JOURNAL OF DIGITAL SYSTEM DEVELOPMENT

https://e-journal.uum.edu.my/index.php/jdsd

How to cite this article:
Fauzi, M. H. A. & Hashim, N. L. (2025). Evaluating the Effectiveness of a Comprehensive Lightweight Application Security Process Framework in Capturing Security Requirements among Novice Developers. *Journal of Digital System Development, 3* (1), 101-116. https://doi.org/10.32890/jdsd2025.3.1.9

EVALUATING THE EFFECTIVENESS OF A COMPREHENSIVE LIGHTWEIGHT APPLICATION SECURITY PROCESS FRAMEWORK IN CAPTURING SECURITY REQUIREMENTS AMONG NOVICE DEVELOPERS

[1]Mohamad Hafizal Ahmad Fauzi & [2]Nor Laily Hashim
School of Computing, Universiti Utara Malaysia, Malaysia
[1]Corresponding author: m_hafizal_ahmad@soc.uum.edu.my

Received: 19/2/2025 Revised: 21/4/2025 Accepted: 21/4/2025 Published: 28/4/2025

ABSTRACT

Existing evaluations of the security requirement framework often have a limited focus on capturing security requirements, leaving gaps in understanding their effectiveness and usability. This study investigates the effectiveness and usability of the Comprehensive, Lightweight Application Security Process (CLASP) framework in capturing and documenting security requirements, particularly for novice developers. This research examines how effective and usable the CLASP framework is in assisting novice developers in identifying security requirements. This study employed an experimental methodology, dividing participants into groups, providing structured educational materials, and guiding them through the CLASP framework using a controlled case study. Participants prepared security requirements by completing the CLASP templates, and CLASP framework effectiveness was evaluated using task completion rates and "task with error" analysis. CLASP's usability was evaluated based on the System Usability Scale (SUS). Using an online bakery system as a case study, 55 undergraduate students assessed CLASP's effectiveness and usability regarding documentation quality and overall usability in enhancing security requirements identification. Results indicate high usability scores, particularly for novice developers, and validate the efficiency of the CLASP frameworks. However, limitations such as a small sample size, reliance on self-reported feedback, and the focus on a single case study are acknowledged. The findings from this study contribute to the existing body of knowledge by providing empirical evidence of CLASP's impact on improving security documentation practices.

Keywords: Security Framework, Effectiveness Evaluation, Usability evaluation, Security Frameworks Evaluation.

INTRODUCTION

Capturing and documenting security requirements effectively is a critical aspect of software development, especially as modern systems' complexity and security demands continue to grow. Security requirements are essential to ensure that software systems are resilient against emerging threats and capable of protecting sensitive data. However, novice developers, often tasked with these responsibilities, face significant challenges due to their limited experience and technical expertise. These challenges often result in incomplete or inaccurate security documentation, leaving systems vulnerable to exploitation. The absence of clear and accessible frameworks further exacerbates this issue, making it imperative to develop tools or methods that simplify the security requirements engineering process for developers at all skill levels.

In today's software development landscape, gathering security requirements has become increasingly critical for developing robust and secure applications. The Comprehensive Lightweight Application Security Process (CLASP) framework represents a systematic approach to integrating security considerations throughout the software development lifecycle, emphasising structured methodologies over ad hoc practices (Viega, 2005). Security must be addressed early in the requirements phase to mitigate risks and avoid costly rework later in development (Tunio & Ahmad, 2021). Despite the growing number of security requirements engineering (SRE) methods, many still rely heavily on expert knowledge, making them inaccessible to novice developers or students without proper guidance (Tunio & Ahmad, 2021; Janisar et al., 2023). A recent study by Janisar et al. (2024) examined various existing security requirements engineering approaches, aiming to help researchers and industry professionals choose the most suitable one from an assurance standpoint. However, the study did not explore which approach would be appropriate for novice developers, nor did it assess the effectiveness of the selected frameworks.

