Accepted to the International Conference on Product-Focused Software Process Improvement (PROFES 2024).
Accepted preprint. Article first published in Lecture Notes in Computer Science (LNCS) by Springer Nature.

Guidelines for Supporting Software Engineers in Developing Secure Web Applications

Klara Svensson[1], Drake Axelrod[1], Mazen Mohamad[1,3], and Rebekka Wohlrab[1,2]

[1] Chalmers University of Technology and University of Gothenburg, Sweden
klara.svensson11@hotmail.com, drakeaxelrod@gmail.com, wohlrab@chalmers.se
[2] Software and Societal Systems Department, Carnegie Mellon University, Pittsburgh, PA, USA
[3] Research Institutes of Sweden (RISE), Gothenburg, Sweden
mazen.mohamad@ri.se

Abstract. As software applications get increasingly connected and complex, cybersecurity becomes more and more important to consider during development and evaluation. Software engineers need to be aware of various security threats and the countermeasures that can be taken to mitigate them. Currently, there is a lack of guidance for software engineers aiming to develop secure web applications. We conducted a design science research study, resulting in a set of guidelines to aid software engineers in developing secure web applications. The set of guidelines was constructed based on interview data with 10 industry practitioners. These guidelines were then evaluated using a survey with 28 respondents. Our results indicate that these proposed guidelines can be applied by software engineers to support the development and assessment of secure web applications in different stages of the software development lifecycle.

Keywords: cybersecurity, software engineering, guidelines, design science research, web applications, interviews, survey

1 Introduction

As the dependence on software applications in various sectors grows, and software systems become more complex, it is becoming increasingly important for software engineers to make sure that the software they create is secure [11]. Previous research [11] found that developers lack proper security knowledge. While software engineers are responsible for designing and developing software, cybersecurity professionals are typically responsible for mitigating security vulnerabilities. To proactively mitigate vulnerabilities, software engineers need to be familiar with various security threats and with the mitigation strategies that can be applied to reduce the risk of these threats [3]. As there is a low threshold for web application development, which is common among developers with a wide range of expertise, and the possible exploitation of sensitive data within web applications, it is even more important that cybersecurity is considered [14,18].

The intersection between cybersecurity and software engineering has gained increased attention in the last years [11,3,21,9,13]. They mostly focus on tools that can be used during development, such as recommender systems for secure coding [17] or mechanisms to create more secure software supply chains [7]. Previous studies often do not focus specifically on guidelines and resources for developers. Education and training for developers are needed to convey an understanding and awareness of cybersecurity concerns. Those concerns are not only relevant during the implementation phase, but also during other lifecycle phases, from planning to testing and maintenance. By examining the information needs of software engineers and the current role of cybersecurity in developing secure web applications, this research aims to fill this gap and identify guidelines for meeting the information needs to develop secure web applications.

In this paper, we present the results of a design science research study. Using two cycles, we developed a set of guidelines to aid software engineers in developing more secure web applications. These guidelines are targeted towards software engineers as they play an important role in ensuring secure systems through incorporating security when coding and implementing best practices during the development lifecycle [21]. We also identify the software life-cycle phase that these guidelines can be integrated into. Additionally, we collected empirical evidence from professionals within the fields of software engineering, cybersecurity, and management. Through cooperation with an external partner[4], the study gained input from industry practitioners and feedback on the usefulness of the created guidelines. Concretely, we focus on the following contributions:

1. An analysis of the information needs of developers aiming to implement secure web applications;
2. A set of guidelines to develop secure web applications;
3. An assessment of the perceived usefulness of the guidelines, based on a survey.

2 Related Work

Several resources for developers aiming to create secure web applications exist. For instance, the Mitre Att&ck (MITRE) framework [5] provides comprehensive adversary tactics and techniques data. Open Web Application Security Project (OWASP)[5] is a repository of resources for developers to make web applications more secure. OWASP is the gold standard for security-related information in the industry. However, OWASP mostly includes the top 10 vulnerabilities and no concrete steps on how these vulnerabilities can be mitigated using threat modeling and other techniques. Our work partially relies on these resources and aims to make them more accessible to developers. Recently, Siderova et al. [22] studied approaches to integrating security concerns into MDE for web applications. They concluded that approaches are needed that better address

[4] `https://www.qestit.se/`
[5] `https://www.owasp.org/`

the OWASP Top 10 vulnerabilities and that researchers should conduct more empirical studies on the topic. We focus on these issues in this paper.

