SAPI-G

Secure API Gateway with AI Threat Detection

Project Report

Secure Software Development

CY-321

Project Team:

Umar Tariq (2022604)
Ayela Israr (2022130)
M Zeeshan (2022644)
Ahmad Amjad (2022063)

Abstract

As APIs become a cornerstone of modern software architecture, securing them against increasingly sophisticated cyber threats is imperative. Traditional rule-based API security methods fall short when confronted with adaptive attacks like injection, session hijacking, and abuse via automation. This project presents **SAPI-G: Secure API Gateway with AI Threat Detection**, a comprehensive solution that combines conventional API security mechanisms with machine learning-based anomaly detection. The system is designed to detect and mitigate threats in real time using AI models such as Random Forest and Grid Search CV, while also enforcing authentication via OAuth 2.0 and JWT. It incorporates features like rate limiting, IP blacklisting, and cryptographic techniques for storing data in the database. A React-based dashboard enables real-time traffic monitoring and threat management. Extensive testing validates the system's resilience against injection attacks, denial-of-service attempts, and session hijacking, while ensuring high availability under load. SAPI-G offers a scalable and intelligent approach to modern API security, suitable for deployment in high-risk environments requiring proactive threat defence.

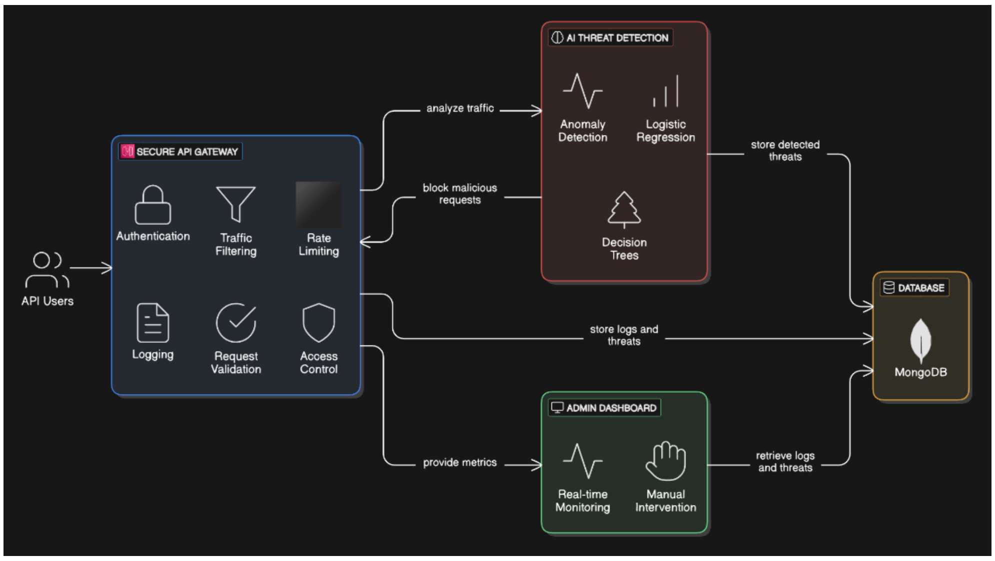
1. Introduction

Modern applications depend heavily on APIs for communication between services and systems. This reliance introduces new vectors for attacks such as SQL injection, DDoS, unauthorized access, and session hijacking. Traditional API security tools depend largely on rule-based systems, which are insufficient against advanced persistent threats and evolving attack patterns.

To overcome these limitations, we developed **SAPI-G**, a Secure API Gateway that integrates **AI-based threat detection** with robust authentication and monitoring mechanisms. The project provides a multi-layered security approach for APIs by combining:

- Rate limiting and IP filtering
- JWT and OAuth 2.0-based authentication
- AI/ML anomaly detection
- Real-time monitoring dashboard

2. System Architecture



2.1 Architecture Components

This system is designed with a multi-layered architecture to ensure secure, scalable, and intelligent API management. At the **Client Layer**, end users and applications interact with the platform by consuming exposed APIs. Requests from clients are first routed through an **API Gateway** built with Node.js, which acts as a frontline defence by handling authentication, logging, rate limiting, and request validation.

