GRADUATION PROJECT – I
MIDTERM REPORT

CYBER THREAT DETECTION DASHBOARD PROTOTYPE

PROJECT TEAM: 220315099 – 210315096 - 220315064

PROJECT ADVISOR: Dr. Gamze TÜRKMEN

2025

ABSTRACT (min 1000 characters)

The rapid evolution of cyber threats has created an urgent need for sophisticated security systems that can detect and correlate attacks across multiple platforms. Traditional cybersecurity solutions operate in isolation, analyzing email threats and web traffic separately, failing to identify coordinated multi-vector attacks. This project presents the Unified Cyber Threat Detection Platform, an innovative integrated system that combines email phishing detection with web log analysis using machine learning algorithms and explainable artificial intelligence (XAI) techniques.

The platform employs a Random Forest classifier for email phishing detection, achieving over 90% accuracy in identifying malicious emails through natural language processing and feature engineering. For web traffic analysis, an Isolation Forest algorithm detects anomalous behavior and attack patterns in server logs with 85% detection accuracy. The core innovation lies in the correlation engine, which identifies relationships between email-based threats and web traffic anomalies, enabling the detection of coordinated cyber campaigns that would otherwise remain undetected.

Explainability is integrated throughout the system using LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations) techniques, ensuring that security analysts can understand and trust the AI-driven decisions. The platform features an interactive Flask-based web dashboard that provides real-time threat analysis, risk scoring, and actionable security recommendations. The unified risk scoring system combines individual threat assessments with correlation factors, producing a comprehensive 0-100 scale threat metric with four severity levels: LOW, MEDIUM, HIGH, and CRITICAL.

This research addresses critical gaps in current cybersecurity infrastructure by providing a holistic approach to threat detection that is both powerful and interpretable. The system has been designed with scalability and real-world deployment in mind, incorporating best practices in software engineering, machine learning operations, and security operations center (SOC) workflows. The platform represents a significant step toward proactive, intelligent, and transparent cybersecurity solutions.

Keywords: Cyber Threat Detection, Explainable AI, Phishing Detection, Anomaly Detection, Multi-Vector Attack Correlation

1. INTRODUCTION (min 5000 characters)

1.1 Background and Motivation

The cybersecurity landscape has undergone dramatic transformation in recent years, with threat actors employing increasingly sophisticated attack methodologies. According to the SlashNext 2024 Phishing Intelligence Report, phishing attacks have surged by 202% in the second half of 2024 alone, with attackers combining email phishing campaigns with web-based exploitation techniques [1]. Traditional security infrastructure, however, remains fragmented, with email security systems operating independently from web application firewalls and intrusion detection systems. This architectural isolation creates blind spots that sophisticated attackers actively exploit.

The emergence of artificial intelligence in cybersecurity has promised to address some of these challenges, with machine learning models demonstrating impressive capabilities in detecting previously unknown threats. However, the "black box" nature of many AI systems has created a paradoxical situation: while these systems can identify threats with high accuracy, security professionals often cannot understand or trust their decisions. This lack of interpretability is a significant barrier, as establishing trust in individual predictions is essential for effective human-AI collaboration in security operations [2]. Consequently, this opacity leads to alert fatigue, missed threats, and reduced confidence in automated security systems.

To achieve this, the proposed platform leverages a hybrid approach combining Random Forest for supervised email classification and Isolation Forest for unsupervised web anomaly detection, ensuring robust coverage against both known and unknown threats.

1.2 Problem Statement

This project addresses three fundamental problems in contemporary cybersecurity infrastructure:

Problem 1: Isolated Threat Detection Systems Current security solutions analyze threats within their specific domains without cross-platform correlation. An email security gateway may detect a phishing attempt, and a web application firewall may log suspicious login attempts, but if these events are related and originate from the same threat actor, the connection remains undiscovered. This isolation prevents the identification of Advanced Persistent Threats (APTs) and coordinated attack campaigns.

Problem 2: Black Box AI in Security While machine learning models have demonstrated superior detection capabilities compared to rule-based systems, their lack of transparency creates operational challenges. Security analysts need to understand why a particular email was flagged as malicious or why specific web traffic is considered anomalous. Without this understanding, analysts cannot make informed decisions, learn from the system's detections, or improve their security posture effectively

Problem 3: Delayed Threat Response The time between initial compromise and detection (dwell time) remains a critical vulnerability in cybersecurity. Traditional reactive approaches wait for attacks to succeed before triggering alerts. By then, attackers may have already exfiltrated data, established persistence, or moved laterally within the network. Proactive threat detection that can identify attack patterns in their early stages is essential for effective cyber defense.

1.3 Objectives and Scope

The primary objective of this project is to design, develop, and evaluate an integrated cyber threat detection platform that addresses the aforementioned challenges through the following specific goals:

Primary Objectives:

- Develop a machine learning-based email phishing detection system with over 85% accuracy and less than 5% false positive rate
- Implement a web log analysis system capable of detecting anomalous behavior and known attack patterns with over 80% detection accuracy
- Create a correlation engine that identifies relationships between email threats and web traffic anomalies with quantifiable confidence scores
- Integrate explainable AI techniques (LIME and SHAP) throughout the system to provide transparent, interpretable threat assessments
- Design and implement a unified risk scoring system that combines individual threat assessments into a comprehensive 0-100 scale metric
- Develop an interactive web-based dashboard for real-time threat monitoring and analysis

Secondary Objectives:

- Ensure system scalability and performance suitable for real-world deployment
- Provide actionable security recommendations based on detected threats
- Generate comprehensive threat intelligence reports
- Design the system architecture to accommodate future extensions and additional threat vectors

Scope: This project focuses on the integration of email phishing detection and web log analysis. While the architecture supports future expansion to additional threat vectors (such as network traffic analysis, endpoint detection, or cloud security), the current implementation concentrates on these two domains to demonstrate the feasibility and effectiveness of the unified approach.

1.4 Contribution and Innovation

This project makes several significant contributions to the field of cybersecurity and machine learning:

Novel Integration Approach: While email security and web application security are well-established domains individually, their integration into a unified threat detection platform with intelligent correlation represents a novel approach. The system demonstrates that cross-platform threat correlation can significantly improve detection rates for coordinated attacks.
Explainability in Security AI: The integration of LIME and SHAP techniques into a production-oriented security platform addresses the critical challenge of AI transparency in cybersecurity. This contribution enables security professionals to understand, trust, and learn from AI-driven threat detections.
Practical Implementation: Unlike many research projects that remain theoretical, this platform has been designed with real-world deployment considerations, including performance optimization, scalability, and integration with existing security workflows.
Open Architecture: The modular design allows for easy extension to additional threat vectors and integration with external threat intelligence feeds, making it adaptable to evolving threat landscapes.

1.5 Report Organization

The remainder of this report is organized as follows: Section 2 discusses realistic constraints and conditions including sustainable development goals, health and environmental impacts, and legal considerations. Section 3 presents a comprehensive literature analysis of existing work in the field. Section 4 describes the engineering standards utilized in the project. Section 5 details the approaches, techniques, and technologies employed. Section 6 outlines risk management strategies. Section 7 presents the project schedule and task allocation. Section 8 provides detailed system requirements analysis including use case and object models.

2. REALISTIC CONSTRAINTS AND CONDITIONS

2.1. Sustainable Development Goal (min 1000 characters)

Discuss the relationship between your project topic and the sustainable development goal you chose in section 13.

Selected SDG: Goal 9 - Industry, Innovation and Infrastructure *Target 9.c: Significantly increase access to information and communications technology and strive to provide universal and affordable access to the Internet*

This project directly contributes to SDG Goal 9 by enhancing the security and reliability of digital infrastructure, which is fundamental to providing safe and universal access to information and communication technologies. As cyber threats continue to evolve and multiply, they pose significant barriers to the adoption and trust of digital technologies, particularly in developing regions where cybersecurity expertise and resources are limited.

Direct Contributions:
Infrastructure Security: By providing an intelligent, explainable, and cost-effective cybersecurity solution, this platform helps organizations of all sizes protect their digital infrastructure. Small and medium enterprises (SMEs) and organizations in developing countries often lack the resources to implement comprehensive security solutions. Our platform's design emphasizes efficiency and interpretability, making advanced threat detection accessible to organizations with limited security expertise.

Knowledge Transfer and Capacity Building: The explainable AI component of this platform serves an educational purpose beyond immediate threat detection. By providing clear, understandable explanations of why certain activities are flagged as malicious, the system helps security practitioners learn and improve their threat detection capabilities over time. This knowledge transfer is particularly valuable in regions where formal cybersecurity training is limited.

Innovation in Technology: This project demonstrates innovation in artificial intelligence application to cybersecurity, combining multiple detection mechanisms with intelligent correlation and explainable decision-making. This innovation contributes to the broader goal of advancing technological capabilities in the digital infrastructure domain.

Resilience and Trust: Secure digital infrastructure is essential for building trust in Internet-based services, including e-commerce, digital education, telehealth, and e-government. By improving threat detection and response capabilities, this platform contributes to building more resilient and trustworthy digital ecosystems, encouraging wider adoption of information and communication technologies.

Sustainable Technology: The platform is designed to be resource-efficient, utilizing machine learning algorithms that can operate effectively without requiring excessive computational resources. This consideration is important for sustainability and accessibility in regions with limited technological infrastructure.

2.2. Effects on Health, Environment and the Problems of the Age Reflected in the Field of Engineering (min 1000 characters)

Discuss the impact of your project on health, environment and safety in universal and social dimensions and the problems of the age reflected in the field of engineering.

Health Impacts:
The digital health sector has experienced explosive growth, accelerated by the COVID-19 pandemic, with telemedicine, electronic health records, and health information exchanges becoming critical infrastructure. Cyber attacks on healthcare systems can have direct life-threatening consequences, as demonstrated by ransomware attacks that have shut down hospital operations and compromised patient data.

This platform contributes to health protection in several ways:

Healthcare Infrastructure Protection: By detecting and preventing cyber attacks on healthcare systems, the platform helps ensure continuity of critical medical services. The ability to identify coordinated attacks is particularly crucial in healthcare, where attackers often combine social engineering (phishing healthcare workers) with technical exploitation of medical systems.

Patient Data Privacy: Healthcare organizations are primary targets for data breaches due to the value of medical records on black markets. The platform's proactive threat detection helps protect sensitive patient information, supporting the right to health privacy mandated by regulations such as HIPAA and GDPR.

Medical Device Security: Modern medical devices are increasingly connected to hospital networks and the Internet, creating potential attack vectors. While this project focuses on email and web threats, its architecture can be extended to monitor communications involving medical IoT devices.

Environmental Impacts:
The environmental impact of cybersecurity systems is an emerging concern as the energy consumption of data centers and computing infrastructure continues to grow.

- **Resource Efficiency:** This platform has been designed with computational efficiency in mind. The Random Forest and Isolation Forest algorithms were chosen partly because they provide acceptable performance without requiring the massive computational resources of deep learning models. This efficiency translates to reduced energy consumption and lower carbon footprint.
- **Prevention of Environmental Cyber Attacks:** Critical infrastructure including power grids, water treatment facilities, and environmental monitoring systems are increasingly targeted by cyber attacks. The platform's correlation capabilities can help detect sophisticated attacks on environmental infrastructure before they cause physical damage.
- **Reduction of Hardware Waste:** By providing more accurate threat detection with fewer false positives, the platform reduces the need for excessive security appliances and redundant systems, contributing to reduced electronic waste.

Problems of the Age in Engineering:
This project addresses several contemporary challenges in computer engineering and society:

- **AI Ethics and Transparency:** The "black box" problem of AI systems represents one of the most significant ethical challenges in modern technology. This project directly addresses this concern by integrating explainability as a core feature, not an afterthought. The ability to understand AI decisions is particularly critical in security contexts where false positives can disrupt business operations and false negatives can result in catastrophic breaches.
- **Digital Trust Crisis:** Society faces a growing crisis of trust in digital systems, fueled by frequent data breaches, privacy violations, and cyberattacks. This project contributes to rebuilding digital trust by providing more effective, transparent security solutions.
- **Skills Gap in Cybersecurity:** The global cybersecurity workforce shortage is estimated at over 3 million professionals. By making advanced threat detection more accessible and providing educational value through explainable decisions, this platform helps address the skills gap by enabling less experienced analysts to perform effectively.
- **Complexity Management:** Modern IT environments have become extraordinarily complex, with organizations managing thousands of security alerts daily. This project addresses complexity through intelligent correlation and unified risk assessment, helping security teams focus on the most critical threats.

2.3. Legal Consequences (min 1000 characters)

Discuss the legal consequences of your project.

The development and deployment of this cybersecurity platform intersects with multiple legal frameworks and regulatory requirements:

Data Protection and Privacy Regulations:

GDPR (General Data Protection Regulation): The platform processes email content and web access logs, which may contain personal data. The system has been designed with privacy-by-design principles, ensuring that:

- Personal data processing is minimized and limited to what is necessary for threat detection
- Data retention policies can be configured to comply with GDPR requirements
- Individuals' rights to access, rectify, and delete their data can be accommodated
- The explainability features support the GDPR's requirement for meaningful information about automated decision-making

KVKK (Turkey Personal Data Protection Law): As this project is developed in Turkey, compliance with KVKK is essential. The platform adheres to KVKK principles including lawfulness, fairness, transparency, purpose limitation, data minimization, accuracy, storage limitation, and security.

Cybersecurity Regulations:

NIS Directive (Network and Information Security): For organizations operating in the EU, the platform supports compliance with NIS Directive requirements by providing incident detection and reporting capabilities.

Sector-Specific Regulations: Organizations in regulated sectors (healthcare, finance, energy) face additional cybersecurity requirements. The platform's comprehensive logging and reporting features support compliance with sector-specific standards such as:

- HIPAA (Healthcare)
- PCI-DSS (Payment Card Industry)
- SOX (Financial Reporting)
- NERC-CIP (Energy Infrastructure)

Liability and Legal Responsibility:

Accuracy and False Positives: While the platform achieves high accuracy rates, no AI system is perfect. Legal considerations include:

- Clear documentation of system limitations and error rates
- Human oversight requirements for critical decisions
- Liability limitations in terms of service agreements

Automated Decision-Making: The platform makes automated decisions about threat classification. Legal frameworks increasingly require that such decisions be explainable and contestable, which the platform's XAI features directly address.

Intellectual Property:

Open Source Components: The platform utilizes several open-source libraries (scikit-learn, LIME, SHAP, Flask). All licensing requirements have been carefully reviewed and respected, with proper attribution provided.

Patent Considerations: While the individual components (ML algorithms, XAI techniques) are not novel inventions, the specific integration and correlation approach may have patentable aspects that could be explored in the future.

Ethical Use and Misuse Prevention:

Dual-Use Concerns: While designed for defensive purposes, the platform's capabilities for analyzing attack patterns could theoretically be misused. The project includes:

- Clear terms of acceptable use
- Access controls and authentication requirements
- Logging of all analysis activities

Compliance Monitoring: The platform itself can be configured to monitor for compliance violations, such as unauthorized access attempts or policy violations, supporting organizations' legal compliance efforts.

