Smart Access Control: Leveraging IoT Infrastructure and Machine Learning for Secure Authentication Systems

Individual Report - Mobile & Ubiquitous Computing

Yash Nimbalkar - 2751317

May 2025

Contents

List of Figures

List of Tables

Abstract

Traditional access control systems, such as keypad-based solutions and Radio Frequency Identification ($RFID$), have become less adaptable in order to satisfy the demands of modern technology and are more vulnerable to security breaches. In order to provide a smarter, more dependable and user-centric option for residential and commercial access control, we will focus on a Smart Access Control System that combines Internet of Things (IoT) technology, machine learning and secure communication protocols.

The system is made up of three interconnected components: a Presence Detection Module with motion sensors, a Face Recognition Unit consisting of ESP32-Cam [1] with an advanced camera and edge processing capabilities and a Smart Lock & Alert Unit with a servo motor and real-time alerts. The MQTT protocol is used for secure communication between these components and a mobile application with interfaces customised for four different user roles controls them.

Machine learning models will be operating both at the edge and in the cloud; these will drive facial recognition and detect unusual access patterns, thus enhancing the overall security and responsiveness. Unlike traditional RFID or keypad-based systems, the proposed solution addresses modern challenges in the currently used AI-integrated IoT systems, including data privacy, edge computing limitations, energy efficiency and network reliability. We will be mitigating these problems by implementing encrypted communications, optimised on-device processing, energy-efficient protocols and adaptive failover mechanisms.

This report will focus on the overall system's architectural framework, user experience design, hardware-software integration, machine learning pipeline development and comprehensive evaluation metrics for the ML model. The resulting solution demonstrates a secure, efficient and scalable approach to next-generation access control, effectively overcoming the limitations of conventional systems while also addressing currently emerging technological vulnerabilities.

1 Application Domain and Problem

1.1 Application Domain:

Access control is necessary for securing residential and commercial properties. Traditional access control systems such as physical keys, keycards and passcodes have long been the standard for authenticating and regulating entry. However, these solutions often lack the flexibility, scalability and adaptability required to address modern security threats. As smart devices become more popular, there is an increasing chance to create secure, intelligent and context-aware access control systems that can adjust to the changing needs of modern environments.

1.2 Problem Statement:

Traditional access control systems suffer from several critical vulnerabilities that lower their effectiveness in highly secure environments. They are often exposed to theft and duplication of credentials and they lack biometric identity verification thereby increasing the risk of impersonation and offer little to no real-time monitoring or situational awareness. Additionally, these systems provide limited options for remote control or user personalisation and are incapable of detecting odd behaviour or adapting to individual usage patterns. In locations like office buildings, data centres and residential complexes where strict access regulation is absolutely necessary, these limitations pose serious security problems.

1.3 Why It's Worth Solving:

As physical security threats grow increasingly complicated, there is an urgent need to adopt smarter and more proactive access control systems. Leveraging Internet of Things (IoT) and Machine Learning (ML) technologies enables us to shift from traditional, reactive and manually managed systems to dynamic and intelligent platforms. These advanced systems can automatically recognise users, learn behaviour patterns over time, detect and flag anomalies and provide granular, remote access control and these all contribute to a more secure and adaptive environment.

1.4 Stakeholders Involved:

1. **Primary Users(*Residents/Employees*):** Require seamless, secure access to premises
2. **Visitors/Guests:** Require temporary, verified access approved by primary users
3. **Technicians/System Admins:** Maintain, configure and audit system health
4. **Security Personnel:** Require tools to monitor access attempts and respond to threats

1.5 Existing Solutions and Limitations:

Current solutions which include RFID badges, numeric keypads and biometric scanners, often operate in isolation, lacking integration, personalization or data-driven intelligence. While some modern systems do implement IoT and ML, they face significant challenges. These systems commonly encounter data exposure risks due to inadequate or weak encryption protocols. Additionally, they struggle with limitations in training data for machine learning

models, which affects recognition accuracy and system reliability. The constraints of edge devices, specifically processing power, memory capacity and battery life, further restrict system capabilities and performance. Finally, these solutions typically demonstrate high dependency on stable network connectivity, creating potential points of failure in environments with unreliable connections [2].

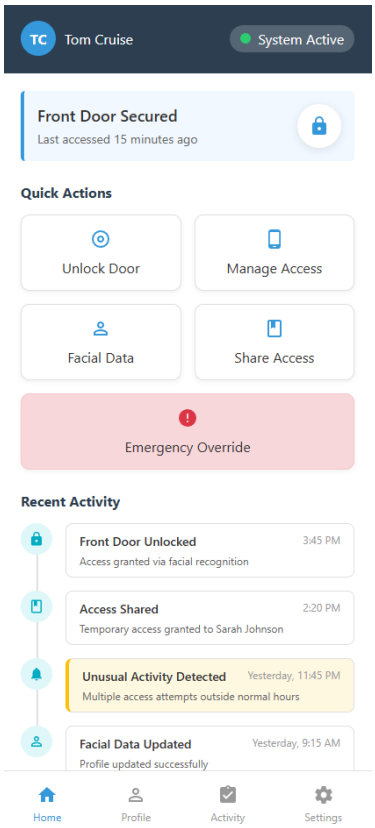
The proposed Smart Access Control System addresses these with encrypted communication, optimised edge processing, energy-efficient protocols and adaptive failover, overcoming traditional and emerging issues.

