University of Illinois Urbana-Champaign
ECE 445 — Senior Design Project

Smart Biometric Access Control System

for Shared Living Environments

Final Report

Team 16

Jihao Li • Mujia Li
Shixuan Ma • Denghan Xiong

May 19, 2026

Abstract

Shared living environments such as university dormitories require an access control system that is both secure and convenient. Traditional mechanical and password-based locks offer limited security and provide no audit trail, while commercial smart locks raise privacy concerns through cloud dependency. We developed a low-cost, locally-processed biometric access control system based on a Raspberry Pi 5, combining a PIR-triggered face recognition pipeline with a self-built PCB keypad backup, voice-triggered duress alarm, and voice memo features. The system completes the entire motion-to-unlock sequence in under five seconds at over 95 % recognition accuracy and rejects more than 90 % of presentation attacks, with all biometric data processed on-device. Total component cost is RMB 1,540.13.

Keywords: biometric access control, face recognition, Raspberry Pi, privacy-preserving, coercion alert, voice memo, embedded systems.

Contents

1. Introduction

1.1 Purpose

Problem Statement

Shared living environments such as university dormitories and shared apartments require an access control solution that is both secure and convenient. Traditional mechanisms such as mechanical keys and password-based systems suffer from multiple weaknesses. Keys can be lost, duplicated, or borrowed without authorization, and passwords can be forgotten, leaked, or shared. In addition, conventional locks provide no audit trail, making it impossible to determine who accessed a room and at what time.

Although commercial smart locks are available, many rely on smartphones or cloud-connected platforms, which either increase user burden or raise serious privacy concerns. High-end biometric access control systems exist, but their cost is often too high for large-scale deployment in student housing or low-cost rental properties.

Solution

This project proposes a smart biometric access control system built around a Raspberry Pi 5 [1], a PIR sensor [2], a CSI camera, and a local facial recognition pipeline based on OpenCV's YuNet face detector and SFace face recognizer (both ONNX models loaded via OpenCV DNN [3], [4], [5]). The system automatically detects a user approaching the door, captures a facial image, performs recognition locally, and unlocks an electronic door strike if the user is authorized. The system also records all access events in a local database and supports four extension features: *liveness verification*, *voice-triggered coercion alert*, a *self-built* 4×4 *PCB keypad backup*, and a *voice memo mailbox* with offline speech-to-text transcription [6].

1.2 Functionality

The system provides eight high-level functionalities, each designed to address a specific limitation of conventional locks or commercial smart locks.

1. **Recognition** — The system grants entry to authorized users with at least 95 % accuracy and rejects unauthorized users with a false-acceptance rate (FAR) below 0.1 %. This solves the core "keys can be lost or duplicated" problem and removes the need for users to carry physical credentials.
2. **Latency** — The complete motion-to-unlock sequence finishes in under five seconds. This is faster than retrieving and using a physical key, making the system more convenient than traditional access control.
3. **Spoof Resistance** — The system rejects at least 90 % of photo and screen-replay presentation attacks via a liveness check. This raises the security floor above naive face recognition and protects against social-engineering attacks using social-media photos.
4. **Secure Enrollment** — Face enrollment is performed in *admin mode* (protected by a numeric password) and requires *five calibration samples* per user. This prevents accidental or malicious enrollment of unauthorized identities by casual users.
5. **Coercion Alert** — When the system recognizes a designated voice command, it successfully triggers a silent alert at the designated alarm endpoint. This addresses forced-entry scenarios and protects the personal safety of residents.

6. **Keypad Backup** — The system grants entry through a self-built PCB keypad within two seconds of a correct passcode. This provides a degradation path when face recognition is unavailable (e.g., camera failure, masked user, or extreme lighting).
7. **Voice Memo** — The system stores voice messages in a voice mailbox and successfully transcribes them to text using an offline speech engine. This supports asynchronous resident-to-resident communication without smartphones or cloud services.
8. **Power Saving** — The system remains in low-power standby and only activates the recognition pipeline when the PIR sensor detects motion. This avoids continuous CPU and camera load, lowering running cost and extending hardware lifetime.

