# Test Plan Document

## Group Name: *TeamOne*

| ID | Team member Name |
|---|---|
| 7741923 | Abdulhayye Maricar |
| 7811056 | Abdul Rahim Kalsekar |
| 8061634 | Ali Sina Mohammad Arif |
| 7869472 | Mansoor Kalemzai |
| 8044661 | Varun Tulsiani |

UNIVERSITY OF WOLLONGONG IN DUBAI

# Proposal Cover Sheet

| | |
|---|---|
| **Subject Code:** | CSIT321 |
| **Subject Name:** | Project |
| **Submission Type:** | Report |
| **Project Title:** | Test Plan Document |

**Student/Team Name:**

| Team Name | *TeamOne* | |
|---|---|---|
| **Team members** | | |
| Student Name | Student ID | Role |
| Ali Sina Mohammad Arif | 8061634 | Leader |
| Mansoor Kalemzai | 7869472 | Presentation Coordinator |
| Varun Tulsiani | 8044661 | Management Tool Coordinator |
| Abdul Rahim Kalsekar | 7811056 | Submission Coordinator |
| Abdulhayye Maricar Asfaq Ahamed | 7741923 | Scribe |

| | |
|---|---|
| **Student Phone/Mobile No.** | +971568648103, +971569879006, +971562859550, +971524628293, +971507907565 |
| **Student E-mail:** | ama438@uowmail.edu.au, mk085@uowmail.edu.au, vmt979@uowmail.edu.au, arrk807@uowmail.edu.au, amaa959@uowmail.edu.au |
| **Lecturer Name:** | Dr. May El Barachi |
| **Due Date:** | 18 Feb, 2025 |
| **Date Submitted:** | 18 Feb, 2025 |

**PLAGIARISM:**
The penalty for deliberate plagiarism is FAILURE in the subject. Plagiarism is cheating by using the written ideas or submitted work of someone else. UOWD has a strong policy against plagiarism.
The University of Wollongong in Dubai also endorses a policy of non-discriminatory language practice and presentation.
**PLEASE NOTE:** STUDENTS MUST RETAIN A COPY OF ANY WORK SUBMITTED

**DECLARATION:**
I/We certify that this is entirely my/our own work, except where I/we have given fully-documented references to the work of others, and that the material contained in this document has not previously been submitted for assessment in any formal course of study. I/we understand the definition and consequences of plagiarism. We/I declare that the project proposal has not been used in any UOWD courses before and that this project idea is a total new idea of the project team

**Signature of Student:** Ali Sina

**Optional Marks:**

**Comments:**

**Lecturer Project Proposal Receipt** (To be filled in by student and retained by Lecturer upon return of assignment)

| | |
|---|---|
| **Subject:** | **Project Title:** |
| **Student / Team Name:** | **Student Number:** |
| **Due Date:** | **Date Submitted:** |
| **Signature of Student:** | |

**Student Project Proposal Receipt** (To be filled in and retained by Student upon submission of assignment)

| | |
|---|---|
| **Subject:** | **Project Title:** |
| **Student/Team Name:** | **Student Number:** |
| **Due Date:** | **Date Submitted:** |
| **Signature of Lecturer:** | |

## Table of Contents

## Table of Figures

# 1. Introduction

We are currently working on **Protego – AI Vulnerability Scanner for Neural Networks**. This project focuses on detecting vulnerabilities through score-based and boundary-based model inversion attacks on pretrained CNN models—specifically ResNet, MobileNet, and EfficientNet. By simulating these attacks, the scanner evaluates how sensitive training data might be reconstructed from model outputs.

In our test plans, we will verify that Protego accurately detects these vulnerabilities. Multiple CNN models will be tested under the defined attack scenarios, while also assessing system performance, scalability, and reliability. Testing using user-provided models via an on-demand scanning interface is essential for identifying any shortcomings and driving improvements. Protego is specifically designed for the deployment phase of CNN-based AI systems.

This document outlines our approach to testing the scanner and the entire project, ensuring that our objectives are met while providing meaningful insights into AI security.

# 2. Test strategy

**2. 1 Overall Testing Approach**

This project will adopt a testing strategy focused primarily on security testing for both frontend and backend components. The approach involves on-demand scanning of pretrained CNN models (ResNet, MobileNet, EfficientNet) using score-based and boundary-based inversion attacks. Testing will primarily be white-box, emphasizing internal code logic, structure, and control paths.

**2.2 Testing Levels**

The testing strategy includes all key levels:

1. **Unit Testing** – Ensures that individual components or functions work correctly.
2. **Integration Testing** – Verifies that different modules interact correctly.
3. **System Testing** – Tests the entire system to validate end-to-end functionality.
4. **Acceptance Testing** – Confirms that the system meets business requirements before release.

**2.3 Testing Environments and Tools**

**Testing Environments:**

- **Development Environment** – Used by developers for initial validation.
- **QA Environment** – Used for functional and performance testing.
- **Staging Environment** – A near-production environment for final validation before release.
- **Production Environment** – The live system where users interact with the product.

**Testing Tools:**

- **Security Testing:** OWASP ZAP for vulnerability scanning.
- **API Testing:** Postman for API security validation.
- **CI/CD Automating:** Security check and automated tests integrated to GitHub workflows.
- **Model Types:** ResNet, MobileNet, EfficientNet.

### 2.4 Entry Criteria

For testing phase to begin, the following conditions must be met:

1. **Code Stability:** The build must be stable and meet minimum development standards.
2. **Test Cases Prepared:** Test scenarios and cases must be documented and approved.
3. **Test Environment:** The required testing environment must be set up and functional.
4. **Access to APIs and External Services:** Any necessary integrations should be available for testing.

### 2.5 Exit Criteria

For testing phase to be considered complete, the following conditions must be met:

1. **Test Case Execution:** All critical test cases must be executed.
2. **Defect Resolution:** Major defects must be fixed, and any remaining issues should have acceptable workarounds.
3. **Performance Benchmarks Met:** System performance should meet expected response times and load requirements.
4. **Sign-Off from Stakeholders:** Test results must be reviewed and approved by relevant stakeholders.

# 3. Functional testing & traceability matrix

**Functional Testing & Traceability Matrix**

**1. Functional Testing Plan** Functional testing ensures that the system features work as intended. Each feature is tested against pre-defined scenarios to validate its expected behavior.

