**Cybersecurity Implementation Plan**

**DineQuest**

**A Website Created by Zachary Wedding**

**Cybersecurity Implementation Plan**

Written by: Zachary Wedding

**I. Executive Summary**

The purpose of this cybersecurity implementation plan is to outline the strategy, steps, and resources necessary to enhance the cybersecurity posture of DineQuest. This plan provides a comprehensive roadmap to mitigate risks.

**II. Objective and Goal**

Primary Objective: To enhance the cybersecurity defenses of DineQuest by implementing industry best practices, security technologies, and policies.

Goal: Mitigate potential cybersecurity threats and vulnerabilities

**III. Risk Assessment**

Conduct a risk assessment to identify potential threats and vulnerabilities within DineQuest.

**Critical Assets**

|  |  |
| --- | --- |
| **Asset** | **Function/Importance** |
| GitHub Pages | Hosts the website |
| Yelp API/API Key | Essential for accessing restaurant data to narrow, book, and display |
| HTML Pages | Structures the site and works with scripts.js for user interaction |
| Scripts | Deals with input, API calls, and displaying results |
| Styling | Boosts the aesthetic side of the website |

**Impact and Likelihood of Risks**

|  |  |  |
| --- | --- | --- |
| **Risk** | **Impact** | **Likelihood** |
| API Key Exposure | High | High |
| Cross-Site Scripting | High | High |
| Injection of Malicious Scripts from Third-Party APIs | High | Medium |
| Bots Abusing Features | Medium | Medium |
| Unencrypted Transmission | High | High |
| Vulnerable Dependencies | Medium | Medium |

**IV. Risk Mitigation Strategy**

|  |  |  |
| --- | --- | --- |
| **Activity** | **Threat Mitigated** | **Implementation** |
| API Key Rotation | Protects against API key exposure | Generate a new key off the Yelp for Developers website |
| HTTPS | Protects against data interception | GitHub pages automatically implements HTTPS |
| Input Sanitation | Protects against cross-site scripting attacks | Used the DOMPurify library to sanitize user input |
| Content Security Policy Header | Protects against malicious content injection and controls what is loaded | Added a content security policy tag to the head |
| Sanitation of API Responses | Ensures data coming from the Yelp API is sanitized | API responses and sanitized before being shown on the frontend |
| Rate Limiting and Bot Protection | Protects against unwanted bots and excessive API calls | Created a function so that there are at least five seconds in between API calls and implemented a robots.txt file |
| Minimize Exposure of Sensitive Data | Protects against unnecessary data being shown | Only necessary data is displayed |
| Security Audit | Helps identify vulnerabilities | Used security headers by Scott Helme |

**V. Review and Evaluation**

Many security measures have been put into place to help mitigate threats and protect the critical assets of DineQuest. Critical assets being protected include the GitHub Pages, the Yelp API key (and the Yelp Fusion API itself), the HTML pages, scripts, and styling of the website. These assets are at risk for things such as API key exposure, cross-site scripting, injection of malicious scripts from Yelp, bots abusing features, unencrypted transmission, and vulnerable dependencies. To mitigate these risks, security features have been implemented. These features include API key rotation, HTTPS, input and API response sanitation, a content security policy header, rate limiting and bot protection, minimizing exposure on sensitive data, and a security audit. It is through these features that DineQuest will mitigate risks effectively and safeguard critical assets.