DOI: https://doi.org/10.12700/btsz.2025.7.1.77

ASSESSING WEB SECURITY AWARENESS: A SURVEY ON DEVELOPERS' PRACTICES, TOOLS, AND LEARNING PREFERENCES

FEJLESZTŐK WEBBIZTONSÁGI TUDATOSSÁGÁNAK FELMÉRÉSE: GYAKORLAT, ESZKÖZÖK ÉS TANULÁSI PREFERENCIÁK

ČOVIĆ Zlatko[1] – PAPP Zoltán[2] – ČOVIĆ Emili[3]

Abstract

This research assesses participants' awareness and understanding of OWASP Top 10 vulnerabilities, their real-world experience with security risks, and web security practices. By examining security challenges developers face and risk mitigation strategies, the study identifies knowledge gaps and misconceptions. It also explores the adoption of security testing tools, the role of AI-based security solutions, and their effectiveness. Additionally, it investigates developer interest in security education, participation in training programs, and preferred learning methods. By comparing security awareness across demographics, experience levels, and roles, this study provides insights into designing targeted training initiatives to integrate security awareness into software development education.

Keywords

OWASP Top 10, Web Security, Software Development, Security Awareness, Threats, Vulnerabilities

Absztrakt

Ez a kutatás a résztvevők OWASP Top 10 sérülékenységekkel kapcsolatos tudatosságát, biztonsági kockázatokkal szerzett tapasztalatait, valamint webbiztonsági gyakorlatait vizsgálja. Elemzi a fejlesztők által tapasztalt biztonsági kihívásokat és kockázatcsökkentési stratégiáikat, hogy azonosítsa a tudásbeli hiányosságokat és tévhiteket. Emellett feltárja a biztonsági tesztelő eszközök és AI-alapú biztonsági megoldások alkalmazását és hatékonyságát. A kutatás vizsgálja a fejlesztők érdeklődését a webbiztonsági oktatás iránt, képzési programokban való részvételi hajlandóságukat, valamint preferált tanulási módszereiket. Az eredmények hozzájárulnak a célzott képzési programok kialakításához, elősegítve a biztonságtudatosság integrálását a szoftverfejlesztési oktatásba.

Kulcsszavak

OWASP Top10, Web biztonság, Web fejlesztés, Biztonsági tudatosság, Fenyegetések, Sérülékenységek

[1] zlatko.covic@uni-obuda.hu | ORCID: 0000-0002-1769-1990 | University professor, researcher, Óbuda University Doctoral School on Safety and Security Sciences | egyetemi oktató, kutató, Óbudai Egyetem Biztonságtudományi Doktori Iskola; Professor, Assistant Director for Public Relations and Student Affairs, Subotica Tech – College of Applied Sciences, Subotica | oktató, Közkapcsolatokért és Hallgatói Ügyekért Felelős Igazgatóhelyettes, Szabadkai Műszaki Szakfőiskola, Szabadka

[2] zoltan.papp@magister.uns.ac.rs | ORCID: 0000-0001-9589-6580 | Associate professor, University of Novi Sad, Hungarian Language Teacher Training Faculty Subotica; University of Dunaújváros, Institute of Information Technology |egyetemi docens, Újvidéki Egyetem, Magyar Tannyelvű Tanítóképző Kar, Szabadka; Dunaújvárosi Egyetem, Informatikai Intézet

[3] 28224008@vts.su.ac.rs | ORCID: 0009-0003-1977-3062 | MSc student in Information Technology, Subotica Tech – College of Applied Sciences, Subotica | MSc hallgató Információs technológiák szakon, Szabadkai Műszaki Szakfőiskola, Szabadka

INTRODUCTION

Web applications are an essential part of modern digital infrastructure, supporting a wide range of business, communication, and entertainment services. As the adoption of web technologies continues to expand, so do the risks associated with cyber threats targeting web applications. The increasing number of security breaches highlights the importance of secure web development practices. Studies show that more than 60% of breaches involve vulnerabilities in web applications, with common attack vectors including SQL injection (SQLi), cross-site scripting (XSS), and broken authentication mechanisms [1]. Research on e-commerce security has identified SQL injection (SQLi), cross-site scripting (XSS), and security misconfigurations as some of the most frequently exploited vulnerabilities, often resulting in significant business losses [2].

Despite the availability of security frameworks and best practices, many developers still lack awareness or training in secure coding. Research has indicated that security is often deprioritized in favor of functionality and speed, leading to the deployment of vulnerable applications [3]. While organizations invest in security tools such as static and dynamic application security testing (SAST/DAST) tools, their effectiveness relies heavily on developers' understanding of security risks and proper implementation of secure coding practices [4].

Insecure web applications have a direct financial, operational, and reputational impact on businesses. Cyberattacks targeting web applications result in data breaches, financial fraud, loss of customer trust, and legal repercussions due to non-compliance with regulations such as GDPR and CCPA [5].

To address these issues, it is necessary to assess developers' awareness of web security vulnerabilities, their use of security tools, and their learning preferences. This study aims to evaluate these aspects by surveying developers in Subotica and analyzing the findings to gain insight into current security practices. The results will serve as the foundation for designing targeted security training programs, bridge the knowledge gap, and improve web application security.

The rest of this section provides a background on common web security threats, discusses the importance of security awareness among developers, presents an overview of the OWASP Top 10 vulnerabilities, reviews previous studies on the topic, and outlines the research objectives and structure of this paper.

Background on Web Security Threats

The increasing reliance on web-based systems has led to a rise in security threats targeting web applications. Cyberattacks such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF) remain among the most commonly exploited vulnerabilities [6]. According to a report by Verizon, web applications account for over 43% of security breaches, highlighting the critical need for secure coding practices [7].

The rapid development of web technologies has also introduced new attack surfaces, making traditional security measures insufficient. The emergence of AI-driven cyber threats, automated exploitation tools, and large-scale data breaches further complicates the security landscape [8]. Given these challenges, the role of developers in maintaining web application security has become more crucial than ever.

