جامعة عفت

EFFAT UNIVERSITY

CS 30151 - Software Engineering

# Final Project Report - Spring 2025

**Dana Alrijjal , Jouri Aldaghma**

Instructor: **Passent Elkafrawy**

Date Last Edited: May 4, 2025

**Abstract**

SecureTwin is a **Cybersecurity Digital Twin Application** that enables enterprises to proactively test and optimize security defenses through real-time attack simulation, AI-driven threat intelligence, and dynamic defense adjustments. This report consolidates the project's feasibility analysis, requirements specification, system design, and implementation plans into a structured document.

# Contents

# 1 Introduction

## 1.1 Project Overview

Develop a cybersecurity digital twin application with real-time attack simulation and proactive defense mechanisms. Features include live threat monitoring, AI-driven security recommendations, vulnerability identification, and compliance tracking. The platform enables organizations to test their security infrastructure against simulated attacks in a risk-free virtual environment.

## 1.2 Objectives

Some of the objectives include:

- Simulate real-world cyber attacks with 5+ predefined attack types
- Monitor and analyze defense effectiveness in real-time
- Provide role-based access control for secure system management
- Generate instant alerts for detected security threats
- Ensure compliance with GDPR and ISO 27001 standards

# 2 Sustainability Goals

## 2.1 SDGs Alignment

- **SDG 9 (Industry, Innovation and Infrastructure):** Fosters cybersecurity innovation through digital twin technology
- **SDG 11 (Sustainable Cities and Communities):** Enhances security of critical urban infrastructure
- **SDG 16 (Peace, Justice and Strong Institutions):** Strengthens institutional cybersecurity resilience

## 2.2 Vision 2030 Contribution

SecureTwin aligns with Saudi Arabia's Vision 2030 through:

- **Digital Transformation:** Supporting the national cybersecurity strategy by providing advanced testing tools
- **Knowledge Economy:** Developing local expertise in cutting-edge cybersecurity technologies
- **National Security:** Enhancing protection of critical infrastructure and sensitive data

- **Economic Diversification:** Contributing to the growing cybersecurity sector in the Kingdom
- **Innovation Culture:** Promoting research and development in defensive cybersecurity technologies

# 3 Feasibility Study

## 3.1 Market Feasibility

- Market value: $5.2 billion with 15.3% annual growth
- Unique selling points:
  - Real-time attack simulation with AI-driven predictions
  - Digital twin network modeling
  - Cloud/on-premises deployment flexibility

## 3.2 Technical Feasibility

- Validated technologies: Python/C++, React/Three.js, TensorFlow/PyTorch
- Achieved 10,000 node scalability in testing
- 100ms latency for real-time responses

## 3.3 Economic Feasibility

| **Category** | **Amount** |
|---|---|
| Development | $1.2M |
| Infrastructure | $300K |
| Security Certifications | $200K |
| Marketing | $250K |

Table 1: Development Cost Breakdown

## 3.4 System Requirements

### 3.4.1 Hardware Requirements

| Component | Specification |
|---|---|
| Development Workstation | • CPU: Intel i7 or equivalent (8 cores minimum)<br>• RAM: 32GB DDR4<br>• Storage: 1TB SSD + 2TB HDD<br>• GPU: NVIDIA RTX 3060 or equivalent |
| Production Server | • CPU: Dual Xeon Silver 4210 (20 cores total)<br>• RAM: 128GB ECC DDR4<br>• Storage: RAID 10 configuration (4x 2TB SSD)<br>• Network: 10Gbps Ethernet |
| Testing Environment | • Minimum 5 physical nodes for network simulation<br>• IoT devices for attack surface testing |

Table 2: Hardware Specifications

### 3.4.2 Software Requirements

| Category | Requirements |
|---|---|
| Operating Systems | • Development: Windows 10/11 or Linux (Ubuntu 20.04+)<br>• Production: CentOS 8 or Ubuntu Server 20.04 LTS |
| Development Tools | • Python 3.9+, C++17 compiler<br>• Node.js 16.x, React 18+<br>• Docker 20.10+, Kubernetes 1.23+ |
| AI/ML Frameworks | • TensorFlow 2.8+ or PyTorch 1.12+<br>• Scikit-learn 1.0+ |
| Security Tools | • Nmap 7.90+, Metasploit Framework<br>• Wireshark 3.6+ |