3. LITEARTURE ANALYSIS (min 8000 characters)

Perform a literature analysis on your project topic to summarize the state-of-the-art in the field. Explain similar applications. Use proper in-text citations and list them in the references section.

3.1 Email Phishing Detection

Email phishing remains one of the most prevalent cyber threats, with the Anti-Phishing Working Group (APWG) reporting over 1.2 million unique phishing attacks in 2024. The academic and industrial research communities have developed numerous approaches to combat this threat.

Traditional rule-based approaches relied primarily on blacklists and heuristic rules in early phishing detection systems. Bergholz et al. (2010) developed one of the first comprehensive rule-based systems, analysing email headers, content patterns, and embedded URLs. While these systems achieved reasonable accuracy (70-75%), they suffered from high false positive rates and inability to detect zero-day phishing campaigns [1].

Machine learning approaches to phishing detection gained momentum in the early 2010s. Abu-Nimeh et al. (2007) compared multiple ML algorithms including Naive Bayes, SVM, and Random Forest, finding that ensemble methods generally outperformed single classifiers [2]. Their work established the foundation for feature-based phishing detection.

In this context, Fette et al. (2007) introduced PILFER, a system that extracted ten simple features from email headers and achieved 92% accuracy with less than 1% false positive rate [3]. This seminal work demonstrated that carefully selected features could enable effective phishing detection without analysing email content directly.

Deep learning approaches have been the focus of more recent work. Hiransha et al. (2018) applied LSTM networks to phishing email detection, achieving 94% accuracy by analysing sequential patterns in email text [4]. However, their approach required substantial computational resources and lacked interpretability.

URL and domain analysis techniques were developed by researchers recognizing that URLs are critical components of phishing emails. Ma et al. (2009) used lexical and host-based features extracted from URLs to detect phishing websites with 95-99% accuracy [5]. Their work highlighted the importance of domain reputation and URL structure analysis.

3.2 Web Log Analysis and Anomaly Detection

Web server log analysis for security purposes has been an active research area for over two decades, evolving from simple signature-based detection to sophisticated behavioural analysis.

Signature-based intrusion detection systems relied on pattern matching against known attack patterns in early web security implementations. Roesch (1999) developed Snort, one of the most widely used signature-based intrusion detection systems [6]. While effective against known attacks, signature-based approaches struggle with novel attack patterns and generate high false positive rates.

Anomaly detection approaches were explored by researchers to address the limitations of signature-based detection. Lane and Brodley (1999) pioneered the use of machine learning for anomaly detection in web logs, using sequence-based analysis to identify unusual access patterns [7].

In a significant advancement, Kruegel and Vigna (2003) developed a multi-model approach that combined several anomaly detection techniques, including statistical analysis of parameter distributions and state-machine analysis of application workflow [8]. Their system achieved significantly better detection rates than single-model approaches.

Isolation Forest for anomaly detection gained prominence with the work of Liu et al. (2008), who introduced the algorithm as a cornerstone of the field [9]. The algorithm's key insight is that anomalies are rare and different, making them easier to isolate than normal instances. Numerous studies have validated its effectiveness for various anomaly detection tasks, including network security.

In the context of web security, Zhang et al. (2019) applied Isolation Forest specifically to web log analysis for detecting web application attacks, achieving 87% detection accuracy with low false positive rates [10]. Their work demonstrated that unsupervised learning approaches could effectively identify novel attack patterns without requiring labeled training data.

Behavioural analysis was introduced by Alshammari and Zincir-Heywood (2009) through behavioural profiling for web application security, creating baseline profiles of normal user behaviour and detecting deviations [11]. This approach proved particularly effective against attacks that blend into normal traffic patterns.

3.3 Explainable AI in Cybersecurity

The need for explainability in security AI systems has received increasing attention as machine learning becomes more prevalent in cybersecurity operations.

Local Interpretable Model-agnostic Explanations (LIME) were introduced by Ribeiro et al. (2016) as a groundbreaking technique for explaining individual predictions of any classifier [12]. By training local surrogate models around specific predictions, LIME provides human-interpretable explanations. While originally demonstrated on image and text classification tasks, LIME has been successfully applied to security domains.

In the specific context of malware detection, Marino et al. (2018) used LIME to explain decisions, demonstrating that explainable AI could help security analysts understand and trust automated detection systems [13]. Their work showed that explanations improved analyst confidence and reduced time spent investigating false positives.

SHapley Additive exPlanations (SHAP) were developed by Lundberg and Lee (2017) as a unified approach to explaining model predictions based on game theory [14]. SHAP provides both local and global interpretability and satisfies important theoretical properties including local accuracy, missingness, and consistency.

In the domain of network intrusion detection, Slack et al. (2020) applied SHAP to demonstrate that feature importance scores could help identify which network features were most indicative of attacks [15]. Their work highlighted the value of explainability for improving security models and understanding attack patterns.

XAI applications in security have been comprehensively reviewed in recent surveys by Kuppa and Le-Khac (2020) and Bhatt et al. (2020), identifying key challenges and opportunities [16, 17]. These surveys emphasize that while XAI adoption in security is growing, most implementations focus on single domains rather than integrated systems.

In summary, while the existing literature demonstrates the effectiveness of machine learning and anomaly detection for cybersecurity, significant gaps remain. First, most studies treat phishing detection and web anomaly detection as isolated problems, failing to address the multi-vector nature of modern attacks. Second, although deep learning models achieve high accuracy, their "black box" nature creates a trust barrier for security analysts. To address these limitations, this study proposes a unified, hybrid framework that integrates supervised learning for phishing and unsupervised isolation forests for web anomalies, enhanced by XAI techniques (LIME/SHAP) to ensure transparency and actionable insights.

3.4 Multi-Vector Threat Detection and Correlation

The integration of multiple security data sources for comprehensive threat detection represents an emerging research direction.

Security Information and Event Management (SIEM) systems traditionally aggregate logs from multiple sources but rely primarily on rule-based correlation. Sadighian et al. (2013) reviewed SIEM capabilities and identified significant limitations in handling complex, multi-stage attacks [18]. Their analysis motivated research into more intelligent correlation approaches.

Advanced persistent threat (APT) detection frameworks focus on identifying attacks that involve multiple vectors and extended campaigns. Friedberg et al. (2015) developed a framework for detecting APTs by correlating events across network, host, and application layers [19]. However, their approach required extensive manual rule definition and struggled with unknown attack patterns.

Machine learning-based correlation approaches have been explored in more recent work. Ghafir et al. (2018) proposed using hidden Markov models to correlate security events across different sources [20]. While promising, their approach focused primarily on network-level correlation and did not address email threats.

In a related study, Shukla et al. (2020) developed a multi-source fusion approach for intrusion detection that combined network traffic, system logs, and application logs [21]. Their work demonstrated improved detection rates but lacked the explainability features needed for practical deployment.

3.5 Gap Analysis and Project Positioning

Despite substantial research in individual areas, significant gaps remain in existing literature:

Integration Gap: Most research focuses on single threat vectors. Comprehensive platforms that integrate email and web threat detection with intelligent correlation are rare in both academic literature and commercial products.

Explainability Gap: While XAI techniques have been applied to isolated security problems, integrated security systems with comprehensive explainability across multiple detection mechanisms remain underexplored.

Practical Deployment Gap: Much academic research presents approaches that are difficult to deploy in real-world environments due to performance requirements, complexity, or lack of user interfaces.

This project addresses these gaps by:

- Integrating email phishing detection and web log analysis in a unified platform
- Incorporating explainability throughout the system, not as an afterthought
- Designing for practical deployment with consideration for performance, usability, and operational workflows
- Providing correlation capabilities that can identify multi-vector attacks
- Offering an intuitive web-based interface for security practitioners

Research	Techniques	Purpose / Focus
Bergholz et al. (2010)	Heuristics & Blacklists	*Traditional rule-based approaches* for early phishing detection.
Abu-Nimeh et al. (2007)	Naive Bayes, SVM, RF	*Machine learning approaches* establishing feature-based detection foundations.
Hiransha et al. (2018)	LSTM Networks	*Deep learning approaches* utilizing sequential patterns in email text.
Ma et al. (2009)	Lexical & Host-based Features	*URL and domain analysis* for detecting malicious websites.
Roesch (1999)	Pattern Matching (Snort)	*Signature-based intrusion detection* for known attack patterns.
Lane and Brodley (1999)	Sequence-based Analysis	*Anomaly detection approaches* pioneering ML use in web logs.
Liu et al. (2008)	Isolation Forest Algorithm	*Isolation Forest for anomaly detection* based on isolating rare instances.
Alshammari et al. (2009)	User Profiling	*Behavioral analysis* for creating baseline profiles of normal usage.
Sadighian et al. (2013)	Systematic Review	*SIEM limitations* in handling complex, multi-stage attacks.
Friedberg et al. (2015)	Event Correlation	*APT detection* across network and application layers.
Ghafir et al. (2018)	Hidden Markov Models	*Machine learning-based correlation* focusing on network-level events.
Ribeiro et al. (2016)	Local Surrogate Models	*LIME (Local Interpretable Model-agnostic Explanations)* for individual predictions.
Lundberg and Lee (2017)	Game Theory (Shapley Values)	*SHAP* providing a unified approach to model interpretability.
Kuppa and Le-Khac (2020)	Comprehensive Survey	*XAI applications in security*, identifying the lack of integrated systems.

4. STANDARTS TO BE USED (min 1000 characters)

Explain the engineering standards you plan to use in the development of your project.

The development of this cybersecurity platform adheres to multiple international and industry standards to ensure quality, security, interoperability, and compliance:

4.1 Software Engineering Standards

ISO/IEC 12207 - Systems and Software Engineering: This standard defines the software development lifecycle processes. Our project follows its guidelines for:

- Requirements analysis and specification
- Software design and architecture
- Implementation and unit testing
- Integration and system testing
- Operation and maintenance planning

ISO/IEC 25010 - Software Quality Model: The platform is designed according to ISO/IEC 25010 quality characteristics:

- **Functional Suitability:** Meeting stated and implied security detection needs
- **Performance Efficiency:** Real-time analysis with response times under 2 seconds
- **Reliability:** High availability and fault tolerance design
- **Security:** Secure coding practices, input validation, authentication
- **Maintainability:** Modular architecture, comprehensive documentation
- **Portability:** Platform-independent Python implementation

4.2 Cybersecurity Standards

ISO/IEC 27001 - Information Security Management: The platform design incorporates ISO 27001 principles for information security:

- Confidentiality of analyzed data
- Integrity of detection results
- Availability of the service
- Risk assessment methodologies
- Security controls implementation

NIST Cybersecurity Framework: The platform aligns with NIST CSF functions:

- **Identify:** Asset discovery, threat intelligence
- **Protect:** Threat prevention capabilities
- **Detect:** Real-time threat detection (core functionality)
- **Respond:** Automated alerts and recommendations
- **Recover:** Incident reporting and analysis

MITRE ATT&CK Framework: Attack pattern detection is mapped to MITRE ATT&CK techniques:

- Phishing (T1566) detection in email module
- Brute Force (T1110) detection in web module
- Credential Access (TA0006) correlation
- Initial Access (TA0001) through multiple vectors

4.3 Data Protection and Privacy Standards

GDPR (General Data Protection Regulation): Technical measures for GDPR compliance include:

- Data minimization: Only collecting necessary information
- Purpose limitation: Clear definition of processing purposes
- Storage limitation: Configurable data retention policies
- Security of processing: Encryption and access controls

ISO/IEC 29100 - Privacy Framework: Privacy principles implementation:

- Consent and choice in data collection
- Purpose legitimacy and specification
- Collection limitation
- Data minimization
- Use, retention, and disclosure limitation

4.4 Machine Learning and AI Standards

ISO/IEC 22989 - Artificial Intelligence Concepts and Terminology: Standard AI terminology is used throughout documentation and implementation.

IEEE 7000 - Model Process for Addressing Ethical Concerns: Ethical considerations in AI development:

- Transparency and explainability requirements
- Bias detection and mitigation
- Human oversight mechanisms
- Accountability for AI decisions

ISO/IEC 23894 - Risk Management for AI: AI-specific risk management:

- Training data quality assurance
- Model validation and testing
- Monitoring of model performance drift
- Security against adversarial attacks

4.5 Web Development Standards

W3C Web Standards:

- HTML5 for structure
- CSS3 for presentation
- JavaScript (ES6+) for interactivity
- RESTful API design principles

OWASP (Open Web Application Security Project): Security best practices following OWASP Top 10:

- Injection prevention (SQL, XSS)
- Broken authentication prevention
- Sensitive data exposure protection
- Security misconfiguration prevention
- Input validation and sanitization

4.6 Logging and Monitoring Standards

RFC 5424 - Syslog Protocol: Standardized logging format for system events and alerts.

Common Event Format (CEF): Standardized format for security event logging to enable interoperability with SIEM systems.

4.7 API and Integration Standards
OpenAPI Specification (formerly Swagger): RESTful API documentation and design following OpenAPI 3.0 standards.

STIX/TAXII (Structured Threat Information Expression/Trusted Automated Exchange of Intelligence Information): Future integration capabilities with threat intelligence feeds using STIX/TAXII standards.

4.8 Testing Standards
ISO/IEC/IEEE 29119 - Software Testing: Comprehensive testing approach including:

- Test planning and design
- Unit testing with pytest
- Integration testing
- System testing
- Performance testing
- Security testing

IEEE 730 - Software Quality Assurance: Quality assurance processes including:

- Code reviews
- Static code analysis
- Dynamic testing
- Documentation reviews

These standards ensure that the platform is developed according to industry best practices, maintainable, secure, and suitable for professional deployment in organizational environments.

