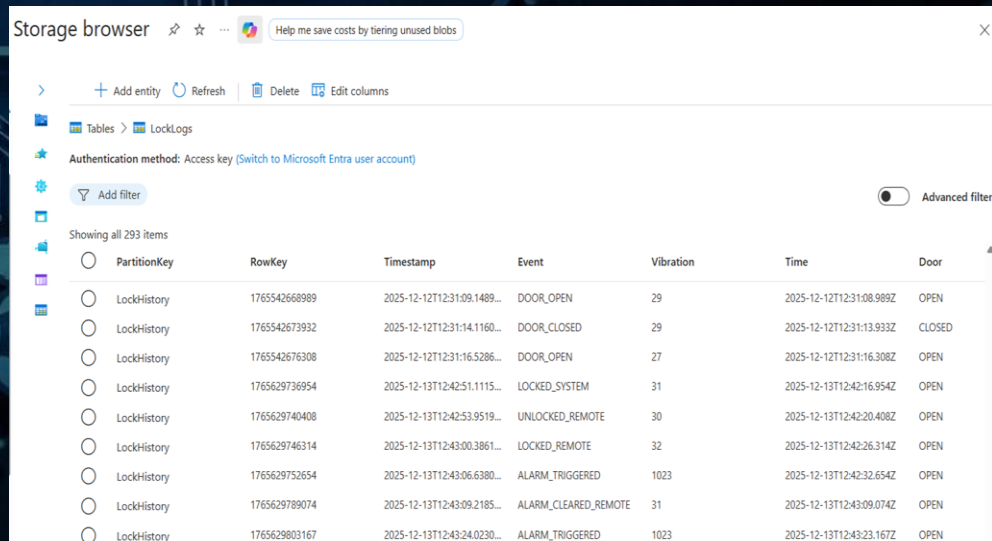
Function: Handles MQTT Telemetry for "Hot Path" connectivity and Cloud-to-Device (C2D) messages for remote control commands.

Azure Table Storage: The Data Warehouse

Role: Provides cost-effective, persistent storage for historical event logs.

Why NoSQL?: Optimized for storing massive amounts of unstructured time-series data (Timestamp, Event, Value) without the overhead of complex SQL relationships.

Security: Accessed via SAS Tokens (Shared Access Signatures), enforcing the "Principle of Least Privilege" by granting only necessary permissions.



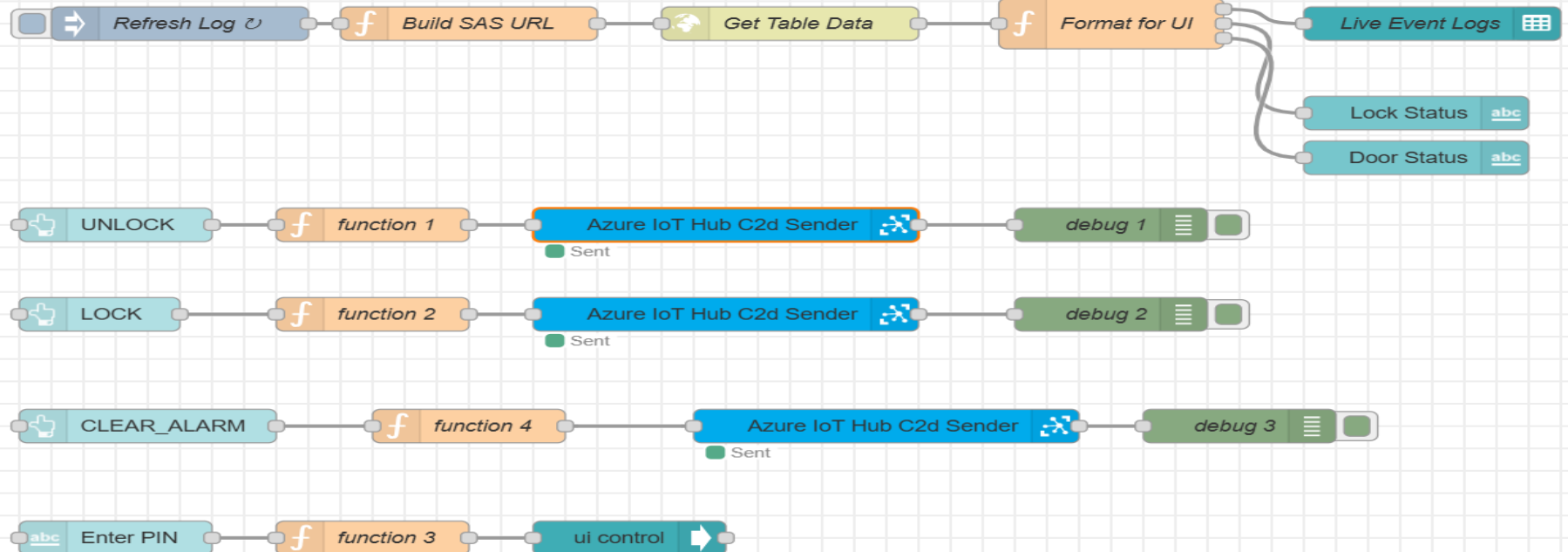
The screenshot shows the Azure Storage browser interface. At the top, there's a header with 'Storage browser' and a help link. Below the header, there are navigation icons and a search bar. The main content area shows a table of lock events. The table has columns for PartitionKey, RowKey, Timestamp, Event, Vibration, Time, and Door. The data is filtered to show 293 items. The table is sorted by RowKey in ascending order. The events include DOOR_OPEN, DOOR_CLOSED, DOOR_OPEN, LOCKED_SYSTEM, UNLOCKED_REMOTE, LOCKED_REMOTE, ALARM_TRIGGERED, and ALARM_CLEARED_REMOTE.

