TASK-3 MERN Stack Internship Tool Ideas (Cybersecurity + Utility)

Created By : Gaurav Gawade - 2036

Secure Auth Starter Kit(MERN template)

Introduction: Building a Production-Ready Gateway

The **Secure Auth Starter Kit** is a comprehensive MERN (MongoDB, Express, React, Node.js) template designed to establish a secure, scalable, and production-ready authentication framework . In modern web development, authentication is more than just a login form; it is a critical layer of defense that must balance rigorous security protocols with a seamless user experience. This project serves as a technical blueprint for implementing industry-standard practices, such as **JWT-based authentication** and **Role-Based Access Control (RBAC)**, to protect sensitive data in real-world applications.

Project Objectives

The primary objective of this project is to design and implement a **secure, scalable, and production-ready authentication system** using modern web technologies. The project focuses on:

- Implementing **JWT-based authentication** with **short-lived access tokens**
- Enhancing security using **refresh tokens stored in HttpOnly cookies**
- Supporting **automatic token renewal** without user interruption
- Enforcing **Role-Based Access Control (RBAC)** for user and admin authorization
- Integrating a **React frontend with Axios interceptors** for seamless authentication handling
- Demonstrating industry-level authentication practices suitable for real-world applications

Tools and Technologies Used

Backend

- **Node.js** – JavaScript runtime
- **Express.js** – Backend framework
- **MongoDB** – Database
- **Mongoose** – ODM for MongoDB
- **JWT (jsonwebtoken)** – Token-based authentication
- **bcryptjs** – Password hashing
- **cookie-parser** – Handling HttpOnly cookies
- **express-rate-limit** – Brute-force protection
- **helmet** – Security headers
- **dotenv** – Environment variable management

Frontend

- **React.js (Vite)** – User interface
- **Axios** – HTTP client
- **Axios Interceptors** – Automatic token refresh handling
- **React Router DOM** – Client-side routing

Project Working Mechanism

Authentication Flow

1. User logs in with valid credentials.
2. Backend generates:
    - **Access Token** (15 seconds expiry)
    - **Refresh Token** (7 days expiry)
3. Access token is stored in **LocalStorage**.
4. Refresh token is stored as an **HttpOnly cookie** (not accessible via JavaScript).

Protected API Access

- Every API request includes the access token in the Authorization header.
- Backend verifies the token before allowing access.

Automatic Token Refresh

- When the access token expires:
  - Backend returns **401 Unauthorized**
  - Axios interceptor automatically calls /api/auth/refresh
  - Backend validates the refresh token and issues a new access token
  - The original request is retried seamlessly

Role-Based Access Control

- Routes are protected based on user roles:
  - user → general protected routes
  - admin → admin-only routes

Key Findings and Insights

- **Refresh token strategy significantly improves security** by limiting access token lifetime.
- Storing refresh tokens in **HttpOnly cookies prevents XSS attacks**.
- Axios interceptors provide a **smooth user experience** by handling token refresh automatically.
- RBAC ensures **controlled access** to sensitive routes.
- Short-lived access tokens reduce the risk of token misuse.
- The system closely follows **industry-standard authentication patterns**.

Challenges Faced

- Managing token expiration and refresh without user logout
- Handling 401 Unauthorized responses correctly
- Ensuring cookies are sent securely using CORS credentials
- Debugging multiple requests triggered by React StrictMode
- Synchronizing frontend token storage with backend authentication logic

Future Recommendations

- Implement **refresh token rotation** for higher security
- Store refresh tokens in the database to support token revocation
- Add **multi-device login management**
- Integrate **OAuth (Google/GitHub login)**
- Deploy using HTTPS for secure cookie handling
- Add **audit logs and session tracking**
- Implement **unit and integration testing**

Project Significance

This project demonstrates a **complete, real-world authentication system** suitable for enterprise applications. It highlights strong security practices, smooth user experience, and scalable architecture. The integration of AI assistance further showcases modern development workflows where human expertise and AI collaboration result in efficient and robust software solutions.

