REAL-TIME SIEM-BASED CYBERSECURITY FRAMEWORK FOR THREAT DETECTION AND PREVENTION IN IOMT ENVIRONMENTS

25-26J-70

Project Proposal Report

FIRAZ M MN – IT22034304

BSc (Hons) Degree in Information Technology
Specializing in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology (SLIIT)

August 2025

REAL-TIME AI THREAT INTELLIGENCE ENGINE FOR CYBERSECURITY FRAMEWORK IN IOMT ENVIRONMENTS

Project ID - 25-26J-70

FIRAZ M MN (IT22034304)

B.Sc. (Hons) Degree in Information Technology
Specializing in Cyber Security

Department of Information Technology
Sri Lanka Institute of Information Technology
Sri Lanka

August 2025

DECLARATION

We declare that this is our own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
FIRAZ M MD	IT22034304	

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Name of supervisor: Kanishka Yapa

Name of co-supervisor: Deemantha Siriwardhana

..... KPY ....20/08/2025....

Signature of the supervisor: Date

(Kanishka Yapa)

ABSTRACT

Health services have been replaced by connecting infusion pumps, imaging systems, patient monitors, and wearables to the clinical network due to rapid adoption of the Internet of Medical Things (IOMT). While this improves operational efficiency and patient treatment, it simultaneously exposes cyber security risk due to old firmware [1], [2]. Such weaknesses make IOMT devices vulnerable for the attackers, while traditional signature-based infiltration systems are struggling against zero-day utilization and advanced persistent threats because they depend on the pattern [7], [13] of predetermined attacks.

This report proposes an AI Threat Intelligence Engine as the second component of a broader SIEM-based cybersecurity framework [21]. The engine leverages lightweight machine-learning and deep-learning models to analyze real-time IoMT traffic, assign anomaly scores, and adapt to evolving threats. It is optimized for CPU-only hospital gateways, integrates self-learning capabilities, and incorporates explainable AI techniques such as SHAP and LIME to increase analyst trust [6], [15]–[17]. By combining Random Forest and gradient-boosting models with autoencoders, isolation forests, and optional hybrid CNN–LSTM architectures, the engine can detect both statistical anomalies and complex temporal patterns [3], [6]

The primary objectives are to minimize false positives, sustain high detection accuracy, and generate interpretable outputs for downstream correlation engines. The methodology encompasses data collection from open IoMT datasets, preprocessing and feature engineering, CPU optimization, adaptive learning, and seamless integration into the larger incident correlation workflow.

Keywords: AI detection engine, IoMT security, SIEM integration, anomaly detection, healthcare cybersecurity

TABLE OF CONTENTS

List of Figures

List of Tables

List of abbreviations

Abbreviation	Description
AI	Artificial Intelligence
API	Application Programming Interface
CSV	Comma-Separated Values
CPU	Central Processing Unit
DDoS	Distributed Denial of Service
DICOM	Digital Imaging and Communications in Medicine
ECG	Electrocardiogram
FPR	False Positive Rate
GPU	Graphics Processing Unit
HL7	Health Level Seven

Abbreviation	Description
IDS	Intrusion Detection System
IoT	Internet of Things
IoMT	Internet of Medical Things
JSON	JavaScript Object Notation
Keras	Deep Learning Framework (Library)
LIME	Local Interpretable Model-agnostic Explanations
LKR	Sri Lankan Rupee
LTS	Long-Term Support (Ubuntu 22.04 LTS)
ML	Machine Learning
MITM	Man-in-the-Middle
MQTT	Message Queuing Telemetry Transport
RAM	Random Access Memory
ROC-AUC	Receiver Operating Characteristic – Area Under the Curve
SHAP	SHapley Additive exPlanations
SVM	Support Vector Machine
TCP	Transmission Control Protocol
OS	Operating System
UNSW-NB15	University of New South Wales NB15 Dataset
CICIDS 2017	Canadian Institute for Cybersecurity Intrusion Detection System 2017
Wi-Fi	Wireless Fidelity

1. INTRODUCTION

1.1 Background

The rapid growth of the Internet of Medical Things (IoMT) has reshaped modern healthcare by connecting devices such as infusion pumps, ventilators, cardiac monitors, imaging systems, smart hospital beds, and wearable health sensors to clinical networks. This interconnected ecosystem supports continuous monitoring, remote adjustments, predictive maintenance, and data-driven diagnostics, ultimately improving both patient care and operational efficiency [1], [4], [18].

