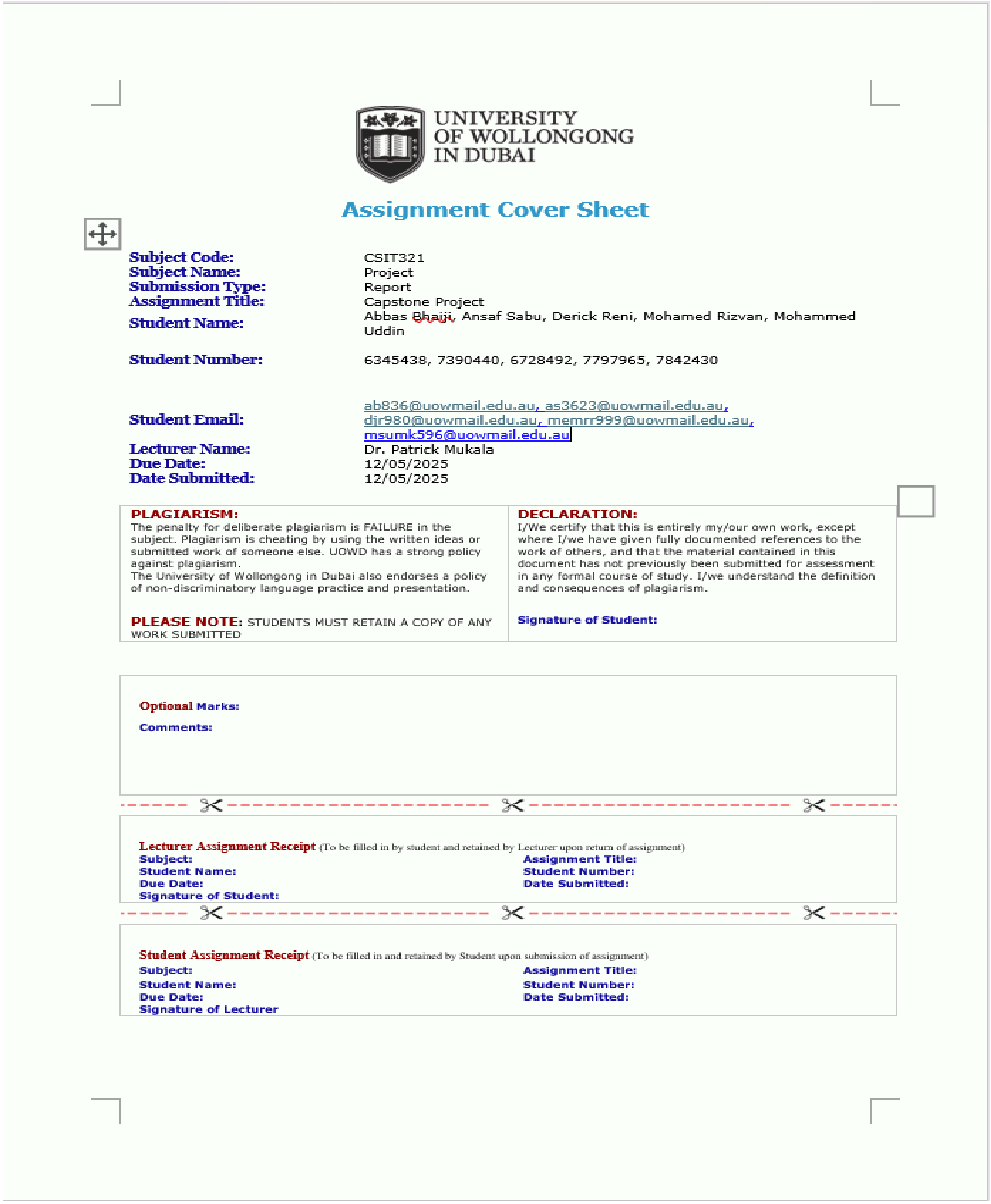
ARMOUREYE

AI - DRIVEN PENTESTER

- BY DEEPTHINK

ASSIGNMENT COVER SHEET



UNIVERSITY OF WOLLONGONG IN DUBAI

Assignment Cover Sheet

Subject Code:	CSIT321
Subject Name:	Project
Submission Type:	Report
Assignment Title:	Capstone Project
Student Name:	Abbas Bhaiji, Ansaf Sabu, Derick Reni, Mohamed Rizvan, Mohammed Uddin
Student Number:	6345438, 7390440, 6728492, 7797965, 7842430
Student Email:	ab836@uowmail.edu.au, as3623@uowmail.edu.au, djr980@uowmail.edu.au, memrr999@uowmail.edu.au, msumk596@uowmail.edu.au
Lecturer Name:	Dr. Patrick Mukala
Due Date:	12/05/2025
Date Submitted:	12/05/2025

PLAGIARISM:
The penalty for deliberate plagiarism is FAILURE in the subject. Plagiarism is cheating by using the written ideas or submitted work of someone else. UOWD has a strong policy against plagiarism.
The University of Wollongong in Dubai also endorses a policy of non-discriminatory language practice and presentation.

PLEASE NOTE: STUDENTS MUST RETAIN A COPY OF ANY WORK SUBMITTED

DECLARATION:
I/We certify that this is entirely my/our own work, except where I/we have given fully documented references to the work of others, and that the material contained in this document has not previously been submitted for assessment in any formal course of study. I/we understand the definition and consequences of plagiarism.

Signature of Student:

Optional Marks:

Comments:

Lecturer Assignment Receipt (To be filled in by student and retained by Lecturer upon return of assignment)

Subject:	**Assignment Title:**
Student Name:	**Student Number:**
Due Date:	**Date Submitted:**
Signature of Student:	

Student Assignment Receipt (To be filled in and retained by Student upon submission of assignment)

Subject:	**Assignment Title:**
Student Name:	**Student Number:**
Due Date:	**Date Submitted:**
Signature of Lecturer	

Tables Of Contents

Project Title