Technical Stack

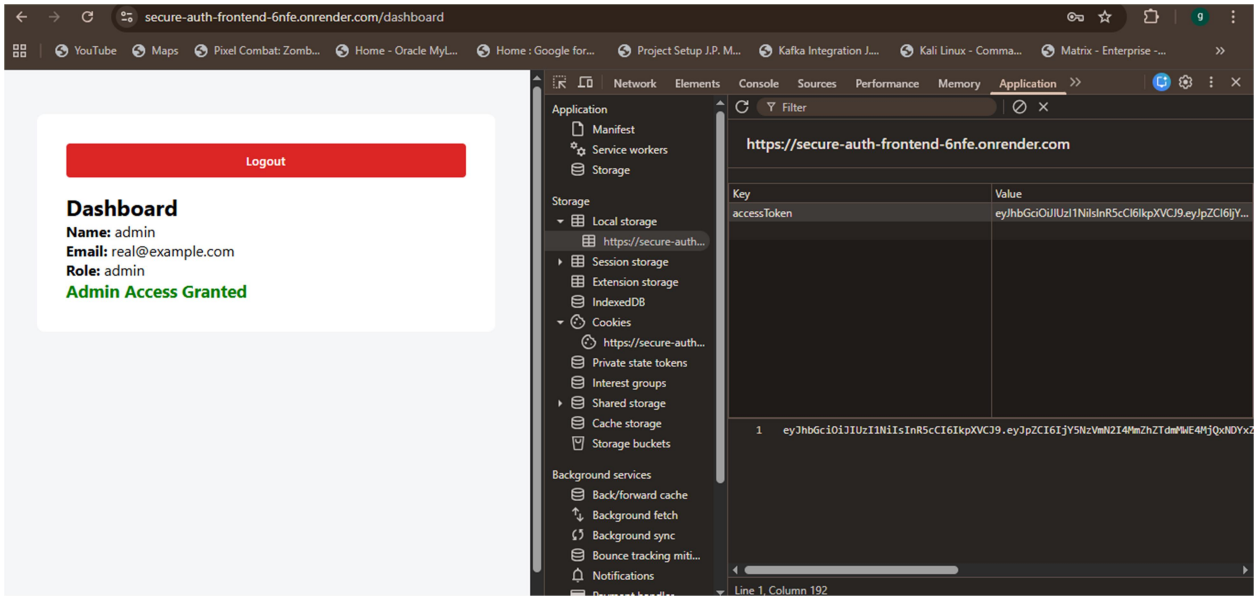
The system is built on a decoupled architecture using industry-standard tools for both the server and client sides.

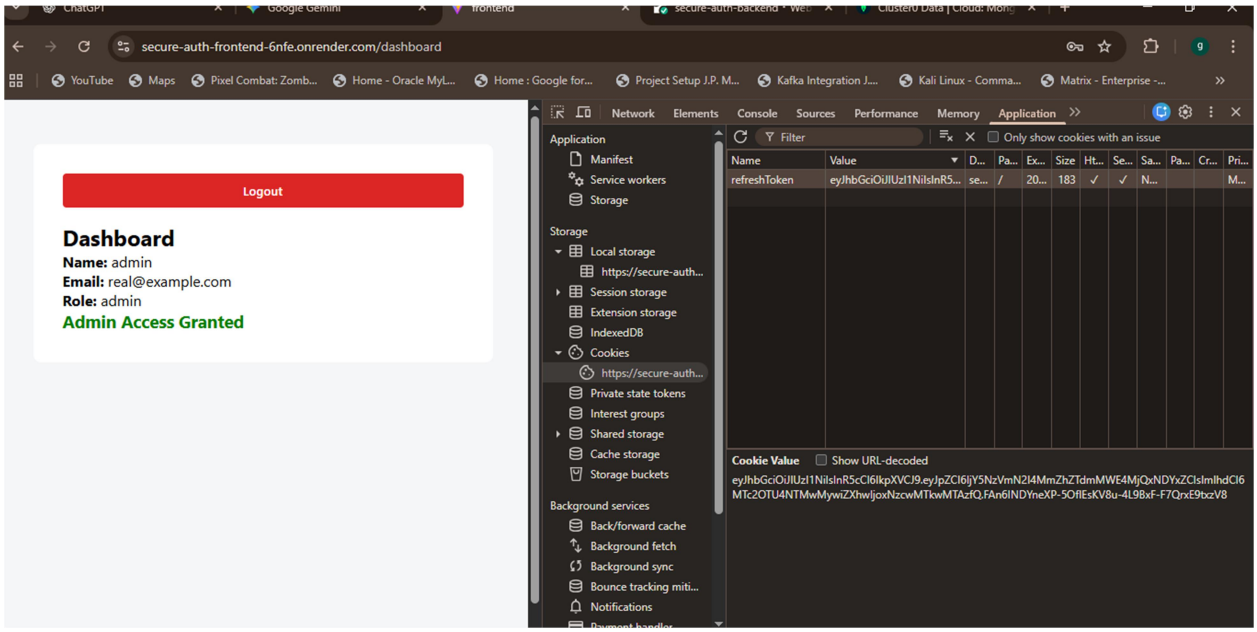
Category	Tool/Library	Purpose
Runtime & Framework	Node.js & Express.js	Core server-side environment.
Database	MongoDB & Mongoose	NoSQL data storage and ODM.
Authentication	`jsonwebtoken` & `bcryptjs`	Token generation and password hashing.
Security	`helmet` & `express-rate-limit`	HTTP header protection and brute-force prevention.
Middleware	`cookie-parser`	Handling secure `HttpOnly` cookies.

