Project plan

1. Project title

Network Traffic Profiler Dashboard

2. Project Overview

This project aims to develop an interactive dashboard that helps users understand their network traffic by uploading PCAP files. The focus is creating an AI model that can detect normal and abnormal activity, classify different types of traffic, and clearly label them. The dashboard will then display the results through simple visuals and easy-to-read summaries.

3. Project scope

The project focuses on developing a network traffic analysis system that leverages machine learning to classify and visualize network activities. The final product will include several key components. Data input will allow users to upload a network traffic PCAP file. Data processing will extract essential features such as the number of packets, average packet size, session duration, and transport protocol. The data will then be cleaned, filtered, and structured into an analysis-ready CSV dataset. Machine learning model training will involve loading the pre-processed dataset to classify network activity types into known activities or anomalies. Finally, dashboard visualization will present traffic statistics and machine learning classification results in a clear and intuitive format.

The system will be designed for small and medium-sized enterprises (SMEs) with up to 50 users as well as supporting the upload and analysis of a single PCAP file at a time to maintain manageability within the defined scope. Future development may extend its scalability to support for large organizations and enable multi-file ingestion. Due to hardware limitations and the unavailability of large-scale datasets, the project will operate with limited machine learning training capabilities. The project also explicitly excludes real-time packet capture and processing of files larger than 500MB, although these features could be considered in future iterations.

4. Project objectives

- Develop an AI model capable of detecting normal and abnormal network activity and classifying different types of traffic.
- Build an interactive dashboard that can display the AI’s results through clear visuals and summaries.
- Ensure the system can process uploaded PCAP files reliably and efficiently.

- Prepare the dashboard and AI model so they can be integrated into a larger project: an AI-driven firewall intrusion detection system designed for small to medium-sized businesses.

5. Stakeholders

Team Members:

The main group responsible for designing, developing, and testing the system. Includes all four team members who contribute to coding, planning, documentation, and quality assurance.

Client (Plymouth City Council): Tomek Bergier

Sprint Leader: Fatma
Oversees individual sprints, ensuring tasks are completed efficiently and on time, coordinating the team to achieve sprint goals.

End Users:

Individuals who will use the final dashboard to analyse PCAP files and detect abnormal network activity.

Future Developers:

The group who will integrate this project into a full AI-driven firewall/IDS. They depend on clear documentation, clean code, and stable foundations.

6. Project Team

Sophia

• **Product Owner**: Defines the product vision, sets priorities, and clarifies requirements.

• Architect: Designs the system structure, including tech stack, folder layout, and key technical decisions.

• Technical Expert: Supports the team in solving technical challenges and choosing effective solutions.

Relationship: Prioritises tasks, assigns coding responsibilities, and works with the Team Leader and Scrum Master to plan workloads and keep requirements clear.

Amelia

• **Team Leader**: Coordinates workflow, monitors To-Do list, removes bottlenecks, and manages the Kanban board.

• Innovator/Explorer: Brings in creative ideas and explores new approaches to tasks.
• Developer: Responsible for programming and testing the product.

Relationship: Team members coordinate with Amelia for day-to-day workflow and task organisation.

Tim

• **Scrum Master**: Ensures Scrum and Agile processes are followed; leads stand-ups and sprint planning.

• Developer: Contributes to coding and testing.

• Independent Tester/Auditor: Responsible for thorough testing to identify issues early and ensure they are resolved, helping the product meet expected quality standards.

Relationship: Works closely with all team members to support collaboration, ensure Scrum practices are followed, and keep the development process running smoothly.

Tomiris

• **Secretary**: Takes meeting minutes and handles team communication and documentation.

• Independent Tester/Auditor: Responsible for thorough testing to identify issues early and ensure they are resolved, helping the product meet expected quality standards.

• Innovator/Explorer: Brings in creative ideas and explores new approaches to tasks.

Relationship: Supports the team by maintaining documentation and assisting wherever needed.

7. Timeline

General Timeline:

Sprint 1 [3 Nov – 15 Nov 2025]

Week 1: Defining team Roles, Project scope, Backlog establishment and Task qssignment
Week 2: Tech stack and solutions research, Data abstraction, Preprocessing and Machine learning research.

