API SECURITY RISK ANALYSIS REPORT

Target: JSONPlaceholder (REST API)

Prepared By: P. Charan Teja

Organization: Future Interns

Task: Cyber Security Task 3 (2026)

Date: February 3, 2026

Executive Summary & Scope

Project Objective

The objective of this assessment is to identify security vulnerabilities within the JSONPlaceholder API. This report focuses on identifying risks related to authentication, data exposure, and server configuration to provide actionable remediation steps for a SaaS environment.

Scope of Work

- **Target:** https://jsonplaceholder.typicode.com
- **Tools:** Postman (API Testing), Browser DevTools (Header Inspection)
- **Methodology:** Read-only security audit based on the **OWASP API Security Top 10**.

Risk Finding #1 – Broken Authentication

Description

The /users endpoint allows unauthorized access to the entire user directory. No API key, Bearer Token, or login credentials are required to retrieve data.

Evidence

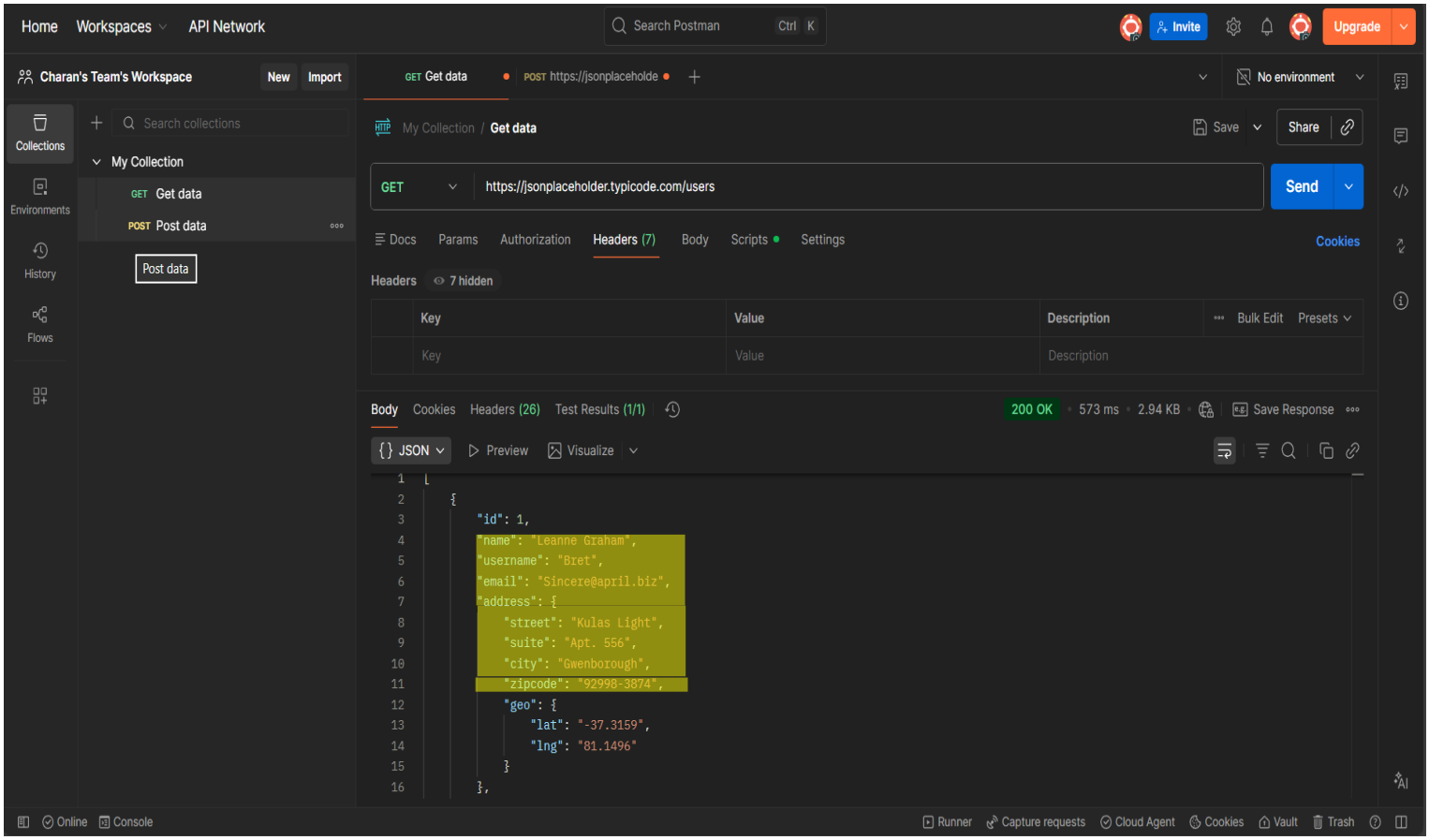

Figure 1: Unauthorized GET request to /users returning 200 OK status.