A systematic literature review on developer-centric development [24] concluded that learning support in the reported literature takes the form of IDE plugins with limited practical validation. The authors also discussed the lack of links to external resources for developers' training, which we target in our study. The authors also discuss that the majority of reviewed papers suggest providing feedback to developers actively, meaning that it requires interrupting them while conducting their work, which might be an issue. In our study, we take a passive approach by providing guidelines for the developers to learn from, i.e., support on request. Lastly, the literature review discussed the difference between practitioners and students as research participants. While the majority of the studies feature students, we have a mix of both, which gives us diverse feedback.

It was found that development teams, when suitably guided, can significantly improve their security maturity even without the support of security specialists [26]. Another study [1] analyzed general advice resources and described the need for evaluating these resources and developing new ones.

Assal et al. [3] examined the human factors of software security and argued that a better understanding of human behavior and motivation can lead to more effective security policies and training methods. Similarly, the importance of integrating security policies and strategies into the development cycle was highlighted in a study on best practices for ensuring security in DevOps [4]. Our research builds upon these studies by conducting research from a software engineering perspective in web application development, as well as introducing valuable empirical evidence from industry practice.

Researchers have also looked into the challenges of implementing DevSecOps practices [2]. They emphasized the need for security standards for code, security tools (e.g., for testing), and security training and education for engineers. Our paper focuses on creating such guidelines that developers can follow to create more secure web applications.

Several other studies have proposed models for integrating security into the software development life cycle. One such model is the Secure Software Development Model [9], which is based on international standards and best practices. Kalharo et al. [13] proposed a cyber hygiene model that identifies key factors influencing software engineers' cyber hygiene behavior. The model is a valuable asset when developing guidelines for secure web application development.

Gasiba et al. [11] addressed the need for effective training methods using a game to raise awareness and provide practical advice. Finally, the study by Tahaei et al. [24] examined developer-centered security and provided tools and processes to better support both developers and secure code production. However, this study did not focus specifically on web applications and suggested further study into developer-focused security. Our research aims to further explore developer-centered security as related to web applications, by developing and evaluating guidelines for different lifecycle phases.

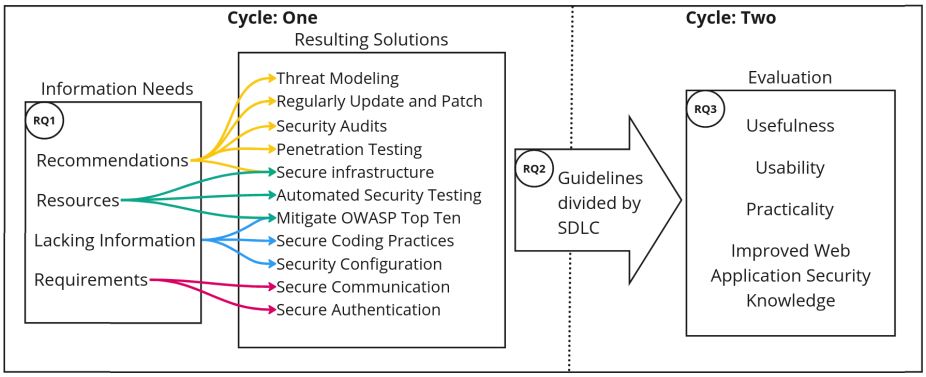

Fig. 1. Design science research process and results overview

3 Research Method

The purpose of this study is to gain a better understanding of the current web application security knowledge within the software engineering domain and design a set of guidelines for developing secure web applications. All interview questions, survey questions, artifact information, and results are available in the supplementary material on Figshare[6]. Note that we could not make the interview transcripts available because we assured the interviewees that their data would be treated confidentially.

Design science involves iteratively designing, developing, and evaluating the artifact [12]. In our case, the artifact is a set of guidelines to aid software engineers in developing more secure web applications. Design science is an appropriate method for researchers who want to get an in-depth understanding of the problem and create a solution that is evaluated and refined in several cycles.

To understand the current state of the practice and investigate the potential use of guidelines, our research focuses on the following research questions:

RQ1 What are the security information needs of software engineers to develop secure web applications?
RQ2 What potential guidelines can support software engineers in developing secure web applications?
RQ3 How can those guidelines be applied to the development of secure web applications?

The research phases and tasks conducted were separated into two cycles. Figure 1 shows an overview of the overall research process.

Cycle 1: This cycle focused on learning more about what kinds of resources and information software developers require to develop secure web applications and

[6] `https://doi.org/10.6084/m9.figshare.27095113.v1`

how to provide these resources as a set of guidelines. This was done by reviewing the literature and conducting semi-structured interviews.

Interview Design The interviews were designed to gather information on the topics of software engineering practices for designing secure web applications, the most common problems, and effective preventative practices. The interview guide is available on Figshare[6]. The interviews were conducted through a video conferencing system. All interviews were recorded and transcribed.

