Progress Report 1: Smart Door Security Framework with Anomaly Detection & Cybersecurity

Student Name: Advitiya Sharda
Student ID: 300395470
Project: Smart Door Security Framework with Anomaly Detection & Cybersecurity for Elderly Care
Course: CSIS 4495 – Applied Research Project, Section 3
Reporting Period: Jan 27 – Feb 9, 2026
Report Date: February 9, 2026
GitHub Repo: https://github.com/advitiyasharda/W26_4495_S3_AdvitiyaS

Work Date/Hours Logs

Date	Hours	Description of Work
Jan 27, 2026	1.0	Looked up basic information about privacy and security expectations for systems used in elderly care; checked that our proposal wording is realistic.
Jan 29, 2026	1.5	Drafted an email to Armin summarizing our planned direction (smart door + anomaly detection + security) and asking for his feedback on scope and priorities.
Jan 30, 2026	1.5	Read examples of simple rule-based checks for access control (e.g., repeated failed attempts, unusual times) and wrote short notes on how they fit our project.
Feb 3, 2026	2.0	In-person meeting with Armin: he showed us the Raspberry Pi, lock hardware, and a Ring-style camera prototype video; we discussed why each component is needed and what is realistic for this term.
Feb 3, 2026	1.5	Team debrief after hardware meeting: talked through what we saw, clarified which parts we will actually use, and adjusted our priorities based on Armin's comments.
Feb 4, 2026	1.5	Researched further about what are the possibilities for project creation
Feb 5, 2026	1.5	Team call to discuss how simple rules (e.g., time ranges, failed attempts) could work together
Feb 6, 2026	3.0	Tried a very small test page locally as a rough dashboard mockup just to understand the flow; this was exploratory, not final code.

Feb 8, 2026	3.5	Short team sync: reviewed the rough dashboard idea, confirmed our individual focus areas (me: security/threat rules), and agreed on what we want ready by midterm.
Feb 9, 2026	1.0	Updated repo documentation committed changes under the documents folder.

Total Hours This Period : ~18 hours

Summary Description of Work Done During This Reporting Period

This reporting period was mainly about **finalizing the project idea, confirming scope with the partner, and doing early research and planning**. We moved from general brainstorming and proposal writing in January into more concrete planning after seeing the actual hardware in early February, especially around how facial recognition, anomaly detection, and compliance requirements will fit together in one system.

- In late January, we finished choosing the project (smart door security with anomaly detection and a security/compliance angle), wrote and refined the proposal, and set up the GitHub repository and basic documentation.
- On **Feb 3**, we met Armin in person. He showed us the **Raspberry Pi**, the **lock hardware**, **a Ring camera and prototype video**, and explained how they fit into the overall Door Face Panels concept. This helped us understand what we are actually working with and what is realistic this term.
- After this meeting, my work shifted toward thinking through how **facial recognition** events should be logged securely, especially what data is necessary versus excessive from a privacy standpoint. I also reviewed basic Flask security patterns for authentication and logging, explored a very small dashboard mock-up to understand data flow, and documented audit-log fields so we do not miss critical security information during implementation.

We are still mostly in the **discussion and planning phase**, with only light exploratory code to test ideas (no real product yet). The current priority is to design the logic for threat detection and logging properly before we build more serious implementation.

Issues Encountered and How We Handled Them

- **Unclear hardware/setup at the start**
  At first, we only had a high-level idea of the smart door concept. The **Feb 3 hardware meeting** fixed this by letting us see the Pi, lock, and camera example. Now we know roughly what performance and integration constraints we have.

- **How much anomaly detection vs. rules?**
  We weren't sure how "heavy" the ML side should be. Talking to Armin and as a group, we decided that a simple, working system with clear rule-based alerts is more important than a complex model. If used, Isolation Forest or simple sequence-based checks will be kept at a realistic level appropriate for edge hardware.
- **How to treat facial recognition data under compliance**
  We had to think about how sensitive face-related data is. For now, the plan is to keep all recognition and logs on the device, store only what we really need, and later apply encryption and access control when we start implementing.

