Guild Audits

OCTOBER
2025

LUNAR FINANCE SECURITY ASSESSMENT FINAL FINDINGS REPORT

By Guild Audits

TABLE OF CONTENTS

Executive Summary	2
Project Summary	2
Assessment Test & Internal Penetration Test	3
Finding Severity Ratings	4
Tester Notes and Recommendations	5
Vulnerability Report Card	6-7
Types of Issues	8
Technical Findings	9-14
Summary	15
Appendix	16

DISCLAIMER

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.
Time-limited engagements do not allow for a full evaluation of all security controls. Guild Audits prioritized the assessment to identify the weakest security controls an attacker would exploit. Guild Audits recommends conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

PROJECT SUMMARY

Project Name: **Lunar Finance**

Description: Lunar Finance is a decentralized finance (DeFi) platform focused on providing cross-chain trading and liquidity aggregation tools. It is a protocol that lets users:

- Swap tokens across different blockchains (i.e. cross-chain swaps), Bridge assets from one blockchain to another, Use limit orders (place trades that execute only when a price condition is met), Automate strategies like DCA (dollar-cost averaging) and “Refuel gas” (help with covering transaction/gas costs across chains).

EXECUTIVE SUMMARY

The security assessment of Lunar Finance’s staging environment revealed multiple critical vulnerabilities that could be exploited to execute unauthorized transactions, expose sensitive data, and manipulate financial records.
A critical CORS misconfiguration (LF-001) allowed cross-origin requests from untrusted domains. When combined with authorization bypasses on transaction execution (LF-002) and insecure direct object references (IDOR) on transaction data (LF-003, LF-004), this introduced a severe risk of fund loss and data leakage.

The absence of application-level rate limiting (LF-005) increased the likelihood of denial-of-service attacks and automated abuse. Additionally, a design flaw permitting users to arbitrarily update transaction statuses (LF-006) undermined the integrity and immutability expected in DeFi applications. Although no exploitable XSS vulnerabilities were identified (LF-007), the lack of a Content Security Policy (CSP) and other security headers left the application more susceptible to future attacks. The remaining findings were low or informational and do not pose immediate risk.

ASSESSMENT OVERVIEW

From September 20th, 2025 to October 1st, 2025, Lunar Finance engaged Guild Audits to evaluate the security posture of its infrastructure compared to current industry best practices that included an internal network penetration test. All testing performed is based on the NIST SP 800-115 Technical Guide to Information Security Testing and Assessment, OWASP Testing Guide (v4), and customized testing frameworks.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.

Web Application & API Security Review

A review of the Lunar Finance web application and API emulated the role of an external attacker. The auditors scanned endpoints, tested for vulnerabilities, and attempted attacks including CORS misconfigurations, IDOR, authorization bypasses, and logic flaws, with the goal of identifying unauthorized access to data or functionality.

FINDING SEVERITY RATINGS

The following table defines levels of severity and corresponding CVSS score rangethat areused throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Description
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.2
High	7.0-8.9	Exploitation is more difficult but could cause elevated privilegesand potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Medium	4.0-6.9	Vulnerabilities exist but are not exploitable or require extrastepssuch as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduceanorganization’s attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Information	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Risk Factors

Risk is measured by two factors: Likelihood and Impact:

Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are givenbasedonthe difficulty of the attack, the available tools, attacker skill level, and client environment.

Impact

Impact measures the potential vulnerability’s effect on operations, including confidentiality, integrity, and availability of client systems and/or data, reputational harm, and financial loss.

Scope:
Frontend (lunar-test.vercel.app) & API (lunar-fi-main-server-staging-7b98.up.railway.app)

Out of Scope;
- Social Engineering
- Production environment testing

Tester Notes and Recommendations

Testing of the Lunar Finance staging environment indicates that the application is still in active development and currently lacks several foundational security controls. The most critical issues identified center around missing authentication and authorization mechanisms on sensitive API endpoints.
The combination of overly permissive CORS settings with endpoints that accept unvalidated user input represents a severe design flaw. In a DeFi context, such weaknesses could directly enable cross-origin attacks leading to theft or manipulation of user funds.