Business Impact:

Attackers can perform "Data Scraping" to collect thousands of user emails and names, which can be sold on the dark web or used for targeted phishing campaigns. **Remediation:** Implement **JWT (JSON Web Tokens)** or **OAuth2**. All endpoints must verify an Authorization header before returning data.

Risk Finding #2 – Excessive Data Exposure

Description

The API provides detailed objects that include non-essential information. Specifically, the response for an individual user includes precise GPS coordinates (Latitude and Longitude).

Evidence

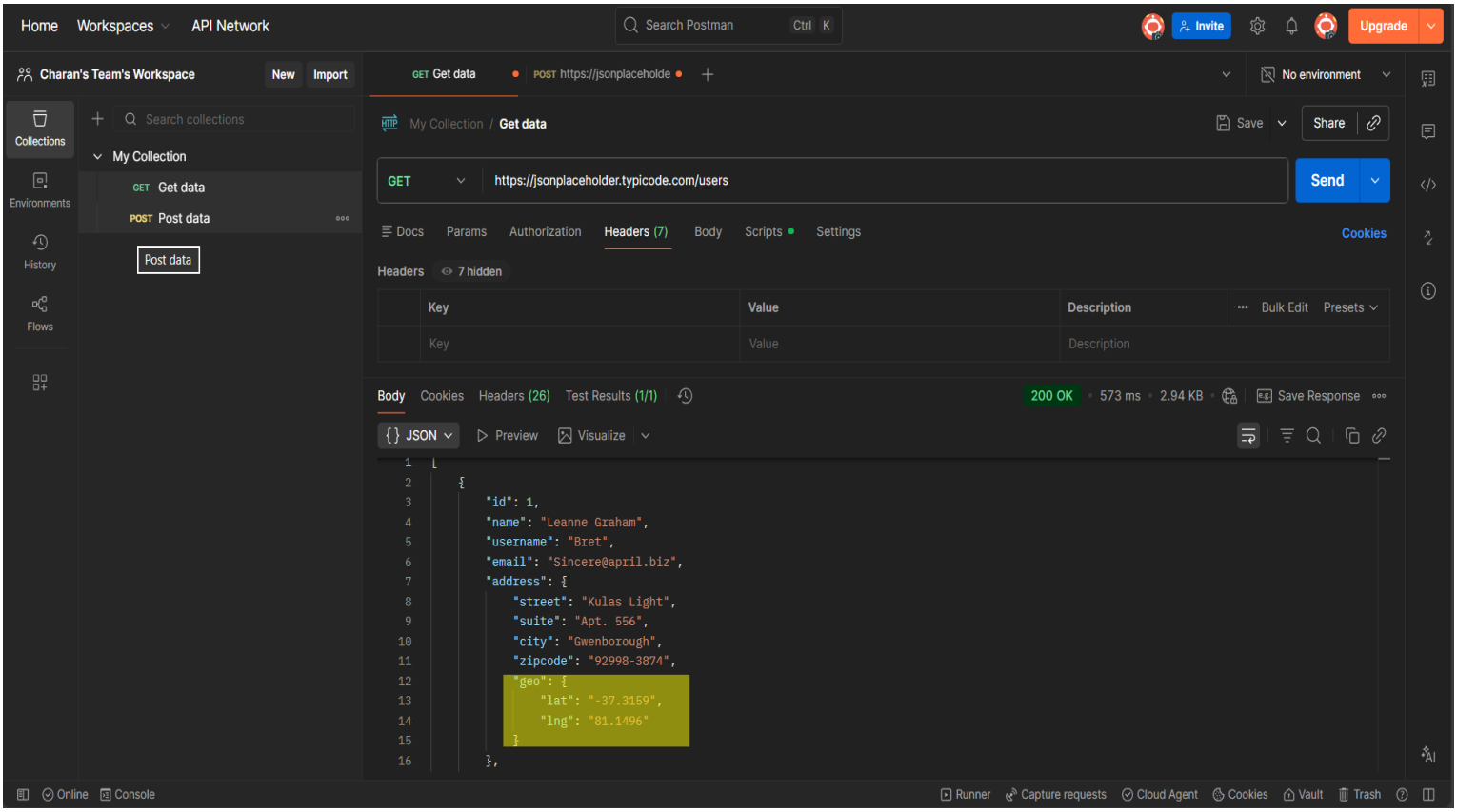

Figure 2: User object exposing "lat" and "lng" geo-location data.

Business Impact:

This is a privacy violation. Under regulations like **GDPR**, leaking a user's physical location data unnecessarily can result in heavy legal fines. **Remediation:** Implement a **Data Transfer Object (DTO)** layer. The API should only send the specific fields needed by the front-end (e.g., name and username).

