IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

PentSecOps – A Vulnerability Management Dashboard

[1]H. Pooja, [2] T. Karthik Kumar, [3] G. Shiva Rama Krishna,[4] Deepthi Kanithi, [5] N. Nihal Chowdary, [6] G. Aparna

[1,2,3,4,5] Students [6] Associate Professor Computer Science and Engineering (Cyber Security)
Hyderabad Institute of Technology and Management,
Medchal, Hyderabad, Telangana, India

Abstract: The goal of PentSecOps, an AI-powered penetration testing management dashboard, is to centralize and simplify an organization's entire vulnerability management lifecycle. Communication fragmentation across various tools, disorganized vulnerability tracking procedures, gaps in technical-business communication, and scalability constraints in conventional pentesting workflows are just a few of the significant operational issues that the platform tackles. PentSecOps employs a complete three-tier architecture built on a contemporary technology stack that includes PostgreSQL for reliable data persistence, React.js with TypeScript for the frontend interface, and Go (Fiber Framework) for the backend REST API. The system has a complex middleware pipeline for security, logging, and request management, and role-based access control using Paseto v4 authentication tokens.

Keywords: Penetration Testing, Vulnerability Management, Role-Based Access Control, Cybersecurity Operations, Web Application Security.

I. Introduction

Penetration testing teams today face several serious challenges that affect how efficiently they work and the quality of what they deliver. One of the biggest pain points is communication fragmentation. Teams often find themselves jumping between tools like email, Slack, and Teams just to stay in sync. As a result, conversations get scattered, important details are lost, and tracking updates across channels becomes a frustrating task. This constant juggling also adds unnecessary manual work, as teams must piece together scattered discussions to build coherent reports. When vulnerability details are buried in different threads or apps, retrieving crucial remediation guidance at the right moment becomes unnecessarily difficult.
Beyond communication, vulnerability management itself is often chaotic. Many organizations lack a single system to manage the full vulnerability lifecycle—from discovery and assessment all the way through to remediation and verification. Instead, findings end up siloed in static reports stored in PDFs, Excel sheets, or Word files, with no centralized way to track progress. This forces security teams to rely on manual updates, while business stakeholders are left waiting for status reports instead of having access to live, visual dashboards. The lack of unified visibility slows down remediation and makes it harder to establish accountability.

II. LITERATURE SURVEY

Modern vulnerability management systems face critical challenges rooted in fragmentation and poor integration across security lifecycles. Organizations wrestle with fragmented pieces of vulnerability information distributed across spreadsheets, email threads, and a variety of project management tools, which result in enormous coordination overhead and major information silos [1]. The tracking in traditional systems is not adequately integrated between discovery, remediation, and verification stages, nor does it provide role-specific views to administrators, penetration testers, and business stakeholders [1]. Such fragmentation creates an urgent need for centralized platforms that aggregate vulnerability information and display it differently for different stakeholders [1]. The integration of Large Language Models into Vulnerability Management is a promising technological advancement. Analysis of research papers shows that approaches based on LLM using BERT, CodeBERT, and GPT variants offer better vulnerability identification accuracy compared to traditional static analysis tools [2]. These can generate remediation recommendations automatically by learning from historical patches and security documentation; research publications in this domain have increased by 27.8% in the year 2024 alone [2]. However, challenges persist, such as high false-positive rates with complex vulnerabilities and explaining AI-generated findings to non-technical stakeholders, which indicate that optimal solutions combine human expertise with AI-powered analysis [2].

Enterprise vulnerability assessment goes beyond technical detection to include dataset quality, compliance reporting, and cross-domain risk management. Organizations continue to struggle with harmonizing different classification taxonomies such as CVE, CWE, and CVSS, and in translating technical data into business risk metrics for executive decision-making [3]. Standard vulnerability management tools have large functional gaps in enabling collaborative remediation workflows, tracking historical intelligence to do trend analysis, and generating compliance reports according to ISO 27001, NIST, and PCI-DSS frameworks [3]. This research validates the need for solutions that provide structured classification with CVSS scoring, SLA tracking through MTTD and MTTR metrics, domain-wise security analysis, and architectural extensibility for evolving compliance requirements [3].

III. EXISTING SYSTEM

Current vulnerability management practices rely heavily on fragmented tools and manual processes that create significant operational inefficiencies. Organizations typically utilize disparate systems including spreadsheets, email communications, and standalone project management tools to track vulnerabilities, resulting in isolated data silos and inconsistent classification methodologies [1]. Traditional vulnerability scanners identify security issues but lack integrated workflows for collaborative remediation, stakeholder-appropriate reporting, and historical trend analysis [3]. These systems struggle to bridge the communication gap between technical teams and business stakeholders, as they cannot effectively translate technical vulnerability data into actionable business risk metrics [3]. Furthermore, existing tools provide limited support for compliance reporting aligned with regulatory frameworks such as ISO 27001, NIST, and PCI-DSS, while failing to maintain comprehensive audit trails and SLA tracking mechanisms [3]. This fragmented approach leads to delayed remediation, inconsistent prioritization, and inadequate visibility into organizational security posture across different domains and stakeholder groups [1].