Importance of Security Awareness Among Developers

Developers play a fundamental role in ensuring the security of web applications. However, multiple studies have shown that developers often lack adequate security training, leading to the introduction of vulnerabilities in the code they write [9]. A survey conducted by NIST found that 79% of developers do not receive formal security training, which increases the risk of security flaws in web applications [10].

Security awareness programs and secure coding training have been proposed as effective strategies for mitigating these risks. However, traditional security training methods often fail to engage developers or align with their real-world development practices. Research suggests that interactive and hands-on training methods, such as hackathons, capture-the-flag (CTF) competitions, and security coding challenges, are more effective in improving developers' security awareness and adoption of best practices [11].

Web applications, as public-facing system components, are vulnerable to attacks. Developers often struggle to grasp the full attack surface. A study found that one-third were unaware clients could intercept and modify HTTP requests, with knowledge mostly limited to XSS and SQL Injection. Using a questionnaire and a Capture the Flag challenge, the study highlighted security knowledge gaps, stressing the need for better training on web threats and defenses [12].

Security awareness is crucial in software development, as insufficient knowledge among developers can lead to poor coding practices and exploitable vulnerabilities. Studies indicate that a significant portion of security vulnerabilities in web applications stem from development errors rather than infrastructure weaknesses [12].

To mitigate these risks, integrating security-by-design principles throughout the Software Development Life Cycle (SDLC) is essential. This approach ensures that security is embedded from the outset, rather than addressed later. Key aspects of security awareness programs include:

- Understanding web vulnerabilities: Familiarity with common threats (e.g., OWASP Top 10) helps developers anticipate and prevent attacks. [13]
- Writing secure code: Adhering to standards from organizations like CERT and NIST ensure robust software security [14].
- Implementing security controls: Measures like input validation, access control, and encryption protect sensitive data and system integrity [15].
- Regular code reviews and security testing: Early detection and remediation of issues enhance software security [13].

Security is a shared responsibility across all roles in software development, including junior developers, testers, and DevOps engineers. By fostering a culture of security awareness and providing comprehensive training, organizations can significantly reduce vulnerabilities and enhance the overall security posture of their software products.

Overview of OWASP Top 10 Vulnerabilities

The Open Web Application Security Project (OWASP) publishes the Top 10 most critical web security risks, serving as an industry standard for secure software development.

[13] The list is regularly updated to address emerging threats, offering key insights for developers, security professionals, and organizations. The latest OWASP Top 10 list includes [16]:

1. Broken Access Control – Unauthorized access resulting from weak or improperly enforced permission policies. This vulnerability is responsible for the highest number of web application breaches.
2. Cryptographic Failures – Improper handling of sensitive data, such as weak encryption algorithms, exposure of passwords, or transmission of confidential information in plaintext, leading to data breaches.
3. Injection Attacks (SQLi, XSS, LDAP, etc.) – Maliciously crafted inputs that exploit poor input validation, allowing attackers to execute unauthorized queries, inject scripts, or manipulate backend systems.
4. Insecure Design – Fundamental flaws in the architecture or logic of an application that introduce security risks, emphasizing the need for security-by-design principles.
5. Security Misconfiguration – Default settings, excessive privileges, and unnecessary features that expose applications to exploitation, making them easier targets for attackers.
6. Vulnerable and Outdated Components – Use of unpatched libraries, outdated frameworks, or unsupported software that increases the risk of exploitation through known vulnerabilities.
7. Identification and Authentication Failures – Weak password policies, improper session management, and lack of multi-factor authentication (MFA) leading to account takeovers and identity theft.
8. Software and Data Integrity Failures – Risks associated with insecure deployment, lack of integrity verification in software updates, and unauthorized modification of critical application data.
9. Security Logging and Monitoring Failures – Insufficient or improperly configured logging mechanisms that prevent timely detection of security incidents, making applications vulnerable to prolonged undetected attacks.
10. Server-Side Request Forgery (SSRF) – Attackers exploiting server-side functionality to access or manipulate internal resources, leading to unauthorized data exposure and system compromise.

To mitigate these vulnerabilities, OWASP recommends implementing the best security practices such as secure authentication mechanisms, strict access control policies, and the use of parameterized queries to prevent injection attacks [17]. Additionally, organizations should adopt secure software development practices, conduct regular security assessments, and integrate automated security testing tools to identify and remediate vulnerabilities throughout the Software Development Life Cycle (SDLC).

Examples of Previous Studies

Several studies have explored developers' security practices, tools, and learning preferences. For example, a systematic review of secure coding practices found that developers often struggle with integrating security measures into their workflow due to time constraints and a lack of formal training [18]. Another study analyzed the effectiveness of security tools such as SAST and DAST scanners, revealing that many developers are unaware of how to properly configure and use these tools [19].

Additionally, empirical research has examined the impact of security awareness programs on software development teams. Findings suggest that security culture and peer learning play a significant role in shaping developers' security behaviors [20]. However, gaps remain in understanding how different experience levels and backgrounds affect security awareness, which this study aims to address.

Ensuring secure software development is crucial for maintaining integrity, confidentiality, and availability, yet security is often overlooked in favor of functionality. Research emphasizes the need to integrate security at every phase of the Software Development Life Cycle (SDLC), as many existing methodologies fail to do so effectively. The findings highlight the need for enhanced security practices to overcome adoption barriers and establish a structured approach to secure software engineering [21].

The authors in [19] present a swarm intelligence-based Orchestrated Continuous Vulnerability Assessment (OCVA) scanning tool to address the limitations of traditional security detection methods, such as signature recognition and anomaly detection, which often miss sophisticated cyber threats. [19] highlights the increasing need for continuous vulnerability assessment to improve security monitoring, analysis, and mitigation across networks, assets, and web applications. Comparative case studies show that OCVA outperforms existing vulnerability scanners in detection accuracy, remediation rates, and consistency. These findings suggest that OCVA provides a more efficient and reliable solution for developers and security auditors in mitigating cybersecurity risks.