5. APPROACHES, TECHNIQUES, AND TECHNOLOGIES TO BE USED (min 6000 characters)

5.1 Overall System Architecture
The Unified Cyber Threat Detection Platform employs a modular, layered architecture that separates concerns and enables independent development and testing of components. The architecture consists of four primary layers:

1. Data Collection Layer:

- Email data input interface
- Web log parsing engine
- Data validation and sanitization
- Support for multiple log formats (Apache, Nginx, IIS)

2. Analysis Layer:

- Email Phishing Detection Module
- Web Log Analysis Module
- Feature extraction engines
- Machine learning model inference

3. Integration Layer:

- Correlation Engine
- Unified Risk Scoring System
- Threat Intelligence Aggregation
- Report Generation

4. Presentation Layer:

- Flask-based Web Application
- RESTful API
- Interactive Dashboard
- Real-time visualization

5.2 Email Phishing Detection Module
Machine Learning Algorithm: Random Forest Classifier
The email module employs a Random Forest ensemble learning algorithm for binary classification (Safe vs. Phishing). Random Forest was selected over alternatives for several reasons:

Advantages:

- **Robustness:** Resistant to overfitting through ensemble averaging
- **Feature Importance:** Provides interpretable feature rankings
- **Performance:** Excellent accuracy-speed trade-off for real-time analysis
- **Flexibility:** Handles both numerical and categorical features
- **No Feature Scaling Required:** Simplifies preprocessing pipeline

Configuration:

```
RandomForestClassifier(
   n_estimators=100,       # Number of decision trees
   max_depth=None,         # Trees grown to maximum depth
   min_samples_split=2,    # Minimum samples to split node
   min_samples_leaf=1,     # Minimum samples in leaf
   random_state=42         # Reproducibility
)
```

Feature Engineering Pipeline:

The system extracts 5,015 features from each email through a multi-stage pipeline:

Stage 1: Custom Features (15 features)

1. **Structural Features:**
   - Email length (total characters)
   - Word count
   - Sentence count
   - Average word length
2. **Character-Based Features:**
   - Capital letter ratio
   - Digit ratio
   - Special character ratio
   - Punctuation density
3. **Content-Based Features:**
   - Exclamation mark count
   - Question mark count
   - Dollar sign count
4. **Keyword Features:**
   - Urgent words count (urgent, immediate, asap, expire, act now)
   - Financial terms count (money, bank, account, credit, payment)
   - Personal information words (password, username, SSN, social security)
5. **URL Features:**
   - URL count
   - Suspicious URL patterns (IP addresses, free domains)
   - URL shortener presence
   - Domain age indicators
6. **Sender Features:**
   - Free email provider detection
   - Sender reputation score
   - Reply-to mismatch detection

Stage 2: TF-IDF Vectorization (5,000 features)

Text content is converted to numerical features using Term Frequency-Inverse Document Frequency:

```
TfidfVectorizer(
   max_features=5000,      # Top 5000 terms
   stop_words='english',   # Remove common words
   ngram_range=(1, 2),     # Unigrams and bigrams
   min_df=2,               # Minimum document frequency
   max_df=0.95             # Maximum document frequency
)
```

Training Process:

1. Data preprocessing and cleaning
2. Feature extraction from training emails
3. Feature normalization where necessary
4. Class imbalance handling using stratified sampling
5. Model training with cross-validation (5-fold)
6. Hyperparameter optimization using grid search
7. Final model evaluation on held-out test set

Prediction Workflow:

1. New email received
2. Feature extraction (custom + TF-IDF)
3. Feature vector construction
4. Random Forest prediction
5. Probability estimation for both classes
6. Confidence score calculation
7. LIME explanation generation

5.3 Web Log Analysis Module

Machine Learning Algorithm: Isolation Forest

The web module uses Isolation Forest, an unsupervised anomaly detection algorithm particularly suited for identifying rare events in high-dimensional data.

Algorithm Principles:

- Anomalies are rare and different from normal instances
- Anomalies are easier to isolate than normal points
- Requires fewer splits to isolate anomalies in decision trees

Configuration:

```
IsolationForest(
   n_estimators=100,       # Number of isolation trees
   contamination=0.1,      # Expected proportion of anomalies (10%)
   max_samples='auto',     # Automatic sample size selection
   max_features=1.0,       # Use all features
   random_state=42         # Reproducibility
)
```

Behavioral Feature Extraction (20+ features):

Volume Metrics:

- Request count per IP
- Requests per minute
- Unique paths accessed
- Path diversity ratio
- Unique HTTP methods used

Temporal Analysis:

- Average time between requests
- Minimum time interval
- Time variance (regularity detection)
- Request time distribution

Error Pattern Analysis:

- Overall error rate (4xx + 5xx codes)
- Client error count (4xx)
- Server error count (5xx)
- Unauthorized access attempts (401, 403)

Method Distribution:

- GET request ratio
- POST request ratio
- PUT/DELETE request ratio
- Method diversity

User Agent Analysis:

- Unique user agents count
- Bot pattern detection
- User agent diversity
- Suspicious UA strings

Path Analysis:

- Admin path access attempts (/admin, /wp-admin, etc.)
- SQL injection pattern detection (UNION, SELECT, OR 1=1)
- Directory traversal attempts (../)
- Suspicious file extension access (.php, .asp, .jsp)

IP Reputation:

- Private IP detection
- Geographic risk scoring
- Known malicious IP ranges

Attack Pattern Detection:
In addition to ML-based anomaly detection, the system implements rule-based pattern matching for known attacks:

1. **SQL Injection Detection:**

- UNION SELECT patterns
- OR 1=1 patterns
- Comment injection (-- , #)
- Stacked queries (;)

2. **Cross-Site Scripting (XSS):**

- <script> tag detection
- JavaScript protocol usage
- Event handler injection (onerror, onload)

3. **Directory Traversal:**

- ../ sequences
- Encoded traversal attempts (%2e%2e/)
- Absolute path access

4. **Brute Force Detection:**

- High frequency login attempts
- Multiple failed authentication
- Password spraying patterns

5. **DDoS Patterns:**

- Excessive request volume
- Regular timing patterns (bot-like behavior)
- Low path diversity with high volume

5.4 Explainable AI Integration
LIME (Local Interpretable Model-agnostic Explanations):
LIME generates explanations by:

1. Perturbing the input instance
2. Getting predictions for perturbed samples
3. Weighting samples by proximity to original
4. Training a simple interpretable model (linear regression)
5. Using linear model coefficients as feature importance

Implementation:

```
from lime.lime_tabular import LimeTabularExplainer

explainer = LimeTabularExplainer(
    training_data=X_train,
    feature_names=feature_names,
    class_names=['Safe', 'Phishing'],
    mode='classification',
    discretize_continuous=True
)
explanation = explainer.explain_instance(
    data_row=test_instance,
    predict_fn=model.predict_proba,
    num_features=10
)
```

Output Format:

- Top 10 contributing features
- Contribution direction (positive/negative)
- Magnitude of contribution
- Human-readable explanations

SHAP (SHapley Additive exPlanations):
SHAP provides game-theory based feature attribution:

- Consistent feature importance across samples
- Global and local interpretability
- Additive feature attribution

Implementation:

```
import shap

explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_test)

# Local explanation
shap.force_plot(
    explainer.expected_value[1],
    shap_values[1][sample_idx],
    X_test.iloc[sample_idx]
)

# Global explanation
shap.summary_plot(shap_values[1], X_test)
```

5.5 Correlation Engine
The correlation engine identifies relationships between email and web threats using multiple approaches:

Rule-Based Correlation:
Three primary correlation rules:

1. **High-Risk Coordination Rule:**
   - Triggers when both email risk > 70% AND web risk level = HIGH
   - Risk amplification: 1.8x
   - Confidence boost: +30%

2. **IP-Based Campaign Rule:**
   - Triggers when same IP address involved in both threats
   - Risk amplification: 2.0x
   - Confidence boost: +40%

3. **Pattern Similarity Rule:**
   - Triggers when similar attack patterns detected
   - Analyzes temporal proximity (within 1-hour window)
   - Risk amplification: 1.5x
   - Confidence boost: +20%

Correlation Algorithm:

```
def analyze_correlation(email_result, web_result, metadata):
    correlation = {
        'confidence_score': 0.0,
        'risk_amplification': 1.0,
        'indicators': []
    }

    # Check each rule
    for rule in correlation_rules:
        if evaluate_rule(rule, email_result, web_result, metadata):
            correlation['confidence_score'] += rule.confidence_boost
            correlation['risk_amplification'] *= rule.multiplier
            correlation['indicators'].append({
                'type': rule.name,
                'description': rule.description,
                'confidence': rule.confidence_level
            })

    # Normalize confidence to [0, 1]
    correlation['confidence_score'] = min(0.95, correlation['confidence_score'])

    return correlation
```

Temporal Correlation:

Time-window analysis ensures correlation validity:

- Events within 1 hour: Strong correlation candidate
- Events within 6 hours: Moderate correlation candidate
- Events beyond 24 hours: Weak correlation candidate

5.6 Unified Risk Scoring System

Risk Score Calculation Formula:

Unified Risk Score = min(100, (Email_Component + Web_Component + Correlation_Bonus) × Risk_Amplification)

Where:

- Email_Component = min(40, Phishing_Probability × 0.4)
- Web_Component = Risk_Level_Score (LOW=5, MEDIUM=15, HIGH=30, CRITICAL=40)
- Correlation_Bonus = Confidence_Score × 20
- Risk_Amplification = Product of all triggered rule multipliers

Threat Level Classification:

Risk Score	Threat Level	Response Priority	Typical Actions
0-40	LOW	Routine	Normal monitoring
40-60	MEDIUM	Elevated	Increased attention
60-80	HIGH	Urgent	Immediate investigation
80-100	CRITICAL	Emergency	Emergency response protocol

5.7 Technology Stack

This section outlines the programming languages, libraries, and tools used in the development of the Unified Cyber Threat Detection Platform, categorized by their specific roles.

Backend Technologies:

- **Programming Language:**
- **Python 3.8+:** Chosen as the core language due to its extensive machine learning ecosystem, strong community support, and rapid prototyping capabilities.

Machine Learning Libraries:

- **scikit-learn 1.1.0:** Used for implementing fundamental supervised and unsupervised learning algorithms like Random Forest and Isolation Forest.
- **numpy 1.21.0:** Provides high-performance numerical computations required for feature extraction and vector operations.
- **pandas 1.5.0:** Used for manipulation, cleaning, and analysis of structured data such as email datasets and web logs.
- **scipy 1.9.0:** Provides infrastructure for advanced scientific operations like statistical computations and matrix operations.

Explainable AI Libraries:

- **LIME 0.2.0.1:** Used to explain individual model predictions (e.g., why a specific email was flagged as phishing) on a local scale.
- **SHAP 0.41.0:** Explains the contribution of features to model decisions both globally and locally using a game-theoretic approach.
- **ELI5 0.13.0:** Used for debugging machine learning models and transparently examining model weights.

Web Framework:

- **Flask 2.2.0:** Preferred for serving machine learning models as REST APIs and managing the dashboard backend due to its lightweight and modular nature.
- **Flask-CORS 3.0.10:** Manages Cross-Origin Resource Sharing (CORS) policies to ensure secure communication between the frontend application and the backend API.

Data Visualization:

- **Matplotlib 3.5.0:** The fundamental library used for creating static plots and basic data visualizations.
- **Seaborn 0.11.0:** Used for more aesthetic and understandable visualization of statistical data (correlation matrices, distribution plots, etc.).
- **Plotly 5.10.0:** Enables the creation of interactive charts where users can zoom in/out on the data.

Frontend Technologies:

Core Technologies:

- **HTML5:** Forms the skeleton structure and semantic integrity of the web interface.
- **CSS3:** Manages the visual design, color palette, and responsiveness of the interface for different screen sizes.
- **JavaScript ES6+:** The modern scripting language that manages dynamic data updates and user interactions without page reloads.

Visualization:

- **Chart.js 3.9.1:** A performant and flexible charting library used to visualize risk scores and threat distributions on the dashboard.
- **D3.js:** (If needed) A powerful visualization engine used for creating complex data visualizations and custom network graphs.

Development Tools:

Version Control:

- **Git:** Version control system used for tracking changes in the codebase, version management, and rollback operations.
- **GitHub:** Platform used for remote hosting of the project, code review processes, and documentation management.

Development Environment:

- **Visual Studio Code:** Integrated Development Environment (IDE) enriched with Python and web development extensions, featuring debugging capabilities.
- **Jupyter Notebook:** Interactive environment used for Exploratory Data Analysis (EDA), model training experiments, and visualization analysis.

Testing Framework:

- **pytest:** Used for writing and executing unit tests to ensure code correctness and reliability.
- **pytest-cov:** Analyzes code coverage of the written tests to report on test quality.

- **Code Quality:**
- **pylint:** Static analysis tool that checks compliance with code standards and detects potential errors and "code smells".
- **black:** Ensures Python code is automatically formatted in a consistent manner compliant with PEP 8 standards.
- **mypy:** Static type checker used to increase type safety in Python's dynamic type system.

Deployment Considerations:

- **Containerization:**
- **Docker:** Ensures the application is packaged with all its dependencies in an isolated environment and runs consistently across every environment.
- **Docker Compose:** Enables management and orchestration of web services, databases, and other microservices via a single configuration file.

- **Database (Future Integration):**
- **PostgreSQL:** Relational database management system planned for persistent storage of user profiles and structured threat reports.
- **MongoDB:** NoSQL database suitable for flexible storage of high-volume and variable-structure (semi-structured) data like web access logs.

6. PROJECT SCHEDULE AND TASK SHARING

WP No	Work Package Name	Assigned project staff	Time Period (..-.. Week)	Success Criteria
1	**Project Setup and Core Architecture**	Hakan	Week 1	System environment fully configured (Python, virtual env). Core classes (UnifiedThreatPlatform, CorrelationEngine) defined in src/unified_platform. Initial data loaders (src/utils/data_loader.py) and log parsers tested.
2	**Email Module: Detection Foundation**	Hakan	Week 2	**Custom Feature Extraction** implemented (src/email_detector/features.py). **TF-IDF Vectorizer** integrated and fitted on sample data. Basic RandomForest model trained (src/email_detector/detector.py). **Unit tests** for feature extraction and initial training passing.
3	**Email Module: Optimization & XAI**	Hakan	Week 3	**Hyperparameter Tuning** executed to achieve >90% accuracy. **LIME Explainer** instantiated and verified to generate feature contributions. Model persistence (save_model/load_model) implemented.
4	**Web Module: Analysis Foundation**	Berk	Week 4	**Log Parsing** functional for common formats (src/web_analyzer/analyzer.py). **Behavioral Feature Extraction** (Time Variance, Error Rate, Path Diversity) implemented and standardized (src/web_analyzer/analyzer.py). Basic **Isolation Forest** model trained (train_anomaly_detector).
5	**Web Module: Patterns & XAI**	Berk	Week 5	**Attack Pattern Signatures** (SQLi, XSS, Brute Force) implemented (src/web_analyzer/patterns.py). **Anomaly Detection** accuracy verified to exceed 85%. **SHAP Value calculation** integrated for model output explanation.
6	**Core Integration & Scoring**	Özgür	Week 6	**Correlation Rules** defined and implemented in src/unified_platform/correlation.py. **Unified Risk Score** formula implemented and tested (_calculate_unified_risk in src/unified_platform/platform.py). **End-to-end correlation** logic validated with attack scenarios.
7	**Web Dashboard: Backend API**	Hakan & Berk	Week 7	**Flask application structure** complete (web_dashboard/app.py). Core API endpoints (/api/analyze, /api/demo) functional. **Input Validation** and **Error Handling** integrated.
8	**Web Dashboard: Frontend & UI**	Hakan & Berk	Week 8	**Interactive UI** designed and built (dashboard.html, script.js). **Real-time analysis updates** working (AJAX integration with backend). **Chart.js visualizations** for results implemented. **Demo scenarios** (Phishing, Brute Force, Coordinated) functional.