2 Smartphone Application and User Interaction

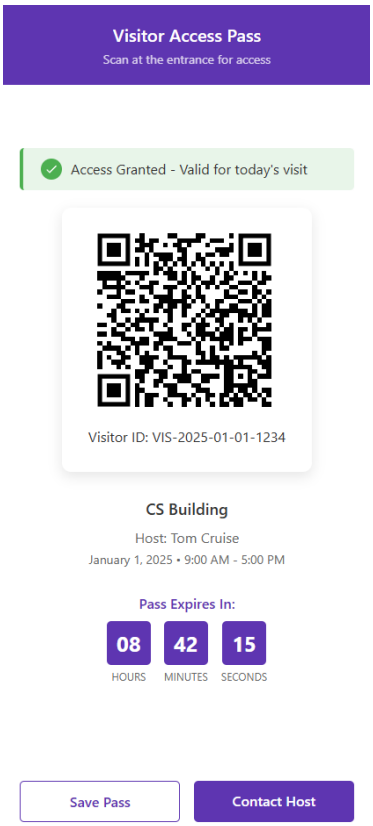
The smartphone application is the primary interface for the Smart Access Control System, leveraging role-based access control ($RBAC$) to deliver secure, personalised and intuitive experiences for all stakeholders while addressing modern IoT and AI challenges.

2.1 Stakeholder Roles & Interactions

1. **Primary Users (*Residents / Employees*): (see Fig. 1a)**
   - **Functionalities:**
     - Securely register and manage facial biometric data
     - Unlock doors via facial recognition or app-based override
     - View access logs and receive real-time security alerts
     - Grant temporary access via QR codes or tokens for visitors
   - **UI Features:**
     - Dashboard displaying lock status and recent activity
     - Facial enrollment with live preview and privacy safeguards
     - Access sharing with calendar-based time slots
     - Emergency override button and anomaly detection alerts
2. **Visitors: (see Fig. 1b)**
   - **Functionalities:**
     - Receive time-bound access tokens or QR codes from users
     - Authenticate at entry points via token/QR scan
     - Get reminders and expiration alerts
   - **UI Features:**
     - Minimalist interface showing access credentials
     - Notifications for access status
     - Countdown timer for token validity
3. **Technicians/System Administrators: (see Fig. 2a)**
   - **Functionalities:**
     - Configure IoT devices remotely