In the paper [22], the authors emphasize the importance of improving employee security awareness to mitigate cybersecurity risks. They propose the Security Awareness Improvement Tool (SAWIT), designed to assess and enhance individual security practices within organizations. The study highlights the significance of continuous education and proactive engagement in fostering a security-conscious workplace culture, ultimately aiming to minimize security vulnerabilities linked to human factors in cybersecurity.

RESEARCH OBJECTIVES

This article aims to assess developers' awareness of web security vulnerabilities, their security practices, and their learning preferences. Specifically, the research questions are:

Q1: OWASP Top knowledge and experience:

- What proportion of developers are familiar with OWASP Top 10 security risks?
- How do developers perceive their own knowledge of OWASP Top 10 vulnerabilities?
- In real-world development, which OWASP Top 10 categories are encountered by the developers?

Q2: How do developers' education level, experience, seniority, and role influence their awareness and adoption of web security practices and OWASP Top10?

Research Methodology

This study is based on a non-representative online survey conducted in January 2025. The survey aimed to assess software developers' web security awareness, particularly focusing on OWASP Top 10 vulnerabilities, security tools, developers' practices, and AI-based security solutions. Additionally, the research aims to understand their interest in web security education and preferred learning methods. The questionnaire was designed by the authors and distributed via Google Forms.

Variables

To address research questions Q1 and Q2, the authors identified and analyzed a set of independent and dependent variables. These variables represent a specific subset of the questionnaire, designed to directly contribute to answering Q1 and Q2. By focusing on these targeted variables, the analysis aims to provide meaningful insights into the factors influencing web security awareness, OWASP Top 10 knowledge, and the adoption of secure development practices.

The key groups of independent variables used in the analysis pertain to demographics and professional background. These variables help assess whether factors such as education, work experience, seniority, and job role influence knowledge of web security and the OWASP Top 10, as well as the adoption of secure development practices.

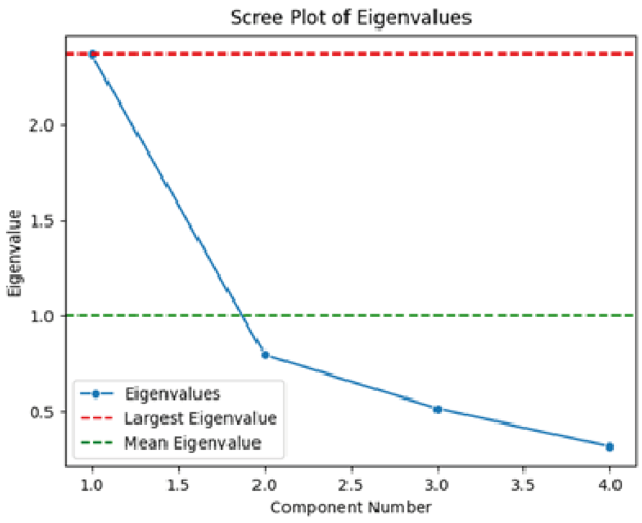
The key dependent variables analyzed in this study include awareness and experience with the OWASP Top 10, responsibility for secure web development, and the frequency of using web security assessment tools. These variables help assess the extent to which developers are knowledgeable about web security best practices, their role in ensuring secure development, and their regular engagement with security evaluation tools.

Internal consistency reliability of the questionnaire

To assess the internal consistency reliability of the questionnaire related to research questions Q1 and Q2, the authors categorized the relevant survey questions into two distinct groups, each designed to measure a specific underlying construct. The first group includes questions related to demographics and professional background, capturing factors such as education, work experience, seniority, and job role, while the second group focuses on OWASP Top 10 knowledge and the adoption of secure development practices, assessing participants' awareness, experience, and engagement with web security principles. Since internal consistency reliability is only meaningful for items measuring the same latent construct, it was evaluated exclusively for the second group, as demographic and background variables represent independent characteristics rather than a unified scale. The analysis was conducted using Python, applying appropriate statistical methods to ensure the reliability of the measured constructs.

For the second question group, McDonald's omega [23], was chosen as the reliability measure due to the mixed nature of the data. This group includes binary, ordinal, and linear scale data, making McDonald's omega a more suitable alternative to Cronbach's alpha [24], which assumes continuous or ordinal data with equal variances. This approach ensures a more accurate assessment of internal consistency across the diverse data types present in the survey. Since many statistical methods do not natively support categorical data, the authors transformed it into numerical form to facilitate analysis [25]. For binary data, such as yes/no responses, they applied the One-Hot Encoding technique in Python.

For ordinal data, where categories have a meaningful order, the authors used the Ordinal Encoding method. By applying these encoding methods, the authors ensured that the data could be effectively analyzed using statistical techniques while maintaining the integrity of the categorical variables. To determine the number of latent factors, the authors conducted an Exploratory Factor Analysis (EFA). However, before performing EFA, they assessed the sampling adequacy of each observed variable in the model using the Kaiser-Meyer-Olkin (KMO) test [26]. The KMO statistic is calculated based on the correlation between variables and ranges from 0 to 1. Higher values, closer to 1, indicate strong correlations among variables, suggesting that the data is well-suited for factor analysis. For the research question group related to Q2, the KMO value was $KMO = 0.7942$, indicating an acceptable level of sample adequacy, which is considered average to good for factor analysis. Additionally, the authors conducted Bartlett's test of sphericity [27] to determine whether the correlation matrix was significantly different from an identity matrix. A significant result from Bartlett's test suggests that there are sufficient correlations among the variables, justifying the application of factor analysis. In this study, Bartlett's test for the Q2 research question group yielded significant results ($p \approx 0.$), confirming that the variables were adequately correlated to proceed with factor analysis. The scree plot generated from the EFA is presented in Fig. 1 and indicates the presence of a single latent factor.