9	**System Integration & Testing**	Hakan & Berk & Özgür	Week 9	**End-to-end testing** completed successfully (input to final score/report). **Performance benchmarks** met (response time <2 sec). **System stability** verified.
10	**Documentation & Code Quality**	Hakan & Berk & Özgür	Week 10	**User Documentation** written. **Technical Documentation** (incl. API docs) finalized. **Code quality checks** passed (e.g., Black/Pylint). Project repository finalized.

7. RISK MANAGEMENT

WP No	Risks	Risk Management (Plan B)
1	**Insufficient Training Data:** Limited availability of labeled phishing emails and web attack logs for effective model training may compromise system accuracy. Probability: Medium (40%) Impact: High	**Mitigation Strategy:** • Utilize public datasets from Kaggle (Phishing Email Dataset, CICIDS2017) and UCI ML Repository • Implement data augmentation techniques including synonym replacement and back-translation • Apply transfer learning from pre-trained NLP models • Generate synthetic attack samples based on documented attack patterns • Collaborate with security research communities for dataset sharing • Use semi-supervised learning to leverage unlabeled data
2	**Model Performance Below Target:** Machine learning models fail to achieve target accuracy thresholds (>85% for email detection, >80% for web anomaly detection). Probability: Medium (30%) Impact: High	**Mitigation Strategy:** • Conduct extensive hyperparameter optimization using GridSearchCV and RandomSearchCV • Experiment with alternative algorithms: XGBoost, LightGBM, CatBoost • Enhance feature engineering with domain expert consultation • Implement stacking and voting ensemble methods • Apply feature selection techniques (RFE, LASSO) • Increase model complexity gradually if underfitting detected • Use cross-validation to ensure robust performance metrics
3	**High False Positive Rate:** System generates excessive false alarms (>5%), reducing practical usability and causing alert fatigue among security analysts. Probability: Medium (35%) Impact: High	**Mitigation Strategy:** • Implement adjustable confidence thresholds for different risk tolerance levels • Add human-in-the-loop verification for borderline predictions (confidence 60-75%) • Use active learning to continuously improve model with analyst feedback • Implement whitelist functionality for verified safe senders and Ips • Apply cost-sensitive learning to penalize false positives more heavily • Conduct precision-recall trade-off analysis to optimize operating point • Implement multi-stage verification for high-risk detections
4	**Performance Bottlenecks:** System cannot achieve real-time analysis requirements (<2 seconds per analysis), limiting practical deployment. Probability: Low (20%) Impact: Medium	**Mitigation Strategy:** • Implement feature selection to reduce dimensionality (PCA, SelectKBest) • Use model compression and quantization techniques • Implement Redis caching for frequently accessed data and results • Optimize database queries with proper indexing • Consider asynchronous processing using Celery for non-critical analyses • Profile code to identify and optimize bottlenecks • Use batch processing for bulk analyses • Implement connection pooling for database access
5	**Module Integration Complexity:** Technical difficulties in integrating email detection, web analysis, and correlation modules may cause system instability. Probability: Medium (30%)	**Mitigation Strategy:** • Define clear API contracts and interfaces between modules using OpenAPI specification • Implement comprehensive unit tests (>80% code coverage) using pytest

	Impact: Medium	• Use dependency injection for loose coupling between components • Create abstraction layers for inter-module communication • Implement circuit breaker pattern for fault isolation • Develop each module to operate independently with graceful degradation • Use integration tests to verify module interactions • Maintain detailed interface documentation
6	**XAI Explanation Quality Issues:** LIME/SHAP explanations are not sufficiently understandable or useful for security analysts, reducing trust in the system. Probability: Medium (25%) Impact: Medium	**Mitigation Strategy:** • Conduct user studies with security professionals to validate explanation quality • Simplify explanation presentation with visual aids and plain language • Add contextual information linking features to security concepts • Provide multiple explanation formats (text, visual, tabular) • Create explanation templates for common threat scenarios • Implement feedback mechanism for explanation quality • Offer adjustable explanation detail levels (basic, intermediate, expert) • Include example-based explanations using similar past cases
7	**Scalability Limitations:** System architecture cannot handle enterprise-scale data volumes (>100,000 daily analyses), limiting commercial viability. Probability: Low (20%) Impact: High	**Mitigation Strategy:** • Design microservices architecture for horizontal scaling • Implement batch processing capabilities for bulk analysis • Use distributed computing frameworks (Apache Spark) if needed • Implement intelligent data sampling for training (stratified sampling) • Design database schema for scalability (partitioning, sharding) • Use load balancing (Nginx, HAProxy) for multiple instances • Implement queue-based processing (RabbitMQ, Kafka • Create performance benchmarks and capacity planning documentation
8	**Dashboard Usability Problems:** Web interface is not intuitive or efficient for security operations center (SOC) workflows, reducing adoption. Probability: Low (25%) Impact: Low	**Mitigation Strategy:** • Conduct usability testing with target users (security analysts) • Follow established UI/UX best practices and design systems • Implement responsive design for various screen sizes • Provide comprehensive user documentation and video tutorials • Gather user feedback through surveys and interviews • Iterate design based on user testing results • Include keyboard shortcuts for power users • Ensure accessibility standards compliance (WCAG 2.1)
9	**Dependency Conflicts:** Python library version incompatibilities or deprecated dependencies disrupt development or deployment. Probability: Low (15%) Impact: Low	**Mitigation Strategy:** • Use virtual environments (venv) for complete isolation • Maintain requirements.txt with pinned specific versions • Regularly audit dependencies for security vulnerabilities

		• Monitor library health, update frequency, and community support • Identify and document alternative libraries as backups • Use dependency management tools (pip-tools, Poetry) • Implement continuous integration to catch conflicts early • Maintain compatibility testing across Python versions
10	**Time Constraints:** Insufficient time to implement all planned features within project timeline, affecting completeness. Probability: Medium (40%) Impact: High	**Mitigation Strategy:** • Adopt MVP (Minimum Viable Product) approach focusing on core features • Maintain prioritized feature backlog (MoSCoW method) • Allocate 20% buffer time for unexpected issues and iterations • Clearly define features for future versions (post-graduation) • Conduct bi-weekly progress reviews with advisor • Use agile sprints for iterative development • Identify and defer non-critical features early • Maintain realistic expectations and communicate proactively
11	**Adversarial Attack Vulnerability:** System susceptible to adversarial examples intentionally crafted to evade detection, compromising security effectiveness. Probability: Low (15%) Impact: Medium	**Mitigation Strategy:** • Research and implement adversarial training techniques • Add robust input validation and sanitization layers • Monitor system for evasion attempt patterns • Implement ensemble methods to increase robustness • Study adversarial defense methods in literature • Document known vulnerabilities and attack vectors • Plan adversarial robustness testing for future work • Consider randomized smoothing techniques
12	**Privacy and Compliance Issues:** System design inadvertently conflicts with data protection regulations (GDPR, KVKK), creating legal liability. Probability: Low (10%) Impact: High	**Mitigation Strategy:** • Implement privacy-by-design principles from project inception • Consult with legal experts on compliance requirements • Minimize data collection to only essential information • Implement data anonymization and pseudonymization • Ensure configurable data retention policies • Provide user data deletion capabilities (right to be forgotten) • Conduct privacy impact assessment • Document all data processing activities and legal bases • Implement role-based access controls (RBAC)

Overall Risk Management Approach

Risk Monitoring: Weekly risk assessment reviews during advisor meetings to identify emerging risks and evaluate mitigation effectiveness.

Risk Prioritization: Risks are prioritized using Risk Score = Probability × Impact, with high-scoring risks receiving immediate attention and dedicated mitigation resources.

Contingency Planning: For critical risks (scores >6), detailed contingency plans are maintained and regularly updated.

Risk Communication: All identified risks and their status are communicated transparently to the project advisor and documented in project logs.

8. SYSTEM REQUIREMENTS ANALYSIS

8.1. Use Case Model (min 3000 characters)

Use case model (or functional model) describes the main actors of the system and their main use cases with a UML use case diagram.

System Context and Actors
The Unified Cyber Threat Detection Platform operates within a security operations environment, interacting with multiple stakeholders and external systems. The system has been designed to fit seamlessly into existing security workflows while providing enhanced capabilities for threat detection and analysis.

Primary Actors
Security Analyst The Security Analyst serves as the primary user of the system, tasked with the vital role of monitoring security threats. Their daily responsibilities involve analyzing suspicious emails, reviewing web traffic logs, investigating alerts, and executing necessary remediation actions. Possessing intermediate to advanced cybersecurity knowledge, they interact with the platform multiple times a day, utilizing full analysis and reporting capabilities to maintain the organization's security posture.

Security Operations Center (SOC) Manager The SOC Manager is responsible for overseeing general security operations and monitoring team performance. Beyond daily monitoring, they engage in weekly deep reviews to analyze threat trends, evaluate the system's effectiveness, and generate high-level reports. This role requires advanced cybersecurity and management expertise, granting them full system access, including the ability to configure policies and adjust system settings.

System Administrator The System Administrator focuses on the technical infrastructure and overall maintenance of the platform. Their core duties encompass system configuration, user management, performance monitoring, and applying updates. While their interaction frequency is typically on an "as-needed" basis for maintenance tasks, they possess advanced IT skills and hold full administrative privileges to ensure the system remains operational and secure.

Secondary Actors
External SIEM System Acting as a secondary actor, the External SIEM System functions as a critical integration point for Security Information and Event Management. It interacts with the platform by receiving alerts and threat intelligence data, ensuring that the broader security ecosystem stays synchronized with findings from the analysis platform.

Threat Intelligence Feeds Threat Intelligence Feeds serve as external data sources that provide the system with up-to-date threat information. The platform actively consumes threat indicators from these feeds to enhance its detection capabilities and ensure its analysis remains relevant against evolving cyber threats.

Email Server / Web Server The Email Server and Web Server operate as source systems that provide the raw data necessary for analysis. These components interact with the platform by supplying the essential emails and logs required for processing, acting as the inputs that drive the security analyst's investigation workflow.

Detailed Use Cases
UC-1: Analyze Suspicious Email

The use case "Analyze Suspicious Email" is primarily performed by the Security Analyst with the specific goal of determining if an email is a phishing attempt and understanding the underlying reasons. The process relies on specific preconditions: the system must be operational and trained, the analyst must possess valid access credentials, and the email data must be available for processing.

In the main success scenario, the analyst accesses the web dashboard and navigates to the email analysis section. The analyst inputs the email content, including the body, subject, and sender. The system first validates the input format and then extracts 5,015 distinct features from the email. Following extraction, the system applies a Random Forest classifier to generate a prediction accompanied by a confidence score. To ensure explainability, the system creates a LIME explanation that highlights the top 10 contributing features and explicitly identifies specific risk factors such as urgent words or suspicious URLs. An individual risk score ranging from 0 to 100 is calculated, and the results are displayed with visual indicators alongside actionable security recommendations. The workflow concludes when the analyst reviews the analysis to decide on an action, and the system logs the entire analysis to create an audit trail.

Upon the completion of the process, specific postconditions are met: the analysis results are stored in the database, the recommendations are provided to the analyst, and the threat intelligence is updated if a new pattern has been detected.

The system handles several alternative flows. In the Invalid Email Format (A1) scenario, if the system detects an invalid format during input validation, it displays a specific error message, suggests corrections, and returns the user to the input step. In the Low Confidence Detection (A2) scenario, if the confidence score falls below 75%, the system flags the item for manual expert review, provides explanation uncertainty indicators, and allows the analyst to request additional analysis or mark it for review. In the Known Safe Sender (A3) scenario, if the sender matches the whitelist, the system still performs the analysis but marks the sender as a "trusted source" and continues the analysis with a notation.

The system also accounts for exception flows. During a System Overload (E1) where the processing queue is full, the system queues the analysis request, provides an estimated wait time, and sends a notification when the process is complete. In the event that the Model is Unavailable (E2), the system performs a rule-based analysis instead, indicates that it is operating in a degraded mode, and suggests retrying when the full system becomes available.

Furthermore, the system supports an extension for Batch Analysis (Ext-1), allowing the analyst to upload multiple emails. The system processes this batch sequentially and provides summary statistics alongside individual results. This use case occurs with a frequency of 20 to 50 times per day per analyst. Finally, strict business rules govern the process: all analyses must complete within 2 seconds, the confidence threshold for automatic blocking is set at 95%, and results must be retained for a configurable period of 90 days.

UC-2: Analyze Web Traffic Anomalies

The use case "Analyze Web Traffic Anomalies" involves the Security Analyst as the primary actor, with the goal of identifying anomalous behavior and potential attacks within web server logs. The process requires specific preconditions: the web analyzer module must be trained on baseline traffic, the log files must be accessible and parseable, and the analyst must be successfully authenticated.

In the main success scenario, the analyst selects the web traffic analysis mode and specifies the input method, which can be a file upload, specific IP address, or a time range. Once the analyst provides the required data, the system validates the input format and parses the log entries using standard Apache or Nginx formats. The system then groups the logs by IP address and, for each IP, extracts over 20 behavioral features. Following feature extraction, the system applies the Isolation Forest algorithm for anomaly detection and simultaneously checks for known attack patterns, including SQL injection, XSS, directory traversal, and brute force attacks. The system calculates an anomaly score and assigns a risk level categorized as LOW, MEDIUM, HIGH, or CRITICAL. It then generates behavioral insights, provides detailed recommendations, and displays the results with a timeline visualization. The scenario concludes when the analyst reviews the findings and initiates a response.

Upon completion, several postconditions are established: suspicious IPs are flagged within the system, analysis results are stored, alerts are generated for any findings classified as HIGH or CRITICAL, and all recommendations are logged.

The system supports three alternative flows. In the Unsupported Log Format (A1) scenario, if the system encounters an unrecognized log format, it attempts a best-effort parsing, displays a warning regarding potential accuracy reduction, and proceeds with the available data. In the Insufficient Data (A2) scenario, if there are fewer than 10 log entries (insufficient for reliable analysis), the system notifies the analyst, requests additional data or a longer time range, and provides a partial analysis with caveats. In the No Anomalies Detected (A3) scenario, if all traffic appears normal, the system confirms baseline behavior and provides a confidence score for the "normal" classification, while still performing checks for known attack patterns.

Two exception flows are defined. In the case of Log Parsing Failure (E1), if the system cannot parse the logs, it displays a detailed error message, suggests format corrections, and provides an example of the expected format. In the case of a Performance Timeout (E2), if processing exceeds the time limit, the system returns partial results, indicates which IPs were analyzed, and offers to continue the remaining processing in the background.

The system includes an extension for Historical Comparison (Ext-1), allowing the analyst to compare current behavior to a historical baseline. In this mode, the system shows trend analysis and highlights deviations from normal patterns. This use case is expected to occur 10 to 30 times per day. The process is governed by strict business rules: the analysis must complete within 2 seconds per IP, the anomaly threshold (contamination) is set at 10%, and a minimum of 10 log entries is required for analysis.