Key Recommendations:

1. Implement Wallet-Based Authentication
○ Use “Sign-In with Solana” or equivalent wallet signature methods to bind all quotes and transaction executions to authenticated sessions.
○ Ensure session tokens or signatures are verified server-side.
2. Enforce Authorization Controls
○ Restrict data access and transaction updates strictly to the owning wallet address.
○ Apply role-based access controls (RBAC) where applicable.
3. Restrict CORS Policy
○ Immediately limit Access-Control-Allow-Origin to trusted frontend domains.
○ Remove Access-Control-Allow-Credentials unless strictly required.

4. Harden Input Validation

◯ Enforce strict schema validation at the API layer.

◯ Reject any unexpected or extraneous fields to prevent bypasses and logic abuse.

Positive Observations:

● Application was not vulnerable to common reflected or stored XSS payloads.

VULNERABILITY REPORT CARD

Severity Breakdown

- 🔴 Critical: 4
- 🔴 High: 2
- 🟠 Moderate: 1
- 🔵 Low/Info: 3

ID	Severity	Finding	Recommendations
LF-001	Critical	**CORS Misconfiguration**	Restrict CORS origins; implement CSP.
LF-002	Critical	**Authorization Bypass on /api/swap/execute**	Implement wallet-based authentication; validate user ownership.
LF-003	Critical	**IDOR on GET /api/transactions**	Add wallet-based authentication; implement RBAC.
LF-004	Critical	**IDOR on PUT /api/transaction/status**	Restrict updates to owners/automation; make statuses immutable.
LF-005	High	**No Application-Level Rate Limiting**	Implement rate limiting per IP/account; add WAF.

LF-006	High	**Design Flaw in User-Editable Transaction Status**	Automate status via blockchain; remove user-editable fields.
LF-007	Medium	**Input Validation Flaws**	Implement strict schema validation; reject unknown fields.
LF-008	Low/info	**XSS Testing (Clean)**	Continue monitoring; add CSP defense-in-depth.
LF-009	Low/info	**Source Map Disclosures (valid)**	Ensure source maps are not served in production.
LF-010	Low/info	**JS Asset Analysis (Google Analytics)**	Review GA configuration for potential PII exposure.

TYPES OF ISSUES

Open: Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved: These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged: Vulnerabilities that have been acknowledged but are yet to be resolved.

Partially Resolved: Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

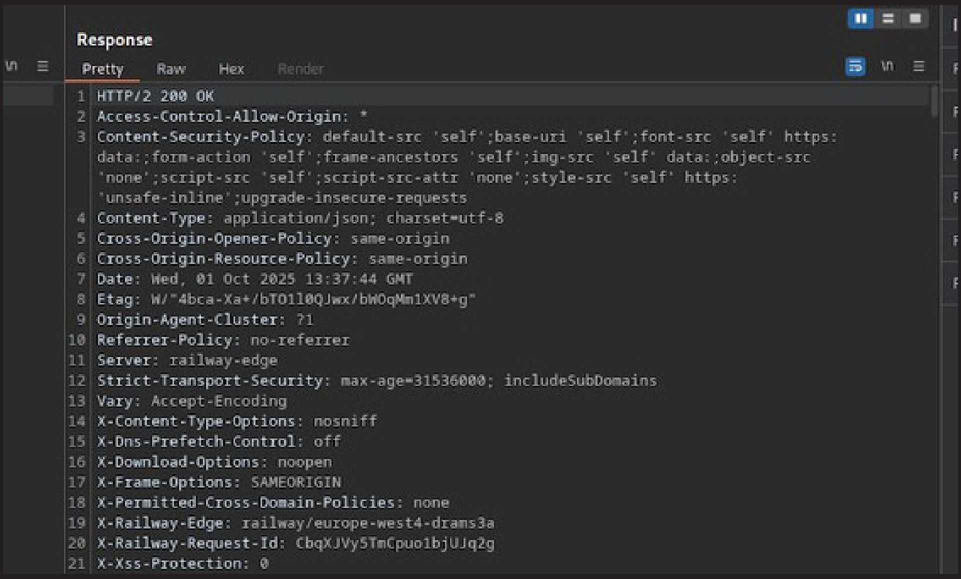
TECHNICAL FINDINGS

CRITICAL SEVERITY ISSUES

1. CORS Misconfiguration

Description: API configured with **Access-Control-Allow-Origin:** while allowing credentials, enabling cross-origin attacks.