Sprint 2 [16 Nov – 30 Nov 2025]

Week 3: Implement PCAP file upload / input system

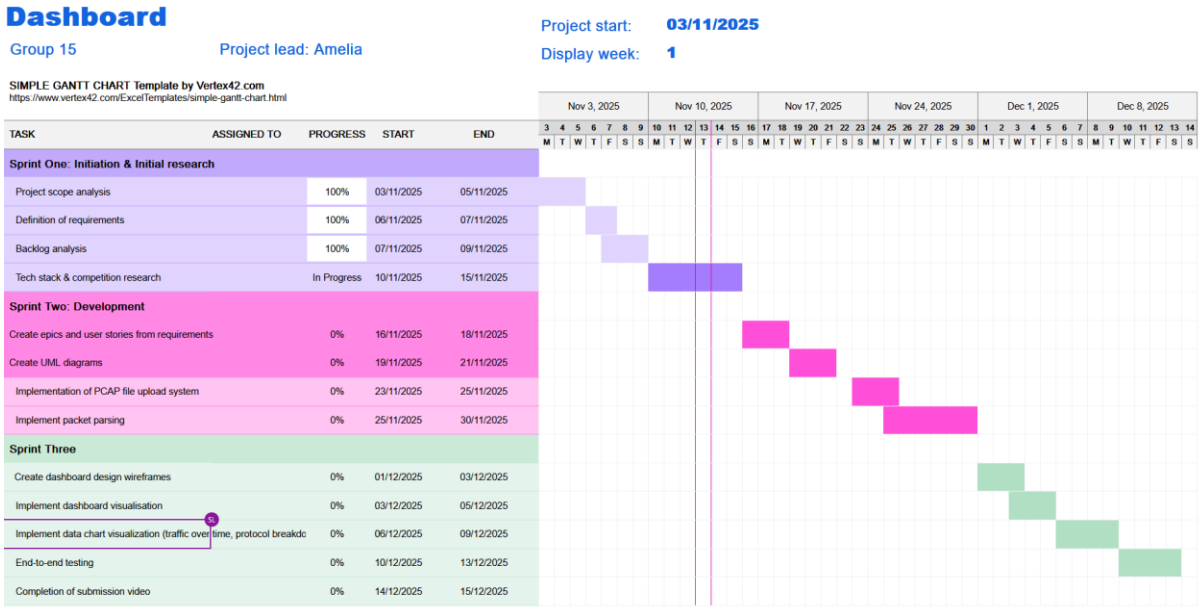
Week 4: Implement basic packet parsing (conversion of raw packets into structured records), Abstraction of relevant information from packets

Sprint 3 [1 Dec – 15 Dec 2025]

Week 5: Dashboard wireframes, Implementation of dashboard/ data chart visualisation

Week 6: Complete prototype, End-to-end testing, Internim Video filming and submission

Ghant Chart (Current):



Dashboard

Group 15 Project lead: Amelia

Project start: 03/11/2025

Display week: 1

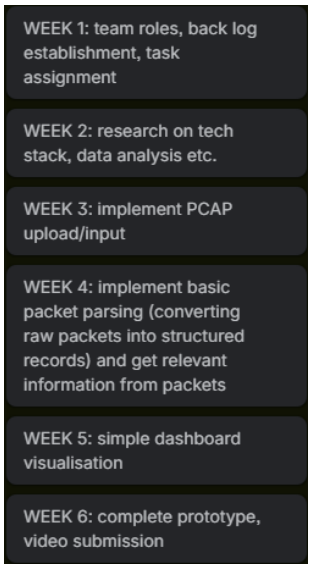
SIMPLE GANTT CHART Template by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