Resume-Ready Line

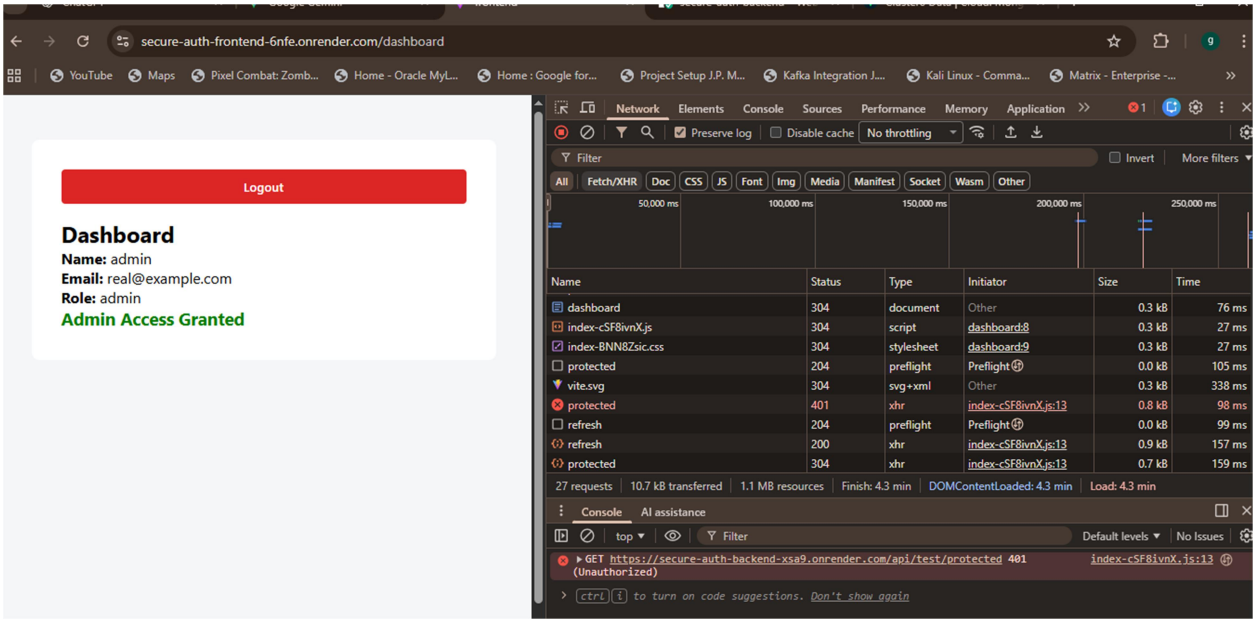
“Built secure MERN authentication boilerplate with RBAC.”

PUC:

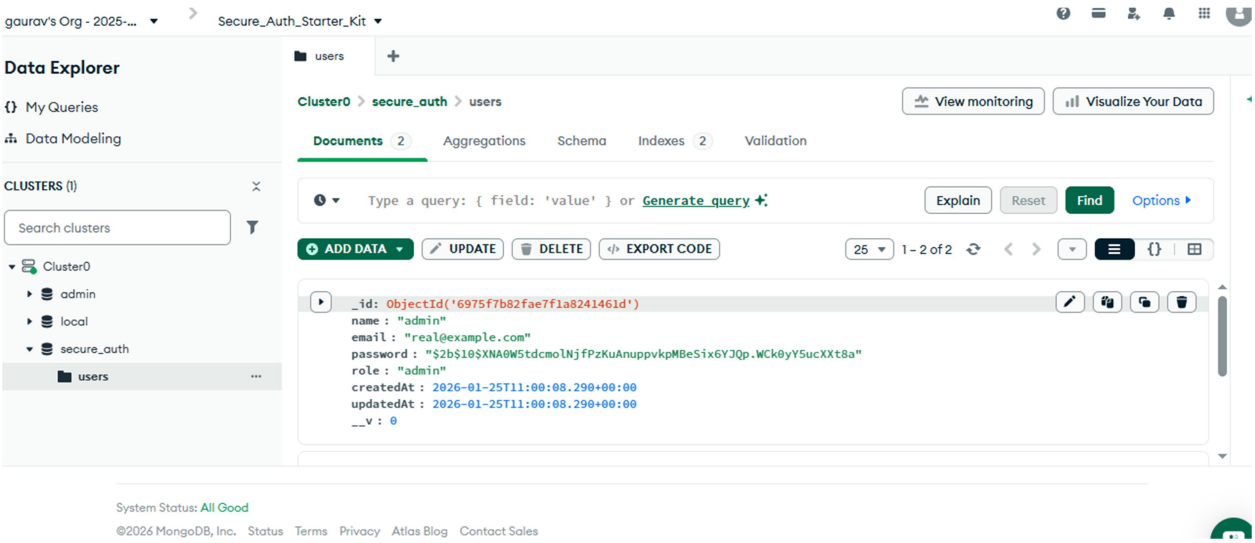




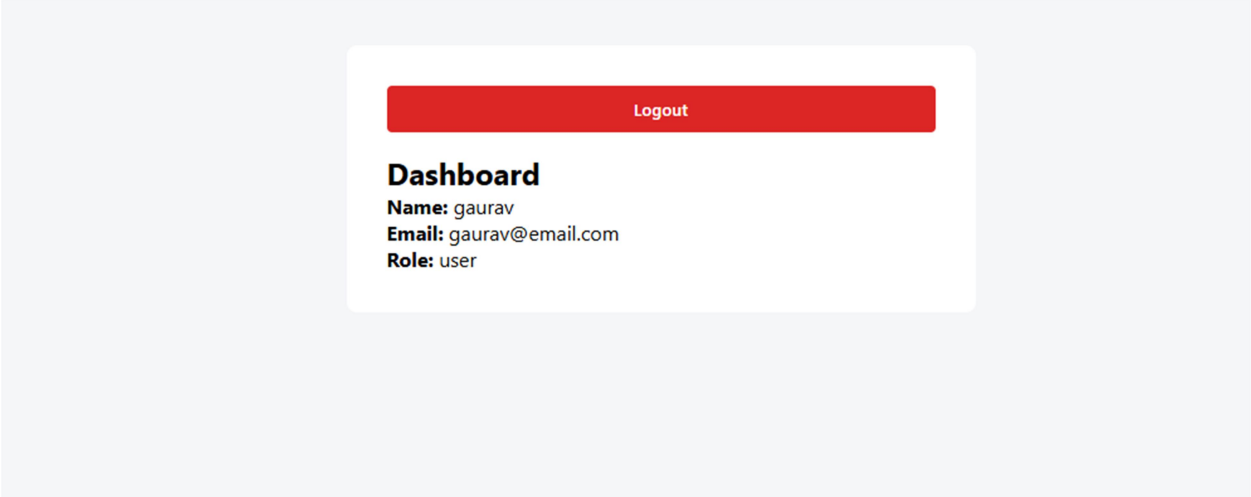
This screenshot confirms that the Refresh Token is successfully stored as an HttpOnly cookie. This ensures that the token is inaccessible to client-side scripts, effectively mitigating Cross-Site Scripting (XSS) vulnerabilities

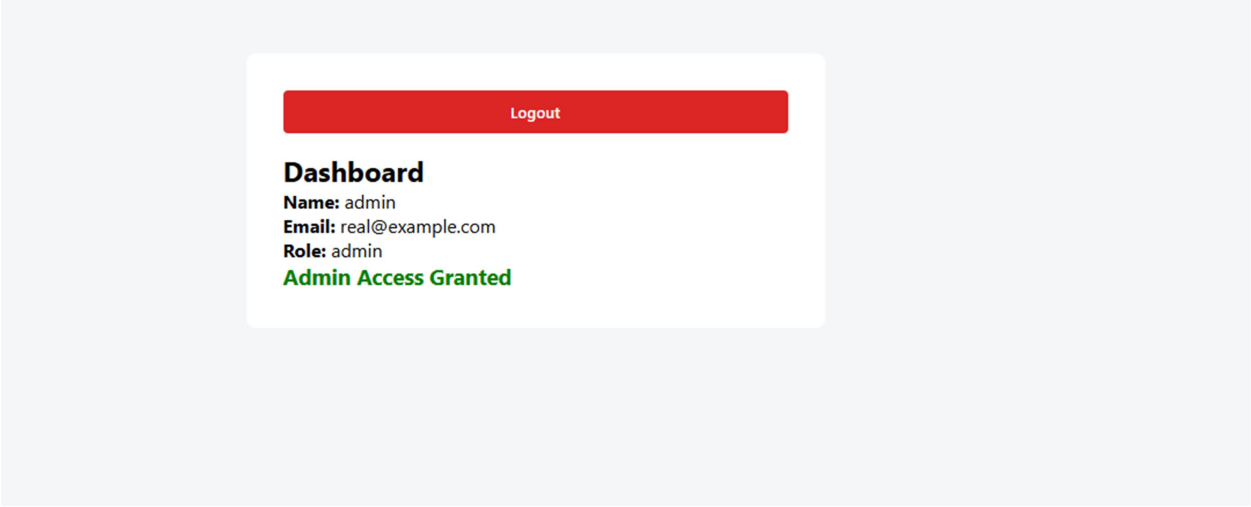


The Network tab demonstrates the automatic token renewal process. When the 15-second access token expires, the Axios interceptor catches the 401 error and silently refreshes the session without requiring the user to re-login .



This database view confirms that user passwords are never stored in plain text. Instead, we utilize bcryptjs for industry-standard one-way hashing.





The UI demonstrates granular Role-Based Access Control (RBAC). The system dynamically renders components and protects routes based on the 'role' field in the JWT payload.