PartitionKey	RowKey	Timestamp	Event	Vibration	Time	Door
LockHistory	1765542668989	2025-12-12T12:31:09.1409...	DOOR_OPEN	29	2025-12-12T12:31:08.989Z	OPEN
LockHistory	1765542673932	2025-12-12T12:31:14.1160...	DOOR_CLOSED	29	2025-12-12T12:31:13.933Z	CLOSED
LockHistory	1765542676308	2025-12-12T12:31:16.5286...	DOOR_OPEN	27	2025-12-12T12:31:16.308Z	OPEN
LockHistory	1765629736954	2025-12-13T12:42:51.1115...	LOCKED_SYSTEM	31	2025-12-13T12:42:16.954Z	OPEN
LockHistory	1765629740408	2025-12-13T12:42:53.9519...	UNLOCKED_REMOTE	30	2025-12-13T12:42:20.408Z	OPEN
LockHistory	1765629746314	2025-12-13T12:43:00.3861...	LOCKED_REMOTE	32	2025-12-13T12:42:26.314Z	OPEN
LockHistory	1765629752654	2025-12-13T12:43:06.6380...	ALARM_TRIGGERED	1023	2025-12-13T12:42:32.654Z	OPEN
LockHistory	1765629789074	2025-12-13T12:43:09.2185...	ALARM_CLEARED_REMOTE	31	2025-12-13T12:43:09.074Z	OPEN
LockHistory	1765629803167	2025-12-13T12:43:24.0230...	ALARM_TRIGGERED	1023	2025-12-13T12:43:23.167Z	OPEN

Layer 2: Cloud Zone - Azure App Service

Deployment Architecture

Node-RED application logic runs inside a Docker container deployed on Azure App Service, enabling portability, scalability, and high availability.



User Interface: Real-Time Monitoring and Remote Control

Dashboard Components

Lock Status Indicator

Real-time display of lock state (Locked/Unlocked) with color-coded visual feedback.

Door State Monitor

Live status of door open/closed state from magnetic sensor telemetry.

Remote Control Panel

Interactive buttons to send lock/unlock commands directly to the device via cloud.

Activity Log Table

Historical record of all lock events: authentication attempts, door open/close, tamper alerts.