TASK	ASSIGNED TO	PROGRESS	START	END
Sprint One: Initiation & Initial research				
Project scope analysis		100%	03/11/2025	05/11/2025
Definition of requirements		100%	06/11/2025	07/11/2025
Backlog analysis		100%	07/11/2025	09/11/2025
Tech stack & competition research		In Progress	10/11/2025	15/11/2025
Sprint Two: Development				
Create epics and user stories from requirements		0%	16/11/2025	18/11/2025
Create UML diagrams		0%	19/11/2025	21/11/2025
Implementation of PCAP file upload system		0%	23/11/2025	25/11/2025
Implement packet parsing		0%	25/11/2025	30/11/2025
Sprint Three				
Create dashboard design wireframes		0%	01/12/2025	03/12/2025
Implement dashboard visualisation		0%	03/12/2025	05/12/2025
Implement data chart visualization (traffic over time, protocol breakdc		0%	06/12/2025	09/12/2025
End-to-end testing		0%	10/12/2025	13/12/2025
Completion of submission video		0%	14/12/2025	15/12/2025

From Semester 2 Onwards:

- Design and Implementation on ML/AI model training
- Dashboard enhancement with Evaluation Metrics

Backlog in Trello:



8. Research

Existing tools for network traffic analysis include traditional IDS solutions like Snort, Suricata, which focus on live monitoring and signature-based detection. While powerful, they require complex setup and are aimed at experts.

Tools for displaying network statistics, such as Arkime and Malcolm, support PCAP uploads and implement dashboard views, but they require network understanding and require to be set up.

A simpler tool that allows to upload a PCAP file and display stats on a simple dashboard on a web UI is A-Packets, however it doesn't classify and differentiate between activity within the data.

Our project is therefore unique, because it fills the gap between expert platforms and overly simple visualisation platforms, by combining:

- a direct PCAP file upload.
- an AI model that classifies normal vs abnormal traffic and labels activity types.
- a user-friendly dashboard aimed at non-experts.
- a very simple tech stack, built using minimal open-source Python libraries, making it lightweight, accessible, and easy to integrate into larger systems.

9. Proposed Solution

The Network Traffic Profiler Dashboard is developed using modern, well-supported Python libraries to ensure simplicity, maintainability, and effective performance. The dashboard is implemented using Streamlit, allowing for easy local deployment, and the project leverages a structured approach to parsing, processing, validating, and visualizing network traffic data.

Development Technology Stack (Python Libraries):

- PCAP Parsing & Feature Extraction: pyshark, NFStream
- Feature Storage: pandas (CSV format)
- Dashboard Development: Streamlit
- Data Visualization: Plotly
- Testing: pytest
- Machine Learning: scikit-learn

The project workflow for sprint 1 to 3 is structured into four main components with their solutions.

1. PCAP Parsing and Feature Extraction: Raw PCAP files are loaded using PyShark or NFStream. Key network features are extracted and returned as a Python dictionary. Outputs include a sample JSON file and a documented list of available features.
2. Dataset Building: Extracted features are aggregated from raw datasets of normal and abnormal traffic. Data is automatically labelled based on folder structure and saved in CSV format for analysis.

3. Validation and Schema Design: Dataset rows are validated for required columns, data types, and numeric ranges. Validator scripts check for missing values and anomalies, ensuring data quality. Outputs include field schema documentation and validation rules.

In the following sprints in semester 2, machine learning will be implemented.

10. WBS (Clear subtasks)

1. Project Management (Pre-Sprint)

1.1 Project overview

1.2 Trello setup

1.3 GitHub repository setup

1.4 Discord server setup

1.5 Role assignment

2. Requirements & Design (Sprint 1)

2.1 Individual research

2.2 Requirements definition

2.3 Project timeline creation (GANTT chart)

2.4 Project plan documentation

2.5 Design document creation

2.6 Tech stack definition

2.7 Initial folder structure setup

3. PCAP Processing Pipeline (Sprint 2)

Team A: extract_features.py, build_dataset.py

3.1 Read PCAP files

3.2 Extract key traffic features (packet counts, protocols, sizes, domains, IPs)

3.3 Save extracted data into CSV format (ready for ML)

Team B: process_dataset.py, README.md, requirements.txt, /tests

3.4 Define dataset schema and validation rules

3.5 Create basic validation script (check missing values, invalid numbers)

3.6 Add dependencies file including python libraries

3.7 Add basic tests

4. Dashboard Implementation, working MVP and interim submission (Sprint 3)

dashboard.py

4.1 Set up Streamlit layout (based off of wireframes)

4.2 Add PCAP upload interface

4.3 Display statistics in simple charts

4.4 Test workflow and ensure everything is working

4.5 Record video for interim submission

4.6 Interim submission

11. Resource Plan

The Resource Plan outlines the human, technical, financial, and data resources required to complete the project effectively.