UC-3: Perform Unified Threat Analysis

The use case "Perform Unified Threat Analysis" is executed by the Security Analyst with the primary goal of conducting a comprehensive multi-vector threat analysis that combines email and web data to identify coordinated attacks. The preconditions for this process require that both the email and web modules are operational, relevant data is available for both vectors, and the correlation engine has been initialized.

In the main success scenario, the analyst initiates the unified analysis from the dashboard, and the system presents the unified analysis interface. The analyst provides the email data (subject, body, sender) and the web data (logs or IP address). After the system validates all inputs, it initiates a parallel analysis where the email module performs phishing detection and the web module performs anomaly detection. The system waits for both analyses to complete before invoking the correlation engine with both sets of results. The correlation engine then evaluates specific rules, including the high-risk coordination rule, IP-based campaign rule, pattern similarity rule, and temporal correlation. Following this evaluation, the system calculates a correlation confidence score, determines a risk amplification factor, and computes a unified risk score ranging from 0 to 100. The system assigns an overall threat level, identifies multi-vector attack indicators, and generates a comprehensive threat intelligence report. Finally, the system provides prioritized recommendations (immediate, short-term, and long-term) and displays the unified results with correlation visualization, allowing the analyst to review the comprehensive assessment and export the report if needed.

Upon completion, the postconditions ensure that the unified analysis is complete, correlated threats are identified and stored, a comprehensive report is generated, the threat intelligence database is updated, and alerts are escalated if the threat level is CRITICAL.

The system handles four alternative flows. In the Only Email Data Available (A1) scenario, if no web data is provided, the system performs an email-only analysis, displays a note about incomplete correlation, and provides an option to add web data later. In the Only Web Data Available (A2) scenario, if no email data is provided, the system performs a web-only analysis, displays a note about limited correlation capability, and suggests looking for related email activity. In the No Correlation Found (A3) scenario, if correlation rules do not trigger, the system presents independent analyses, indicates that threats are likely unrelated, and provides separate risk scores for each vector. In the Weak Correlation (A4) scenario, if the correlation confidence is low (20-40%), the system flags the findings as a "possible correlation," provides evidence for both possibilities, and suggests additional investigation.

There are three exception flows. In the event of an Email Module Failure (E1), the system operates with web analysis only, logs the error for the administrator, and notifies the analyst of the degraded capability. Similarly, during a Web Module Failure (E2), the system operates with email analysis only, logs the error, and notifies the analyst of the limited analysis. In the case of a Correlation Engine Error (E3), if the correlation logic fails, the system presents independent results, logs the correlation error, and continues without correlation insights.

The system includes two extensions. Historical Threat Correlation (Ext-1) allows the system to search for similar past threats, providing context from previous incidents and showing the evolution of attack patterns. Automated Response Trigger (Ext-2) enables the system to trigger automated responses for CRITICAL threats, such as blocking an IP, quarantining an email, or alerting the SOC, provided that pre-configured response playbooks are available. This use case occurs 5 to 15 times per day. The process adheres to strict business rules: the unified analysis must complete within 3 seconds, the correlation confidence threshold for high confidence is 70%, the maximum risk amplification is 2.5x, and a CRITICAL threat (risk score >80) triggers an immediate alert.

UC-4: Review AI Explanation

The use case "Review AI Explanation" is performed by the Security Analyst with the goal of understanding why the AI system classified a threat in a particular way. The preconditions for this process require that an analysis (email, web, or unified) has been completed and that explanation data has been generated.

In the main success scenario, the analyst selects a completed analysis from the history, and the system retrieves the analysis results and explanations. The system displays LIME feature contributions, and for each top feature, it presents the feature name in a human-readable format, the contribution weight (positive or negative), the direction of influence, and the specific value in this instance. Additionally, the system displays a SHAP values visualization and presents a human-readable explanation summary, highlighting the top 5 most important factors. The analyst reviews the explanation to gain an understanding of the detection reasoning and can drill down into specific features for more detail.

Upon completion, the postconditions ensure that the analyst is educated on threat characteristics and that the understanding is documented if the analyst provides feedback.

The system supports two alternative flows. In the Request Extended Explanation (A1) scenario, if the analyst wants more detail, the system provides an extended feature list showing the top 20 features, shows additional context and examples, and includes a statistical comparison to the baseline. In the Disagree with Assessment (A2) scenario, if the analyst believes the detection is incorrect, the system accepts feedback through a feedback form, logs the disagreement for model improvement, and allows the analyst to override the classification with justification. This use case occurs daily for learning purposes and as-needed for specific cases.

UC-5: Configure System Settings

The use case "Configure System Settings" involves the System Administrator or SOC Manager as the primary actor, with the goal of adjusting system parameters and thresholds to optimize for organizational needs. The preconditions for this process require that the user has administrative privileges and is authenticated with MFA.

In the main success scenario, the administrator accesses the configuration panel, and the system displays the current configuration. The administrator selects a configuration category from the available options: detection thresholds, correlation rules, alert policies, data retention, or integration settings. After the administrator modifies the desired settings, the system validates the configuration values and displays an impact preview if possible. The administrator confirms the changes, and the system applies the new configuration. Finally, the system logs the configuration change with the user ID and timestamp, and notifies affected users if necessary.

Upon completion, the postconditions ensure that the system operates with the new configuration, the change is logged in the audit trail, and a backup of the previous configuration is maintained.

The system handles two exception flows. In the Invalid Configuration (E1) scenario, if the validation fails, the system rejects the configuration, displays specific errors and requirements, and suggests valid ranges or values. In the Critical Setting Change (E2) scenario, if a change affects critical functionality, the system requires additional confirmation, may require approval from a second administrator, and creates an automatic backup before applying the change. This use case occurs on a weekly to monthly basis.

UC-6: Generate Threat Intelligence Report

The use case "Generate Threat Intelligence Report" is performed by the Security Analyst or SOC Manager with the goal of creating a comprehensive report of threats detected over a specified period. The preconditions for this process require that analysis data is available for the requested timeframe and the user has reporting permissions.

In the main success scenario, the user accesses the reporting section and selects a report type from the available options: Summary report, Detailed incident report, Trend analysis report, or Executive summary.

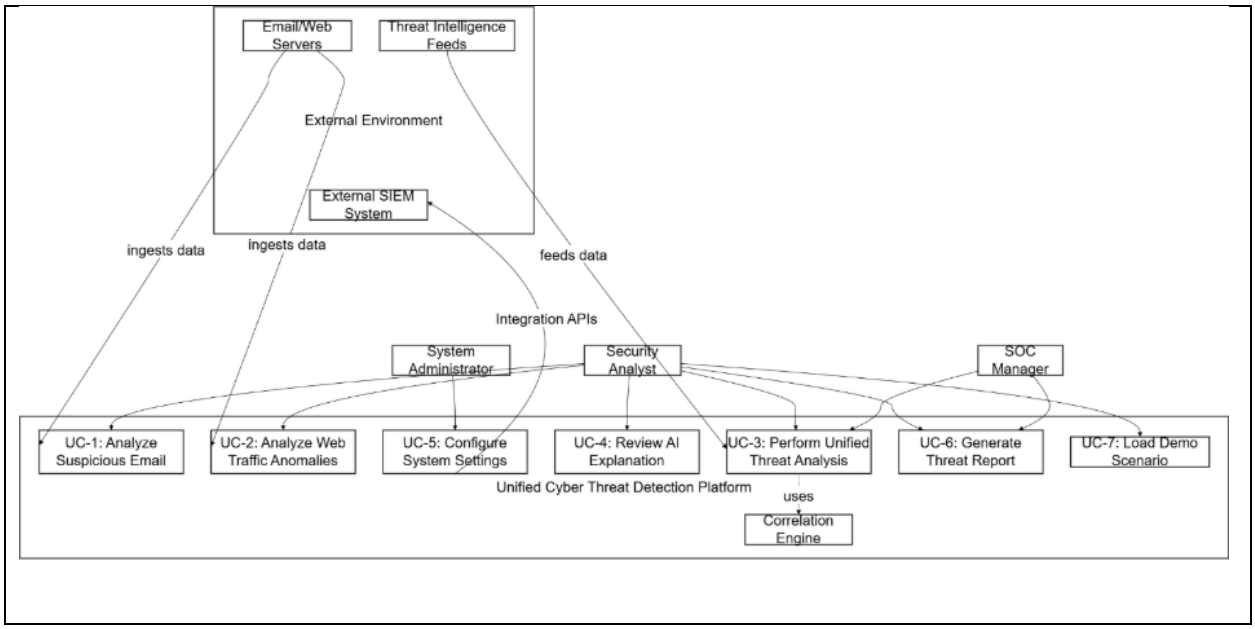
UC-7: Load Demonstration Scenario

The use case "Load Demonstration Scenario" involves the Security Analyst, Evaluator, or Trainer as the primary actor, with the goal of loading a pre-configured demonstration scenario for testing, training, or evaluation purposes. The process requires specific preconditions: the demo data must be available in the system, and the user must have access to demo features.

In the main success scenario, the user accesses the dashboard demo section, and the system displays the available demo scenarios, which include Phishing Email Attack, Brute Force Web Attack, SQL Injection Attempt, and Coordinated Multi-Vector Attack. The user selects the desired scenario, and the system loads the scenario data, populates the input fields with the demo data, and displays the scenario description and context. The user reviews the pre-loaded data and initiates the analysis. The system then performs the full analysis workflow, displays the expected results, and highlights key learning points.

Upon completion, the postconditions ensure that the demo is completed successfully, the results are displayed, and the learning outcomes are achieved.

The system supports one alternative flow. In the Custom Demo Data (A1) scenario, if the user wants to modify the demo during the selection step, the system allows editing of the demo parameters. The user customizes the scenario, and the system runs with the modified data. This use case occurs daily for training purposes and as-needed for demonstrations.



8.2. Object Model (min 3000 characters)

Object model describes the main objects in the system and their relationships with the help of a UML class diagram.

Object-Oriented Design Overview
The system employs a modular, object-oriented architecture that emphasizes separation of concerns, high cohesion, and loose coupling. The design follows SOLID principles and incorporates established design patterns to ensure maintainability, extensibility, and testability.

Core Domain Classes

Class 1: EmailPhishingDetector

The EmailPhishingDetector class is designed to encapsulate all email phishing detection logic, including feature extraction, model training, prediction, and explainability. The class maintains several critical attributes: it stores the trained machine learning model as model (using RandomForestClassifier), handles text vectorization via the vectorizer attribute (TfidfVectorizer), and maintains a list of feature_names representing all 5,015 features . It tracks its training status with the is_trained boolean flag and utilizes specific explainers, including lime_explainer (LimeTabularExplainer) and shap_explainer (TreeExplainer) for generating insights . Additionally, the class holds configuration parameters in a config dictionary and utilizes a helper component, feature_extractor (EmailFeatureExtractor), for extracting features.

The class provides a comprehensive set of methods to handle its responsibilities. The __init__(config: dict) method initializes the detector with the necessary configuration, while train(emails_df: DataFrame, labels: List[int]) is responsible for training the model on labeled data. The core analysis is performed by predict_with_explanation(email_text: str, sender: str, subject: str), which returns a PredictionResult object containing the analysis and explanation. Custom features are handled by extract_email_features(email_text: str, sender: str, subject: str) which returns a dictionary. The class also includes private helper methods such as _count_urgent_words(text: str) and _count_financial_words(text: str) to quantify specific terms, and _analyze_sender(sender: str) to evaluate sender suspiciousness . Furthermore, it supports persistence through save_model(filepath: str) and load_model(filepath: str) methods.

In terms of relationships, the class demonstrates composition by containing the EmailFeatureExtractor in a 1:1 relationship. It has a dependency on LIME and SHAP libraries and is responsible for creating PredictionResult objects in a 1:many relationship. The EmailPhishingDetector is used by the UnifiedThreatPlatform in a 1:1 capacity. The design applies specific patterns: the Facade Pattern is used to provide a simple interface to the complex machine learning pipeline, and the Strategy Pattern is employed so that the feature extraction strategy can be varied.

Class 2: WebLogAnalyzer

The WebLogAnalyzer class is designed to perform comprehensive web log analysis, encompassing parsing, behavioral feature extraction, anomaly detection, and attack pattern recognition . The class maintains a set of critical attributes to support its operations: it utilizes anomaly_detector (an IsolationForest model) for detecting anomalies and scaler (StandardScaler) for feature normalization . It stores the names of over 20 behavioral features in feature_names and tracks the training status with the is_trained boolean flag . Furthermore, the class relies on specialized components stored as attributes: attack_pattern_detector (AttackPatternDetector) for the pattern matching engine and behavioral_analyzer (BehavioralAnalyzer) for the behavioral metrics calculator . It also maintains a config dictionary for parameters such as contamination rate and a normal_patterns dictionary to store baseline behavior statistics .

The class implements a variety of methods to execute its responsibilities. The __init__(config: dict) method initializes the analyzer. Training is handled by train_anomaly_detector(logs_df: DataFrame), which trains the model on normal traffic. The primary analysis method, analyze_ip_with_explanation(ip_logs: List[dict], ip_address: str), analyzes specific IP behavior and returns an AnalysisResult object. Data processing capabilities are provided by parse_log_line(log_line: str) which parses single log entries, extract_behavioral_features(ip_logs: List[dict]) which extracts the features, and detect_attack_patterns(ip_logs: List[dict]) which identifies known attacks . The class also includes private methods for result generation: _calculate_risk_level determines the risk level (LOW/MEDIUM/HIGH/CRITICAL), _generate_behavioral_insights creates human-readable insights, and _generate_recommendations produces action recommendations .

In terms of relationships, the class demonstrates composition by containing instances of AttackPatternDetector (1:1) and BehavioralAnalyzer (1:1). It is responsible for creating AnalysisResult objects in a 1:many relationship and is used by the UnifiedThreatPlatform in a 1:1 capacity .

Class 3: UnifiedThreatPlatform

The UnifiedThreatPlatform class serves the critical purpose of orchestrating the entire threat detection system, effectively coordinating email detection, web analysis, and correlation engines . To fulfill this role, the class maintains several key attributes: email_detector (EmailPhishingDetector) for the email analysis module, web_analyzer (WebLogAnalyzer) for the web log analysis module, and correlation_engine (CorrelationEngine) for cross-platform correlation logic . Additionally, it includes threat_intelligence (ThreatIntelligence) for intelligence aggregation and report_generator (ReportGenerator) as a report creation utility .

The class implements a robust set of methods to manage platform operations. The __init__(email_config: dict, web_config: dict) method initializes the platform with the necessary configurations, while initialize(email_data: Tuple[DataFrame, List[int]], web_logs: DataFrame) is responsible for training all modules and returning a boolean success status . The core analysis functionality is provided by analyze_unified_threat(email_data: dict, web_logs: List[dict], ip_address: str), which performs a comprehensive analysis and returns a UnifiedThreatReport. System health monitoring is handled by get_platform_status(), which returns a dictionary of system status. The class also utilizes private helper methods: _calculate_unified_risk(results: dict) computes the unified risk score as a float, _determine_threat_level(risk_score: float) maps the score to a specific threat level string, and _generate_unified_recommendations(results: dict) creates a dictionary of recommendations .

