CyberHawk: AI-Powered Intrusion Detection and Prevention System with Ransomware & Malware Analysis

M AHMED

HASSAN JAVED

DEPARTMENT OF COMPUTER SCIENCES

COMSATS UNIVERSITY ISLAMABAD, WAH CAMPUS

WAH CANTT – PAKISTAN

SESSION 2022-2026

CyberHawk: AI-Powered Intrusion Detection and Prevention System with Ransomware & Malware Analysis

Undertaken By:

M AHMED

REG. NO. CIIT/SP22-BSE-055/WAH

HASSAN JAVED

REG. NO. CIIT/SP22-BSE-057/WAH

Supervised By:

DR KASHIF AYYUB

A DISSERTATION SUBMITTED AS A PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELORS IN COMPUTER SCIENCE / SOFTWARE ENGINEERING

DEPARTMENT OF COMPUTER SCIENCES

COMSATS UNIVESITY ISLAMABAD, WAH CAMPUS

WAH CANTT – PAKISTAN

SESSION 2022-2026

DEDICATION

We begin with the name of ALLAH who is gracious and most merciful. We are indeed blessed by Almighty, who strengthens us with wisdom and acumen to fulfil our project. With the clemency blessings of lord of Wisdom, we were able to do the project with confidence and utmost satisfaction.

This project is also wholeheartedly dedicated to our beloved parents, who have been our source of inspiration and gave us strength when we thought of giving up, who continually provide their moral, spiritual, emotional, and financial support. Without their support, we would not be able to complete our project.

We especially thank to our teachers who are always motivated and helped us out in our difficult times. To our brothers, sisters, mentors, and friends who share their advice and encouragement to finish this project. And lastly, once again we dedicated this project to the Almighty ALLAH who has blessings upon us and with His help, we can complete our project within stipulated time, thank You for Your guidance, strength, power of mind, protection, and skills and for giving us a healthy life. All of these, we offer to You.

M Ahmed

Hassan Javed

ACKNOWLEDGEMENT

All praises are due to Almighty ALLAH, the Most Merciful and Compassionate. Without His blessings and guidance, we would not have been able to complete this project.

This project would not have been possible without the support, encouragement, and guidance of several individuals. We express our sincere gratitude to our respected project supervisor, Dr. Kashif Ayyub, for his continuous guidance. Despite his busy schedule, he always took time for us whenever we sought help. His valuable insights and direction were crucial at every stage of this project.

We are also deeply thankful to all our teachers, whose motivation and dedication inspired us to work hard and stay focused throughout our journey.

Our heartfelt salutations go to our loving parents, whose invaluable prayers, sincere advice, and unwavering support strengthened our resolve to strive for knowledge and integrity. Their encouragement enabled us to reach this milestone.

Lastly, we would like to express our gratitude to our friends who stood by us and assisted us whenever we needed help during the completion of this project.

PROJECT BRIEF

PROJECT NAME	CYBERHAWK
ORGANIZATION NAME	COMSATES UNIVERSITY ISLAMABAD, WAH CAMPUS
OBJECTIVE	TO DEVELOP AN ADVANCED, REAL-TIME INTRUSION DETECTION AND PREVENTION SYSTEM (IDPS) WITH INTEGRATED MALWARE ANALYSIS AND RANSOMWARE MITIGATION, PROVIDING AUTOMATED, INTELLIGENT, AND BEHAVIOR-BASED THREAT DETECTION.
UNDERTAKEN BY	M AHMED HASSAN JAVED
SUPERVISED BY	DR KASHIF AYYUB ASSISTANT PROFESSOR DEPARTMENT OF COMPUTER SCIENCE COMSATS UNIVERSITY ISLAMABAD-WAH CAMPUS
STARTED ON	MARCH 2025
COMPLETED ON	DECEMBER 2025
COMPUTER USED	DELL LAPTOP.
SOURCE LANGUAGE	PYTHON (SCAPY, MACHINE LEARNING MODELS) PHP, JAVASCRIPT, AJAX HTML, CSS, BOOTSTRAP JSON-BASED LOGGING

OPERATING SYSTEM	WINDOWS
TOOLS USED	VS CODE, XAMPP

Table of Contents

1. INTRODUCTION 1
   1.1 Background of the System: 1
   1.2 Objectives of the System: 2
   1.3 Significance of the System: 2
2. REQUIREMENT SPECIFICATIONS 3
   2.1 Product Scope: 3
   2.2 Product Description: 3
      2.2.1 Product Perspective 3
      2.2.2 Product Functionality 4
      2.2.3 Users and Characteristics 5
      2.2.4 Operating Environment 5
   2.3 Specific Requirements 5
      2.3.1 Functional Requirements 5
      2.3.2 Behavioral Requirements 6
      2.3.3 Use Case 7
      2.3.4 External Interface Requirements 8
   2.4 Non-functional Requirements 15
      2.4.1 Performance Requirements 15
      2.4.2 Real-Time Monitoring Performance 15
      2.4.3 Malware and Ransomware Modules 16
      2.4.4 Performance Management 17
      2.4.5 Safety and Security Requirements 17
      2.4.6 Software Quality Attributes 18
3. DESIGN SPECIFICATIONS 20
   3.1 Introduction 20
      3.1.1 Deployment 20
   3.2 Logical Viewpoint 21
      3.2.1 Class Diagram 21
   3.3 Information Viewpoint 22
   3.4 Interaction Viewpoint 23
   3.5 State Dynamics Viewpoint 26
   3.6 Algorithmic Viewpoint 27

3.6.1 Algorithmic Viewpoint of Network Intrusion Detection System (IDS) 27
3.6.2 Algorithmic Viewpoint of Malware Analysis System 28
3.6.3 Algorithmic Viewpoint of Ransomware Detection System 29
3.6.4 Algorithmic Viewpoint of User Authentication System 31
3.6.5 Algorithmic Viewpoint of Reporting System 32
4. DEVELOPMENT AND TOOLS 33
4.1 Introduction 33
4.2 Development Plan 34
4.3 Development Tools 36
4.3.1 Configuration management tools 36
4.4 Conclusion and Future Work/Extensions 37
5. QUALITY ASSURANCE 39
5.1 Introduction 39
5.2 Traceability Matrix 39
5.2.1 Requirement Traceability Matrix 40
5.3 Test Plan 40
6. USER MANUAL 48
6.1 Introduction 48
6.2 Hardware/Software Requirements for the System 48
6.3 Installation Guide for Application 49

List Of Tables

Table 1 3.6.1 Algorithmic Viewpoint of Network Intrusion Detection System..... 27
Table 2 3.6.2 Algorithmic Viewpoint of Malware Analysis System..... 28
Table 3 3.6.3 Algorithmic Viewpoint of Ransomware Detection System 29
Table 4 3.6.4 Algorithmic Viewpoint of User Authentication System 31
Table 5 3.6.5 Algorithmic Viewpoint of Reporting System..... 32
Table 6 Development Plan Table..... 34
Table 7 timeline Table 35
Table 8 Test Case 1 41
Table 9 Test Case 2 42
Table 10 Test case 3..... 43
Table 11 Test case 4..... 44
Table 12 Test Case 5 45
Table 13 Test Case 6 46
Table 14 Test Case 7 47

List of Figures

Figure 2-1 Project perspective 4
Figure 2-2 Use case Diagram 7
Figure 2-3 login Page 8
Figure 2-4 Dashboard 10
Figure 2-5 Ransomware Page 10
Figure 2-6 Malware Analysis Page 11
Figure 2-7 Reporting page 12
Figure 2-8 Settings Page 13
Figure 2-9 Profile Page 14
Figure 3-1 Deployment Diagram 20
Figure 3-2 Class Diagram 21
Figure 3-3 ERD Diagram 22
Figure 3-4 User login sequence diagram 23
Figure 3-5 Alert Generation Sequence Diagram 24
Figure 3-6 Malware Module Sequence Diagram 24
Figure 3-7 system sequence 25
Figure 3-8 State Machine diagram 26

1. INTRODUCTION

In the modern digital landscape, cybersecurity is a critical concern for individuals, organizations, and governments alike. CyberHawk emerges as a comprehensive solution to safeguard digital assets by integrating Network Intrusion Detection, Malware Analysis, and Ransomware Protection into a unified web-based platform. This chapter introduces the context, purpose, and scope of the CyberHawk project, highlighting the importance of real-time threat detection and proactive system defense. By combining advanced machine learning techniques, behavioral analysis, and multi-source threat intelligence, CyberHawk aims to protect users from the evolving landscape of cyber threats.

The main purpose of this system is to provide organizations with a secure, real-time monitoring and threat mitigation platform. It allows users to detect and respond to intrusions, analyze suspicious files, and prevent ransomware attacks, all through an intuitive web interface.

1.1 Background of the System:

CyberHawk is developed from scratch to address the lack of an integrated cybersecurity monitoring solution that combines intrusion detection, malware scanning, and ransomware mitigation in a single platform. Many organizations rely on isolated tools that do not provide real-time visibility or automated response capabilities.

1.2 Objectives of the System:

The main objectives are as follows:

- Provide real-time network intrusion detection using machine learning and traffic analysis.
- Detect and analyze malware using multi-source intelligence platforms such as VirusTotal and MalwareBazaar.
- Implement proactive ransomware detection and file monitoring to prevent data loss.
- Offer a web-based dashboard for visualization, alert management, and system control.
- Enable secure user authentication and session management for multiple user roles.

1.3 Significance of the System:

CyberHawk enhances cybersecurity posture by providing continuous monitoring, automated detection, and alerting. Organizations benefit from timely threat intelligence, reduced manual intervention, and improved operational security. By integrating all key modules into a single platform, it simplifies security management and increases overall efficiency.

2. REQUIREMENT SPECIFICATIONS

The requirement specifications outline the system's features and functionalities, ensuring CyberHawk meets organizational cybersecurity needs effectively.

2.1 Product Scope:

The project scope involves developing CyberHawk, a comprehensive network security system. The platform integrates intrusion detection, malware analysis, and ransomware prevention. It provides a web-based interface for real-time monitoring, alert visualization, threat reporting, and administrative controls. The system is designed to support multiple users, secure sessions, and automated response to detected threats.

2.2 Product Description:

This section presents a detailed overview of CyberHawk's specifications and functionalities. The system is designed to be robust, user-friendly, and capable of handling high volumes of network traffic while providing actionable insights. Its modular design allows for future expansions such as advanced ML models and SIEM integration.

2.2.1 Product Perspective

CyberHawk offers a centralized cybersecurity solution combining intrusion detection, malware scanning, and ransomware mitigation. It provides organizations with a platform to monitor their networks, analyze suspicious activities, and respond to threats efficiently. The system is designed for real-time threat detection while being scalable for enterprise environments.