Yet, this transformation has also widened the cybersecurity attack surface. Many IoMT devices still rely on outdated firmware, proprietary communication protocols, and limited hardware resources, making them difficult to update or patch without disrupting critical services or undergoing regulatory approval [2]. Consequently, even the compromise of a single device—such as a networked infusion pump or patient monitor—can threaten patient safety, expose sensitive health records, and serve as an entry point for lateral attacks within hospital networks [1], [20].

Conventional intrusion detection systems (IDS) have long depended on signature- or rule-based methods, which compare network traffic against known attack patterns. While effective against familiar threats, these approaches struggle to recognize zero-day exploits and polymorphic malware that disguise their behavior [7], [13]. Anomaly-based IDS were developed to overcome this limitation by modeling normal behavior and flagging irregularities. However, the highly dynamic and diverse nature of IoMT traffic—driven by patient-specific conditions, device variability, and protocol heterogeneity—often leads to excessive false positives, overwhelming security teams and disrupting clinical workflows [2], [15].

To tackle these challenges, this report introduces an **AI Threat Intelligence Engine** as the second core component of a broader SIEM-based cybersecurity framework [21]. The engine employs lightweight machine-learning and deep-learning models optimized for CPU-only hospital gateways, enabling real-time detection of anomalies while remaining resource-efficient. In addition, explainable AI methods such as SHAP and LIME are integrated to provide interpretable insights into each prediction, supporting analyst trust and compliance with healthcare regulations [6], [16], [17]. By working alongside monitoring, correlation, and automated response modules, the engine aims to deliver higher detection accuracy, reduce false positives, and strengthen the resilience of connected healthcare infrastructures [18], [20].

An **AI Threat Intelligence Engine** designed for IoMT must address three core requirements:

1. **Awareness** – Understand healthcare workflows and domain-specific protocols to distinguish meaningful activities from noise [1], [3].
2. **Adaptive Learning** – Continuously refine intelligence models to anticipate and respond to emerging cyber threats [6], [19].
3. **Resource Efficiency** – Deliver real-time intelligence on CPU-only hospital gateways without straining system resources [9], [20].

1.2 Literature Survey

Evolution of Intrusion and Threat Intelligence in IoMT/IoT

Early intrusion defense relied on expert rules and signature matching, which provided high precision for previously seen exploits but offered little resilience against zero-day or polymorphic attacks [13]. Classical machine learning (e.g., SVM, k-NN) broadened detection beyond exact signatures, yet struggled with nonlinearity and high-dimensional network telemetry typical of modern healthcare networks [20].

Ensemble learning techniques including Random Forest and gradient boosting (e.g., XGBoost, LightGBM) improved accuracy by aggregating multiple decision trees and focusing on misclassified examples. LightGBM is particularly suited to intrusion detection because of its leaf-wise tree growth, efficient handling of categorical variables, and high speed. Research shows that hybrid detection frameworks using LightGBM and deep neural networks can achieve classification accuracies above 99% with F1-scores exceeding 0.99

In recent years, deep learning has transformed intrusion detection. Convolutional Neural Networks (CNNs) are increasingly applied to IoT and IoMT traffic to automatically capture spatial patterns in network data, while hybrid architectures such as CNN–LSTM combine spatial and temporal features for stronger anomaly detection. For example, Ebrahimi et al. (2025) demonstrated that lightweight CNNs paired with explainable AI techniques like SHAP and LIME can effectively detect intrusions in IoT environments while maintaining transparency for analysts [9]. Similarly, Xiang et al. (2025) introduced FFT-RDNet, which leverages depth-wise separable convolutions in a residual framework to improve computational efficiency without sacrificing detection accuracy in IoT security scenarios [10]. These advances show that CNN-based architectures, particularly when optimized for efficiency, can deliver high accuracy while remaining suitable for resource-constrained IoMT gateways.

IoMT-Specific Challenges

Healthcare networks exhibit characteristics that complicate threat intelligence. Devices often run constrained hardware and aging firmware, with patch schema limited by continuous clinical operation and regulatory overheads, extending vulnerability exposure [1], [2]. Protocol heterogeneity (HL7, DICOM, MQTT) and clinical workflow variability produce traffic distributions that point with patient state, modality, and scheduling, elevating false positive risk for normal anomaly models [2], [20].

Meanwhile, the safety and privacy stakes are higher than typical enterprise IoMT: compromise of a single infusion pump or monitoring station can pose direct patient-safety risks and sensitive-data exposure [1], [20]. These constraints motivate architectures that (i) understand domain context, (ii) adapt to drift, and (iii) run efficiently on CPU-only gateways close to devices.