**2. Traceability Matrix** The Traceability Matrix links system requirements to their corresponding test cases to ensure complete test coverage.

| **Requirement Number** | **Test Case Number** | **Test Case Name** | **Test Case Description** | **Sequence of Steps** | **Expected Results** | **Actual Results** | **Pass/ Fail** |
|---|---|---|---|---|---|---|---|

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2.1 Model Vulnerability Scanning | TC-01 | User Login and Module Access | Verify user login and access to scanning module. | 1. Open login page.<br>2. Enter valid credentials.<br>3. Navigate to scanning module. | User successfully logs in and accesses module. | User successfully logs in and accesses module. | Pass |
| | TC-02 | Model Selection | Check model selection functionality. | 1. Open scanning module. 2. Select a model from the list. | User selects a model for scanning. | User selects a model for scanning | Pass |
| | TC-03 | Scan Configuration Validation | Validate scan configuration options. | 1. Open scan settings. 2. Enter valid parameters. 3. Submit configuration. | System accepts valid scan parameters. | System accepts valid scan parameters | Pass |
| | TC-04 | Execute Vulnerability Scan | Execute vulnerability scan. | 1. Select model. 2. Start scan. 3. Wait for completion. 4. View results. | System performs scan and generates a report. | System performs scan and generates a report | Pass |
| | TC-05 | Invalid Scan Parameters Handling | Handle invalid scan parameters. | 1. Enter incorrect parameters. 2. Attempt to start scan. | System prompts user to correct errors. | System prompts user to correct errors | Pass |
| 2.2 Risk Mitigation | TC-06 | Generate Risk Mitigation Insights | Generate risk mitigation insights. | 1. Open scan report. 2. View recommended mitigation steps. | System provides actionable mitigation steps. | System provides actionable mitigation steps. | Pass |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2.3 User Access Management | TC-07 | Create a new User | Creating a new User | 1. Open registration page. 2. Enter details. 3.Create user. | A new user to be created in database. | New user created in database. | Pass |
| | TC-08 | Login | Login for existing users. | 1. Navigate login. 2. Enter credentials. 3. Login. | System verifies user credentials. | System verifies user credentials. | Pass |
| 2.4 User Dashboard and Notification | TC-09 | View Dashboard Alerts | View notifications and alerts on dashboard. | 1. Open user dashboard. 2. Verify displayed alerts. | System displays relevant notifications . | System displays relevant notifications | Pass |
| 2.5 Reports | TC-10 | Generate Reports | Generate reports. | 1. Navigate to dashboard 2. Download report. | System delivers reports. | System delivers reports. | Pass |

**3. Testing Methodology**

- **Manual Testing**: Functional verification using step-by-step execution.
- **Automated Testing**: Test scripts for repetitive scenarios.
- **Performance Testing**: Validate response time and scalability.

**4. Conclusion** This document ensures each system feature is validated against defined criteria, ensuring security, compliance, and efficiency.

# 4. Nonfunctional testing

## 4.1. Performance testing

**Define performance expectations (e.g., response time, resource utilization).**

- Ensure the AI Vulnerability Scanner provides fast and efficient scanning without significant latency.
- Maintain an average scan response time of under 5 seconds for small AI models and under 30 seconds for large-scale models.

- Optimize background processes to avoid excessive resource consumption while ensuring real-time scanning capabilities.

**Specify methods to test system performance.**

- **Response Time Measurement:** Track the time taken for the scanner to detect and report vulnerabilities across different AI models.
- **Resource Utilization Analysis:** Monitor CPU, memory, and network usage during scan operations to ensure optimal performance.
- **Concurrency Testing:** Simulate multiple users initiating scans simultaneously to verify system responsiveness under peak loads.

**Identify success criteria (e.g., response time, throughput).**

- **Response Time:** The system should complete vulnerability scans within 5 seconds for small models and 30 seconds for large models.
- **Resource Utilization:** CPU and memory usage should not exceed 70% under typical workloads and should not exceed 90% during peak conditions.
- **Throughput:** The scanner should be capable of processing at least 50 concurrent AI model scans per minute without performance degradation.

## 4.2. Usability testing

- Define user experience goals.

  Ensure the AI Vulnerability Scanner provides an intuitive and user-friendly interface, even for users with limited cybersecurity or AI expertise.

  Allow seamless navigation through well-defined UI components, such as scan initiation, result visualization, and vulnerability mitigation actions.

  Minimize user effort by automating key processes such as anomaly detection and compliance reporting.

- Specify how you will test usability (e.g., user surveys, A/B testing, heuristic evaluation).

  **User Surveys:** Conduct user satisfaction surveys with security engineers and compliance officers to assess ease of use.

  **A/B Testing:** Compare different UI layouts and workflows to determine which design enhances user interaction.

**Heuristic Evaluation:** Conduct expert reviews to ensure the system aligns with usability best practices, such as error prevention and efficient task completion.

- Identify success criteria (e.g., task completion time, error rates).

  **Task Completion Time:** Users should be able to initiate a model scan within 3 clicks and retrieve results in under 10 seconds.

  **Error Rates:** System interactions should have an error rate of less than 5% in usability tests.

  **User Satisfaction Score:** Aim for a score of at least 80% in user surveys regarding ease of navigation and feature accessibility.

## 4.3. Reliability testing

- Define reliability expectations (e.g., system uptime, failure recovery).

  Maintain at least **99% uptime** to ensure continuous real-time scanning for vulnerabilities.

  Ensure that the AI Vulnerability Scanner operates **fail-safe**, meaning it logs errors and prevents disruptions during scanning failures.

  Guarantee that compliance reports and security logs remain **intact and retrievable** even in the event of system failure.

- Specify methods to test system stability over time.

  **Uptime Monitoring:** Deploy automated tools to track system uptime and detect unplanned downtimes.

  **Recovery Testing:** Simulate system crashes and evaluate the time taken for automatic recovery.

  **Data Integrity Checks:** Validate that no security logs or scan results are lost in case of failure.

- Include stress testing and load testing plans.

  **Stress Testing:** Push the scanner to its limits by running simultaneous scans on multiple AI models and measuring response times.

  **Load Testing:** Simulate real-world usage by testing performance with **high concurrent user activity** and ensure the system does not degrade beyond acceptable limits.

  **Failover Testing:** Check if the system can **switch to backup servers** without losing ongoing scans or reports.

## 4.4. Security testing

**Identify key security threats (e.g., SQL injection, XSS, authentication flaws).**

Security threats that need to be tested include:

- **SQL Injection:** Ensure that user input is properly sanitized to prevent unauthorized database access.
- **Cross-Site Scripting (XSS):** Test for injection of malicious scripts that can manipulate user sessions.
- **Authentication and Authorization Flaws:** Verify that only authorized users can access sensitive modules.
- **Man-in-the-Middle Attacks (MITM):** Assess encryption and secure communication protocols to prevent interception.
- **Data Exposure and Leakage:** Ensure that sensitive model and user data is properly protected and encrypted.
- **Denial-of-Service (DoS) Attacks:** Simulate high request volumes to assess system resilience.
- **Broken Access Control:** Test for privilege escalation vulnerabilities that allow unauthorized access to system functions.
- **Insecure API Endpoints:** Check API requests and responses to prevent data breaches.