1. Figure: Number of latent factors for the second question group

A scree plot is a graphical representation of eigenvalues associated with each factor or principal component, with eigenvalues plotted on the y-axis and factors/components on the x-axis, arranged in descending order. The primary goal of this visualization is to identify the "elbow" point, where the slope of the curve significantly levels off. This point suggests a natural cutoff for the number of factors or components to retain in the analysis. Given the presence of a single latent factor, the authors calculated McDonald's omega ($\omega = 0.6925$) to assess internal consistency reliability. This value indicates moderate reliability—acceptable for exploratory research but with room for improvement.

Sample

To ensure a broader reach and higher response rate, the survey link was initially sent to company directors, who were requested to share it with their employees. Recognizing the multilingual environment of Subotica, the questionnaire was made available in both Serbian and Hungarian, allowing respondents to complete it in their preferred language. A total of 51 programmers, all employed by software development companies in Subotica, Serbia, participated in the study. All developers responded to every question in the questionnaire. The collected data contains no missing values.

The sample distribution across variables related to demographics and professional background is presented in Table 1.

Variable		Count	Percentage
education	high school	6	11.76
	BSc	36	70.59
	MSc	9	17.65
experience (year)	<1	3	5.88
	1-3	11	21.57
	4-6	9	17.65
	7-10	10	19.61
	10>	18	35.29
seniority	junior	9	17.65
	mid-level	18	35.29
	senior	24	47.06
role	frontend developer	11	12.09
	backend developer	23	25.27
	full stack developer	15	16.48
	mobile app developer	6	6.59
	software architect	7	7.69
	tech lead	11	12.09
	project manager	7	7.69
	DevOps	6	6.59
	other	5	5.50

1. Table: Sample distribution related to demographics and professional background

Table 2 presents the distribution of the sample across variables related to awareness and experience with the OWASP Top 10, responsibility for secure web development, and the frequency of using web security assessment tools.

Variable		Count	Percentage
OWASP Top 10 self assessment	I'm not sure	25	49,02
	superficial	20	39,22
	good	6	11,76
OWASP Top 10 experiences in projects	no	22	43.14
	yes	29	56.86
Responsibility for the web secure development	I'm not sure.	3	5.88
	Only security experts.	1	1.96
	The entire team.	47	92.16
Frequency of using security tools.	Never	18	35.29
	Rarely	20	39.22
	Occasionally	10	19.61
	Regularly	3	5.88

2. Table: Sample distribution related to awareness and experience with the OWASP Top 10 and secure web development

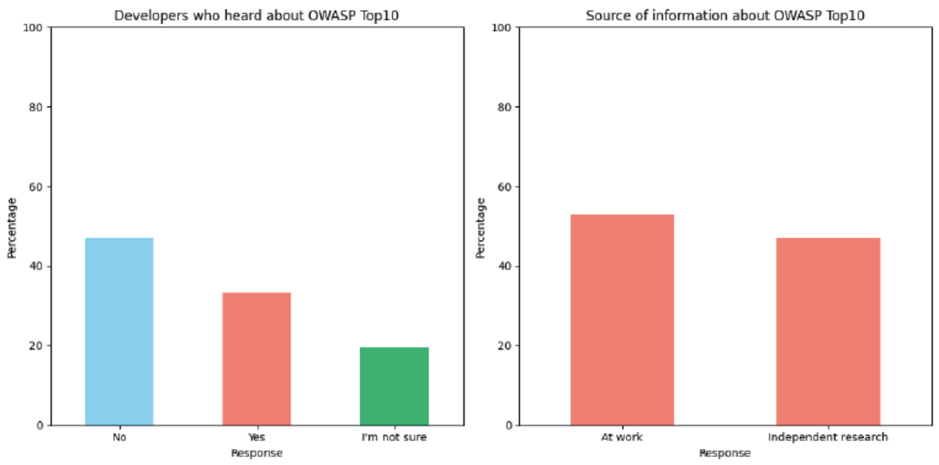
Plan of analysis

For the quantitative analysis of the collected data, the authors utilized the Python programming language. To preprocess the data obtained from the questionnaire, they employed One-Hot Encoding and Ordinal Encoding methods. To calculate internal consistency reliability, the authors first determined the number of latent factors through EFA and scree plot analysis. To justify the application of EFA, they conducted the KMO test and Bartlett's test of sphericity. For measuring internal reliability, the McDonald's omega coefficient was used. To answer the research questions, the authors employed descriptive statistics and correlation. This comprehensive approach ensured the robustness and validity of their findings.

RESULTS

Research question Q1

Fig2. presents the distribution of answers to the question: "What proportion of developers are familiar with OWASP Top 10 security risks?



2. Figure: Developers' awareness of OWASP Top10 and source

Fig. 2 illustrates whether developers have heard about the OWASP Top 10. Approximately 45% of developers have not heard about the OWASP Top 10, about 30% have heard about it, and around 25% are unsure if they have heard about it. Among those who have heard about the OWASP Top 10, approximately 55% learned about it at work, while about 45% gained their knowledge through independent research.