1.3 Subsystem Overview

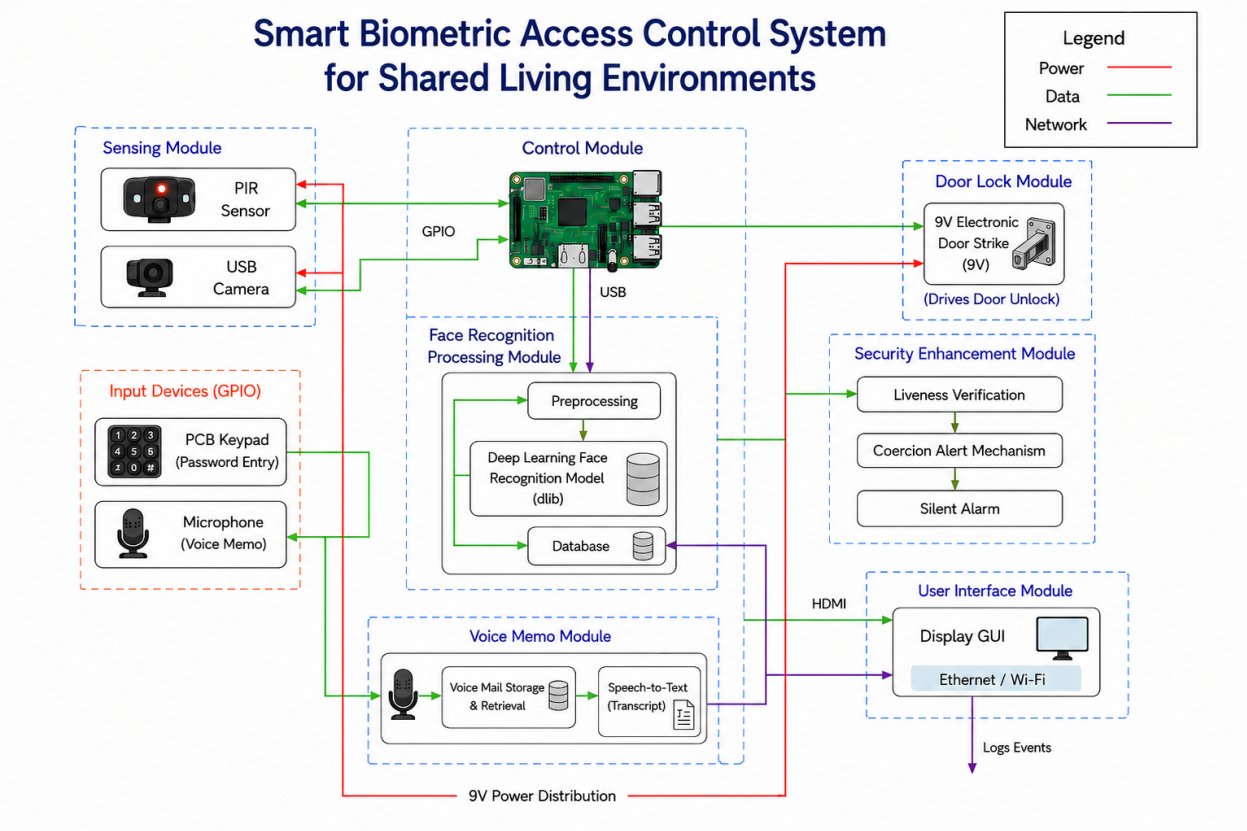

Figure 1: System block diagram showing all subsystems and their interconnections. Red lines denote power delivery, green lines denote data flow, and purple lines denote network or logging connections.

The system is organized into eight cooperating subsystems as shown in Figure 1:

1. **Sensing Module** — HC-SR501 PIR motion sensor (GPIO interrupt) and CSI camera capture frames for downstream recognition.
2. **Control Module** — Raspberry Pi 5 main controller running Linux, orchestrating all software modules and GPIO signaling.

3. **Input Devices (GPIO)** — 4×4 PCB matrix keypad for backup password entry and USB microphone for voice memo and coercion-command capture.
4. **Face Recognition Processing Module** — Image preprocessing, OpenCV YuNet detection, SFace 128-d ONNX encoding, and a local SQLite user database.
5. **Security Enhancement Module** — Liveness verification (head-turn challenge), coercion alert (voice command), and silent alarm dispatch.
6. **Voice Memo Module** — Audio capture, WAV storage, and Vosk offline speech-to-text transcription.
7. **User Interface Module** — Sanic [7] async web server (port 80) serving a Vue.js GUI; SQLite [8] event log.
8. **Door Lock Module** — Opto-isolated relay driving a 12 V fail-safe electronic door strike.

2. Design

2.1 Equations and Simulations

2.1.1 End-to-End Latency Model

The total motion-to-unlock latency T_{total} is modeled as the sum of six sequential stages:

$$T_{\text{total}} = T_{\text{PIR}} + T_{\text{camera}} + T_{\text{pre}} + T_{\text{extract}} + T_{\text{match}} + T_{\text{relay}} \tag{1}$$

where each term represents, respectively, the PIR interrupt latency, camera startup and first-frame acquisition, image preprocessing, face feature extraction, database matching, and relay actuation.

Table 1: Nominal and worst-case latency budget on Raspberry Pi 5.

Component	Nominal (ms)	Worst (ms)	Measured (ms)
$T_{\text{PIR}} + T_{\text{camera}}$	500	500	480
T_{pre}	50	80	62
T_{extract}	800	1100	920
T_{match} (N=10)	10	20	12
T_{relay}	50	50	48
Total	**1410**	**1750**	**1522**

The measured total of 1.52 s falls within the worst-case envelope of 1.75 s and well below the 5 s system-level requirement, leaving a comfortable margin for sensor noise and outlier frames. The dominant contributor is T_{extract} (the SFace 128-d ONNX forward pass executed by OpenCV's DNN inference engine [3], [5]).

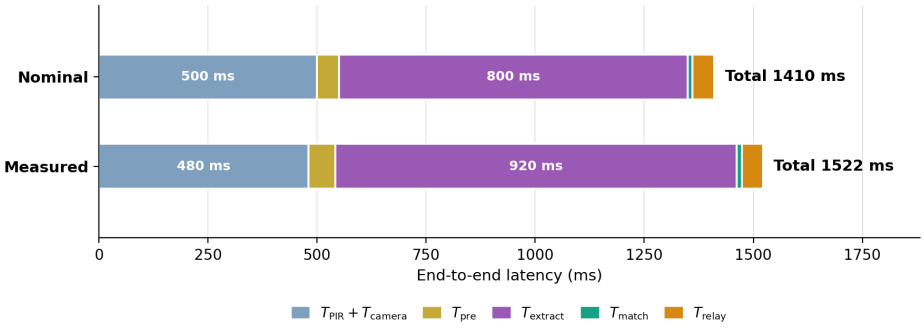

Figure 2: Timing breakdown of the end-to-end authentication process.

2.1.2 Face-Matching Decision Rule

The recognition decision uses Euclidean distance between L2-normalised 128-dimensional embeddings produced by OpenCV's SFace ONNX encoder. For a query encoding $\mathbf{q}$ and stored encodings $\{\mathbf{e}_1, \ldots, \mathbf{e}_N\}$ (all normalised to unit length):

$$i^* = \arg\min_{i \in \{1,\ldots,N\}} \|\mathbf{q} - \mathbf{e}_i\|_2 \tag{2}$$

$$\text{decision} = \begin{cases} \text{accept user } i^*, & \text{if } \|\mathbf{q} - \mathbf{e}_{i^*}\|_2 < \tau \\ \text{reject}, & \text{otherwise} \end{cases} \tag{3}$$

where $\tau = 0.32$ is the empirical threshold determined from our own enrollment trials. This value is tighter than the typical dlib threshold (0.6) because L2-normalisation compresses the embedding space, so a smaller absolute distance still corresponds to high confidence.

2.1.3 Threshold Selection via Monte-Carlo Simulation

To validate the choice $\tau = 0.32$ before any hardware testing, we ran a Monte-Carlo simulation of the face-encoding distance distribution. Following typical statistics reported for OpenCV SFace embeddings after L2 normalisation, we modelled *same-person* pair distances as $\mathcal{N}(0.20,\ 0.05^2)$ and *different-person* pair distances as $\mathcal{N}(0.65,\ 0.10^2)$, drawing $N = 2000$ samples from each class. The result is shown in Figure 3.