Risk Finding #3 – Security Misconfiguration (Headers)

Description

The server response lacks critical security headers that protect the API from browser-based attacks like Clickjacking and Man-in-the-Middle (MitM).

Evidence

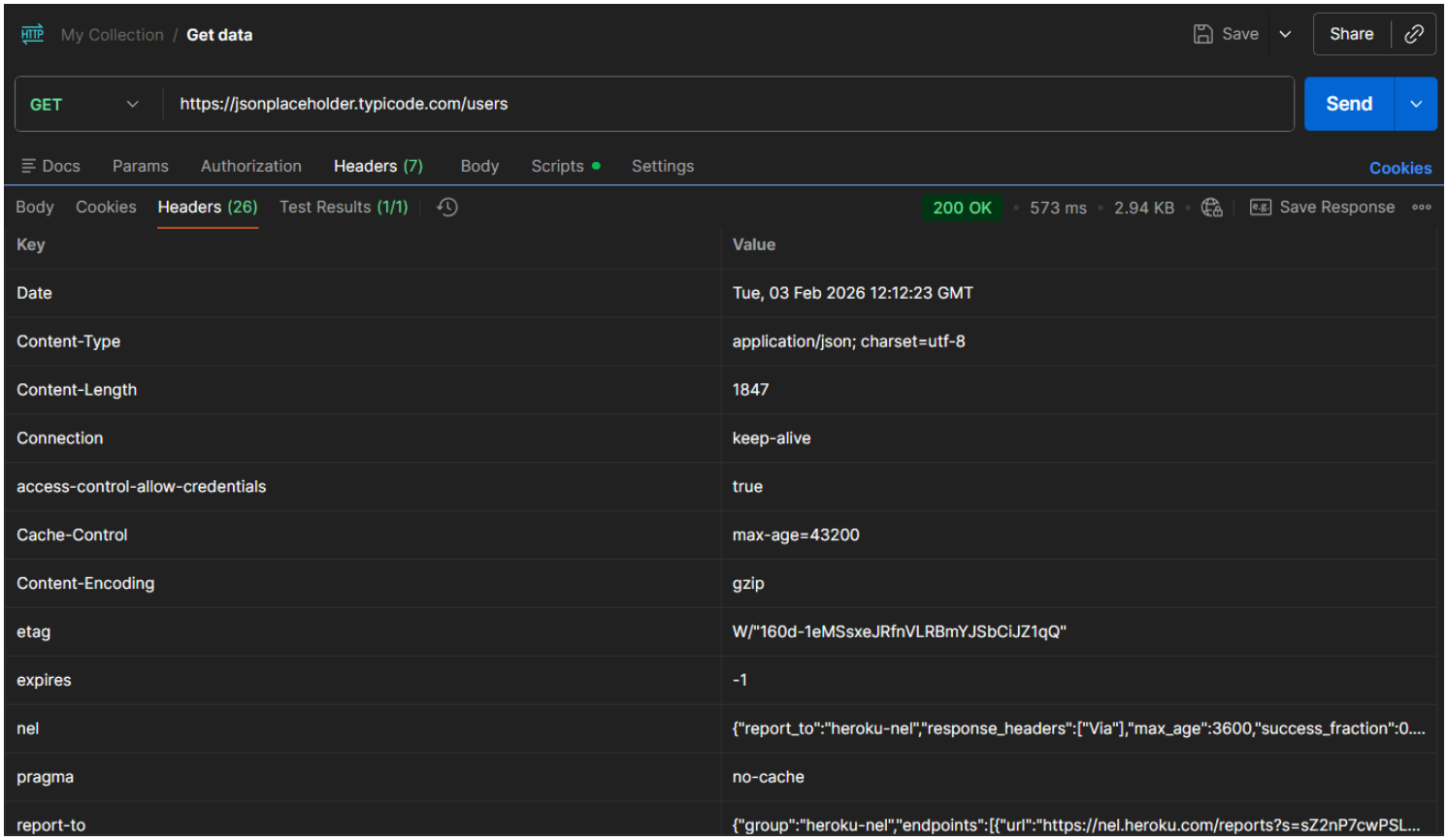

Figure 3: Inspection of Response Headers showing missing HSTS and CSP.

Business Impact:

Without these headers, the API communication is easier to intercept and exploit, especially on public networks. **Remediation:** Configure the web server to include:

- Strict-Transport-Security (HSTS)
- Content-Security-Policy (CSP)
- X-Content-Type-Options: nosniff

Risk Finding #4 – Unauthorized Data Creation

Description

The /posts endpoint allows unauthenticated users to perform POST requests. This means any external actor can simulate the creation of content on the server.