Selection of Participants We aimed to understand how software engineers can be supported in developing more secure web applications. To this end, we selected a mix of participants. All of them worked at the same company[4]. Some were cybersecurity experts, whereas others came from the web application domain. We decided to include both experienced and junior participants, given that the target audience of our guidelines is rather broad. The interviews were conducted with individuals of different roles with at least six months of experience in their current positions. Some of the interviewees had overlap among the domains which resulted in six interviewees from the software engineering domain, three from the management domain, and five from the cybersecurity domain. All interviewees are involved in activities connected to the development of a web application. Table 1 shows an overview of the interviewees in cycle one.

Table 1. Overview of the interviewees in Cycle 1.

#	**Exp.**	**Interviewee Job Title**
1	> 10 yrs.	Consultant Manager, QA & Agile Coaching
2	> 10 yrs.	Security Engineer, Penetration Technician
3	1-2 yrs.	DevOps engineer
4	1-2 yrs.	Developer
5	2-5 yrs.	Penetration Tester
6	5-10 yrs.	Penetration Tester, Manager
7	5-10 yrs.	Architect
8	2-5 yrs.	Front-End Developer
9	> 10 yrs.	Penetration Tester
10	1-2 yrs.	Developer

The guidelines were created in Cycle 1 using the information gathered from the interviews and the literature. A visual model in a tabular format was created.

To evaluate the solution, we performed an internal evaluation workshop. In this workshop, the first cycle of guidelines was discussed, with a focus on making the guidelines more accessible, practical, and usable.

Cycle 2: Based on the evaluation of Cycle 1, we performed an additional literature search to support the design and content decisions of the guidelines. As a

result of cycle one, some problem areas of the artifact were found, such as too much information and a low level of intuitive design.

Some guidelines were removed due to insufficient backing data or level of importance. A website representation of the guidelines was created with all guidelines. Design improvements were made, such as aligning categories according to the Software Development Life Cycle (SDLC) process.

The evaluation involved a survey to gather data on the practicality, usability, and relevance of the artifact. The survey was distributed to participants within the domains of software engineering, cybersecurity, and management.

Survey Design The survey was created using Google Forms. The survey questions were designed to complement the interview questions, answer RQ3, and confirm or challenge previously gathered data for RQ1. The survey was distributed through online platforms, including LinkedIn groups within the relevant domains. The surveys were further distributed by encouraging participants to invite others within their networks to answer the survey. Moreover, our industry partner helped us to recruit practitioners from the software engineering domain, cybersecurity professionals, and management. Our respondents represented different industries, organizations, and levels of experience. We decided to include 6 students with varying levels of experience, as long as they had worked on web application development projects before. After all, our intention was to create guidelines that should be useful to a variety of participants—not only expert developers or people involved with cybersecurity. We received 28 responses from the domains of software engineering, cybersecurity, and management. The survey consisted of three questions to establish the demographic of the respondents, followed by Likert scale questions to collect feedback.

The interviews and surveys were conducted following ethical guidelines for research involving human participants [23]. All data collected was kept confidential and anonymized, and identifying information was removed from the data.

3.1 Data Analysis

The interviews were transcribed and analyzed using thematic analysis to identify key themes in the data. The surveys were analyzed using descriptive statistics to provide a quantitative perspective on the data. The interview data was analyzed using inductive coding to identify emerging themes and patterns [27]. Two researchers independently coded the data and compared the results. Once the themes were identified, they were aggregated and discussed to arrive at conclusions about the research questions. As an example for the data analysis, we consider the following quote: “I would say that kind of depending on what kind of a web application it is, it is in the developers’ interest to keep it secure, but in some cases, the benefit does not outweigh the cost.” The quote talks about the cost-benefit tradeoff connected to security. It was coded with “cost” and “decisions” in the “management” theme.

We used descriptive statistics to analyze the survey responses. Concretely, the data was analyzed using measures such as frequency distributions, central

tendencies, and variability to describe the characteristics of the sample and the distribution of responses to the survey questions.

The interview and survey data were integrated to provide an understanding of the guidelines necessary for software engineers to develop secure web applications. The qualitative and quantitative data was triangulated with related literature, interviews, and survey data. The interviews mainly provided information regarding domain knowledge and present issues, whereas the surveys primarily confirmed assumptions based on the interviews and evaluated the artifact.

3.2 Threats to Validity

We discuss threats to validity using Easterbrook et al.'s criteria [6].