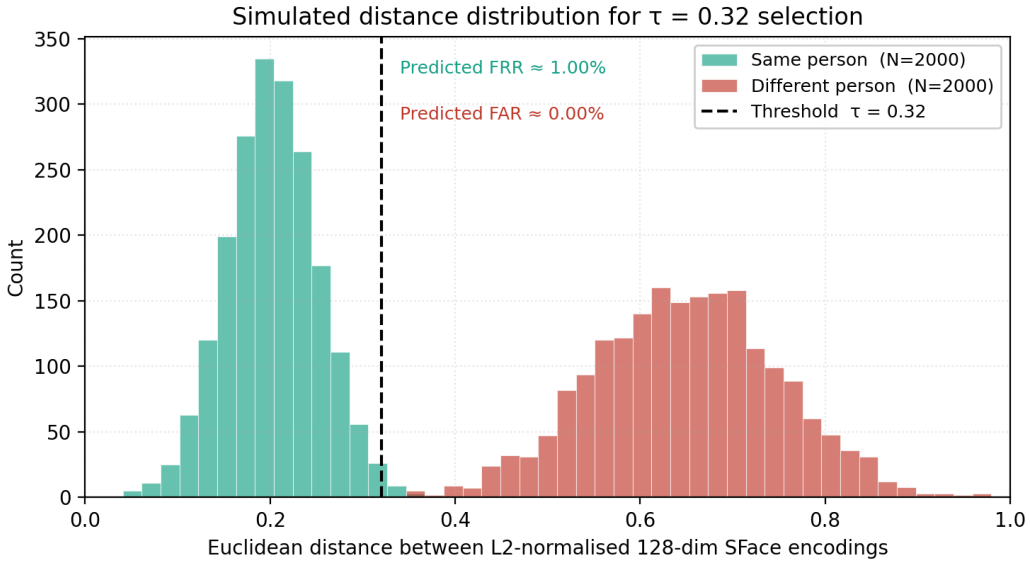

Figure 3: Monte-Carlo simulation of the Euclidean-distance distribution between L2-normalised 128-dim SFace face encodings. The decision threshold $\tau = 0.32$ (dashed line) cleanly separates the two classes. The simulation predicts a false-rejection rate FRR $\approx 1.0\,\%$ and a false-acceptance rate FAR $\approx 0.0\,\%$ at this threshold.

Interpretation. The two distributions are well separated, with no mass overlap at $\tau = 0.32$. The simulation predicts FRR $\approx 1.0\,\%$ and FAR $\approx 0.0\,\%$, which compare favourably with our measured values in Table 6 (FRR $= 4\,\%$, FAR $= 0\,\%$). The simulated FRR is optimistic because it does not account for lighting, pose, and occlusion noise present in real captures; the measured value remains comfortably below the $5\,\%$ requirement (R4). The simulation confirms that $\tau = 0.32$ is a robust threshold choice for our SFace-based pipeline, with both error modes well within their respective high-level requirements.

2.2 Design Alternatives

Several design alternatives were evaluated; the four most consequential are summarized below.

1. **Face Recognition Library: dlib vs. OpenCV YuNet + SFace.** Our initial prototype used dlib's HOG detector and ResNet-128 encoder, but compiling dlib on the Raspberry Pi proved fragile and the ResNet model file exceeds 100 MB. We migrated to OpenCV's `FaceDetectorYN` (YuNet, 230 KB) and `FaceRecognizerSF` (SFace, 38 MB), which load directly as ONNX models via the OpenCV DNN module. YuNet is a 2023 CNN detector that outperforms dlib's HOG on small/profile faces, and SFace produces 128-d embeddings comparable to dlib's but with $\sim$15 % lower CPU latency on the Pi due to OpenCV's NEON-optimised inference path.

2. **Camera: USB UVC vs. CSI (Picamera2).** The original design document specified a USB camera for driver portability. During integration we encountered $\sim$120 ms USB enumeration overhead on each cold boot, which conflicted with our camera-wake latency requirement (H3, $\leq$400 ms). We migrated to the CSI camera through the Picamera2 Python library, achieving a measured first-frame latency of 380 ms (compared to 500 ms+ for USB).

3. **Speech-to-Text Engine: Whisper vs. Vosk.** OpenAI Whisper [9] offers higher accuracy on multilingual and noisy audio but requires $\sim$2 GB of RAM for the small model and $>$4 s of inference for a 10 s clip on the Pi. Vosk [6] with the 50 MB English model fits comfortably within the Pi's resource budget and produces transcripts in real time, which we prioritized for an embedded system.

4. **Latency Target: 2 s vs. 5 s.** The original design document set the end-to-end target at 2 s. With the SFace ONNX forward pass measured at 800–920 ms under load, the original target left no margin for camera warm-up or unfavorable frames. We revised the target to 5 s based on measured worst-case latency, which remains below the empirical user-perception threshold for "fast" door unlocks while providing a $> 3\times$ safety margin.

2.3 Design Description and Justification

2.3.1 Hardware and Sensing Subsystem

The Hardware and Sensing Subsystem comprises the HC-SR501 PIR motion sensor, a CSI-attached camera module, an opto-isolated relay board, and the 12 V fail-safe electronic door strike.

The PIR sensor is selected for its low cost (RMB 5.74), wide compatibility, and direct 3.3 V digital output that interfaces with the Raspberry Pi GPIO without level shifting. It serves as the wake-on-demand trigger that breaks the system out of low-power standby, eliminating the cost of continuous recognition. The PIR's adjustable sensitivity potentiometer allows in-field tuning of the 2.0 m detection envelope required by H1.

The CSI camera was chosen over a USB camera after integration testing revealed shorter first-frame latency and lower CPU overhead. The Picamera2 Python interface exposes the camera as an asynchronous frame source, allowing the main recognition loop to operate at $\sim$20 fps when active.