Advances in AI-Driven Approaches

Recent literature converges on hybrid pipelines that combine complementary learners to reduce error and stabilize performance under drift. Typical stacks fuse supervised ensembles (Random Forest/LightGBM) with unsupervised components (autoencoders, Isolation Forest) so both discriminative cues and reconstruction-based anomalies are captured [3], [6], [20]. Temporal modeling with LSTM or CNN–LSTM further improves sensitivity to staged or low-and-slow behaviors in medical telemetry [6], [7]. Beyond modeling, systematic reviews emphasize *explainability* as a first-class requirement in regulated settings: SHAP and LIME expose per-feature contributions that help analysts validate alerts and support auditability [15], [16], [17]. For scale and privacy, hierarchical and federated IDS designs allow collaborative learning across sites without centralizing raw patient data, aligning with healthcare compliance needs [18]. Finally, NIST guidance on anomaly detection in cyber-physical systems underscores the importance of robust baselining, drift monitoring, and engineered controls that respect real-time constraints [13].

Synthesis. The literature indicates that no single model suffices for IoMT. Effective engines integrate (1) protocol-aware feature pipelines, (2) hybrid model ensembles spanning supervised and unsupervised views, (3) temporal modeling, (4) continuous adaptation to distribution shift, (5) CPU-efficient deployment, and (6) built-in explainability which together addresses IoMT's safety, privacy, and operational realities [2], [3], [6], [15], [18], [20].

1.3 Research Gap

Despite recent advances in AI-driven cybersecurity have improved intrusion detection across IoT and general IT environments, applying these methods directly to IoMT remains problematic.

- **High False Positives:** Existing anomaly-based approaches often struggle with the variability of medical traffic. results in frequent false alarms that overwhelm clinical security teams [2], [15].
- **Limited Adaptability:** Many deployed models are static and cannot evolve with new attack vectors. Without mechanisms for continuous learning or drift adaptation [3], [6].
- **Resource Constraints:** Sophisticated deep-learning models developed in research often require GPU acceleration, making them impractical for deployment on CPU-only hospital gateways or edge devices [9], [20].
- **Lack of Explainability:** Many black-box models generate predictions without interpretable reasoning. In healthcare, where regulatory compliance and clinician trust are essential, this lack of transparency hinders adoption [15], [16].

These gaps highlight the need for an **AI Threat Intelligence Engine** specifically optimized for IoMT[21].

1.4 Research Problem

The growing use of IoMT devices has strengthened healthcare delivery but also exposed hospitals to new cyber risks. Legacy firmware, proprietary protocols, and resource-constrained hardware make these devices difficult to secure [1], [2]. Conventional intrusion detection is inadequate as those are signature-based methods which may miss zero-day threats, while anomaly-based models generate high false positives in dynamic medical traffic [7], [13]. Existing AI solutions often demand GPU resources or lack explainability, limiting their use in clinical settings [9], [15].

The central research problem is therefore:

How can an AI Threat Intelligence Engine be designed to provide accurate, adaptive, and explainable anomaly intelligence for IoMT traffic while operating efficiently on resource-constrained hospital infrastructure?

2. OBJECTIVES

2.1 Main Objective

To design and implement an **AI Threat Intelligence Engine** tailored for IoMT environments that can generate reliable, interpretable, and resource-efficient threat intelligence for integration into hospital SIEM frameworks.