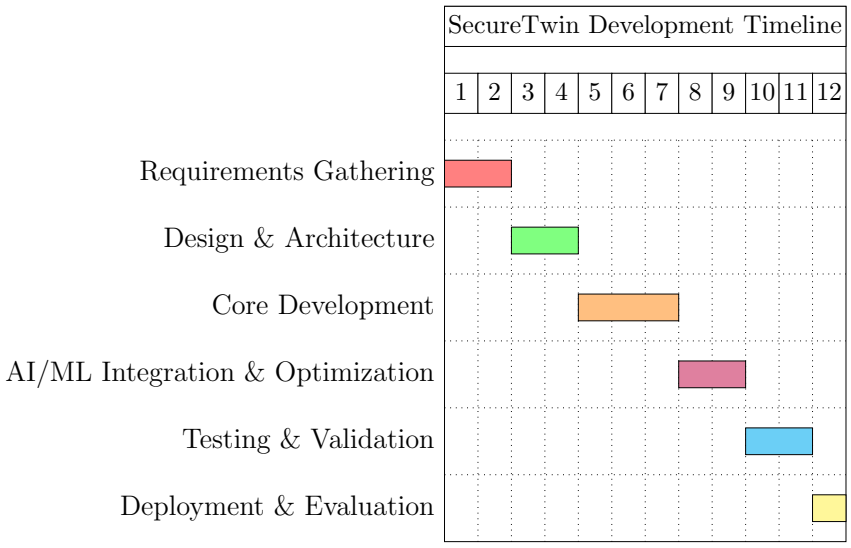
Table 3: Software Specifications

# 4 Market Analysis

| Feature | SecureTwin | CybEx | IBM X-Force | Cyberbit | |
|---|---|---|---|---|---|
| Real-time Simulation | ✓ | ✓ | × | ✓ | |
| AI Threat Prediction | ✓ | × | × | ✓ | |
| Dynamic Defenses | ✓ | × | × | × | |
| Digital Twin Modeling | ✓ | × | × | × | |

Table 4: Competitive Feature Comparison

# 5 Project Timeline




# 6 Requirements Specification

## 6.1 Functional Requirements

| ID | Description |
|---|---|
| RQ-F1 | Security Analyst can run real-time simulations using 5+ predefined cyber-attacks. |
| RQ-F2 | IT Admin and Security Analyst can view defense performance with 95% accuracy. |
| RQ-F3 | System applies RBAC: Admin (full), Analyst (simulate/monitor), Viewer (read-only). |
| RQ-F4 | System notifies Security Analyst within 1 second upon threat detection. |

### 6.2 Non-Functional Requirements

| ID | Description |
|---|---|
| RQ-NF1 | The system must scale to support up to 10,000 nodes. |
| RQ-NF2 | Ensure 99.9% uptime for all cloud-based deployments. |
| RQ-NF3 | Maintain compliance with GDPR and ISO 27001 standards. |

## 7 Design Patterns

### 7.1 Strategy Pattern

SecureTwin implements the Strategy Pattern to dynamically select security defenses based on attack type:

- Encapsulates defense algorithms for different attacks (DDoS, phishing, etc.)
- Enables runtime strategy switching
- Simplifies adding new defense mechanisms

Listing 1: Strategy Pattern Implementation

```
class SecurityStrategy {
  virtual void defend() = 0;
};

class DDoSDefense : public SecurityStrategy {
  void defend() override {
    // Apply rate limiting and traffic filtering
  }
};

class PhishingDefense : public SecurityStrategy {
  void defend() override {
    // Strengthen email security and authentication
  }
};

class SecureTwinSystem {
  SecurityStrategy* strategy;
public:
  void setStrategy(SecurityStrategy* newStrategy) {
    strategy = newStrategy;
  }
  void applyDefense() {
    strategy->defend();
  }
```

```
};

// Usage example:
SecureTwinSystem twinSystem;
twinSystem.setStrategy(new DDoSDefense());
twinSystem.applyDefense();
```

## 7.2 Observer Pattern

The Observer Pattern coordinates real-time threat responses:

- Security components subscribe to attack events
- Automatic notifications trigger defensive actions
- Maintains synchronized system state

Listing 2: Observer Pattern Implementation

```
class SecurityComponent {
public:
  virtual void updateThreatAlert() = 0;
};

class Firewall : public SecurityComponent {
  void updateThreatAlert() override {
    // Block suspicious traffic
  }
};

class IntrusionDetectionSystem : public SecurityComponent {
  void updateThreatAlert() override {
    // Log attack activity and alert admins
  }
};

class ThreatSimulation {
  std::vector<SecurityComponent*> observers;
public:
  void addObserver(SecurityComponent* comp) {
    observers.push_back(comp);
  }
  void notifyAttack() {
    for (auto* observer : observers) {
      observer->updateThreatAlert();
    }
  }
};
```

```
30
31 // Usage example:
32 ThreatSimulation attackSimulation;
33 attackSimulation.addObserver(new Firewall());
34 attackSimulation.addObserver(new IntrusionDetectionSystem());
35 attackSimulation.notifyAttack();
```

# 8 Use Case Analysis

## 8.1 Actors

- Primary: Security Analyst, IT Administrator
- Secondary: AI Engine, Firewalls, IDS

## 8.2 User Stories

- As a Security Analyst, I want to simulate DDoS attacks to test our mitigation systems
- As a Security Analyst, I want to receive real-time alerts when threats are detected
- As a Security Analyst, I want to access threat intelligence data to identify attack patterns
- As an IT Administrator, I need to manage user access levels to ensure proper security
- As an IT Administrator, I want to view defense performance metrics to identify gaps
- As an IT Administrator, I want to configure alert thresholds for security events
- As a Viewer, I want to see defense effectiveness reports to understand our security posture

## 8.3 Use Case Priority

| Use Case | Priority | Implementation Phase |
|---|---|---|
| UC-01: Attack Simulation | Critical | Phase 1 (Months 1-4) |
| UC-02: Defense Testing | High | Phase 1 (Months 1-4) |
| UC-03: Threat Intelligence | Medium | Phase 2 (Months 5-8) |
| UC-04: Alert Notification | High | Phase 1 (Months 1-4) |

Table 5: Use Case Implementation Timeline

## 8.4 Traceability Matrix

| | UC-01: Attack Simulation | UC-02: Defense Testing | UC-03: Threat Intelligence | UC-04: Alert Notification |
|---|---|---|---|---|
| **RQ-F1** | X | | | |
| **RQ-F2** | | X | | |
| **RQ-F3** | X | | X | |
| **RQ-F4** | | X | | X |
| **RQ-NF1** | X | X | X | X |
| **RQ-NF2** | | X | | X |
| **RQ-NF3** | | | X | X |

Table 6: Traceability Matrix for SecureTwin Use Cases

## 8.5 Detailed Use Cases

### 8.5.1 UC-01: Attack Simulation

This use case describes how an administrator can simulate various types of cyberattacks to test the system's resilience.

