Security-First TCG Platform

Isaac Johnson
Computer Science Department, Utah Valley University
Faculty Advisor: Dr. Sayeed Sajal

Introduction

In the modern web landscape, protecting user data and maintaining application integrity is more crucial than ever. This project presents a fully developed web platform — the Security-First TCG Platform — built using Django and React, with a backend powered by PostgreSQL. Designed for managing hero notes and team information for the Flesh and Blood Trading Card Game (TCG), the application was developed from the ground up with security as a top priority. The platform implements modern security practices aligned with the OWASP Top 10 and includes detailed testing using real-world tools such as Burp Suite and manual penetration testing techniques.

OWASP Issues

- Broken Authentication
- Cross-Site Scripting
- Cross-Site Request Forgery
- Broken Access Control
- Sensitive Data Exposure

Security Features

JWT-Based Authentication

- Stateless login with signed tokens
- Tokens stored securely in frontend storage

Token Expiry

- Each token includes expiration (`exp`) claim
- Users are logged out automatically when tokens expire

Password Hashing with bcrypt

- Salted and hashed using Django's `make_password()`
- Resistant to brute-force and rainbow table attacks

Cross-Site Request Forgery (CSRF) Protection

- CSRF tokens enforced in admin panel
- JWT-based routes guarded with `IsAuthenticated`

Cross-Site Scripting (XSS) Protection

- Notes displayed using JSX with automatic escaping
- Inputs sanitized and validated on backend
- Confirmed safe via payload testing

Role-Based Access Control (RBAC)

- Admin-only routes protected in backend with permission classes
- Frontend conditionally hides admin content unless authorized

CORS Configuration

- `django-cors-headers` used to allow only trusted origins
- Prevents unauthorized frontend access

Secure Database Practices (PostgreSQL + Django ORM)

- Parameterized queries to prevent SQL injection
- Model-level validation and migration system
- No raw SQL used

Platform Features

User Registration and Login

- Secure JWT-based authentication
- Passwords hashed using bcrypt

Hero Notes System

- Select heroes via dropdown categorized by class
- Submit and view strategy notes for each hero
- Notes are securely stored and safely rendered

Admin-Only Pages

- Special routes only visible and accessible to admin users
- Admin role set via custom user model

Token Expiry Handling

- Users automatically logged out when their token expires

Tournament Report Tracking

- Submit reports from events including matchups and outcomes
- Stored for long-term team and meta analysis

Testing

Stored XSS Payload Testing

- Injected `<script>alert('XSS')</script>` into hero notes
- Confirmed script was stored but safely rendered as inert text

Token Expiry Validation

- Waited for JWT to expire
- Verified auto-logout and redirected login behavior

Burp Suite Interception

- Captured and inspected login, note submission, and API requests
- Verified JWT presence in headers and rejected unauthorized attempts

Admin Access Testing

- Tried accessing admin-only routes with non-admin account Verified proper access denial

Database Verification

- Used Django shell to confirm proper database insertion
- Inspected PostgreSQL tables to validate secure, correct storage

Deployment

Deployment is the final and critical step in delivering a secure and reliable application to users. For this project, we intend to deploy the Security-First TCG Platform using NGINX as a reverse proxy and static asset server. NGINX is a high-performance web server that also functions as a reverse proxy, load balancer, and caching system. In our case, it will:

- Serve the react frontend
- Forward API requests to the django backend
- handle TLS termination using HTTPS
- Provide rate limiting and other protections

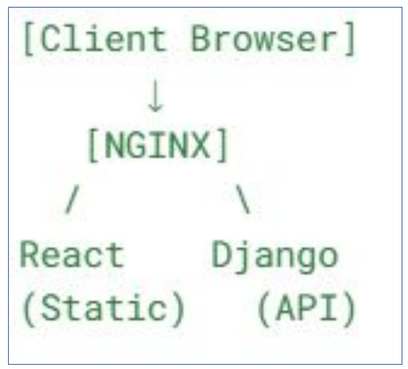

FIgure1. Deployment Diagram

References

JWT vs Session Cookies: What to Use for Authentication. https://auth0.com/blog/cookies-vs-tokens/, 2024.
jango Software Foundation. Security in Django. https://docs.djangoproject.com/en/stable/topics/security/, 2024.
Mozilla Developer Network. Cross-Origin Resource Sharing (CORS). https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS, 2024.