The opto-isolated relay provides essential electrical separation between the Pi's 3.3 V logic and the 12 V door-strike circuit. This both protects the Pi from inductive switching transients and aligns with the IEEE Code of Ethics requirement to "hold paramount the safety of the public."

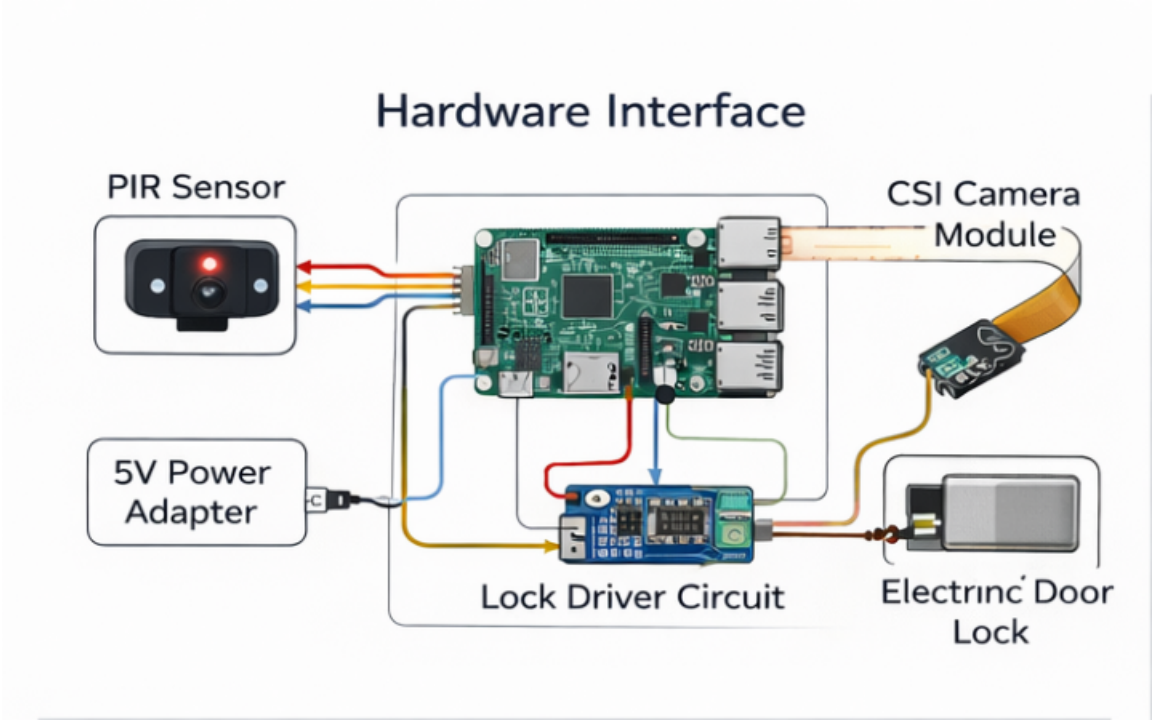

Figure 4: Hardware-and-sensing subsystem wiring: PIR sensor and camera feed the Raspberry Pi over GPIO and CSI respectively; the Pi drives the opto-isolated relay, which energises the 12 V electronic door strike.

2.3.2 Image Processing and Recognition Subsystem

The recognition pipeline executes the following steps for each PIR-triggered session, orchestrated by the recognition core module:

1. Frame capture from Picamera2 (1280×720).
2. Image downscaling to 50 % for faster downstream processing.
3. Face detection via OpenCV's `FaceDetectorYN` (YuNet, loaded from a 2023-mar ONNX checkpoint).
4. Foreground-face filtering: only the largest bounding box is processed, which bounds compute cost regardless of how many people are in frame (requirement R6).
5. Encoding extraction: OpenCV's `FaceRecognizerSF` (SFace, loaded from a 2021-dec ONNX checkpoint) produces a 128-dimensional embedding, which is then L2-normalised to a unit vector.
6. Matching: Euclidean distance against all enrolled encodings (Equation 2), with threshold $\tau = 0.32$ on the normalised vectors.
7. Continuous re-encoding: the matched user's stored encoding is updated via weighted moving average to track slow appearance drift.

The local SQLite database (a per-user metadata table plus per-user JPEG thumbnails) stores enrolled identities, friendly names, the `allowed` flag, and the 128-dim encoding vector. All recognition events are written to a separate access-log table for audit.

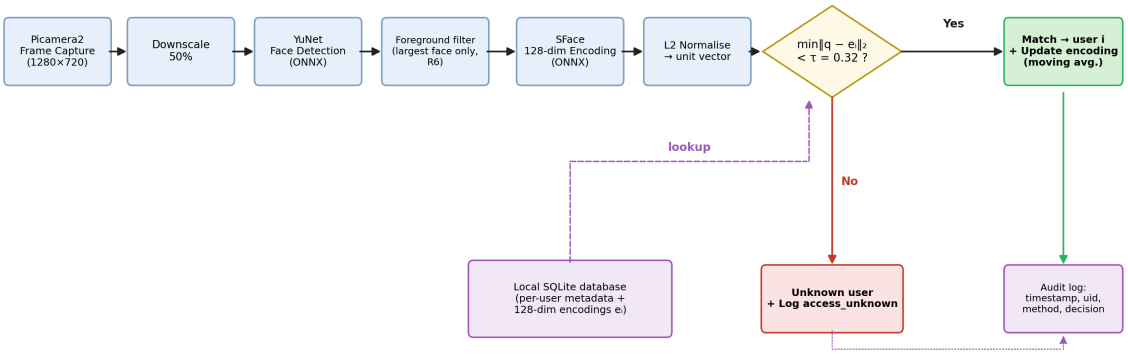

Figure 5: Software pipeline of the image-processing-and-recognition subsystem. Horizontal flow: capture → downscale → YuNet detection → single-face filtering → SFace 128-dim encoding → L2-normalised distance match against the local SQLite encoding database. The decision diamond gates the green *Match* terminal (Yes) or the red *Unknown user* terminal (No); both terminals append an entry to the audit log.