Security Audit Checklist Web Application Report

Application Name

Pentest Tracker

Target Audience

- Professional penetration testers
- Security consultants and audit firms
- Blue teams and internal security engineers
- Cybersecurity students and trainees
- Organizations conducting regular application security assessments

1. Executive Summary

Pentest Tracker is a web-based security audit management application designed to streamline and standardize penetration testing workflows. The platform centralizes audit tracking, integrates the OWASP Top 10 security checklist, and enables structured documentation of findings.

By automating checklist generation and providing progress visibility, Pentest Tracker reduces manual effort, improves audit consistency, and enhances reporting accuracy. The application supports both technical pentesters and non-technical stakeholders by offering a clear, organized view of security posture and remediation progress.

2. Application Overview

Pentest Tracker serves as a centralized workspace for managing security audits from initiation to completion. Each audit is associated with a target system and automatically includes a complete OWASP Top 10 checklist, ensuring no critical security category is overlooked.

The application focuses on **repeatability, accountability, and clarity**, enabling teams to conduct structured audits while maintaining detailed records suitable for compliance, reporting, and client communication.

3. Key Features

3.1 Secure Authentication and User Management

- Email-based signup and login with confirmation
- Role-based access control (where applicable)
- Secure session handling

Benefit: Ensures audit data confidentiality and controlled access.

3.2 Audit Management Dashboard

- Create, view, update, and delete security audits
- Associate audits with clients, applications, or infrastructure targets
- Visual overview of audit status and progress

Benefit: Provides quick visibility into ongoing and completed assessments.

3.3 OWASP Top 10 (2021) Checklist Integration

- Automatic generation of the OWASP Top 10 checklist for each audit
- Individual tracking of findings per OWASP category
- Status indicators (e.g., Open, In Progress, Mitigated)

Benefit: Guarantees comprehensive coverage of the most critical web application risks.

3.4 Progress Tracking and Findings Documentation

- Track audit completion percentage
- Record observations, evidence, and remediation notes
- Maintain historical audit data

Benefit: Improves accountability and supports long-term security improvements.

3.5 Report-Ready Structure

- Audit data organized for easy export or reporting
- Clear separation between findings, risk categories, and remediation notes

Benefit: Reduces time spent preparing client or management reports.

4. Use Cases

4.1 Professional Penetration Testing Engagements

Consultants can manage multiple client audits simultaneously while ensuring standardized assessments aligned with OWASP Top 10 guidelines.

4.2 Internal Security Assessments

Organizations can conduct recurring internal audits and track security maturity over time.

4.3 Security Training and Education

Students and junior pentesters can use the platform as a guided framework to learn structured security testing methodologies.

4.4 Compliance and Risk Reporting

Audit records can be referenced during compliance reviews, security assessments, or risk evaluations.

5. Technical Specifications

Layer	Technology	Role
Frontend	**React 18 + Vite**	Provides a lightning-fast development experience and a component-based UI.
Language	**TypeScript**	Adds static typing to JavaScript, preventing common errors like `undefined` findings.
Styling	**Tailwind CSS**	A utility-first CSS framework used for the clean, dashboard-style layout.
Backend	**Supabase**	Handles the PostgreSQL database, Authentication (User Login), and Real-time updates.
Icons	**Lucide React**	Consistent, lightweight vector icons used throughout the dashboard.

Application leverages a "Full-Stack TypeScript" architecture, which minimizes bugs by ensuring data types are consistent from the database all the way to the UI.

6. User Experience Insights

Pentest Tracker prioritizes **clarity and usability** over complexity. The interface is designed to minimize cognitive load during audits, allowing users to focus on testing rather than documentation.

- Clean dashboard with actionable insights
- Logical navigation between audits and checklist items
- Minimal learning curve for new users
- Accessible reporting structure for non-technical stakeholders

This balance makes the application suitable for both hands-on pentesters and managerial reviewers.

7. Workflow

Phase 1: Setup & Scoping (The Dashboard)

Everything starts at the **Dashboard**.

1. **Create New Audit:** You click "Create New Audit" and define the boundaries. This is where you input the Client Name, the Target (IPs/URLs), and the timeframe.
2. **Audit Status:** The audit begins in the **Planning** state. This alerts the team that the legal work (ROE/Contracts) is being finalized but testing hasn't started.

Phase 2: Active Assessment (The Audit View)

Once the audit is marked as **In Progress**, you move into the core testing view. 3. **Methodology Checklist:** You follow the **OWASP Top 10** checklist. As you test for "Injection" or "Broken Access Control," you tick off items to track your manual progress. 4. **Logging Discoveries:** When a bug is found, you switch to the **Findings** tab. You hit "Add Finding" to document the technical details, CVSS score, and Proof of Concept (PoC).