To enhance security, an **AI Threat Detection** component, implemented in Python, continuously monitors API traffic using machine learning models. It detects anomalies and can dynamically block suspicious or malicious behaviour in real time.

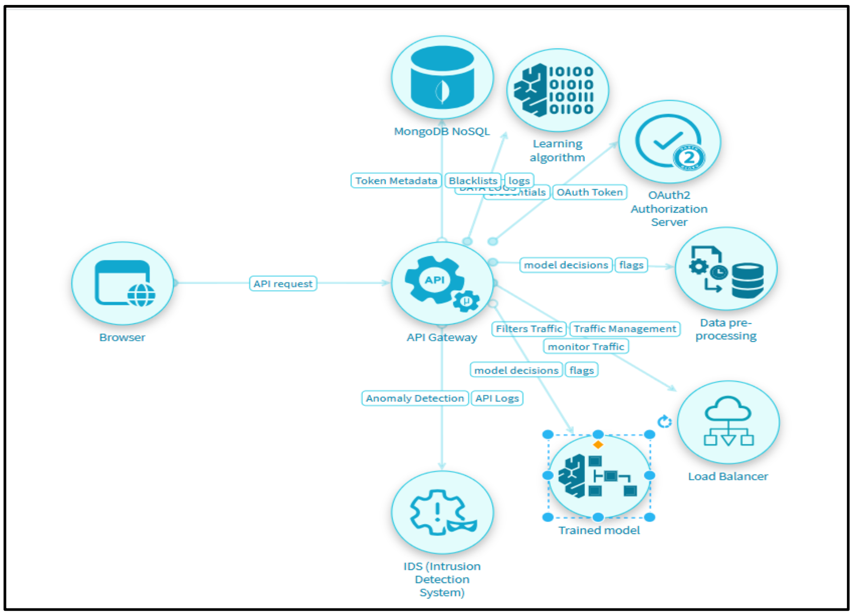
Behind the scenes, **Backend Services**—structured as microservices—carry out core business logic. These services interact with databases such as MongoDB to store critical data including logs, IP addresses, and identified threats.

For operational visibility, an **Admin Dashboard** built with React.js provides administrators with a real-time interface to monitor traffic, view threat analytics, and manage security actions.

All API activity, security alerts, and audit logs are consolidated in a **Database** system, serving as the central repository to support monitoring, compliance, and forensic analysis.

3. Threat Modeling & Risk Assessment

Threat modeling is done using IruisRisk.



3.1 Threat Vectors

- SQL Injection: Unsanitized inputs modify backend queries (e.g., 'OR 1=1--).
- Rate Limiting Bypass: Automated bots flood APIs to perform brute-force attacks.
- D-DoS: Performing many dummy requests on system.
- XSS: Malicious scripts or forced actions executed via the user's browser.
- Insider Threats: Authorized users misuse access, such as leaking API keys.

3.2 Mitigation

- SQL/Command Injection: Use input validation and parameterized queries.
- Rate Limiting Bypass: Apply adaptive rate limiting and bot protection.
- D-DoS: Applying rate limiting and AI model to detect attempts.

- XSS: Sanitize inputs and use detect using AI model.
- Insider Threats: Enforce least privilege, RBAC and monitor user activities.