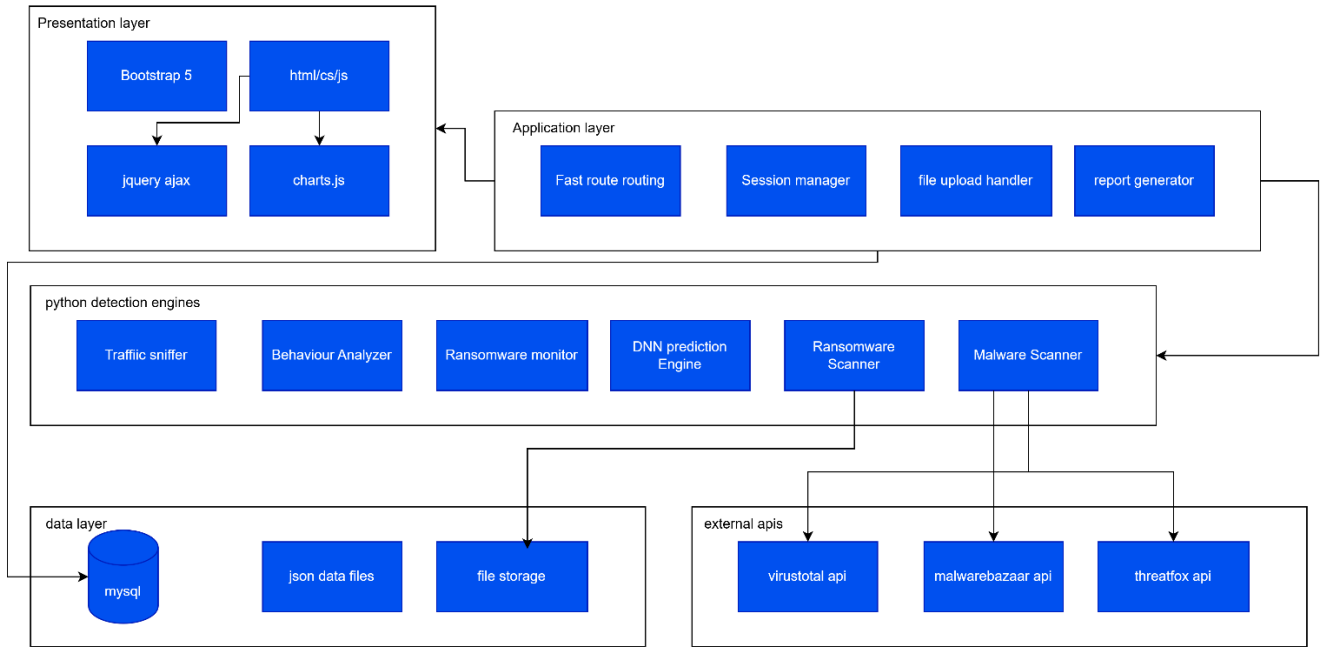


Figure 2-1 Project perspective

2.2.2 Product Functionality

CyberHawk provides essential security features to protect organizational networks and systems.

- **Network Intrusion Detection**: Monitors network traffic in real-time, detects anomalies, and generates alerts for attacks such as DoS/DDoS, port scans, and brute force attempts.
- **Malware Analysis**: Scans files using VirusTotal, MalwareBazaar, and ThreatFox, generating reports with threat levels, malware family, and recommended actions.
- **Ransomware Protection**: Monitors file operations, detects suspicious encryption patterns or ransom notes, and quarantines high-risk files to prevent data loss.

- **Web Dashboard**: Offers an interactive interface with live metrics, charts, and controls to manage monitoring, view alerts, and generate reports.
- **User Management**: Provides secure, role-based access, session management, and customized permissions for administrators, IT staff, and general users.

2.2.3 Users and Characteristics

- **Administrators**: Monitor system status, configure alerts, and manage user access.
- **IT Security Staff**: Analyze alerts, review malware reports, and respond to incidents.
- **General Users**: Upload files for malware scanning and receive notifications on threat status.

2.2.4 Operating Environment

The system operates in the following environments:

- Windows 10 and above
- Linux distributions (Ubuntu, CentOS)
- Web browsers: Chrome, Firefox, Edge

2.3 Specific Requirements

In this section we will be discussing the functional requirements of the system.

2.3.1 Functional Requirements

- **Real-time Network Traffic Capture and Analysis**

The system captures network traffic in real-time and analyzes it for unusual patterns. It allows administrators to monitor flows and detect potential threats quickly.

- **Detection of DoS/DDoS, Port Scanning, and Brute Force Attacks**

  CyberHawk identifies attacks such as DoS/DDoS, port scanning, and brute force attempts. Alerts are generated with relevant details for immediate response.

- **Malware Scanning with Multi-Engine Threat Intelligence**

  Users can upload files to scan with VirusTotal, MalwareBazaar, and ThreatFox. The system reports threat levels, malware family, and recommended actions.

- **Ransomware Monitoring with Auto-Quarantine**

  The platform monitors files for suspicious encryption or ransom notes. High-risk files are automatically quarantined to protect data.

- **Alert Management and Reporting through the Web Dashboard**

  All alerts and events are displayed on a dashboard with live metrics and visualizations. Reports can be generated for analysis and record-keeping.

2.3.2 Behavioral Requirements

The system provides secure login for all users, ensuring that only authorized personnel can access sensitive data. Its intuitive navigation allows administrators and staff to manage alerts and monitor network activity efficiently. Real-time updates display ongoing threats and system status, enabling quick response to incidents. The platform also ensures data integrity and reduces false positives for accurate threat detection.

2.3.3 Use Case

CyberHawk provides use cases such as intrusion detection, malware analysis, ransomware mitigation, and alert response, which guide the workflow of administrators and IT staff in handling real-world cybersecurity scenarios.

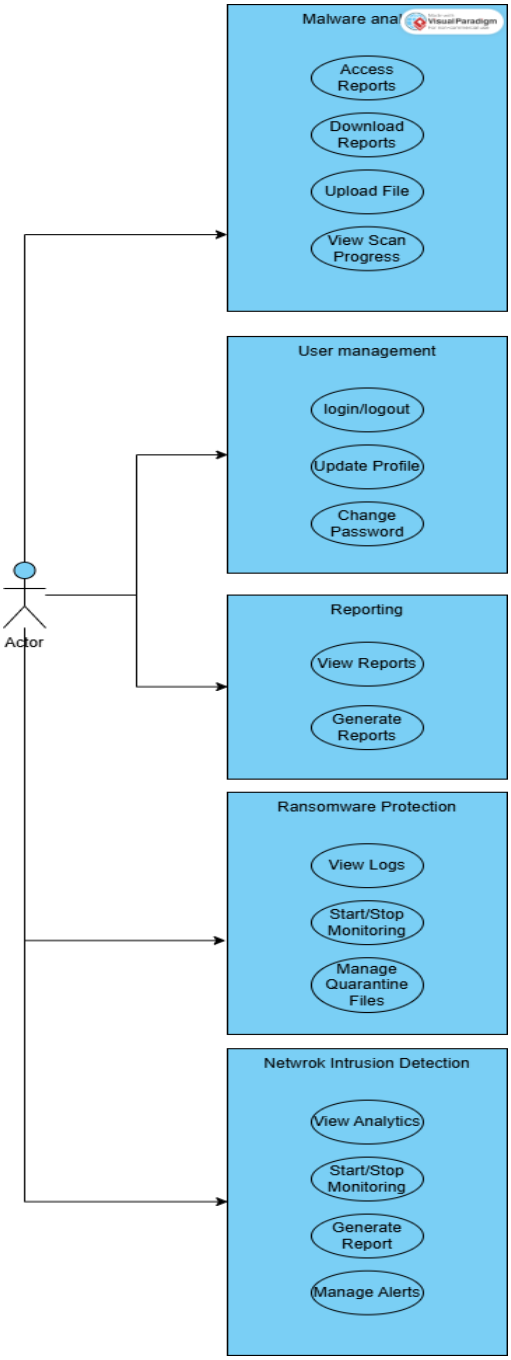


Figure 2-2 Use case Diagram

2.3.4 External Interface Requirements

Navigating CyberHawk's web platform is designed to be clear, organized, and user-friendly, providing administrators, IT staff, and general users with intuitive access to all features. The dashboard serves as a central hub, displaying real-time alerts, traffic statistics, malware reports, and ransomware activity. The interface is responsive and works across desktops and modern browsers, ensuring smooth usability for monitoring and management tasks. Functional clarity and visual simplicity allow users to quickly understand threat levels, access controls, and system settings without confusion.

- **Login page**

The following Figure Shows the interface of login page:

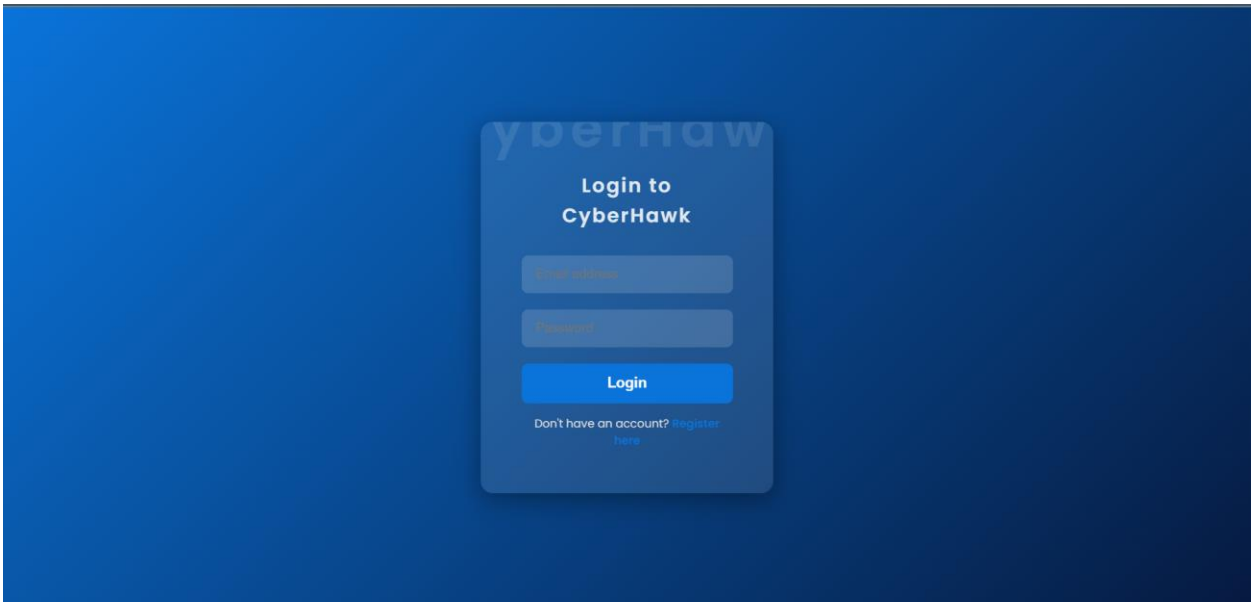


Figure 2-3 login Page

- **IDS Dashboard**

The following Figure shows the interface of dashboard of cyberhawk.