IV. PROPOSED SYSTEM

PentSecOps addresses these complex challenges with an all-encompassing platform that reinvents the management of pentesting from first principles. Instead of general-purpose tool adaptations, the system implements features purposefully built for security testing workflows. The platform recognizes that different stakeholders require fundamentally different interfaces and capabilities, and as such, has implemented sophisticated role-based access control to tailor functionality and information presentation to each user type. Innovation is first and foremost about architectural decisions. Among other things, this means making choices that balance performance, security, and maintainability. Go with the Fiber framework has been selected for backend services because of its exceptional request handling performance with very minimal resource consumption compared to traditional web frameworks. Its middleware architecture facilitates sophisticated request processing pipelines, implementing authentication, authorization, logging, rate limiting, and error handling as composable layers. This allows each cross-cutting concern to be implemented once and applied consistently across all API endpoints. Client-side implementation using React.js together with TypeScript provides type-safe component development that catches errors during compilation, rather than runtime. Component-based architecture fosters code reuse while keeping presentation logic clearly separated from business rules. Integration with shadcn/ui components and Tailwind CSS facilitates fast development of modern, responsive interfaces that assure great user experiences across devices and screen sizes. The client-side architecture implements protected routes that enforce role-based access control, ensuring users can only access interfaces appropriate to their authorization level. Careful consideration of performance, data integrity, and query efficiency is reflected in database design. PostgreSQL provides enterprise-grade reliability with sophisticated indexing capabilities that ensure queries remain performant as data volumes grow. Comprehensive foreign key 3 relationships are implemented in the schema; these maintain referential integrity automatically, preventing orphaned records and ensuring that data remains consistent. Efficient filtering and sorting operations through strategic index placement on frequently queried columns enable optimized join paths to support even complex analytical queries without performance degradation. Authentication and authorization mechanisms implement industry best practices through the use of Paseto v4 tokens, which offer strong cryptographic guarantees without vulnerabilities associated with older token formats. A dual-token strategy utilizing both an access token and a refresh token effectively balances security against user experience. It allows access tokens, which are short-lived, to minimize exposure while refresh tokens enable seamless session continuation. Token-based authentication eliminates the need for session storage on the server-side, allowing for true stateless operations and making horizontal scaling much easier.

4.1 How we had designed the PentSecOps

The PentSecOps technology stack is designed to optimize performance, security, and scalability for penetration testing management. The backend is built with Go using the Fiber framework, which delivers fast, concurrent request handling and simplifies deployment by compiling to single-binary executables. Fiber's middleware allows for consistent implementation of cross-cutting concerns like authentication, logging, and error handling. The frontend leverages React.js with TypeScript for type-safe, modular UI development, supporting reusable components, strong typing, and maintainable code. PostgreSQL is chosen for its enterprise-grade reliability, ACID compliance, and advanced query capabilities, ensuring data integrity and efficient complex operations. For authentication, Paseto v4 tokens are used to provide robust, modern cryptographic protection while supporting stateless, scalable APIs through a dual-token approach. The architecture follows clean separation between presentation, application, and data layers, enabling independent evolution, easier testing, and deployment flexibility. RESTful APIs enable predictable integration, while layered security—including role-based access—ensures comprehensive protection across the system, supporting secure, real-time collaboration for multiple distinct user roles.

4.1.1 Advantages of Proposed System

1. Vulnerability management that is centralized
There is no more dispersed data across emails, spreadsheets, and chat tools because all pentest activities, findings, reports, and communications are kept on one platform.
2. Access Control Based on Roles
Different dashboards for stakeholders, administrators, and pentesters guarantee that each user sees only pertinent data, enhancing security and lowering information overload.
3. Simplified Process
Workflows for structured task assignment, status monitoring, and report submission decrease manual coordination and boost team output.
4. Visibility of Real-Time Progress
Through dashboards, stakeholders can keep an eye on project timelines, remediation progress, and vulnerability status without having to ask for manual updates.

4.1.2 Impact of Proposed System

1. Quicker Fixing of Vulnerabilities
The window of exposure to security risks is reduced by centralized tracking and clear ownership, which shorten remediation cycles from weeks to days.
2. Enhanced Pentester Efficiency
Pentesters increase overall productivity by 30–40% by devoting more time to actual security testing as opposed to administrative duties like report formatting and status updates.
3. Better Decision-Making
Stakeholders can make well-informed decisions about security investments and risk acceptance with the help of real-time dashboards and analytics.
4. Improved Teamwork
By removing information silos, a unified platform makes it possible for remote teams to work together efficiently and have a common understanding of tasks and results.
5. Shorter Communication Divides
By bridging the gap between technical security teams and business stakeholders, role-specific interfaces guarantee that everyone is aware of the priorities and status of the project.

