**Reflection Report**

**Student Name:** Arun Ragavendhar Arunachalam Palaniyappan  
**Student ID:** 104837257  
**Unit Code:** COS70008  
**Submission Date:** 2 June 2025

**Table of Contents**

1. Introduction
2. Project Reflection  
   a. Group Work Reflection  
   • Strategies/Processes that Worked  
   • Strategies/Processes that Did Not Work  
   • Improvements for Future Group Work  
   • Outstanding Events/Actions  
   b. Individual Work Reflection  
   • Tasks in Each Phase of the Project  
   • Contributions to the Group  
   • Involvement in Team Environment  
   • Conclusion and Recommendations
3. Lessons Learnt
4. Conclusion and Future Recommendations
5. References

**1. Introduction**

This report is a reflection of the individual and team experiences gained over 12 weeks as part of the Technology Innovation Project unit. The main goal is to present key observations, work completed, and learning outcomes during the entire project timeline. It highlights the teamwork processes used, individual roles played, and areas that can be improved in future group-based academic or industry projects. All insights are drawn from real activities, meetings, and outcomes achieved during Weeks 2 to 11.

**2. Project Reflection**

**a. Group Work Reflection**

**Strategies/Processes that Worked:**

* Work was divided based on each team member's strengths (e.g., backend, frontend, ML models).
* Shared folders and cloud repositories were used to store and update documents and code.
* Weekly team sync-ups helped monitor progress and quickly solve issues.
* Active group chat ensured fast and flexible communication.
* Feedback loops after each sprint helped refine ongoing tasks.

**Strategies/Processes that Did Not Work:**

* Design selection took longer than expected in the first phase.
* Not all members contributed equally to early documentation.
* Task tracking was not formally maintained (e.g., via Trello or Gantt charts).

**Improvements for Future Group Work:**

* Define specific timelines for each phase with deadlines early in the semester.
* Assign a lead for each sprint to monitor progress and ensure task clarity.
* Include a checklist during reviews to ensure nothing is missed.
* Allocate buffer time for integration and last-minute testing.

**Outstanding Events/Actions:**

* **Team Organisation:** The team created a detailed charter in Week 2 outlining roles, rules, and deliverables.
* **Meetings:** Week 6 included a full design review session where all five designs were discussed, and three were finalised.
* **Delivery of Project Design Ideas:** Arun’s design was selected and actively supported with ideas and resources by all team members.
* **Delivery of Innovation Concept:** In Week 9, all members contributed to building and proofreading the pitch deck.
* **Delivery of Final Presentation:** In Week 11, Arun presented the live demonstration, and other members handled UI and documentation walkthroughs.

*Part b. Individual Work Reflection* and remaining sections will follow next.

**2b. Individual Work Reflection**

**Tasks in Each Phase**

**Weeks 2–4: Identifying Learning Gaps**

* Found gaps in my understanding of combining two machine learning models and using public malware datasets.
* Studied how to build a hybrid ML pipeline using Autoencoders and Random Forests.
* Explored the CIC-MalMem2022 dataset, learned its structure, and confirmed it was reliable and publicly available.

**Weeks 4–6: Designing Architecture and Prototyping**

* Designed the complete system architecture for CyberShield AI (Design 1).
* Created Figma prototypes covering all user flows including login, scanning, analytics, and retraining.
* Documented all technical decisions and system specifications.

**Week 7–10: Implementation**

* Trained and validated the hybrid ML model using the chosen dataset.
* Built REST APIs using Flask to connect the ML model with the frontend.
* Created the frontend using Vue.js with Tailwind CSS, implementing all the UI components as per Figma design.
* Developed the MySQL database schema to store user data and scan records.

**Week 10–11: Testing and Final Review**

* Conducted testing using sample malware and benign files.
* Debugged issues in prediction, model loading, and analytics dashboard.
* Finalised the demonstration and presented it with a working system and source code.

**Contributions to the Group**

**Overall Team Support**

* Took full ownership of Design 1 – CyberShield AI, completing all components end-to-end.
* Shared learnings about hybrid models and dataset handling with team members.
* Guided peers on how to structure implementation tasks and set up Flask and Vue integration.

**Collaboration and Documentation**

* Acted as the main writer for multiple sections of the group’s Innovation Concept Report.
* Created reusable documentation templates for system architecture and feature tables.
* Supported others during presentations by giving clear walkthroughs of design and implementation.

**Team Environment**

* Helped resolve doubts during model training and UI setup.
* Stepped in to motivate the team during mid-semester weeks when tasks were pending.
* Reviewed and validated other designs to ensure alignment with client requirements.

**3. Lessons Learnt**

During the 12-week project, several useful lessons were learnt that helped the team work better:

* **Form teams early**  
  Starting early gave us more time to plan roles, understand the task, and avoid confusion later.
* **Assign clear roles**  
  A team charter helped divide tasks properly. Everyone knew what they were responsible for.
* **Track progress weekly**  
  Weekly updates made it easier to see what was done, what was left, and where help was needed.
* **Communicate often**  
  Regular meetings and a shared group chat helped the team stay connected and solve issues quickly.
* **Use time wisely**  
  Starting tasks early allowed more time for testing and fixing. Rushing near deadlines caused stress.
* **Help each other**  
  Supporting others with design, testing, or documentation helped the whole team move forward.
* **Decide quickly**  
  Making design and tool choices early helped avoid delays during development.
* **Raise issues early**  
  Talking about problems during meetings helped solve them quickly without slowing down progress.
* **Follow client needs**  
  We kept checking the client’s requirements so we didn’t add extra or unnecessary features.
* **Work as a team**  
  Even when working on different parts, helping each other improved the final quality of the project.

These lessons made the project more organised and successful, and they will be useful in future group work.

**4. Conclusion and Recommendation**

This project aimed to meet the client's need for a working malware detection web system using hybrid machine learning. I successfully designed and developed **CyberShield AI – Design 1**, which detects both known and unknown malware, classifies them into types (trojan, ransomware, spyware), and provides a scan report with risk scoring. The system fully meets the client’s requirements:

* It uses a hybrid ML model (Autoencoder + Random Forest) for better detection of obfuscated threats.
* It provides clear, easy-to-understand results for users and admins.
* It includes a full admin dashboard for scanning history, model retraining, and analytics.
* It is built as a working three-tier web application, ready to deploy.

All parts of the client brief—detection, classification, behaviour-based analysis, scan history, and model updates—were addressed through the design, Figma prototype, and final implementation.

**Recommendations**

To take this project further and improve its real-world usability, the following practical upgrades are suggested:

* **Enable Cloud Hosting**: Move the system from local server to cloud (e.g., AWS or Azure) to allow public access and remote usage.
* **Add Real-Time Detection**: Integrate file and memory monitoring to detect threats while the system is running, not just from uploaded files.
* **Support More File Types**: Add support for scanning .exe, .docx, and .pdf files by safely extracting features before passing them to the model.
* **Use Larger, Multi-Class Datasets**: Train the model on updated datasets that include more malware families for wider detection coverage.
* **Integrate Threat Intelligence**: Connect to live feeds like VirusTotal or MISP to enhance results with updated threat information.
* **Add Sandboxing for Behaviour Tracking**: Build a basic dynamic analysis module to safely observe how suspicious files behave in a controlled environment.
* **Improve User Management**: Include features like role-based access control, password recovery, and login history for better security.
* **Track Model Versions**: Store version history of trained models and allow rollback in case a retrained model causes issues.

These steps will help move CyberShield AI from a working university prototype to a scalable and dependable tool ready for real-world malware detection use cases.