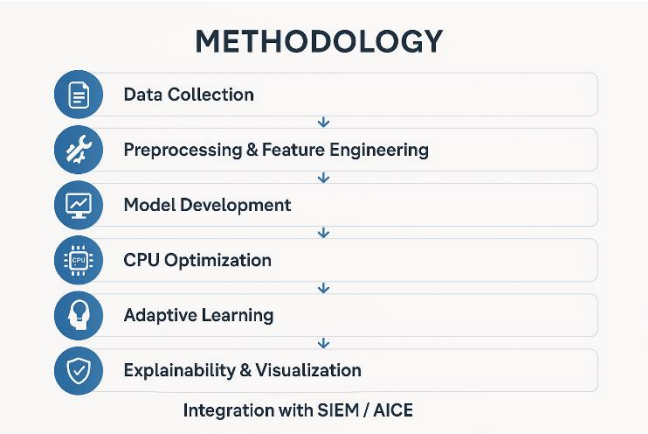
2.2 Specific Objectives

1. **Hybrid Model Ensemble**
    - Develop an ensemble of lightweight machine-learning and deep-learning models (Random Forest, LightGBM, Autoencoder, Isolation Forest, and optional CNN–LSTM) to combine statistical, unsupervised, and temporal perspectives for robust intelligence generation [3], [6].
2. **CPU Optimization**
    - Apply pruning, quantization, and algorithmic simplification to ensure that models operate effectively on CPU-only hospital gateways without requiring high-end GPUs [9], [20].
3. **Adaptive Learning**
    - Incorporate sliding-window retraining, drift detection, and feedback loops to enable the engine to evolve continuously against emerging threats [6], [19].
4. **Protocol-Aware Intelligence**
    - Engineer features from healthcare-specific protocols such as HL7, DICOM, and MQTT to capture domain-relevant semantics that general IDS solutions often overlook [1], [3].
5. **Explainability and Visualization**
    - Integrate SHAP and LIME to produce interpretable outputs, allowing analysts to understand why an event was flagged and building trust in automated intelligence [15], [16].
6. **Seamless SIEM Integration**
    - Standardize outputs in JSON format for smooth integration with the Adaptive Incident Correlation Engine, ensuring compatibility with hospital SOC workflows [21].

3. METHODOLOGY

3.1 Overview

The proposed AI Threat Intelligence Engine is developed through seven sequential stages. Each stage ensures that the system remains accurate, resource-efficient, and explainable while aligning with the operational requirements of hospital environments.



(Fig 3.1 – AI Threat Intelligence Methodology)

3.2 Stage 1: Data Collection

Training and evaluation will be conducted using publicly available datasets to ensure reproducibility. The CICIoMT2024 dataset provides diverse multi-protocol traffic including HL7, MQTT, and DICOM, and covers a wide range of attack scenarios such as denial-of-service, man-in-the-middle, and data exfiltration [4]. To enrich attack diversity, UNSW-NB15, CICIDS2017, and Custom log Dataset. In addition, a controlled testbed will simulate realistic hospital traffic by deploying HL7 servers, DICOM repositories, and MQTT brokers connected to mock patient monitoring devices. This hybrid approach ensures both dataset variety and healthcare-specific protocol fidelity while maintaining anonymization and ethical compliance [2].

Dataset Name	Source	Size (Records)	Format	Purpose
CICIDS 2017	Canadian Institute	10 M	CSV	Benchmark intrusion detection
UNSW-NB15	UNSW Canberra	2.5M	CSV	Network attack classification

CICIDS2017	Canadian Inst. for Cybersecurity	3 M	CSV	Temporal modeling for CNN–LSTM
Custom Log Dataset	Hospital Gateway	500K	JSON	Domain-specific real-time tuning

(Table 3.1- Data set and purposes)

3.3 Stage 2: Pre Processing and Feature Engineering

3.3.1 Cleaning and Normalization

- Duplicate entries and incomplete records will be removed.
- Missing values will be imputed using statistical methods.
- Timestamp formats will be standardized.
- Sequence numbers will be normalized to handle out-of-order delivery.

3.3.2 Feature Extraction

To maintain low computational load:

- Features will be derived at the packet, flow, and session levels.
- Statistical features include packet count, byte count, flow duration, and inter-arrival times.

3.4 Stage 3: Model Development

The chosen models are optimised for CPU-only execution without GPU acceleration:

Model	Type	Pros	Cons	Ref
Random Forest	Supervised	High accuracy, interpretable, robust	Slower training	[9]
Isolation Forest	Unsupervised	Zero-day detection, low memory usage	Needs threshold tuning	[7]
LightGBM	Supervised	Extremely fast, low RAM usage, scalable	Sensitive to params	[28]
Shallow Autoencoder	Unsupervised	Detects subtle anomalies	Needs scaling	[23]

(Table 3.2 – AI model section)

3.4.1 Training Configuration

- Environment: Intel i5 10th Gen CPU, 8GB RAM, Ubuntu/Kali Linux [29].
- Hyperparameter Tuning: Grid Search with cross-validation for each model [14].

- Implementation: Scikit-learn for Random/Isolation Forest, LightGBM Python package, TensorFlow (CPU mode) for Autoencoder [23].