V. SYSTEM ARCHITECTURE

The PentSecOps design demonstrates deliberate thought regarding the operational realities of penetration testing workflows. Instead of providing an arbitrary outline, we examined how security teams actually work and designed the system to support and to augment naturally occurring workflows. This section will discuss both the decisions we made regarding the architecture, and the rationale behind such decisions. A Design Principles There were four design principles that guided our design decisions: Role-Appropriate Information Access: Different actors require different views of security data. Penetration testers need to see technical specifics about vulnerabilities, exploitation techniques, and what those vulnerabilities could mean for remediation. Administrators need to see the operational oversight of projects related to resource allocation and project progress. A business stakeholder is interested in a high-level overview of security posture without the technical minutiae. The underpinnings of our design is that each actor sees exactly what they need to see. Workflow-Centric Architecture: Rather than designing the system around the notion of a simple database of vulnerabilities, we drew from the actual workflow of penetration testing operations. Tasks flow from an administrator to a tester. Findings flow from recognition to assessment to remediation. Reports flow from draft status to final status. Each aspect of the system references these natural workflows.

5.1.1 Implementation

PentSecOps is based on a modular, layered architecture: the frontend is developed separately from the backend, with integration through secured REST APIs. The backend is divided into business logic, HTTP handling, and persistence in separate packages. The frontend organizes components and contexts by user role, ensuring separation of concerns and maintainability across views for Admin, Pentester, and Stakeholder users.

5.1.2 Implementation Flow

Users hit the React frontend, where role-based route protection starts the authentication process.
The Go backend receives login requests, checks the credentials for validity, and returns secure Paseto tokens to manage sessions.
Role-based Dashboards fetches the protected data through APIs, passing authentication tokens at every request.
For error handling, rate limiting, input validation, and role-based authorization—where users are allowed to access only resources that the system permits. Business logic is decoupled from HTTP routes, and each dashboard's data flows through use cases to the repositories, which are mapped to PostgreSQL for robust storage and integrity. Report submission, vulnerability management, project/task assignment, notification broadcasts, and admin-review workflows are implemented as independent, reusable modules.

The database schema enforces strict constraints and relationships between users, projects, vulnerabilities, reports, and assets, ensuring that the flow of data is accurate and audited.
Security Measures Modern cryptography is used for authentication, including Paseto v4 and argon2id passwords. Authorization enforces roles and resources. Inputs are validated to prevent injection. All traffic uses HTTPS. Sessions rotate tokens and monitor for password changes or lockouts. Audit logs track all actions. API requests are rate-limited. Robust error handling protects user privacy and shields internal operations. This design creates a secure, scalable foundation that facilitates real-time collaboration among multiple roles and enables the platform to have further AI-driven enhancements, along with research-ready extensibility.

5.2 Backend Implementation (Go + Fiber)

5.2.1 Structure
Clean architecture with:

- **Entities** (domain models)
- **Use cases** (business logic)
- **Adapters** (HTTP handlers, DB repositories)
- **Infrastructure** (database connection)
- **Auth, Logger, Middleware** utilities

5.2.2 Startup Flow

1. Initialize logger
2. Load configurations
3. Connect to PostgreSQL
4. Run migrations
5. Setup Paseto auth keys
6. Initialize repositories + use cases
7. Create handlers + middleware
8. Register routes
9. Start Fiber server with graceful shutdown

5.2.3 Key Features

- **Authentication**
  - Paseto v4 tokens (Access: 15 min, Refresh: 7 days)
  - Login attempt tracking + account lockout
  - Forced password change on first login
- **User Management**
  - Admin creates & manages users, roles, status
  - Secure password email delivery
- **Projects & Tasks**
  - Project setup, pentester assignment, phases, status updates
- **Vulnerabilities**
  - CVSS scores, attachments, lifecycle tracking
- **Reports**
  - Upload, review workflow, admin approval

5.2.4 Security Measures

- Middleware for recovery, CORS, rate-limits
- Role-based authorization
- Token validation + context injection
- Parameterized queries (SQL injection prevention)
- Strict input validation + request size limits

5.3 Frontend Implementation (React + TypeScript)

5.3.1 Structure

- **Dashboards** for Admin, Pentester, Stakeholder
- **AuthContext** for global auth state
- **ProtectedRoute** for role-based access

5.3.2 Features

- Login → Token stored in memory (not localStorage)
- Role-based auto-redirect after login
- Unauthorized access redirect to login page
- Dashboards:
  - Admin: User mgmt, project assignment, vulnerability overview
  - Pentester: Tasks, vulnerability submission, report upload
  - Stakeholder: Remediation progress, approved reports

5.4 Database Implementation (PostgreSQL)

5.4.1 Schema

- 13 tables including **users, projects, tasks, vulnerabilities, reports, notifications, activity_logs, refresh_tokens** etc.