Internal Validity: The data collected is based on self-reported information obtained through interviews and surveys, which may introduce bias. To reduce the likelihood of bias, the interview questions were designed to be clear and non-leading. Two interviewers were present to collect data, which reduces the impact of interviewer bias. The interview questions were reviewed by the four participating researchers. Anonymity and non-evaluation of answers were ensured to encourage truthful responses. We collected rich data using interviews as a qualitative data source and a survey as a quantitative data source. This approach allowed us to triangulate the findings, further improving internal validity. Additionally, although the interviewees might not have been aware of best practices in cybersecurity, they possessed sufficient knowledge to understand the questions and provide informed responses. This ensured the data collected was still relevant and valuable for the study.

External Validity: The findings of this study are not fully generalizable to other software development contexts, as the study was conducted with a specific population and geographic location. Efforts have been made to include a variety of roles, but the sample might not fully represent the populations being studied [15]. We decided to include less experienced participants, as long as they were familiar with web application development. We wanted to design guidelines for a broad audience—not only expert developers or developers who were familiar with cybersecurity issues. Some of the involved survey respondents were software engineering students, who can be seen as junior software engineers. It would be beneficial to run further evaluations with more experienced practitioners.

Construct Validity: The interview and survey questions may not have fully captured the constructs being studied. To address this, the terminology used in the questions was explained to the participants, considering that terms such as "OWASP" may not be widely known in the specific context.

The survey questions may not accurately measure the construct of interest (usability and helpfulness of the artifact). There is also a risk that important aspects have not been covered by the survey questions.

Reliability: We provide supplementary material to enable others to replicate this study. The results of the survey may have been influenced by factors such as the timing of the survey, the way we asked the questions, or the participants' level of engagement. Additionally, the tool being evaluated in the survey is one made by the authors for this research. The questions we asked were framed in a mainly positive manner, potentially producing biased answers. These factors should be considered when interpreting the results.

4 Findings

4.1 RQ1 - Security information needs

The findings from the conducted interviews revealed problems related to cybersecurity education. Firstly, there is a significant lack of education on cybersecurity, with many developers reporting that it is not covered satisfactorily in their education, job descriptions, or workflows. One of the penetration testers stated that "you're not taught how to build it securely or how to even in any way at all mitigate these kinds of vulnerabilities that are actually quite common." Another developer mentioned that "I don't think I have the knowledge right now to identify these potential issues or vulnerabilities that might happen from the code I write" and even so, mentions that security does come to mind when developing. This lack of security is further exemplified by one of the Cybersecurity Managers, saying "I know that despite that there's awareness that there's vulnerabilities, the work done to mitigate these is not sufficient."

The lack of appropriate resources that explain cybersecurity concepts in a way that developers can understand or apply is also a common challenge. Many developers are unaware of resources such as OWASP that are well-known among cybersecurity professionals. One security engineer interviewee mentions that "I would say the OWASP model but then again it requires them to actually have the time and the interest to understand the model" when asked what they think is the most important to think about when developing a secure web application, further confirming the importance OWASP but exemplifying the potential issue of information being too complex. To confirm this, one of the developers was asked about OWASP, they stated that "I'd say it's quite useful. But I just haven't found any sort of practical application for it just yet. Because I'm not too experienced with the resource itself."

We found that the reuse of insecure code found online is a prevalent issue among developers. A penetration tester mentioned that "if you find an example code somewhere like let's say Stack Overflow without even validating that the code is secure enough, you're basically implementing a code that might be vulnerable without even knowing about it." This practice increases the risk of common vulnerabilities, as developers may not know how to properly implement certain features or what to look for in terms of security vulnerabilities, including those outlined in the OWASP Top 10.

Inadequate access to cybersecurity professionals and a lack of security culture within development teams are also identified challenges, "[...] there's a huge

gap in both ways because the company I'm working for has developers and we have a security team. And I realize that there's actually a big gap between us, even if we're working for the same company" and "I'd say the overall level of security awareness when it comes to companies and developers is quite low." Developers may not know where to look for information or have limited access to cybersecurity experts who can provide guidance.

Securing user data and backend systems are also recognized as critical concerns. An interviewee stated that "guidelines should be at the bare minimum. At least make sure that the code you write doesn't put your [...] users' data at risk." Developers acknowledge the importance of these areas but do not always know how to effectively address them.

Summary

Based on the findings from the interviews and literature review, the following are the information needs of software engineers to develop secure web applications:

1. Education & training on cybersecurity, to help understand mitigation strategies and key concepts.
2. Information on resources that explain a way that developers can understand and apply.
3. Guidance on securing user data and backend systems.
4. Awareness of vulnerabilities and threats, such as the OWASP Top 10.
5. Importance of secure coding practices, including avoiding the reuse of insecure code found online.
6. Security culture within development teams and adequate access to cybersecurity professionals who can provide guidance and support.