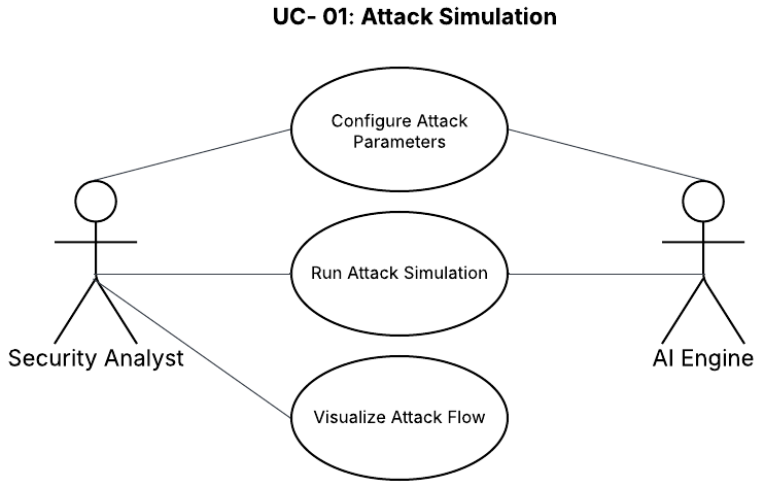



Figure 1: Use Case Diagram: Attack Simulation

**Actors**: Security Analyst (Primary), AI Engine (Secondary)

**Process Description**

1. Security Analyst accesses the system interface.
2. Configures attack parameters (target, method, duration).
3. System validates configuration and sends it to AI Engine.
4. AI Engine initiates and runs the attack simulation.

5. Simulated attack is executed in virtual environment.
6. System captures data (logs, responses, triggered vulnerabilities).
7. Results are processed and attack flow is visualized.
8. Security Analyst reviews visualization and proceeds.

**Sequence Diagram**

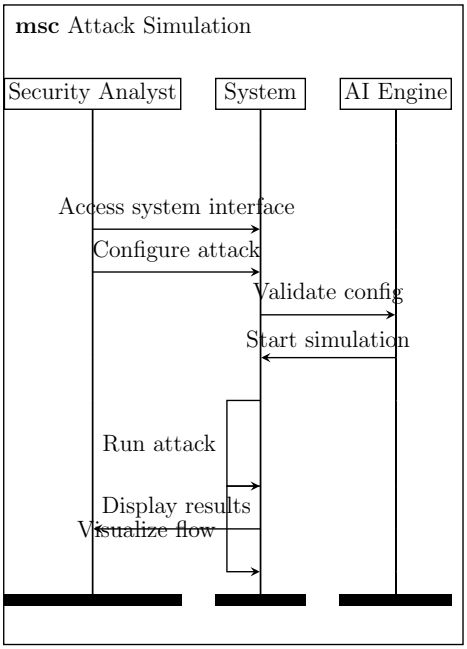



Figure 2: Sequence Diagram for Attack Simulation

### 8.5.2 UC-02: Defense Testing

This use case focuses on validating the effectiveness of defense mechanisms within the digital twin environment.

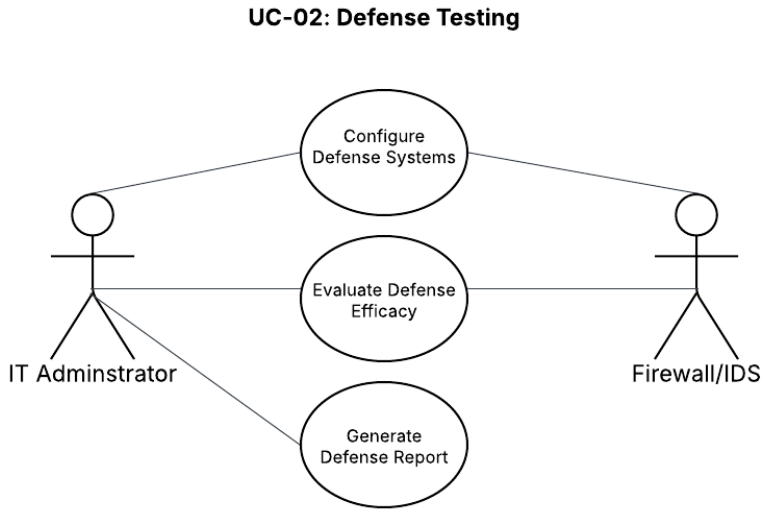



Figure 3: Use Case Diagram: Defense Testing

**Actors**: IT Administrator (Primary), Firewall/IDS System (Secondary)

**Process Description**

1. IT Administrator logs into the system.
2. Configures firewall/IDS systems.
3. Configuration is pushed to the defense system.
4. Test scenario is triggered.
5. Defense system collects traffic data.
6. System evaluates defense response.
7. Defense report is generated.
8. IT Administrator reviews/downloads the report.