Evidence:



```
HTTP/2 200 OK
Access-Control-Allow-Origin: *
Content-Security-Policy: default-src 'self';base-uri 'self';font-src 'self' https:
data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src
'none';script-src 'self';script-src-attr 'none';style-src 'self' https:
'unsafe-inline';upgrade-insecure-requests
Content-Type: application/json; charset=utf-8
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-origin
Date: Wed, 01 Oct 2025 13:37:44 GMT
Etag: W/"4bca-Xa+/bTO1l0QJwx/bWOqMm1XV8+g"
Origin-Agent-Cluster: ?1
Referrer-Policy: no-referrer
Server: railway-edge
Strict-Transport-Security: max-age=31536000; includeSubDomains
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
X-Dns-Prefetch-Control: off
X-Download-Options: noopen
X-Frame-Options: SAMEORIGIN
X-Permitted-Cross-Domain-Policies: none
X-Railway-Edge: railway/europe-west4-drams3a
X-Railway-Request-Id: CbqXJVy5TmCpuo1bjUJq2g
X-Xss-Protection: 0
```

Impact of the vulnerability : Attackers can read swap quotes and execute transactions from malicious domains.

Recommendation: Restrict CORS to trusted domains; implement CSP headers.

Status: **Resolved**

2. Authorization Bypass on /api/swap/execute

Description: Endpoint builds transactions for any token address without authentication. Each execute transaction should bound to a user account, with *quoteId* of nonce (single use) and short expiry .

Evidence:

~~~
```http
POST /api/swap/execute HTTP/1.1
Content-Type: application/json

{
  "userAddress": "EPjFWdd5AufqSSqeM2qN1xzybapC8G4wEGGkZwyTDt1v",
  "quoteId": "abc123"
}

HTTP/1.1 200 OK
{
  "transaction": "4hXQkuz9WcRr5a...",

}
```
~~~

Impact: Enables CSRF attacks tricking users into signing malicious transactions.

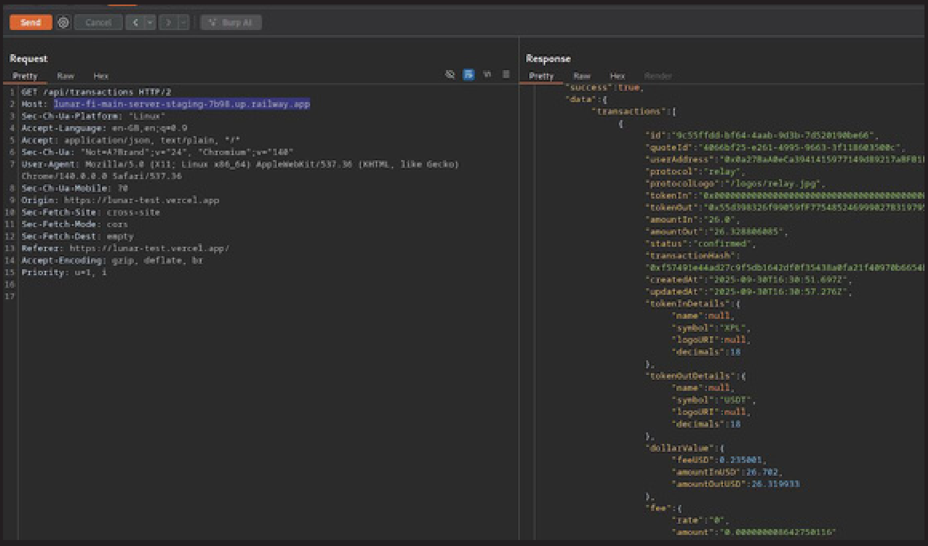
Recommendation: Implement wallet-based authentication; validate user ownership such that each execution call with a nonce *quoteid* is bound to an authenticated user, this prevents replay attacks and unauthorized transaction.

Status: Resolved

3. IDOR on GET /api/transactions

Description: Exposes transaction history for any user address without authorization. Any authenticated user can initiate a massive data breach by downloading the entire transaction history of the platform.