(a) Resident Interface



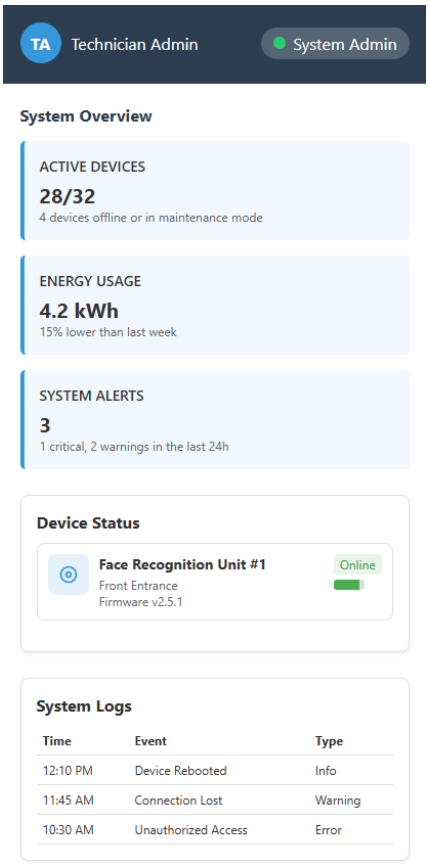
(b) Visitors Interface

Figure 1: User Interface Mockups - Resident & Visitors

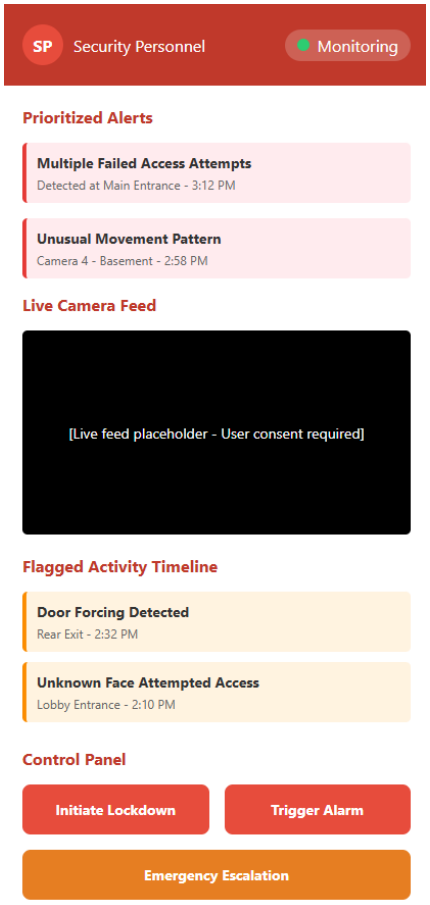
  - Monitor system health (*battery, connectivity, firmware updates*)
  - Access audit logs and perform security checks
- **UI Features:**
  - Admin dashboard with Green/Yellow/Red health indicators
  - Diagnostics panel showing signal strength and energy metrics
  - Log export option and system reboot tools

4. **Security Personnel: (see Fig. 2b)**

- **Functionalities:**
  - Receive prioritized alerts for suspicious events (*e.g., repeated failed attempts*)
  - Live camera feed (*if permitted by user consent*)
  - Initiate lockdowns, alarms, or emergency escalation
- **UI Features:**
  - Incident dashboard with threat level indicators
  - Timeline of flagged activity
  - Control panel for manual overrides and emergency responses



(a) Technician / System Administrator Interface



(b) Security Interface

Figure 2: User Interface Mockups - Technician & Security

2.2 UI/UX Design Principles

The system employs a minimalist user interface with a clean, intuitive design that prioritises essential controls for users of all technical backgrounds. A consistent colour system is implemented using green for success states, blue for informational content, orange for warnings and red for alerts. The interface also provides adaptive theming option that automatically switches between dark and light modes based on ambient light conditions or user preferences, which improves visibility and reduces battery consumption. Security is implemented in progressive layers, with standard PIN authentication serving as a backup when biometric authentication fails.

2.3 Navigation Flow (*Example: Primary User*)

The navigation path for primary users begins when users log in to the app and they are directed to the dashboard. Users can access the real-time camera feed with status indicators by tapping on the door camera icon. Access logs are viewed by swiping up on the interface. To share access with others, users click the "Share Access" button, select a time period

and contact and generate a QR code or access token. Upon completion, users receive a confirmation notification.

2.4 User Interaction Scenarios

Scenario 1: Resident Access

In this scenario, a resident approaches the door and the PIR sensor is activated, which in turn starts the camera. The facial recognition system subsequently verifies the resident's identity when confidence levels are $\geq 85\%$. Once verified, the lock unlocks, accompanied by an audio signal, while the system records the event with a timestamp and user details and sends a confirmation alert to the resident's smartphone.

Scenario 2: Guest Access Management

To grant access to guests or visitors, residents select "Share Access" in the application and define parameters including recipient information, time window and access repetition. The system generates a secure one-time QR code which the guest presents at the door. The system verifies the validity of the code, logs the entry and sends a confirmation notification to the resident. Access credentials automatically expire after use or when the time limit is reached.

Scenario 3: Security Incident Response

When the system detects anomalies such as multiple failed attempts or some kind of tampering, it sends a high-priority alert to the primary user with incident details. Based on user settings, optional security service notifications may be sent. The notification offers a remote lockdown option. The incident is recorded with continuous video capture and an audit trail remains available for review in the security log.

2.5 Privacy & Data Management

Biometric data is processed on-device whenever possible to enhance privacy. All biometric data is stored with AES-256 encryption. Users have access to a one-tap option to delete all personal data from the system. The system reduces data storage by automatically purging temporary visitor data after access expiration. During face data registration, users are presented with a clear data policy to ensure transparent collection practices.

2.6 Accessibility Features

The system offers complete functionality through voice commands to improve accessibility. Visual accommodations include adjustable text size, high-contrast mode and screen reader compatibility. A simplified interface option that displays only essential functions is available for elderly or non-technical users.

The mobile application serves as the central control hub for the Smart Access Control System, allowing intuitive management of access permissions while maintaining robust security protocols. The interface adapts to different user roles (*residents, guests, administrators*) to present only relevant controls and information based on permission levels.

3 Description of Each Thing

This Smart Access Control System consists of three main Things, each designed with a specific role and clear affordances. Each Thing integrates a sensor, a processor and an output. Their design follows the W3C Web of Things (WoT) JSON model for interoperability.