5.4.2 Constraints & Optimization

- Foreign keys for consistency
- Unique + check constraints for role/status
- Indexes for performance

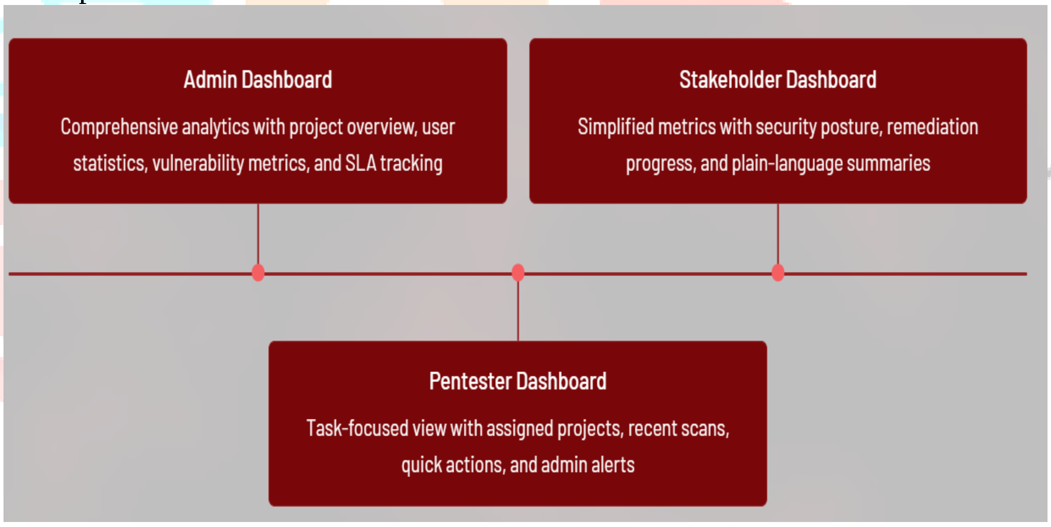

Admin Dashboard
Comprehensive analytics with project overview, user statistics, vulnerability metrics, and SLA tracking
Stakeholder Dashboard
Simplified metrics with security posture, remediation progress, and plain-language summaries
Pentester Dashboard
Task-focused view with assigned projects, recent scans, quick actions, and admin alerts

Fig.1 Implementation overflow

VI. WORKING MODEL OF THE PROJECT

To set up and run the PentSecOps platform locally, follow these steps for a smooth development workflow:

STEP - 1. Start PostgreSQL and Create Database

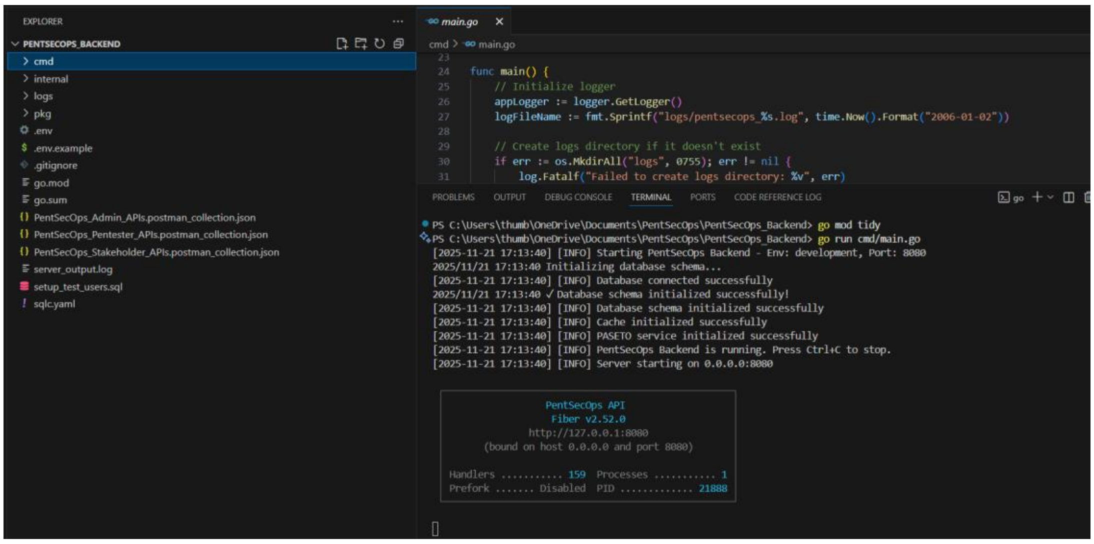
First, launch your PostgreSQL service and create the required database:

```
sql >  CREATE DATABASE pentsecops;
```