3.5 Stage 4: CPU Optimization

The AI Threat Intelligence Engine will be optimized for CPU-only execution in hospital gateways through the following strategies [9], [20]:

- **Quantization:**
  - Convert model weights and activations from 32-bit floating point to 8-bit integers.
  - Reduces memory footprint and speeds up inference.
- **Pruning:**
  - Remove redundant branches from decision trees and unnecessary neurons in neural networks.
  - Maintains performance while lowering computational load.
- **Batch Inference:**
  - Group 50–100 flows at a time for vectorized processing.
  - Minimizes context-switching overhead and improves throughput.
- **C++ Acceleration:**
  - Implement protocol parsing and feature extraction in C++ and expose via Python bindings.
  - Provides faster runtime for computation-heavy tasks.
- **Lightweight Inference Frameworks:**
  - Deploy models using TensorFlow Lite or ONNX Runtime.
  - Ensures portability and efficiency across different hospital systems.
- **Performance Target:**
  - Maintain CPU usage below 70% and memory consumption under 2 GB.
  - Achieve latency <3 seconds per flow batch to meet real-time detection needs.

3.6 Stage 5: Adaptive learning

• Sliding Window Retraining: Periodically retrain models on the most recent traffic.

• Drift Detection: Monitor distributional changes using KL divergence and Population Stability Index.

• Feedback Integration:

- Analyst-validated alerts (true positives, false positives) reintegrated into training.
- Improves precision and reduces recurring false alarms.

• Reinforcement Learning: Agents dynamically tune thresholds based on analyst/system feedback.

• Federated Learning: Hospitals collaborate by sharing model updates instead of raw data, ensuring privacy [18].

3.7 Stage 6: Explainability & Visualization

SHAP (Shapley Additive Explanations):

- Quantifies each feature's contribution to the prediction. Produces global (average) and local (instance-level) explanations [15].

LIME (Local Interpretable Model-Agnostic Explanations):

- Builds local surrogate models to explain individual flagged flows [16].

Visualization Dashboards:

- Time-series anomaly trends, feature heatmaps, device-specific alerts.

Auditing: Store prediction-explanation logs for compliance and post-incident forensics.

3.8 Stage 7: AICE Integration

• **Output Schema:** JSON messages containing flow ID, anomaly score, model probabilities, SHAP/LIME explanations, and timestamps.

• **Integration:** Outputs forwarded to AICE, where they are correlated with IDS alerts (e.g., Suricata, Zeek), contextual hospital data, and device criticality [21].

• **Feedback Loop:** Analyst validation from AICE (true positives, false positives) is reintegrated into the engine's adaptive learning process.

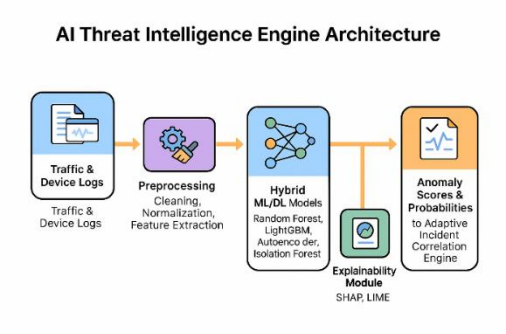
• **Outcome:** Actionable, interpretable threat intelligence delivered to SIEM/AICE for downstream correlation and response.

4. SYSTEM ARCHITECTURE AND TECHNOLOGIES

4.1 Overview

The AI Threat Intelligence Engine follows a **layered pipeline** that ingests IoMT traffic, extracts protocol-aware features, applies a hybrid ensemble of models, explains results, and exports standardized alerts to **AICE/SIEM**. It is optimized for **CPU-only hospital gateways** with clear boundaries to downstream correlation and response components.

Flow: Ingestion → Feature Engineering → Model Ensemble → Score Aggregation → Explainability → Persistence → AICE/SIEM Integration



(Fig 4.1 – AI Threat Intelligence engine architecture)

4.2 Core Components

- **Ingestion & Normalization:** Collect logs/flows from Zeek, Suricata, and testbeds; align timestamps; anonymize identifiers.
- **Feature Engineering:** Extract statistical, temporal, and protocol-specific features (HL7, DICOM, MQTT). Dimensionality reduced via PCA and MIFS.
- **Model Ensemble:** Random Forest, LightGBM, Autoencoder, Isolation Forest, and optional CNN–LSTM run in parallel.
- **Score Aggregation:** Combine per-model scores via weighted averaging/stacking to produce an anomaly probability.
- **Explainability:** Use SHAP for feature attribution and LIME for instance-level rationales.
- **Persistence:** Store predictions, explanations, and drift metrics for audit.
- **Integration with AICE:** Export JSON alerts (flow ID, anomaly score, explanations, metadata). Analyst feedback from AICE is looped back for adaptive learning.