ArmourEye - AI Driven PenTester

Capstone Team Members

Student ID	Student Name	Roles
6345438	Abbas Bhaiji	Team Leader AI/ML Engineer
7390440	Ansaf Sabu	Scribe AI/ML Engineer
7842430	Mohammed Uddin	Team Member Business Sys. Lead
7797965	Mohamed Rizvan	Team Member CyberSec Lead
6728492	Derick Reni	Team Member CyberSec Lead

Project Leader

Abbas Bhaiji - AI/ML Engineer

Project Scribe

Ansaf Sabu - AI/ML Engineer

Mentor Details

Dr Mouhannad Alattar - Cyber Security Mentor

Project Description and Objectives

What is our project?

Armour Eye is an AI-powered platform that automates the penetration testing lifecycle. It is designed to simulate the process a human red team would follow, including:

- Reconnaissance and scanning
- Exploit identification and deployment
- Post-exploitation analysis
- Patching recommendation and fallback handling
- Risk reporting and visualization

The project integrates cybersecurity tools (like Nmap, ZAP, Metasploit) with AI models such as reinforcement learning agents and large language models to build an intelligent, self-improving penetration testing system.

Project Objectives

- Build a full-stack autonomous pentesting agent that mimics attacker workflows.
- Reduce the time and cost of security testing.
- Deliver business-relevant risk data via a user-friendly dashboard.

Main problems

- Penetration testing is largely manual, requiring skilled professionals and significant time.
- Existing tools are siloed and don't offer end-to-end automation.
- Traditional tools identify vulnerabilities but often fail to provide immediate patching or mitigation guidance, leaving a critical gap between discovery and remediation.
- Most tools lack adaptability. They follow static rule sets rather than learning from previous tests.

- Business stakeholders struggle to interpret technical security reports in a meaningful way.

How do we address them?