Evidence:



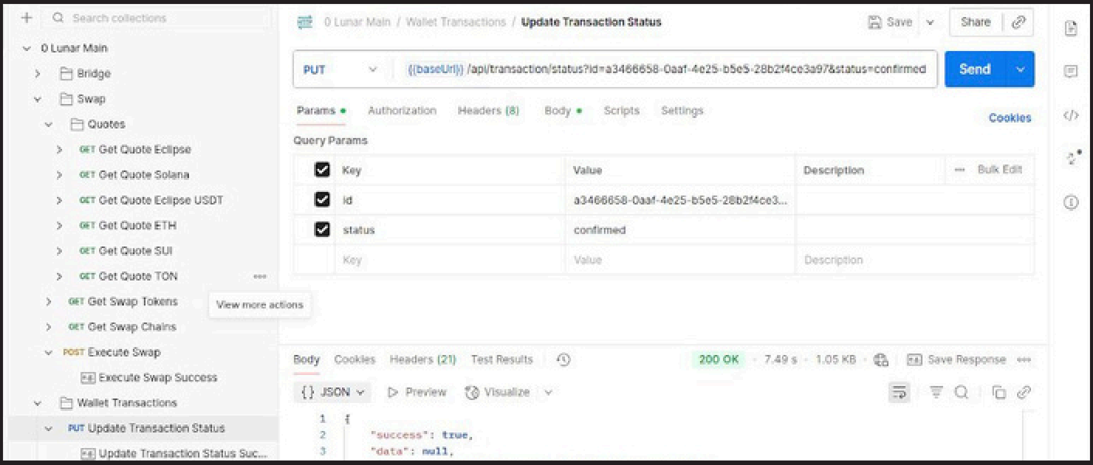
```
GET /api/transactions HTTP/2
Host: lunar-fi-main-server-staging-7b98.up.railway.app
Sec-Ch-Ua-Platform: "Linux"
Accept-Language: en-GB,en;q=0.9
Accept: application/json, text/plain, */*
Sec-Ch-Ua: "Not=A?Brand";v="24", "Chromium";v="140"
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/140.0.0.0 Safari/537.36
Sec-Ch-Ua-Mobile: ?0
Origin: https://lunar-test.vercel.app
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://lunar-test.vercel.app/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i
```

```
"success":true,
"data":{
    "transactions":[
        {
            "id":"9c55ffdd-bf64-4aab-9d3b-7d520190be66",
            "quoteId":"4066bf25-e261-4995-9663-3f118603508c",
            "userAddress":"0x0a278aA8eCa394141597714 9d89217a8F81
            "protocol":"relay",
            "protocolLogo":"/logos/relay.jpg",
            "tokenIn":"0x0000000000000000000000000000000000000000
            "tokenOut":"0x55d398326f99059fF775485246999027B31979
            "amountIn":"26.0",
            "amountOut":"26.328806085",
            "status":"confirmed",
            "transactionHash":
            "0xf57491e44ad27c9f5db1642df0f35438a0fa21f40970b66548
            "createdAt":"2025-09-30T16:30:51.697Z",
            "updatedAt":"2025-09-30T16:30:57.276Z",
            "tokenInDetails":{
                "name":null,
                "symbol":"XPL",
                "logoURI":null,
                "decimals":18
            },
            "tokenOutDetails":{
                "name":null,
                "symbol":"USDT",
                "logoURI":null,
                "decimals":18
            },
            "dollarValue":{
                "feeUSD":0.235001,
                "amountInUSD":26.702,
                "amountOutUSD":26.319933
            },
            "fee":{
                "rate":"0",
                "amount":"0.000000008642750116"
```

Impact: Mass enumeration of transaction data and financial patterns.
Recommendation: Add wallet-based authentication; implement access controls.
Status: Resolved

4. IDOR on PUT /api/transaction/status

Description: Allows unauthorized modification of any transaction status. Any user can arbitrarily change their transaction status (e.g., from Pending to Confirmed, or Failed to Confirmed). This indicates an inadequate authorization scope on the status field.
Evidence:



PUT {{baseUrl}} /api/transaction/status?id=a3466658-0aaf-4e25-b5e5-28b2f4ce3a97&status=confirmed

Query Params

Key	Value	Description
id	a3466658-0aaf-4e25-b5e5-28b2f4ce3...	
status	confirmed	

200 OK · 7.49 s · 1.05 KB

```
{
  "success": true,
  "data": null,
```

Exploitation:
A user could set a transaction to Confirmed before the blockchain confirms it, confusing reconciliation systems, customer support, or triggering internal rewards/crediting prematurely. More critically, an attacker could change another user's status, leading to financial disputes and system integrity failure.