**Define security test cases (e.g., penetration testing, access control validation).**

| Test Case ID | Test Case Name | Description | Steps | Expected Outcome |
|---|---|---|---|---|
| SEC-01 | SQL Injection Prevention | Test input fields against SQL injection attempts | 1. Enter malicious SQL query in input fields. 2. Monitor system response. | System should not process malicious queries. |
| SEC-02 | Cross-Site Scripting (XSS) | Validate input sanitization against XSS attacks | 1. Inject a script into form fields. 2. Submit input. | System should block script execution. |

| | | | | |
|---|---|---|---|---|
| SEC-03 | Broken Authentication Test | Test login vulnerabilities | 1. Attempt login using weak passwords. 2. Test session hijacking methods. | System should enforce strong authentication. |
| SEC-04 | API Security Test | Test API endpoints for security flaws | 1. Send unauthorized requests. 2. Monitor data exposure. | API should deny unauthorized access. |
| SEC-05 | Denial-of-Service (DoS) Test | Simulate high request volume to check system resilience | 1. Flood system with requests. 2. Observe performance impact. | System should remain stable and mitigate excessive requests. |
| SEC-06 | Data Exposure Validation | Check for improper data storage and transmission | 1. Inspect stored user data. 2. Intercept data packets during transmission. | Data should be encrypted and protected. |

**Specify tools used (e.g., Burp Suite, OWASP ZAP).**

- **Burp Suite:** Used for penetration testing and vulnerability assessment.
- **OWASP ZAP:** Conduct automated security scans to detect vulnerabilities.
- **Metasploit:** Test for exploitation and system vulnerabilities.
- **Nmap:** Scan network infrastructure for security flaws.
- **Wireshark:** Analyze network traffic and detect potential MITM attacks.
- **Kali Linux Security Tools:** Perform ethical hacking and security assessments.

## 5. Risks and mitigation

**Identify potential risks in the testing process (e.g., unclear requirements, environment issues).**

- **Unclear Requirements:** Lack of detailed documentation may result in incomplete testing.
- **Environment Issues:** Inconsistent testing environments could lead to unreliable results.
- **False Positives in Security Testing:** Some vulnerabilities detected may not be exploitable in real-world scenarios.
- **Limited Testing Time:** Deadlines may restrict comprehensive security testing.

- **Data Privacy Concerns:** Handling of sensitive data during tests could pose legal risks.
- **Third-Party Dependencies:** Vulnerabilities in third-party tools or libraries could impact security.
- **Resource Constraints:** Limited access to security professionals or tools can affect test coverage.

**Provide mitigation strategies (e.g., buffer time, additional documentation, backups).**

- **Requirement Clarity:** Conduct detailed requirement analysis and ensure well-documented test plans.
- **Standardized Test Environments:** Maintain consistent test setups for accurate comparisons.
- **Risk-Based Prioritization:** Focus on high-risk vulnerabilities first to maximize security impact.
- **Regular Security Audits:** Conduct periodic security reviews to identify and address new risks.
- **Data Handling Protocols:** Use anonymized test data and adhere to compliance guidelines.
- **Backup and Disaster Recovery:** Implement backup solutions to restore system integrity in case of attacks.
- **Continuous Learning & Training:** Keep the testing team updated on the latest security threats and techniques.

## 6. Conclusion

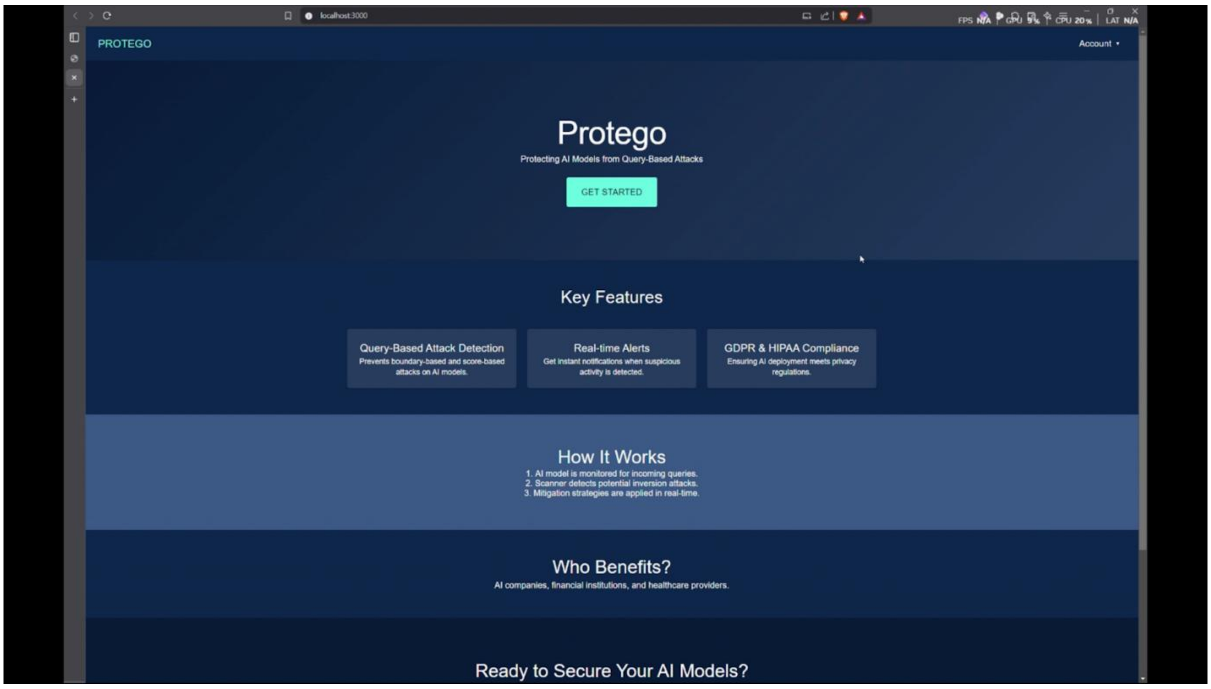
In summary, our testing strategy allowed us to evaluate the performance, security, and usability of our AI Vulnerability Scanner. We focused on assessing how effectively the scanner detects model inversion attacks via score-based and boundary-based query attacks, while also confirming that the scanner operates on many types of models by using open-source CNN models. This approach proved the accuracy of our detection as well as highlighting areas where performance and user experience could be improved.

Our next steps will be working on any identified issues, and re-testing the components to make sure that the defects are resolved. We will continue tracking key benchmarks such as response time and system scalability. Approval for these changes will be based on the execution of all the test cases, the resolution of the defects, and confirmation from both all the team members and our industry partner(DTS).

By following these testing plans, we are confident that our AI Vulnerability Scanner will meet the project objectives as well as provide proper insights into AI security as a whole.

# 7. Actual Results

## 7.1 Landing Page



Protego

Protecting AI Models from Query-Based Attacks

Key Features

Query-Based Attack Detection
Prevents boundary-based and score-based attacks on AI models.

Real-time Alerts
Get instant notifications when suspicious activity is detected.

GDPR & HIPAA Compliance
Ensuring AI deployment meets privacy regulations.

How It Works

1. AI model is monitored for incoming queries.
2. Scanner detects potential inversion attacks.
3. Mitigation strategies are applied in real-time.

Who Benefits?

AI companies, financial institutions, and healthcare providers.

Ready to Secure Your AI Models?

*Figure 1 Landing Page*

This showcases Protego's primary landing page. It highlights the scanner's key features, briefly explains how Protego works, and provides quick links to **Register** or **Login**. The page serves as the user's first point of contact, introducing the system's main functionalities—such as query-based attack detection and compliance adherence—and guiding new users to create an account or sign in to begin scanning their AI models.