This research addresses the fundamental challenge of effectively capturing security requirements in software projects by leveraging structured approaches like CLASP and templates designed to aid non-experts. The relevance of this study stems from the increasing complexity of cyber threats and the urgent need for standardised approaches to protect sensitive data in modern applications (Tunio & Ahmad, 2021; Viega, 2005). By focusing on methodologies incorporating iterative refinement and measurable metrics, such as CLASP, this research aims to provide practical tools for students and practitioners to ensure comprehensive security requirement identification and documentation (Janisar et al., 2023).

The CLASP framework addresses this gap by offering a structured methodology designed to assist developers in capturing security requirements effectively. CLASP provides user-friendly templates, step-by-step guidelines, and systematic processes that help developers document security vulnerabilities and implement mitigations early in the software development lifecycle. By emphasising simplicity and practicality, CLASP ensures that even novice developers can engage with the framework confidently, leading to improved security outcomes.

This study evaluates the effectiveness and usability of the CLASP framework in assisting novice developers with security requirements capture. Using an online bakery system as a controlled case study, this research explores several critical dimensions of the framework, including its usability, documentation quality, and overall effectiveness in guiding security practices. The case study was selected for its relevance in simulating real-world scenarios, allowing participants to apply the framework in a practical and meaningful context.

The following section presents the literature review, providing an overview of relevant studies and frameworks related to security requirements engineering. This section is followed by the methodology,

which details the experimental design, participant selection, and data collection process. The analysis and results section then presents the findings, including statistical evaluations of CLASP's effectiveness. The discussion interprets these findings in the context of existing research, highlighting key insights and limitations. Finally, the conclusion summarises the study's contributions, acknowledges its limitations, and suggests directions for future research.

LITERATURE REVIEW

The field of security requirements engineering has long recognised the challenges developers face, particularly those with significant experience. Novice developers are often disadvantaged when identifying, documenting, and addressing security vulnerabilities, making the need for accessible frameworks more pressing. Frameworks like the Comprehensive Lightweight Application Security Process (CLASP) and the Security Requirements Specification (SecRS) have been introduced to address these challenges by offering step-by-step methodologies that guide developers through the intricacies of security requirements capture. Research by Brown et al. (2020) underscores the effectiveness of such frameworks in simplifying complex processes and ensuring that critical security measures are not overlooked.

One of the distinctive features of CLASP, as highlighted by Johnson and Lee (2021), is its lightweight and user-friendly approach, which makes it particularly suitable for novice developers. Unlike more complex frameworks that may overwhelm inexperienced users, CLASP provides structured templates and guidelines that simplify documenting security requirements. This accessibility ensures that developers can focus on the core aspects of security without becoming bogged down by overly technical details. Research by Thomas et al. (2018) further supports the utility of structured frameworks, showing that their use significantly reduces errors in documentation and enhances compliance with established security standards.

The importance of usability in security frameworks cannot be overstated. Usability directly impacts the adoption and effectiveness of these tools, as developers are more likely to use intuitive and easy-to-navigate frameworks. Smith et al. (2019) emphasise that usability is critical in determining whether a security framework will be successfully implemented. Despite the potential of existing frameworks, many fail to address novice developers' unique needs. This gap leaves room for improvement and highlights the significance of frameworks like CLASP that prioritise simplicity and accessibility while maintaining robust functionality.

CLASP's emphasis on usability and its tailored approach to security requirements engineering make it a valuable tool for less experienced developers. CLASP bridges the gap between theoretical knowledge and practical application by providing clear instructions, practical templates, and an iterative process. This study builds on existing literature by empirically evaluating CLASP's effectiveness in real-world scenarios. The findings contribute to the ongoing conversation about the importance of accessible and efficient tools in security requirements engineering and demonstrate the potential of CLASP to serve as a benchmark for future frameworks aimed at novice users.

While frameworks like CLASP and SecRS have made significant strides in simplifying security requirements engineering, gaps remain in their application and usability for diverse developer groups. This study assesses CLASP's effectiveness and provides insights into how such frameworks can be further refined to meet the evolving needs of software developers in an increasingly complex and interconnected

digital environment. By focusing on usability, effectiveness, and practical application, this research lays the groundwork for future advancements in the field of security requirements engineering.