Evidence

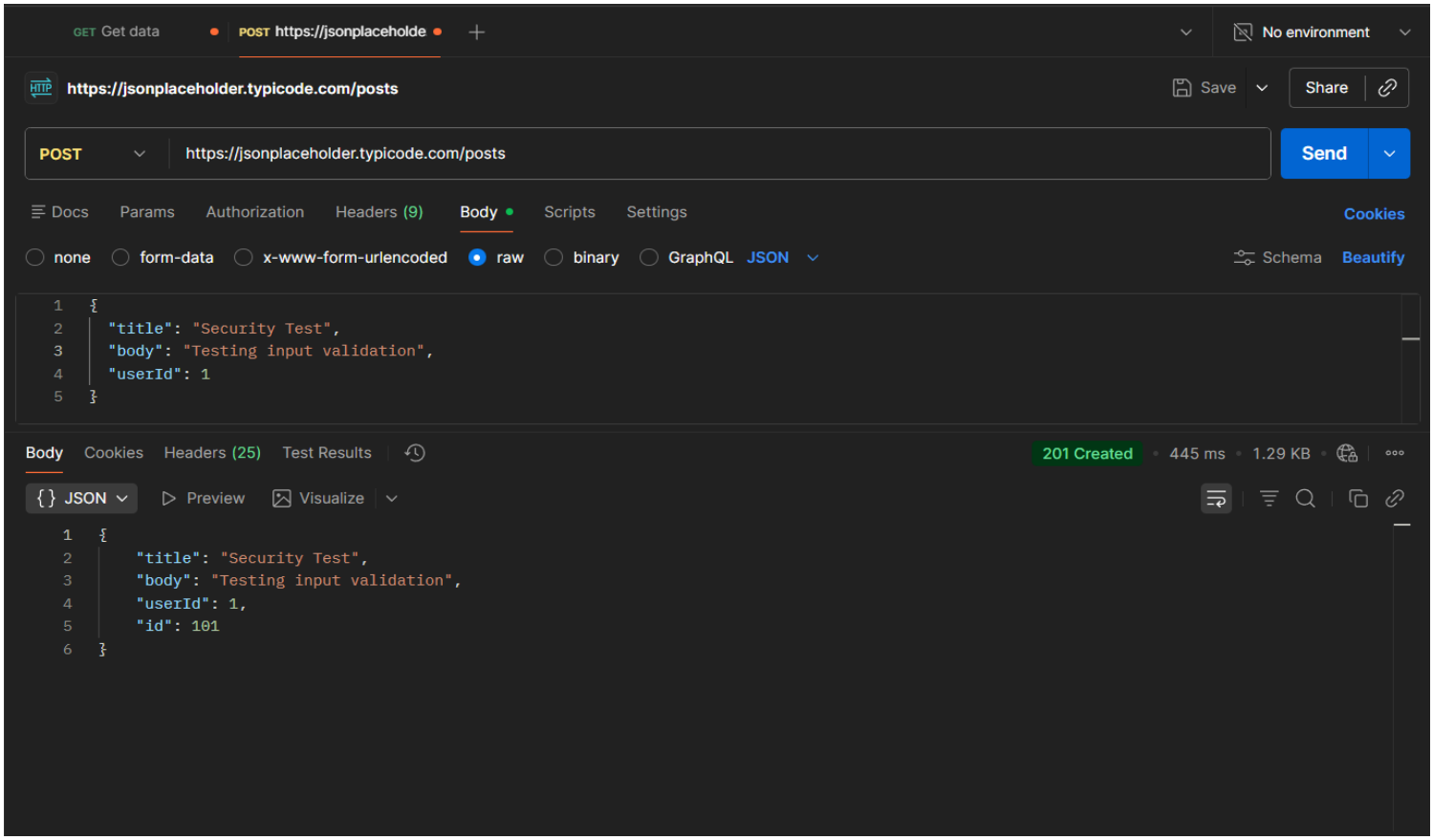

Figure 4: POST request successfully creating a new record without a token.

Business Impact:

This vulnerability leads to "Data Pollution." An attacker could use a script to flood the database with millions of fake posts, causing service downtime or high storage costs. Remediation: Restrict state-changing methods (POST, PUT, DELETE) to authenticated users with specific roles (Role-Based Access Control).

Conclusion

This API Security Risk Analysis of the JSONPlaceholder service has identified several critical vulnerabilities, most notably **Broken Authentication** and **Excessive Data Exposure**. While the API is designed for testing and development, these findings represent real-world risks that could lead to significant data breaches, legal non-compliance, and system abuse in a production SaaS environment.

By implementing the remediation steps outlined in this report—specifically the use of **JSON Web Tokens (JWT)**, **Data Transfer Objects (DTOs)**, and **Security Headers**—the security posture of the API can be transformed from a "Public Access" model to a "Secure Enterprise" model.

Adopting these industry-standard security practices is essential for protecting user privacy, maintaining data integrity, and ensuring the long-term reliability of modern web services.