Thing 1: Presence Detection Module

- **Sensor:** PIR Motion Sensor
- **Processor:** ESP32
- **Output:** Presence detected/not detected signal

Affordances (*WoT JSON model*):

```
{
  "name": "Presence Detection Module",
  "type": "thing",
  "description": "PIR-based motion sensor",
  "properties": {
    "motionDetected": {
      "type": "boolean",
      "description": "Motion detected or not"
    },
    "sensitivityLevel": {
      "type": "integer",
      "description": "Sensor sensitivity"
    }
  },
  "actions": {
    "resetSensor": {
      "description": "Reset sensor"
    },
    "adjustSensitivity": {
      "description": "Set sensitivity level",
      "input": {
        "type": "integer"
      }
    }
  },
  "events": {
    "continuousMotion": {
      "description": "Ongoing motion detected",
      "data": {
        "duration": {
          "type": "integer",
          "description": "Seconds of motion"
```

```
        }
      }
    }
  }
}
```

Functionality: This module detects human movement near the entry point. Upon detecting presence, it wakes up the Face Recognition Unit to start scanning. It ensures energy efficiency by keeping the system in a low-power state during inactivity.

Thing 2: Face Recognition Unit

- **Sensor:** Camera Module (*ESP32-CAM*)
- **Processor:** ESP32
- **Output:** Face match success/failure signal

Affordances (*WoT JSON model*):

```
{
  "name": "Face Recognition Unit",
  "type": "thing",
  "description": "Camera-based face recognition",
  "properties": {
    "faceDetected": {
      "type": "boolean",
      "description": "Face present or not"
    },
    "matchConfidence": {
      "type": "number",
      "description": "Match accuracy"
    },
    "batteryLevel": {
      "type": "number",
      "description": "Battery percentage"
    }
  },
  "actions": {
    "enrollFace": {
      "description": "Add a new face"
    },
    "verifyFace": {
      "description": "Check face match"
    },
    "adjustSensitivity": {
      "description": "Set detection threshold",
      "input": {
```

```
        "type": "number"
      }
    }
  },
  "events": {
    "unauthorizedAttempt": {
      "description": "Unknown face detected",
      "data": {
        "confidence": {
          "type": "number",
          "description": "Confidence level"
        }
      }
    }
  }
}
```

Functionality: The Face Recognition Unit captures real-time images and processes them locally using a lightweight machine learning model. If a face matches with $\geq 85\%$ confidence, it signals the Smart Lock to unlock. Failed attempts are logged with timestamps and security is optionally alerted.

Thing 3: Smart Lock & Alert Unit

- **Sensor:** External digital inputs (*Face Recognition/App commands*)
- **Processor:** ESP32
- **Output:** Servo Motor for lock; Notification alerts

Affordances (*WoT JSON model*):

```
{
  "name": "Smart Lock & Alert Unit",
  "type": "thing",
  "description": "Controls door lock and sends alerts",
  "properties": {
    "lockStatus": {
      "type": "string",
      "enum": ["locked", "unlocked"],
      "description": "Current lock state"
    },
    "batteryLevel": {
      "type": "number",
      "description": "Battery percentage"
    },
    "lastAccessTime": {
      "type": "string",
      "format": "date-time",
```

```
      "description": "Last unlock time"
    }
  },
  "actions": {
    "lockDoor": {
      "description": "Lock the door"
    },
    "unlockDoor": {
      "description": "Unlock the door"
    },
    "sendNotification": {
      "description": "Push alert to user"
    },
    "toggleAutoLock": {
      "description": "Enable or disable auto-lock"
    }
  },
  "events": {
    "forcedEntry": {
      "description": "Possible break-in detected",
      "data": {
        "time": {
          "type": "string",
          "format": "date-time",
          "description": "Time of event"
        }
      }
    },
    "lowBattery": {
      "description": "Battery is low",
      "data": {
        "batteryLevel": {
          "type": "number",
          "description": "Current level"
        }
      }
    }
  }
}
```

Functionality: This unit locks/unlocks the door via a servo motor based on signals from the Face Recognition Unit or app commands. It auto-locks after 30 seconds and sends notifications for anomalies like failed attempts or tampering, maintaining an access log.

System Interconnection

The connections are established as follows:

- **PIR Sensor → Face Recognition Unit:** Activates camera when presence is detected, thus saving power.
- **Face Recognition Unit → Smart Lock:** Unlocks the door upon successful verification.
- **Mobile App → Smart Lock:** Enables remote management, override and notifications.

Communication Protocol: MQTT over TLS with AES-256 encryption between all components and the mobile app.

3.1 System Architecture Diagram

The system architecture is illustrated using a diagram as shown in Figure 3. The key components of the system are:

- **Motion Detection Unit:** A PIR sensor (*motion sensor*) that detects presence and triggers the face recognition process.
- **Face Recognition Unit:** An ESP32-CAM module that captures an image and attempts face recognition when presence is detected.
- **Smart Lock Controller:** An ESP32 microcontroller that receives MQTT unlock signals upon successful face recognition and controls a servo motor.
- **Physical Lock Mechanism:** A door lock operated by the servo motor, unlocking the door when authorized access is granted.