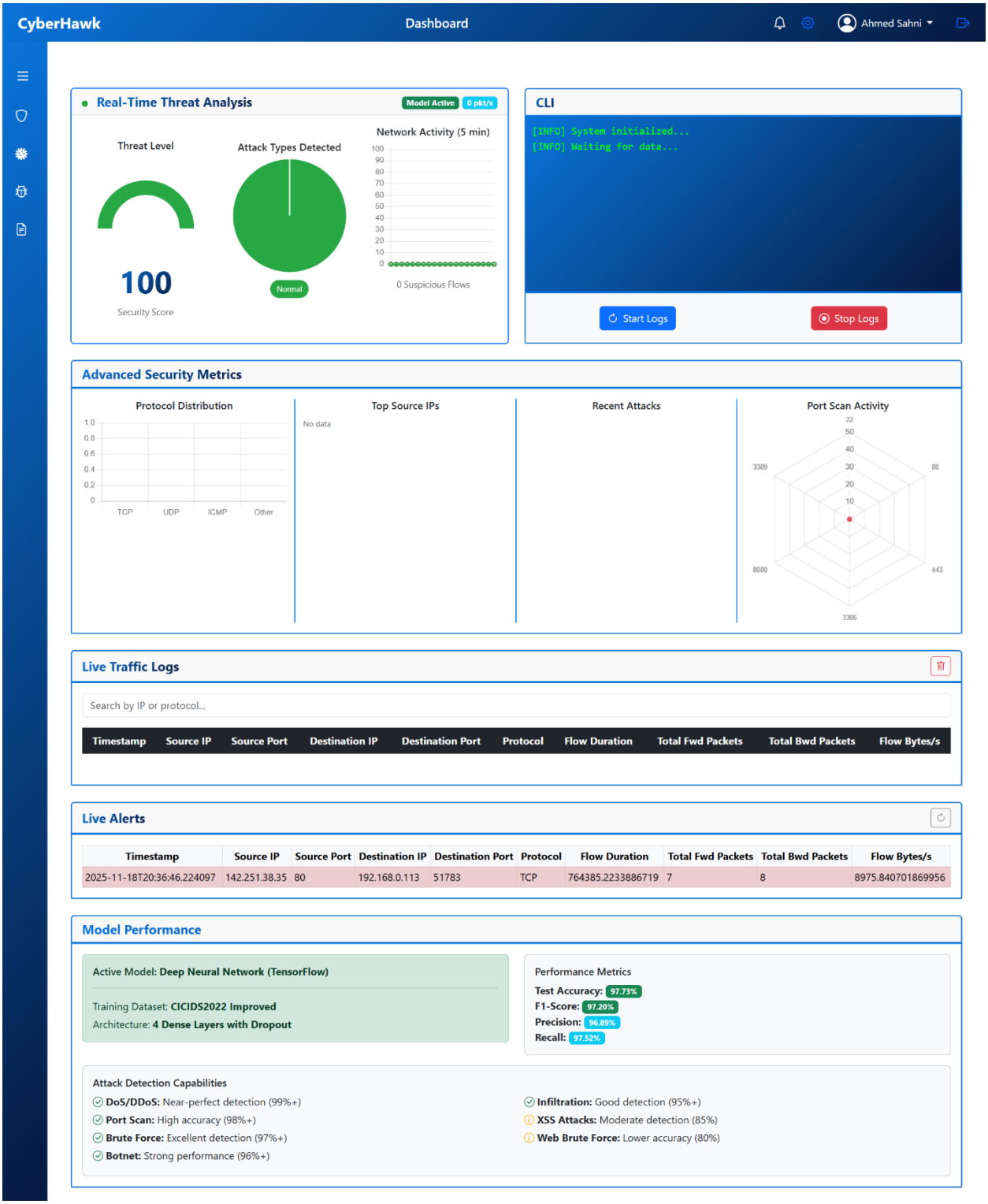


Figure 2-4 Dashboard

- **Ransomware Scan Interface**

The following Figure shows the Ransomware scan interface of cyberhawk.

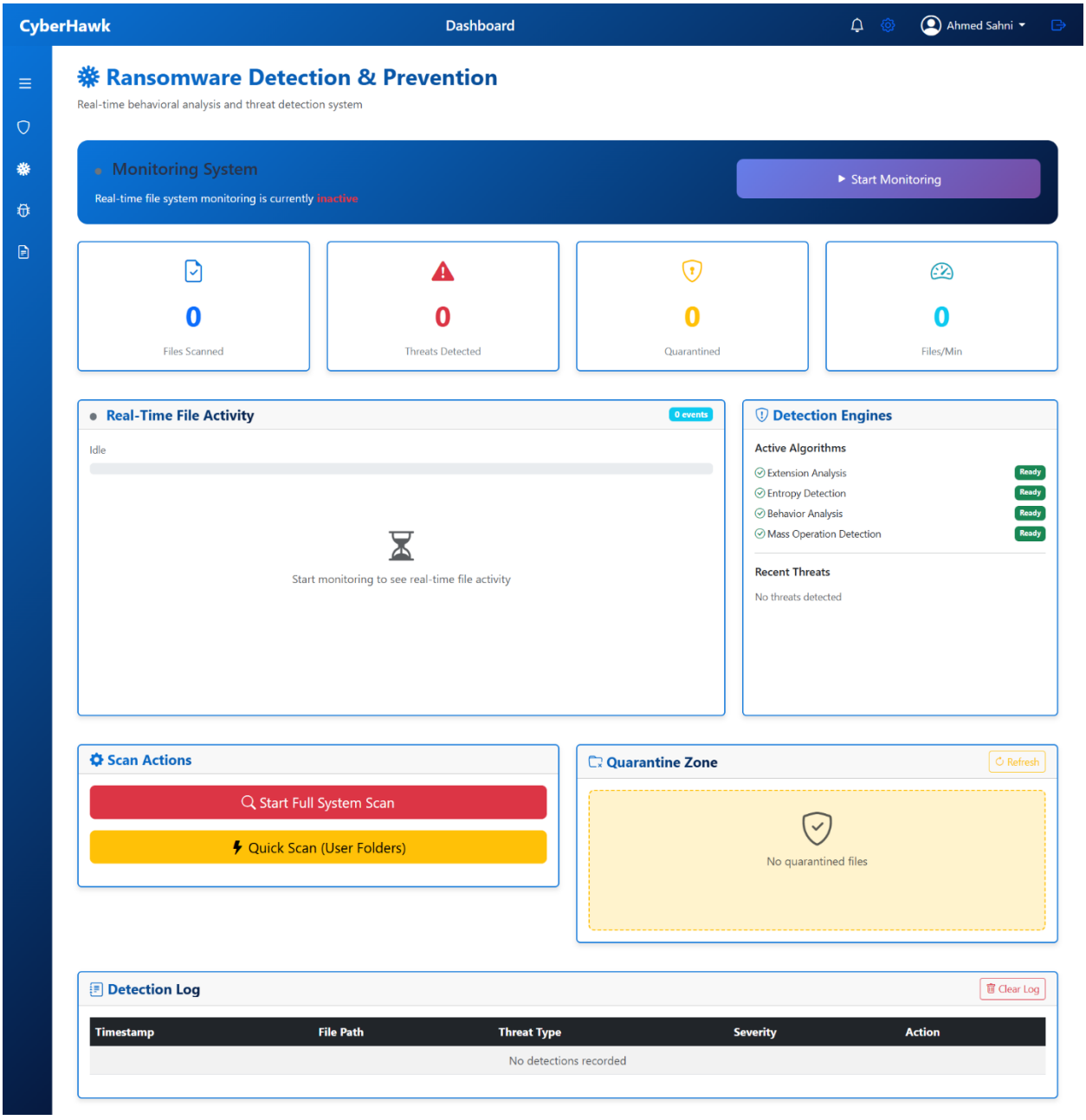


Figure 2-5 Ransomware Page

- **Malware scan Interface**

The following Figure shows the malware scan interface of dashboard of cyberhawk.

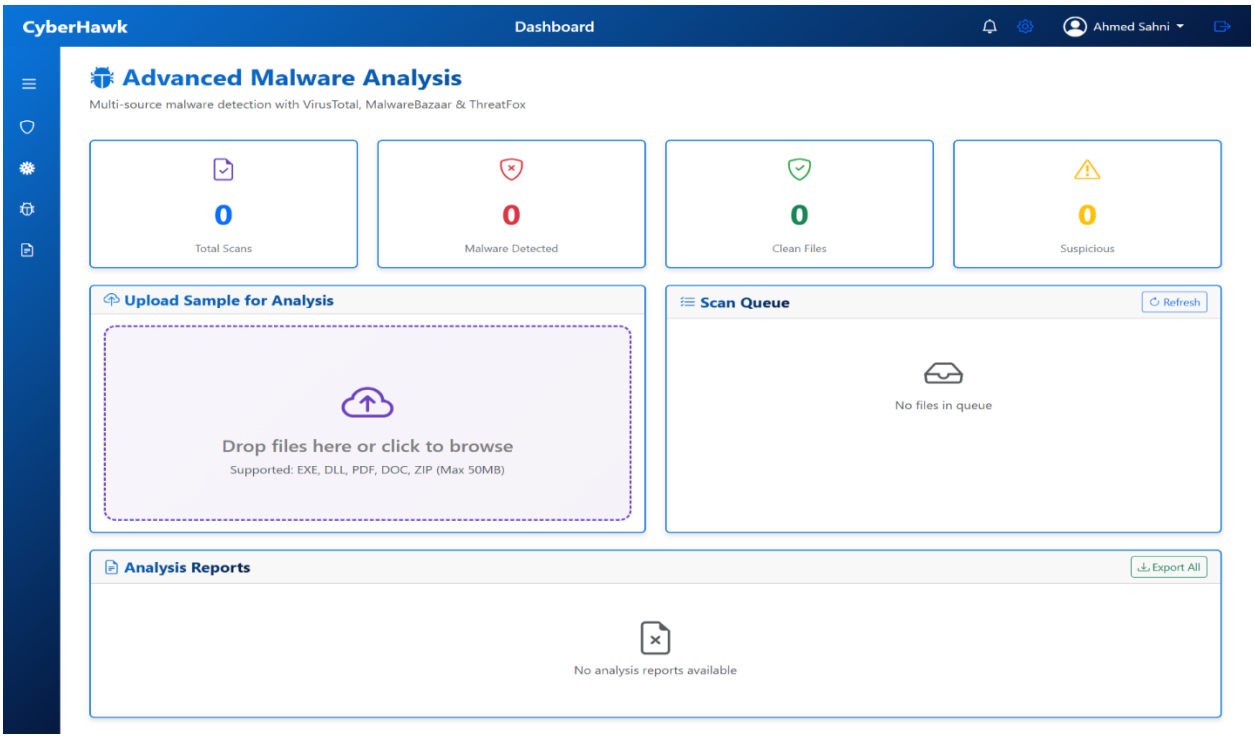


Figure 2-6 Malware Analysis Page

- **Reporting Interface**

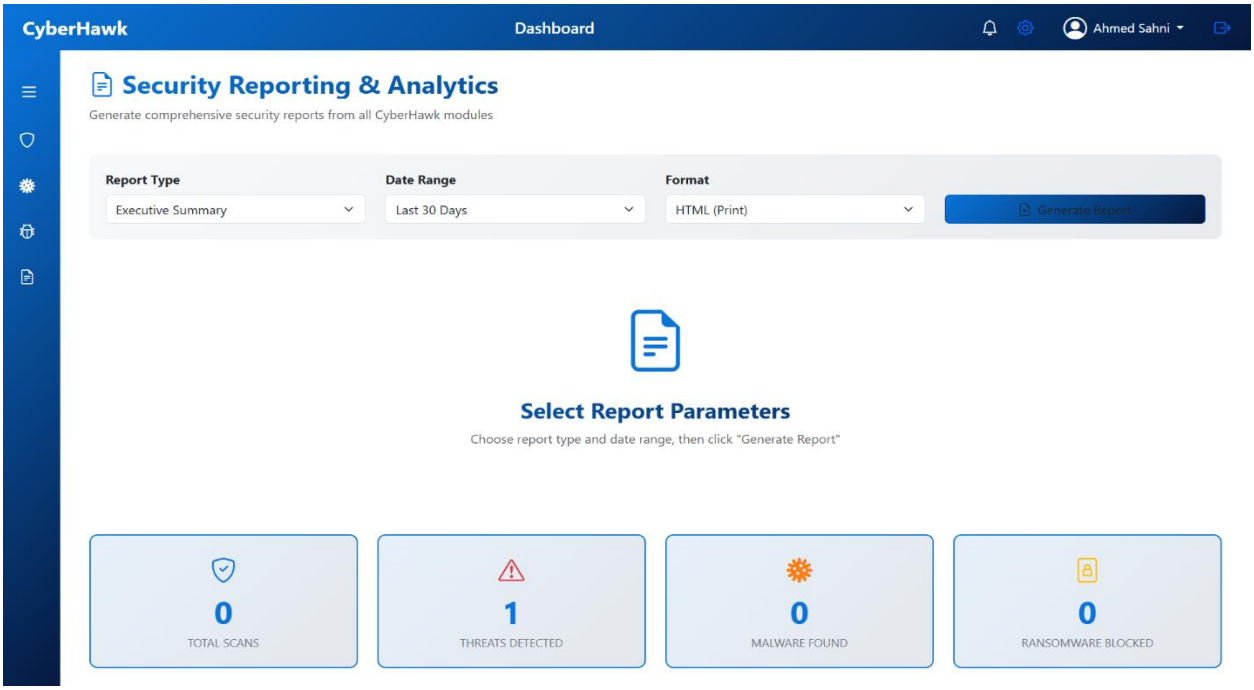


Figure 2-7 Reporting page

- **Settings of cyberhawk**

The following Figure shows the settings of cyberhawk.