CLASP was chosen over SecRS because it offers a lightweight, user-friendly, and structured approach that is particularly well-suited for novice developers, providing clear templates and step-by-step guidelines that simplify capturing and documenting security requirements (Viega, 2005). Research highlights that CLASP's emphasis on usability and accessibility enables less experienced users to focus on essential security aspects without being overwhelmed by technical complexity, a common challenge with more intricate frameworks like SecRS (Johnson & Lee, 2021). Additionally, empirical studies have shown that CLASP significantly improves documentation quality and reduces errors among novice developers, validating its effectiveness and practical utility in educational contexts (Smith et al., 2019).

METHODOLOGY

The study aimed to evaluate the effectiveness and usability of the CLASP framework in capturing security requirements. The effectiveness of the CLASP framework was measured by the score on the documentation requirements and the score on the CLASP template. The usability score for novice developers measured the usability of the CLASP framework. The experimental methodology centred on assessing how well novice developers could use CLASP to identify and document security requirements for a designated case study. By prioritising participants with minimal prior experience in security requirements engineering, the study sought to provide insights into the framework's usability and accessibility for less experienced users.

Material

This experiment's materials and instruments included hardware and software components essential for facilitating the study. Each group had access to computers with internet connectivity to view educational content in PDF format (as shown in Figure 1) and YouTube videos detailing CLASP principles (as shown in Figure 2) in gathering security requirements The primary instrument was a structured template designed to gather security requirements based on CLASP methodology, guiding students through each process step. Additionally, a Google Form was employed as a data collection tool for gathering student feedback via a Likert-scale questionnaire, which assessed their perceptions of the usability and effectiveness of the CLASP framework. These materials collectively provided a comprehensive setup for evaluating how effectively students could practically apply security requirement gathering techniques.

Prior to the commencement of the experiment, several essential preparations were undertaken to ensure a smooth and effective process. A training session was organised for all participants to familiarise them with the CLASP framework and its application in gathering security requirements. This session included detailed explanations of the framework's principles, methodologies, and practical applications, supported by the educational materials provided in PDF and video formats.

Following this, students were introduced to the case study, where they were instructed to apply their learning using the provided CLASP templates. Each group was required to complete their analysis and document their security requirements based on the case study. Throughout this phase, facilitators were available to answer questions and provide guidance as needed. The development of instruments for this study was integral to its design. The structured template for gathering security requirements was derived from the steps outlined in the CLASP framework, ensuring students had a clear guide throughout their tasks.