4.3 Deployment Topology

- Runs locally on hospital gateways (Intel® i7, 8 GB RAM).
- Supports optional central Elasticsearch/Kibana for visualization.

4.4 Technologies Used

- **Languages & Frameworks:** Python 3.10+ for pipeline logic; C++ for protocol parsing; TensorFlow Lite/ONNX Runtime for optimized inference.
- **Machine Learning:** Scikit-learn (RF, Isolation Forest), LightGBM, PyTorch/TensorFlow (Autoencoder, CNN–LSTM), SHAP & LIME for explainability.
- **Data & Visualization:** Kafka/RabbitMQ for streaming, Elasticsearch & Kibana for indexing and dashboards, PostgreSQL for configurations.
- **Integration & APIs:** FastAPI/gRPC with JSON Schema validation; compatible with IDS logs (Suricata, Zeek).
- **Deployment & CI/CD:** Docker containers, optional Kubernetes, GitHub Actions for testing/builds, Prometheus + Grafana for monitoring.
- **Security & Compliance:** TLS/mTLS, RBAC, secret vaults; PII minimization; compliance with HIPAA, IEC 80001-5-1, and Sri Lankan Data Protection Act.
- **Performance Optimizations:** INT8 quantization, pruning, batch inference (50–100 flows), vectorization, and optional model distillation for CPU efficiency.

5. PROJECT REQUIREMENTS

The AI Threat Intelligence Engine for IoMT cybersecurity will be developed to meet specific functional and non-functional requirements derived from the project's objectives, target environment constraints, and intended evaluation process. These requirements ensure that the system is both technically feasible and aligned with operational demands in healthcare networks.

5.1 Functional Requirements

1. **Real-Time Processing:** The engine must analyze network flows and generate anomaly scores within 3 seconds of data capture.
2. **Hybrid Model Ensemble:** Must support Random Forest, LightGBM, Autoencoder, Isolation Forest, and optional CNN–LSTM models, with modular integration.
3. **Explainability:** Provide SHAP and LIME-based explanations for every flagged anomaly, both globally (feature importances) and locally (per-instance)
4. **Adaptive Learning:** Implement sliding-window retraining, drift detection, and analyst feedback integration.
5. **Protocol Parsing:** Extract features from healthcare-specific protocols including HL7, DICOM, and MQTT.
6. **Standardized Outputs:** Generate JSON messages containing anomaly scores, probabilities, explanations, and metadata for SIEM/AICE ingestion.
7. **Logging:** Maintain immutable logs of predictions, retraining events, and system diagnostics for auditing and compliance.

5.2 Non-Functional Requirements

The non-functional requirements ensure the system's reliability, efficiency, and compliance with healthcare operational constraints:

1. **Performance:** Achieve F1 scores ≥0.95 and keep false positive rates <5% in benchmark tests.
2. **Efficiency:** Operate with CPU utilization under 70% and memory consumption under 2 GB on Intel® Core™ i7 (10th Gen) processors with 8 GB RAM.
3. **Reliability:** Ensure continuous uptime with automatic failover, health checks, and recovery mechanisms.
4. **Scalability:** Support horizontal scaling via container orchestration (e.g., Kubernetes) for high-traffic hospital networks.
5. **Security:** Encrypt all traffic between modules using TLS; apply role-based access control and input validation to prevent injection attacks.
6. **Usability:** Provide intuitive dashboards and visualization interfaces for analysts, with drill-down features for anomaly exploration.
7. **Maintainability:** Ensure modular code structure, unit/integration tests, and CI/CD pipelines for rapid updates.

6. DESCRIPTION OF PERSONNEL AND FACILITIES

6.1 Personnel

Candidate:

- **Name:** Firaz M M N
- **Student ID:** IT22034304
- **Specialisation:** B.Sc. (Hons) in Information Technology – Specialising in Data Science
- **Role in Project:** Responsible for **designing, developing, training, and evaluating** the AI Threat Intelligence Engine for IoMT environments.

