



# SLIIT

*Discover Your Future*

Sri Lanka Institute of Information Technology

## Bug Bounty Report 01

api-prod-digitalgym.aswatson.com

IE2062 – Web Security

Submitted by:

**IT22227836 – ADHIKARI A.M.V.B.M**

Date of submission

2024.10.31

## Table of Contents

Report 01 – api-prod-digitalgym.aswatson.com .....	3
Vulnerability .....	5
1. Title - Application Error Disclosure .....	5
.....	5
Description .....	6
Impact Assessment .....	7
Steps to reproduce .....	7
Proof of concept (if applicable) .....	8
Vulnerability scanning using OWASP ZAP .....	8
Vulnerability finding using namp .....	9
Proposed mitigation or fix .....	10
2. Title - Multiple X-Frame-Options Header Entries .....	11
Description .....	11
Impact Assessment .....	12
Steps to reproduce .....	13
Proof of concept (if applicable) .....	14
Vulnerability scanning using OWASP ZAP .....	14
Vulnerability finding using namp .....	15
Vulnerability finding using nikto .....	16
Proposed mitigation or fix .....	16

## Report 01 – api-prod-digitalgym.aswatson.com

Main domain	<a href="https://www.aswatson.com/">https://www.aswatson.com/</a>
Sub domain	api-prod-digitalgym.aswatson.com
IP address	125.252.231.73
platform	HackerOne



The A.S. Watson Group is the world's largest health and beauty retail group, with over 15,700 stores in 25 markets worldwide serving over 28 million customers per week, and over 3 billion customers and members.

A.S. Watson Group looks forward to working with the security community to discover vulnerabilities to keep our businesses and customers safe. As we operate in many different countries, we will be rolling out our bug bounty program in phases. Our focus within this rollout, is our retail websites (and in the near future, mobile apps on both Android and IOS).

## Vulnerability detected

Alert Type	Risk	Count
<a href="#">Application Error Disclosure</a>	Medium	6 (33.4%)
<a href="#">Content Security Policy (CSP) Header Not Set</a>	Medium	327 (1,773.2%)
<a href="#">Multiple X-Frame-Options Header Entries</a>	Medium	92 (484.2%)
<a href="#">Cookie No HttpOnly Flag</a>	Low	1 (5.2%)
<a href="#">Cookie Without Secure Flag</a>	Low	15 (78.4%)
<a href="#">Cookie without SameSite Attribute</a>	Low	1 (5.2%)
<a href="#">Private IP Disclosure</a>	Low	7 (36.8%)
<a href="#">Secure Pages Include Mixed Content</a>	Low	1 (5.2%)
<a href="#">Strict-Transport-Security Header Not Set</a>	Low	370 (1,947.4%)
<a href="#">Timestamp Disclosure - Unix</a>	Low	180 (947.4%)
<a href="#">X-Content-Type-Options Header Missing</a>	Low	137 (721.2%)
<a href="#">Charset Mismatch (Header Versus Meta-Content-Type-Charset)</a>	Informational	34 (178.4%)
<a href="#">Content-Type Header Missing</a>	Informational	2 (10.5%)
<a href="#">Information Disclosure - Suspicious Comments</a>	Informational	9 (47.4%)
<a href="#">Loosely Scoped Cookie</a>	Informational	1 (5.2%)
<a href="#">Modern Web Application</a>	Informational	23 (121.2%)
<a href="#">Re-examine Cache-control Directives</a>	Informational	210 (1,105.2%)
<a href="#">Session Management Response Identified</a>	Informational	16 (84.2%)
<a href="#">User Controllable HTML Element Attribute (Potential XSS)</a>	Informational	2 (10.5%)
<b>Total</b>		<b>18</b>

File Edit View Analyse Report Tools Import Export Online Help

Standard Mode

History Search Alerts Output Spider AJAX Spider Active Scan

Alerts (16)

- > Application Error Disclosure (6)
- > Content Security Policy (CSP) Header Not Set
- > Multiple X-Frame-Options Header Entries (8)
- > Cookie No HttpOnly Flag (4)
- > Cookie Without Secure Flag (3)
- > Cookie without SameSite Attribute (4)
- > Private IP Disclosure (4)
- > Strict-Transport-Security Header Not Set (371)
- > Timestamp Disclosure - Unix (232)
- > X-Content-Type-Options Header Missing (126)
- > Information Disclosure - Suspicious Comments
- > Loosely Scoped Cookie (4)
- > Modern Web Application (4)
- > Re-examine Cache-control Directives (121)
- > Session Management Response Identified (4)
- > User Agent Fuzzer (60)

Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

## Vulnerability

### 1. Tittle - Application Error Disclosure

<https://api-prod-digitalgym.aswatson.com> (2)

#### Application Error Disclosure (1)

► GET <https://api-prod-digitalgym.aswatson.com/docs/manager-howto.html>

<https://api-prod-digitalgym.aswatson.com> (2)

#### Application Error Disclosure (1)

▼ GET <https://api-prod-digitalgym.aswatson.com/docs/manager-howto.html>

##### Alert tags

- [WSTG-v42-ERRH-02](#)
- [WSTG-v42-ERRH-01](#)
- [OWASP\\_2021\\_A05](#)
- [OWASP\\_2017\\_A06](#)
- [CWE-200](#)

##### Alert description

This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.

## Description

Application Error Disclosure occurs when an application displays detailed internal error messages to users, which is commonly caused by unhandled exceptions or debug mode being activated. These messages can provide sensitive information such as database structure, file directories, and server characteristics, allowing attackers to better understand the system's design and exploit potential flaws.

To reduce this risk, programs should always present generic error messages to users while recording detailed information inside for developers. Debugging information should never be exposed in production situations, and adequate error handling procedures should be in place to prevent accidental data leaking.

## Affected components

- **Web Application Frameworks:** If not correctly setup, frameworks (such as Django, Flask, and Laravel) might display full stack traces or debug information during problems.
- **Web servers:** such as Apache or Nginx, may display server-specific error messages or version information if error handling is not configured securely.
- **Databases:** When database mistakes occur (for example, SQL syntax issues), users may see sensitive information such as table names, field names, or queries.
- **APIs:** Poorly managed API faults might result in extensive answers that expose internal logic, database queries, or application states.
- **Third-Party Libraries:** Libraries or plugins that are integrated into the system may reveal internal processing data if they fail, particularly if they are not properly secured or error-handling.

These components, when not properly configured, are vulnerable to leaking valuable information through error messages.

## Impact Assessment

- **Information leakage:** Detailed error messages might reveal crucial system information including database schemas, server file locations, and program versions. Attackers can utilize this information to improve their understanding of the system's architecture and find prospective attack vectors.
- **Increased Attack Surface:** Error messages that reveal valuable information make it easier for attackers to create vulnerabilities like SQL Injection, Cross-Site Scripting (XSS), and Remote Code Execution. Attackers can circumvent security measures by understanding how the program processes data or resolves faults.
- **Reconnaissance:** During the reconnaissance phase of an attack, these errors offer attackers with information about the underlying technologies, frameworks, and infrastructure, allowing them to execute more targeted and effective attacks.
- **Reputation and Compliance Risks:** If an attacker successfully exploits the exposed information, it might result in data breaches or service interruptions, which harms the organization's reputation. Furthermore, noncompliance with security requirements (such as GDPR or PCI-DSS) may result in legal and financial consequences.

## Steps to reproduce

- **Trigger an Error via User Input:**  
Navigate to the input fields of the web application (e.g., search box, login form, contact form). Enter invalid or malformed data (e.g., special characters, incorrect SQL queries, or excessively long strings) to intentionally cause an error.  
Example input: `" OR 1=1 --` in a login form to test for SQL errors.
- **Observe the Error Message:** Submit the form or action and check whether the application displays a detailed error message. Look for details like stack traces, file paths, database error codes, or server configuration.
- **Check HTTP Responses:** Use a tool like Burp Suite or browser developer tools (F12) to inspect the HTTP responses from the server. Verify if the response contains sensitive data in the error message, headers, or body.

- **Repeat with Various Elements:** To cause various kinds of failures, try comparable faulty inputs in other application areas like search forms, URLs, or API endpoints.

