ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	1
Low	4
Informational	0

Alert Detail

High (Medium)	**SQL Injection**
Description	SQL injection may be possible.
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	submit
Attack	Login AND 1=1 --
URL	https://dat-250-test4.herokuapp.com/register.html
Method	POST
Parameter	submit
Attack	Sign Up' AND '1'='1' --
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	password
Attack	ZAP' AND '1'='1' --
Instances	3
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply a 'whitelist' of allowed characters, or a 'blacklist' of disallowed characters in user input. Apply the principle of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Other information	The page results were successfully manipulated using the boolean conditions [Login AND 1=1 --] and [Login AND 1=2 --] The parameter value being modified was NOT stripped from the HTML output for the purposes of the comparison Data was returned for the original parameter. The vulnerability was detected by successfully restricting the data originally returned, by manipulating the parameter
Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Source ID	1
Medium (Medium)	**X-Frame-Options Header Not Set**
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	GET
Parameter	X-Frame-Options
URL	https://dat-250-test4.herokuapp.com/
Method	GET
Parameter	X-Frame-Options
URL	https://dat-250-test4.herokuapp.com/register.html
Method	GET
Parameter	X-Frame-Options
URL	https://dat-250-test4.herokuapp.com/index.html
Method	GET
Parameter	X-Frame-Options
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	X-Frame-Options
URL	https://dat-250-test4.herokuapp.com/register.html
Method	POST
Parameter	X-Frame-Options
Instances	6
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	16
WASC Id	15
Source ID	3
Low (Medium)	**Cookie Without SameSite Attribute**
Description	A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/register.html
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/index.html
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/register.html
Method	POST
Parameter	session
Evidence	Set-Cookie: session
Instances	6
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.
Reference	https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site
CWE Id	16
WASC Id	13
Source ID	3
Low (Medium)	**Cookie Without Secure Flag**
Description	A cookie has been set without the secure flag, which means that the cookie can be accessed via unencrypted connections.
URL	https://dat-250-test4.herokuapp.com/index.html
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/register.html
Method	POST
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/register.html
Method	GET
Parameter	session
Evidence	Set-Cookie: session
URL	https://dat-250-test4.herokuapp.com/
Method	GET
Parameter	session
Evidence	Set-Cookie: session
Instances	6
Solution	Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.
Reference	https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/02-Testing_for_Cookies_Attributes.html
CWE Id	614
WASC Id	13
Source ID	3
Low (Medium)	**Incomplete or No Cache-control and Pragma HTTP Header Set**
Description	The cache-control and pragma HTTP header have not been set properly or are missing allowing the browser and proxies to cache content.
URL	https://dat-250-test4.herokuapp.com/register.html
Method	GET
Parameter	Cache-Control
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	GET
Parameter	Cache-Control
URL	https://dat-250-test4.herokuapp.com/static/style.css
Method	GET
Parameter	Cache-Control
Evidence	public, max-age=43200
URL	https://dat-250-test4.herokuapp.com/register.html
Method	POST
Parameter	Cache-Control
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	Cache-Control
URL	https://dat-250-test4.herokuapp.com/
Method	GET
Parameter	Cache-Control
URL	https://dat-250-test4.herokuapp.com/index.html
Method	GET
Parameter	Cache-Control
Instances	7
Solution	Whenever possible ensure the cache-control HTTP header is set with no-cache, no-store, must-revalidate, and that the pragma HTTP header is set with no-cache.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-caching
CWE Id	525
WASC Id	13
Source ID	3
Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	POST
Parameter	X-Content-Type-Options
URL	https://dat-250-test4.herokuapp.com/register.html
Method	POST
Parameter	X-Content-Type-Options
URL	https://dat-250-test4.herokuapp.com/register.html
Method	GET
Parameter	X-Content-Type-Options
URL	https://dat-250-test4.herokuapp.com/
Method	GET
Parameter	X-Content-Type-Options
URL	https://dat-250-test4.herokuapp.com/login_page.html
Method	GET
Parameter	X-Content-Type-Options
URL	https://dat-250-test4.herokuapp.com/index.html
Method	GET
Parameter	X-Content-Type-Options
URL	https://dat-250-test4.herokuapp.com/static/style.css
Method	GET
Parameter	X-Content-Type-Options
Instances	7
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scanner will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx https://owasp.org/www-community/Security_Headers
CWE Id	16
WASC Id	15
Source ID	3