- Automation of the full pentesting lifecycle using AI agents to handle reconnaissance, exploitation, and reporting.
- Integration of ML and RL models to dynamically prioritize targets and select optimal exploit strategies.
- Use of LLMs to generate customized attack scripts.
- Deploying a scraping patching system that identifies and suggests or applies relevant fixes using real-time data from trusted vulnerability and patch databases.
- Interactive risk dashboards with React and Grafana that translate technical issues into business risk scores and SLA-driven action items.
- Modular architecture allows continual learning from previous test runs, improving efficiency and adaptability over time.

Project scope

The *Armour Eye* project focuses on developing an AI-driven, fully automated penetration testing system for Linux systems. The system is designed to replicate the workflow of a professional red team, covering all core stages: reconnaissance, vulnerability scanning, exploitation, post-exploitation, and reporting. Unlike traditional tools that require human oversight at each stage, ArmourEye aims to use machine learning, reinforcement learning, and large language models (LLMs) to execute these steps autonomously and intelligently. The scope includes integrating widely used open-source tools, orchestrating them with AI agents, generating business-oriented dashboards, and incorporating patch management suggestions or mitigation plans. The system is intended to be scalable, modular, and capable of learning from past tests to improve future performance. It will also include real-time analytics and a customizable dashboard interface for IT and business stakeholders, enabling risk-aware decision-making.

Background of the Study/Project

Cyberattacks on organizations have become more frequent, sophisticated, and damaging. Traditional cybersecurity defenses often rely on reactive mechanisms or scheduled manual assessments like penetration testing (pentesting) to identify vulnerabilities before attackers exploit them. Manual pentesting, however, presents challenges: it is labor-intensive, prone to human error, and cannot scale with modern, constantly evolving systems.

As organizations adopt complex architectures including cloud-native systems, microservices, and APIs the need for scalable, intelligent security validation is growing. With the advancement of machine learning (ML) and large language models (LLMs), there is now a viable path to automate the pentesting lifecycle. Automating this process can improve speed, consistency, and adaptability in vulnerability discovery and risk assessment.

Moreover, by integrating AI-driven insights with business-focused dashboards, organizations can bridge the gap between technical findings and executive-level decision-making. This project, Armour Eye, aims to fill that gap by building a fully autonomous, AI-orchestrated penetration testing framework.

Competitor Analysis

Competitor Solution and their Limitation

The landscape of penetration testing tools includes a mix of traditional manual utilities and newer AI-assisted platforms. On the manual side, widely used tools such as Nessus, Burp Suite, and Metasploit require skilled professionals to configure, operate, and interpret results. While powerful and customizable, they are not autonomous — users must manually connect scanning, exploitation, and reporting steps. These tools often produce raw vulnerability data that need translation into risk insights, and they lack mechanisms for learning from past assessments.

In the AI-driven space, solutions like RidgeBot (Ridge Security), Horizon3.ai, Pentera, and AttackIQ represent the new wave of autonomous or continuous security validation platforms. Horizon3.ai offers automated red-teaming but tends to rely on a predefined library of attacks, making it less adaptive to novel or customized attack surfaces. Pentera focuses on automated security validation and internal network testing but emphasizes compliance checklists over exploit chaining or AI-based reasoning. AttackIQ, while strong in breach and attack simulation (BAS), is designed primarily for validating defensive controls rather than performing true offensive security or live exploitation. It often assumes known attack behaviors rather than discovering new ones dynamically.

Despite their advancements, these platforms share some limitations: they are typically closed-source, offer limited integration with external tools, and lack the transparency or customization needed by advanced users. Most notably, they do not yet fully leverage large language models (LLMs) for custom payload generation, or AI-driven reporting. They also often fall short in translating technical findings into business-friendly remediation actions and dashboards.

Armour Eye differentiates itself by aiming for a fully automated, AI-orchestrated penetration testing framework that not only discovers and exploits vulnerabilities but also reasons through next steps, suggests or applies patches, and communicates findings through a business-oriented dashboard. Unlike its competitors, it uses reinforcement learning for exploit selection and fine-tuned LLMs for generating custom scripts and mitigation plans, making it more adaptable, explainable, and extensible.