4.2 RQ2 - Guidelines

The findings suggested that potential guidelines can provide software engineers with the necessary information to develop secure web applications. Utilizing such guidelines and promoting a security culture within development teams can help raise awareness and emphasize the importance of cybersecurity among software engineers. We found that presenting cybersecurity concepts, practices, and resources in a more accessible way for developers indicates the possibility of improving their understanding and chances of using secure coding practices.

The guidelines created from problem identification and interviews were condensed and coupled with backing literature and sources. Our data indicated that accessibility and ease of understanding were the main requirements for our artifact. Therefore, a website representation of the guidelines was created[7]. It can simplify information and improve retention, as many of the existing sources include technical concepts that may be difficult to understand solely through written text. By using visual elements, the guidelines can be simplified and presented in a more visually engaging way, making it easier for readers to grasp the key concepts and remember them, along with finding them easily. Table 2 shows an overview of the guidelines, which we explain in the following.

[7] `https://dev-swag.vercel.app/`

Table 2. Tabular representation of the guidelines.

Phase	Technique
Planning	Threat Modeling
Design	Secure Communication Protocols, Secure Authentication
Implementation	Secure Coding, Mitigate OWASP Top 10, Penetration Testing
Testing and Integration	Automated Security Testing
Deployment	Secure Configurations
Maintenance	Regular Update and Patch the Application, Conduct Regular Security Audits and Vulnerability Assessments

Threat Modeling: Many interviewees in the cybersecurity domain recommended threat modeling as a useful tool. When cybersecurity management was asked what advice they would give to developers, they answered that "it's at tops [a] two to three days session where we sit down and go through what their solution is going to look like". Another penetration tester stated that "I love threat modeling and that's something I also recommend going back to what I said before about being part of the projects from the beginning". Similarly, Myagmar et al. [16] also recommends threat modeling by discussing how it can be used as a foundation for specifying security requirements.

Secure Communication: If exchanges are not secured through secure communication channels, web applications might have sensitive data leaked or compromised. Our findings suggest that a large issue that comes with web applications is the exchange of sensitive information or data. According to the developers interviewed, communication channels are one of the most known potential security risks. A penetration technician also mentioned that using encrypted communications protocols is one of the most important features for potential tools aiding developers in securing their web applications. However developers may not always know how to implement or maintain it, therefore it should be one of the recommendations in the guidelines.

Secure Authentication: When discussing the security levels of projects that the interviewees have worked on, an architect responded with the security being low to medium-low and authentication being "probably the biggest problem". Another interviewee stated that a big part is to "be aware of the users that will be using your application, what their sort of normal level of technical experiences, isolate your components and secure them through authentication or authorization methods." Authentication is clearly a concern that developers should be more aware of when developing, and therefore it is present in the guidelines.

OWASP Top 10: OWASP was brought up in several interviews, primarily by those working within cybersecurity as penetration testers or management. While OWASP is well known within the domain of cybersecurity, it is rarely known

within software engineering. 50% of our respondents strongly disagree with being familiar with the OWASP Top 10. Therefore, we see that there is a need for increased awareness regarding resources such as OWASP. As one penetration tester mentioned, "the recommendation is to follow OWASP, use OWASP as your main go-to reference when it comes to designing your application."

Secure Coding Practices: Establishing and using secure coding techniques is essential for creating secure web applications. It is also crucial when a developer assesses whether someone else's code is secure. This encapsulates most of the issues present in the security of web applications, and developers may lower the risk of vulnerabilities and attacks by adhering to secure coding standards and recommendations, such as those provided by OWASP. The practice of secure coding guidelines has previously been found to be impactful when it comes to the quality and security of products [8]. Similarly, when a manager was asked what it means for cybersecurity in a project to be up to date, one of the things they look at is if "Do they actively review code from a security standpoint and both correct code that's insecure and also rewrite guidelines for secure coding?".

Penetration Testing: In order to create secure web applications, penetration testing can be an important measure as it reveals weaknesses that could go unnoticed throughout the development process. Penetration testing is often performed as part of the implementation phase and thought of before other testing activities start. One penetration tester mentioned that "even before starting to developing the application [...] you can involve security consultants or penetration testers", though this may not be possible in all projects. The recommendations of penetration testing could vary depending on company size and resources, though automated web penetration tests and tools should easily be implemented to help reveal risks and vulnerabilities.

Automated Security Testing: Automated security testing can assist in quickly and effectively identifying security vulnerabilities and is a crucial component of creating secure web applications. Integrating automated testing also allows for saving both time and efficiency when developing and deploying web applications. As one manager suggested during their interview to perform "at least some basic security testing". Automated security testing is already an established way of ensuring continuous security [20], and is thus present in the guidelines.