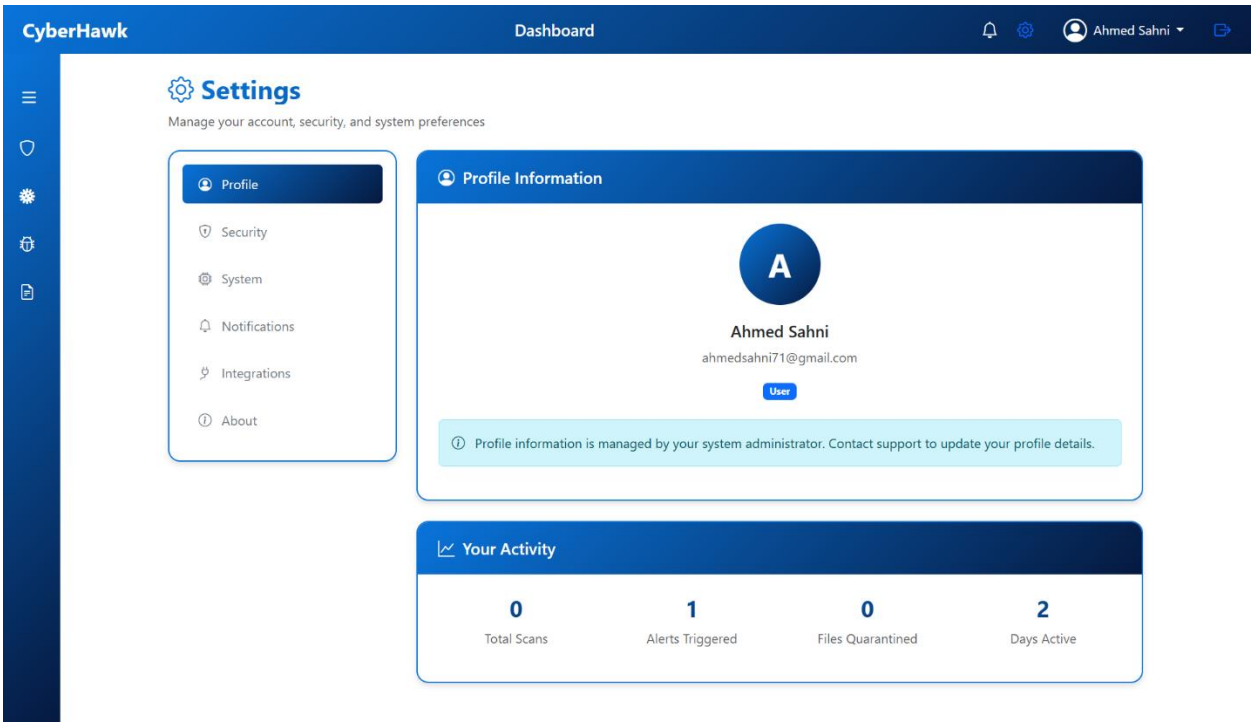


Figure 2-8 Settings Page

- **Profile Interface**

The following Figure shows the user profile of cyberhawk.

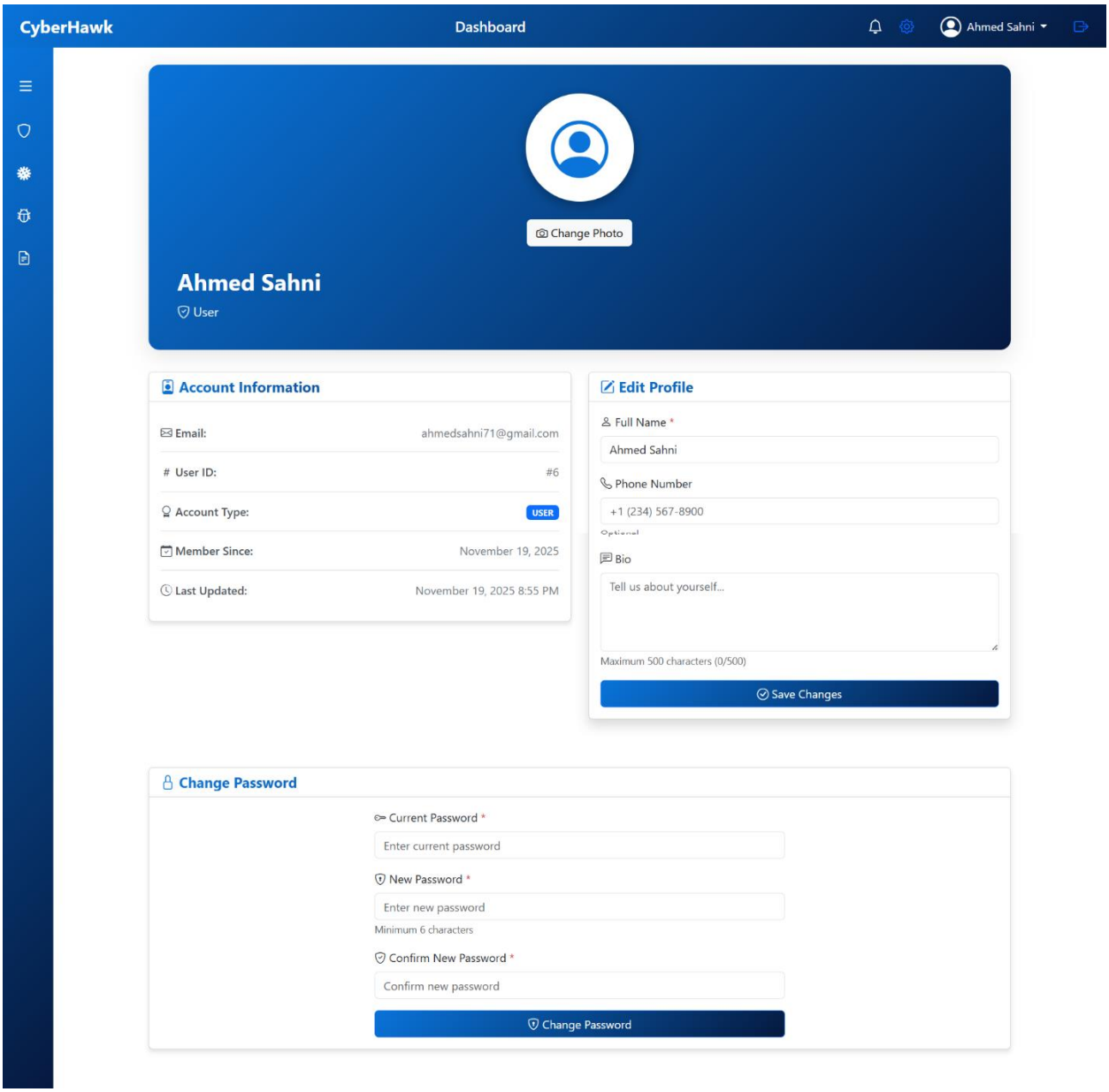


Figure 2-9 Profile Page

2.4 Non-functional Requirements

The non-functional requirements define the system's operational and quality standards, ensuring that CyberHawk performs reliably, securely, and efficiently while providing a user-friendly experience.

2.4.1 Performance Requirements

System Performance

- Dashboard must load within 2 seconds on standard networks.
- AJAX updates must be completed within 500 ms.
- Chart rendering must not exceed 1 second even with 1000+ datapoints.

Backend & Processing Performance

- Python IDS engine must process 1000 flows/second minimum.
- ML batch predictions (32 flows) must complete within 100 ms.
- File scanning throughput: ≥5 MB/sec (static analysis).
- Ransomware filesystem events must be detected within 100 ms.

Dashboard Updates

- Alerts, charts, and reports must refresh automatically without noticeable delay, maintaining smooth visualization of ongoing threats.

2.4.2 Real-Time Monitoring Performance

Network Monitoring

- Packet capture must work without packet drops at 100 Mbps.
- Flow aggregation must be completed within 100 ms after packet capture.

- Features (49 attributes) must compute in <50 ms per flow.

Alerting

- Maximum allowable delay from detection → alert display: 10 seconds.
- Critical alerts must show on dashboard within 3 seconds.

Dashboard Update Requirements

- Dashboard polls backend every 3 seconds.
- Real-time charts must be updated at 60 FPS without lag.
- Alerts table must update incrementally without reloading entire page.

2.4.3 Malware and Ransomware Modules

Malware Scanning Performance

- VirusTotal API response time: ≤ 5 seconds (API dependent).
- Static analysis of 10 MB file: ≤ 3 seconds.
- Behavioral analysis (strings/APIs) for 50 MB file: ≤ 10 seconds.
- Full report generation after scan: ≤ 2 seconds.

Ransomware Detection Performance

- File event detection: ≤ 100 ms.
- Entropy calculation: ≤ 50 ms.
- Suspicious extension/filename detection: ≤ 20 ms.
- Auto-quarantine execution: ≤ 1 second after critical detection.

Scanning Optimization

- Progress JSON updated every 5 files.
- RAM usage during scanning ≤ 1 GB.
- CPU usage capped to 50% of single core to avoid system lag.

2.4.4 Performance Management

The system must maintain logs and historical data for traffic, alerts, malware, and ransomware events. Administrators should be able to generate periodic reports and evaluate trends within the system without affecting ongoing real-time operations. Performance should be monitored to detect bottlenecks and ensure consistent system reliability.

2.4.5 Safety and Security Requirements

- CyberHawk must ensure secure login and role-based access for all users to prevent unauthorized access.
- Sensitive data, including network logs, malware reports, and quarantine files, must be encrypted and protected.
- The system should comply with standard security practices, including prevention of SQL injections, XSS attacks, and unauthorized process access.
- Automated backup and recovery mechanisms must be in place to safeguard critical data in case of failures.