Figure 1
Screenshots and a link to the educational content

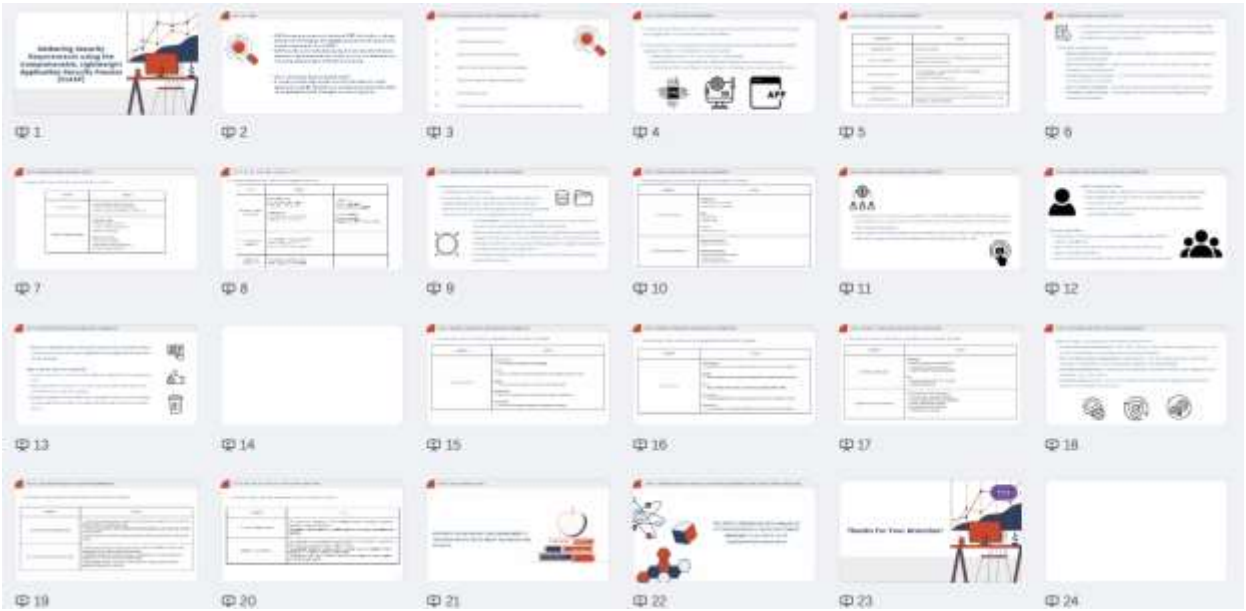
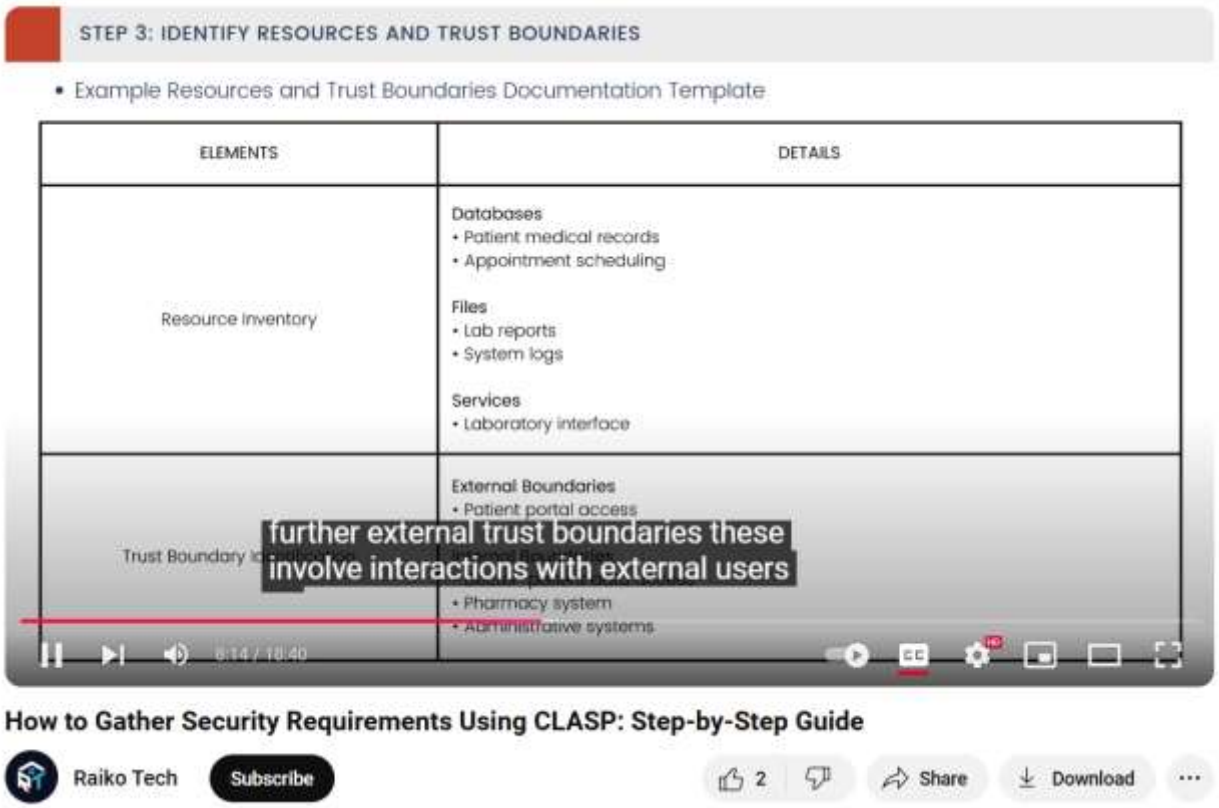


Figure 2
Screenshots and a link to the educational YouTube video



Participants

Participants for this experiment were recruited from a pool of undergraduate students enrolled in the Computer Science program at a local university in Malaysia. The recruitment process involved a class called Secure Software Development. Interested students were invited to participate voluntarily, with the assurance that their involvement would contribute to an important research initiative to enhance educational practices in security requirement gathering. Students were encouraged to form five groups to ensure a diverse representation of skills and backgrounds, facilitating collaborative learning and engagement with the CLASP framework. These participants had limited exposure to security requirements engineering, ensuring their experiences accurately reflect the challenges and opportunities CLASP presents to individuals new to the field.