## Proof of concept (if applicable)

### Vulnerability scanning using OWASP ZAP

#### ▪ Request

Request

▼ Request line and header section (776 bytes)

```
GET https://api-prod-digitalgym.aswatson.com/docs/manager-howto.html HTTP/1.1
host: api-prod-digitalgym.aswatson.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:125.0) Gecko/20100101 Firefox/125.0
pragma: no-cache
cache-control: no-cache
referer: https://api-prod-digitalgym.aswatson.com/
Cookie:
ak_bmsc=0C33ED3541A2897A0568193C74DF4DE9~00000000000000000000000000000000~YAAQF6bWfSI8TDGSAQAAK041QhkGo1hU5gTU91V77p4FgdMFEfpU7UdBaA5HzqovLZJWjFaRYUmNr9f0cSSDZyxuIeWPgadea0qbbFqMYsBUY7CnhN/H7Wx5q3Vf5zJyXgbYpcdY1s29D7a/Z+q1mnWeWPkSa+Y/uGWstxKva7AnMwjbr89JEJh5wWz5EbxY9i+KEtaeRvZU6MB9UqCmqbLBBVxRoHtRJ1rX6aX987xCbaxqxpqBTFzWyMk3T/ft9uSykAFY0cTDMrPrFG2IEEtjV5AKraT2H0rUVFN5ySbDIPnr+k2hZely0pETP+qifDkpc7s1s2LgxmWJAm0ljuTxCdZy+DG/u00LgF4m10mgibLu89Kif/LbWUOw==
```

▼ Request body (0 bytes)



## ▪ Response

**Response**

▼ Status line and header section (608 bytes)

HTTP/1.1 200 OK  
Server: Apache  
X-Frame-Options: SAMEORIGIN, SAMEORIGIN  
ETag: W/"74325-1638455407000-gzip"  
Last-Modified: Thu, 02 Dec 2021 14:30:07 GMT  
Content-Type: text/html; charset=UTF-8  
Cache-Control: max-age=0  
Expires: Mon, 30 Sep 2024 09:14:45 GMT  
X-Akamai-Transformed: 9 19867 0 pmb=mRUM,2  
Date: Mon, 30 Sep 2024 09:14:45 GMT  
Connection: keep-alive  
Connection: Transfer-Encoding  
Server-Timing: cdn-cache; desc=MISS  
Server-Timing: edge; dur=97  
Server-Timing: origin; dur=9  
Server-Timing: ak\_p;  
desc="1727687685236\_2111219223\_279699471\_10566\_9949\_3  
33\_0\_-";dur=1  
content-length: 78948

► Response body (78948 bytes)

## Vulnerability finding using nmap

### ▪ Used command

- nmap api-prod-digitalgym.aswatson.com

```
(malmi@kali)-[~/Desktop]
└─$ nmap api-prod-digitalgym.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-30 14:30 EDT
Nmap scan report for api-prod-digitalgym.aswatson.com (125.214.166.34)
Host is up (0.073s latency).
Other addresses for api-prod-digitalgym.aswatson.com (not scanned): 125.214.166.27
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https
```

- **Founded open port**
  - 80/tcp open http
  - 443/tcp open https

## Proposed mitigation or fix

- **Custom error pages:**

Implement customized error pages that do not reveal sensitive information. Instead of displaying stack traces or extensive error messages, provide user-friendly notifications informing them that an error has happened without disclosing any technical information.
- **Logging Errors:**

Log extensive error information on the server side for debugging purposes, but make sure it is not visible to end users. Use logging frameworks that allow you to tailor the level of detail logged to the environment (development, staging, production).
- **Error Handling Mechanisms:**

Create a centralized error handling system that logs exceptions and displays generic error messages. Ensure that this approach is regularly applied throughout the application to handle all forms of problems.
- **Security Configuration:**

Review and configure the application server's error handling options. Disable detailed error messages for production environments in web server configurations (e.g., Apache or Nginx).
- **Input Validation and Sanitization:**

Implement rigorous input validation and sanitization to avoid unexpected errors that may reveal sensitive information via error messages. This reduces the risk of error exposure by ensuring that only legitimate data is processed.
- **Use of Frameworks:**

Use web frameworks with built-in error handling facilities, making sure they are configured to conceal detailed error messages in production.
- **Security Testing:**

Conduct regular security testing and code reviews to discover any error messages that could reveal sensitive information. This involves both automatic vulnerability scanning and manual penetration testing.

