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

Student Name: Reubin Chatta
Student ID: 300394193
Project: Smart Door Security Framework with Anomaly Detection & Cybersecurity for Elderly Care
Course: CSIS 4495 – Applied Research Project, Section 3
Reporting Period: Jan 19 – Feb 9, 2026
Report Date: February 9, 2026

Work Date/Hours Logs

Date	Hours	Description of Work
Jan 27, 2026	1	Looked at examples of simple web dashboards to get ideas for how we could display door access logs and alerts without overcomplicating the design.
Jan 29, 2026	1.5	Helped review and adjust the email to Armin so that our description of the dashboard and overall project direction matched what we discussed as a team.
Jan 30, 2026	1.5	Read a few beginner-friendly articles/tutorials on building small Flask-based dashboards and noted which patterns would fit our project.
Feb 3, 2026	2.0	In-person meeting with Armin: looked closely at the Raspberry Pi, door lock, and Ring-style camera prototype video; discussed how a caregiver might interact with a dashboard connected to this setup.
Feb 3, 2026	1.5	Team debriefs after hardware meeting: talked about how data might flow from the Pi and camera into a web view; sketched a very rough idea of the screens we might need.
Feb 4, 2026	1.5	Read more on basic Flask app structure and template organization; wrote down a simple layout plan for a main dashboard page and a login page.

Date	Hours	Description of Work
Feb 5, 2026	1.5	Team call: discussed integration points between the dashboard, threat detection logic, and anomaly detection; noted what APIs/endpoints will be needed from the backend.
Feb 6, 2026	1.5	Built a very small local test page to see how a log table and an alert section could look; this was only for practising layout, not final code.
Feb 7, 2026	1.0	Read a short article on making dashboards clearer for non-technical users and wrote notes on how to keep our caregiver interface simple.
Feb 8, 2026	1.5	Short team sync: walked through my rough dashboard layout, took feedback from Advitiya and Eric, and aligned on keeping the first version very basic.

Total Hours This Period: 17 hours

Summary Description of Work Done During This Reporting Period

During this period, my focus was on the **integration and dashboard side** of the project. In January, I took part in the general brainstorming and helped push towards a project where a simple, clear dashboard would be useful to caregivers and match Door Face Panels' product vision.

The **Feb 3 meeting with Armin's** was important. Seeing the **Raspberry Pi**, **lock**, and **camera prototype** made it much easier to imagine the actual user flow: a resident or caregiver uses the door, the Pi logs the event, and the dashboard later shows who came and when, plus any alerts. After that, most of my time went into:

- Looking at simple examples of **web dashboards** and Flask-based apps.
- Thinking about what a **caregiver actually needs to see** (logs, alerts, status) and how to avoid clutter.
- Sketching a basic layout (on paper and with a small HTML/CSS test) for how logs and alerts might appear on one page.
- Writing notes about which **API endpoints** we are likely to need from the backend (for logs, alerts, maybe a simple settings area later).

There is no real integrated product yet and no polished UI. Right now, I am focusing on structure: what screens we need, what data they show, and where that data comes from.

Issues Encountered and How We Handled Them

- **Risk of overcomplicating the dashboard**
  It's tempting to add charts, filters, and lots of features. To keep the project realistic, we agreed to focus the dashboard on **just the essentials**: a log view, an alert area, and maybe a very basic status indicator.
- **Unclear data flow before seeing hardware**
  Before the Feb 3 meeting, it was hard to imagine how data would move from the door to the UI. Seeing the Pi and the lock, and hearing Armin describe the expected behaviour, helped clarify that the dashboard will likely just read from a local log/DB on the device (no big distributed system in this phase).
- **Balancing technical detail with caregiver usability**
  Some of the information (like anomaly scores) is technical. We discussed as a team that we should show caregivers **plain-language alerts** (e.g., "Unusual access time") instead of raw model outputs.

Changes to Proposal / Plan

No formal changes were made to the written proposal, but we clarified for the dashboard:

- The first version will be **simple and practical**, focused on logs and alerts rather than advanced visualizations.
- The dashboard will be designed with **non-technical users** in mind (caregivers), so wording and layout should be straightforward.
- Integration-wise, the dashboard will primarily call a small set of backend APIs to fetch logs and alerts, rather than trying to control everything directly.

Updated Individual Timeline

Next 1–2 weeks (up to Midterm):

- Turn the rough layout ideas into a small Flask-based skeleton app (routes + templates, still using fake data at first).
- Agree with Advitiya and Eric on API endpoint names and basic JSON formats for logs and alerts.
- Start wiring the dashboard to use sample data from the backend once those endpoints are available.

- Keep the UI minimal so we can focus on making sure the data flow and integration work correctly before polishing.

Repo Check-In of Implementation Completed

So far, my contributions have mostly been **design and exploratory front-end work**, not a finished dashboard. Examples:

- **Docs / Planning**
  - Added a short design note in the documents folder describing the planned dashboard layout (sections for logs, alerts, and basic status).
  - Wrote down a simple list of planned API calls (e.g., /logs, /alerts) and what each should return in general terms.
- **Exploratory UI Code**
  - Created a tiny HTML/CSS test file (and possibly a simple Flask route) just to try out a possible layout for logs and alerts.
  - This is clearly marked as experimental and will likely be replaced or heavily revised as the backend and real data shape up.

There is **no fully integrated dashboard** yet, which matches our current project phase.

AI Use Section

Tools I Used and How

- **Chat-based AI (e.g., ChatGPT / Gemini)**
  - To get ideas for **simple dashboard layouts** for log/alert pages.
  - To ask for plain-language explanations on how to make dashboards easier for non-technical users.
  - To rephrase some of my notes and proposal text so they were clearer.
- **Code suggestion tool (e.g., Copilot)**
  - Used lightly while experimenting with a small HTML/CSS/Flask test page.
  - Helped with basic boilerplate (routes, template structure), but I still made manual changes based on what our project actually needs.

What I Did Myself

- Decided which layout and wording ideas actually made sense for our specific use case (elderly care caregivers), instead of copying generic admin dashboard designs.
- Adjusted any suggested code to match our planned endpoints and data, rather than using examples as-is.
- Took into account what we saw at Armin's office when thinking about what information is realistic and useful to display.

Examples of Actual Prompts (Appendix)

Some example prompts I used:

- "Show me a simple HTML layout for a page with a table of log entries and an alert box at the top."
- "Tips for designing a web dashboard that is easy for non-technical users to understand."
- "Basic Flask example with one route that renders a template with a table of sample data."

These were used as starting points; I edited the results to fit our project's context and our team's ideas.

Short Reflection

For me, this period was about figuring out **what the dashboard should actually do** and how it will plug into the rest of the system. Seeing the hardware and talking with Armin helped a lot, because it made the user flow and constraints much more concrete. Going forward, I want to move from sketches and mockups to a small but working dashboard skeleton that can talk to the backend once the logging and threat detection pieces come online.

Date: February 9, 2026