Key Responsibilities:

1. **Research & Literature Review** – Conduct an in-depth study of AI-based intrusion detection methods and IoMT-specific security challenges [4], [18].
2. **Dataset Management** – Acquire and preprocess datasets such as CIC IoMT 2024 and UNSW-NB15 [5], [6].
3. **Model Design & Development:** Implement and optimize ML/DL models (Random Forest, LightGBM, Isolation Forest, Autoencoder, CNN–LSTM) for anomaly prediction under CPU-only constraints [9], [28].
4. **Pipeline Implementation:** Develop a batch-stream processing pipeline to enable real-time flow ingestion, feature engineering, inference, and alert generation [2], [28].
5. **Explainable AI Integration:** Embed SHAP and LIME for transparent anomaly explanations, enabling analysts to interpret flagged events [10], [30].
6. **Evaluation & Testing:** Perform systematic validation using offline benchmarks and simulated hospital testbeds to measure accuracy, false positive rate, and latency [15], [23].
7. **Documentation & Reporting:** Record design decisions, prepare intermediate progress reports, and compile the final dissertation for academic submission.

6.2 Supervisory and Academic Guidance

Supervisor: Mr. Kanishka Yapa

- **Role:** Providing technical direction, reviewing methodology, ensuring compliance with research ethics, and facilitating access to datasets or computational resources.

Co Supervisor: Mr. Deemantha Siriwardana

- **Role:** Assisting in providing technical direction, reviewing methodology, ensuring compliance with research ethics, and other support

6.3 Facilities

The following facilities will be used for the research, with a focus on leveraging **existing resources** to minimize cost:

6.3.1 Hardware

Development Machine:

- Processor: Intel Core i7, 10th Generation
- RAM: 8 GB DDR4 and Storage: 512 GB SSD
- Operating Systems: Windows & Kali Linux (dual boot)
- Networking Equipment:
- Standard Wi-Fi router for internal simulations
- Virtual network setup using VirtualBox for traffic replay.

6.3.2 Software (Free/Open Source)

- Programming & ML Libraries: Python 3.x, pandas, NumPy, Scikit-learn, LightGBM, TensorFlow/Keras (CPU mode)
- Explainability Tools: SHAP, LIME
- Version Control: GitHub (private repository)
- Data Streaming: Apache Kafka or ZeroMQ

6.3.3 Datasets

- CIC IoMT 2024 [5] — Primary dataset for IoMT traffic and attack simulation.
- UNSW-NB15 [6] — Secondary dataset for attack diversity and feature enrichment.
- CICIDS2017 Dataset,Synthetic IoMT Dataset — Locally generated for protocol-specific testing.
- Collected datasets by visiting hospitals.

7. BUDGET AND BUDGET JUSTIFICATION

7.1 Estimated Budget

The budget for this project is minimal since the development will be performed using **personal hardware** and **free/open-source software**. The only costs incurred are related to operational expenses, data storage, and printing of required documents.

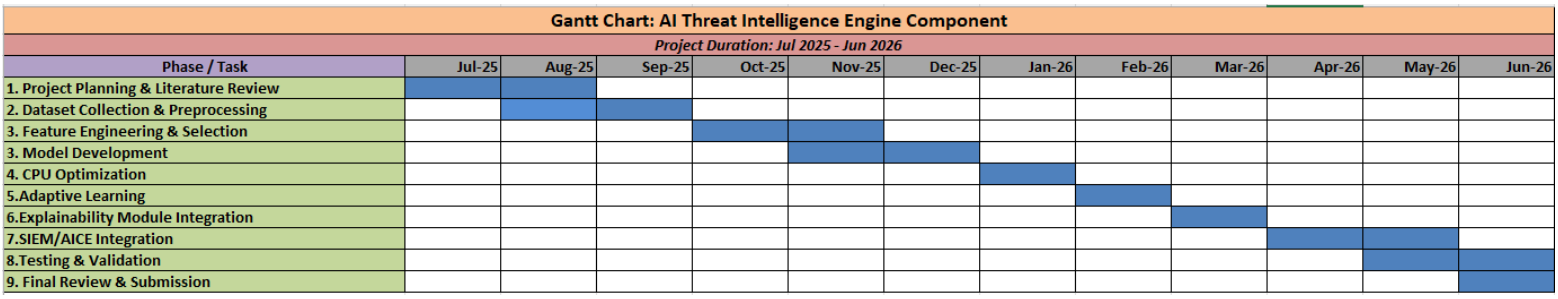
(Table 6.1: Estimated Budget for AI Detection Engine Development)

