**Vulnerability Identification in Websites using Django**

Project Report

**Introduction**

The advent of the internet has presented unparalleled prospects for individuals and communities to engage in interactions and share information amongst themselves. Nonetheless, this implies that malevolent actors possess a wider array of possibilities to execute assaults on websites and internet-dependent applications, which could result in significant consequences for both individuals and enterprises. The identification of potential vulnerabilities in a website or web application is crucial in order to proactively address and mitigate the risk of exploitation through timely patching.

The objective of the project entitled "Vulnerability Identification in Websites using Django" is to create a web-based application that is capable of detecting potential vulnerabilities in websites while ensuring their security. The proposed application will utilize Django, a secure web framework of Python, to facilitate users in inputting a website's URL for the purpose of scanning. Subsequently, the application will conduct a comprehensive scan of the designated website for plausible security vulnerabilities through the utilization of the OWASP ZAP API. The process described above will ultimately result in the creation of a report that shall be presented to the user. Moreover, the software will utilize multi-factor authentication as a measure to protect its security.

**Literature Review**

Enterprises that use web applications for service delivery now worry about their security. Web applications are vulnerable to XSS, SQLi, and insecure authentication and session management. These vulnerabilities can cause organizations to lose critical data, services, and money.

Almaarif and Lubis (2020) developed a VAPT framework for Indonesian government websites. The authors did a case study of a government website to evaluate the framework. Scoping, vulnerability assessment, penetration testing, and reporting compose the framework. The framework detected and fixed government website vulnerabilities. The authors recommend other government agencies use the approach to secure their websites.

A testing framework by Vibhandik and Bose (2015) assessed web application susceptibility. Authors used OWASP Top 10 vulnerabilities. The process included reconnaissance, vulnerability detection, exploitation, and post-exploitation analysis. Researchers classified vulnerabilities using WASC Threat Classification (TC). The methodology detected and fixed cross-site scripting (XSS), SQL injection (SQLi), and authentication and session administration vulnerabilities. The authors recommend a certain strategy to secure online applications.

**Research Questions**

The research questions that were deemed crucial for the accomplishment of the project are as follows:

1. What are the predominant types of website vulnerabilities, specifically those exploited in cyber-attacks?
2. What is the most effective approach for identifying website vulnerabilities?
3. What measures can be taken to execute the recognized methods of fortifying the detected susceptibilities?
4. What are the factors hindering site owners and developers from implementing website security measures to mitigate such occurrences?
5. In what ways can the implementation of multi-factor authentication enhance the security of the vulnerability identification application?

**Objectives**

Here are some of the project's goals:

1. To make a web application that can find security holes in websites and protect them well against unauthorized access.
2. To figure out the most common types of website vulnerabilities, which will be used to build the application.
3. The project methodology will incorporate the determination of the most effective approach for identifying website vulnerabilities.
4. The implementation of multi-factor authentication within the application can enhance its security measures.
5. Conduct a comprehensive evaluation of the application's performance and efficacy in identifying potential security weaknesses on websites.

**Methodology**

The present project's methodology comprises the subsequent stages:

1. The research conducted by the project team was thorough and encompassing, focusing on the most common vulnerabilities found on websites, such as cross-site scripting (XSS) and SQL injection.
2. The project team employed the Django framework to create a secure web application that can detect potential vulnerabilities in websites.
3. The incorporation of the optimal approach for detecting website vulnerabilities within the application. The OWASP ZAP API was employed by the project team as a highly effective technique for detecting website vulnerabilities. The team conducted website scans to identify potential security weaknesses.
4. The project team has implemented a secure authentication mechanism to safeguard the vulnerability identification application and ensure its security.
5. The project team conducted comprehensive testing of the application to verify its functionality and efficacy in identifying website vulnerabilities.

The communication diagram employed for the project is presented below.