2.4.6 Software Quality Attributes

- **Reliability**
  - The system should operate continuously with minimal downtime, targeting 99.9% availability.
  - Error handling mechanisms must ensure the system can recover gracefully from failures.
  - Logs and monitoring must detect and record failures for quick troubleshooting.
  - Backup and recovery procedures should maintain data integrity during unexpected crashes.
- **Usability**
  - CyberHawk should have an intuitive interface with clear navigation and visualizations.
  - Users must receive appropriate feedback for their actions to understand system responses.
  - The interface should be responsive across devices, supporting desktops and tablets.
  - Usability testing should be conducted to identify areas for improvement and ensure smooth user interaction.
- **Maintainability**
  - The system should use modular, well-structured code to facilitate updates and enhancements.

- Clear documentation, including code comments and architecture diagrams, must be provided.
- Version control should be implemented for efficient change management.
- Regular code reviews and refactoring should ensure maintainability and reduce technical debt.

- **Security**
  - Role-based access and authentication mechanisms must prevent unauthorized access.
  - Sensitive data such as network logs, malware reports, and quarantine files must be encrypted.
  - Secure coding practices should prevent common vulnerabilities like SQL injection or XSS.
  - Regular security assessments and penetration testing must be conducted to identify weaknesses.
- **Performance**
  - The system should process network traffic, malware scans, and ransomware monitoring efficiently.
  - Dashboard updates and report generation must be completed quickly, even under high loads.
  - Real-time threat detection should maintain minimal latency for alerts and notifications.
  - Performance monitoring should identify bottlenecks and allow optimization when needed.

3. DESIGN SPECIFICATIONS

3.1 Introduction

This section of the design document consists of fundamental design which represents logical and the structural view of the system, and the algorithm that highlights an abstract view of functional requirements of this system Composite Viewpoint.

3.1.1 Deployment

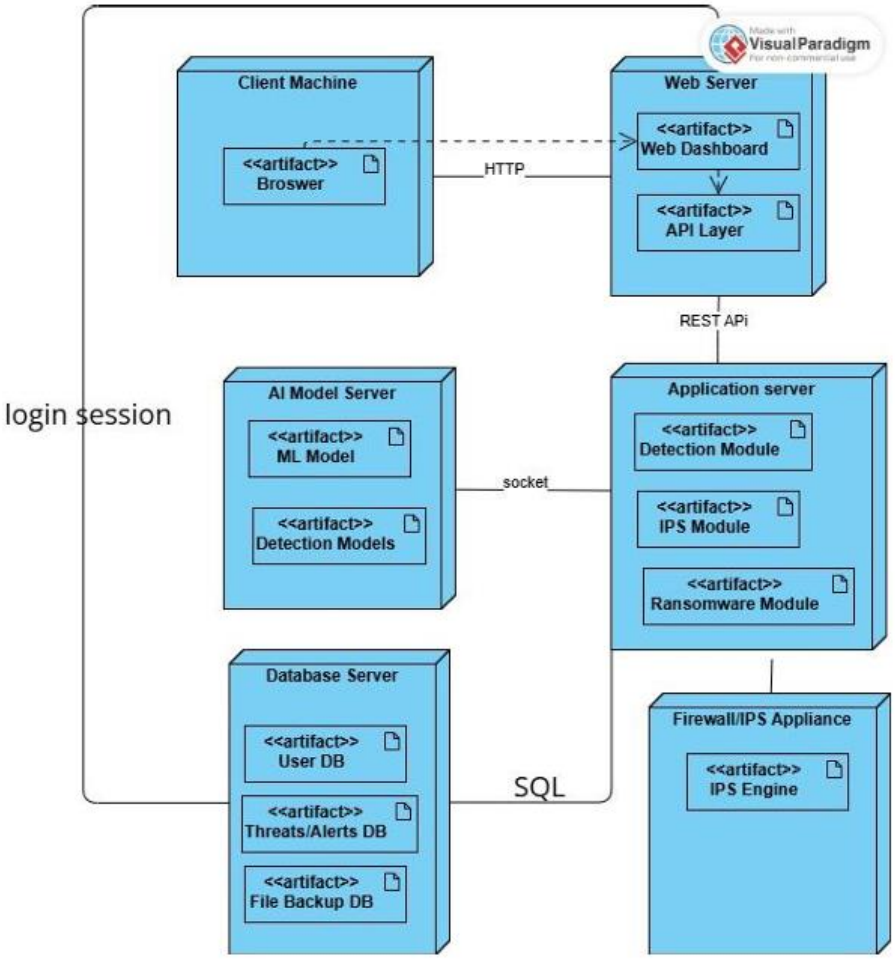


Figure 3-1 Deployment Diagram

3.2 Logical Viewpoint

In this section the logical viewpoint of the system is represented.

3.2.1 Class Diagram

Following is the class diagram of the cyberhawk.

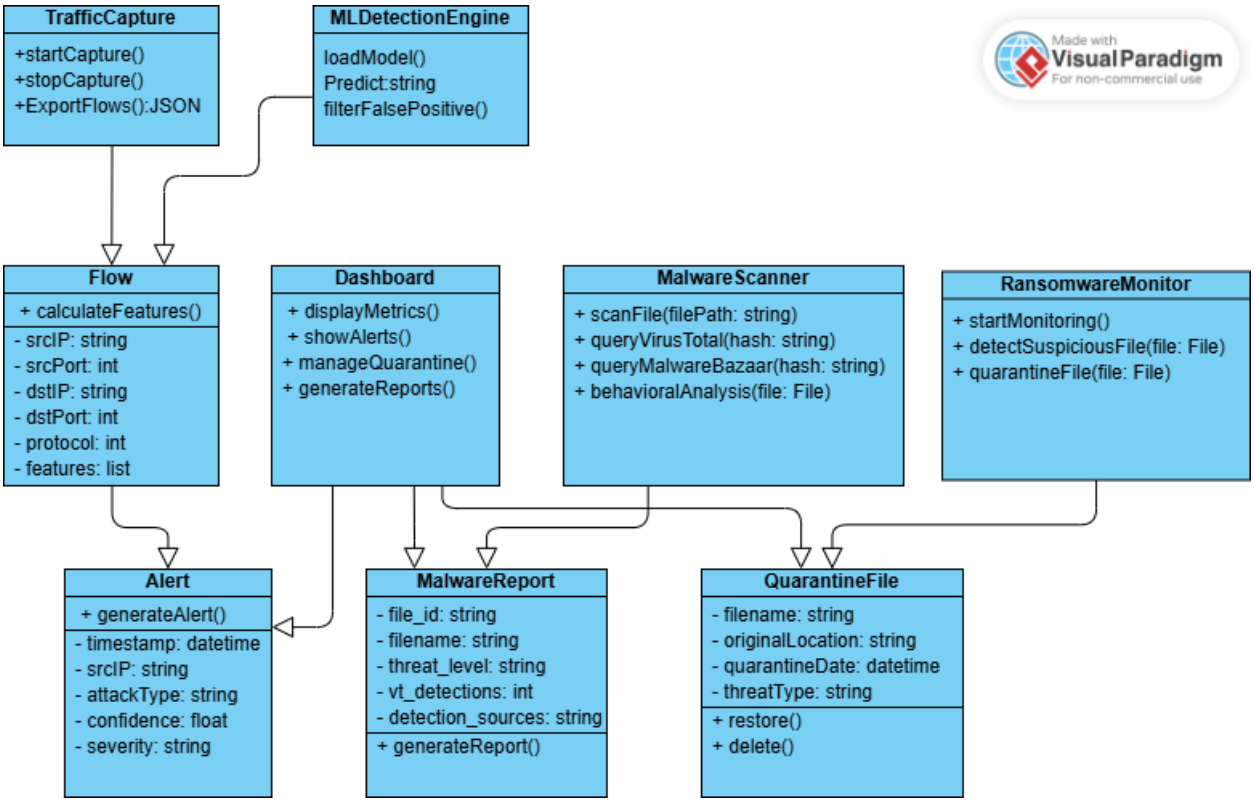


Figure 3-2 Class Diagram

3.3 Information Viewpoint

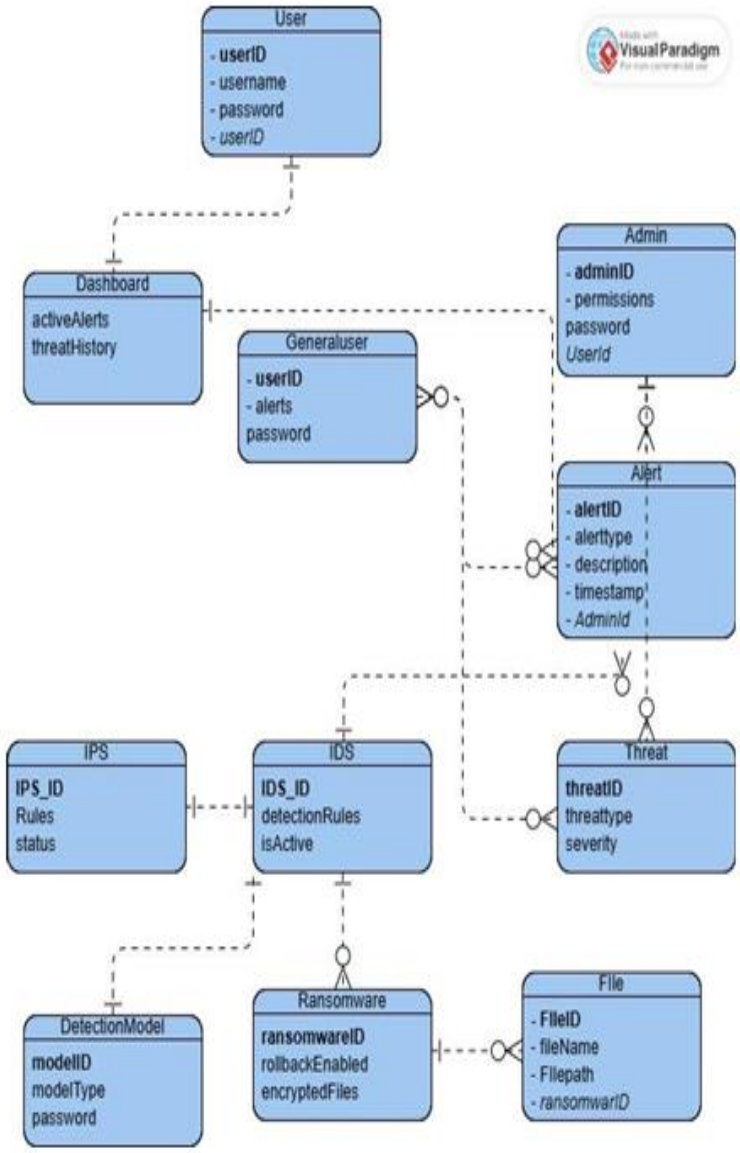


Figure 3-3 ERD Diagram

3.4 Interaction Viewpoint

This section represents the cyberhawk in term of sequence diagram.