Impact: Transaction manipulation, fraud, and service disruption.
Recommendation: Restrict updates to transaction owners; automate status from blockchain.

Status: Resolved

HIGH SEVERITY ISSUES

1. **No Rate Limiting**

Description: API lacks application-level rate limiting.

Evidence:

~~~
```bash
# Vegeta load test results
Requests: [500] - Success: [100%] - Duration: [1s]
No rate limiting detected up to 500 requests/second
```
~~~

Impact: Denial-of-service and resource exhaustion attacks.

Recommendation: Implement IP-based rate limiting (100 req/min).

Status: Resolved

2. **Design Flaw**: User-Editable Transaction Status

Description: Users can arbitrarily modify transaction statuses.

Evidence:

~~~
```http
PUT /api/transaction/status
{
&quot;executionId&quot;: &quot;tx_67890&quot;,
&quot;status&quot;: &quot;pending&quot;
}
→ Status accepted despite contradicting blockchain state
```
~~~

Impact: Undermines financial record integrity and enables fraud.

Recommendation: Derive status exclusively from blockchain data.

Status: Resolved

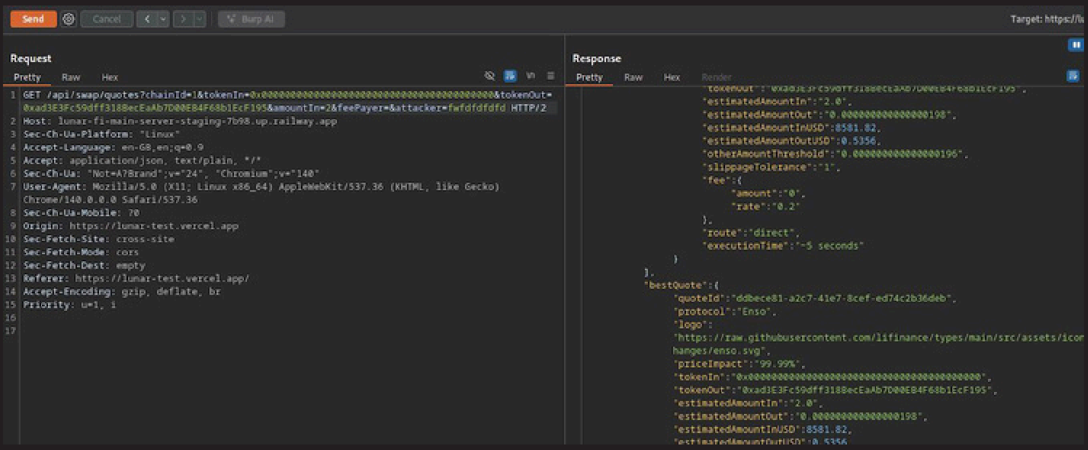
MEDIUM SEVERITY ISSUES

1. Input Validation Flaws

Description:

Certain API endpoints accept inputs without strict schema validation. This creates opportunities for attackers to inject unexpected or malicious fields, potentially leading to logic manipulation, bypassing of security controls, or denial of service through malformed requests. While no direct exploit was identified during testing, this weakness lowers the overall resilience of the application against injection-style attacks.