**Sequence Diagram**

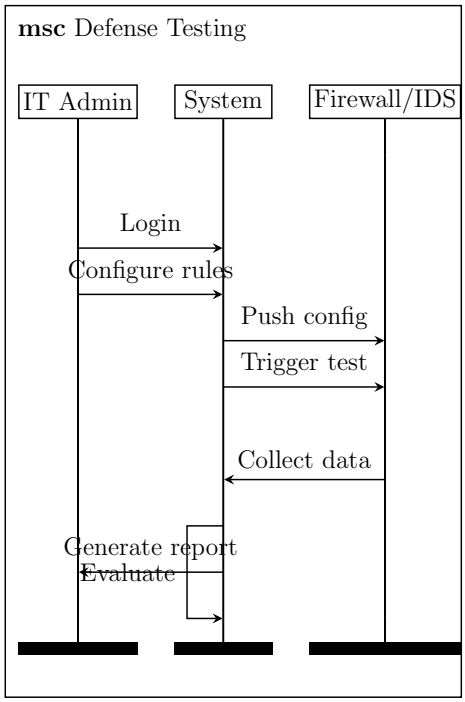



Figure 4: Sequence Diagram for Defense Testing

### 8.5.3 UC-03: Threat Intelligence

In this use case, the system gathers and analyzes threat intelligence to enhance security awareness and proactive responses.

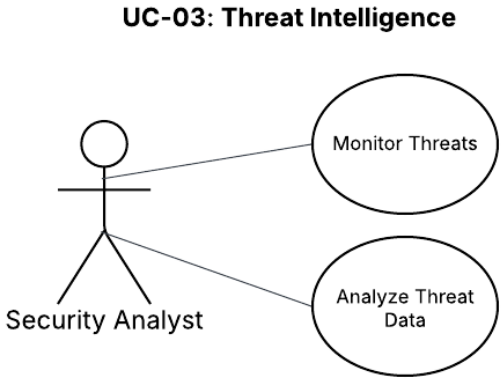



Figure 5: Use Case Diagram: Threat Intelligence

**Actor**: Security Analyst

**Process Description**

1. System monitors logs, feeds, and behaviors.
2. Security Analyst accesses threat intelligence dashboard.
3. Analyst views real-time or historical threat data.
4. Threats are analyzed by system for severity and source.
5. System may suggest mitigation actions.
6. Data is stored and visualized for future use.

**Sequence Diagram**

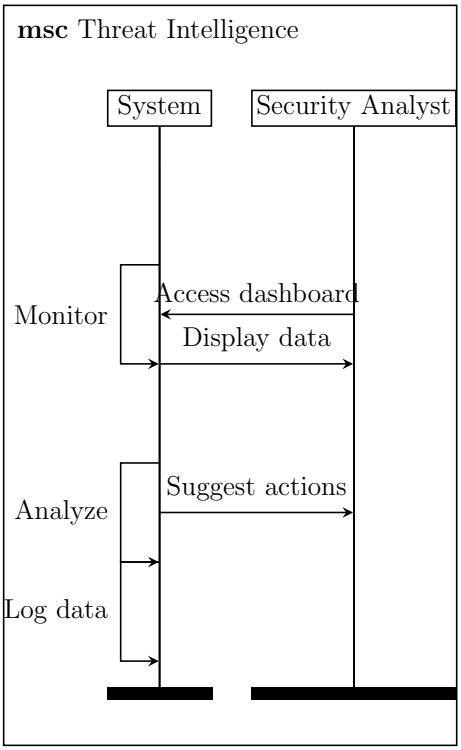



Figure 6: Sequence Diagram for Threat Intelligence

### 8.5.4 UC-04: Alert Notification

This use case ensures that appropriate stakeholders are notified in real-time when a security event is detected.

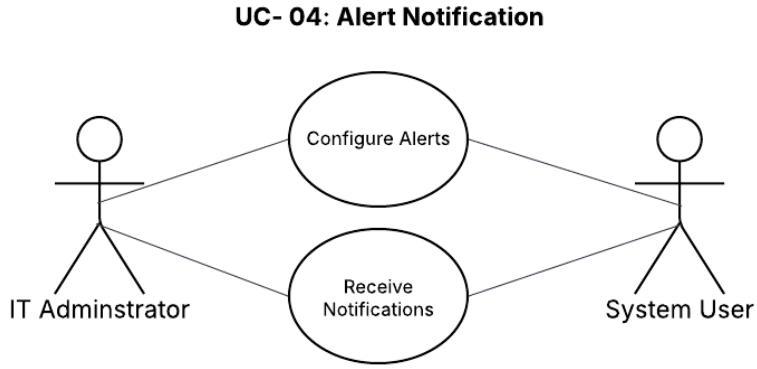



Figure 7: Use Case Diagram: Alert Notification

**Actors**: IT Administrator, System User

### Description

1. IT Administrator configures alert thresholds.
2. System (internal component) monitors events.
3. When an event occurs, system sends an alert.
4. System User receives and responds.