STEP - 2. Run Backend Services

Install all dependencies and start the backend API server:

```
bash
go mod tidy            # Install Go module dependencies
go run cmd/main.go     # Start the backend server
```



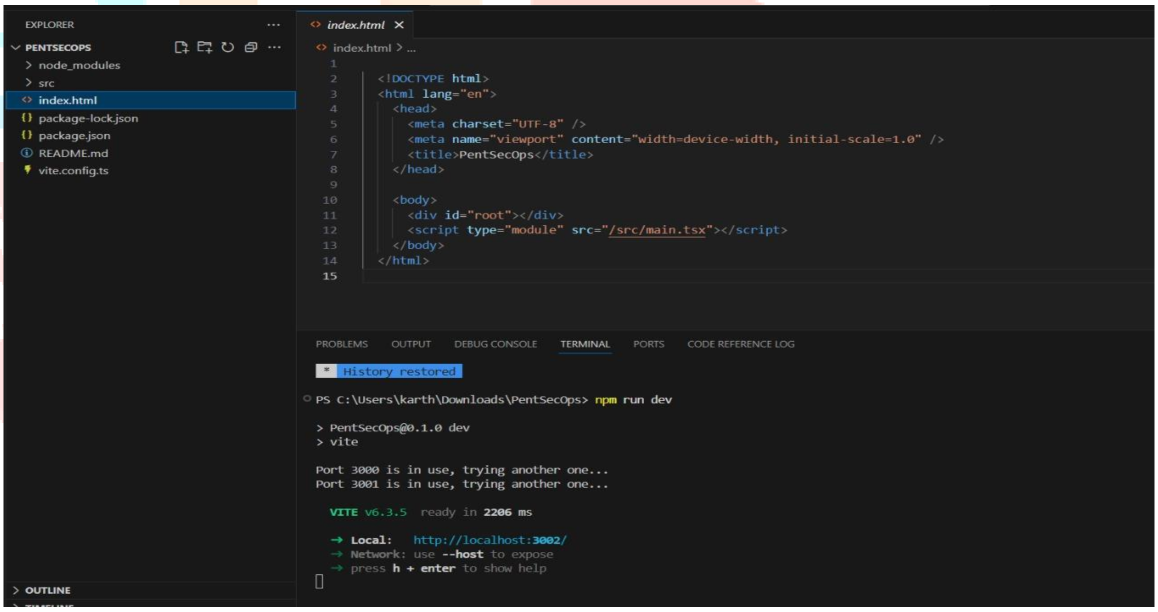
```go
func main() {
    // Initialize logger
    appLogger := logger.GetLogger()
    logFileName := fmt.Sprintf("logs/pentsecops_%s.log", time.Now().Format("2006-01-02"))

    // Create logs directory if it doesn't exist
    if err := os.MkdirAll("logs", 0755); err != nil {
        log.Fatalf("Failed to create logs directory: %v", err)
```

```
PS C:\Users\thumb\OneDrive\Documents\PentSecOps\PentSecOps_Backend> go mod tidy
PS C:\Users\thumb\OneDrive\Documents\PentSecOps\PentSecOps_Backend> go run cmd/main.go
[2025-11-21 17:13:40] [INFO] Starting PentSecOps Backend - Env: development, Port: 8080
2025/11/21 17:13:40 Initializing database schema...
[2025-11-21 17:13:40] [INFO] Database connected successfully
2025/11/21 17:13:40 ✓ Database schema initialized successfully!
[2025-11-21 17:13:40] [INFO] Database schema initialized successfully
[2025-11-21 17:13:40] [INFO] Cache initialized successfully
[2025-11-21 17:13:40] [INFO] PASETO service initialized successfully
[2025-11-21 17:13:40] [INFO] PentSecOps Backend is running. Press Ctrl+C to stop.
[2025-11-21 17:13:40] [INFO] Server starting on 0.0.0.0:8080

                  PentSecOps API
                  Fiber v2.52.0
               http://127.0.0.1:8080
       (bound on host 0.0.0.0 and port 8080)

 Handlers ........... 159  Processes ........... 1
 Prefork ....... Disabled  PID ............. 21888
```

Fig 2. Backend Server application

STEP - 3. Run Frontend Client

Install necessary packages and start the frontend application:
npm install # Install Node.js dependencies
npm run dev # Launch the frontend in development mode



```html

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>PentSecOps</title>
  </head>

  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.tsx"></script>
  </body>
</html>

```

```
 *  History restored

PS C:\Users\karth\Downloads\PentSecOps> npm run dev

> PentSecOps@0.1.0 dev
> vite

Port 3000 is in use, trying another one...
Port 3001 is in use, trying another one...

  VITE v6.3.5  ready in 2206 ms

  →  Local:   http://localhost:3002/
  →  Network: use --host to expose
  →  press h + enter to show help
```