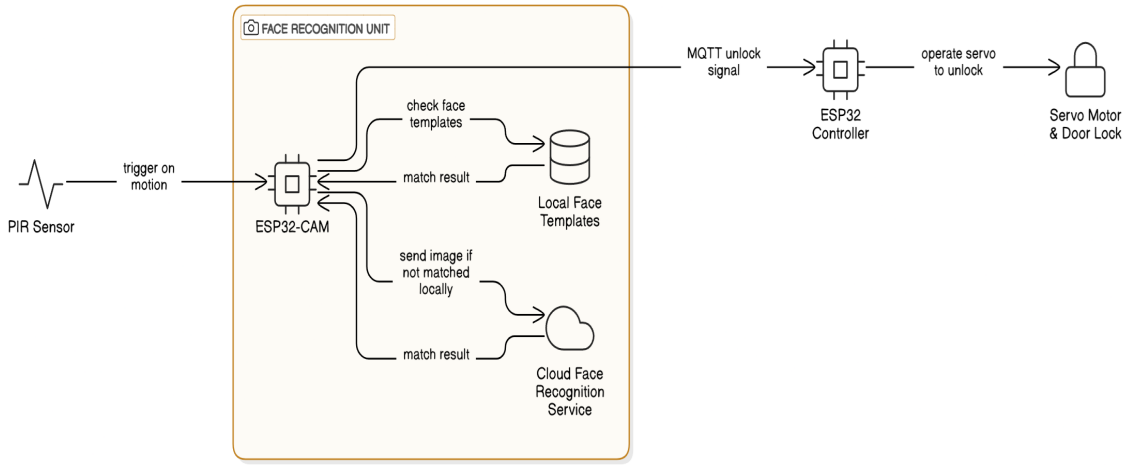


Figure 3: System architecture showing the interaction between the sensors, processor & lock

The diagram demonstrates the flow of interaction where the PIR sensor detects presence, triggering the ESP32-CAM for face recognition. Upon successful recognition, an MQTT unlock signal is sent to the ESP32 controlling the servo motor to unlock the door.

4 Machine Learning

The Smart Access Control System uses machine learning to provide intelligent, adaptive and secure physical access control. This section details the data sources, training challenges, processing pipeline, deployment strategy and evaluation metrics of our ML implementation.

4.1 Data Provenance and Collection

The system collects data from multiple integrated sensors and devices:

- **Camera Feed (*ESP32-CAM*):** Captures facial images (*resolution 1280×720*) at the entry point when motion is detected. These images form the core dataset for facial recognition training and inference.
- **Motion Detectors (*PIR*):** Generate binary motion data (*presence/absence*) with timestamps, used to identify presence anomalies and track occupancy patterns.
- **Access Logs:** Record structured data including timestamps, user IDs, access methods, authentication outcomes and duration of access events.

Data governance procedures ensure compliance with GDPR and CCPA regulatory frameworks. Raw facial images are processed at the edge and immediately discarded after embedding extraction, minimizing persistent storage of sensitive biometric data. Users provide explicit opt-in consent during the onboarding process, documenting their agreement to data collection and processing activities. All transmitted data is encrypted using AES-256 encryption to maintain confidentiality during network communications. Data retention policies limit storage to 90 days for non-essential information, after which it is permanently deleted from system databases.

4.2 Training Challenges

The system faces a limited training data challenge as each authorised user typically provides only 5-10 facial samples during enrolment, creating significant data scarcity. Additionally, the distribution of access events shows a high imbalance between regular users and occasional visitors. To address these limitations, we implement several techniques. Data augmentation methods include random rotation, brightness variation, horizontal flipping and slight perspective transformations. A few-shot learning approach using Siamese networks helps learn similarity metrics rather than direct classification. For underrepresented classes, we generate synthetic data using adversarial techniques. Transfer learning is applied by fine-tuning a pre-trained MobileNetV2 model [3] on our specific user database.

Resource constraints present another significant challenge, as the ESP32 microcontroller has severe memory limitations of 520KB internal RAM and computational constraints that make standard ML model implementation difficult. We address these constraints through model quantisation that decreases the model size by reducing floating-point operations to 8-bit integer arithmetic. Through knowledge distillation, a simpler "student" model gains insights from a more complex "teacher" model. Additionally, we utilise the CMSIS-NN library [4] to perform custom layer optimisations, enhancing neural network operations on microcontrollers.