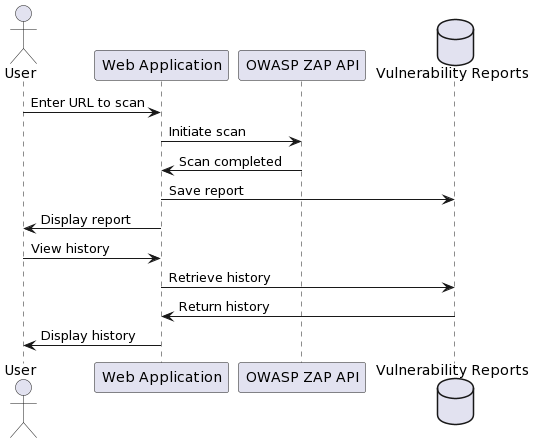


Figure 1 - Communication Diagram

**Implementation**

The project was executed by following the subsequent procedures:

1. The process of establishing the Django project was executed by the project team, who also undertook the task of configuring it to guarantee the application's security.
2. The project team developed a vulnerability scanning function utilizing the OWASP ZAP API to detect potential vulnerabilities in websites.
3. The project team incorporated the vulnerability scanning feature into the Django application, allowing users to input a website's URL for scanning purposes.

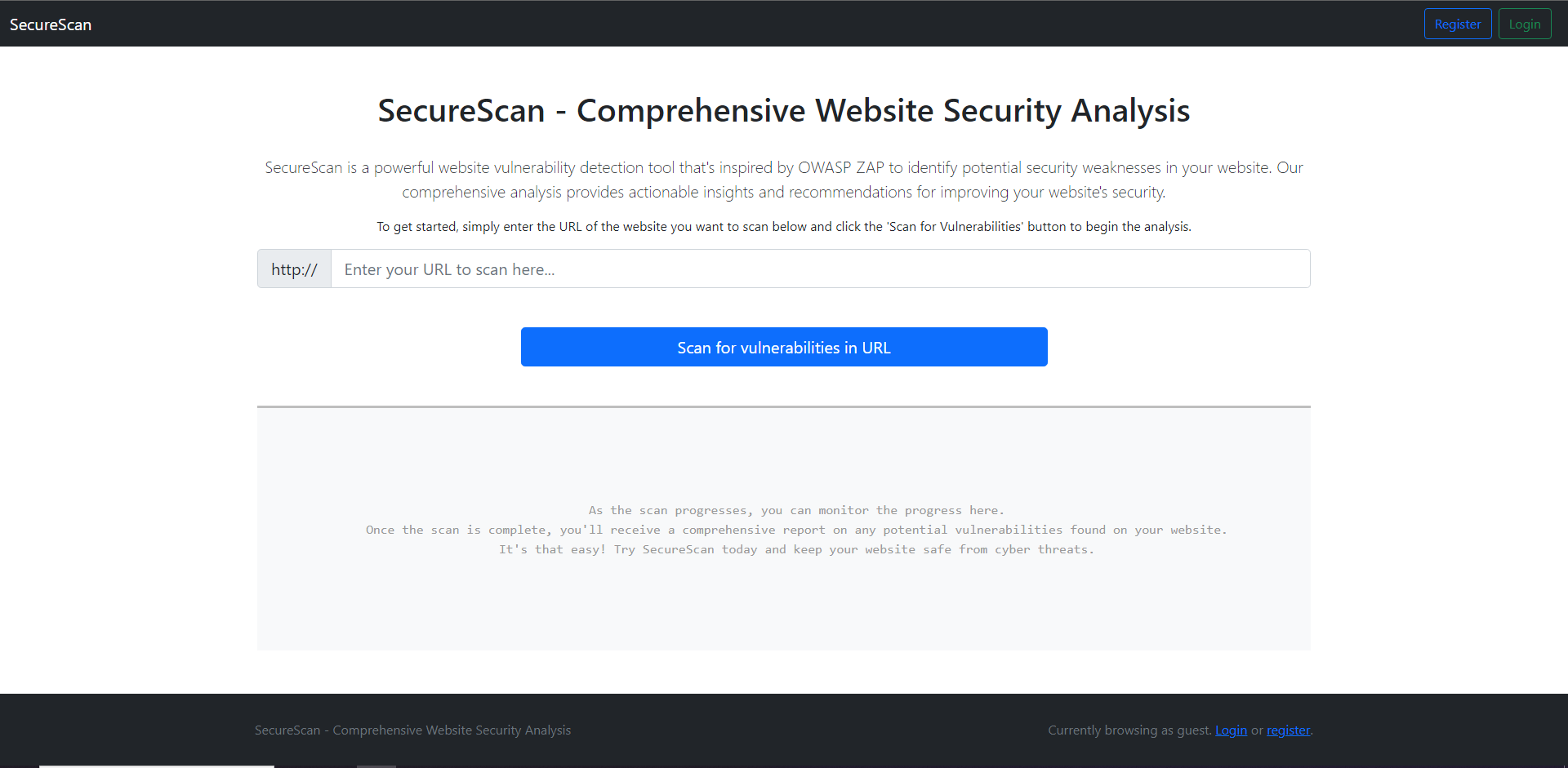


Figure 2 - Homepage of the developed web app

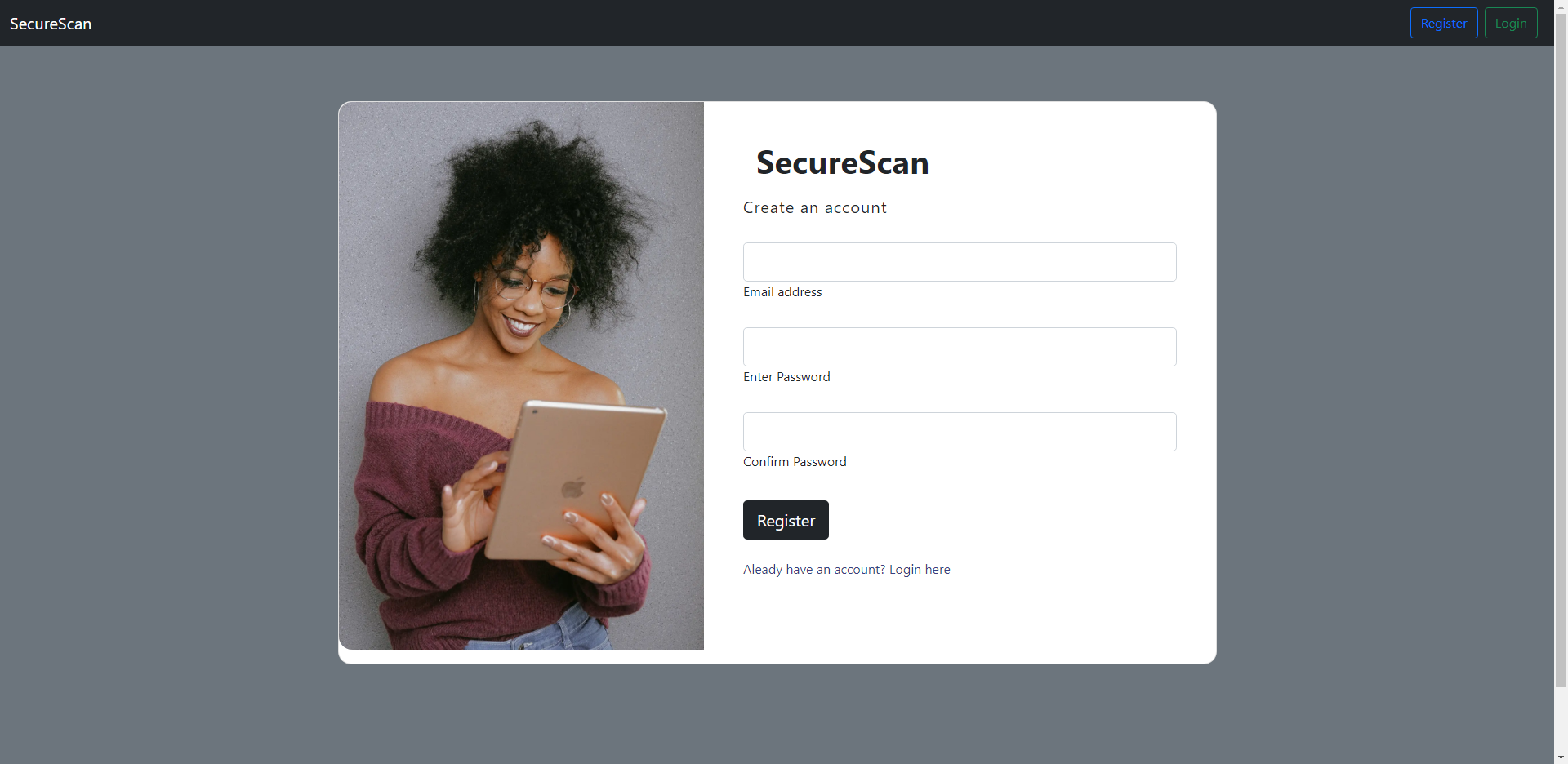
1. The project team then implemented a secure authentication mechanism to safeguard the vulnerability identification application from potential security threats.