4. Security Features

4.1 Authentication & Access Control

- **OAuth 2.0** for delegated access
- **JWT** for secure, stateless sessions
- **RBAC** to define roles and restrict access to APIs
- **CORS policy enforcement** for restricting domains

4.3 AI-Based Threat Detection

- Models: **Random Forest** and **GridSearchCV** for hyper parameter tuning.
- Detects anomalies like unusual request behaviour or payload structures
- Blacklists IPs in real-time and auto-blocks future attempts
- AI models trained on synthetic attack datasets and validated using test data

4.4 API Gateway Security Policies

- Rate limiting: restrict requests per user/IP
- IP blacklisting: dynamic
- Request validation: strict parameter schemas

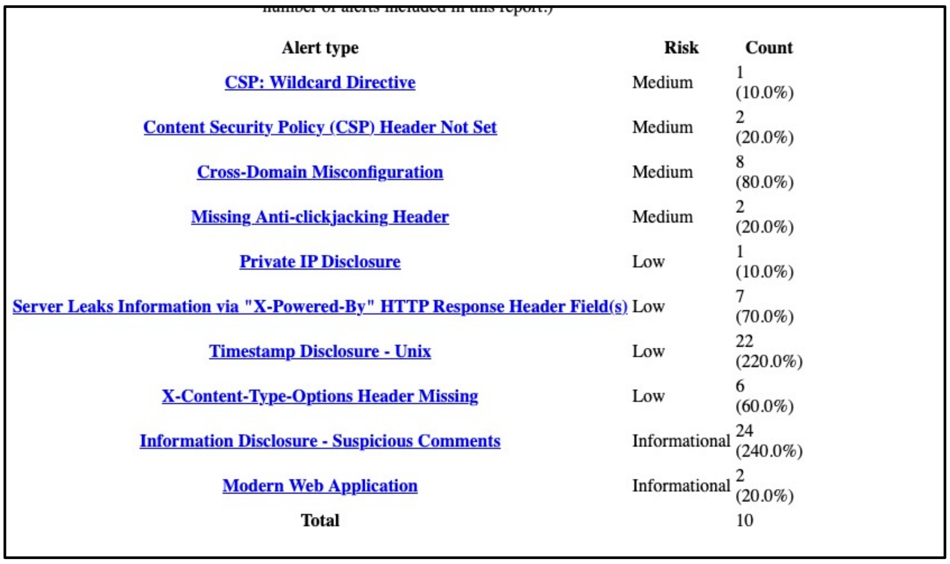
4.5 Logging, Monitoring & Dashboard

- React-based dashboard shows live API traffic
- Alerts on suspicious patterns
- Log viewer with search and filter by IP.

5. Testing Results

5.1 ZAP Testing

Below is the ZAP analysis that shows that all the high-level threats and risks have been mitigated. Only some medium and low level are remaining.



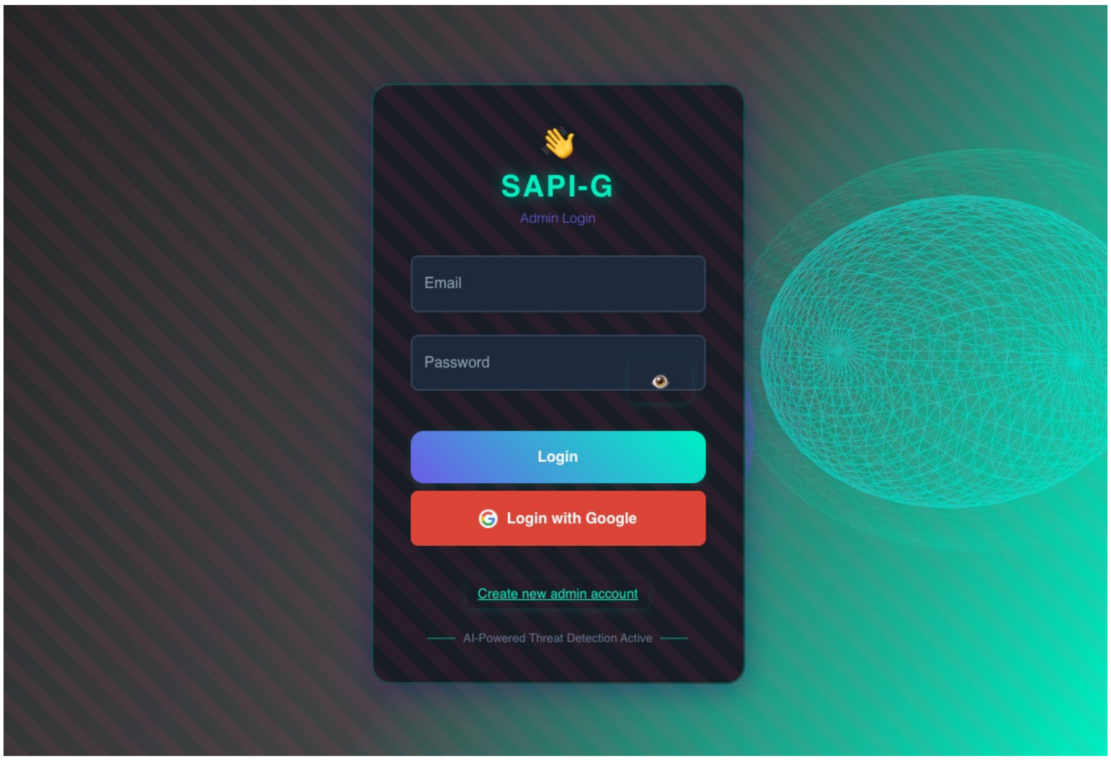
number of alerts included in this report.)