User login Sequence:

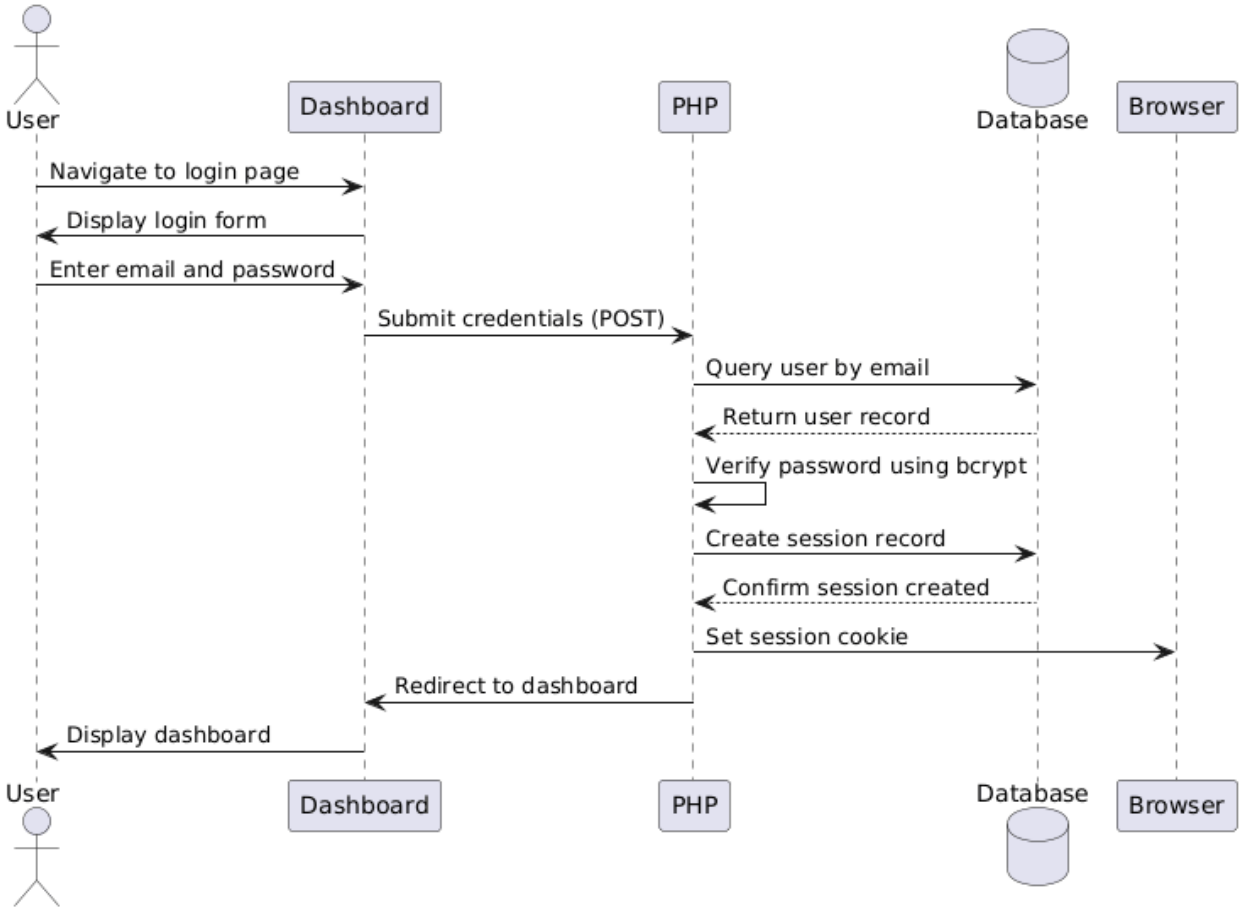


Figure 3-4 User login sequence diagram

Alert Generation Sequnece:

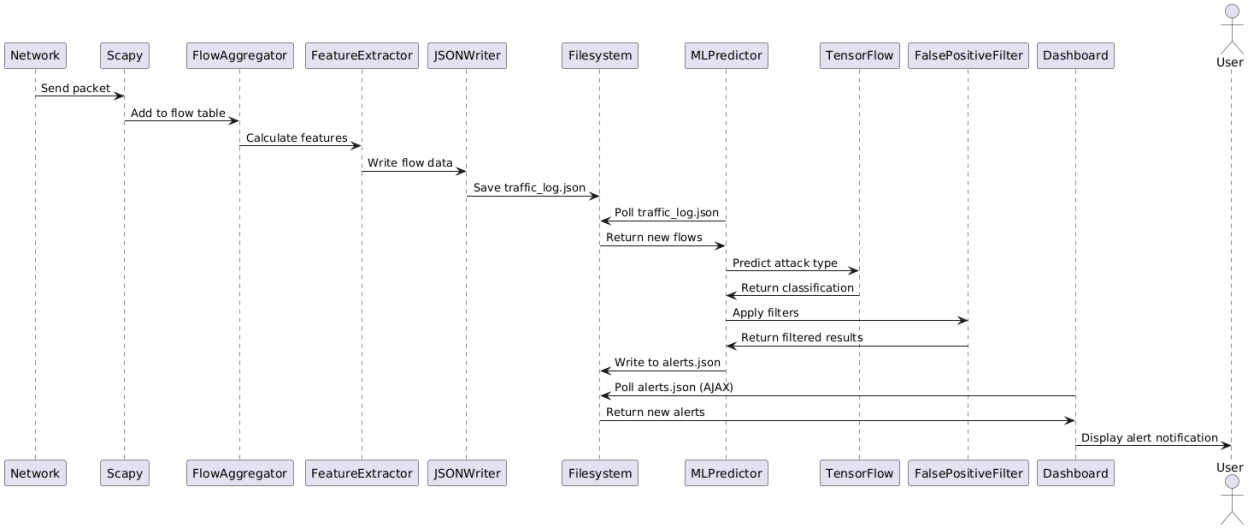


Figure 3-5 Alert Generation Sequence Diagram

Malware Sequence:

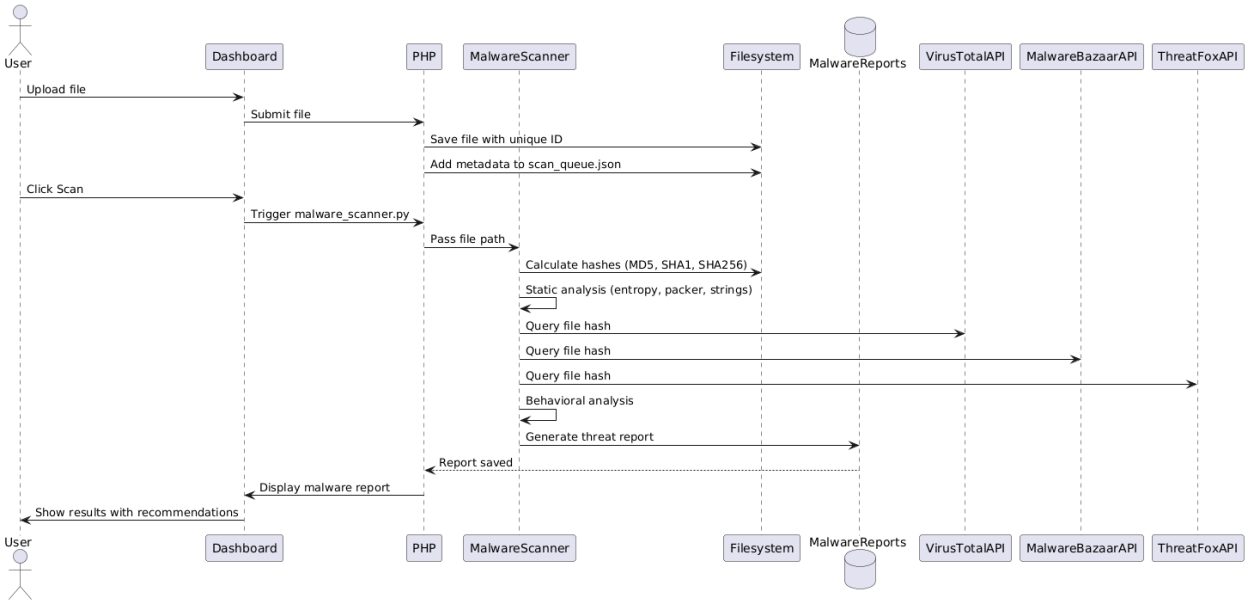


Figure 3-6 Malware Module Sequence Diagram

System sequence Diagram

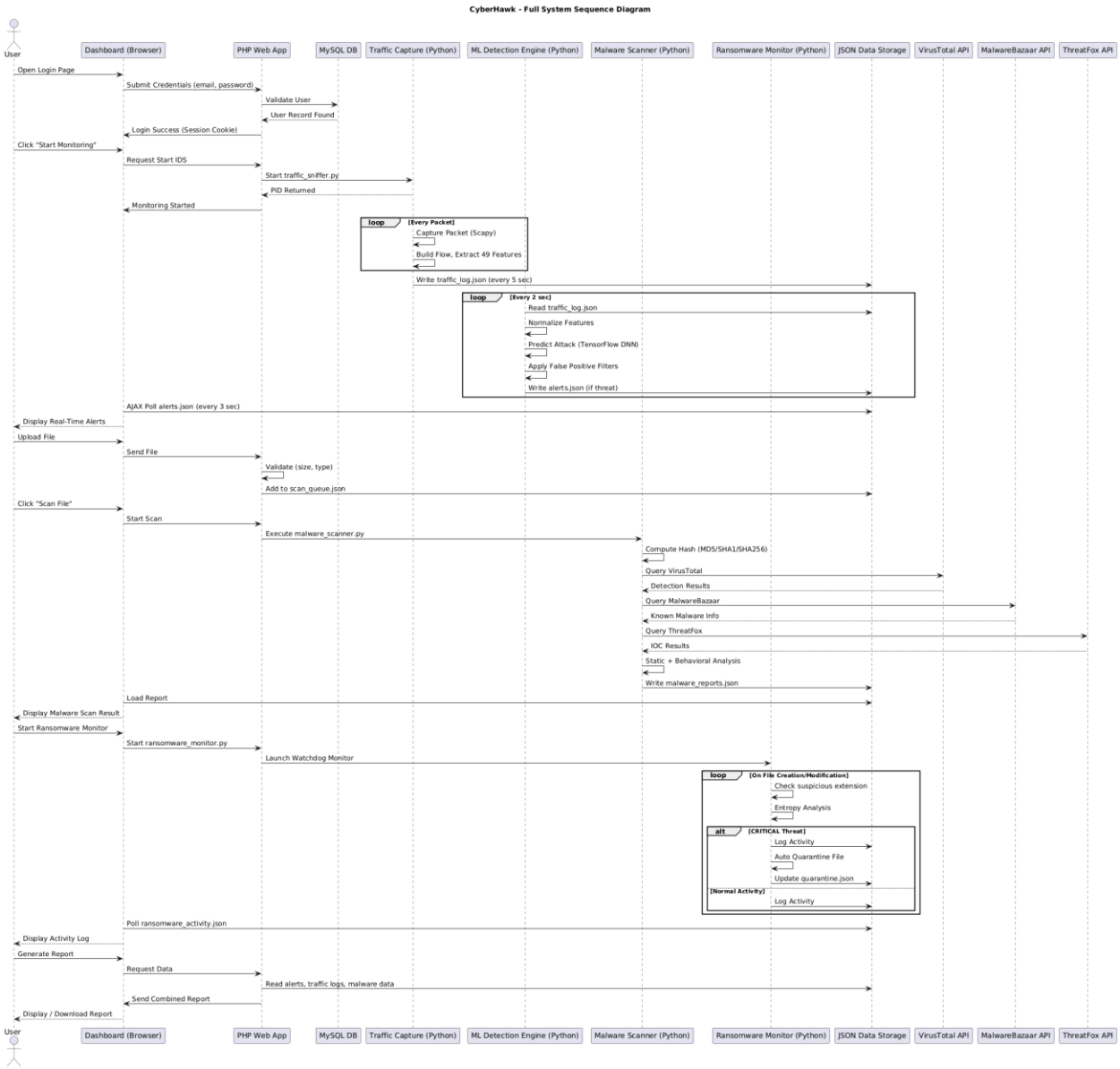


Figure 3-7 system sequence

3.5 State Dynamics Viewpoint

The following figure is about the state dynamics viewpoint.