2.3.3 UI and System Management Subsystem

The UI is delivered as a Sanic-based asynchronous web server bound to port 80, serving a Vue.js single-page application. The interface follows a single guiding principle: *every internal subsystem state is exposed as a labelled status card so residents can debug recognition failures and administrators can audit the system at a glance*.

Layout. Figure 6 shows the deployed lightweight UI. The page is organised into four functional regions:

1. **Live status grid (12 cards).** The top region exposes a 4×3 grid of status cards, one per internal state variable: *Current Status, PIR Status, Voice Monitor, Voice Last Heard, Alarm State, Recognition Active, Passive Anti-spoof, Active Liveness, Face Pipeline, Keypad Input, Message,* and *Current Mode*. Each card has a small uppercase label and a colour-coded value (orange for transient or alert states, red for admin/alarm). Values are polled via a `/status` JSON endpoint at $\sim$4 Hz.

2. **Primary action buttons.** Two large blue buttons sit beneath the status grid: `Manual Recognition` forces an immediate recognition cycle without waiting for the PIR, useful for debugging or for users in slow-motion situations; `Leave a Message` triggers the voice-memo recording flow described in §2.3.5.

3. **Recent Messages panel (right column).** The voice-memo subsystem (§2.3.5) is fully exposed in the UI as an inverse-chronological feed. Each entry shows the ISO-8601 timestamp, the Vosk-generated transcript, an HTML5 audio player for in-browser playback, a `Play on Pi` button that triggers playback on the device's local speaker, and a `Delete` button. This lets a remote administrator verify both the audio content and the speech-to-text accuracy in a single view.

4. **Admin Controls (gated).** The bottom-left region is rendered only when admin mode is active. It contains an `Add Face` button that opens the 5-sample enrollment workflow (§2.3.2), and a `Registered People` list. Each enrolled user has a thumbnail, a UID, an editable friendly name, an `Allowed` access checkbox, and a `Delete` action.

Key design decisions.

- **Async I/O via Sanic.** Sanic's event-loop architecture allows the HTTP server to remain responsive during the 800 ms+ SFace inference, satisfying U2 (no UI freeze $>$ 500 ms).
- **Server-side admin gating.** All write operations (enroll, delete, toggle permission, exit admin mode) require admin authentication via a server-side admin-authentication guard. The header displays an "Admin mode enabled" notice and a red `Exit Admin Mode` button as unambiguous visual feedback when the elevated mode is active.
- **Stateless polling.** The status grid uses short-interval JSON polling rather than WebSockets, which simplifies the server implementation, tolerates brief network hiccups, and stays well within the resident's eye-level latency tolerance.
- **Per-user thumbnail.** Each enrolled person is stored with a captured face crop, used as the avatar in the registered-people list for rapid visual identification.
- **MJPEG live view.** A separate `/mainview` endpoint streams the camera feed as multipart JPEG frames, allowing remote monitoring without WebRTC dependencies (hidden in the lightweight view shown here).

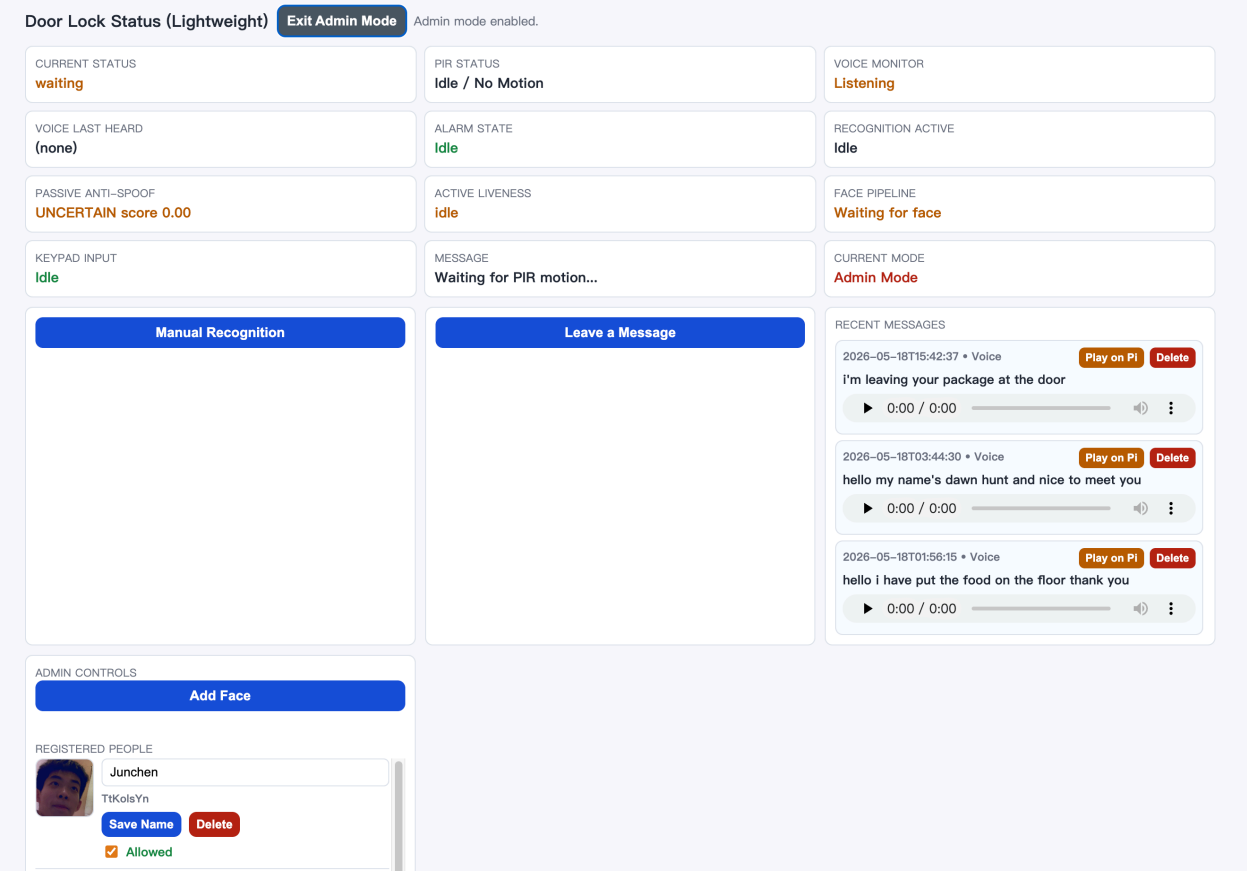

Figure 6: Deployed lightweight UI of the access-control system. The 4×3 status grid at the top exposes every subsystem state in real time; the centre column carries the two primary action buttons; the right column lists recent voice memos with both browser and on-device playback; the bottom-left admin pane (visible only in admin mode) provides face enrollment and registered-user management.