Alert type	Risk	Count
CSP: Wildcard Directive	Medium	1 (10.0%)
Content Security Policy (CSP) Header Not Set	Medium	2 (20.0%)
Cross-Domain Misconfiguration	Medium	8 (80.0%)
Missing Anti-clickjacking Header	Medium	2 (20.0%)
Private IP Disclosure	Low	1 (10.0%)
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	7 (70.0%)
Timestamp Disclosure - Unix	Low	22 (220.0%)
X-Content-Type-Options Header Missing	Low	6 (60.0%)
Information Disclosure - Suspicious Comments	Informational	24 (240.0%)
Modern Web Application	Informational	2 (20.0%)
Total		10

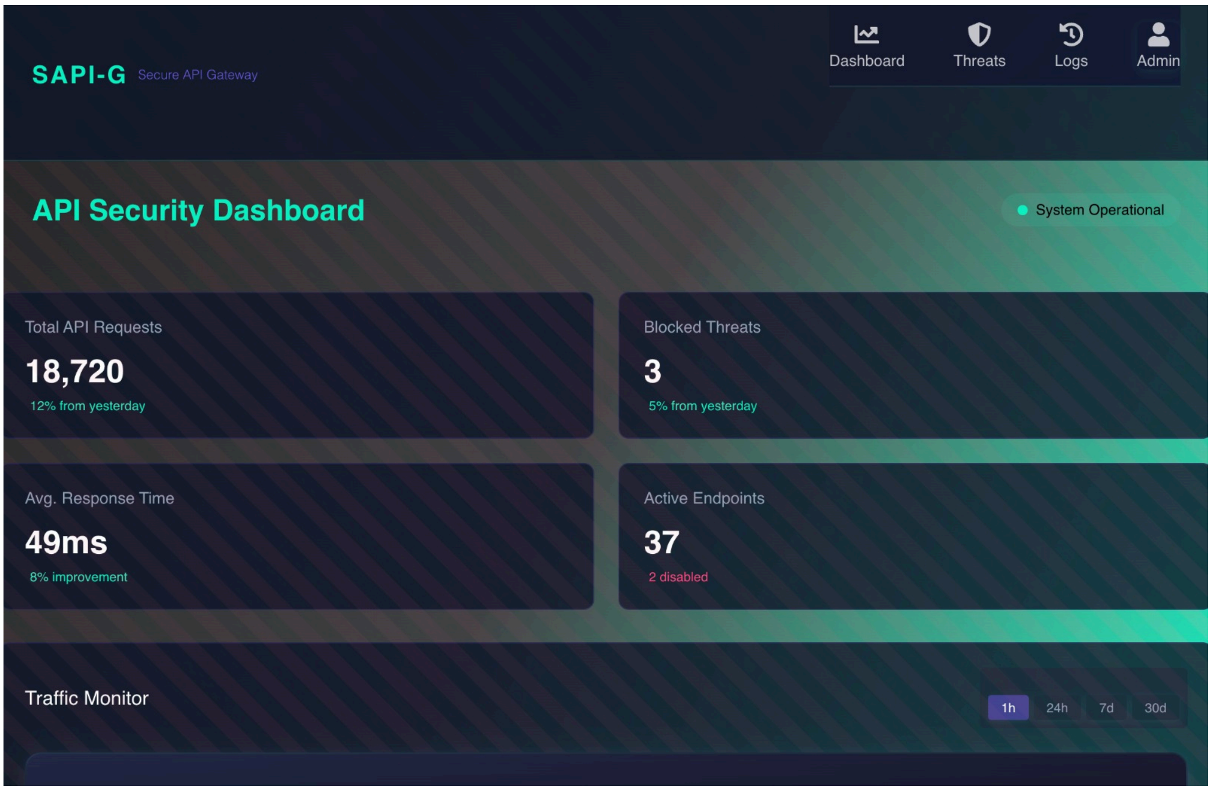
5.2 Manual testing of the working of Web App