Human Resources

- Project Team Members: Roles and responsibilities as defined in Section 6
- Client Involvement: Availability for requirements meetings, Providing feedback throughout development
- Lecturer Support: Academic and technical guidance, Oversight of project milestones, Evaluation and quality assurance

Technical & Equipment Resources

- Hardware: Personal laptops/workstations, Reliable internet connection
- Project Storage and Repository: GitHub repository for code, documentation, reports, meeting minutes, and version control

Development Technology Stack (Python Libraries):

- PCAP parsing: pyshark
- Feature extraction: nfstream
- Data storage (CSV): pandas
- Dashboard development: Streamlit
- Data visualisation: Plotly
- Testing: pytest
- Machine Learning: scikit-learn

Data Resources:

- Sample PCAP datasets provided by the client
- Additional self-generated or publicly available PCAP datasets
- CSV data files for model training, dashboard visualisation, and analysis.

Financial Resources:

- As this is a student project, development is free for both the team and the client.
- All tools, libraries, and frameworks used are open-source and cost-free.

Potential Future Maintenance Costs (beyond project scope):

- Dashboard hosting
- Optional cloud compute resources
- Security patches and updates
- Integration with the client's firewall system

12. Risk Management

PCAP files may be too large or fail to load: Prompt user to upload smaller file.

Feature extraction doesn't work on the PCAP file (e.g. malformed packets, lacks relevant traffic): Catch error early and report back to the user.

The ML model does not perform well: Test different models, clean the dataset properly. In worst case, switch to a simple anomaly-detection approach if accuracy is too low.

GitHub or code conflicts slow down development: Use branches, commit regularly, revert to a working commit if code becomes unstable.

Missing or unclear documentation: Update the README whenever new features are added, comment the implemented code, explain code to each other.

Changes in requirements: Adapt using Agile methodology, avoid hardcoding.

13. Communication plan

Discord Server Within the Team: Used for overall project organization and quick communication among team members.

Trello Board Within the Team: Tracks task division and progress for each team member on a weekly basis.

Weekly Stand-up Meetings Within the Team: Held on Mondays and Thursdays, with Saturdays as additional meetings if needed. These meetings primarily cover: Progress made (tasks completed), Upcoming tasks (what to do next), Issues or challenges encountered, Division of subtasks among team members

Bi-Weekly Sprint Review Meetings with the tutor (Fatma): Held on Mondays from 12:40 PM to 1:00 PM at Seamstons Building, SMB 100. Key discussion points include progress made during the current sprint, any issues or blockers, updates and planning for the next sprint

Weekly Client Meetings with the client (Tomek Bergier): Held on Thursdays in person at Portland Square, Room A03, or via Teams if necessary. General meeting objectives includes sharing work progress to ensure alignment with client requirements, confirming that the next steps planned meet client expectations and discuss any considerations, concerns, changes that need to be made.

14. Quality Management

Code Quality

- Abstraction by splitting up code in smaller files.
- Consistent formatting and naming across Python files.
- Catch errors early through exception handling.

Version Control Practices

- Regular GitHub commits on separate branches.
- Team members review each other's code before merging into main branch.
- Clear commit messages to track and understand changes.

Sprint Reviews

- After each sprint, review what worked and what didn’t.
- Adjust tasks and improve code where necessary.
- Make sure each sprint ends with a stable and working version of the system.

Testing

- Unit tests for critical functions (e.g., feature extraction, dataset processing).
- Test PCAP parsing with several sample files to ensure stability.
- Test the dashboard manually to confirm that uploads, charts, and outputs work correctly.
- Test the ML model after training, ensuring predictions are accurate.

UI/UX principles

- The dashboard should be simple, clean, and easy to navigate.
- Text outputs and charts must be readable and clearly labelled.
- Errors (e.g., invalid PCAP file) should be clearly displayed.