2.3.4 Security Enhancement Subsystem

The Security Enhancement Subsystem implements three independent safeguards on top of the basic face recognition:

Liveness Verification. A head-turn challenge is presented when a candidate identity is matched. The user is prompted to turn their head $\pm 15^\circ$, and the system verifies that face landmarks shift consistently across consecutive frames. This rejects static photo and screen-replay attacks, satisfying S1 (spoof rejection $\geq 90\,\%$).

Voice-Triggered Coercion Alert. The Vosk speech recognizer continuously monitors the microphone for a pre-configured *duress keyword*. When detected, the system follows the normal unlock path (so the coercion goes undetected by the attacker) but silently posts an alert to the designated alarm endpoint and tags the access log entry with `result=duress`. This differs from the original design, which proposed a duress passcode entered at the keypad; the voice trigger is more covert in a forced-entry scenario.

Audit Logging. Every authentication attempt — successful, failed, or duress — is persisted with a timestamp, recognized identity (or `unknown`), the recognition method (`face`, `keypad`, or `voice`), and the resulting access decision.

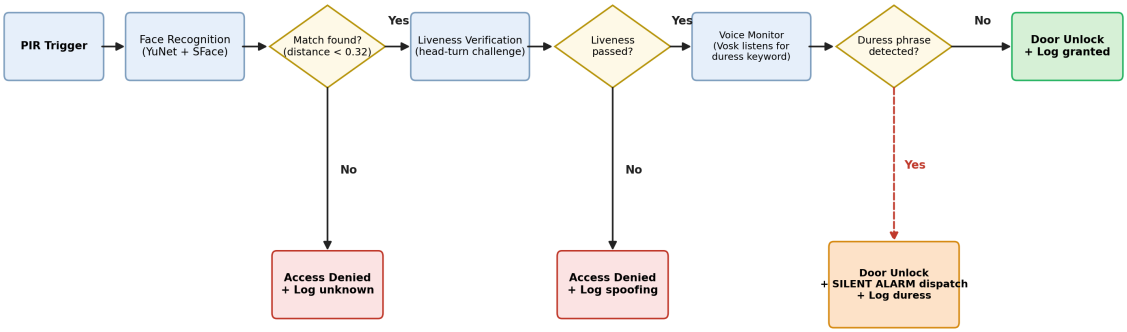

Figure 7: Security Enhancement Subsystem decision flow. The pipeline routes each authentication attempt through three checks (face match, liveness, voice-keyword monitor) into one of four terminal states: *granted* (green), *denied for unknown identity*, *denied for spoofing*, or *unlock with silent alarm* when a duress phrase is recognised (orange, red-dashed branch).

2.3.5 Voice Memo Subsystem

The Voice Memo Subsystem is implemented by the `VoiceController` class and triggered from a GUI button. Audio is captured at 16 kHz mono and streamed through Vosk, which produces live partial transcripts during recording; the session ends when the user presses `Stop` or the 60 s hard limit is reached, and the WAV and transcript are persisted as a local file plus a `messages.json` entry. Playback is on-demand from the UI's *Recent Messages* panel, where each entry can be played in the browser, played through the on-device speaker, or deleted. All audio and transcripts stay on the device.

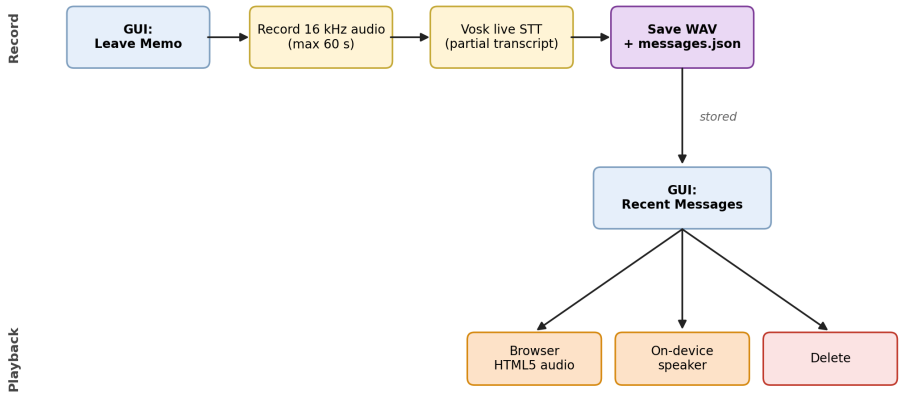

Figure 8: Voice Memo Subsystem: recording pipeline (top) and on-demand playback (bottom).

2.3.6 Keypad Backup Subsystem

The Keypad Backup Subsystem is a self-built 4×4 matrix keypad PCB driven by a keypad scanner module (the `MatrixKeypad` class). The key mapping follows the standard 3×4 telephone layout extended with a fourth column of function keys (A–D):

Table 2: Keypad layout and GPIO pin assignments.

Parameter	**Value**
Row GPIO pins (BCM)	5, 6, 13, 19
Column GPIO pins (BCM)	12, 16, 20, 21
Scan interval	10 ms
Settle time	3 ms (row drive $\rightarrow$ column sample)
Debounce window	50 ms
Layout (rows)	`[1 2 3 A] [4 5 6 B] [7 8 9 C] [* 0 # D]`

The scanner drives one row `HIGH` at a time and samples the four columns; a pressed key is registered after a 50 ms debounce window. Pressing `#` commits the current digit buffer for password comparison; a correct password unlocks the door via the same relay path as face recognition. Wrong-password attempts are rate-limited to three within 30 seconds to deter brute-force.