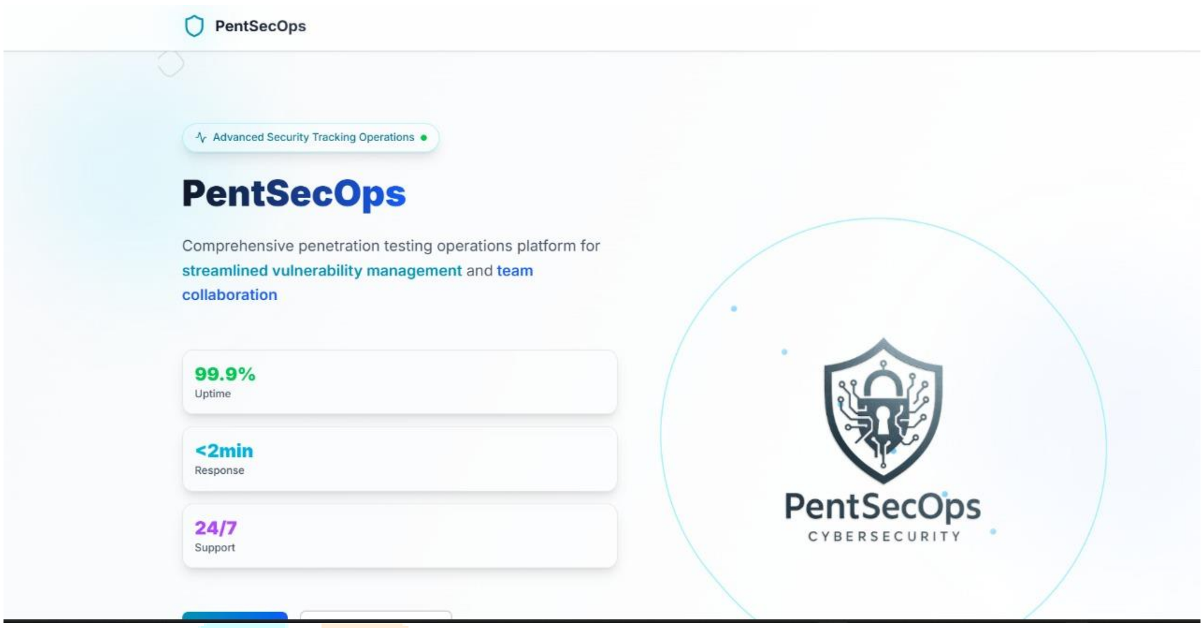
Fig 3. Run frontend application

This setup makes the backend (Go + Fiber) and frontend (React + TypeScript) communicate through secure REST APIs, with PostgreSQL as the persistent storage layer—creating a complete, modular penetration testing management ecosystem.

VII. Result & Conclusion

7.1 Results

PentSecOps successfully addresses the critical operational challenges facing modern penetration testing teams by providing a unified, role-based platform that streamlines vulnerability management workflows. The implementation demonstrates that purpose-built tools tailored to specific domain requirements deliver substantially better outcomes than adapting general-purpose solutions. The platform centralizes all penetration testing activities within a secure, auditable system eliminating the communication fragmentation and vulnerability tracking chaos that previously hindered efficiency. Role-specific interfaces ensure each user type accesses exactly the information and functionality relevant to their responsibilities without overwhelming complexity.


PentSecOps

Advanced Security Tracking Operations

PentSecOps

Comprehensive penetration testing operations platform for **streamlined vulnerability management** and **team collaboration**