Below is the working of the web app

Login Page:



Dashboard Page:



API Security Dashboard

● System Operational

Total API Requests
18,720
12% from yesterday

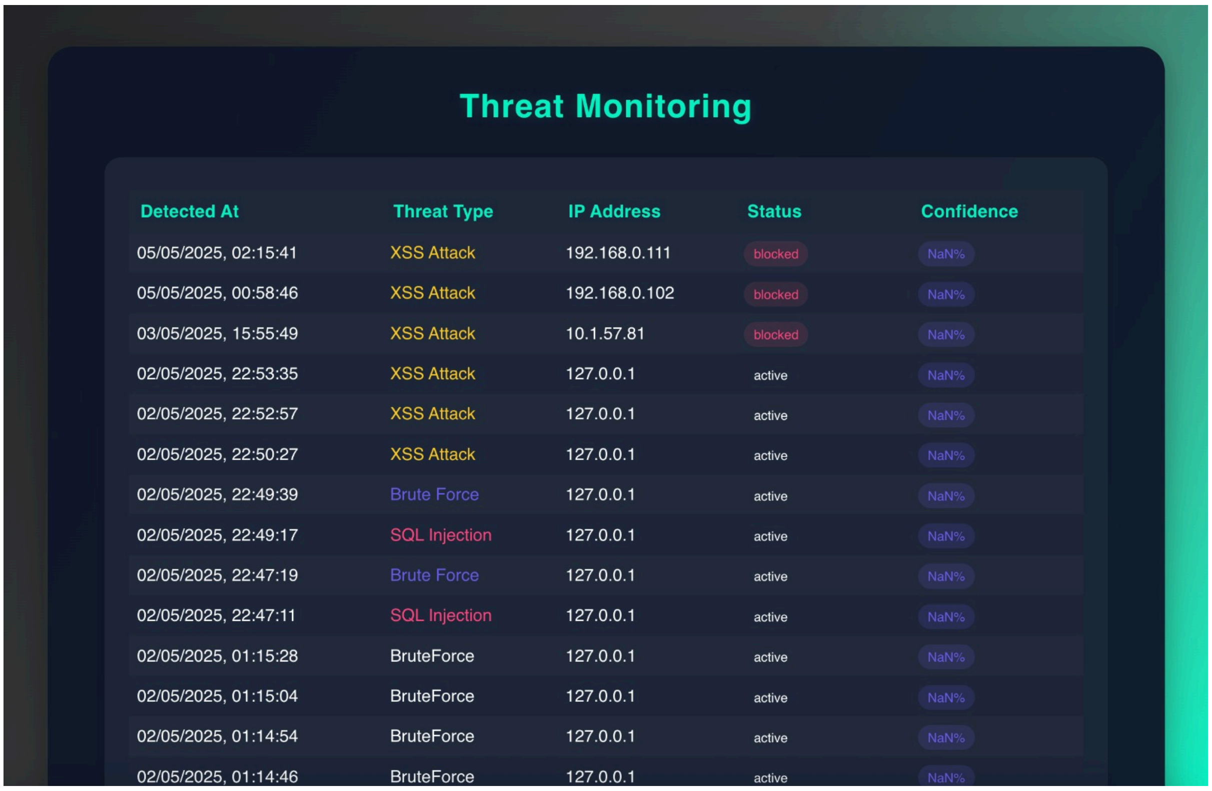
Blocked Threats
3
5% from yesterday

Avg. Response Time
49ms
8% improvement

Active Endpoints
37
2 disabled

Traffic Monitor

Threat Page:

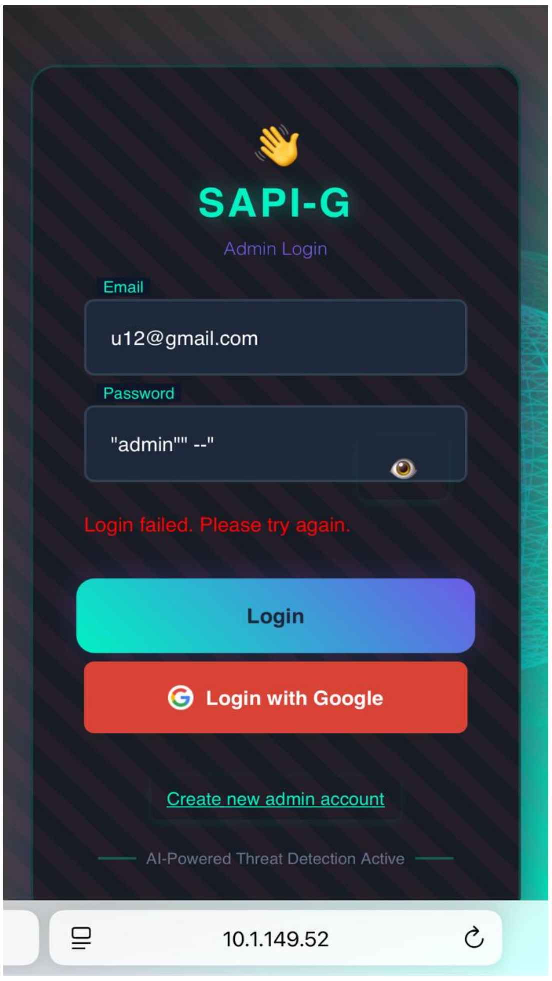


Threat Monitoring

Detected At	Threat Type	IP Address	Status	Confidence
05/05/2025, 02:15:41	XSS Attack	192.168.0.111	blocked	NaN%
05/05/2025, 00:58:46	XSS Attack	192.168.0.102	blocked	NaN%
03/05/2025, 15:55:49	XSS Attack	10.1.57.81	blocked	NaN%
02/05/2025, 22:53:35	XSS Attack	127.0.0.1	active	NaN%
02/05/2025, 22:52:57	XSS Attack	127.0.0.1	active	NaN%
02/05/2025, 22:50:27	XSS Attack	127.0.0.1	active	NaN%
02/05/2025, 22:49:39	Brute Force	127.0.0.1	active	NaN%
02/05/2025, 22:49:17	SQL Injection	127.0.0.1	active	NaN%
02/05/2025, 22:47:19	Brute Force	127.0.0.1	active	NaN%
02/05/2025, 22:47:11	SQL Injection	127.0.0.1	active	NaN%
02/05/2025, 01:15:28	BruteForce	127.0.0.1	active	NaN%
02/05/2025, 01:15:04	BruteForce	127.0.0.1	active	NaN%
02/05/2025, 01:14:54	BruteForce	127.0.0.1	active	NaN%
02/05/2025, 01:14:46	BruteForce	127.0.0.1	active	NaN%

5.3 Manual Testing of the Detector

We will be sending a malicious SQL injection payload from our mobile device to test the effectiveness of our AI model. The payload will be "admin"" --".