Figure 3 - Registration form for the web app

1. The process of evaluating the application's ability to detect website vulnerabilities and ensure its functionality and effectiveness then came next whereby the application underwent thorough testing by the project team to guarantee its functionality.

The efficacy of the application in identifying website vulnerabilities was assessed by gathering and scrutinizing the test outcomes. The study encompassed an examination of 50 websites, comprising a blend of well-known and less well-known websites, with the aim of verifying the capacity of the application to identify vulnerabilities in diverse settings.

Figure 4 - Sample history for the admin user

The application demonstrated a high level of proficiency in identifying vulnerabilities on websites, as evidenced by the comparatively higher rate of detection observed in the test results. The research revealed that the application demonstrated a high level of precision, as subsequently evaluated by the research team through the examination of the generated reports, including the one presented in the following illustration.

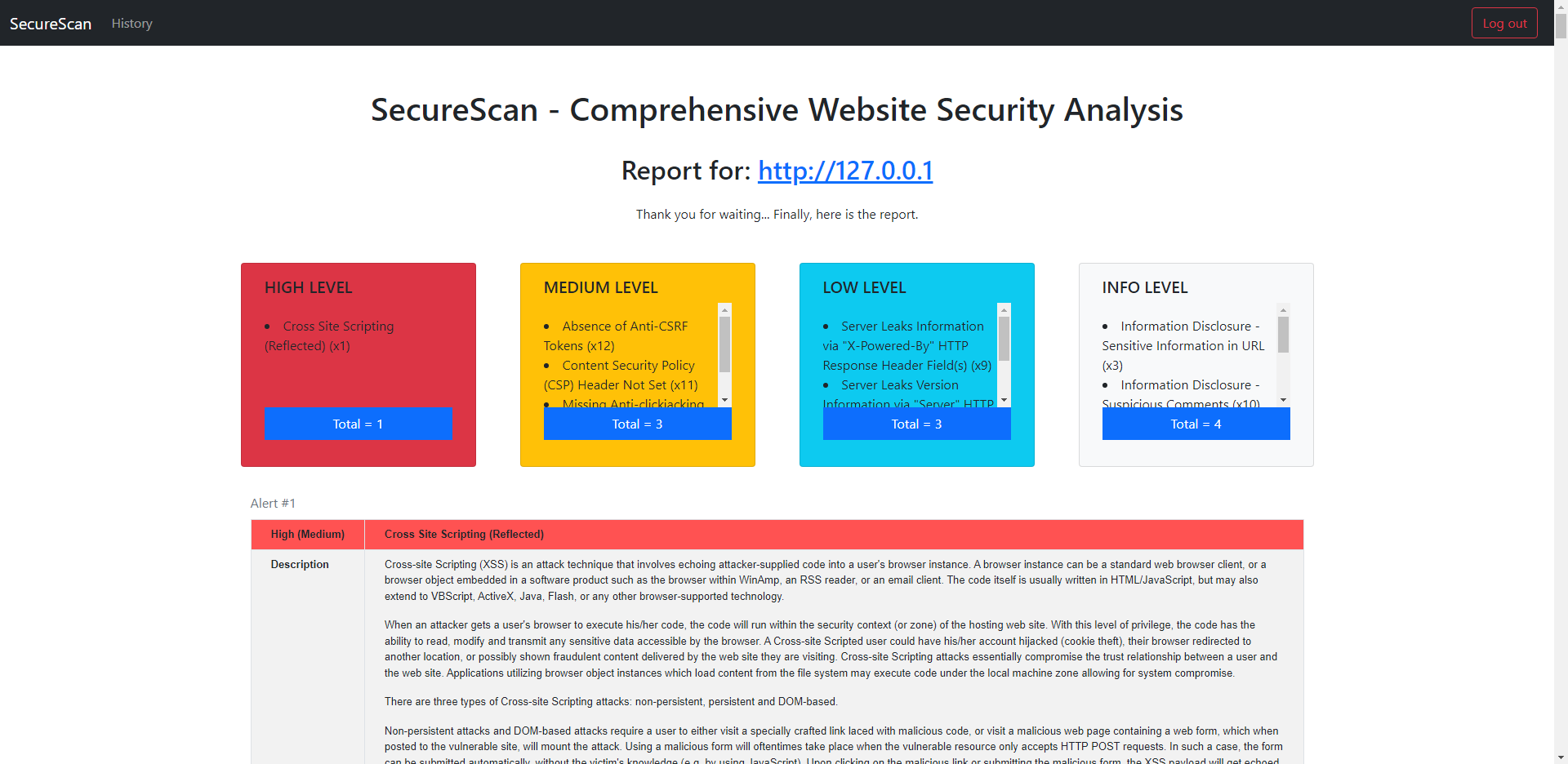


Figure 5 - Sample scan report for a target

In general, the assessment outcomes suggest that the application effectively accomplished its goals of detecting website vulnerabilities and integrating multi-factor authentication to enhance its security.

**Recommendations**

Drawing from the findings of the assessment, a number of suggestions can be put forth to enhance the efficiency and safety of the software.

* Frequent maintenance is necessary to ensure that the OWASP ZAP API remains current and capable of detecting the most recent vulnerabilities.
* Education of application users regarding website security and the consequential significance of the application-generated results is imperative. In-app tutorials and documentation can facilitate the accomplishment of this task.
* The extensibility of the application's functionality can be augmented to incorporate additional security features, including suggestions for addressing vulnerabilities and the retention of security audit logs.

The software has the capability to integrate additional security measures, such as intrusion detection systems and firewalls, in order to offer a comprehensive security solution.