Target Niche

Armour Eye targets a niche audience that includes startups, small companies lacking dedicated cybersecurity teams, and students or educational institutions seeking accessible, automated penetration testing solutions. By reducing the need for manual expertise and integrating AI-driven remediation, the platform empowers users with limited resources or experience to assess and improve their system security effectively.

Approach to the Project

The development of Armour Eye follows a modular, AI-driven methodology aimed at automating the entire penetration testing lifecycle. The project adopts a hybrid strategy that combines classical cybersecurity tools with advanced machine learning (ML), reinforcement learning (RL), and large language models (LLMs) to simulate and enhance red-team operations autonomously.

The approach begins with a top-down system design, breaking down the red team workflow into discrete stages: reconnaissance, scanning, exploitation, post-exploitation, patch recommendation, and risk reporting. Each of these stages is handled by a dedicated module orchestrated through containerization and managed via Kubernetes, allowing for scalability and fault isolation.

To ensure adaptability, the system integrates:

- Machine Learning classifiers for dynamic target prioritization during reconnaissance.
- Reinforcement Learning agents that learn optimal exploit paths over time, using feedback from previous test runs.

- LLMs fine-tuned on cybersecurity datasets to autonomously generate tailored payloads, craft custom attack scripts, and translate complex results into human-readable summaries.

The architecture prioritizes automation and repeatability, replacing manual decision points with intelligent agents capable of adjusting to new environments and threats. Integration with real-time CVE and patch databases enables the system to suggest or apply remediation steps, bridging the gap between vulnerability discovery and mitigation.

To make results meaningful beyond technical teams, interactive dashboards built with React present risk metrics and remediation suggestions in a business-friendly format. This empowers decision-makers with timely, actionable security insights without needing deep technical knowledge.

The entire platform is developed using containerized services to ensure reproducibility and deployment ease, with AI workloads offloaded to systems equipped with NVIDIA GPUs for efficient inference. Development follows an iterative agile model, continuously improving through user feedback, performance metrics, and reinforcement learning loops.

Skills/Knowledge Acquisition through the Project

Working on the Armour Eye project enables the development of a wide range of technical and conceptual skills across cybersecurity, AI, and software engineering domains. Key areas of knowledge acquisition include:

Cyber Security Fundamental and Tools

- Deep understanding of penetration testing methodologies and red-team operations.
- Hands-on experience with industry-standard tools such as Nmap, Metasploit, OWASP ZAP, Burp Suite, and Sqlmap.
- Familiarity with vulnerability scanning, exploit chaining, and post-exploitation techniques.

Artificial Intelligence & Machine Learning

- Application of reinforcement learning for exploit strategy selection and adaptive behavior.
- Fine-tuning and prompt engineering of large language models (LLMs) to generate scripts, summaries, and patch suggestions.
- Integration of ML classifiers for prioritizing attack surfaces and interpreting scan results.

Software Engineering & Automation

- Modular software design using Docker and orchestration via Kubernetes.
- Development of autonomous pipelines that integrate cybersecurity tools with AI workflows.
- Building robust backend services with Python, Flask, and orchestration frameworks.

Full Stack Development & Visualization

- Frontend development with React and TypeScript for building interactive dashboards.
- API design and integration for data flow between security modules and visualization layers.

Research & Problem Solving

- Investigation into open-source threat intelligence, vulnerability databases (CVEs), and patch repositories.
- Ethical and legal considerations in automated cybersecurity practices.
- Addressing challenges such as model accuracy, LLM reliability, and secure system integration.