In terms of relationships, the class demonstrates aggregation by containing the EmailPhishingDetector, WebLogAnalyzer, and CorrelationEngine in 1:1 relationships. It is responsible for creating UnifiedThreatReport objects and is used by the FlaskApplication in a 1:1 capacity . The design explicitly applies the Facade Pattern to simplify interactions with complex subsystems and the Mediator Pattern to coordinate communication between the various modules .

Class 4: CorrelationEngine

The CorrelationEngine class is designed to identify relationships and correlations between email threats and web traffic anomalies . To achieve this, the class maintains specific attributes: correlation_rules, which serves as a list of active CorrelationRule objects; time_window, an integer defining the temporal correlation window in seconds with a default value of 3600; and confidence_threshold, a float representing the minimum confidence required for correlation with a default of 0.5 .

The class implements several methods to handle correlation logic. The __init__() method initializes the engine with default rules, while analyze_correlation(email_result: PredictionResult, web_result: AnalysisResult, metadata: dict) is used to find correlations and returns a CorrelationResult object . The setup_default_rules() method is responsible for initializing standard correlation rules. The class also utilizes private methods for internal processing: _check_rule(rule: CorrelationRule, email_result: PredictionResult, web_result: AnalysisResult, metadata: dict) evaluates a single rule and returns a boolean; _get_web_risk_score(risk_level: str) converts a risk level string to a numeric integer score; and _generate_indicators(email_result: PredictionResult, web_result: AnalysisResult, matched_rules: List[str]) creates a list of CorrelationIndicator objects .

In terms of relationships, the class uses CorrelationRule definitions, creates CorrelationResult objects, and is used by the UnifiedThreatPlatform .

Result and Data Transfer Objects (DTOs)

Class 5: PredictionResult

The PredictionResult class serves the specific purpose of encapsulating the results derived from email phishing detection . To effectively store this data, the class maintains a comprehensive set of attributes: it holds the classification result as a string in prediction ('Safe' or 'Phishing'), the confidence percentage as a float in confidence (0-100), and the specific probabilities for both classes in phishing_probability and safe_probability . Furthermore, it stores identified risk indicators in risk_factors, LIME feature contributions in lime_explanation, and the most important features in top_features . The class also records the exact time of analysis in the timestamp attribute. Functionally, the class provides methods for serialization, including to_dict() to convert the object to a dictionary and to_json() to convert it to a JSON string .

Class 6: AnalysisResult

The AnalysisResult class is designed to encapsulate the results obtained from web log analysis . To comprehensively store this analytical data, the class maintains a specific set of attributes: it records the analyzed IP address as a string in ip_address and stores the anomaly detection score as a float in anomaly_score . It utilizes a boolean attribute, is_anomaly, for binary anomaly classification and a string attribute, risk_level, to categorize the risk as LOW, MEDIUM, HIGH, or CRITICAL . Furthermore, the class stores lists of strings for attack_patterns (detected attack types), behavioral_insights (behavioral descriptions), and recommendations (suggested actions) . The exact time of the analysis is recorded in the timestamp attribute. In terms of functionality, the class includes methods to facilitate data handling: to_dict() converts the object into a dictionary, while to_json() converts it into a JSON string .

Class 7: UnifiedThreatReport

The UnifiedThreatReport class serves the purpose of acting as a comprehensive report that combines all analysis results . To achieve this, the class maintains a specific set of attributes: it records the report generation time in timestamp. It aggregates results from other components by storing email_analysis (as a PredictionResult object), web_analysis (as an AnalysisResult object), and correlation_analysis (as a CorrelationResult object) . It also stores the overall assessment metrics, including the unified_risk_score as a float ranging from 0 to 100 and the threat_level as a string indicating overall severity (LOW, MEDIUM, HIGH, or CRITICAL) . Furthermore, it holds recommendations as a dictionary containing categorized recommendations (immediate, short_term, and long_term) and threat_intelligence as a dictionary of aggregated threat intelligence . Functionally, the class provides methods for data representation and output: to_dict() converts the report to a dictionary, to_json() converts it to a JSON string, and generate_summary() creates a text summary of the report .

Class 8: CorrelationResult

The CorrelationResult class is designed to encapsulate the results derived from correlation analysis . To effectively store these findings, the class maintains a specific set of attributes: it records the correlation confidence as a float ranging from 0 to 1 in confidence_score and stores the risk multiplier factor as a float in risk_amplification . Furthermore, it maintains a list of found correlations in indicators (defined as CorrelationIndicator objects) and a list of triggered rule names as strings in matched_rules . The exact time of the analysis is recorded in the timestamp attribute.

Utility and Helper Classes

Class 9: EmailFeatureExtractor

The EmailFeatureExtractor class is designed for the purpose of performing specialized feature extraction from email content . To support this function, the class maintains an attribute named suspicious_keywords, which is a dictionary containing categorized keyword lists . The class implements specific methods to execute its tasks: extract_url_features(text: str) returns a dictionary of URL-based features, extract_text_features(text: str) provides a dictionary of text statistics, and extract_keyword_features(text: str) returns a dictionary based on keyword counting . Additionally, it includes a private helper method, _is_suspicious_url(url: str), which returns a boolean indicating the result of a URL reputation check.

Class 10: AttackPatternDetector

The AttackPatternDetector class is designed for the purpose of signature-based attack pattern detection . To facilitate this, it maintains an attribute named patterns, which is a dictionary storing attack signature patterns organized by type . The class implements specific methods to perform detection tasks: detect_pattern(text: str, pattern_type: str) returns a boolean indicating the result of a single pattern check, while detect_all_patterns(text: str) returns a list of strings representing the results of a comprehensive scan .

Class 11: BehavioralAnalyzer

The BehavioralAnalyzer class is designed for the purpose of performing behavioral pattern analysis from web logs . To achieve this, the class implements specific methods: analyze_request_timing(timestamps: List[datetime]) returns a dictionary representing temporal analysis, analyze_user_agent(user_agents: List[str]) returns a dictionary of user agent patterns, and calculate_diversity_metrics(paths: List[str]) returns a dictionary representing path diversity .

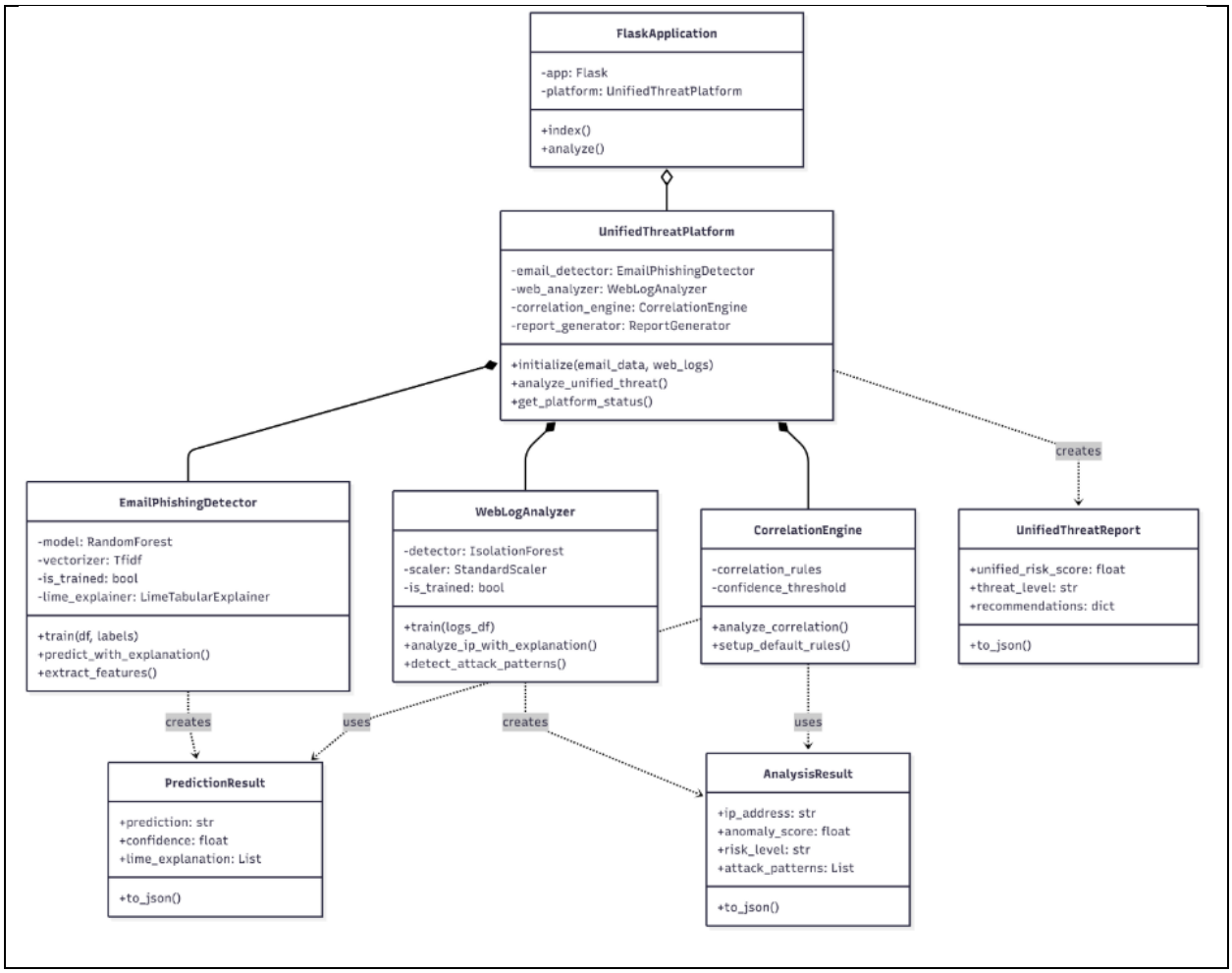
Class 12: ReportGenerator

The ReportGenerator class is designed for the purpose of generating reports in multiple formats . To fulfill this function, the class implements specific methods: generate_text_report(analysis_results: UnifiedThreatReport) creates a report in plain text format and returns a string , generate_json_report(analysis_results: UnifiedThreatReport) creates a report in JSON format and returns a string , and generate_html_report(analysis_results: UnifiedThreatReport) creates a report in HTML format and returns a string. Additionally, the class includes save_report(content: str, filepath: str, format: str), which returns a boolean indicating whether the report was successfully persisted.

Web Application Classes

Class 13: FlaskApplication

The FlaskApplication class is designed for the purpose of handling web application requests and routing . To facilitate these operations, the class maintains specific attributes: it holds the Flask application instance in app (Flask) and maintains a reference to the core platform in platform (UnifiedThreatPlatform) . The class implements a set of methods to manage web interactions: index() is used to render the main dashboard and returns a string, while analyze() handles the analysis API endpoint and returns a Response object . Furthermore, load_demo(demo_type: str) is responsible for loading demonstration data and returning a Response, and get_stats() returns platform statistics as a Response . Finally, the class includes initialize_platform(), which performs the task of initializing the system on startup.



9. SYSTEM DESIGN (final)

9.1. Software Architecture (min 2000 characters)

Describe the decomposition of your system into subsystems. Use a UML component or package diagram to show your SW architecture.

9.2. Hardware Architecture (final) (if exists)

9.3. Persistent Data Management (final) (if exists)

Persistent data management describes the persistent data stored by the system and the data management infrastructure required for it. This section typically includes the description of data schemes, the selection of a database, and the description of the encapsulation of the database.

10. SYSTEM TEST DESIGN (final) (min 5000 characters)

Design a test to evaluate your system. The test design depends on the project topic (Some possibilities: user evaluation, surveys, performance tests, unit tests, etc.)

11. DISCUSSION OF THE RESULTS (final) (min 3000 characters)

Summarize your study. Discuss the quantitative results obtained by the test you performed in Section 9.

12. REFERENCES

[1] SlashNext, "The State of Phishing 2024: A Mid-Year Assessment," SlashNext Threat Intelligence Report, 2024. [Online]. Available: https://slashnext.com/
[2] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 1135–1144.
[3] A. Bergholz, J. De Beer, S. Glahn, M. F. Moens, G. Paaß, and S. Strobel, "New filtering approaches for phishing email," Journal of Computer Security, vol. 18, no. 1, pp. 7–35, 2010.
[4] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in Proceedings of the Anti-Phishing Working Groups 2nd Annual eCrime Researchers Summit, Pittsburgh, PA, USA, 2007, pp. 60–69.
[5] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in Proceedings of the 16th International Conference on World Wide Web, Banff, Alberta, Canada, 2007, pp. 649–656.
[6] M. Hiransha, N. A. Unnithan, R. Vinayakumar, K. P. Soman, and A. Verma, "Deep learning based phishing E-mail detection," in Proceedings of the 1st International Conference on Advanced Informatics for Computing Research, Shimla, India, 2018.
[7] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: learning to detect malicious web sites from suspicious URLs," in Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 2009, pp. 1245–1254.
[8] M. Roesch, "Snort: Lightweight intrusion detection for networks," in LISA '99: Proceedings of the 13th Systems Administration Conference, Seattle, WA, USA, 1999, pp. 229–238.
[9] T. Lane and C. E. Brodley, "Temporal sequence learning and data reduction for anomaly detection," ACM Transactions on Information and System Security (TISSEC), vol. 2, no. 3, pp. 295–331, 1999.
[10] C. Kruegel and G. Vigna, "Anomaly detection of web-based attacks," in Proceedings of the 10th ACM Conference on Computer and Communications Security, Washington, DC, USA, 2003, pp. 251–261.
[11] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation forest," in 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 413–422.
[12] W. Zhang, Y. Xie, J. Wang, Y. Luo, and K. P. Chow, "Effective web application firewall rule generation using isolation forest," in International Conference on Information Security Practice and Experience, Springer, 2019, pp. 287–301.
[13] R. Alshammari and A. N. Zincir-Heywood, "Machine learning based encrypted traffic classification: Identifying SSH and Skype," in 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 2009, pp. 1–8.
[14] D. L. Marino, C. S. Wickramasinghe, and M. Manic, "An adversarial approach for explainable AI in intrusion detection systems," in IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 2018, pp. 3237–3243.
[15] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 2017, pp. 4765-4774.
[16] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, "Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods," in Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, 2020, pp. 180–186.
[17] A. Kuppa and N. A. Le-Khac, "Black box attacks on explainable artificial intelligence (XAI): A survey," arXiv preprint arXiv:2001.01211, 2020.
[18] U. Bhatt et al., "Explainable machine learning in deployment," in Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, 2020, pp. 648-657.
[19] P. Sadighian, J. M. Fernandez, A. Lemay, and S. T. Zargar, "ONTIDS: A highly flexible context-aware and ontology-based alert correlation framework," in Proceedings of the 6th International Symposium on Foundations and Practice of Security, La Rochelle, France, 2013, pp. 161-177.
[20] I. Friedberg, F. Skopik, G. Settanni, and R. Fiedler, "Combating advanced persistent threats: From network event correlation to incident detection," Computers & Security, vol. 48, pp. 35-53, 2015.