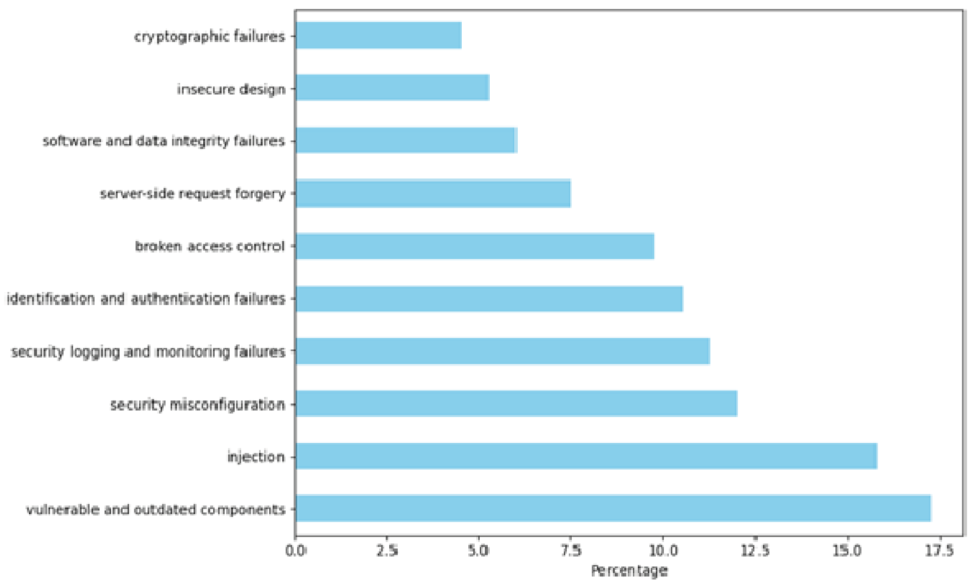
Table 3. gives an answer on the research question "How do developers perceive their own knowledge of OWASP Top 10 vulnerabilities? "

Independent variable		**Count**	**Percentage**
OWASP Top 10 self-assessment	I'm not sure	25	49,02
	superficial	20	39,22
	good	6	11,76

3. Table: Sample distribution related to OWASP Top 10 related security issues in real projects

Most developers are uncertain about their knowledge of OWASP Top 10 vulnerabilities. Additionally, 39.22% of developers assess their understanding as superficial, while only 11.76% believe they have a good grasp of these vulnerabilities.

The research question "In real-world development, which OWASP Top 10 categories are encountered by the developers?", could also help the authors to assess the OWASP Top10 awareness of the developers. Understanding which OWASP Top 10 categories are most encountered in real-world development is crucial for several reasons. By identifying the most frequently encountered security risks, developers and organizations can prioritize their efforts and resources to address the most critical vulnerabilities first. Fig. 3 provides a detailed breakdown of the specific OWASP Top 10 security vulnerabilities that developers have faced in real-world projects. The most prevalent vulnerabilities are related to the use of vulnerable and outdated components, injection flaws, and security misconfigurations.



3. Figure: Source of OWASP Top 10 security issues in real-world development

Research question Q2

Understanding how developers' education level, experience, seniority, and role influence their awareness and adoption of web security practices, including OWASP Top 10, is important for several reasons. By identifying how these factors affect web security awareness, organizations can develop targeted training programs. These programs can be tailored to address the specific needs and gaps of different groups, ensuring that all developers are adequately equipped with the necessary knowledge and skills. Insights into how these factors influence security practices can help organizations allocate resources more effectively.

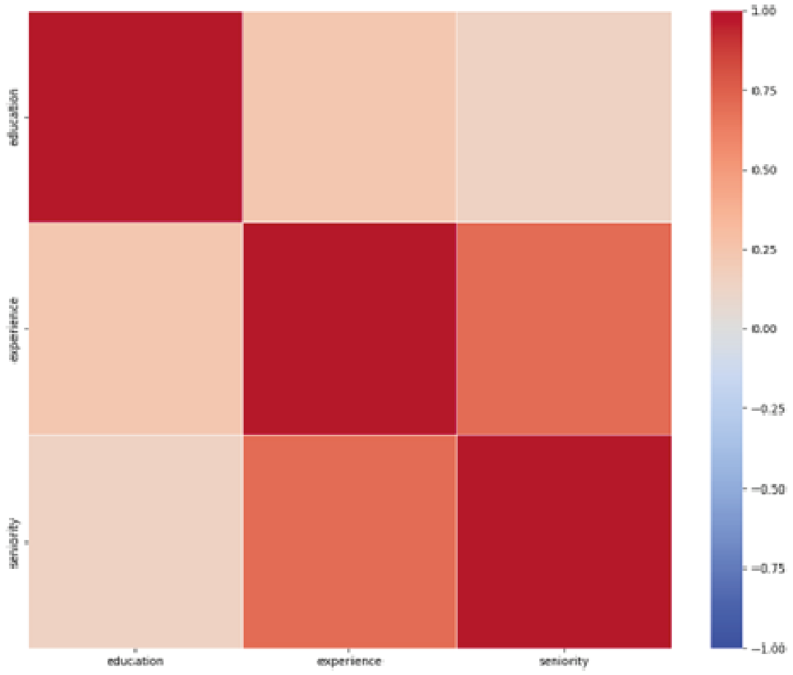
As a measure of the developers' awareness and adoption of web security practices, including OWASP Top 10 the authors considered the following variables.

- Awareness of OWASP Top10: ordinal variable. The variable's sample distribution is presented in Fig. 2.
- Encountering OWASP Top10 security issues: categorical variable, with the possibility of selecting multiple options. The variable's distribution is presented in Fig. 3.
- Responsibility for secure web development: ordinal variable. The distribution of this variable is presented in Table 2.
- Frequency of using security testing tools: ordinal variable. The distribution of this variable is presented in Table 2.

Before performing statistical analysis, the authors preprocessed the data by converting binary and categorical variables to numerical values.

To avoid analyzing all possible pairs of dependent-independent variables, the authors opted to construct a single dependent variable by index construction method which consists of standardization of variables and aggregation of standardized variables. aggregating the dependent variables [28]. This method is particularly useful when measuring complex constructs, such as developers' awareness and adoption of web security practices

The authors initially conducted a correlation analysis and calculated variance inflation factors (VIF) [29] on demographic variables, including developers' education level, experience, and seniority. VIF values greater than 5 are considered indicative of problematic multicollinearity. To prevent potential multicollinearity issues, the authors excluded highly correlated independent variables. The VIF values were calculated, and the Spearman correlation coefficient was employed due to the categorical nature of the data and its non-normal distribution. Fig. 4 presents the correlation matrix using a heatmap. Shades of red indicate a positive correlation. Darker red signifies a stronger positive correlation, closer to 1.00. Shades of blue indicate negative correlations. Darker blue signifies a stronger negative correlation, closer to -1.00. Seniority and experience are highly correlated, which implies multicollinearity. To avoid multicollinearity, the authors excluded experience.