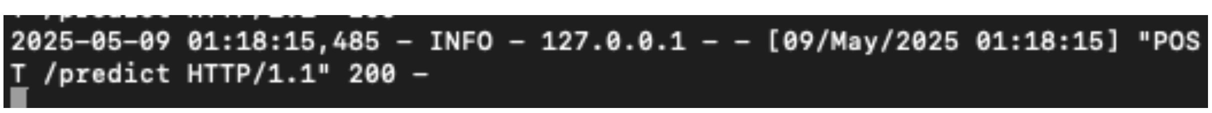


Email: u12@gmail.com

Password: "admin"" --"

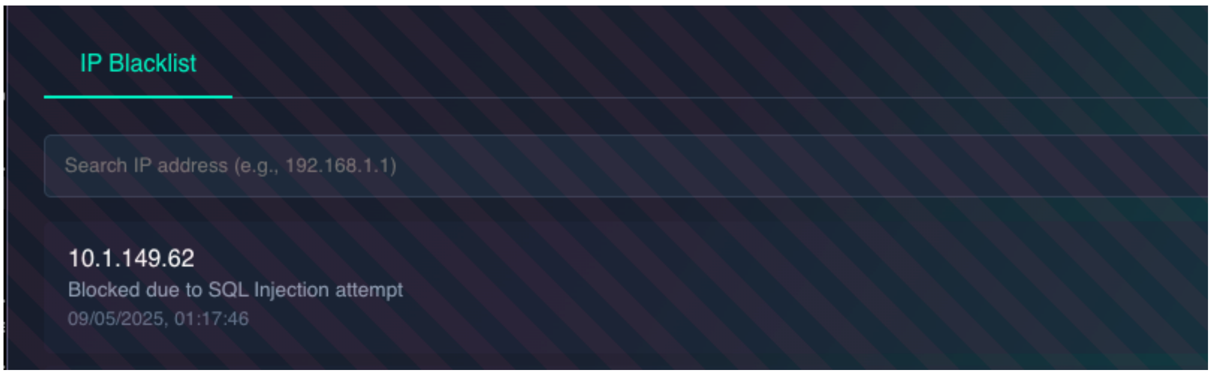
Login failed. Please try again.

Successfully captured by the model:



```
2025-05-09 01:18:15,485 - INFO - 127.0.0.1 - - [09/May/2025 01:18:15] "POS
T /predict HTTP/1.1" 200 -
```

Successfully blocked by the AI model and displayed on the dashboard



10.1.149.62
Blocked due to SQL Injection attempt
09/05/2025, 01:17:46

6. Deployment

- Hosted on **AWS EC2 instance** with **Docker containers**
- Used **NGINX** for reverse proxy and load balancing
- MongoDB hosted with managed cloud services

7. Conclusion

The SAPI-G project successfully integrates traditional API security with modern AI-powered anomaly detection, offering a comprehensive solution for secure and intelligent API management. It provides strong access control, ensuring that only authorized users can access the APIs. The dynamic attack prevention feature enables the system to actively defend against threats in real-time. Additionally, SAPI-G offers real-time visibility and control, allowing administrators to monitor API activity and respond to potential risks as they arise. With a scalable architecture, it is capable of adapting to growing demands while maintaining robust security. Furthermore, the AI-powered system learns and evolves over time, improving its threat detection capabilities. This solution is ready for deployment in environments where secure and intelligent API management is essential.