Changes to Proposal / Plan

Few changes to the written proposal, but we clarified a few points internally:

- The **edge-first** approach (everything on Raspberry Pi, no cloud dependency) is confirmed as a strict requirement.
- **Rule-based checks** and a basic anomaly model should work together, but the project will not aim for an overly complex ML system in this term.
- The **dashboard** will be simple: show access events, alerts, authentication, and a basic status, not a full-blown UI.
- We also confirmed that compliance is not a separate add-on, but something we have to consider from the start (Facial recognition logs).

Updated Individual Timeline

Next 1–2 weeks (up to Midterm):

- Turn the rule ideas (failed attempts, unusual time, long inactivity) into Python.
- Sketch how an **audit log** should be stored (fields, structure) so it can later be encrypted and queried.
- Write down what information we actually need to store from facial recognition events (e.g., ID, time, success/fail) so we don't keep unnecessary sensitive data.
- Work with the team to agree on the API endpoints where threat detection, facial recognition results, and logging will live.

Repo Check-In of Implementation Completed

So far, my work has mostly been **exploration**, not full implementation. The types of things I've checked into the repo are:

- **Docs / Planning**

    - Updated **README** with a clearer project overview and roles.
    - Added notes from the **hardware meeting** (Raspberry Pi, lock, camera prototype) so we remember the context.
    - Added a short note on facial recognition event logging (what to log and how it ties into privacy/compliance) to the documents folder.
- **Exploratory Code (Not final)**
    - A very small **test page** in an Implementation subfolder, just to see how an access log table and an alert box might look. This is clearly labelled as experimental so it does not confuse anyone as “finished code”.

I have been committing in small chunks in a private repo. There is **no production-ready security or threat detection code** .

AI Use Section

Tools I Used and How

- **Chat-based AI (e.g., ChatGPT / Gemini)**
  I used these mainly for:
    - Rephrasing parts of the proposal and report to sound clearer and more formal.
    - Getting simple explanations of concepts like “audit logs”, “encryption at rest”, and “basic Flask security”.
    - Brainstorming a few example rules for suspicious door access patterns (e.g., too many failed attempts, access late at night).
    - Quickly recap what facial recognition systems usually log (e.g., ID, timestamp, match result) so I could think about the privacy side.
- **Code suggestion tool (GitHub Copilot)**
  I used this when trying out the **small webapp:**
    - It suggested some basic route and template structure.
    - I treated this as a shortcut for boilerplate only and did not rely on it for any security-related logic.

What I Did Myself (Value Beyond AI)

- Chose which AI suggestions to keep or ignore based on:
    - What Armin told us in the meetings.

  - What the course expects.
  - What is realistic on our hardware.
- Read my own sources (docs, articles) when it came to **security/compliance** and rewrote the important points in my own words instead of copying AI suggestions.
- Adjusted the mock dashboard layout and logging ideas based on team discussions and the actual hardware we saw, not just generic AI advice.

Examples of Actual Prompts (Appendix)

Some examples of the kind of prompts I used:

- "Explain in simple terms what an audit log should record for a small web app that tracks door access."
- "Give me a few example rules for suspicious door access)."
- "How can we build and integrate two parts of the project, the facial recognition and anomaly detection"
- "Show a tiny example that renders a page with a list of fake log entries."

These prompts were starting points; I always edited and adapted the responses so they fit our project and what we learned from Armin.

Short Reflection

Overall, this period was about **understanding the problem and the hardware, and setting a realistic direction**. The most useful part was seeing the actual Raspberry Pi + lock + camera setup and being able to link our ideas about facial recognition, anomaly detection, and logging to something physical. Going forward, my main focus will shift from reading and planning to actually implementing simple threat detection rules and the logging pieces we've been discussing, with compliance in mind from the start