## 7.2 Register Page

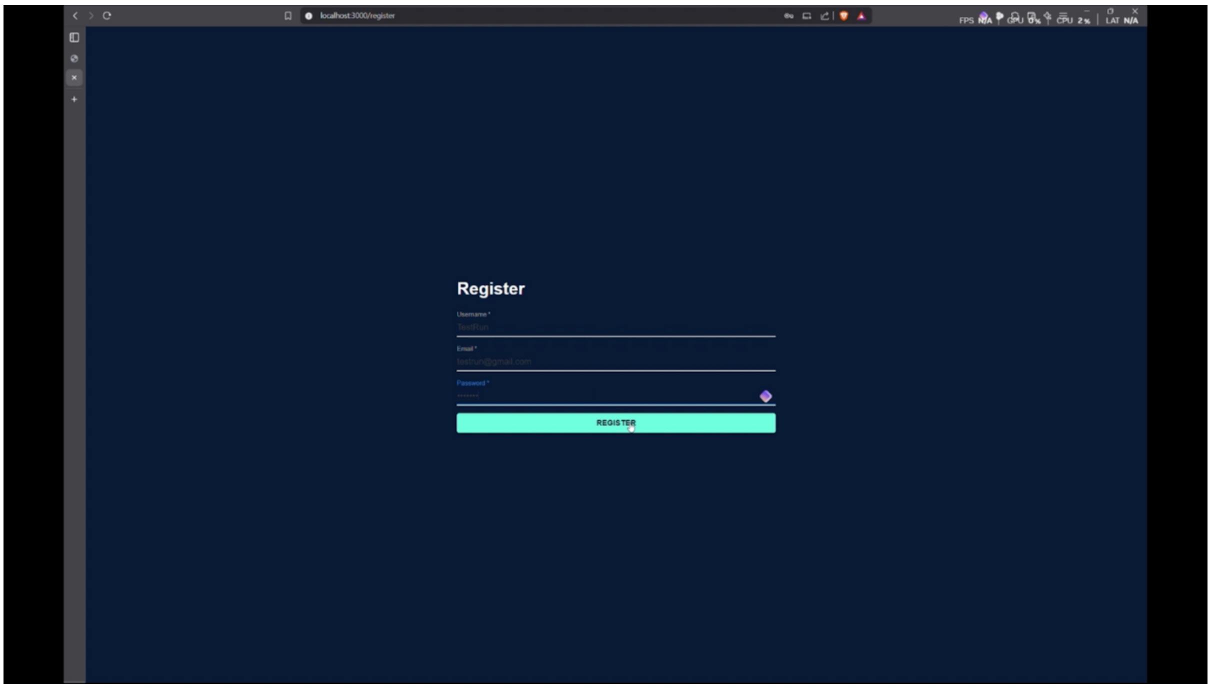


*Figure 2 Register Page*

This displays the Protego registration page. New users are prompted to create an account by providing necessary details. The page features a clean, user-friendly design that emphasizes security and ease of onboarding.

## 7.3 Login Page

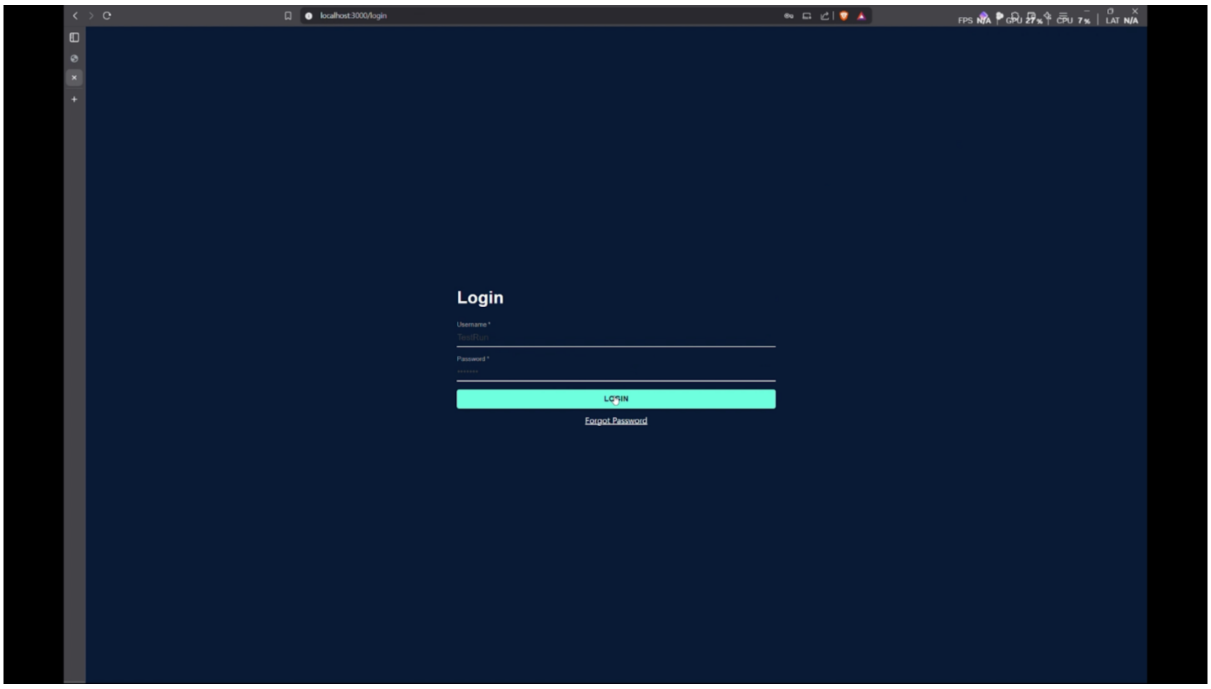
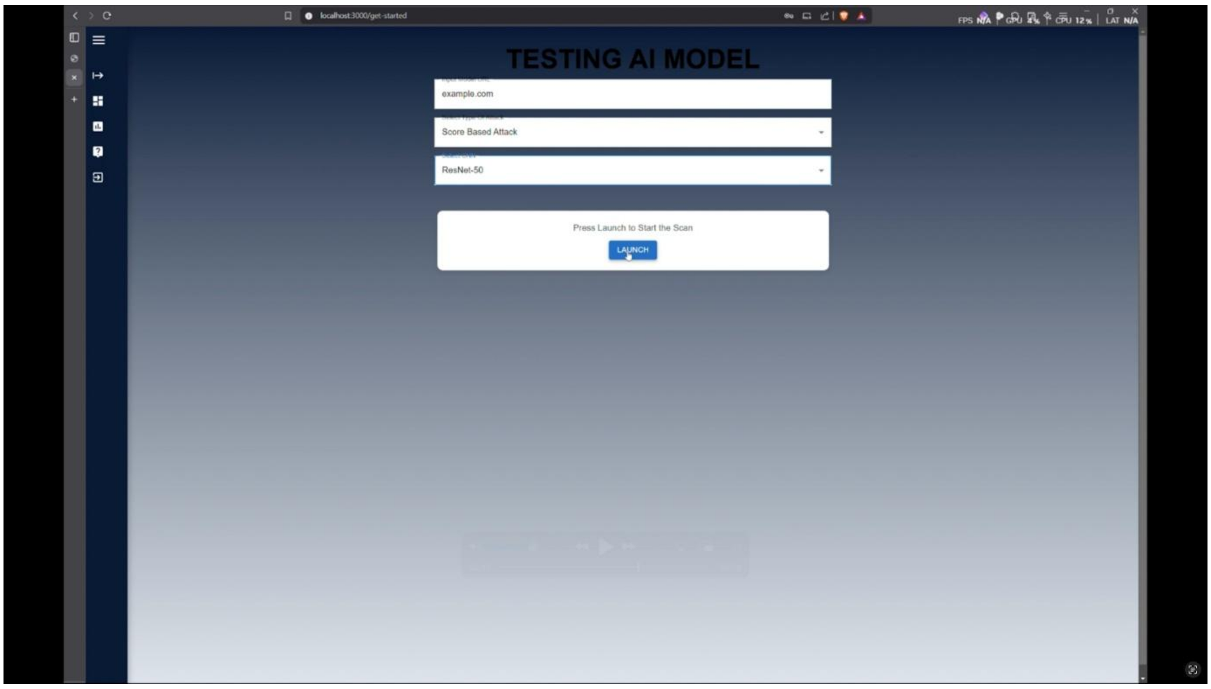


*Figure 3 Login Page*

This shows the login page for Protego. Users enter their credentials to access the system. The interface is straightforward, ensuring secure access for authorized users.