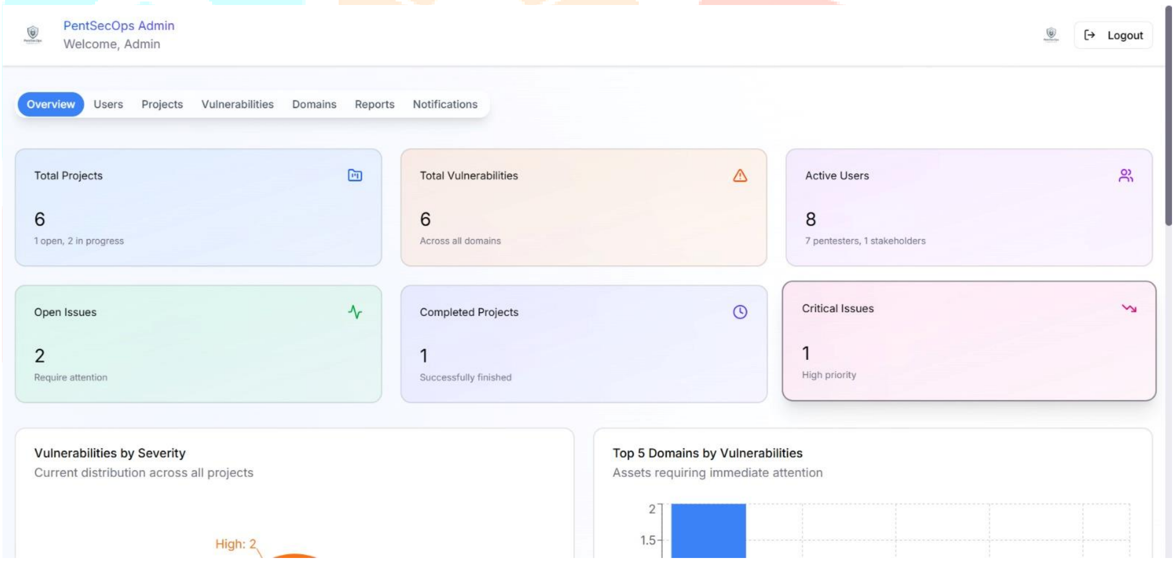
99.9%
Uptime

<2min
Response

24/7
Support

PentSecOps
CYBERSECURITY

Fig 4. PentSecOps Home Page


PentSecOps Admin
Welcome, Admin

Logout

Overview Users Projects Vulnerabilities Domains Reports Notifications

Total Projects
6
1 open, 2 in progress

Total Vulnerabilities
6
Across all domains

Active Users
8
7 pentesters, 1 stakeholders

Open Issues
2
Require attention

Completed Projects
1
Successfully finished

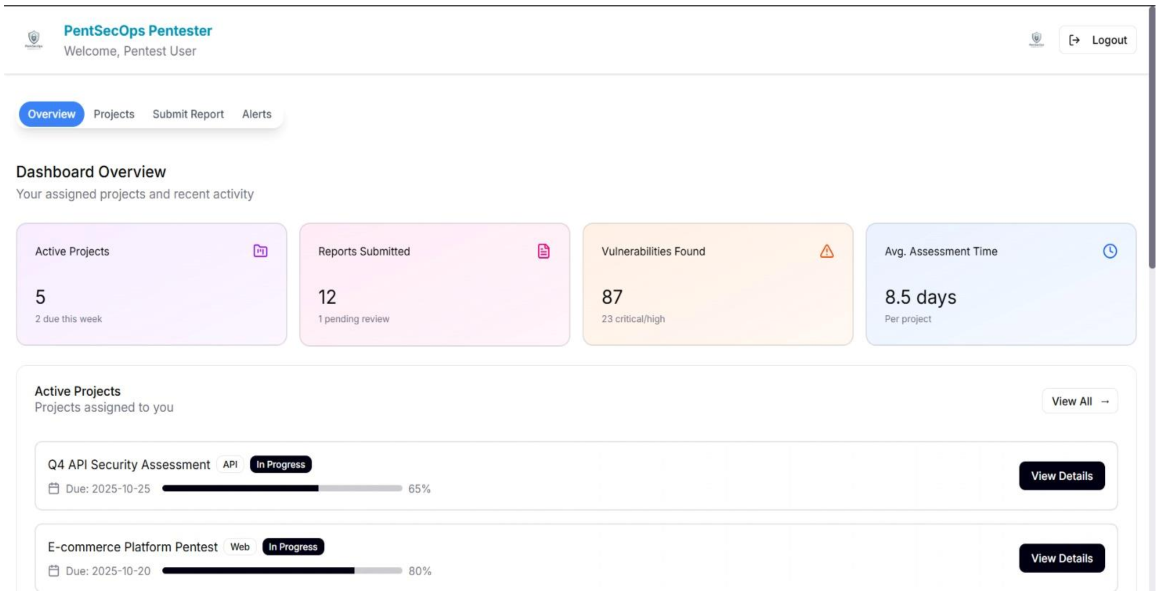
Critical Issues
1
High priority

Vulnerabilities by Severity
Current distribution across all projects

High: 2

Top 5 Domains by Vulnerabilities
Assets requiring immediate attention

Fig 5. PentSecOps Admin Dashboard


PentSecOps Pentester
Welcome, Pentest User
Logout

Overview Projects Submit Report Alerts

Dashboard Overview
Your assigned projects and recent activity

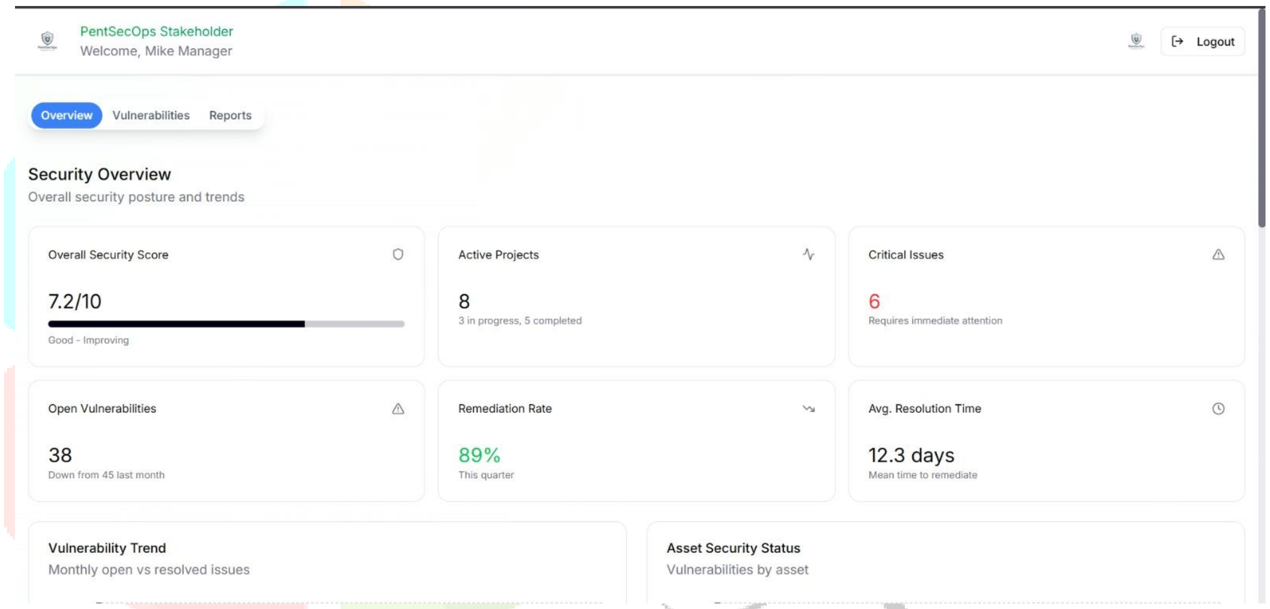
Active Projects 5 — 2 due this week
Reports Submitted 12 — 1 pending review
Vulnerabilities Found 87 — 23 critical/high
Avg. Assessment Time 8.5 days — Per project

Active Projects
Projects assigned to you
View All →

Q4 API Security Assessment API In Progress
Due: 2025-10-25 65%
View Details

E-commerce Platform Pentest Web In Progress
Due: 2025-10-20 80%
View Details

Fig 6. PentSecOps Pentester Dashboard


PentSecOps Stakeholder
Welcome, Mike Manager
Logout

Overview Vulnerabilities Reports

Security Overview
Overall security posture and trends

Overall Security Score 7.2/10 — Good - Improving
Active Projects 8 — 3 in progress, 5 completed
Critical Issues 6 — Requires immediate attention
Open Vulnerabilities 38 — Down from 45 last month
Remediation Rate 89% — This quarter
Avg. Resolution Time 12.3 days — Mean time to remediate

Vulnerability Trend
Monthly open vs resolved issues

Asset Security Status
Vulnerabilities by asset

Fig 7. PentSecOps Stakeholder Dashboard