[21] I. Ghafir et al., "A hidden Markov model for detecting advanced persistent threats," in 2018 17th IEEE International Conference on Trust, Security And Privacy In Computing And Communications, New York, NY, USA, 2018.

13. CHOOSE INTERDISCIPLINARY DOMAIN OF YOUR STUDY

Industry/Production

Selected Interdisciplinary Domain: Computer Science + Cybersecurity + Artificial Intelligence + Human-Computer Interaction
This graduation project operates at the convergence of multiple academic and professional disciplines, demonstrating the increasingly interdisciplinary nature of modern technological solutions. The successful development and deployment of the Unified Cyber Threat Detection Platform requires deep integration of knowledge from diverse fields, each contributing essential perspectives and methodologies.

Primary Disciplines

1. Computer Science
Contribution to Project: Computer Science provides the foundational framework for system development, including software engineering principles, algorithm design, data structures, system architecture, and computational thinking methodologies that underpin the entire platform.

Specific Applications:

- **Software Architecture:** Modular design with clear separation of concerns, implementing design patterns (Facade, Strategy, Factory, Mediator) to ensure maintainability and extensibility
- **Algorithm Design:** Efficient implementation of machine learning pipelines, optimization of feature extraction algorithms, and performance tuning for real-time analysis
- **Data Structures:** Selection of appropriate data structures for storing features, managing analysis results, and maintaining system state
- **Software Engineering Practices:** Version control with Git, unit testing with pytest, continuous integration principles, code quality assurance with linting tools
- **API Development:** RESTful API design following OpenAPI specifications for dashboard communication
- **Database Management:** Efficient data storage and retrieval strategies for analysis results and system logs
- **Performance Optimization:** Profiling and optimization techniques to achieve sub-2-second response times

Key Concepts Utilized:

- Object-oriented programming and SOLID principles
- Design patterns for scalable architecture
- Software testing methodologies (unit, integration, system testing)
- Performance analysis and optimization
- Documentation and code maintainability
- Agile development methodology

Why Essential: Without strong computer science foundations, the sophisticated machine learning models and security analyses would remain theoretical concepts unable to be deployed in practical systems. Proper software engineering ensures the platform is reliable, maintainable, and scalable for real-world use.

2. Cybersecurity
Contribution to Project: Cybersecurity provides essential domain knowledge about threat landscapes, attack methodologies, defense mechanisms, security operations workflows, incident response procedures, and compliance requirements that shape every aspect of the platform's design and functionality.

Specific Applications:

- **Threat Modeling:** Understanding adversary tactics, techniques, and procedures (TTPs) mapped to MITRE ATT&CK framework
- **Attack Pattern Recognition:** Identification of phishing indicators (urgent language, suspicious URLs, sender spoofing) and web attacks (SQL injection, XSS, brute force, directory traversal)
- **Security Operations:** Alignment with SOC workflows, alert prioritization, incident response integration
- **Risk Assessment:** Multi-factor risk scoring methodology, threat level classification
- **Vulnerability Analysis:** Understanding system vulnerabilities that attackers exploit
- **Defense Strategy:** Implementing defense-in-depth principles through multi-layered detection
- **Compliance Requirements:** Ensuring system design meets security standards (ISO 27001, NIST CSF)

Key Concepts Utilized:

- MITRE ATT&CK framework for attack categorization
- Indicators of Compromise (IOCs)
- Security Information and Event Management (SIEM) concepts
- Threat intelligence and threat hunting methodologies
- Security operations center (SOC) workflows
- Incident response procedures
- Security standards and compliance frameworks

Why Essential: Domain expertise in cybersecurity ensures that the platform detects meaningful threats rather than arbitrary patterns. Understanding how attacks actually work and how security teams operate enables the development of a practical tool that integrates into existing security workflows and provides actionable intelligence.

3. Artificial Intelligence & Machine Learning

Contribution to Project: AI and Machine Learning provide intelligent decision-making capabilities, pattern recognition algorithms, predictive modeling techniques, and explainability methods that enable automated threat detection with human-understandable reasoning.

Specific Applications:

- **Supervised Learning:** Random Forest classification for email phishing detection with labeled training data
- **Unsupervised Learning:** Isolation Forest for web traffic anomaly detection without requiring labeled attack examples
- **Feature Engineering:** Extraction and selection of meaningful features from emails (5,015 features) and web logs (20+ behavioral features)
- **Natural Language Processing:** TF-IDF vectorization for email text analysis, keyword extraction, linguistic feature analysis
- **Ensemble Methods:** Combining multiple decision trees in Random Forest for robust predictions
- **Model Optimization:** Hyperparameter tuning using GridSearchCV and RandomSearchCV, cross-validation for reliable performance estimation
- **Explainable AI:** LIME for local interpretations of individual predictions, SHAP for feature importance analysis
- **Performance Metrics:** Accuracy, precision, recall, F1-score, confusion matrix analysis, ROC-AUC curves

Key Concepts Utilized:

- Classification and anomaly detection algorithms
- Feature extraction and dimensionality reduction
- Model training, validation, and testing procedures
- Overfitting prevention and regularization
- Bias-variance tradeoff
- Interpretability vs. accuracy tradeoff
- Transfer learning concepts

Why Essential: Machine learning enables the platform to detect previously unseen threats and adapt to evolving attack patterns without manual rule updates. Explainable AI bridges the gap between powerful but opaque AI models and security analysts who need to understand and trust automated decisions.

4. Human-Computer Interaction (HCI)
Contribution to Project: HCI provides principles for designing intuitive interfaces, effective information visualization, cognitive load management, and user-centered design that ensure security analysts can effectively use the platform's capabilities.

Specific Applications:

- **User Interface Design:** Interactive web dashboard with clear visual hierarchy, intuitive navigation, responsive design
- **Information Visualization:** Charts and graphs for risk scores, timeline visualizations for attack patterns, heatmaps for correlation analysis
- **Explanation Presentation:** Human-readable formatting of LIME and SHAP explanations, progressive disclosure of complex information
- **Cognitive Load Management:** Prioritization of critical information, use of color coding for threat levels, organized layout reducing mental effort
- **Feedback Mechanisms:** Real-time system responses, confirmation messages, error handling with helpful guidance
- **Accessibility:** Keyboard navigation support, color-blind friendly palettes, screen reader compatibility considerations
- **User Testing:** Iterative design based on usability principles and security analyst workflows

Key Concepts Utilized:

- User experience (UX) design principles
- Visual hierarchy and information architecture
- Gestalt principles of perception
- Cognitive psychology of decision-making
- Interaction design patterns
- Responsive web design
- Accessibility standards (WCAG)

Why Essential: Even the most accurate threat detection system is useless if security analysts cannot understand its findings or efficiently act on its recommendations. Good HCI design ensures that complex AI-driven insights are accessible to human decision-makers, enabling effective human-AI collaboration.

Secondary Disciplines

5. Data Science & Analytics
Contribution: Statistical analysis methods, data preprocessing pipelines, exploratory data analysis techniques, and scientific methodology for validating results.
Applications: Feature engineering, statistical hypothesis testing, data cleaning and normalization, correlation analysis, trend identification, performance benchmarking.

6. Natural Language Processing (NLP)
Contribution: Specialized techniques for analyzing and understanding text data, linguistic feature extraction, semantic analysis.
Applications: Email content analysis, TF-IDF text vectorization, keyword extraction and categorization, language pattern recognition, text classification methodologies.

7. Network Security
Contribution: Understanding of network protocols, traffic analysis techniques, distributed system security, and network-level threat detection.
Applications: Web log analysis, HTTP protocol understanding, IP address reputation analysis, traffic pattern recognition, session analysis.

8. Law & Regulatory Compliance
Contribution: Legal frameworks governing data protection, privacy rights, cybersecurity regulations, and organizational liability.
Applications: GDPR compliance design, KVKK (Turkish data protection law) adherence, privacy-by-design implementation, data retention policies, audit trail requirements.

Why Interdisciplinarity is Essential for This Project
Synergistic Integration
The effectiveness of this platform fundamentally depends on the seamless integration of knowledge from all these disciplines. Each discipline alone would be insufficient:

Computer Science Alone:

- Could build a well-engineered system but wouldn't know what to detect
- Would lack domain knowledge to identify meaningful threat indicators
- Might optimize for wrong metrics or implement irrelevant features

Cybersecurity Alone:

- Would understand threats but lack tools for automated detection at scale
- Could not leverage modern AI capabilities for pattern recognition
- Would be limited to manual analysis or simple rule-based systems

AI/ML Alone:

- Might achieve high accuracy on meaningless or inappropriate features
- Would lack context for practical deployment in security operations
- Could produce systems that work in lab but fail in real-world scenarios

HCI Alone:

- Could design beautiful interfaces but without meaningful underlying functionality
- Would lack the technical capability to implement complex detection systems
- Could not ensure that visualizations accurately represent threat intelligence

Integrated Approach: The project's strength comes from combining:

- **Computer Science + AI:** Robust, efficient implementation of ML algorithms
- **Cybersecurity + AI:** Meaningful feature selection guided by domain expertise
- **AI + HCI:** Explainable, interpretable AI through careful presentation design
- **All Disciplines + Law:** Compliant system design respecting privacy and legal requirements

Real-World Problem Solving

Cybersecurity threats are inherently interdisciplinary challenges that exploit multiple dimensions:
Technical Vulnerabilities: Require computer science expertise to address
Human Factors: Require HCI and psychology knowledge to understand social engineering
Sophisticated Attacks: Require AI/ML to detect subtle patterns
Operational Context: Require security expertise to understand organizational impact
Legal Implications: Require compliance knowledge to handle data appropriately
Defending against such multifaceted threats demands an equally multifaceted solution that integrates expertise from all relevant disciplines.

Innovation Through Cross-Pollination

The novel contributions of this project emerge specifically from combining concepts across traditional disciplinary boundaries:

1. **Email + Web Detection Integration:** Cybersecurity insight that coordinated attacks use multiple vectors + Computer Science capability to build integrated systems
2. **Explainable Security AI:** AI advancement in interpretability + HCI principles for effective presentation + Cybersecurity need for actionable intelligence
3. **Practical ML Deployment:** Computer Science engineering practices + AI model development + Cybersecurity operational requirements
4. **User-Centered Security:** HCI design principles + Cybersecurity workflow understanding + AI capability integration

These innovations would not emerge from work confined within a single discipline.

Preparation for Modern Cybersecurity Careers

The cybersecurity job market increasingly demands professionals with interdisciplinary expertise:

Required Modern Skills:

- **Build and deploy systems** (Computer Science)
- **Understand attack methodologies** (Cybersecurity)
- **Leverage AI/ML effectively** (Data Science/AI)
- **Design usable security tools** (HCI)
- **Navigate legal requirements** (Compliance)

This Project Provides: Hands-on experience in all these areas, preparing for roles such as:

- Security Data Scientist
- ML Security Engineer
- SOC Analyst/Engineer
- Threat Intelligence Analyst
- Security Product Developer

Learning Outcomes from Interdisciplinary Approach
Through this interdisciplinary project, the following comprehensive learning outcomes have been achieved:

1. Technical Breadth:

- Competence in software engineering, machine learning, and cybersecurity
- Ability to work with diverse technologies and tools
- Understanding of how different technical domains interconnect

2. Integration Skills:

- Capability to synthesize concepts from different fields
- Ability to identify relevant knowledge from each discipline
- Skill in combining approaches to solve complex problems

3. Holistic Problem-Solving:

- Approaching challenges from multiple perspectives
- Recognizing that technical solutions must consider human factors
- Understanding that perfect technical systems can fail due to usability or compliance issues

4. Communication Across Disciplines:

- Explaining AI concepts to security professionals
- Describing security threats to software engineers
- Communicating technical capabilities to non-technical stakeholders

5. Critical Thinking:

- Evaluating trade-offs across different dimensions (accuracy vs. interpretability, security vs. usability, innovation vs. compliance)
- Recognizing limitations of single-discipline approaches
- Understanding when to prioritize different disciplinary perspectives

6. Ethical and Legal Awareness:

- Understanding societal implications of security systems
- Recognizing privacy concerns in data processing
- Appreciating regulatory requirements and their rationale

7. Practical Implementation Skills:

- Building systems that work in real environments, not just theoretical contexts
- Balancing ideal solutions with practical constraints
- Iterating designs based on feedback from multiple stakeholder perspectives

Conclusion
This project exemplifies modern engineering education's shift toward interdisciplinary problem-solving. The complex challenges of cybersecurity in the AI age cannot be addressed by specialists working in isolation but require professionals who can integrate knowledge across Computer Science, Cybersecurity, Artificial Intelligence, and Human-Computer Interaction.
The Unified Cyber Threat Detection Platform demonstrates that the most innovative and effective solutions emerge at the intersections of disciplines, where different perspectives and methodologies combine to address multifaceted real-world challenges.

14. CHOOSE SUSTAINABILITY DEVELOPMENT GOAL OF YOUR PROJECT

9. Sanayi, yenilikçilik ve altyapı

Selected SDG: Goal 9 - Industry, Innovation and Infrastructure
"Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation"

Specific Target: 9.c
"Significantly increase access to information and communications technology and strive to provide universal and affordable access to the Internet in least developed countries by 2020"

Note: While the 2020 target date has passed, the underlying objective of ensuring safe, secure, and accessible information and communications technology (ICT) infrastructure remains critical and ongoing. The target continues to guide efforts to bridge the digital divide and ensure that all populations can safely participate in the digital economy.

Comprehensive Justification and Alignment
Primary Contribution: Strengthening Digital Infrastructure Security
This project directly addresses SDG 9's core objective of building resilient infrastructure by enhancing the security and reliability of digital systems, which constitute critical infrastructure in the modern world.

How the Project Contributes:
1. Making Advanced Security Accessible
The Challenge:

- Advanced cybersecurity solutions typically require significant financial investment (tens to hundreds of thousands of dollars annually)
- Enterprise-grade security tools demand specialized expertise that small organizations cannot afford
- Developing countries and small-to-medium enterprises (SMEs) face disproportionate cyber risks due to limited security capabilities
- This creates a dangerous digital divide where only wealthy organizations can protect themselves adequately

Project's Solution:

- **Open-Source Approach:** Platform design enables free or low-cost deployment, dramatically reducing financial barriers
- **Explainable AI:** Makes sophisticated AI-driven security accessible to analysts with varying expertise levels through clear explanations of detection reasoning
- **Educational Value:** XAI components serve a training function, helping users learn about threats while using the system
- **Efficient Architecture:** Optimized algorithms work effectively without requiring expensive high-performance hardware
- **Scalable Design:** Can be deployed at various scales from small businesses to large enterprises

Impact on SDG 9.c: By democratizing access to advanced threat detection, the project supports universal access to safe ICT infrastructure. When more organizations can afford and effectively use cybersecurity tools, the entire digital ecosystem becomes more trustworthy, encouraging broader Internet adoption and use - particularly in developing regions where cost is a primary barrier.