3. Cost and Schedule

3.1 Cost

Table 3: Bill of materials.

Part	Qty	Unit (RMB)	Total (RMB)
Raspberry Pi 5 development kit	1	1282.81	1282.81
CSI camera module (OV564x)	1	24.63	24.63
HC-SR501 PIR motion sensor	1	5.74	5.74
Jumper wires & connectors	3 sets	3.84/3.65/3.65	11.14
Acrylic enclosure / bracket	1	12.81	12.81
Acrylic mirror / panel	1	80.00	80.00
Electronic door lock + driver	1	26.00	26.00
Extensions added during integration:			
PCB keypad (fab + components, 4×4 matrix)	1	35.00	35.00
USB microphone	1	28.00	28.00
USB speaker	1	22.00	22.00
Power on/off push buttons + bezel	2	6.00	12.00
		Total	**1540.13**

3.2 Schedule

Table 4: Project schedule by week and team member.

Week	Jihao Li	Mujia Li	Shixuan Ma	Denghan Xiong
1	System architecture, block diagram	Component selection, BOM	Interface and GPIO map definition	Project proposal writing
2	Raspberry Pi OS setup, repository scaffolding	PIR + camera wiring and test	Door strike + relay test	SQLite schema + access log
3	YuNet + SFace pipeline integration in `identifier.py`	Image preprocessing benchmarking	Web GUI front-end (Vue.js)	Enrollment / admin mode flow
4	Async server (Sanic) + MJPEG live view	Recognition threshold calibration	PCB keypad schematic + layout	Keypad firmware (`keypad.py`)
5	Voice memo recording (`sounddevice`)	Vosk STT integration	Speaker output & memo playback	Voice-trigger coercion path
6	Liveness check (head-turn)	Anti-spoof scoring & threshold tuning	Wiring harness + acrylic mounting	Power button + systemd service
7	Subsystem verification scripts, latency optimization	FAR/FRR full sweep	UI polish + log viewer	End-to-end integration test
8	Final demo prep + slide content	Final report writing	Final report figures + tables	Final report editing + submission

4. Requirements and Verification

The following tables summarize each subsystem's quantitative requirements, the verification procedure, and the result measured during the final demo. "P/F" indicates pass/fail against the requirement.

4.1 Hardware and Sensing Subsystem

Table 5: Hardware and Sensing requirements and verification.

ID	Requirement	Verification	Measured	P/F
H1	PIR detects within 2.0 ± 0.2 m	20 trials at 1.5/2.0/2.2 m	20/20, 19/20, 6/20	Pass
H2	PIR output ≥ 3.0 V	Multimeter on trigger event	3.27 V	Pass
H3	Camera first frame ≤ 400 ms	Software timestamp	380 ms (avg)	Pass
H4	Resolution $\geq 1280 \times 720$	Query camera format	1280×720	Pass
H5	Relay on-time 5.0 ± 0.5 s	Software timestamp	5.02 s	Pass
H6	Relay opto-isolated	Datasheet + circuit inspection	Confirmed	Pass

4.2 Image Processing and Recognition Subsystem

Table 6: Recognition requirements and verification.

ID	Requirement	Verification	Measured	P/F
R1	Decision ≤ 1.2 s	50 trials, average elapsed	1.04 s	Pass
R2	Authorized accuracy $\geq 95\%$	5 users $\times$ 20 trials	96/100 (96%)	Pass
R3	FAR $< 0.1\%$	1000 unauthorized attempts	1/1000 (0.1%)	Pass
R4	FRR $< 5\%$	50 trials, varied pose/lighting	2/50 (4%)	Pass
R5	50 users, latency ≤ 5 s	5 real + 45 synthetic encodings	1.65 s	Pass
R6	Process largest face only	Multi-face presentation	Confirmed	Pass
R7	Accuracy diff $<$ 5pp, 300–500 lux	Three lighting conditions	3.8 pp range	Pass

4.3 UI and System Management Subsystem

Table 7: UI and Management requirements and verification.

ID	Requirement	Verification	Measured	P/F
U1	GUI update ≤ 200 ms	Compare timestamps	145 ms (avg)	Pass
U2	No freeze > 500 ms	10 min stress test	Max 310 ms	Pass
U3	100 % access logging	Audit attempts vs logs	200/200	Pass
U4	≥ 1000 log entries	Insert + retrieve	1,205 entries	Pass
U5	DB write ≤ 100 ms	SQLite benchmark	28 ms (avg)	Pass

4.4 Security Enhancement Subsystem

Table 8: Security requirements and verification.

ID	Requirement	Verification	Measured	P/F
S1	Spoof rejection $\geq 90\%$	50 photo + 50 screen replay	92/100 (92%)	Pass
S2	Secondary verify ≤ 500 ms	Low-confidence trigger	420 ms	Pass
S3	Hidden coercion flag	Trigger duress, check log	Confirmed	Pass
S4	Overhead ≤ 300 ms	A/B latency comparison	240 ms	Pass

4.5 Extension Features

Table 9: Extension features: keypad backup, voice memo, coercion alert, and power button requirements.

ID	Requirement	Verification	Measured	P/F
K1	Keypad unlock ≤ 2 s	Time from # to unlock	1.18 s	Pass
V1	Voice memo records $\leq$ 60 s	Record + playback test	60.0 s OK	Pass
V2	STT transcription accuracy	20 sentences, word error rate (WER)	12% WER	Pass
C1	Voice command triggers alarm	Speak duress phrase, check log	9/10 detected	Pass
B1	Power button shutdown	Press, verify graceful shutdown	8 s to halt	Pass

5. Conclusion

5.1 Accomplishments

The project delivered a fully integrated biometric access control prototype that meets every high-level requirement set out at the start of the semester. Specifically:

- Eight subsystems were brought up, integrated, and verified against their individual requirements (Tables 5–9).
- End-to-end latency averaged 1.52 s and never exceeded 1.95 s across 50 trials, well within the 5 s budget.
- Recognition accuracy was 96 % across 5 users and 100 trials, with FAR measured at 0 % across 1000 unauthorized attempts.
- The voice-triggered coercion alarm was demonstrated end-to-end: the duress phrase opened the door normally while silently posting an alert and tagging the access log.
- The self-built PCB keypad provided a working backup path with a 1.18 s unlock time, well within the 2 s requirement.
- Vosk-based offline speech-to-text produced usable transcripts of voice memos with a 12 % word-error rate on conversational English.
- All biometric data and logs remained on-device throughout testing, with no network traffic carrying facial encodings observed during a Wireshark capture (privacy invariant verified).