Smart Lock

Lock

UNLOCK

LOCK

CLEAR_ALARM

Status

Lock Status

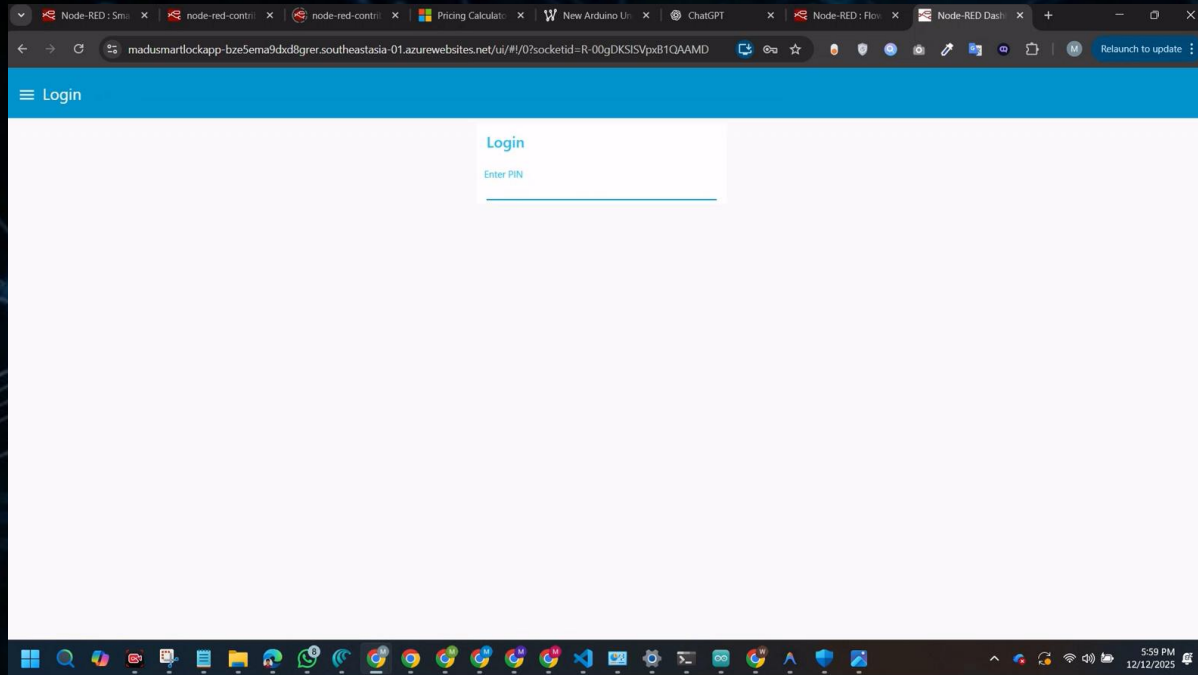
Door Status

LOCKED

OPEN


Table

Time	Event	Vibration
2025-12-13T12:43:30.652Z	ALARM_CLEARED_REMOTE	29
2025-12-13T12:43:23.167Z	ALARM_TRIGGERED	1023
2025-12-13T12:43:09.074Z	ALARM_CLEARED_REMOTE	31
2025-12-13T12:42:32.654Z	ALARM_TRIGGERED	1023
2025-12-13T12:42:26.314Z	LOCKED_REMOTE	32
2025-12-13T12:42:20.408Z	UNLOCKED_REMOTE	30
2025-12-13T12:42:16.954Z	LOCKED_SYSTEM	31



Team MG

Suriyabandara S.M.M. 210626L
Ravimindu P.H.G. 220530V
Keembiyage J.S. 220319H

 **Key Insight:** All actions are logged in Azure Table Storage and visible in the real-time dashboard, demonstrating full-stack IoT integration.

Scan The QR - Access the Dashboard



Password: 1234



The Backwards Flow

1. Web Browser

HTTPS / WebSocket

2. Azure App Service

Node-RED Dashboard Logic

3. Azure IoT Hub

C2D Message (Cloud-to-Device)

4. Raspberry Pi Gateway

MQTT Subscription (TLS Port 8883)

5. Arduino UNO

Serial UART / USB (/dev/ttyACM0)

6. Servo Motor

PWM Control Signal

Security Architecture: Three Pillars of Protection

1

Device Identity

Each RPI has a unique identity registered in Azure IoT Hub's Identity Registry.

Prevents spoofing attacks by validating device credentials.

Only authenticated devices can publish telemetry.

2

Access Control

SAS Tokens enforce Principle of Least Privilege for database access.

Time-limited tokens that expire automatically.

Each token grants only necessary permissions (read/write logs).

Prevents privilege escalation and lateral movement.

3

Physical Security

Arduino is air-gapped from the internet, isolated by the RPI gateway.

Cannot be hacked directly from the web.

All network communication goes through RPI.