Software and Hardware Requirements

Software Requirements

CyberSec Tools (From Kali)

- Nmap
- OWASP ZAP
- Metasploit Framework
- Wireshark
- Burp Suite
- UnicornScan
- Tcpdump
- John the Ripper
- Hydra
- Aircrack-ng
- Kismet
- Fern
- Bettercap
- Arpwatch
- Sqlmap
- Social Engineer Toolkit
- Netcat
- CrackMapExec
- Nikto

AI/ML Tools (Python)

- PyTorch
- Keras
- Tensorflow
- Hugging Face Transformers
- LangChain
- Scikit
- Pandas

Orchestration & Automation

- Google Colab
- Docker
- Kubernetes

Frontend Visualization

- React
- TypeScript
- Flask

Project & Workflow Management

- GitHub
- Notion

Hardware Requirements

System Setup

- Linux system
- Minimum: 8-core CPU, 8 GB RAM, 256 GB Storage

Specialized Hardware

- NVIDIA GPU (RTX 4080 or better) for AI inference
- Stable internet connection for model updates, CVE sync

Deliverables

The following are the key deliverables expected from the Armour Eye project, categorized into functional, technical, and documentation components:

Autonomous PenTesting System

- A fully functional, AI-powered platform that simulates the penetration testing lifecycle with minimal human intervention.
- Includes modules for reconnaissance, vulnerability scanning, exploitation, post-exploitation, and reporting.

AI Integration Components

- Reinforcement Learning agents for adaptive exploit path selection.
- Large Language Models (LLMs) for generating custom attack payloads and producing readable, accurate reports.
- ML classifiers to prioritize targets during reconnaissance and scanning.

Cyber Security Tool Orchestration

- Seamless integration with widely used open-source tools such as Nmap, OWASP ZAP, Metasploit, Sqlmap, Burp Suite, etc.
- Orchestrated using Docker containers and Kubernetes for modularity and scalability.

Patch Management System

- Real-time patch discovery system that maps CVEs to available fixes or mitigation strategies.
- Integration with public vulnerability databases and package repositories.

Interactive Dashboard

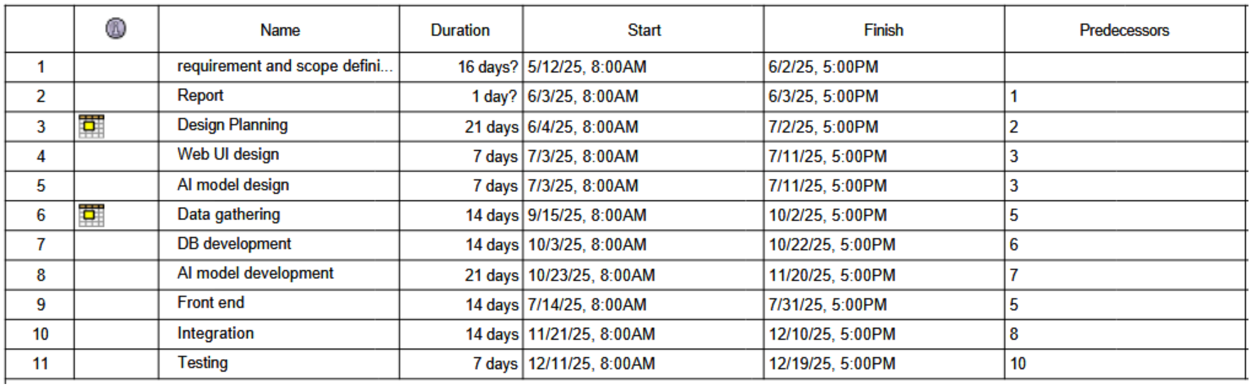
- A full-stack web-based dashboard using React and Grafana.
- Visualizes exploit results, system health, business risk impact, SLA-driven action items