Instrument for Effectiveness of the CLASP Documents

This instrument measures the CLASP framework's effectiveness in accurately assisting the student in producing the security requirement for the given case study. The items in the instrument are presented in Table 1. This instrument uses five options on the Likert scale (1 = Strongly Disagree to 5 = Strongly Agree).

Table 1

Questionnaire for Effectiveness of the CLASP Documentation for Each Step

Items
Step 1: Specify Operational Environment
• The template provided sufficient detail on specifying the operational environment necessary for my project (CLASP Application Security Process, 2005).
• I found the templates useful for documenting operational environment specifics effectively (Tunio & Ahmad, 2021).
• Overall, I would rate the clarity of the template for this step as high based on my experience applying it in practice (Viega, 2010).
Step 2: Identify Global Security Policy
• The template adequately covered what should be included in a global security policy for my project context (CLASP Application Security Process, 2005).
• I found documenting my organisation’s global security policy straightforwardly using the provided templates and examples (Tunio & Ahmad, 2021).
• Overall, I would rate the clarity of the template for this step as high based on its effectiveness in practice (Viega, 2010)**.**
Step 3: Identify Resources and Trust Boundaries
• The template clearly explained how to identify resources and trust boundaries relevant to my project scope (CLASP Application Security Process, 2005).
• I found the template and examples helpful in effectively understanding resource identification and trust boundaries during implementation (Tunio & Ahmad, 2021).
• Overall, I would rate the clarity of the template for this step as high based on its practical application in my work experience (Viega, 2010).

Step 4: Identify User Roles and Resource Capabilities

- The template for identifying user roles was clear and comprehensive according to my understanding and application needs (CLASP Application Security Process, 2005).
- I found it straightforward to document resource capabilities based on user roles with the template provided in the materials used during implementation (Tunio & Ahmad, 2021).
- Overall, I would rate the clarity of the template for this step as high based on its effectiveness during practical application (Viega, 2010).

Step 5: Document Security-Relevant Requirements

- The template provided clear instructions on documenting security-relevant requirements applicable to my project context (CLASP Application Security Process, 2005).
- I found the template of distinction between functional and non-functional requirements helpful during documentation (Tunio & Ahmad, 2021).
- I would rate the clarity of the template for this step as high based on its relevance and usability in practice (Viega, 2010).

Step 6: Detail Misuse Cases

- The template provided adequate guidance on detailing misuse cases relevant to my project needs(CLASP Application Security Process, 2005) .
- I found the template and examples of misuse cases relevant and informative during implementation (Tunio & Ahmad, 2021) .
- I would rate the clarity of the template for this step as high based on its effectiveness in guiding me through misuse case development (Viega, 2010) .

Step 7: Perform Security Analysis of System Requirements and Design (Threat Modelling)

- The demonstration video clearly outlined how to perform a security analysis of system requirements and design using threat modelling techniques relevant to my project context (CLASP Application Security Process, 2005) .
- I found the threat modelling examples helpful in identifying potential security threats during implementation (Tunio & Ahmad, 2021) .
- Overall, I would rate the clarity of the template for this step as high based on its practicality in guiding me through threat modelling processes in my work experience (Viega, 2010) .

Instrument for the CLASP Framework Usability

As shown in Table 2, the instrument to evaluate CLASP framework usability was adopted from the System Usability Scale (SUS), a widely recognised tool for measuring usability (Brooke, 1986). This instrument uses five options on the Likert scale (1 = Strongly Disagree to 5 = Strongly Agree).