## 2. Tittle - Multiple X-Frame-Options Header Entries

### Multiple X-Frame-Options Header Entries (1)

► GET <https://api-prod-digitalgym.aswatson.com/examples/>

### Multiple X-Frame-Options Header Entries (1)

▼ GET <https://api-prod-digitalgym.aswatson.com/examples/>

#### Alert tags

- [WSTG-v42-CLNT-09](#)
- [OWASP\\_2021\\_A05](#)
- [OWASP\\_2017\\_A06](#)
- [CWE-1021](#)

#### Alert description

X-Frame-Options (XFO) headers were found, a response with multiple XFO header entries may not be predictably treated by all user-agents.

## Description

Multiple X-Frame-Options Header Entries: This is a vulnerability wherein a web application, upon response, includes more than one X-Frame-Options HTTP header. The X-Frame-Options header is a security feature that protects a web page against clickjacking attacks by specifying whether a page can be rendered out-of-frame; that is, whether a page can be displayed in an HTML frame. If there are multiple headers, browsers may react to them differently; hence, an attacker could maybe make the page get embedded in a malicious iframe and deceive the user to perform unauthorized actions.

Because of this vulnerability, it's best if the application is set up to only issue one X-Frame-Options header. Also, by using Content Security Policy with the frame-ancestors directive, better control over what sites can frame this content might be achieved.

## Affected components

- **Web Applications:** This affects any web application that controls frame embedding using the X-Frame-Options header. This comprises:
  - websites for e-commerce
  - Applications for banking
  - Systems for managing content (CMS)
- **Web hosts:** This vulnerability could be exacerbated by the server configuration. This problem can be unintentionally introduced by web servers such as Apache, Nginx, or IIS that permit numerous header entries in their response settings.
- **Content Delivery Networks (CDNs):** Depending on how caching and headers are handled, utilizing a CDN to serve content may result in the sending of several X-Frame-Options headers if incorrectly configured.
- **Web Frameworks:** Security headers are implemented by several web development frameworks, including as Django, Ruby on Rails, and Express.js. Incorrect configurations or custom implementations may result in multiple entries.

## Impact Assessment

- **Clickjacking Attacks:** The possibility of clickjacking attacks is the main risk connected to this vulnerability. A malicious iframe can be used by attackers to embed a target webpage, fooling users into engaging with hidden content. Unauthorized actions, like altering account settings, completing transactions, or disclosing private information, could result from this.
- **Loss of User Trust:** Users may stop trusting the website or company if they become victims of clickjacking attacks. Reputational harm, client attrition, and decreased user engagement may arise from this.
- **Compliance Risks:** Permitting clickjacking could result in infractions for businesses that must go by regulations (such as GDPR and PCI DSS). If this vulnerability leads to the compromise of sensitive data, there may be legal consequences or fines.
- **Exploitation of Other Vulnerabilities:** If attackers are able to successfully embed the program in a malicious frame, they may use that information in conjunction with other

vulnerabilities to carry out more complex exploits or escalate attacks, ultimately compromising the system even further.

In general, having several X-Frame-Options header entries might erode an application's security posture, so it's critical to fix this problem to protect user privacy.

## Steps to reproduce

- **Select the Target Application:** Decide the web application to check for vulnerabilities in.
- **Issue a Request:** To inspect the response headers, issue a request to the program using a web browser or a tool.
- **Examine Response Headers:** See if the X-Frame-Options header is present in the response headers. Make a note if more than one entry appears.
- **Test frame embedding** make a basic HTML page that attempts to use a to embed the target application.
- **Observe Behavior:** Check if the program loads inside the iframe by opening the HTML page in a web browser. If it does, it may signal a vulnerability due to conflicting X-Frame-Options headers.

You can determine whether the application is susceptible to the problem with the aid of this process.