## 7.4 Testing AI Model



example.com

Score Based Attack

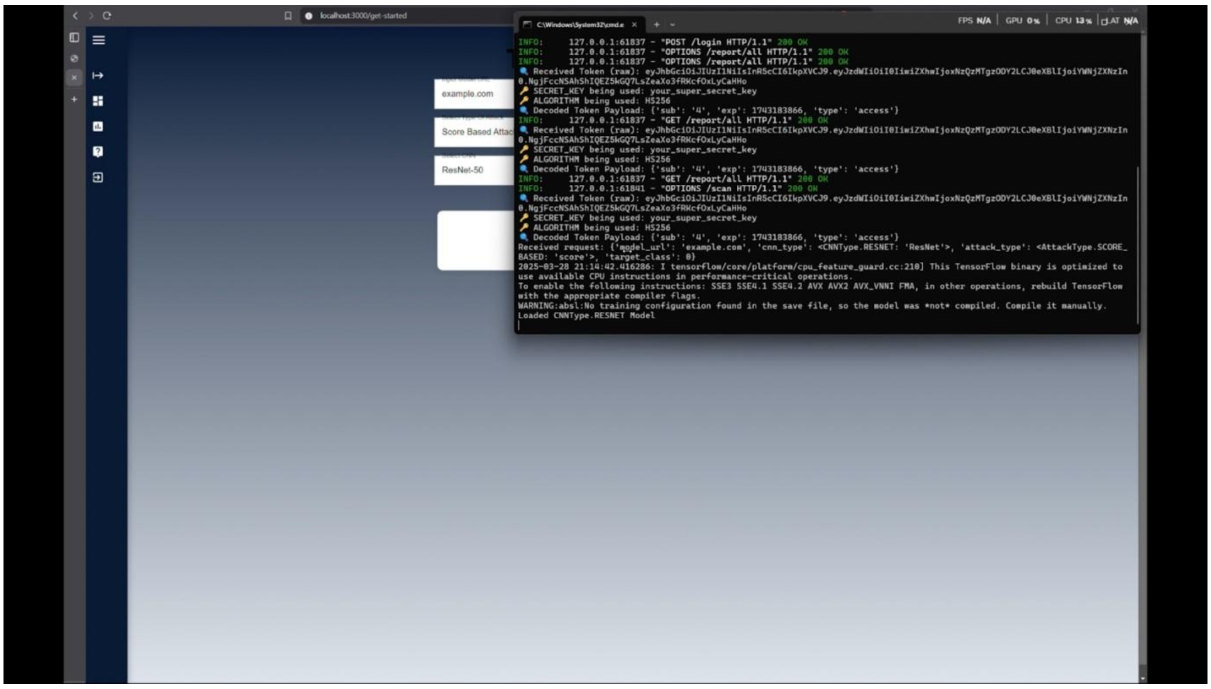
ResNet-50

Press Launch to Start the Scan

*Figure 4 Testing AI Model*

This illustrates the section where a user is testing an AI model. The interface allows users to upload a pretrained CNN model (e.g., ResNet, MobileNet, EfficientNet) for vulnerability scanning, highlighting the system’s focus on model-based assessments.