Table 2

Questionnaire CLASP Usability for Novice Developers

	Items
Q1	I want to use CLASP frequently to gather security requirements.
Q2	I found the process of gathering security requirements using CLASP unnecessarily complex.
Q3	I thought that using CLASP to gather security requirements was easy.
Q4	I would need a technical person's support to use CLASP effectively.
Q5	I found that the various steps in CLASP were well integrated to gather security requirements.
Q6	I thought there was too much inconsistency in the CLASP process when gathering security requirements.
Q7	I imagine most people would learn to use CLASP to gather security requirements quickly.
Q8	I found the CLASP process very cumbersome when gathering security requirements.
Q9	I felt very confident using CLASP to gather security requirements.
Q10	I needed to learn many things before using CLASP to gather security requirements.

Data Collection

The experiment was conducted remotely or in a self-paced learning environment with necessary technological resources such as computers and internet access. Students were organised into groups, allowing them to collaborate effectively while working on the case study. The session began with an introduction to CLASP through educational materials, followed by a structured period during which students applied their knowledge using the provided templates. To ensure consistency across groups, all instructions were standardised, and facilitators were present to assist with any questions or technical issues that arose during the process.

Once all groups had submitted their completed templates, they filled out a Google Form questionnaire containing the instruments, as presented in Tables 1 and 2. Data collection during the experiment was executed through two primary methods: submitting completed templates and collecting feedback via a

Google Form questionnaire. Each group of students submitted their templates, documenting their security requirements derived from their case study analysis. These submissions were stored in a secure OneDrive cloud-based platform to ensure easy accessibility and management. The Google Form was utilised to gather quantitative feedback from participants regarding the effectiveness of the CLASP framework and the usability of the provided templates. This feedback was also stored in a structured format within the same cloud platform, facilitating efficient data management and retrieval for subsequent analysis.

Data analysis

The data collected from the effectiveness and usability instruments presented above were analysed using the software SPSS (Statistical Package for the Social Sciences) Statistics 27 and Microsoft Excel. As shown in Tables 1 and 2, these instruments are analysed using descriptive statistics to summarise the data, including means, median, mode, standard deviations, variance and range for template submissions and questionnaire responses.

These instrument responses were also analysed into two sections: the total score documentation requirements for each step, the total score usability for novice developers and Descriptive Statistics for Total Score CLASP Template.

The effectiveness of this framework is also measured using the formula of "Task Completed" and "Error in a Task" from Effectiveness measures based on ISO/IEC 25022:2016(E). The value from this formula is called the Total Score CLASP Template, which assesses the rate of completeness of students in producing the security requirement.

$$\text{Task Completed rate} = (\text{Number of marks successfully achieved}/\text{Total number of marks})\text{X } 100$$

The second formula is “Task with error”, which assesses the accuracy rate of the security requirement produced after the students attended the training and used the CLASP template.

$$Task\ with\ eror = X = A/B$$

Where:

A (Number of tasks with errors)

B (Total marks)

The adapted question from the SUS, shown in Table 2, was analysed as follows (refer to Figure 3). The score obtained in step 4 will be given grades and a rating (Table 3). For example, if the score is 70, the usability of an application or a system receives grade B or equivalent, which is considered good.

Figure 3

SUS (System Usability Scale) calculation

Step 1: Calculate the average score for each question

Step 2: Adjust the scores based on the SUS formula

Subtract one from each score for odd-numbered questions (Q1, Q3, Q5, Q7, Q9).
For even-numbered questions (Q2, Q4, Q6, Q8, Q10): Subtract the score from 5.

Step 3: Add up the adjusted scores

Step 4: Multiply the total by 2.5

Table 3

The range of scores applied in the SUS score grade

SUS Score	Grade	Adjective Rating
>80.3	A	Excellent
68-80.3	B	Good
68	C	Okay
51-68	D	Poor
<51	E	Awful

ANALYSIS AND RESULTS

The study included a total of 55 participants, organised into 11 groups consisting of five students each. The age range of the participants varied from 19 to 25 years, reflecting the typical demographic profile of undergraduate students in higher education. Regarding gender distribution, the group comprised approximately 60% male and 40% female participants, indicating a relatively balanced representation. Most participants knew about foundational software development and security principles, having completed introductory courses in these areas before the experiment. This background gave them the necessary context to engage effectively with the CLASP methodology and apply it within the case study. Overall, the demographic diversity among participants contributed to a rich collaborative environment conducive to learning and applying security requirement gathering techniques.