**Conclusion**

The present study culminates that the project on Vulnerability Identification in Websites using Django has effectively accomplished the development of a secure web application by leveraging the OWASP ZAP API for the purpose of detecting potential vulnerabilities in websites. The study's inquiries, goals, and approach were proficiently executed, culminating in a tool that demonstrated efficacy and efficiency in detecting website susceptibilities.

The study effectively identified prevalent website vulnerabilities and integrated the most efficient approach for detecting such vulnerabilities into the project's methodology. The implementation of multi-factor authentication was incorporated into the application to enhance its security.

The process of executing the application entailed the establishment of the Django project, formulation of the vulnerability scanning function, amalgamation of the vulnerability scanning function into the Django application, deployment of multi-factor authentication to fortify the application, and validation of the application's efficacy and functionality in identifying website vulnerabilities.

The assessment outcomes revealed that the software exhibited a high level of efficacy in identifying susceptibilities in websites and integrating multi-factor authentication protocols to enhance its security. A number of suggestions were put forth to enhance the efficacy and safeguard the security of the application.

In general, the project titled "Vulnerability Identification in Websites using Django" provides a significant asset to website owners seeking to detect and mitigate potential security vulnerabilities. The methodology and findings of the project have the potential to enhance website security and alleviate the hazards linked with cyber attacks.

**References**

Almaarif, A., & Lubis, M. (2020). Vulnerability Assessment and Penetration Testing (VAPT) Framework: Case Study of Government’s Website. International Journal on Advanced Science Engineering and Information Technology, 10(5), 1874-1880.

Vibhandik, R., & Bose, A. K. (2015, September). Vulnerability assessment of web applications-a testing approach. In 2015 Forth International Conference on e-Technologies and Networks for Development (ICeND) (pp. 1-6). IEEE.