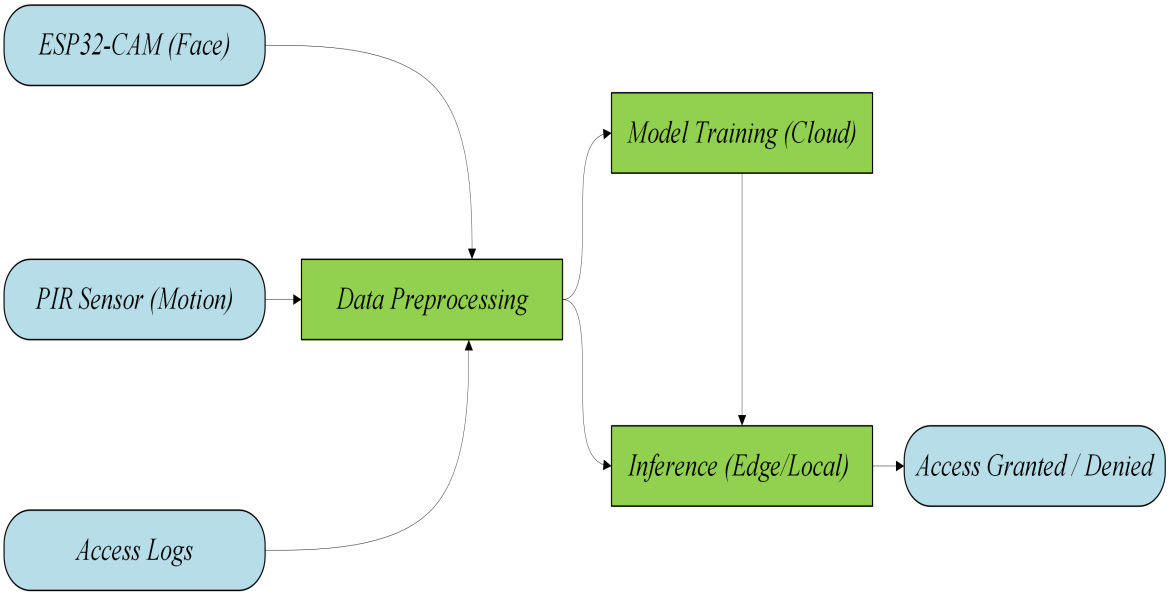


Figure 4: Machine Learning High-Level Diagram: Data from sensors and access logs is collected, preprocessed, used to train a cloud-based model, then deployed to edge devices for real-time access decisions.

4.3 Training and Inference Pipeline

The facial image processing stage includes face detection using a lightweight HOG-based detector [5], alignment based on detected facial landmarks and resizing to 224×224 pixels to match MobileNetV2 input requirements. Images then undergo normalisation to the [-1, 1] range and augmentation as described in the challenges section. Behavioural data processing involves temporal aggregation of access events into hourly and daily patterns, feature extraction including frequency, duration and sequence of access attempts, normalisation of time-series data and outlier removal using the IQR method.

Model training occurs off-device in a cloud environment following a defined workflow. This process begins with initial model pre-training on public face datasets, followed by transfer learning and fine-tuning on the authorised user dataset. Quantisation-aware training optimises the model for edge deployment. The behavioural model trains using historical access patterns and all models then undergo validation against held-out test data. The training schedule includes initial training upon system setup with all enrolled users, model updates every two weeks to incorporate new usage patterns and complete model retraining every three months or sooner if performance decreases significantly.

The inference process comprises two main components. Real-time facial recognition runs entirely on-device using the ESP32-CAM, triggered by PIR sensor activation. This process uses a quantised TensorFlow Lite model to perform similarity matching against stored templates [6], with decision thresholds dynamically adjusted based on environmental conditions. Anomaly detection employs a hybrid approach with preliminary detection on-device and complex pattern analysis in the cloud. The system utilises the Isolation Forest algorithm [7] and runs both periodic batch processing and real-time checks.

4.4 Deployment Strategy

Our system implements a hybrid deployment architecture that balances on-device and cloud processing. Edge processing on the device handles face detection and recognition inference, real-time authentication decisions, basic anomaly detection and temporary data storage. The cloud component manages model training and retraining, complex anomaly detection, model performance monitoring and long-term storage of embeddings only. Model updates are delivered through secure OTA (*Over-The-Air*) updates using encrypted channels with certificate-based authentication. The smartphone application serves as the update management interface, allowing users to approve updates before installation.

4.5 Performance Evaluation Metrics

We employ the following metrics to evaluate our ML system performance:

1. **False Acceptance Rate (*FAR*):** Measures security effectiveness by calculating the percentage of unauthorised access attempts incorrectly granted access. Our target threshold is FAR $\leq 0.1\%$.
2. **False Rejection Rate (*FRR*):** Quantifies usability by measuring legitimate users incorrectly denied access. Our system targets FRR $\leq 2\%$ to ensure user convenience.
3. **Authentication Latency:** Measures the time from motion detection to access decision. Our threshold is ≤ 1 second total processing time to maintain a smooth user experience.
4. **Equal Error Rate (*EER*):** The point where FAR equals FRR, indicating the optimal security-usability trade-off. Our system achieves an EER $\approx 1.5\%$.

Additional performance metrics include model size, which shows the memory footprint of deployed models; power efficiency, which is measured as energy consumption per authentication attempt; user satisfaction, which is measured on a scale of 1 to 5 through periodic application surveys; and the F1 Score, which is the harmonic mean of precision and recall for anomaly detection.

5 Managing Human-Computer Interaction

The Smart Access Control System implements a user-centric design approach that balances security with intuitive usability. This section outlines how different stakeholders interact with the system through multiple interaction channels.