## 7.5 Attack in Progress

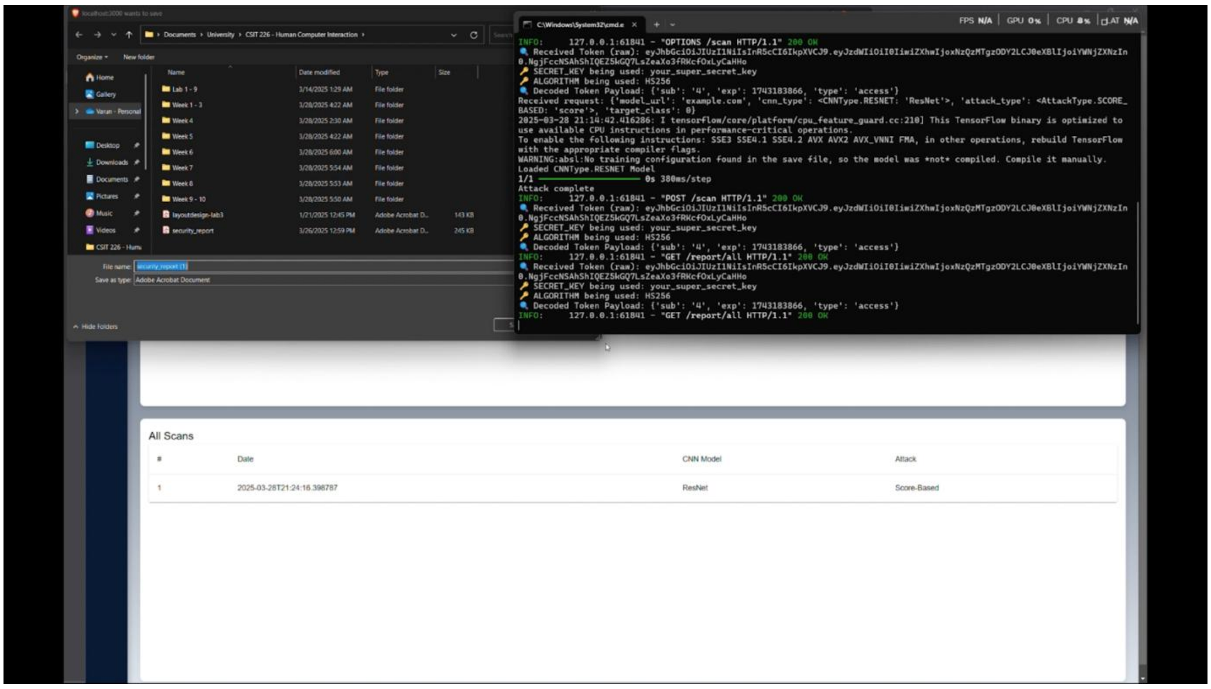

localhost:3000/get-started
example.com
Score Based Attac
ResNet-50
C:\Windows\System32\cmd.e
FPS N/A | GPU 0% | CPU 13% | LAT N/A

```
INFO:     127.0.0.1:61837 - "POST /login HTTP/1.1" 200 OK
INFO:     127.0.0.1:61837 - "OPTIONS /report/all HTTP/1.1" 200 OK
INFO:     127.0.0.1:61837 - "OPTIONS /report/all HTTP/1.1" 200 OK
🔍 Received Token (raw): eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0IiwiZXhwIjoxNzQzMTgzODY2LCJ0eXBlIjoiYWNjZXNzIn
0.NgjFccNSAhShIQEZ5kGQ7LsZeaXo3fRKcfOxLyCaHHo
🔑 SECRET_KEY being used: your_super_secret_key
🔑 ALGORITHM being used: HS256
🔍 Decoded Token Payload: {'sub': '4', 'exp': 1743183866, 'type': 'access'}
INFO:     127.0.0.1:61837 - "GET /report/all HTTP/1.1" 200 OK
🔍 Received Token (raw): eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0IiwiZXhwIjoxNzQzMTgzODY2LCJ0eXBlIjoiYWNjZXNzIn
0.NgjFccNSAhShIQEZ5kGQ7LsZeaXo3fRKcfOxLyCaHHo
🔑 SECRET_KEY being used: your_super_secret_key
🔑 ALGORITHM being used: HS256
🔍 Decoded Token Payload: {'sub': '4', 'exp': 1743183866, 'type': 'access'}
INFO:     127.0.0.1:61837 - "GET /report/all HTTP/1.1" 200 OK
INFO:     127.0.0.1:61841 - "OPTIONS /scan HTTP/1.1" 200 OK
🔍 Received Token (raw): eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0IiwiZXhwIjoxNzQzMTgzODY2LCJ0eXBlIjoiYWNjZXNzIn
0.NgjFccNSAhShIQEZ5kGQ7LsZeaXo3fRKcfOxLyCaHHo
🔑 SECRET_KEY being used: your_super_secret_key
🔑 ALGORITHM being used: HS256
🔍 Decoded Token Payload: {'sub': '4', 'exp': 1743183866, 'type': 'access'}
Received request: {'model_url': 'example.com', 'cnn_type': <CNNType.RESNET: 'ResNet'>, 'attack_type': <AttackType.SCORE_
BASED: 'score'>, 'target_class': 0}
2025-03-28 21:14:42.416286: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to
use available CPU instructions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow
with the appropriate compiler flags.
WARNING:absl:No training configuration found in the save file, so the model was *not* compiled. Compile it manually.
Loaded CNNType.RESNET Model
```

*Figure 5 Attack in Progress*

This is the Protego scanning engine in action. It shows the simulation of model inversion attacks (score-based and boundary-based) on the selected AI model, with visual indicators demonstrating that the attack is actively in progress.

## 7.6 Password-Protected Report (PDF Format)


localhost:3000 wants to save
Documents › University › CSIT 226 - Human Computer Interaction ›
Organize ▾
New folder
Home
Gallery
Varun - Personal
Desktop
Downloads
Documents
Pictures
Music
Videos
CSIT 226 - Hum
Hide Folders
C:\Windows\System32\cmd.e
FPS N/A | GPU 0% | CPU 8% | LAT N/A

| Name | Date modified | Type | Size |
|---|---|---|---|
| Lab 1 - 9 | 3/14/2025 1:29 AM | File folder | |
| Week 1 - 3 | 3/28/2025 4:22 AM | File folder | |
| Week 4 | 3/28/2025 2:30 AM | File folder | |
| Week 5 | 3/28/2025 4:22 AM | File folder | |
| Week 6 | 3/28/2025 6:00 AM | File folder | |
| Week 7 | 3/28/2025 5:54 AM | File folder | |
| Week 8 | 3/28/2025 5:53 AM | File folder | |
| Week 9 - 10 | 3/28/2025 5:50 AM | File folder | |
| layoutdesign-lab3 | 1/21/2025 12:45 PM | Adobe Acrobat D... | 143 KB |
| security_report | 3/26/2025 12:59 PM | Adobe Acrobat D... | 245 KB |

File name: security_report (1)
Save as type: Adobe Acrobat Document

```
INFO:     127.0.0.1:61841 - "OPTIONS /scan HTTP/1.1" 200 OK
🔍 Received Token (raw): eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0IiwiZXhwIjoxNzQzMTgzODY2LCJ0eXBlIjoiYWNjZXNzIn
0.NgjFccNSAhShIQEZ5kGQ7LsZeaXo3fRKcfOxLyCaHHo
🔑 SECRET_KEY being used: your_super_secret_key
🔑 ALGORITHM being used: HS256
🔍 Decoded Token Payload: {'sub': '4', 'exp': 1743183866, 'type': 'access'}
Received request: {'model_url': 'example.com', 'cnn_type': <CNNType.RESNET: 'ResNet'>, 'attack_type': <AttackType.SCORE_
BASED: 'score'>, 'target_class': 0}
2025-03-28 21:14:42.416286: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to
use available CPU instructions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow
with the appropriate compiler flags.
WARNING:absl:No training configuration found in the save file, so the model was *not* compiled. Compile it manually.
Loaded CNNType.RESNET Model
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 380ms/step
Attack complete
INFO:     127.0.0.1:61841 - "POST /scan HTTP/1.1" 200 OK
🔍 Received Token (raw): eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0IiwiZXhwIjoxNzQzMTgzODY2LCJ0eXBlIjoiYWNjZXNzIn
0.NgjFccNSAhShIQEZ5kGQ7LsZeaXo3fRKcfOxLyCaHHo
🔑 SECRET_KEY being used: your_super_secret_key
🔑 ALGORITHM being used: HS256
🔍 Decoded Token Payload: {'sub': '4', 'exp': 1743183866, 'type': 'access'}
INFO:     127.0.0.1:61841 - "GET /report/all HTTP/1.1" 200 OK
🔍 Received Token (raw): eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0IiwiZXhwIjoxNzQzMTgzODY2LCJ0eXBlIjoiYWNjZXNzIn
0.NgjFccNSAhShIQEZ5kGQ7LsZeaXo3fRKcfOxLyCaHHo
🔑 SECRET_KEY being used: your_super_secret_key
🔑 ALGORITHM being used: HS256
🔍 Decoded Token Payload: {'sub': '4', 'exp': 1743183866, 'type': 'access'}
INFO:     127.0.0.1:61841 - "GET /report/all HTTP/1.1" 200 OK
```