Phase 3: Real-Time Monitoring

The app serves as a "Live Feed" for stakeholders. 5. **Dashboard Sync:** As findings are added, the high-level **Dashboard** automatically updates its count of "Critical" and "High" vulnerabilities. 6. **Progress Tracking:** The **Checklist Progress** bar (e.g., 4/10) gives an immediate visual of how much of the attack surface has been covered.

Phase 4: Reporting & Remediation

The final goal of any pentest is the report. 7. **Export Report:** You use the "Export" feature to generate a document that includes your executive summary and the list of findings. 8. **Closure:** Once the report is delivered and the client fixes the bugs, you update the finding status to **Resolved** and move the Audit status to **Completed**.

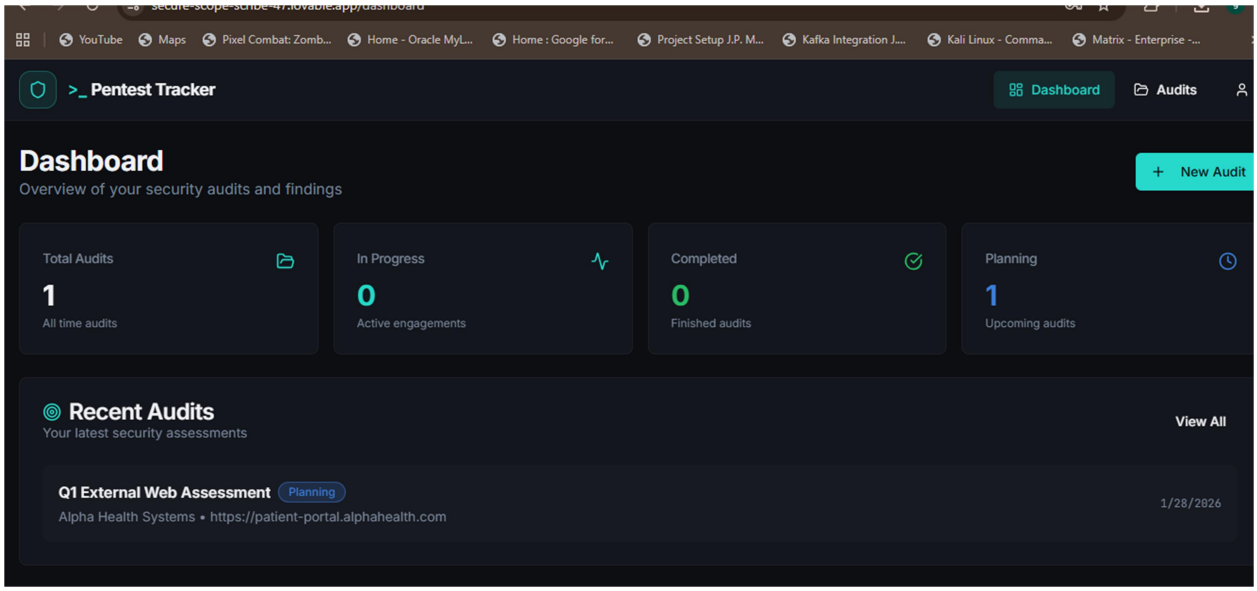
Trigger	Action in App	Result
New Client	Create Audit Modal	Dashboard Card created
Testing Started	Tick OWASP Checklist	Progress bar increases
Bug Found	Add Finding Modal	Risk counter (Critical/High) updates
End of Week	Click Export	PDF/CSV generated for client