Stakeholder Interaction Flow

Residents interact with the system primarily through facial recognition. If authentication fails or is unavailable, the user can manually authenticate using the app's secure PIN fallback or biometric unlock. The system provides real-time feedback on the mobile app and the on-device screen (*e.g., "Access Granted"*).

Visitors receive temporary access links generated by the resident or building administrator. These are time-bound and QR code-based. Once scanned at the entrance module, access is granted or denied based on policy.

Administrators use a dashboard within the app to register new users, monitor activity logs, review anomaly alerts and retrain the ML model using newly acquired data.

Security personnel receive alerts on unauthorised access attempts or anomalies, along with live camera feeds and logs to take further action.

Storyboard: A Typical Access Scenario

- **Step 1:** User walks to the entrance; the camera detects motion and captures the face.
- **Step 2:** The device performs on-device facial recognition and displays feedback.
- **Step 3:** If verified, the door unlocks; else, the app notifies the user and offers alternatives.
- **Step 4:** The log is updated and anomaly checks are triggered if access occurs at an unusual time.

This interaction model ensures intuitive usability for all stakeholders while maintaining strong security and privacy controls.

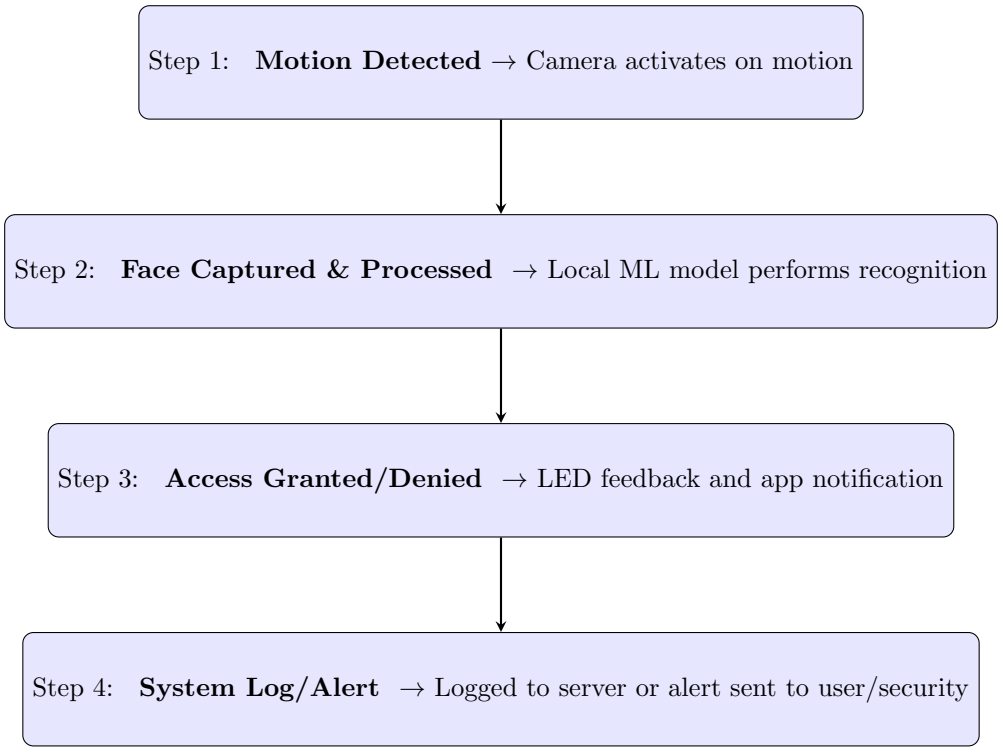


Figure 5: Access Control Flowchart: Shows the sequence from motion detection and face recognition to access decisions and event logging

6 Managing Security and Privacy

The Smart Access Control System integrates comprehensive security measures and privacy protections throughout its architecture, addressing potential vulnerabilities while maintaining usability for authorized users.

Security Measures

Data in transit between IoT components (*ESP32-CAM, sensors and app*) is end-to-end encrypted using TLS1.3. For local device communication, mutual authentication and encrypted Wi-Fi credentials prevent unauthorized access.

Access to the mobile application is protected using biometric authentication or a secure PIN. Backend APIs are safeguarded through token-based authorization, rate-limiting and strict role-based access controls (*RBAC*) for residents, admins and security personnel.

Edge computing enhances security by processing biometric data locally on the ESP32-CAM, ensuring raw facial images never leave the device. Only encrypted feature vectors are transmitted when necessary for model updates, minimizing data exposure during authentication processes.

Privacy Considerations

To protect user privacy, facial images are processed in real-time and immediately discarded after authentication, with only anonymized embeddings retained. All access logs are timestamped and hashed before being archived.

Visitors using QR codes are granted temporary tokens with strict expiration and usage constraints, ensuring no permanent data storage for non-residents.

