Web Application Vulnerability Report

**Scan Date:** 2025-08-01 12:43:46 UTC **Target URL:** https://kb.builtwith.com/general-questions/expired-technology-website/

# Executive Summary

**Scan Results Summary:**- Medium: 4

# Detailed Findings

## 1. Outdated jQuery

**Severity:** Medium **CVSS Score:** CVSS: 5.0 (Medium)

### Description:

Using an outdated jQuery version exposes your application to known security vulnerabilities, such as cross-site scripting (XSS) and cross-site request forgery (CSRF) attacks, which can be exploited by attackers to steal sensitive data or take control of user sessions. Outdated jQuery versions may also lack important security patches and updates, making them more susceptible to DOM-based attacks and other client-side vulnerabilities. This can lead to unauthorized access, data breaches, and other malicious activities, compromising the integrity and confidentiality of your application.

### Proof of Concept:

Version 3.5.1 in use.

### Remediation:

* Upgrade to 3.7.1.

### Business Impact:

The use of an outdated jQuery version poses a significant financial impact, as a successful attack could result in costly data breaches, stolen sensitive information, and potential regulatory fines. Reputation risks are also high, as a security incident could lead to a loss of customer trust and damage to the company's brand image. From a compliance perspective, failing to update jQuery could lead to non-compliance with industry standards and regulations, such as GDPR and PCI-DSS, resulting in additional fines and penalties. Furthermore, an attack exploiting this vulnerability could cause operational disruption, leading to downtime, lost productivity, and additional costs associated with incident response and remediation.

## 2. Outdated Bootstrap

**Severity:** Medium **CVSS Score:** CVSS: 5.0 (Medium)

### Description:

Using an outdated Bootstrap version poses significant security risks, as it may contain known vulnerabilities that can be exploited by attackers to inject malicious code or steal sensitive data. Outdated Bootstrap versions may lack essential security patches, leaving applications susceptible to Cross-Site Scripting (XSS) attacks, Cross-Site Request Forgery (CSRF) attacks, and other types of malicious exploits. This can compromise the integrity and confidentiality of user data, ultimately leading to financial and reputational losses for organizations.

### Proof of Concept:

Version 4.3.1 in use.

### Remediation:

* 1. \*\*Assess Current Version and Dependencies (Priority: High)\*\*
* 2. \*\*Update to the Latest Version of Bootstrap (Priority: High)\*\*
* 3. \*\*Address Breaking Changes and Compatibility Issues (Priority: Medium)\*\*

### Business Impact:

The potential financial impact of an outdated Bootstrap vulnerability could be substantial, with costs associated with incident response, data breach notifications, and potential regulatory fines. Reputation risks are also significant, as a security breach could lead to a loss of customer trust and loyalty, ultimately affecting revenue and long-term business sustainability. Non-compliance with industry security standards and regulations, such as GDPR and PCI-DSS, may result in additional fines and penalties, exacerbating the financial burden. Furthermore, operational disruption risks are also a concern, as a successful exploit could lead to system downtime, compromising business continuity and causing further financial losses.

## 3. Outdated Microsoft ASP.NET

**Severity:** Medium **CVSS Score:** CVSS: 5.0 (Medium)

### Description:

Using an outdated Microsoft ASP.NET version can expose applications to known security vulnerabilities, allowing attackers to exploit them and potentially gain unauthorized access to sensitive data. Outdated ASP.NET versions may lack essential security patches, leaving them susceptible to common web attacks such as SQL injection and cross-site scripting (XSS). This can lead to technical impacts like data breaches, compromised user sessions, and malicious code execution, ultimately compromising the integrity and confidentiality of the application.

### Proof of Concept:

Version 4.0 in use.

### Remediation:

* Upgrade to 4.8.

### Business Impact:

No business impact analysis provided.

## 4. CVE-2019-8331 - Bootstrap -

**Severity:** Medium **CVSS Score:** CVSS: 5.0 (Medium)

### Description:

CVE-2019-8331 is a vulnerability in the Bootstrap JavaScript library that allows for Cross-Site Scripting (XSS) attacks due to insufficient sanitization of user-input data in the `tooltip` and `popover` components. An attacker can exploit this vulnerability by injecting malicious JavaScript code through crafted input, potentially leading to unauthorized access, data theft, or other malicious activities. The technical impact of this vulnerability is moderate to high, as it can be triggered through various attack vectors, including phishing emails, malicious websites, or compromised web applications that utilize the affected Bootstrap components.

### Proof of Concept:

Version: 4.3.1 is affected by CVE-2019-8331

### Remediation:

* 1. \*\*Upgrade to Bootstrap version 4.3.1 or later\*\*: The most effective mitigation step is to upgrade to a version of Bootstrap that has fixed the vulnerability. According to the Bootstrap GitHub page, version 4.3.1 includes the fix for CVE-2019-8331. [1]
* 2. \*\*Use a Content Security Policy (CSP)\*\*: Implementing a Content Security Policy (CSP) can help mitigate the vulnerability by defining which sources of content are allowed to be executed within a web page. This can help prevent an attacker from injecting malicious JavaScript code. The CSP header can be set to only allow scripts from trusted sources, such as the website's own domain. [2]
* 3. \*\*Validate and sanitize user input\*\*: To prevent an attacker from injecting malicious data, it's essential to validate and sanitize all user input. This can be done using libraries or frameworks that provide input validation and sanitization functionality. For example, using HTML escaping to prevent XSS attacks. [3]
* 4. \*\*Use a Web Application Firewall (WAF)\*\*: A Web Application Firewall (WAF) can help detect and prevent attacks that exploit the CVE-2019-8331 vulnerability. A WAF can be configured to block requests that contain malicious JavaScript code or other suspicious patterns. [4]
* 5. \*\*Monitor for suspicious activity\*\*: Finally, it's essential to monitor the website for suspicious activity, such as unusual JavaScript errors or unexpected requests. This can help detect potential attacks and allow for swift mitigation. [5]

### Business Impact:

The CVE-2019-8331 vulnerability in Bootstrap poses a moderate financial impact, as a successful XSS attack could lead to data breaches, unauthorized access, or other malicious activities, potentially resulting in significant costs for incident response and remediation. The reputation of an organization exploiting this vulnerability could be severely damaged, leading to loss of customer trust and potential legal repercussions. From a compliance perspective, failure to address this vulnerability could lead to non-compliance with relevant regulations, such as GDPR or HIPAA, resulting in fines and penalties. Additionally, the operational disruption risk is significant, as a successful attack could compromise critical web applications, leading to downtime, data loss, and disruption of business operations.