4. Figure: Spearman correlation matrix for demographics data

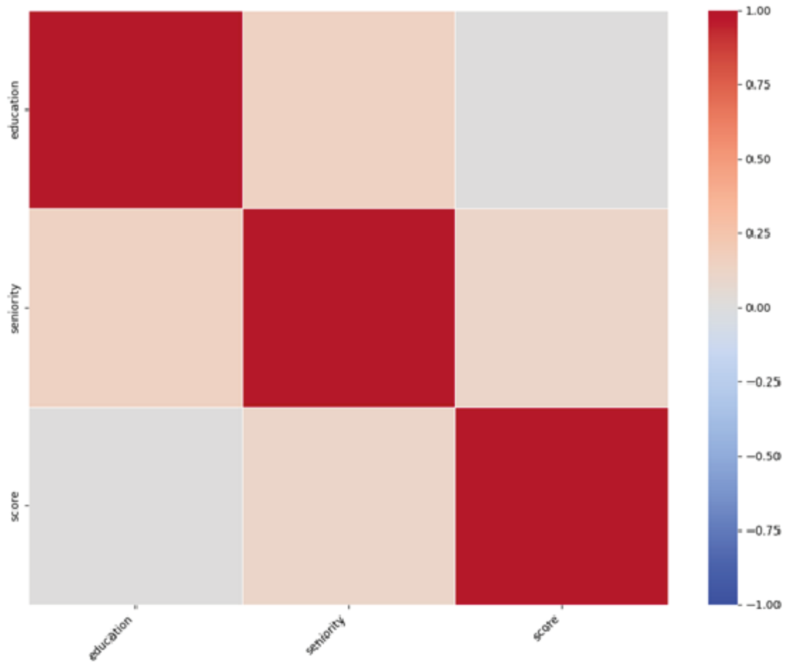
Correlation analysis is employed on dependent variables to identify and exclude variables that don't contribute meaningful information to the index and can even dilute its accuracy. These are variables with weak or no correlation to other dependent variables and are essentially noise. The Spearman correlation matrix is presented in Table 4. The variable representing the frequency of using security testing tools has a low correlation with both the awareness of OWASP Top 10 and the variable measuring encounters with OWASP Top 10 security issues. This suggested the authors exclude this variable.

	OWASP Top10 awareness	Encountering OWASP Top11 security issues	Responsibility for secure web development	Frequency of using security testing tools
OWASP Top10 awareness	1.00	0.27	0.12	0.03
Encountering OWASP Top11 security issues	0.27	1.00	0.33	0.05
Responsibility for secure web development	0.12	0.33	1.00	0.24
Frequency of using security testing tools	0.03	0.05	0.24	1.00

4. Table: Spearman correlation matrix for dependent variables

After excluding experience as an independent variable and the frequency of using security testing tools from the calculation of the composite index score, a correlation analysis was performed. The Spearman correlation matrix in Fig. 5 reveals that there are no strong correlations between any of the independent variables and the dependent composite index score. However, the correlation between seniority and the composite index score is relatively higher. Possible reasons for low correlation could be that education level, years of experience, or seniority may not necessarily translate into better security awareness. Security responsibilities often belong to dedicated security teams, meaning general developers (regardless of seniority) may not engage deeply with security practices.

The authors conducted a correlation analysis using developers' roles as independent variables and the composite index score as the dependent variable. To avoid multicollinearity, they calculated the Spearman correlation matrix and VIF values. The heatmap presented in Fig. 6 revealed some higher correlation values. Based on these findings, the authors decided to exclude the roles of tech lead, head of software development, mobile app developer, and project manager.



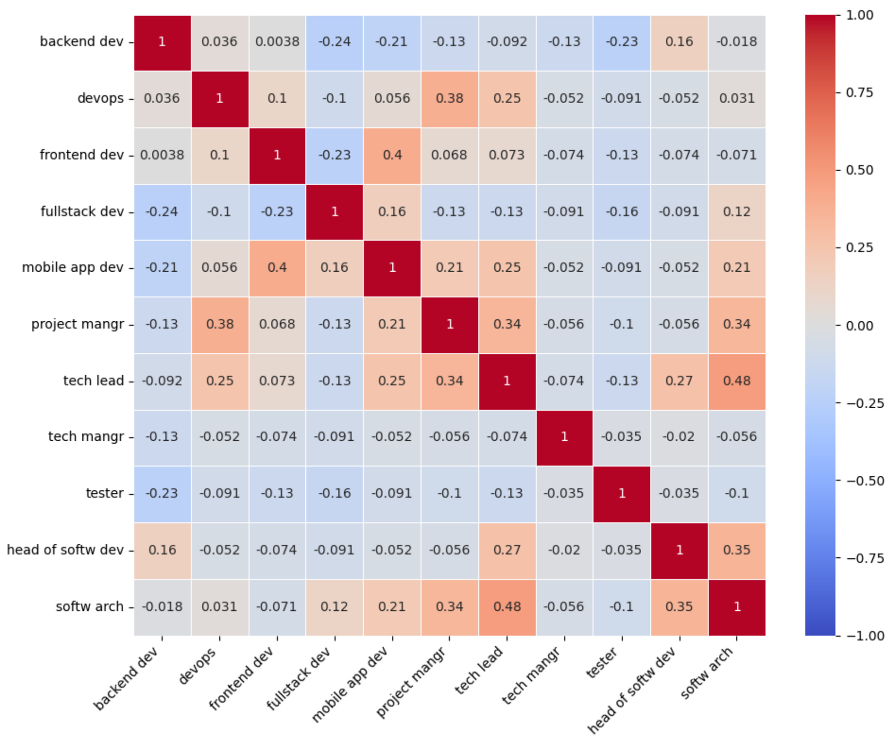
5. Figure: Spearman correlation matrix for demographics and composite index score

After excluding the roles of tech lead, head of software development, mobile app developer, and project manager, the Spearman correlation matrix of developers' roles and the composite index score is presented in Fig. 7. Fig. 7 reveals that the highest correlation is between the DevOps role and the composite index score, with a correlation coefficient of 0.42. This indicates a moderate positive monotonic relationship between the two variables. Possible reasons for this finding include the variation in security responsibilities across different roles and the fact that security knowledge is not a formal requirement for all roles.