Security Configuration: By configuring web servers, databases, and other components according to secure configurations, developers can reduce the attack surface of web applications and make them more resilient to threats. One of the developers mentioned that the company they work for is making sure the security of the projects is up to date by "All of our internal-external projects share the same security configurations" and continues by mentioning the implementation of regular security checks. This is something the respondent considered to work well and showcased that the company was communicating those routines to developers. In the OWASP Top 10 in 2021, security misconfiguration is at the

fifth spot of most significant security threats for web applications, and should thus be taken into more consideration during development.

Regularly Update and Patch: By regularly applying security updates and patches on the software and components of their web applications, developers can mitigate some of the most common threats and attacks. One developer described a scenario in which they had to update dependencies and packages. They stated that "for example, the react app is not really good at updating the dependencies. So the whole package is really outdated, to be able to outcome that we need to add some resolutions individually". To mitigate these issues, they remove unused dependencies and continuously inventory the versions and dependencies within the applications. Another developer interviewed mentioned that the current practices of regularly updating include that "we use Ansible to configure everything so everything runs on the same version of a library runs on the same OS image. They all get the same updates and that helps us keep things consistent". Because it is such a common problem, it is included in the guidelines.

Security Audits: Security audits seem to be common within the industry, one of the developers interviewed mentioned that they are using Yarn audit[8] to detect if there are any major security threats. One penetration tester also mentioned security audits when asked what level of cybersecurity they would rank the projects they have worked on, they then mentioned that customers they work with often have to adhere to using security audits "Due to the fact that they actually have some regulations controlling them, they won't be allowed to do stuff without security audits." By regularly conducting security audits such as vulnerability assessments and code reviews, developers can find and fix potential weaknesses. Security audits are a common countermeasure to cybersecurity attacks, and should therefore also be part of the guidelines.

Summary
We created guidelines for developing secure web applications, as listed in Table 2.

4.3 RQ3 - Applying Guidelines in Practice

Most interviewees stated that integrating cybersecurity as early as possible can potentially make the process less of an issue later on. A cybersecurity manager mentioned that cybersecurity "does play a role and needs to be done early and also continuously throughout the development process." We found that to make cybersecurity more understandable for software engineers, it is useful to consider the SDLC phases to make it clear to software engineers when and where they need to be aware of certain cybersecurity measures.

All developers interviewed mentioned utilizing some sort of development process, largely some sort of agile adaptation "The development process is usually guided by the company that I work for. And at present, it's an Agile Scrum

[8] `https://classic.yarnpkg.com/en/docs/cli/audit/`

approach that we are using." Given that the SDLC phases in the guidelines can be mapped to pretty much any development process, they are considered to be generally applicable by our participants.

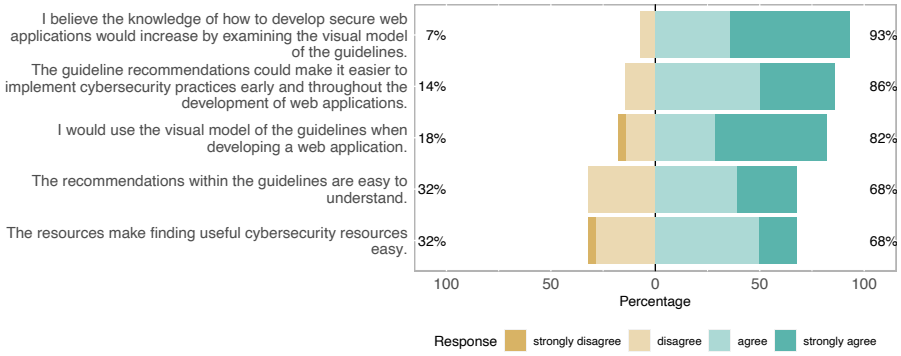

Fig. 2. Responses to our Likert-scale questions (n=28)

Figure 2 shows our survey responses. The demographics of the survey participants can be found in the supplementary material. Our participants had varying levels of experience. 57.1% of the participants worked in Software Engineering (Developer, UX, Architecture, etc.). The remaining participants worked in Cybersecurity, Management, Architecture, or System Administration. The survey results were overall positive with most respondents being unanimous.