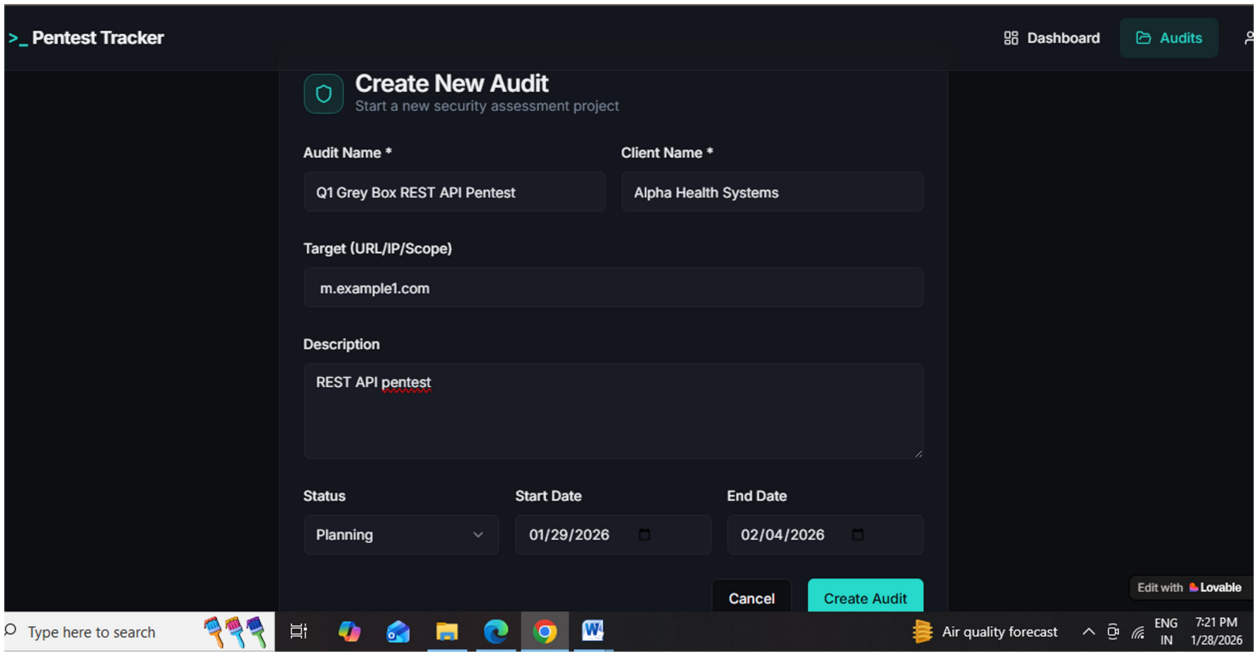
8. Conclusion

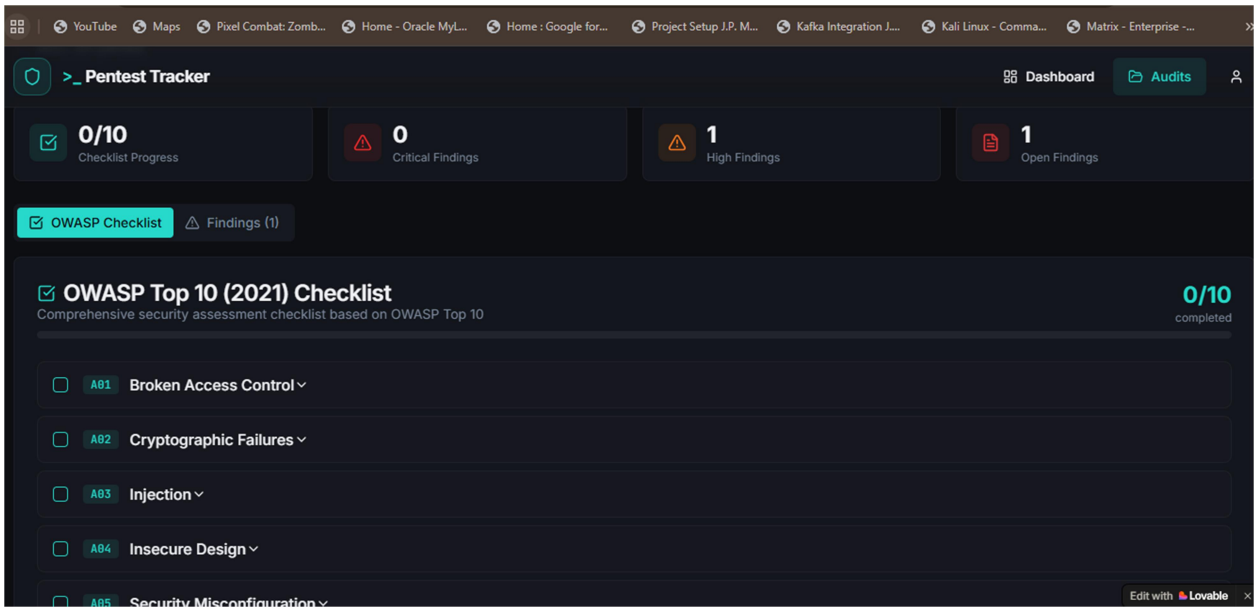
Pentest Tracker is a practical and purpose-built security audit checklist application that enhances the penetration testing lifecycle. By integrating the OWASP Top 10 framework, automating checklist creation, and providing structured audit tracking, the platform improves consistency, efficiency, and reporting quality.