Evidence:



```
GET /api/swap/quotes?chainId=1&tokenIn=0x0000000000000000000000000000000000000000&tokenOut=
0xad3E3Fc59dff318BecEaAb7D00EB4F68b1EcF195&amountIn=2&feePayer=&attacker=fwfdfdfdfd HTTP/2
Host: lunar-fi-main-server-staging-7b98.up.railway.app
Sec-Ch-Ua-Platform: "Linux"
Accept-Language: en-GB,en;q=0.9
Accept: application/json, text/plain, */*
Sec-Ch-Ua: "Not=A?Brand";v="24", "Chromium";v="140"
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/140.0.0.0 Safari/537.36
Sec-Ch-Ua-Mobile: ?0
Origin: https://lunar-test.vercel.app
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://lunar-test.vercel.app/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i
```

```
            "tokenOut":"0xad3E3Fc59dff318BecEaAb7D00EB4F68b1EcF195",
            "estimatedAmountIn":"2.0",
            "estimatedAmountOut":"0.000000000000000198",
            "estimatedAmountInUSD":8581.82,
            "estimatedAmountOutUSD":0.5356,
            "otherAmountThreshold":"0.000000000000000196",
            "slippageTolerance":"1",
            "fee":{
                "amount":"0",
                "rate":"0.2"
            },
            "route":"direct",
            "executionTime":"~5 seconds"
        }
    ],
    "bestQuote":{
        "quoteId":"ddbece81-a2c7-41e7-8cef-ed74c2b36deb",
        "protocol":"Enso",
        "logo":
        "https://raw.githubusercontent.com/lifinance/types/main/src/assets/icon
        hanges/enso.svg",
        "priceImpact":"99.99%",
        "tokenIn":"0x0000000000000000000000000000000000000000",
        "tokenOut":"0xad3E3Fc59dff318BecEaAb7D00EB4F68b1EcF195",
        "estimatedAmountIn":"2.0",
        "estimatedAmountOut":"0.000000000000000198",
        "estimatedAmountInUSD":8581.82,
        "estimatedAmountOutUSD":0.5356
```

Recommendation:

Apply OWASP ASVS Input Validation (V5.x) and OWASP API Security best practices by enforcing strict schema validation for all API inputs. Reject requests containing unexpected fields, enforce proper data types, and sanitize all user-provided input before processing. If a user sends extra fields, wrong data types, or malformed JSON, the server should immediately reject the request with a 400 Bad Request.

Status: **Resolved**

LOW SEVERITY ISSUES

1. **XSS Testing (Clean)**

Description:
Certain API endpoints accept inputs without strict schema validation. This creates opportunities for attackers to inject unexpected or malicious fields, potentially leading to logic manipulation, bypassing of security controls, or denial of service through malformed requests. While no direct exploit was identified during testing, this weakness lowers the overall resilience of the application against injection-style attacks.

Recommendation:
Continue monitoring for XSS risks in future development.
Implement a Content Security Policy (CSP) to add defense-in-depth and reduce the impact of any potential script injection vulnerabilities.

Status: **Resolved**

2. Source Map Exposure (Valid):

Description:
JavaScript source maps were observed in the staging environment. While not exploitable on their own, source maps reveal internal code structure, backend url, variable names, and potentially sensitive logic that could aid attackers in discovering vulnerabilities.

Recommendation:
Disable or remove source map files in production environments.
Ensure that only minified and obfuscated JavaScript bundles are deployed publicly.

Status: **Resolved**

3. **JavaScript Asset Analysis (Google Analytics)**

Description:
The application uses Google Analytics (GA) scripts for user tracking. While this is common practice, there is a potential risk of unintentionally transmitting personally identifiable information (PII) or sensitive financial metadata through GA events. No direct evidence of PII leakage was observed during testing.

Recommendation:
Review the Google Analytics configuration to ensure that no sensitive or personally identifiable data is transmitted. Adhere to privacy and compliance requirements (e.g., GDPR) for analytics data collection. This is inline with " OWASP ASVS V9.2 – Data Protection in Transit" and "CWE-359: Exposure of Private Personal Information to an Unauthorized Actor", best practices

Status: **Resolved**

CLOSING SUMMARY

The assessment reveals fundamental security gaps requiring immediate attention before production deployment. Critical vulnerabilities in authentication and authorization pose direct risks to user funds. Prioritize implementing wallet-based authentication, proper CORS configuration, and authorization checks.

Immediate Actions Required

1. Implement wallet signature authentication
2. Fix CORS misconfiguration
3. Add authorization checks to all endpoints
4. Implement rate limiting
5. Remove user-editable transaction statuses

A re-assessment is recommended after remediation to verify all issues are resolved.

GUILD AUDITS

Guild Audits is dedicated to delivering top-tier security solutions across both Web2 and Web3. In today's rapidly evolving Web3 ecosystem, our mission is to combat the rising wave of hacks and exploits that undermine trust and adoption. We are driven by a passion to safeguard protocols, strengthen infrastructure, and ensure a more secure digital future.