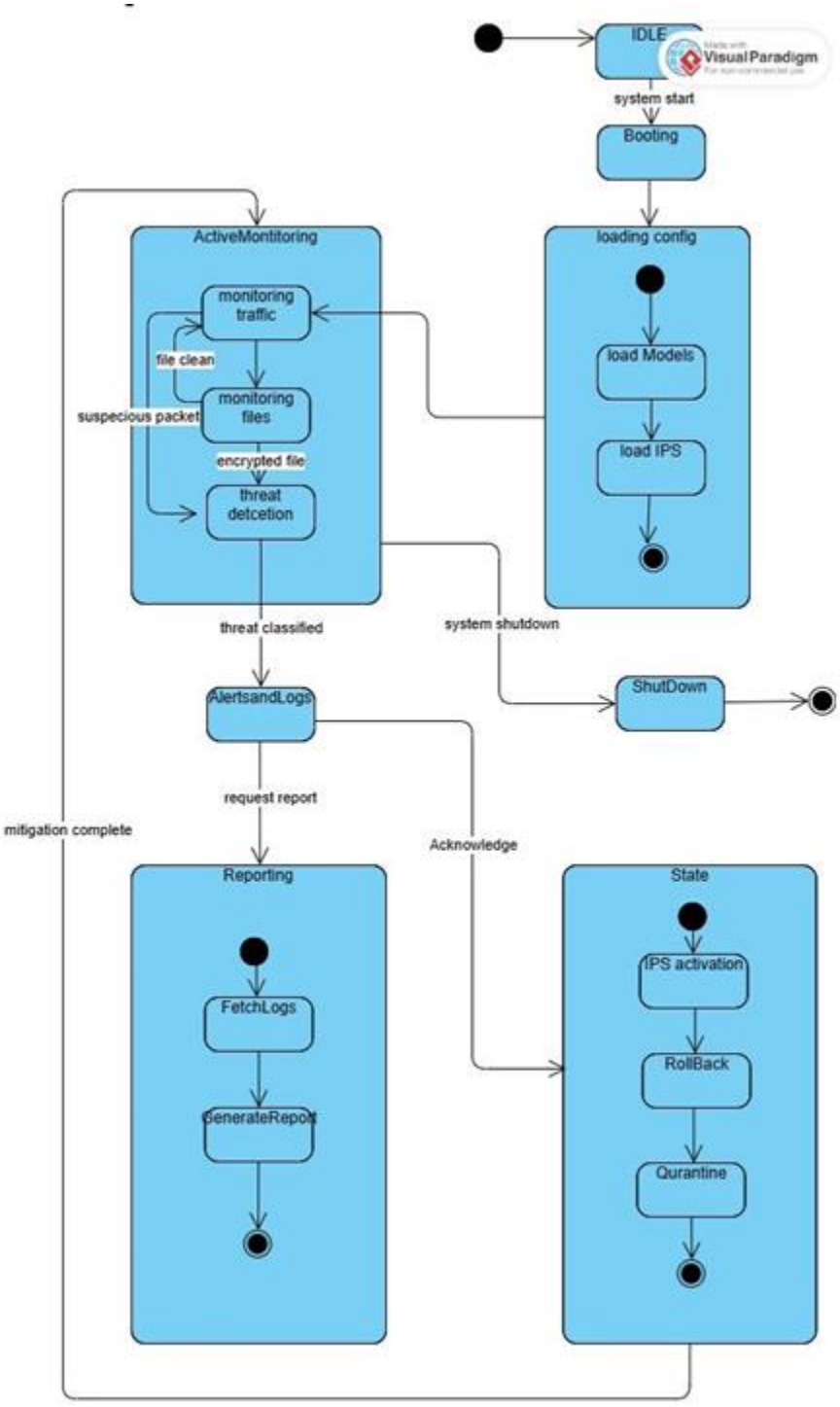


Figure 3-8 State Machine diagram

3.6 Algorithmic Viewpoint

3.6.1 Algorithmic Viewpoint of Network Intrusion Detection System (IDS)

Table 1 3.6.1 Algorithmic Viewpoint of Network Intrusion Detection System

Network Intrusion Detection System (IDS)	
Description: This module monitors network packets in real time, extracts flow features, evaluates them using a trained ML model, and generates attack alerts.	
Attributes	**Description**
Network Interface	Stores the selected interface (Wi-Fi/Ethernet) for packet capturing.
Features (49)	Stores extracted statistical features of each flow
Model Output	Stores predicted attack class & confidence score.
Alerts	Stores alert records (type, confidence, timestamp).
Methods	**Description**
StartMonitoring()	**Pseudo Code**
	1. Detect available network interfaces. 2. User selects interface. 3. Start Scapy sniffer on selected interface. 4. For each packet: a. Identify flow using 5-tuple. b. Add packet to flow table. c. Trigger ExtractFeatures(). 5. Export flow data to traffic_log.json every 5 seconds.
PredictAttack()	**Pseudo Code**
	1. Read traffic_log.json.

	2. For each flow: a. Normalize features. b. Run ML model → get class & confidence. c. If confidence ≥ 85%: Call GenerateAlert(). 3. Save alerts to alerts.json.

Table 2 3.6.2 Algorithmic Viewpoint of Malware Analysis System

3.6.2 Algorithmic Viewpoint of Malware Analysis System

Malware Analysis System	
Description: This module monitors network packets in real time, extracts flow features, evaluates them using a trained ML model, and generates attack alerts.	
Attributes	**Description**
File Hashes	Stores MD5, SHA1, SHA256 of uploaded file.
Entropy score	Represents randomness of file (high = suspicious).
Strings Extracted	Contains suspicious strings from file.
API Indicators	Suspicious Windows API calls detected.
Threat Score	Combined score representing malware risk.
Methods	**Description**
StaticAnalysis()	**Pseudo Code**
	1. Calculate SHA256, MD5, SHA1 hashes. 2. Read first 8KB of file. 3. Compute entropy:

	entropy = -Σ(p(x) * log2(p(x))). 4. Detect packer signatures. 5. Extract printable strings (≥4 chars). 6. Save all attributes.
MultiSourceDetection()	**Pseudo Code**
	1. Query VirusTotal with SHA256. 2. Query MalwareBazaar with SHA256. 3. Query ThreatFox with SHA256. 4. Combine detection results.
GenerateReport()	**Pseudo Code**
	1. Collect: hashes, entropy, strings, behavior, API results. 2. Compute threat score. 3. Assign risk level (Low/Medium/High/Critical). 4. Save to malware_reports.json.

3.6.3 Algorithmic Viewpoint of Ransomware Detection System

Table 3 3.6.3 Algorithmic Viewpoint of Ransomware Detection System

Ransomware Detection System	
Description: This module provides real-time monitoring of critical folders to detect suspicious file operations indicating ransomware behavior.	
Attributes	**Description**
Monitored Paths	Directories selected for ransomware monitoring.

Suspicious Extensions	List of known ransomware extensions (.locky, .enc, etc.)
Entropy Threshold	Default threshold = 7.5 bits/byte.
Activity Log	Stores file modification events.
Methods	**Description**
StartMonitoring()	**Pseudo Code**
	1. User selects folders to monitor. 2. Start Watchdog observer. 3. For each file event (create/modify): a. Call DetectExtension(). b. Call CheckEntropy(). c. Call DetectRansomNote(). d. Determine threat level. e. If CRITICAL → AutoQuarantine().
CheckEntropy()	**Pseudo Code**
	1. Read first 1KB of file. 2. Calculate Shannon entropy. 3. If entropy > 7.5: return HIGH-RISK. 4. Else: return SAFE.

3.6.4 Algorithmic Viewpoint of User Authentication System

Table 4 3.6.4 Algorithmic Viewpoint of User Authentication System

User Authentication	
Description: Handles user login, session management, and access control to ensure secure usage of CyberHawk.	
Attributes	**Description**
Email	Registered email address of user.
Password	Hashed password stored in database.
Session ID	Unique session created after login.
Methods	**Description**
Login()	**Pseudo Code**
	1. Get email and password from form. 2. Query database for email. 3. If email not found: Print “Invalid Credentials”. 4. Verify password (bcrypt). 5. If incorrect: Print “Invalid Credentials”. 6. Else: Create session. Redirect to Dashboard.
Logout()	Destroys active session.
ValidateSession()	Checks if user is logged in and authorized.

Table 5 3.6.5 Algorithmic Viewpoint of Reporting System

3.6.5 Algorithmic Viewpoint of Reporting System

Reporting System	
Description: Generates security reports including network logs, attack stats, malware results, and ransomware activity.	
Attributes	**Description**
Report Type	Traffic, Malware, Ransomware, Combined.
Date Range	Time period selected by user.
Visualization Data	Charts and analytics generated from logs.
Methods	**Description**
GenerateReport()	**Pseudo Code**
	1. User selects report type & date range. 2. Fetch data from JSON logs. 3. Generate charts and analytics. 4. Format report template. 5. Display report to user.
ExportReport()	Allows downloading of selected format.

4. DEVELOPMENT AND TOOLS

This chapter highlights the development strategy, planning approach, and technologies used throughout the creation of CyberHawk. Since cybersecurity systems require continuous monitoring, real-time processing, and stability, careful planning and the selection of efficient tools were essential. This section therefore outlines the development plan, the tools used, configuration management practices, and finally the future enhancements planned for CyberHawk.

4.1 Introduction

During the development of CyberHawk, a systematic approach was adopted to divide the workload, schedule tasks, and identify the most suitable tools for each required component.

The project involved implementing:

- Network Intrusion Detection (IDS)
- Malware Analysis System
- Ransomware Detection System
- Web Dashboard (Frontend + Backend)
- Database and Reporting Layer
- System Integration (PHP ↔ Python)

This section describes the complete development workflow, team contributions, tools used for coding/testing, and future improvements.

4.2 Development Plan

The development plan includes task distribution among team members, timelines, and overall progress. This plan ensured that all modules—IDS, malware scanning, ransomware detection, UI development, and integration—were completed within academic deadlines.

Table 6 Development Plan Table

Team Member	Responsibilities
M Ahmed	1. Backend Development (PHP) 2. Python Detection Engines (IDS, Malware, Ransomware) 3. Security Integrations (VirusTotal, MalwareBazaar, ThreatFox) 4. Documentation
Hassan Javed	1. Frontend Development (HTML/CSS/Bootstrap/JavaScript) 2. Dashboard & Charts 3. UI Testing 4. Documentation Support
Dr Kashif Ayyub (Supervisor)	Verified testing, guidance, and implementation approval

Gantt Chart

Table 7 timeline Table

ID	Task Name	Start Date	End Date	Duration	Completion
1	Planning	05/01/2025	12/01/2025	7 days	100%
2	Proposal Writing	12/01/2025	22/01/2025	10 days	100%
3	Proposal Review & Approval	22/01/2025	30/01/2025	8 days	100%
4	Requirements Gathering (SRS)	30/01/2025	12/02/2025	13 days	100%
5	System Architecture + UML	12/02/2025	25/02/2025	13 days	100%
6	IDS Module Development	25/02/2025	20/03/2025	23 days	100%
7	Malware Analyzer Development	20/03/2025	10/04/2025	21 days	100%
8	Ransomware Detection Module	10/04/2025	28/04/2025	18 days	100%
9	Backend Dashboard Development	28/04/2025	20/05/2025	22 days	100%
10	Frontend UI + Charts	20/05/2025	10/06/2025	21 days	100%
11	Integration (PHP ↔ Python ↔ Database)	10/06/2025	18/06/2025	8 days	100%
12	Testing & Optimization	18/06/2025	30/06/2025	12 days	100%
13	Final Report & Submission	01/07/2025	15/07/2025	14 days	100%

4.3 Development Tools

To implement a multi-module system like CyberHawk, the project used a combination of programming languages, frameworks, APIs, and platforms. Each tool was selected based on performance, compatibility, and cybersecurity relevance.