2. Building Trust in Digital Infrastructure
The Problem:

- Frequent data breaches and cyberattacks erode public trust in digital services
- Fear of cyber threats discourages individuals and organizations from embracing digital transformation
- This trust deficit is particularly acute in developing countries where digital literacy is lower and past experiences with cyber incidents are limited

Project's Contribution:

- **Proactive Defense:** Early detection and prevention of attacks before damage occurs
- **Transparency:** Explainable AI builds user trust by showing why threats are identified
- **Reliability:** High accuracy rates (>90% for email, >85% for web) with low false positives (<5%) demonstrate dependable protection
- **Multi-Vector Detection:** Comprehensive security covering both email and web threats provides confidence in overall protection

Impact on SDG 9: Trusted digital infrastructure is essential for:

- **E-commerce adoption** (economic growth)
- **Digital education platforms** (SDG 4)
- **Telemedicine services** (SDG 3)
- **E-government services** (SDG 16)
- **Digital financial inclusion** (SDG 8)

When people trust that digital systems are secure, they are more willing to engage with technology, driving the broader digital transformation necessary for sustainable development.

3. Fostering Innovation in Cybersecurity
Innovation Contributions:

Technical Innovation:

- **Novel Integration Architecture:** First platform to unify email phishing detection and web log analysis with intelligent cross-platform correlation
- **Practical XAI Implementation:** Moving explainable AI from research concept to deployable security tool
- **Efficient Correlation Engine:** Rule-based system with machine learning that identifies coordinated multi-vector attacks
- **Open Architecture:** Modular design enables community contributions and extensions to additional threat vectors

Methodological Innovation:

- **Human-AI Collaboration Model:** System designed for humans and AI to work together effectively, not for AI to replace humans
- **Graduated Response Framework:** Risk scoring and threat leveling that guides appropriate response intensity
- **Learning-Oriented Security:** Explanations that educate users while protecting them

Alignment with SDG 9.5: *"Enhance scientific research, upgrade the technological capabilities of industrial sectors in all countries, in particular developing countries"*
The project demonstrates that:

- Graduate-level research can produce practical innovations addressing real-world needs
- Open-source approaches enable technology transfer to developing countries
- Documented methodologies allow replication and adaptation by researchers globally
- The platform serves as a foundation for future security innovation by the broader community

4. Enabling Inclusive and Sustainable Industrialization
Supporting SME Digital Transformation:
Small and Medium Enterprises (SMEs) constitute:

- **99.8% of all Turkish enterprises**
- **73% of employment in Turkey**
- **Similar proportions in most developing economies**

Yet SMEs face disproportionate cybersecurity challenges:

- Cannot afford dedicated security staff
- Lack expertise to manage complex security tools
- Are increasingly targeted by cybercriminals who view them as soft targets
- May avoid digitalization entirely due to security fears

Project's Impact on SMEs:

- **Affordable Protection:** Low-cost deployment enables SME security budgets to cover advanced capabilities
- **Low Technical Barrier:** Explainable interface allows non-expert staff to operate the system effectively
- **Confidence to Digitalize:** Adequate security enables SMEs to adopt digital business models
- **Competitive Parity:** Reduces security advantage of large enterprises, leveling competitive playing field

Connection to SDG 9.2: *"Promote inclusive and sustainable industrialization and, by 2030, significantly raise industry's share of employment and GDP"*
When SMEs can safely digitalize, they:

- Access new markets through e-commerce
- Improve operational efficiency through digital tools
- Create jobs in the digital economy
- Contribute more significantly to GDP growth

5. Strengthening Resilience of Critical Infrastructure

Digital Infrastructure as Foundation: Modern critical infrastructure increasingly depends on interconnected digital systems:

- Power grids with smart grid technologies
- Water treatment facilities with automated controls
- Transportation networks with intelligent traffic management
- Healthcare systems with electronic health records
- Financial systems with digital payment networks
- Communication infrastructure enabling internet connectivity

Cybersecurity as Infrastructure Protection: Attacks on digital systems can cause:

- Physical damage (industrial control system attacks)
- Service disruption (ransomware shutting down hospitals)
- Economic harm (financial system breaches)
- Loss of life (compromised healthcare systems)

Project Contribution to Infrastructure Resilience:

- **Early Warning:** Detection of threats in their initial stages before escalation
- **Coordinated Attack Detection:** Identification of sophisticated multi-phase attacks that single-vector systems miss
- **Rapid Response Enablement:** Real-time analysis supports quick incident response
- **Continuous Protection:** 24/7 automated monitoring scales beyond human capacity

Alignment with SDG 9.1: *"Develop quality, reliable, sustainable and resilient infrastructure, including regional and transborder infrastructure, to support economic development and human well-being"*
Protected digital infrastructure is:

- **More reliable:** Continues operating despite attack attempts
- **More resilient:** Recovers quickly from security incidents
- **More sustainable:** Reduces costly breaches and rebuilding
- **Higher quality:** Meets modern security standards

Indirect Contributions to Related SDGs

While primarily aligned with SDG 9, this project contributes to several interconnected Sustainable Development Goals:

SDG 3: Good Health and Well-Being
Healthcare Cybersecurity:

- Protects patient data (electronic health records) from breaches
- Ensures continuity of telemedicine services
- Prevents ransomware attacks that shut down hospitals
- Secures medical IoT devices and systems

Impact: Healthcare organizations can safely digitalize, improving service delivery while protecting patient privacy and safety.

SDG 4: Quality Education
E-Learning Security:

- Protects e-learning platforms from disruption
- Secures student data and educational records
- Enables safe online collaboration tools
- Protects research data at universities

Impact: Educational institutions can confidently adopt digital learning technologies, expanding access to education.

SDG 8: Decent Work and Economic Growth
Business Protection:

- Prevents cyber attacks that disrupt business operations
- Protects intellectual property from theft
- Enables secure e-commerce
- Reduces economic losses from cybercrime (estimated at $6 trillion globally in 2021)

Job Creation:

- Contributes to growing cybersecurity employment sector
- Enables digital businesses to operate safely
- Supports remote work security

Impact: Businesses can operate safely in digital economy, creating jobs and contributing to economic growth.

SDG 16: Peace, Justice and Strong Institutions
Institutional Security:

- Protects government digital services from attack
- Secures electoral systems from interference
- Enables safe digital governance
- Protects judicial and law enforcement systems

Impact: Strengthens democratic institutions through secure digital infrastructure.

SDG 17: Partnerships for the Goals
Knowledge Sharing:

- Open-source approach enables global collaboration
- Documented methodologies support knowledge transfer
- Academic publication contributes to research community
- Platform serves as foundation for further research partnerships

Impact: Facilitates international cooperation on cybersecurity challenges.

Regional Context: Turkey and Developing Countries
Specific Relevance to Turkey
1. Digital Transformation Initiative: Turkey is undergoing comprehensive digital transformation:

- **e-Government:** Expansion of digital government services (e-Devlet)
- **Smart Cities:** Development of intelligent urban infrastructure
- **Industry 4.0:** Manufacturing digitalization initiatives
- **Digital Health:** National Health Information System (NHIS) implementation

Security Imperative: Success of these initiatives fundamentally depends on cybersecurity. The platform supports national digitalization by providing accessible security capabilities.

2. SME-Driven Economy:

- 99.8% of Turkish enterprises are SMEs
- 73% of employment comes from SMEs
- SMEs contribute 55% of GDP

Platform Impact: Enables Turkish SMEs to securely participate in digital economy, supporting national economic development.

3. Cybersecurity Capacity Building:

- Growing need for cybersecurity professionals (shortage of ~10,000 in Turkey)
- Development of national cybersecurity ecosystem
- Reduction of dependence on foreign security solutions

Platform Contribution:

- Training tool for security professionals through XAI features
- Foundation for Turkish cybersecurity product development
- Demonstration of domestic innovation capability

4. Regional Technology Hub Potential:

- Geographic position between Europe, Asia, and Middle East
- Growing technology sector
- Increasing software export market

Strategic Value: Advanced cybersecurity capabilities support Turkey's positioning as a regional technology hub.

Broader Impact in Developing Regions

1. Technology Transfer:

- Open-source nature enables free adoption globally
- Documented implementation allows local adaptation
- No licensing restrictions prevent developing country use

2. Capacity Building:

- XAI features serve as training tool for local security professionals
- Platform operation builds technical skills
- Reduced need for expensive external consultants

3. Resource Efficiency:

- Optimized algorithms work with limited computational resources
- Does not require latest generation hardware
- Can operate in bandwidth-constrained environments

4. Localization Potential:

- Modular architecture supports adaptation to local contexts
- Can be trained on local threat data
- Interface can be translated to local languages

5. Economic Development:

- Enables small businesses in developing countries to operate online safely
- Reduces cyber-risk barrier to digital economy participation
- Supports entrepreneurship in digital sectors

Sustainability Dimensions

Environmental Sustainability

1. Energy Efficiency:

- **Algorithm Selection:** Random Forest and Isolation Forest algorithms provide excellent performance with moderate computational requirements
- **Comparison:** Deep learning alternatives require 10-100x more computational power for similar accuracy
- **Impact:** Reduced energy consumption per analysis, lower carbon footprint from data center operations

2. Extended Hardware Lifecycle:

- **Efficient Processing:** System runs effectively on 3-5 year old hardware
- **Reduced Upgrades:** Organizations don't need constant hardware refresh cycles
- **Environmental Benefit:** Reduced electronic waste, decreased demand for new hardware manufacturing

3. Green Security Operations:

- **Automated Detection:** Reduces need for always-on human monitoring (energy for offices, commuting)
- **Remote Operation:** Platform supports distributed security operations, reducing physical infrastructure needs

Economic Sustainability

1. Open Source Model:

- **No Licensing Costs:** Organizations save thousands to hundreds of thousands annually
- **Community Development:** Distributed maintenance burden reduces long-term costs
- **Vendor Independence:** Organizations not locked into expensive proprietary solutions

2. Skill Transfer:

- **Reduced Consulting Dependence:** XAI features enable internal capability building
- **Training Value:** Platform serves as learning tool, reducing training costs
- **Local Expertise:** Builds domestic cybersecurity capacity

3. Scalable Economics:

- **Elastic Deployment:** Organizations pay only for resources they use
- **Incremental Adoption:** Can start small and scale as needs/budgets grow
- **Sustainable Business Models:** Supports various revenue models (SaaS, managed service, enterprise support)

Social Sustainability

1. Inclusive Design:

- **Varying Expertise Levels:** Useful for both novice and expert security analysts
- **Educational Component:** Helps users learn, not just detect threats
- **Accessible Interface:** Follows accessibility standards for users with disabilities

2. Privacy Protection:

- **Human Rights:** Supports digital privacy as fundamental right
- **Trust Building:** Transparent operations build public confidence
- **Ethical AI:** Explainability addresses AI ethics concerns

3. Knowledge Commons:

- **Shared Benefits:** Security improvements benefit entire community
- **Collaborative Security:** Open approach enables collective defense
- **Global Public Good:** Contributes to safer Internet for all

Measuring Impact Against SDG 9

Quantitative Indicators

1. Accessibility Metrics:

- Number of organizations deploying the system
- Geographic distribution of deployments
- Breakdown by organization size (particularly SME adoption)
- Cost reduction compared to commercial alternatives (target: 80% reduction)

2. Capacity Building:

- Number of security professionals trained/using the system
- Growth in local cybersecurity expertise
- Reduction in external security consulting needs

3. Technical Performance:

- Threat detection accuracy rates (target: >90% email, >85% web)
- False positive rates (target: <5%)
- System uptime and reliability (target: 99.9%)
- Response time performance (target: <2 seconds)

4. Innovation Metrics:

- Academic publications produced
- Community contributions and extensions
- Derivative projects and adaptations
- Integration with other security tools

Qualitative Indicators

1. Trust and Confidence:

- User satisfaction with system explanations
- Confidence in automated detection decisions
- Willingness to act on system recommendations
- Overall trust in digital services

2. Empowerment:

- Ability of non-expert users to understand threats
- Reduction in dependency on external expertise
- Increased internal security capability
- Improved security awareness

3. Resilience:

- Reduction in successful cyber attacks
- Faster incident detection and response
- Improved recovery from security incidents
- Enhanced organizational cyber resilience

4. Knowledge Transfer:

- Spread of cybersecurity knowledge to underserved regions
- Development of local security communities
- Adaptation to local contexts and needs
- Contribution to global security knowledge base

Long-Term Vision and Future Directions

This project represents more than a technical solution; it embodies a vision for the future of cybersecurity that aligns with sustainable development principles:

1. Democratized Cybersecurity: A future where advanced security capabilities are accessible to all, not just organizations with large budgets. Where small businesses, educational institutions, healthcare providers, and government agencies in developing countries can protect themselves as effectively as major corporations.

2. Transparent and Trustworthy AI: A future where artificial intelligence serves humanity through understandable, accountable decisions. Where people trust AI systems because they can see and understand the reasoning behind automated decisions.

3. Resilient Digital Infrastructure: A future where societies can safely leverage technology for sustainable development. Where cyber threats don't prevent adoption of beneficial digital services or create insurmountable barriers to digital economy participation.

4. Collaborative Security: A future where security knowledge is shared as a public good. Where the global community collaborates on cybersecurity challenges rather than competing, because everyone benefits from a safer Internet.

5. Sustainable Technology Development: A future where technology solutions consider environmental impact, economic accessibility, and social benefit from the design stage, not as afterthoughts.

Conclusion

By aligning with SDG 9, this project contributes to the broader 2030 Agenda for Sustainable Development. It recognizes that secure, reliable digital infrastructure is not merely a technical concern but a foundational requirement for achieving multiple sustainable development goals.

The Unified Cyber Threat Detection Platform demonstrates that academic research can directly contribute to sustainable development by:

- Making advanced technology accessible to all
- Building capacity in underserved regions
- Supporting economic development through secure digitalization
- Protecting critical infrastructure
- Fostering innovation through open collaboration

As the world becomes increasingly digital, the security and resilience of that digital infrastructure becomes inseparable from broader sustainable development objectives. This project represents one contribution to ensuring that the digital transformation of society proceeds in a way that is inclusive, sustainable, and beneficial for all.

Summary: This project supports SDG 9 (Industry, Innovation and Infrastructure) by making advanced cybersecurity accessible, building trust in digital systems, fostering technical innovation, enabling SME digitalization, and strengthening critical infrastructure resilience. Through these contributions, it indirectly advances multiple related SDGs (3, 4, 8, 16, 17) and demonstrates how technical innovation can directly serve sustainable development objectives.

15. SIMILARITY REPORT (final)

The similarity report obtained from Turnitin should be attached to the final report. The required actions will be announced later.