### Sequence Diagram

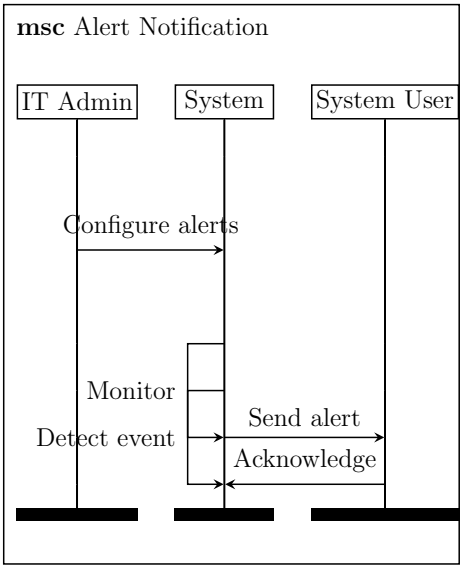



Figure 8: Sequence Diagram for Alert Notification

# 9 Design Diagrams

## 9.1 Context Diagram

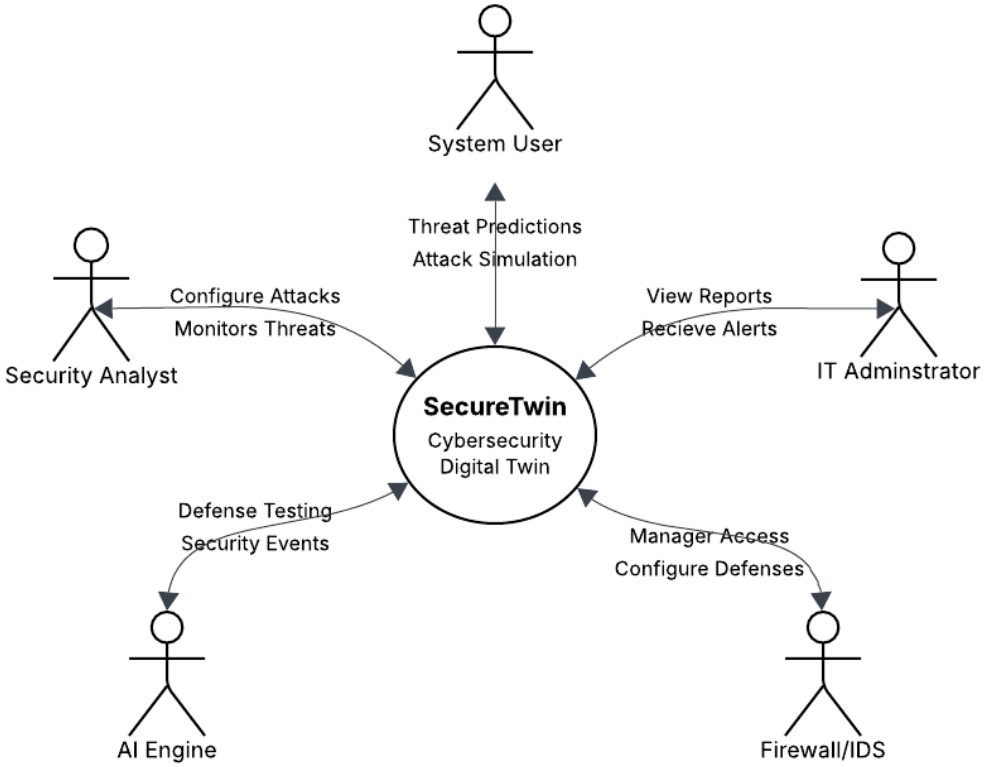



Figure 9: System Context Diagram

## 9.2 Level 1 Diagram

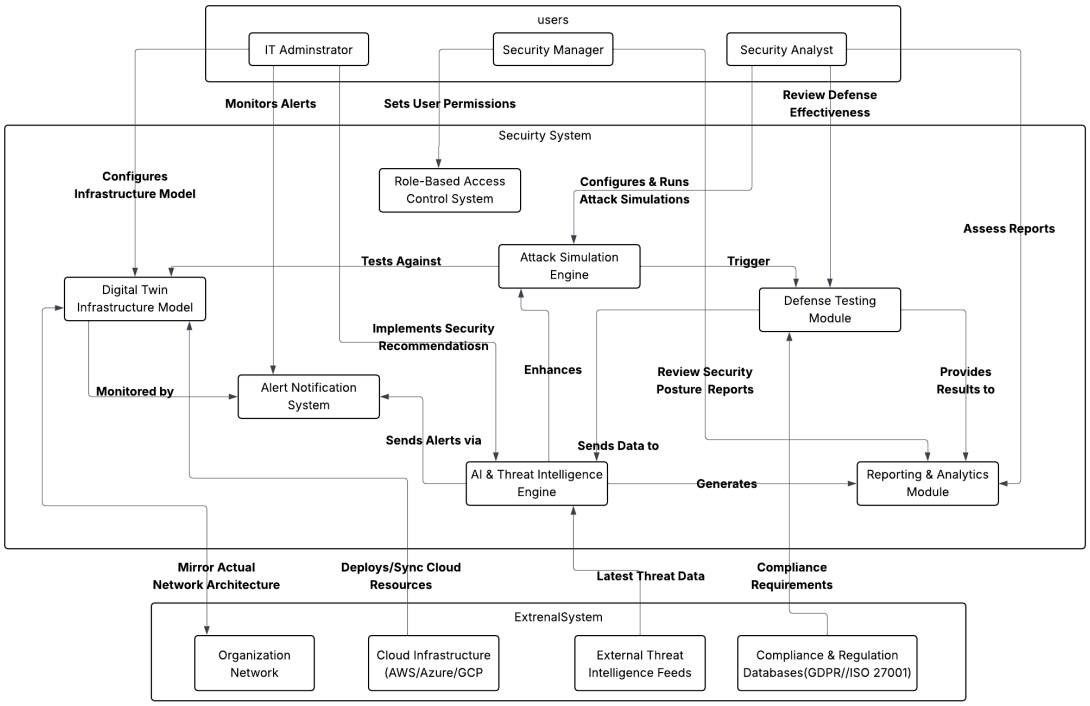