CONCLUSIONS

This study provides a comprehensive assessment of developers' awareness and adoption of web security practices, with a particular focus on OWASP Top 10 vulnerabilities. The findings reveal that a significant proportion of developers lack familiarity with OWASP Top 10, with 45% having never heard of it and only 11.76% considering their knowledge to be strong. Additionally, developers frequently encounter security risks such as vulnerable and outdated components, injection flaws, and security misconfigurations in real-world projects, highlighting key areas requiring further training and awareness.

A crucial insight from this research is that formal education, years of experience, and seniority do not strongly correlate with security awareness and adoption.



6. Figure: Spearman correlation matrix for developers' role

The highest correlation (0.42) was observed between the DevOps role and security awareness, suggesting that security responsibilities are often concentrated in specific job roles rather than being uniformly distributed across all developers.

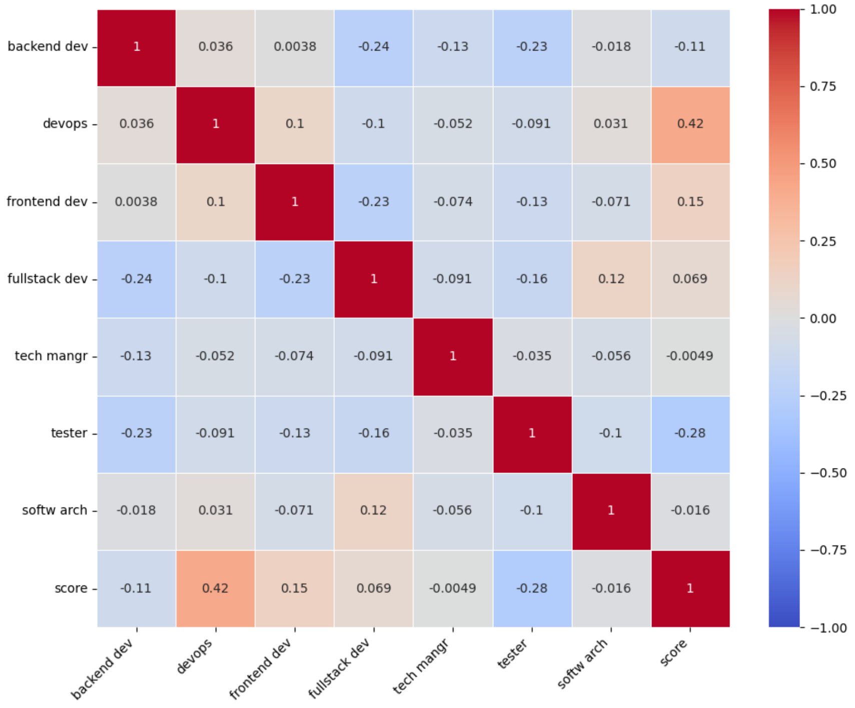
These findings emphasize the need for targeted training initiatives that go beyond general software development education and focus on practical, role-specific security knowledge.

Ultimately, this study underscores the importance of bridging the gap between security awareness and implementation in software development. Future research should explore how different training methodologies impact long-term security behavior and investigate the effectiveness of AI-driven security education tools in improving developers' understanding of web security risks.

Future research should further explore the following objectives:

1. Evaluate developers' understanding of the OWASP Top 10 vulnerabilities and their impact on web security.
2. Analyze the security tools and practices commonly adopted by developers.
3. Assess developers' learning preferences for security education and training.
4. Identify key challenges developers face in implementing secure coding practices.

By addressing these objectives, this study seeks to bridge the gap between security awareness and practical implementation, ultimately contributing to more effective developer-centric security training programs. Based on the feedback received, the plan is to provide developers with a web-based system for additional learning and knowledge assessment in the field of web application security. This system utilizes GPT-based AI models to generate questions and answers from specific categories defined within the OWASP Top 10 standard. The system is a proprietary software solution and has successfully passed functional beta testing.



7. Figure: Spearman correlation matrix for developers' roles

REFERENCES

[1] Verizon, "Data Breach Investigations Report." 2023. [Online]. Available: https://www.verizon.com/dbir/

[2] P. Bhattacharya, "The Critical Analysis of E-Commerce Web Application Vulnerabilities," *Cybersecurity Trends Threats E-Commer.*, 2023, [Online]. Available: https://www.igi-global.com/chapter/the-critical-analysis-of-e-commerce-web-application-vulnerabilities/325544

[3] R. Chatterjee, S. Egelman, and S. Consolvo, "The usability of secure coding practices: A review of security training for developers," *ACM Trans. Priv. Secur.*, vol. 25, no. 1, pp. 1-29, 2022, doi: 10.1145/3519601.

[4] W. Charoenwet, P. Thongtanunam, V.-T. Pham, and C. Treude, "Toward Effective Secure Code Reviews: An Empirical Study of Security-Related Coding Weaknesses," *IEEE Trans. Softw. Eng.*, 2023, doi: 10.1109/TSE.2023.1234567.

[5] A. Khan and A. Mustafa, "Survey of Websites and Web Application Security Vulnerabilities," *J. Comput. Sci.*, vol. 14, no. 1, pp. 25-40, 2018.