Documentation

- Up-to-date README with installation steps and usage instructions.

15. Monitoring and evaluation

Project progress will be monitored through consistent communication, structured Agile practices, and performance indicators. The following methods are used to ensure the project remains on track and aligned with the client expectations:

Client Check-ins: Weekly meetings with the client to gather feedback, update progress, and ensure requirements are being met.

Gantt Chart Tracking: Ensuring that each sprint milestone is achieved according to the project timeline.

Sprint Monitoring: Ensuring that each sprint meets its defined goals and deliverables.

Team Stand-up Meetings: For the team to track progress, identify blockers, and plan immediate next steps.

Requirement Alignment: Continuous review of project requirements to confirm that all development aligns with agreed features and constraints.

Peer Cross-Checking: Team members regularly review each other's work to ensure accuracy, quality, and consistency.

Quality of Data Output: Monitoring the integrity and accuracy of extracted features and ML outputs.

***Co*ntinuous & Meticulous Testing:** Using both normal and abnormal traffic data to evaluate and refine AI performance, while applying consistent and meticulous testing during each sprint to maintain quality.

These Key Performance Indicators (KPIs) will be used to measure project progress and team effectiveness:

Sprint Goal Completion Rate: Achieve ≥ 80% completion of planned sprint tasks on time.

Task Progress Adherence: Achieve a target of ≥ 90% of tasks that is being updated and progressed through “To Do,” “In Progress,” and “Done” on the Trello board according to the planned timeline.

Team Communication Efficiency: Maintain ≥ 90% attendance and active participation in stand-ups, sprint reviews, and team meetings.

Issue Resolution Rate: Resolve a ≥ 85% of identified issues and challenges within the same sprint which they are identified.

16. Budget

This project is designed to be low-cost, with a strong focus on using open-source technologies and freely available tools. Because the system is built entirely with Python and community-maintained libraries, there are no licensing fees or commercial software costs involved.

17. Approval Process

Clear and consistent communication with the client will be maintained throughout each sprint via email and weekly client meetings. The main areas requiring client approval during the project includes Project Requirements, Milestones, Design/Architecture, and the Final Product.

Relevant documents, including links or files for review, will be sent to the client via email and discussed during the client meetings. Approvals will be properly documented and acknowledged using a combination of email, meeting discussions, and meeting minutes to ensure clarity and traceability.

All approvals will be confirmed through written acknowledgment, either via email or signed documents, to prevent misunderstandings. Any changes requested by the client will be reviewed and formally approved before implementation, ensuring alignment with client expectations throughout the project lifecycle.

client’s approval/signoff*

18. Change Management

Our agile approach allows the team to respond effectively to evolving needs.

1. Record the change request
2. Evaluate the impact on work and timeline
3. Approve or decline the change
4. Update sprints and Trello
5. Implement the change
6. Review and document the results

19. Closure and Evaluation

Project Closure:

- Conduct thorough end-to-end testing to verify that the final product is fully functional and meets all specified requirements
- Ensure all project objectives, requirements, and deliverables are fully completed, verified against the project specifications, and finalised.
- Record all client approvals, including signed UAT forms or endorsement letters)
- Finalise all documentations (reports, code, design diagrams, presentation materials, and meeting minutes) and ensure they are complete, properly formatted, accessible, and stored in the GitHub repository.
- Carry out a formal client handover, including: A walkthrough meeting, Provision of all relevant files, Delivery of a client instruction manual (installation, usage, troubleshooting
- Prepare all final presentation materials, including the project poster, slides, team member contribution form, and demonstration video for the showcase.

Post-Implementation Review:

- Hold two review meetings: One involving both the client and project team, One internal review within the project team.
- Compare actual project outcomes against initial objectives, success criteria, and KPIs.
- Identify any unresolved issues, outstanding tasks, or opportunities for future development.
- Gather feedback from both the client and project team on the project process, deliverables, and overall performance.

Lessons Learned

- Document key successes and challenges encountered during the project lifecycle.
- Record insights on what worked well and areas needing improvement in planning, execution, teamwork, and communication.
- Provide recommendations for future projects, including technical, managerial, and procedural improvements.

20. Appendices