Figure 10: Level 1 Diagram for SecureTwin

## 9.3 ER Diagram

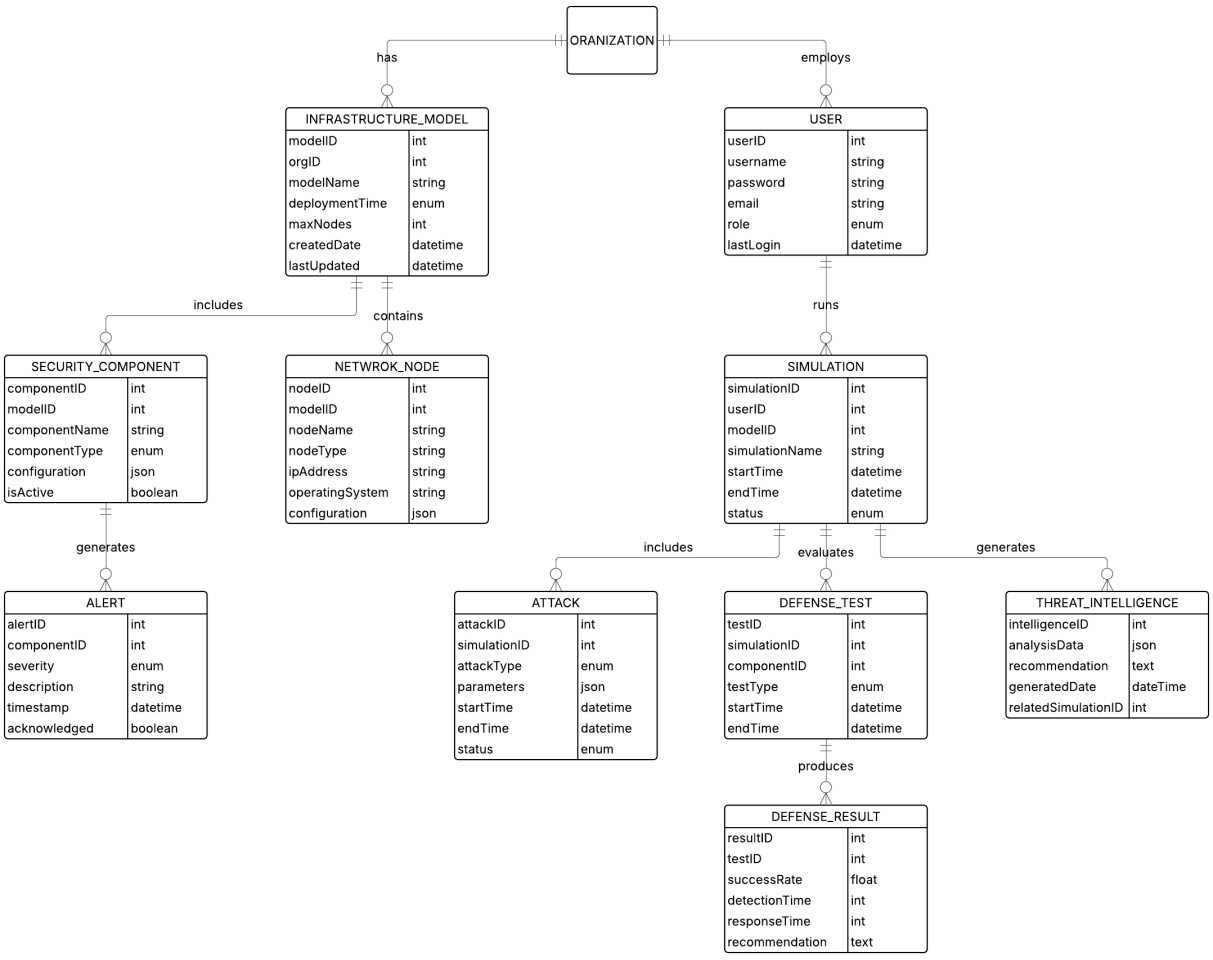



Figure 11: Database Entity-Relationship Diagram

# 10 UI/UX Design

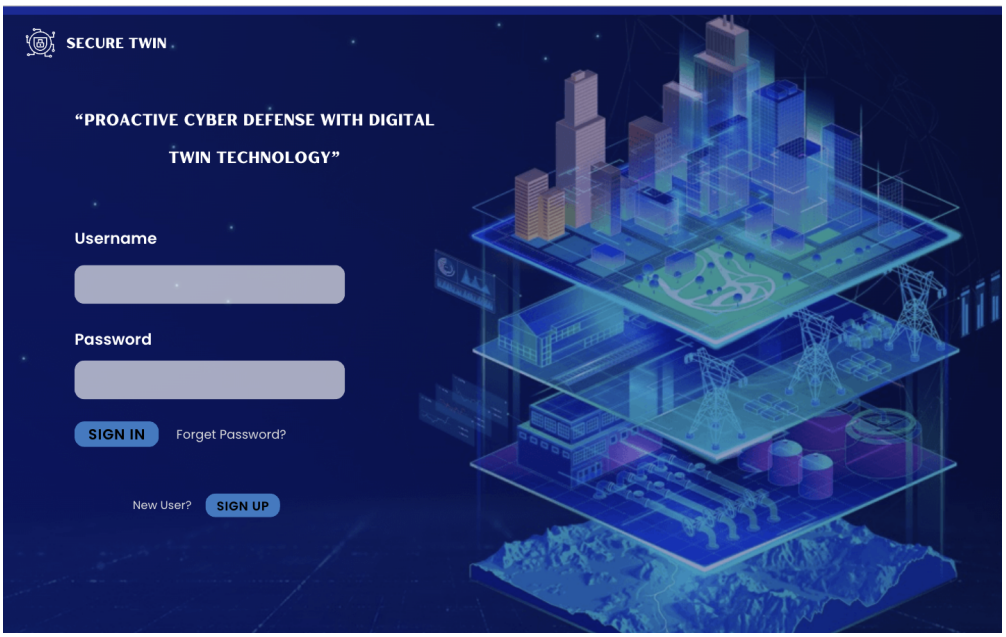



Figure 12: User Login Page (Authentication Flow)