When discussing the survey results, we refer to mean and standard deviation as a measure of central tendency and dispersion, respectively. The values ranged from 0 to 5. A mean of 2.5 or less is considered a negative response, and a mean of 2.5 or greater is considered a positive response. The statement "I am familiar with cybersecurity" resulted in a mean of 2.54 with a standard deviation of 1.04, indicating a moderate level of familiarity with cybersecurity. Answers to the statement "I find it difficult to identify security issues when developing web applications" resulted in a mean of 2.32 and a standard deviation of 0.82 showing that respondents did find it difficult to identify security issues. However, they had a moderate level of understanding of where to find information about securing web applications (with a mean of 2.61 and a standard deviation of 0.92). Respondents found it difficult to understand the information given in various sources regarding developing secure web applications (with a mean of 2.21 and a standard deviation of 1.03). This indicates that there needs to be a more understandable source of information for software developers.

Answers to the statement "I wish there was a site that collected the most common cybersecurity practices and sources for secure web application development." resulted in a mean of 3.57 and a standard deviation of 0.84. This indicates

that presenting the guidelines in a website format was reasonable. They also believed that web development would benefit from guidelines to improve security.

Overall, the survey results indicate that participants found the guidelines to be useful in developing secure web applications, especially when the guidelines are presented as a website and when broken down by the SDLC. The survey indicated that participants found written guidelines less compelling than an interactive and visually compelling website.

Summary

We found that our guidelines are perceived as applicable, especially if they are integrated into the stages of the SDLC. By breaking down the recommendations into the SDLC phases, developers can obtain specific recommendations that are relevant to the stage they are currently at.

5 Summary and Discussion

We performed 10 semi-structured interviews with practitioners and a survey with 28 responses to design a set of guidelines that aid software engineers in developing secure web applications. Our guidelines can be used by developers to improve their cybersecurity practices, by security analysts to communicate with developers and agree on common ways of working, and by managers or process experts to assess whether an organization has the appropriate mechanisms in place to ensure that secure web applications can be developed.

Our findings provide important empirical evidence on the gap between developers and cybersecurity experts. They also indicate what techniques are prioritized today and where better support is needed. In particular, our results shed light on developers' information needs (RQ1). The findings partially confirm developers' lack of security knowledge [19] and align with the findings of Gasiba et al. [10] that despite the lack of secure coding policies awareness among the developers, the majority acknowledges its importance. Kalhoro et al. [13] found that several social, personal, and technological factors influence cyber hygiene practices. While their findings confirm some of ours, they do not stress the issue of code reuse and its potential impact on the security of a system. This might be because, in web application contexts, it is more common than in other domains to reuse code from online sources. Moreover, since the introduction of Large Language Models (LLMs), developers have leveraged these tools for various tasks, including web development. Tóth et al. [25] found that websites created by GPTs are vulnerable which adds to the problem of reusing online code. We believe that the need for secure coding guidelines is increasing along with the increased use of AI for coding. Our participants also acknowledge the importance of secure coding guidelines and of cybersecurity in general and consider it to be a high-priority concern. Hence, our results do not confirm the findings of Kalhoro et al. [13] that cybersecurity was not prioritized because of budget constraints.

In this study, we designed a set of guidelines that points to what was considered most crucial by the participants of the study and by related work. These

guidelines consider all stages of the development life-cycle including early phases, which aligns with previous research indicating the need to include cybersecurity concerns early in the process [19,9]. Our findings indicate that it is important to think of tasks such as penetration testing already during implementation, so that cybersecurity concerns are not considered as an afterthought.

Our findings indicate a need for further research to identify and assess alternative approaches for improving web application security. We found that developers struggle with finding starting points to improve their cybersecurity practices and developing lightweight techniques for threat modeling is a promising area of future work to address that challenge. It appears promising to integrate secure coding guidelines into IDEs or other tools, so that regular updates, audits, testing, and analysis activities can be performed. Moreover, revising and adapting the guidelines to include practices for using LLMs in web development is an important future work. It would also be interesting to conduct a longitudinal study of the solution presented, where developers would use the guidelines for an extended period. This would allow concrete observations of the effects of working with the guidelines on their security knowledge levels.