## Proof of concept (if applicable)

### Vulnerability scanning using OWASP ZAP

#### Multiple X-Frame-Options Header Entries (1)

▶ GET https://api-prod-digitalgym.aswatson.com/examples/

#### ▪ Request

##### Request

▼ Request line and header section (762 bytes)

```
GET https://api-prod-digitalgym.aswatson.com
/examples/ HTTP/1.1
host: api-prod-digitalgym.aswatson.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64; rv:125.0) Gecko/20100101 Firefox/125.0
pragma: no-cache
cache-control: no-cache
referer: https://api-prod-digitalgym.aswatson.com/
Cookie:
ak_bmsc=0C33ED3541A2897A0568193C74DF4DE9~000000000000
00000000000000000000~YAAQF6bWfSI8TDGSAQAAK041QhkGo1hU5g
TU91V77p4FgdMFEfpU7UdBaA5HzqovLZJWjFaRYUmNr9f0cSSDZyx
uIeWPgadea0qbbFqMYsBUY7CnhN/H7Wx5q3Vf5zJykXgbYpcdY1s
29D7a/Z+qlmnWeWPkSa+Y
/uGwstxKva7AnMwjbr89JEJh5wWz5EbxY9i+KEtaeRvZU6MB9UqCm
qbLBBVxRoHtRJ1rX6aX987xCbaxqxpqBTFzWyMk3T
/ft9uSykAFY0cTDMrPrFG2IEEtjV5AKraT2H0rUVFN5ySbDIPnr+k
2hZely0pETP+qi fDkpc7s1s2LgxmwJAm0ljuTxCdZy+DG
/u00LgF4m10mgibLu89Kif/LbWUOW==
```

▼ Request body (0 bytes)

## ▪ Response

**Response**

▼ Status line and header section (661 bytes)

HTTP/1.1 200 OK  
Server: Apache  
X-Frame-Options: SAMEORIGIN, SAMEORIGIN  
X-Frame-Options: DENY  
X-Content-Type-Options: nosniff  
X-XSS-Protection: 1; mode=block  
ETag: W/"1126-1638455407000-gzip"  
Last-Modified: Thu, 02 Dec 2021 14:30:07 GMT  
Content-Type: text/html; charset=UTF-8  
Cache-Control: max-age=0  
Expires: Mon, 30 Sep 2024 09:14:44 GMT  
X-Akamai-Transformed: 9 643 0 pmb=mRUM,2  
Date: Mon, 30 Sep 2024 09:14:44 GMT  
Content-Length: 5749  
Connection: keep-alive  
Server-Timing: cdn-cache; desc=MISS  
Server-Timing: edge; dur=83  
Server-Timing: origin; dur=7  
Server-Timing: ak\_p;  
desc="1727687684775\_2111219223\_279699100\_9230\_9057\_34  
7\_0\_-";dur=1

► Response body (5749 bytes)

## Vulnerability finding using nmap

### ▪ Used command

- nmap api-prod-digitalgym.aswatson.com

```
(malmi@kali)-[~/Desktop]
$ nmap api-prod-digitalgym.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-30 14:30 EDT
Nmap scan report for api-prod-digitalgym.aswatson.com (125.214.166.34)
Host is up (0.073s latency).
Other addresses for api-prod-digitalgym.aswatson.com (not scanned): 125.214.166.27
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https
```

### ▪ Founded open port

- 80/tcp open http

- 443/tcp open https

## Vulnerability finding using nikto

### ▪ Command

- nikto -h api-prod-digitalgym.aswatson.com

```
(malni@kali)-[~/Desktop]
$ nikto -h api-prod-digitalgym.aswatson.com

- Nikto v2.5.0

+ Multiple IPs found: 125.214.166.27, 125.214.166.34
+ Target IP: 125.214.166.27
+ Target Hostname: api-prod-digitalgym.aswatson.com
+ Target Port: 80
+ Start Time: 2024-09-30 14:24:14 (GMT-4)

+ Server: AkamaiGHost
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Uncommon header 'server-timing' found, with multiple values: (cdn-cache; desc=HIT,edge; dur=1,ak_p; desc="1727720696807_2111219230_401960730_10_10740_211_0_-";dur=1,).
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Root page / redirects to: https://api-prod-digitalgym.aswatson.com/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /: Cookie ak_bmsc created without the httponly flag. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
```

## Key Findings from the Nikto Output

- The anti-clickjacking X-Frame-Options header is not present.
- The X-Content-Type-Options header is not set.

## Proposed mitigation or fix

- **Single Header Configuration:** Verify that the HTTP response from your web application has only one X-Frame-Options header. To get rid of any duplicate headers, check the server and application settings.
- **Apply Content Security Policy (CSP):** To determine which sources are permitted to embed your content in frames, use the Content Security Policy in conjunction with the frame-ancestors directive. This method is regarded as a best practice and provides more flexibility.
- **Testing and Validation:** Make sure the application is fully tested to make sure the answers contain only the intended header after making configuration modifications. Use security scanning software to verify that the vulnerability has been fixed.



- **Frequent Security Reviews:** To avoid similar problems in the future, audit the headers and configurations of your application on a frequent basis.