Technical Documentation

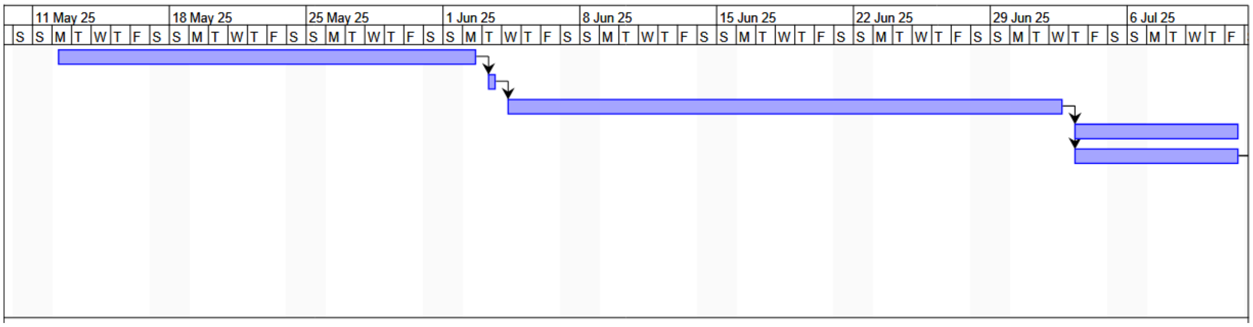
- System architecture diagrams
- Setup and deployment instructions
- API reference and module usage guide
- Limitations and ethical use guidelines