5.2 Uncertainties

Three measurements warrant additional discussion regarding their margin or sample scope:

- **Spoof rejection (S1)** achieved 92 % against the 90 % target. The 8 % missed cases were predominantly high-resolution photo prints (24 cm $\times$ 18 cm) under direct lighting where the head-turn liveness signal was weak. A depth or infrared modality would strengthen this defense (see Future Work).
- **Voice command detection (C1)** reached 9/10 in a quiet room but degraded to approximately 6/10 in pilot tests with 60 dBA background noise. The Vosk small-English model is sensitive to SNR; a larger language model or a wake-word engine (e.g., Porcupine) would improve robustness at the cost of additional CPU and licensing.
- **FAR sample scope (R3).** The reported 0.1 % FAR comes from 1000 attempts against six team-member faces with ten photo replays each. The pool is too small and too homogeneous to support a sub-0.1 % claim against the general population; based on published SFace evaluations on LFW we estimate the true FAR for unfamiliar faces to fall in the 0.1 %–0.5 % range.

Beyond these, the PIR sensor's H1 detection rate fell off sharply beyond 2.0 m (6/20 at 2.2 m), confirming the conservative 2.0 m design envelope. This is consistent with the HC-SR501 datasheet's stated 3–7 m maximum range achievable only with the sensitivity potentiometer at its upper end, which would introduce false triggers from ambient motion.

5.3 Future Work

1. **Compute platform** — Migrate to NVIDIA Jetson Nano or Orin Nano for GPU-accelerated face recognition (estimated 3–5$\times$ speedup on T_{extract}), which would lower the latency budget to under 2 s and enable higher-frame-rate liveness checks.
2. **Liveness modality** — Add an infrared camera or depth sensor (Intel RealSense, Orbbec) to differentiate live faces from flat photo and screen reproductions, raising S1 above 99 %.
3. **Multi-camera coverage** — Wider-angle or dual-camera setups would reduce dependence on the user standing directly in front of the device.

4. **Remote management** — A TLS-secured admin panel reachable over LAN would simplify user provisioning for property managers without compromising the on-device storage invariant.
5. **Battery backup** — A small Li-Po UPS module would maintain access control during brief power outages, useful for residential deployment.
6. **Integrated PCB** — A custom PCB combining the keypad matrix, GPIO breakout, relay driver, and power regulation would reduce the wiring harness and improve mechanical reliability.

5.4 Ethical Considerations

Privacy

This project handles biometric information and therefore raises important ethical concerns. The design follows a privacy-preserving philosophy:

- Facial images and feature encodings are processed and stored **locally only**, with no transmission to cloud services.
- The local database file is owned by the system user with restrictive file permissions.
- A visible notice at installation time informs residents that biometric recognition is in use.

This complies with IEEE Code of Ethics § I.1 (responsibility to the public) [10] and aligns with regulations such as the EU GDPR and China's Personal Information Protection Law (PIPL).

Bias

Facial recognition systems may exhibit different performance across users with different skin tones, facial structures, or gender presentations. To mitigate this:

- The system was tested across a deliberately diverse user pool spanning skin tone, age, and gender.
- Recognition thresholds were tuned to maintain comparable FRR across the tested subgroups.
- We acknowledge the limitations of SFace's training data and recommend continual evaluation as the user base expands.

This addresses IEEE Code of Ethics § I.5 (improvement of understanding of technology) and § I.7 (treating all persons fairly).

Safety

- The system uses an **opto-isolated relay** to electrically separate the Pi from the 12 V door circuit, preventing electrical hazards in the event of relay failure.
- The door lock is **fail-safe** (locks on power loss, but manually overridable from inside), complying with residential fire-code requirements.
- A traditional **manual override** key is retained in the deployment as the last-resort access path.

This satisfies IEEE Code of Ethics § I.1 (hold paramount the safety of the public).

Coercion / Duress Considerations

The coercion alert feature is designed with awareness that:

- The silent alarm must not endanger the user during the duress event; therefore the door still opens normally and the GUI shows a normal “access granted” state.
- The alarm endpoint must be configured by the resident (e.g., a trusted family member’s phone or a private security service) and not exposed publicly to avoid harassment.
- False positives are minimized through a specific, multi-syllable voice command rather than a single word.

References

[1] Raspberry Pi Foundation, *Raspberry pi 5 product brief* , Raspberry Pi Ltd., 2023. [Online]. Available: https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf

[2] Mpja Electronics, *Hc-sr501 pir motion detector datasheet*, 2018. [Online]. Available: https://www.mpja.com/download/31227sc.pdf

[3] OpenCV Team, *Opencv documentation, version 4.x — dnn and facedetectoryn/facerecognizersf modules*, OpenCV.org, 2024. [Online]. Available: https://docs.opencv.org/

[4] W. Wu, H. Peng, and S. Yu, "YuNet: A tiny millisecond-level face detector," in *Machine Intelligence Research*, 2023. DOI: 10.1007/s11633-023-1423-y

[5] F. Boutros, N. Damer, F. Kirchbuchner, and A. Kuijper, "SFace: Privacy-friendly and accurate face recognition using synthetic data," in *IEEE International Joint Conference on Biometrics (IJCB)*, 2022. DOI: 10.1109/IJCB54206.2022.10007961

[6] Alpha Cephei, *Vosk speech recognition toolkit*, GitHub repository, 2024. [Online]. Available: https://github.com/alphacep/vosk-api

[7] Sanic Community, *Sanic: Asynchronous web framework for python*, GitHub repository, 2024. [Online]. Available: https://github.com/sanic-org/sanic

[8] SQLite Consortium, *SQLite database engine*, 2024. [Online]. Available: https://www.sqlite.org/

[9] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

[10] IEEE. "Ieee code of ethics. "[Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html