4.3.1 Configuration management tools

Programming Languages

- **Python 3.9+**
  Used for IDS, malware analysis, ransomware monitoring, and machine learning model execution.
- **PHP 8.x**
  Used to develop the backend, APIs, routing, sessions, and integration logic.

- **JavaScript (ES6+) + AJAX (jQuery)**
  Used for asynchronous dashboard updates and UI interactivity.
- **HTML5/CSS3 + Bootstrap 5**
  Used for responsive frontend development.

External API Tools

- **VirusTotal API v3** – Multi-engine malware scanning
- **MalwareBazaar API (abuse.ch)** – Known malware hash lookup
- **ThreatFox IOC API** – Indicators of compromise lookup

Database Tools

- **phpMyAdmin** – Database management

Development & Testing Tools

- **XAMPP** – Apache, PHP, MySQL local server
- **Visual Studio Code** – Code development
- **Postman** – Testing backend routes & APIs
- **Git & GitHub** – Version control

4.4 Conclusion and Future Work/Extensions

In conclusion, the development of CyberHawk marks a significant advancement toward creating an intelligent, automated, and user-friendly cybersecurity platform capable of detecting network intrusions, identifying malware, and monitoring

ransomware activity in real time. The system successfully integrates multiple detection engines, external threat-intelligence APIs, and a responsive web dashboard, providing users with improved visibility and control over their digital security. Although CyberHawk meets its core objectives, future work may further enhance its capabilities through deeper automation, stronger machine-learning models, improved user experience, cloud deployment, and extended threat-intelligence integration. By adopting these potential enhancements, CyberHawk can evolve into a more scalable, enterprise-grade solution capable of countering increasingly sophisticated cyber threats and providing a more robust and comprehensive protection framework for users and organizations.

5. QUALITY ASSURANCE

In the system development process, ensuring quality is just as important as meeting functional requirements, especially for a cybersecurity system where accuracy, reliability, and stability are critical. This chapter discusses the quality assurance measures used during the development of CyberHawk. Multiple testing techniques, traceability matrices, and verification procedures were applied to ensure that the system performs as intended. A set of structured test cases and requirement mapping was used to validate each module of the system.

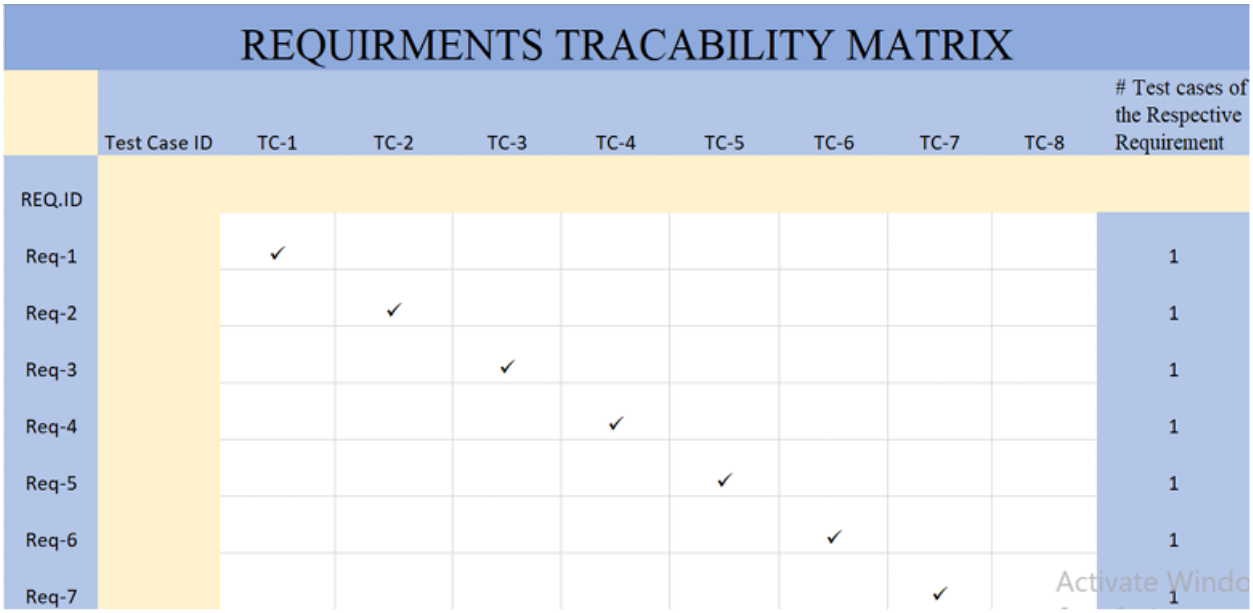
5.1 Introduction

During the quality assurance phase, a comprehensive testing mechanism was designed to ensure that CyberHawk meets user expectations and security standards. Various test cases were created to assess functional correctness, system reliability, performance, and user interface behavior. To ensure complete coverage, a **Requirement Traceability Matrix (RTM)** was developed, mapping each requirement to its corresponding test case. This ensures that every functional component—such as IDS monitoring, malware scanning, ransomware alerts, login system, and reporting—has been verified and aligned with defined requirements.

5.2 Traceability Matrix

The traceability matrix helps ensure that each requirement is linked to at least one valid test case and that no functionality is left untested. It provides a clear overview of the coverage between system requirements and test efforts.

5.2.1 Requirement Traceability Matrix



REQUIRMENTS TRACABILITY MATRIX

	Test Case ID	TC-1	TC-2	TC-3	TC-4	TC-5	TC-6	TC-7	TC-8	# Test cases of the Respective Requirement
REQ.ID										
Req-1		✓								1
Req-2			✓							1
Req-3				✓						1
Req-4					✓					1
Req-5						✓				1
Req-6							✓			1
Req-7								✓		1

5.3 Test Plan

The table below represents the test cases executed during the quality assurance phase of the CyberHawk system. These test cases were designed to validate core functionalities such as login, IDS monitoring, malware scanning, ransomware detection, API integrations, and overall system interaction.

Test Case 1:

Table 8 Test Case 1

Test Case ID	**TC-1**
Test Name	User Login with Valid Credentials
Date of Test	10/01/2025
Pre-condition	User is on login page
Actions	**System Response**
1. Users enter a valid email and password. 2. User clicks the "Login" button.	• System verifies hashed password. • User is redirected to the dashboard.
Expected Result	Users should successfully log into the dashboard.
Actual Result	Login successful and dashboard displayed.
Test Role (Actor):	Team Member
Test Verified By	Supervisor and Team Member
Status	Pass

Test case 2:

Table 9 Test Case 2

Test Case ID	**TC-2**
Test Name	Start IDS Monitoring
Date of Test	10/01/2025
Pre-condition	User is logged in and on IDS page
Actions	**System Response**
1. User clicks “Start Monitoring”. 2. Python engine starts packet sniffing.	• Network interface list displayed. • Monitoring begins and live data is updated.
Expected Result	IDS should start capturing and logging packets.
Actual Result	Monitoring started successfully.
Test Role (Actor):	Team Member
Test Verified By	Supervisor and Team Member
Status	Pass

Test case 3:

Table 10 Test case 3

Test Case ID	**TC-3**
Test Name	Malware File Upload & Scan
Date of Test	10/01/2025
Pre-condition	User is on Malware Analysis page
Actions	**System Response**
1. User uploads a suspicious file. 2. System calculates hashes and performs static analysis. 3. External APIs queried.	• VirusTotal/MalwareBazaar results displayed. • Final report generated.
Expected Result	System should analyze and classify the file.
Actual Result	File successfully scanned and report created.
Test Role (Actor):	Team Member
Test Verified By	Supervisor
Status	Pass

Test case 4:

Table 11 Test case 4

Test Case ID	**TC-4**
Test Name	Ransomware Detection – High Entropy File
Date of Test	11/01/2025
Pre-condition	Ransomware module active
Actions	**System Response**
1. User places encrypted-like file in monitored folder. **2.** Watchdog triggers event.	• High entropy detected. • System generates alert.
Expected Result	System should classify files as suspicious or critical.
Actual Result	Alert generated as expected.
Test Verified By	Supervisor
Status	Pass

Test case 5:

Table 12 Test Case 5

Test Case ID	**TC-5**
Test Name	API Integration – VirusTotal
Date of Test	11/01/2025
Pre-condition	Internet connectivity available
Actions	**System Response**
1. System sends hash to VirusTotal API.	• API returns engine results.
Expected Result	API should return correct malware classification data.
Actual Result	API successfully retrieved detection results.
Status	Pass

Test case 6:

Table 13 Test Case 6

Test Case ID	**TC-6**
Test Name	View Reports
Date of Test	12/01/2025
Pre-condition	System has existing logs
Actions	**System Response**
1. User selects report type. 2. Clicks “Generate Report”.	• Graphs and tables displayed.
Expected Result	Reports should load without errors.
Actual Result	Reports displayed correctly.
Status	Pass

Test case 7:

Table 14 Test Case 7

Test Case ID	**TC-7**
Test Name	Logout Function
Date of Test	12/01/2025
Pre-condition	User must be logged in
Actions	**System Response**
1. Click “Logout”.	• Session destroyed. • User redirected to login page.
Expected Result	Logout successful.
Actual Result	Passed
Status	Pass

6. USER MANUAL

Every system requires a clear and concise user manual to ensure that users—technical or non-technical—can operate the application effectively. This manual provides detailed guidelines prepared by the development team to help users understand the workings of CyberHawk. It explains the installation process, hardware/software requirements, and step-by-step operating instructions to ensure the system functions smoothly. This universal guide enables users to fully benefit from the features of CyberHawk.

6.1 Introduction

The purpose of this user manual is to provide end-users with a complete guide to interact with the CyberHawk security system. It outlines the hardware and software requirements needed to run the system efficiently and provides installation and operation instructions for all major modules, including the Network Intrusion Detection System (IDS), Malware Analyzer, Ransomware Detector, and the Web Dashboard. This manual ensures that both beginners and advanced users can operate the system with ease.

6.2 Hardware/Software Requirements for the System

Hardware Requirements

- Laptop or Desktop Computer
- Minimum 8 GB RAM (recommended: 16 GB for traffic analysis)
- 2.0 GHz Quad-Core Processor or higher
- At least 10 GB free disk space

- Network Adapter supporting packet capture

Software Requirements

- Windows 10/11 (recommended) or Linux
- Python 3.9+ installed
- PHP 8.x and Apache Server (XAMPP recommended)
- MySQL/MariaDB Database
- Internet connection for API-based malware lookup
- Web Browser (Chrome/Edge/Firefox)

6.3 Installation Guide for Application

Instead of installing Python engines locally, CyberHawk will be deployed on a **central server** where:

- IDS engine runs continuously on the server
- Malware & ransomware engines run on server-side Python services
- Users only access CyberHawk through a **web interface** (dashboard)