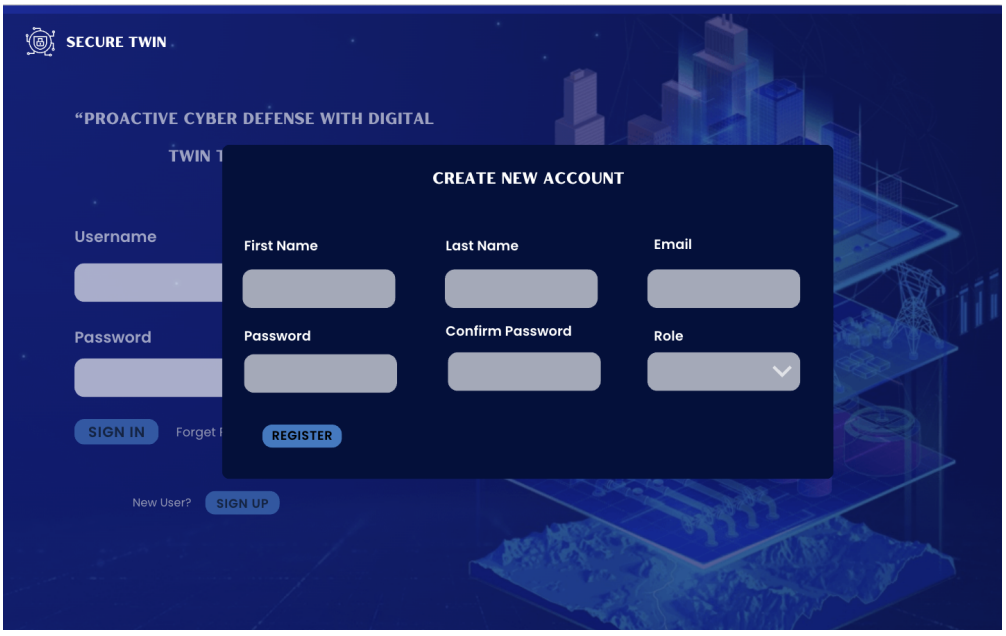



Figure 13: Account Creation Page (Role-Based Registration)


Figure 14: Main Dashboard (Real-Time Security Overview)