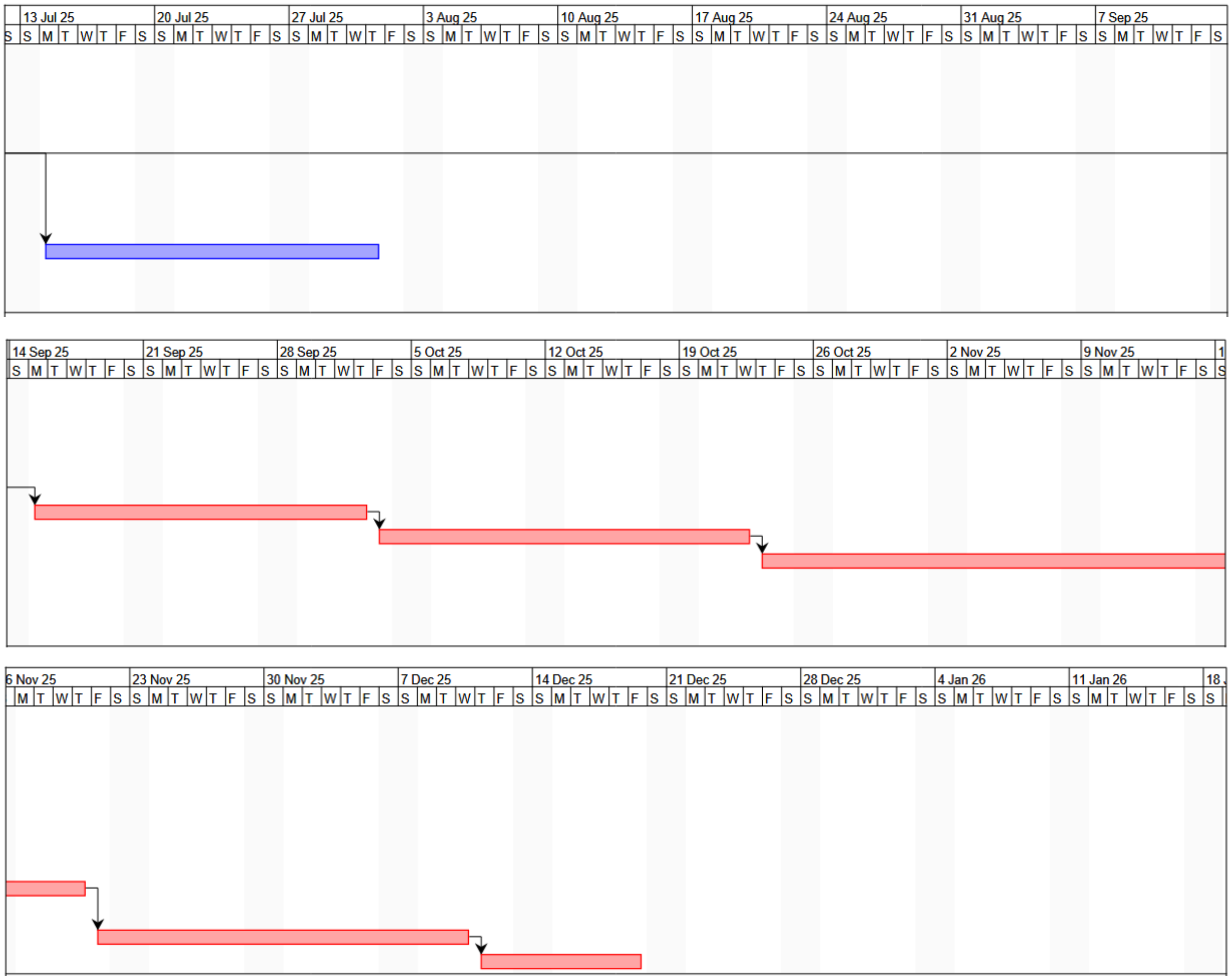
Initial Timeline

Gantt Charts



	ⓘ	Name	Duration	Start	Finish	Predecessors
1		requirement and scope defini...	16 days?	5/12/25, 8:00AM	6/2/25, 5:00PM	
2		Report	1 day?	6/3/25, 8:00AM	6/3/25, 5:00PM	1
3	📅	Design Planning	21 days	6/4/25, 8:00AM	7/2/25, 5:00PM	2
4		Web UI design	7 days	7/3/25, 8:00AM	7/11/25, 5:00PM	3
5		AI model design	7 days	7/3/25, 8:00AM	7/11/25, 5:00PM	3
6	📅	Data gathering	14 days	9/15/25, 8:00AM	10/2/25, 5:00PM	5
7		DB development	14 days	10/3/25, 8:00AM	10/22/25, 5:00PM	6
8		AI model development	21 days	10/23/25, 8:00AM	11/20/25, 5:00PM	7
9		Front end	14 days	7/14/25, 8:00AM	7/31/25, 5:00PM	5
10		Integration	14 days	11/21/25, 8:00AM	12/10/25, 5:00PM	8
11		Testing	7 days	12/11/25, 8:00AM	12/19/25, 5:00PM	10





Project Management Tools to be used

- **GitHub:** Source code, version control
- **Google Meet:** Communication
- **Google Drive:** Collaboration on reports
- **Canva:** Collaboration on presentations

References

Techtarget.com. (2025). *What is Pen Testing and Why is it Important?* [online] Available at: https://www.techtarget.com/whatis/video/An-explanation-of-pen-testing [Accessed 11 May 2025].

Hammar, K. (2024). *Limmen/awesome-rl-for-cybersecurity*. [online] GitHub. Available at: https://github.com/Limmen/awesome-rl-for-cybersecurity.

Xu, H., Wang, S., Li, N., Wang, K., Zhao, Y., Chen, K., Yu, T., Liu, Y. and Wang, H. (2024). *Large Language Models for Cyber Security: A Systematic Literature Review*. [online] arXiv.org. doi:https://doi.org/10.48550/arXiv.2405.04760.

Arxiv.org. (2017). *LLM-Assisted Proactive Threat Intelligence for Automated Reasoning*. [online] Available at: https://arxiv.org/html/2504.00428v1 [Accessed 11 May 2025].

Dinil Mon Divakaran and Sai Teja Peddinti (2024). *LLMs for Cyber Security: New Opportunities*. [online] arxiv.org. Available at: https://arxiv.org/pdf/2404.11338 [Accessed 12 May 2025].

GitLab. (n.d.). *Exploit-DB / Exploits + Shellcode + GHDB · GitLab*. [online] Available at: https://gitlab.com/exploit-database/exploitdb.

Ridge Security. (2025). *Automated Penetration Testing Tool | RidgeBot | Ridge Security*. [online] Available at: https://ridgesecurity.ai/ridgebot/ [Accessed 11 May 2025].

Garn, D. (2025). *Top Kali Linux tools and how to use them | TechTarget*. [online] Security. Available at: https://www.techtarget.com/searchsecurity/tip/Top-Kali-Linux-tools-and-how-to-use-them.