7.2 Conclusion

In Conclusion, By offering a centralized, secure, and role-based platform for managing vulnerabilities and workflows, PentSecOps simplifies penetration testing operations. It guarantees that users only access what they require, lowers communication gaps, and enhances accountability. The system, which was developed using Go, React, and PostgreSQL, offers quick performance along with robust security features like enforced password policies and audit logging. Easy maintenance and future scalability are made possible by its clear architecture and thorough testing. The platform helps organizations make better decisions and improves visibility into security posture. All things considered, PentSecOps shows how domain-specific tools can greatly increase cybersecurity teams' efficacy and efficiency.

VIII. ACKNOWLEDGMENT

References

[1] "Vulnerability Management Practices and Challenges in Modern Systems," *IEEE Symposium on Security and Privacy 2024*, DOI: 10.1109/SP54263.2024.00123
[2] "Large Language Models for Automated Vulnerability Detection and Remediation," *arXiv preprint*, 2024, DOI: arXiv:2404.02525
[3] "Enterprise Vulnerability Assessment and Compliance Reporting," *International Journal of Information Security*, 2024, DOI: 10.1007/s10207-024-00888-y
[4] IEEE Papers on Vulnerability Management Systems (2021-2024) - Academic research on vulnerability tracking, remediation prioritization, and security metrics
[5] "The Art of Software Security Assessment" by Mark Dowd, John McDonald, Justin Schuh - Comprehensive vulnerability identification and analysis
[6] "Security Metrics: Replacing Fear, Uncertainty, and Doubt" by Andrew Jaquith - Quantitative security measurement and reporting
[7] "Tribe of Hackers" by Marcus J. Carey and Jennifer Jin - Industry insights from security professionals on effective security operations