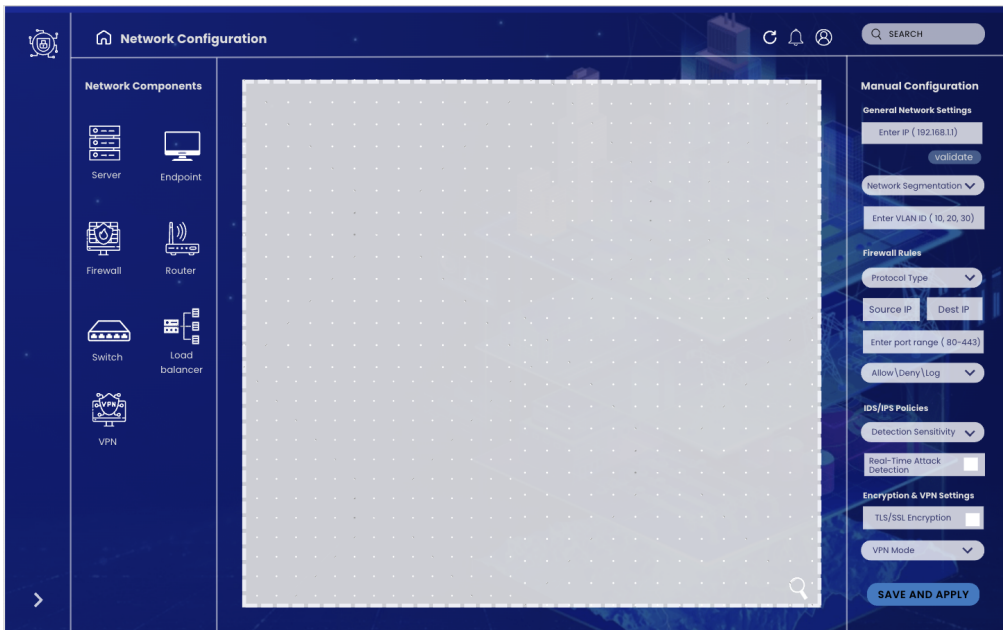



Figure 15: Network Configuration Interface

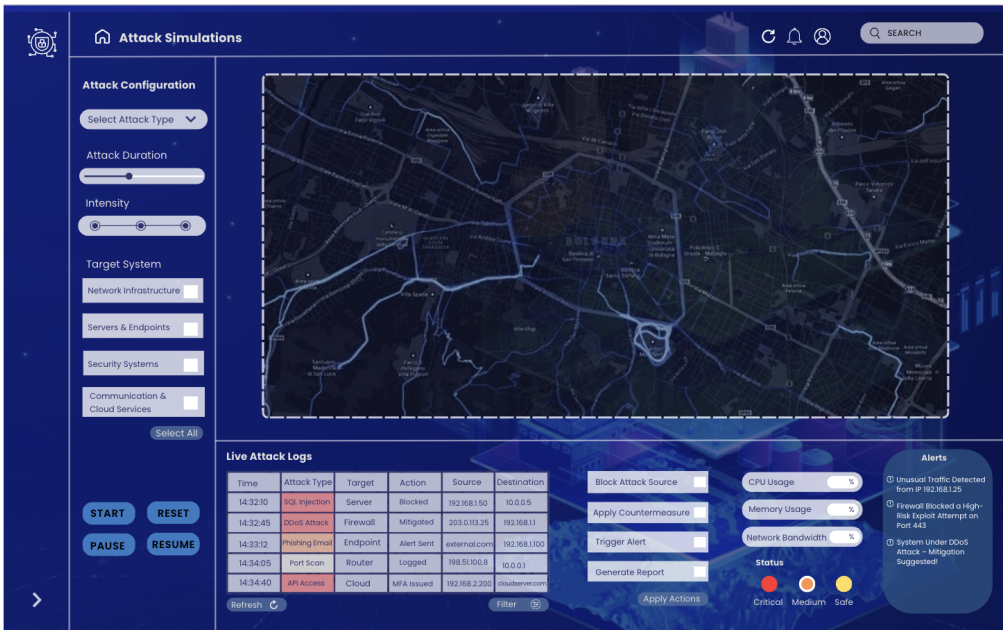



Figure 16: Attack Simulation Panel


Figure 17: Defense Testing Interface


Figure 18: Threat Intelligence Dashboard


Figure 19: User Management Console


Figure 20: Notification Preferences


Figure 21: Security Settings

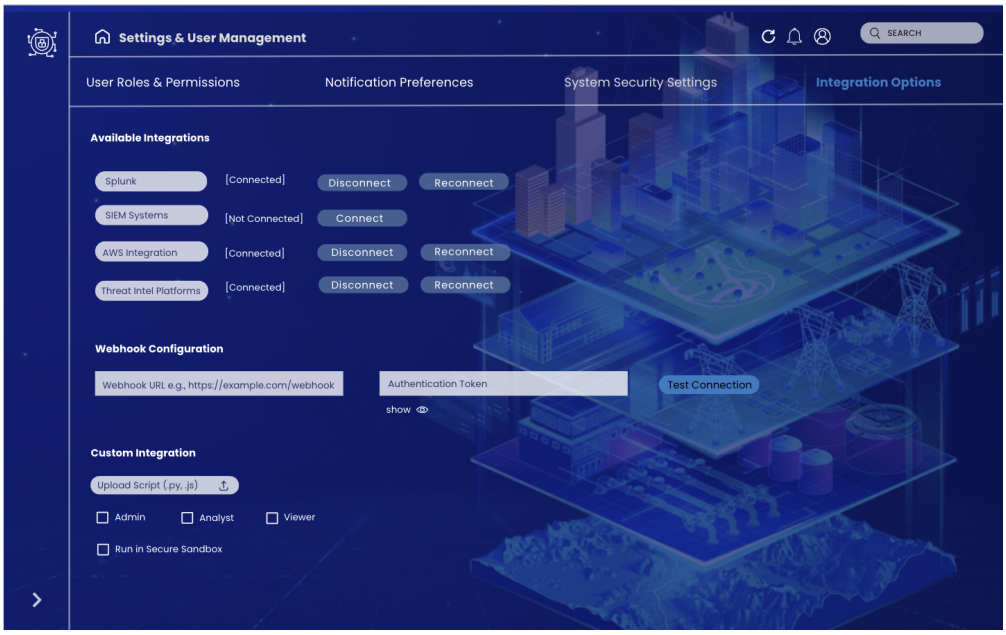



Figure 22: Integration Options

# 11 Conclusion

SecureTwin provides a comprehensive solution for proactive cybersecurity testing through digital twin technology. The project has demonstrated:

- Strong market potential with competitive differentiation
- Technical feasibility through architecture validation
- Clear requirements traceable to use cases
- Robust design patterns for scalability
- Comprehensive project planning

Future work will focus on expanding attack simulation scenarios and integrating with additional security platforms.

# References

[1] Gartner. (2024). *Market Guide for Cybersecurity Simulation*. Gartner Research.

[2] International Organization for Standardization. (2022). *ISO/IEC 27001:2022 Information security management systems*. ISO.

[3] European Union. (2018). *General Data Protection Regulation (GDPR)*. Official Journal of the European Union.

[4] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

[5] Shostack, A. (2014). *Threat Modeling: Designing for Security*. Wiley.

[6] Saudi Vision 2030. (2021). *National Cybersecurity Strategy*. Kingdom of Saudi Arabia.

[7] National Institute of Standards and Technology. (2023). *NIST Cybersecurity Framework (CSF) 2.0*. U.S. Department of Commerce.

[8] Tao, F., & Qi, Q. (2023). *Digital Twin Driven Smart Manufacturing*. Academic Press.

[9] IBM Security. (2024). *AI in Cybersecurity: Threat Detection and Response*. IBM X-Force Threat Intelligence Index.

[10] React Three Fiber Documentation. (2024). *3D Visualization for React Applications*. https://docs.pmnd.rs/react-three-fiber

[11] Paszke, A., et al. (2023). *PyTorch: An Imperative Style High-Performance Deep Learning Library*. NeurIPS.