Acknowledgments: This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

1. Acar, Y., Stransky, C., Wermke, D., Weir, C., Mazurek, M.L., Fahl, S.: Developers need support, too: A survey of security advice for software developers. In: SecDev. pp. 22–26. IEEE (2017)
2. Akbar, M.A., Smolander, K., Mahmood, S., Alsanad, A.: Toward successful DevSecOps in software development organizations: A decision-making framework. Information and Software Technology **147** (2022)
3. Assal, H., Chiasson, S.: Think secure from the beginning: A Survey with Software Developers. In: CHI 2019. pp. 1–13 (2019)
4. Battina, D.S.: Best practices for ensuring security in devops: A case study approach. International Journal of Innovations in Engineering Research and Technology **4**(11), 38–45 (2017)
5. Blake E. Strom, Andy Applebaum, et al.: MITRE ATT&CK: Design and Philosophy. Tech. rep., The MITRE Corporation (Jul 2018)
6. Easterbrook, S., Singer, J., Storey, M.A., Damian, D.: Selecting Empirical Methods for Software Engineering Research, pp. 285–311. Springer London (2008), `https://doi.org/10.1007/978-1-84800-044-5_11`
7. Ellison, R.J., Goodenough, J.B., Weinstock, C.B., Woody, C.: Evaluating and mitigating software supply chain security risks. Tech. Rep. CMU/SEI-2010-TN-016 (2010)
8. Espinha Gasiba, T., Beckers, K., Suppan, S., Rezabek, F.: On the Requirements for Serious Games Geared Towards Software Developers in the Industry. In: RE 2019. pp. 286–296 (2019)

9. Futcher, L.: SecSDM: A Model for Integrating Security into the Software Development Life Cycle. In: 5th World Conference on Information Security Education;. pp. 41–48 (Jan 2007)
10. Gasiba, T., Lechner, U., Albuquerque, M., Fernandez, D.: Awareness of Secure Coding Guidelines in the Industry - A First Data Analysis. In: TrustCom. pp. 345–352 (2020)
11. Gasiba, T., Lechner, U., Pinto-Albuquerque, M.: CyberSecurity Challenges for Software Developer Awareness Training in Industrial Environments. In: Innovation Through Information Systems. pp. 370–387 (2021)
12. Hevner, A., R, A., March, S., T, S., Park, Park, J., Ram, Sudha: Design Science in Information Systems Research. MIS Quarterly **28**, 75 (Mar 2004)
13. Kalhoro, S., Rehman, M., Ponnusamy, V., Shaikh, F.B.: Extracting Key Factors of Cyber Hygiene Behaviour Among Software Engineers: A Systematic Literature Review. IEEE Access **9** (2021)
14. Kumar, S., Mahajan, R., Kumar, N., Khatri, S.K.: A study on web application security and detecting security vulnerabilities. In: ICRITO. pp. 451–455 (2017)
15. Larsen, K.R., Lukyanenko, R., Mueller, R.M., Storey, V.C., VanderMeer, D., Parsons, J., Hovorka, D.S.: Validity in Design Science Research. In: Designing for Digital Transformation. Co-Creating Services with Citizens and Industry (2020)
16. Myagmar, S., Lee, A.J., Yurcik, W.: Threat Modeling as a Basis for Security Requirements. University of Pittsburgh (Aug 2005)
17. Nembhard, F.D., Carvalho, M.M., Eskridge, T.C.: Towards the application of recommender systems to secure coding. EURASIP Journal on Information Security **2019**(1), 9 (2019)
18. Perwej, D.Y., Abbas, S.Q., Dixit, J.P., Akhtar, D.N., Jaiswal, A.K.: A Systematic Literature Review on the Cyber Security. Int'l Journal of Scientific Research and Management **9**(12), 669 (Dec 2021)
19. Petranović, T., Žarić, N.: Effectiveness of using OWASP TOP 10 as AppSec standard. In: IT. pp. 1–4 (Feb 2023)
20. Rangnau, T., v. Buijtenen, R., Fransen, F., Turkmen, F.: Continuous Security Testing: A Case Study on Integrating Dynamic Security Testing Tools in CI/CD Pipelines. In: EDOC. pp. 145–154 (Oct 2020)
21. Rastogi, A., Nygard, K.: Cybersecurity Practices from a Software Engineering Perspective. In: SERP (2017)
22. Siderova, A., Daneva, M., Bukhsh, F.A., Arachchige, J.J.: Security approaches in model-driven engineering for web applications: the state-of-the-art in the last 10 years. In: RE Workshops. pp. 155–163 (2024)
23. Strandberg, P.E.: Ethical Interviews in Software Engineering. In: ESEM 2019. pp. 1–11 (Sep 2019)
24. Tahaei, M., Vaniea, K.: A survey on developer-centred security. In: EuroS&PW. pp. 129–138. IEEE (2019)
25. Tóth, R., Bisztray, T., Erdodi, L.: LLMs in web-development: Evaluating LLM-generated PHP code unveiling vulnerabilities and limitations. In: SAFECOMP Workshops (2024)
26. Weir, C., Becker, I., Blair, L.: A passion for security: Intervening to help software developers. In: ICSE-SEIP. pp. 21–30. IEEE (2021)
27. Wicks, D.: The Coding Manual for Qualitative Researchers (3rd edition). Qualitative Research in Organizations and Management (2), 169–170 (2017)