Lock logic remains functional even if cloud is compromised.

Overall Security Posture

This system uses multiple levels of protection, called **defense-in-depth**, to keep the lock secure. First, it confirms the identity of the device trying to access it, and then it only grants a temporary, **time-limited digital key**. Crucially, the physical lock mechanism, run by an **Arduino**, is physically isolated ("air-gapped"). This means that even if the network or cloud security is breached, the lock remains **functional and protected** because its core mechanism is disconnected from those digital systems.

Performance Metrics and System Constraints

Local Latency

< 500ms

Keypad entry to servo actuation. Real-time response with zero network dependency.

Cloud Latency

< 2s

Dashboard button press to servo actuation via Azure IoT Hub and MQTT.

System Constraints & Reliability

Constrained Device Architecture

Arduino UNO lacks computational resources for TLS encryption and MQTT protocol. RPi Gateway is mandatory for secure cloud communication.

Network Resilience

Local lock control operates independently. If cloud connectivity is lost, the system defaults to local keypad-only mode with full functionality.

Connected Status Monitoring

Node-RED continuously monitors RPi-to-IoT Hub connection state. Dashboard displays real-time connectivity status to users.

Implementation Deviations and Justification

Database Migration

Implemented Azure Table Storage instead of MariaDB as originally proposed.

Reason

Cost-Effectiveness for Time-Series Data. Azure Table Storage is a NoSQL structure optimized for fast writes and simple log data (Timestamp, Event, Value) without complex JOINS, significantly cheaper than a traditional SQL database like MariaDB for massive event history.

Cloud Broker Integration

Integrated Azure IoT Hub instead of a simple, self-hosted MQTT broker.

Reason

Enterprise-Grade Device Management & Security. IoT Hub provides an Identity Registry to prevent spoofing attacks and is built for scalability, demonstrating a more robust and secure architecture than a simple, self-hosted MQTT broker.

Database Migration

Implemented Azure Table Storage instead of MariaDB as originally proposed.

Reason

High Availability & Decoupling. Hosting the dashboard in the cloud ensures accessibility from anywhere and decouples the UI from the physical device. If the RPi loses power, the dashboard remains available with the last known status.

Data Persistence - Cost-Effective NoSQL for Telemetry

Original Proposal

MariaDB (SQL)

Actual Implementation

Azure Table Storage (NoSQL)

Storage

Redundancy: ⓘ

LRS

10 × \$0.115 = \$1.15

GB Per GiB

MariaDB (SQL)

Type:

Table Storage

Tier:

Standard

Redundancy: ⓘ

LRS

Capacity

10 GB × \$0.0450 = \$0.45

Per GiB

Azure Table Storage (NoSQL)

Nature of Data

Our data is simple time-series telemetry: Timestamp, Event Type, Value. This is unstructured log data that rarely requires complex relational JOINS or transactions.

Cost-Effectiveness

NoSQL is significantly cheaper for massive event history storage. Pay-per-transaction pricing vs. per-instance SQL Database costs.

Scales horizontally without expensive database upgrades.

Performance Benefits

Optimized for fast writes and key-based retrieval.

No schema migration overhead for evolving telemetry formats.

Future Development

Event-Driven Recording: A CCTV camera will be strategically positioned to cover the door, only initiating power and recording upon the detection of a significant door **vibration event**.

How it Works ?

The camera will be offline until the vibration sensor detects any kind of a vibration and if a vibration occurred the cctv will automatically powered and it also sends a video clip of pre-defined time period to the database.

Benefits

- Power Efficiency
- Storage & Cost Management



Reasons for Some Architectural Decisions

- ❖ Why not ESP 32?

If connection loses , the lock is locked/ feezed while trying to reconnect.

- ❖ Why not ESP 32 instead of RPi?

Handling cloud connection & managing complex certificates and SAS tokens is computationally heavy.

- ❖ Why MQTT between RPi & IoT hub?

Lightweight & low power usage & data overhead is less.

- ❖ Why QoS 01?

To ensure that commands & alerts are guaranteed to be delivered.

Azure IoT hub does not standardly support QoS 2.

- ❖ Why cold path for UI?

User experience.

If we used the hot path , it may show two different things in the table & the status Section.





THANK YOU

MULTIPURPOSE PRESENTATION TEMPLATE