Other Risks and Mitigation

Risks such as device tampering, spoofing and model poisoning are mitigated via hardware-based secure boot, liveness detection techniques and secure model update verification mechanisms.

By embedding privacy and security by design, the system ensures robust protection against both digital and physical threats during real-world deployment.

Risk Matrix

Threat	**Likelihood**	**Impact**	**Mitigation Strategy**
Cloud service compromise	Low	Critical	Minimize cloud reliance; store minimal data; encrypt data at rest
Spoofing via printed photo or screen replay	Medium	High	Implement liveness detection (*e.g., blink detection, depth sensing*)
Data leakage during transmission	Medium	High	Use end-to-end encryption (*TLS*), mutual authentication and secure API tokens
Unauthorized access to admin portal	Medium	High	RBAC, MFA, audit logging and brute-force detection
Tampering with on-device components	Low	High	Use tamper-evident casing and hardware secure boot
QR code misuse (*e.g., sharing by visitor*)	Medium	Medium	Enforce time-bound, single-use tokens with geofencing
ML model poisoning or drift	Low	Medium	Secure model updates, use anomaly detection to flag drift

Table 1: Risk Matrix and Mitigation Strategies

7 Approaches to Evaluation

To ensure that the Smart Access Control System meets its functional and non-functional requirements, a structured evaluation strategy is employed. The following table outlines the core system requirements, associated evaluation criteria and methods of assessment:

Requirement	**Evaluation Criteria**	**Evaluation Method**
Accurate Facial Recognition	FAR ($\leq$0.1%), FRR ($\leq$2%), EER ($\approx$1.5%)	Use test dataset; compare predictions vs actual labels; confusion matrix
Low Latency Access Decision	Time to authenticate	Measure average time from detection to access decision in real-time
Robust Security	Resistance to spoofing, tampering and replay attacks	Perform penetration testing and simulate common attacks (*e.g., photo spoofing*)
User-Friendly Interaction	Stakeholder satisfaction and ease-of-use	Conduct usability testing; survey feedback from mock users
Privacy Preservation	Minimal sensitive data exposure	Audit data lifecycle; verify encryption in use
System Reliability	Uptime and failure rate	Simulate high usage scenarios and hardware failures; log system uptime
Scalability	Support for increased number of users/devices	Add synthetic users to simulate scale and measure performance degradation

Table 2: Evaluation Requirements, Criteria and Methods

This evaluation approach ensures both technical soundness and real-world usability. Security is tested through simulations of common attack vectors, while machine learning performance is evaluated using standard metrics such as accuracy and inference time. User experience is validated via feedback and interaction logging during prototype trials.

8 Conclusion

This report presents a comprehensive design and prototyped implementation of a Smart Access Control System that integrates IoT, machine learning, and role-based user interaction via a smartphone application. The system demonstrates how modern security and usability principles can be combined to create a secure, intuitive and adaptive environment for residential and commercial access management.

The system architecture assigns specific functions to four user roles: residents, visitors, technicians, and security personnel. This role-based structure enables appropriate access levels and functionality for each user category. The integration of facial recognition, time-bound guest access, real-time alerts, and layered authentication reflects a strong focus on both security and user experience.

Furthermore, the interface was implemented by keeping minimalism and simplicity in mind. These implementations, showcased through UI mockups, prove that the system's conceptual architecture can be translated into practical, user-ready software.

To ensure data confidentiality and integrity, the system adopts AES-256 encryption for secure storage of access credentials and facial recognition embeddings. Communication between IoT devices and the central server is protected using TLS 1.3, preventing man-in-the-middle attacks and eavesdropping. Additionally, all personally identifiable information (PII) is anonymised where possible, and the system enforces strict data minimisation and retention policies in compliance with modern privacy frameworks like GDPR.

Overall, the project demonstrates a functional model for access control systems that address security needs while maintaining user privacy and system scalability. It shows practical applications of IoT and machine learning technologies in everyday security systems.

References

[1] Yuyu Wahyu et al. "A Performance evaluation of ESP32 Camera Face Recognition for various projects". In: *IOTA Journal* 2.1 (2022), pp. 10–21.

[2] Yash Nimbalkar et al. "Smart Door Lock System Using Face Recognition". In: *International Journal for Research in Applied Science and Engineering Technology* 10.5 (2022), pp. 3111–3113.

[3] Mark Sandler et al. "Mobilenetv2: Inverted residuals and linear bottlenecks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[4] Liangzhen Lai, Naveen Suda, and Vikas Chandra. "Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus". In: *arXiv preprint arXiv:1801.06601* (2018).

[5] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee. 2005, pp. 886–893.

[6] Pierre-Emmanuel Novac et al. "Quantization and deployment of deep neural networks on microcontrollers". In: *Sensors* 21.9 (2021), p. 2984.

[7] Dong Xu et al. "An Improved Data Anomaly Detection Method Based on Isolation Forest". In: *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*. Vol. 2. 2017, pp. 287–291. DOI: 10.1109/ISCID.2017.202.