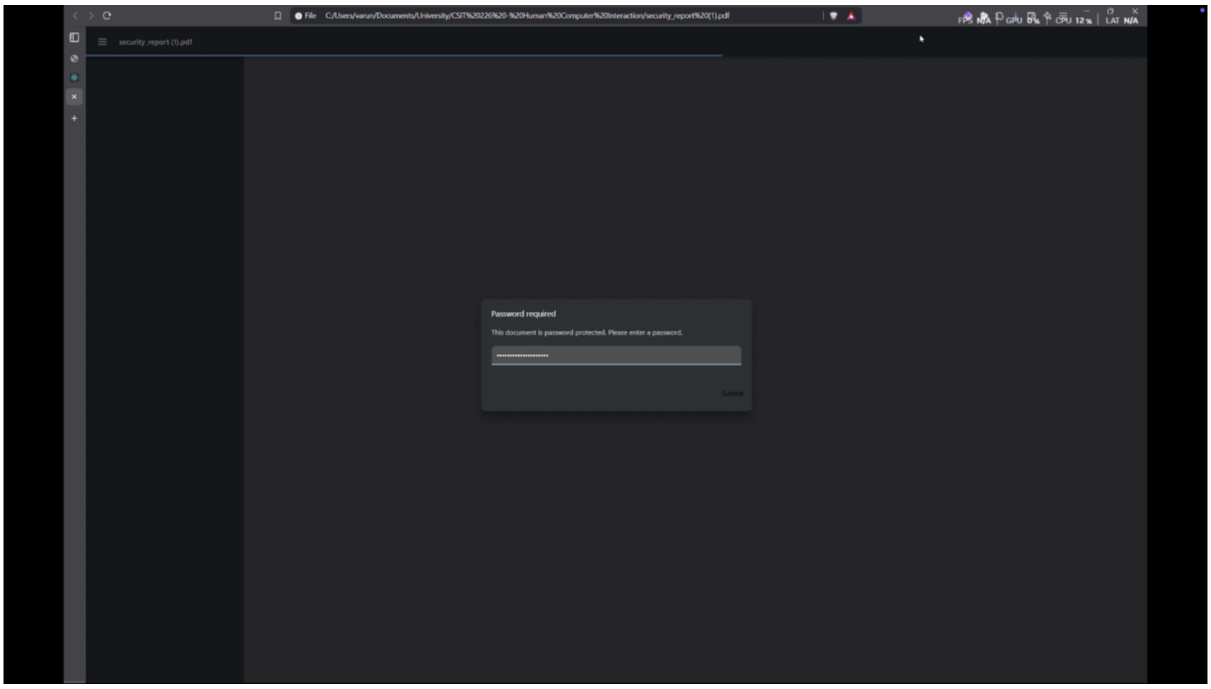
All Scans

| # | Date | CNN Model | Attack |
|---|---|---|---|
| 1 | 2025-03-28T21:24:16.398787 | ResNet | Score-Based |

*Figure 6 Password-Protected Report (PDF Format)*

This displays the generated vulnerability report in PDF format. The report is securely produced and password-protected, ensuring that sensitive findings and recommendations remain confidential.

## 7.7 Report Access Authentication


File C:/Users/varun/Documents/University/CSIT%20226%20-%20Human%20Computer%20Interaction/security_report%20(1).pdf
security_report (1).pdf
FPS N/A GPU 0% CPU 12% LAT N/A

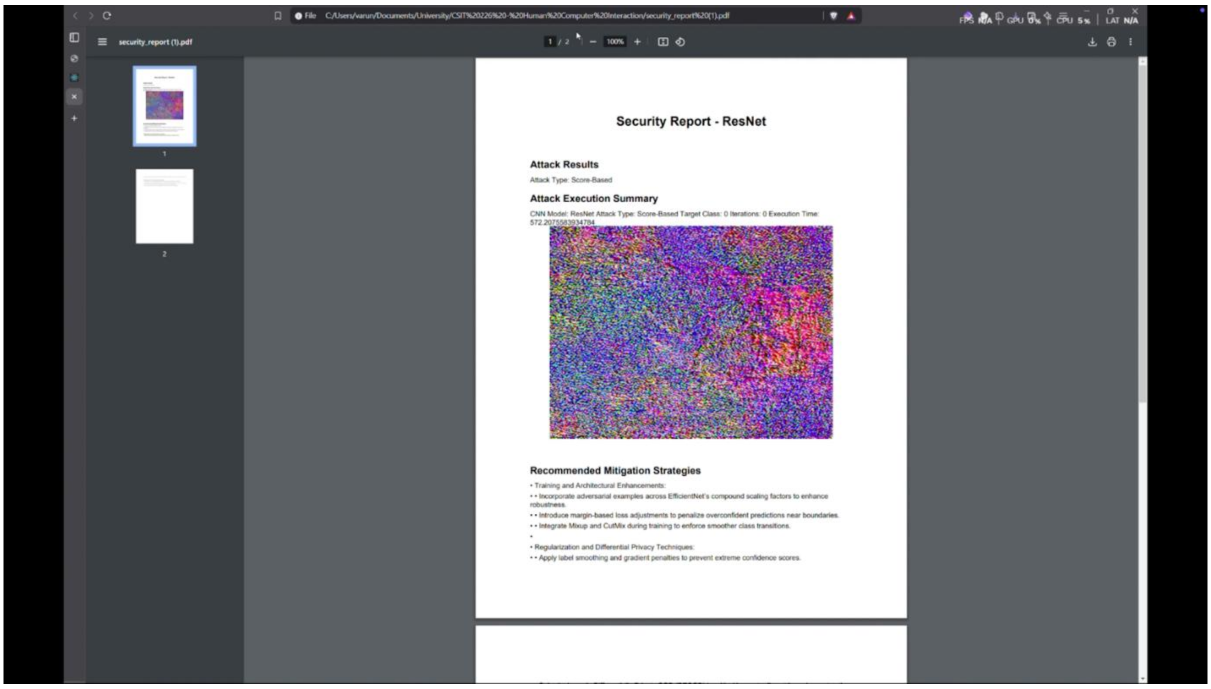
Password required

This document is password protected. Please enter a password.

*Figure 7 Report Access Authentication*

This demonstrates the access control mechanism for the generated report. Users are required to enter a password to view the detailed report, ensuring that only authorized personnel can access sensitive scan results.

## 7.8 Comprehensive Report


C:/Users/varun/Documents/University/CSIT%20226%20-%20Human%20Computer%20Interaction/security_report%20(1).pdf
security_report (1).pdf
1 / 2
100%

**Security Report - ResNet**

**Attack Results**

Attack Type: Score-Based

**Attack Execution Summary**

CNN Model: ResNet Attack Type: Score-Based Target Class: 0 Iterations: 0 Execution Time: 572.2075583934784

**Recommended Mitigation Strategies**

• Training and Architectural Enhancements:

• • Incorporate adversarial examples across EfficientNet's compound scaling factors to enhance robustness.

• • Introduce margin-based loss adjustments to penalize overconfident predictions near boundaries.

• • Integrate Mixup and CutMix during training to enforce smoother class transitions.