Item	Description	Estimated Cost (LKR)
Personal Laptop	Intel i7 10th Gen, 8 GB RAM, 512 GB SSD (already owned)	0.00
Electricity & Internet	Power and connectivity costs during training, testing, and simulation	4,000.00
External Storage	1 TB portable HDD for dataset and model backups	30,000.00
Cloud Backup (Optional)	Google Drive 200 GB plan (1-year)	3,000.00
Printing & Binding	Proposal, progress reports, and final dissertation	2,500.00
Contingency	Miscellaneous costs (e.g., USB drives, cables)	2,000.00
Total		41,500.00

7.2 Budget Justification

1. **Personal Laptop**: The system will be developed and tested on an existing Intel i5 10th Gen machine with 8 GB RAM, which meets the performance requirements for lightweight AI model execution [29]. This eliminates the need for new hardware purchases.
2. **Electricity & Internet**: Continuous internet access is required for downloading datasets such as CIC IoMT 2024 [5] and UNSW-NB15 [6], as well as for updating Python libraries and tools. Electricity costs cover extended operational hours during model training and testing.
3. **External Storage**: Large datasets and model checkpoints require secure backups to prevent data loss. A 1 TB portable HDD will be used for offline storage, ensuring availability during hardware failures or reinstallation.
4. **Printing & Binding**: Proposal, progress reports, and the final dissertation will be printed according to SLIIT's formatting guidelines. Binding costs cover both draft and final submissions.

7. GANTT CHART.



(Figure 7.1 – Gantt chart)

8. REFERENCES

[1] Sarah Bin hulayyil, Shancang Li, Neetesh Saxena,Explainable AI-based intrusion detection in IoT systems,Internet of Things,Volume 31,2025,

[2] Balhareth, G.; Ilyas, M. Optimized Intrusion Detection for IoMT Networks with Tree-Based Machine Learning and Filter-Based Feature Selection. *Sensors* **2024**,

[3] Dadkhah, S., Pinto Neto, E. C., Ferreira, R., et al., "AI-based intrusion detection in Internet of Medical Things," 2024.

[4] University of New Brunswick, "CICIoMT2024: Attack vectors in healthcare devices – A multi-protocol IoMT dataset," 2024.

[5] Tan, Hanxiao. “Evaluating Explanation Robustness to Model Pruning.” *2024 International Joint Conference on Neural Networks (IJCNN)* (2024): 1-8.

[6] Ashraf, J., and Latif, S., "Intrusion detection in IoT using deep learning: A systematic review," *IEEE Communications Surveys & Tutorials*, 2021.

[7] Yin, C., Zhu, Y., Fei, J., and He, X., "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, 2017.

[8] Mandru, Deena Babu et al. “Assessing Deep Neural Network and Shallow for Network Intrusion Detection Systems in Cyber Security.” *Computer Networks and Inventive Communication Technologies* (2021): .

[9] F. Ebrahimi, A. Riazi, M. Jamzad, “Intrusion Detection in the Internet of Things using Convolutional Neural Network with Explainable AI,” *Cybersecurity*, SpringerOpen, 2025. DOI: 10.1186/s42400-025-00369-2.

[10] B. Xiang, K. Wu, Y. Zhang, et al., "FFT-RDNet: A Time–Frequency-Domain-Based Intrusion Detection Model for IoT Security," *Sensors*, vol. 25, no. 15, 2025

[11] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, “Explainable AI: Interpreting, Explaining and Visualizing Deep Learning,” *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, Mar. 2021.

[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[13] NIST, "Anomaly detection for cyber-physical systems," *NIST Special Publication 800-193*, 2020.

[14] B. Susilo, T. Mustapha, R. D. K. Kumar, “Intelligent Intrusion Detection System Against Various Attacks in IoT Environments using Scikit-Learn,” Sensors, vol. 25, no. 2, 2025.

[15] S. M. Mohamed et al., "Explainable artificial intelligence models in intrusion detection systems," *Elsevier Computers & Security*, 2023.

[16] J. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, 2017.

[17] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?" *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[18] Y. Liu et al., "A hierarchical intrusion detection system for IoT networks," *IEEE Internet of Things Journal*, vol. 7, no. 10, 2020.

[19] V. K. Pandey, S. Prakash, T. K. Gupta, P. Sinha, T. Yang, R. S. Rathore & S. Tahir, “Enhancing Intrusion Detection in Wireless Sensor Networks using a Tabu Search Based Optimized Random Forest,” *Scientific Reports*, vol. 15, Article 18634, 2025

[20] P. K. Sharma, S. Singh, and R. Kumar, “A Comprehensive Survey on Network Anomaly Detection Systems: Methods, Datasets, and Challenges,” *IEEE Access*, vol. 9, pp. 110841–110875, 2021