[6] O.W.A.S.P., "OWASP Top 10 – 2021: The Ten Most Critical Web Application Security Risks." 2021. [Online]. Available: https://owasp.org/www-project-top-ten/

[7] N.I.S.T., "The State of Software Security Training." 2023. [Online]. Available: https://csrc.nist.gov/publications/

[8] A. Hannousse, S. Yahiouche, and M. C. Nait-Hamoud, "Twenty-two years since revealing cross-site scripting attacks: a systematic mapping and a comprehensive survey." 2022. [Online]. Available: https://arxiv.org/abs/2205.08425

[9] B. M. Shuaibu, N. M. Norwawi, M. H. Selamat, and A. Al-Alwani, "Systematic review of web application security development model," *Artif. Intell. Rev.*, vol. 40, no. 4, pp. 507-536, 2013, doi: 10.1007/s10462-011-9270-2.

[10] R. Velasco, "What is IAST? All About Interactive Application Security Testing," *Hdiv Secur.*, 2020, [Online]. Available: https://hdivsecurity.com/what-is-iast

[11] M. Korolov, "Latest OWASP Top 10 looks at APIs, web apps," *CSO Online*, 2017, [Online]. Available: https://www.csoonline.com/article/3191047/latest-owasp-top-10-looks-at-apis-web-apps.html

[12] T. E. Gasiba, U. Lechner, M. Pinto-Albuquerque, and D. Mendez, "Is Secure Coding Education in the Industry Needed? An Investigation Through a Large Scale Survey," Feb. 10, 2021, *arXiv*: arXiv:2102.05343. doi: 10.48550/arXiv.2102.05343.

[13] OpenSSF, "Why are Organizations Struggling to Implement Secure Software Development? – Open Source Security Foundation." Accessed: Feb. 15, 2025. [Online]. Available: https://openssf.org/blog/2024/07/05/why-are-organizations-struggling-to-implement-secure-software-development/

[14] "Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Security-by-Design and -Default".

[15] "Security by design: Security principles and threat modeling." Accessed: Feb. 15, 2025. [Online]. Available: https://www.redhat.com/en/blog/security-design-security-principles-and-threat-modeling

[16] "OWASP Application Security Verification Standard," *OWASP*, 2021, [Online]. Available: https://owasp.org/www-project-application-security-verification-standard/

[17] "Web Application Vulnerability Scanners," *NIST*, 2021, [Online]. Available: https://samate.nist.gov/index.php/Web_Application_Vulnerability_Scanners.html

[18] A. Torbunova, A. Ashraf, and I. Porres, "A Systematic Mapping Study on Teaching of Security Concepts in Programming Courses." [Online]. Available: https://arxiv.org/abs/2407.07511

[19] T. Kim, H. Lee, and J. Park, "A survey on developers' security knowledge and adoption of secure coding practices," *Springer J. Softw. Eng. Res. Dev.*, vol. 11, no. 4, pp. 95-112, 2021, doi: 10.1007/s10207-020-00501-w.

[20] T. E. Gasiba, U. Lechner, M. Pinto-Albuquerque, and D. M. Fernandez, "Awareness of Secure Coding Guidelines in the Industry—A First Data Analysis." [Online]. Available: https://arxiv.org/abs/2101.02085

[21] M. F. Fauzi, V. R. Mohan, Y. Qi, C. Chandrasegar, and S. Muzafar, "Secure Software Development: Best Practices," *Int. J. Emerg. Multidiscip. Comput. Sci. Artif. Intell.*, vol. 2, no. 1, Art. no. 1, Nov. 2023, doi: 10.54938/ijemdcsai.2023.02.1.256.

[22] V. Diyora and N. Savani, “Blockchain or AI: Web Applications Security Mitigations,” in *2024 First International Conference on Pioneering Developments in Computer Science & Digital Technologies (IC2SDT)*, Aug. 2024, pp. 418–423. doi: 10.1109/IC2SDT62152.2024.10696861.

[23] R. P. McDonald, *Test Theory*. Psychology Press, 2013.

[24] L. J. Cronbach, “Coefficient alpha and the internal structure of tests,” *Psychometrika*, vol. 16, no. 3, pp. 297–334, 1951, doi: 10.1007/BF02310555.

[25] N. Kosaraju, S. R. Sankepally, and K. Mallikharjuna Rao, “Categorical Data: Need, Encoding, Selection of Encoding Method and Its Emergence in Machine Learning Models—A Practical Review Study on Heart Disease Prediction Dataset Using Pearson Correlation,” presented at the Proceedings of International Conference on Data Science and Applications. Lecture Notes in Networks and Systems, M. Saraswat, C. Chowdhury, C. Kumar Mandal, and A. H. Gandomi, Eds., Springer, 2023, pp. 369–382.

[26] H. F. Kaiser, “An Index of Factorial Simplicity,” *Psychometrika*, vol. 39, no. 1, pp. 31–36, 1974, doi: 10.1007/bf02291575.

[27] M. S. Bartlett, “Properties of sufficiency and statistical tests,” *Proc. R. Soc. Lond. Ser. - Math. Phys. Sci.*, vol. 160, no. 901, pp. 268–282, 1937, doi: 10.1098/rspa.1937.0109.

[28] J. R. Centre, *Handbook on constructing composite indicators: Methodology and user guide*. OECD publishing, 2008. Accessed: Feb. 14, 2025. [Online]. Available: https://books.google.com/books?hl=hu&lr=&id=N-jVAgAAQBAJ&oi=fnd&pg=PA13&dq=OECD+%26+JRC+(2008).+Handbook+on+Constructing+Composite+Indicators:+Methodology+and+User+Guide&ots=flza36kTag&sig=8drchm88kaF0oodWgmobdosgjVQ

[29] M. H. Kutner, Ed., *Applied linear statistical models*, 5th ed. in The McGraw-Hill/Irwin series operations and decision sciences. Boston: McGraw-Hill Irwin, 2005.