•

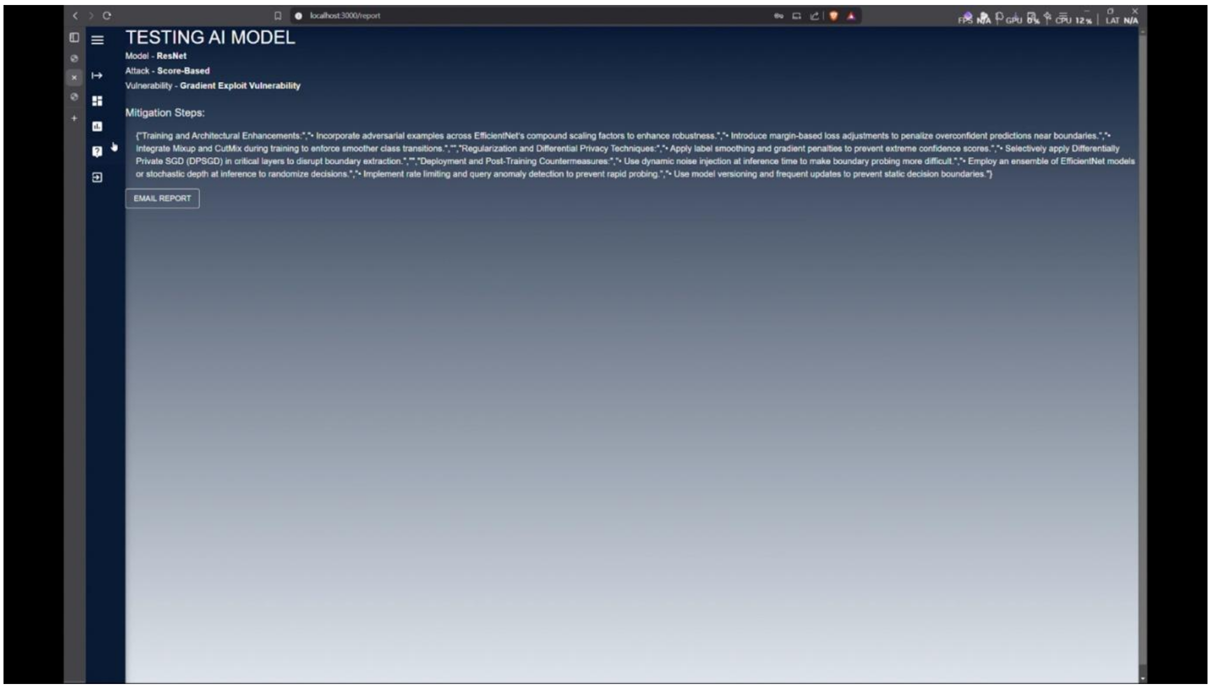
• Regularization and Differential Privacy Techniques:

• • Apply label smoothing and gradient penalties to prevent extreme confidence scores.

*Figure 8 Comprehensive Report*

This shows the actual detailed report generated by Protego. It includes a comprehensive summary of the attack execution, details of the type of model inversion attack performed, and the corresponding mitigation steps recommended to secure the model.

## 7.9 Post Attack Dashboard



TESTING AI MODEL

Model - **ResNet**

Attack - **Score-Based**

Vulnerability - **Gradient Exploit Vulnerability**

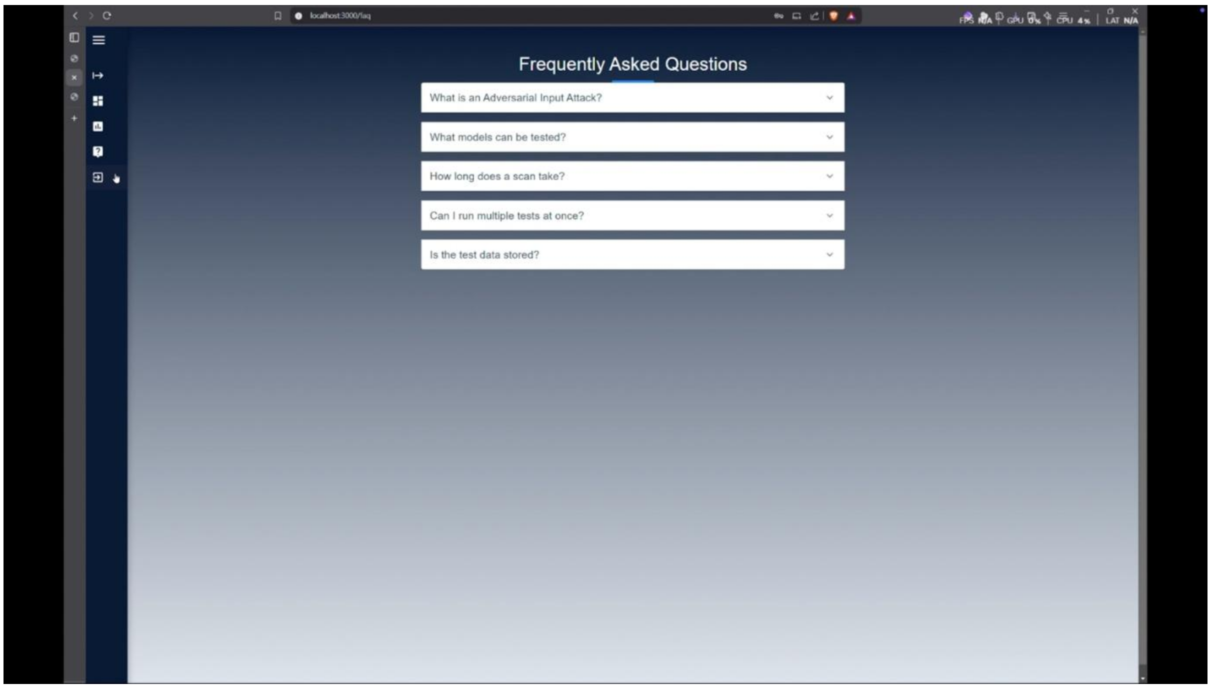
Mitigation Steps:

{"Training and Architectural Enhancements:","• Incorporate adversarial examples across EfficientNet's compound scaling factors to enhance robustness.","• Introduce margin-based loss adjustments to penalize overconfident predictions near boundaries.","• Integrate Mixup and CutMix during training to enforce smoother class transitions.","","Regularization and Differential Privacy Techniques:","• Apply label smoothing and gradient penalties to prevent extreme confidence scores.","• Selectively apply Differentially Private SGD (DPSGD) in critical layers to disrupt boundary extraction.","","Deployment and Post-Training Countermeasures:","• Use dynamic noise injection at inference time to make boundary probing more difficult.","• Employ an ensemble of EfficientNet models or stochastic depth at inference to randomize decisions.","• Implement rate limiting and query anomaly detection to prevent rapid probing.","• Use model versioning and frequent updates to prevent static decision boundaries."}

*Figure 9 Post Attack Dashboard*

This illustrates the Protego dashboard after an attack has been completed. The dashboard displays key information such as model vulnerabilities, executed attack details, and mitigation recommendations. Additionally, it features an option to send the generated report via email for further review and collaboration.

### 7.10 FAQ Page


Frequently Asked Questions

What is an Adversarial Input Attack?

What models can be tested?

How long does a scan take?

Can I run multiple tests at once?

Is the test data stored?

*Figure 10 FAQ Page*

This shows the Frequently Asked Questions (FAQ) page within Protego. It provides users with detailed answers and guidance on common queries related to the scanner's functionality, usage, and troubleshooting.

## 8. References

| Title | Author | Version | Date | Source Location |
|---|---|---|---|---|
| Planning & Feasibility Report | TeamOne | 2.0 | 23/10/2024 | Available upon request from Team One at: ama438@uowmail.edu.au mk085@uowmail.edu.au vmt979@uowmail.edu.au arrk807@uowmail.edu.au amaa959@uowmail.edu.au |

| | | | | |
|---|---|---|---|---|
| Requirement Analysis Document | TeamOne | 2.0 | 06/11/2024 | Available upon request from Team One at the above emails |
| Software Design Document | TeamOne | 2.0 | 27/11/2024 | Available upon request from Team One at the above emails |

**Document Update Notice**
This Test Plan Document has been updated as of **25 March 2025**.