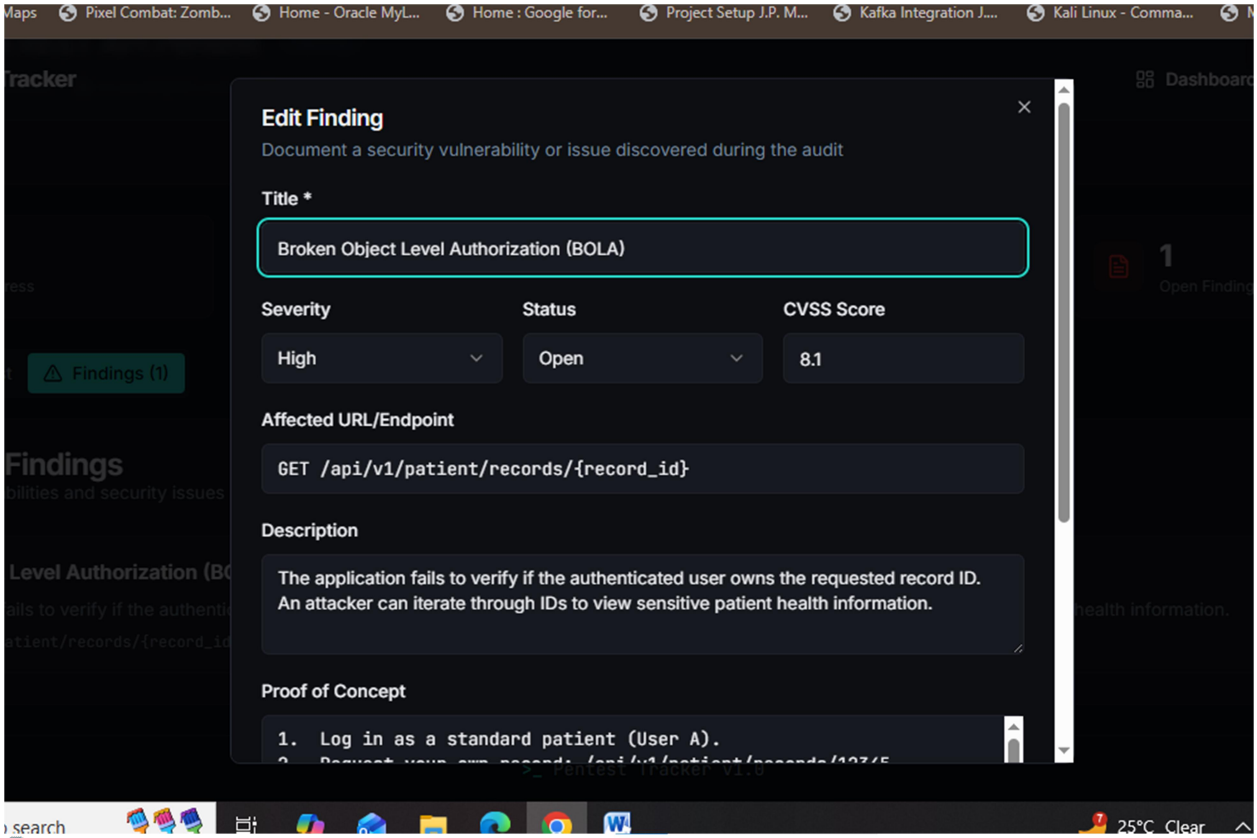
Its ability to bridge the gap between technical findings and stakeholder communication makes it a valuable tool for modern security teams. Pentest Tracker not only supports effective security assessments but also promotes a repeatable and defensible approach to application security auditing.

PUC:




>_ Pentest Tracker
Dashboard
Audits
Create New Audit
Start a new security assessment project
Audit Name *
Q1 Grey Box REST API Pentest
Client Name *
Alpha Health Systems
Target (URL/IP/Scope)
m.example1.com
Description
REST API pentest
Status
Planning
Start Date
01/29/2026
End Date
02/04/2026
Cancel
Create Audit
Edit with Lovable
Type here to search
Air quality forecast
ENG
IN
7:21 PM
1/28/2026


YouTube
Maps
Pixel Combat: Zomb...
Home - Oracle MyL...
Home : Google for...
Project Setup J.P. M...
Kafka Integration J....
Kali Linux - Comma...
Matrix - Enterprise -...
>_ Pentest Tracker
Dashboard
Audits
0/10
Checklist Progress
0
Critical Findings
1
High Findings
1
Open Findings
OWASP Checklist
Findings (1)
OWASP Top 10 (2021) Checklist
Comprehensive security assessment checklist based on OWASP Top 10
0/10
completed
A01 Broken Access Control
A02 Cryptographic Failures
A03 Injection
A04 Insecure Design
A05 Security Misconfiguration
Edit with Lovable



Q1_Grey_Box_REST_API_Pentest_Report - Notepad

File Edit Format View Help

SECURITY AUDIT REPORT
==========

Audit Name: Q1 Grey Box REST API Pentest
Client: Alpha Health Systems
Target: m.example1.com
Status: planning
Start Date: 2026-01-29
End Date: 2026-02-04

EXECUTIVE SUMMARY

REST API pentest

FINDINGS SUMMARY

Total Findings: 1
- Critical: 0
- High: 1
- Medium: 0
- Low: 0
- Info: 0

Q1_Grey_Box_REST_API_Pentest_Report - Notepad

File Edit Format View Help

CHECKLIST STATUS

[ ] A01: Broken Access Control
Notes: No notes

[ ] A02: Cryptographic Failures
Notes: No notes

[ ] A03: Injection
Notes: No notes

[ ] A04: Insecure Design
Notes: No notes