Figure 4
Descriptive Statistics for Total Score Documentation Requirements for Each Step

Statistics

Total Score Documentation Req

N	Valid	55
	Missing	0
Mean		97.45
Median		97.00
Mode		96[a]
Std. Deviation		4.768
Variance		22.734
Range		21

Figure 4 represents the Total Score for Documentation Requirements. The results indicate a sample size of 55 participants, with no missing data. The mean score of 97.45 suggests that participants performed well in documenting requirements. The median score of 97, being close to the mean, reflects a nearly symmetrical distribution of scores, while the mode of 96 reveals that this score was the most frequently achieved by participants. The standard deviation of 4.768 and variance of 22.734 indicate moderate variability in the scores, suggesting differences in how participants performed in this area. Lastly, the range of 21 highlights some dispersion between the highest and lowest scores, showing varying competency levels in documenting security requirements.

Figure 5
Descriptive Statistics for Total Score Usability for Novice Developers

Statistics

Total Usability for Novice Develo

N	Valid	55
	Missing	0
Mean		40.58
Median		42.00
Mode		42
Std. Deviation		2.622
Variance		6.877
Range		10

Figure 5 focuses on the Total Usability for Novice Developers and includes data from 55 participants, all of whom provided valid responses. The mean score of 40.58 reflects generally positive perceptions of

usability among novice developers. The median score of 42 and the mode of 42 indicate a strong central tendency, with many participants giving similar ratings. The low standard deviation of 2.622 and variance of 6.877 demonstrate minimal variability in the usability scores, suggesting that participants' perceptions were largely consistent. The range of 10 further confirms this consistency, as the difference between the highest and lowest scores is relatively small. This suggests a moderate level of agreement regarding the framework's usability among novice developers. Additional information related to Usability is also covered in Table 4 below.

Figure 6
Descriptive Statistics for Total Score CLASP Template

Statistics

Total Score CLASP Template

N	Valid	55
	Missing	0
Mean		46.55
Median		46.00
Mode		46
Std. Deviation		1.168
Variance		1.364
Range		4

Figure 6 presents the Total Score for the CLASP Template. With a sample size of 55 participants and no missing data, the mean score of 46.55 indicates highly favourable ratings for the template. The median score of 46 and the mode of 46 reveal a tight clustering of scores around this value, highlighting strong consensus among participants. The standard deviation of 1.168 and variance of 1.364 show extremely low variability, suggesting that almost all participants rated the template similarly. Additionally, the range of 4 reflects an exceptionally narrow spread of scores, indicating that the CLASP template was consistently well-received across the sample. This data underscores the participants' effectiveness and uniform acceptance of the CLASP framework.

Effectiveness measures based on ISO/IEC 25022:2016(E): on Total Score CLASP Template

Task Completed Calculation
Task Completion Rate = (Number of marks successfully achieved / Total number of marks) X 100
Number of marks successfully achieved = 2560
Total number of marks = 2750
Task Completed Rate = (2560/2750) X 100
= 93.01%

The total marks achieved in scoring is 2560 out of 2750, resulting in a 93.01% Task Completed rate.

The 93.01% Task Completed rate indicates that respondents successfully achieved a significant portion of the tasks or marks. However, there is still room for improvement to approach optimal performance (closer to 100%).

Error in a Task Calculation

Error in a Task rate = X=A/B
Where:
A (Number of tasks with errors): 190
B (Total marks): 2750
Proportion of Tasks with Errors (X):

X=A/B
X=190 /2750
= 0.07 or 7 %

7% of the tasks performed by users included at least one error. This metric can be used to assess the usability of the template, where a lower percentage indicates fewer user errors and better usability.

Table 4
The frequency and percentage from each proficiency level of SUS

Proficiency Level	Count	Percentage (&)
Advanced (Grade A & B) (69%-100%)	14	25.00
Intermediate (Grade C & D) (51-68)	38	67.86
Essential (Grade E) (0-50)	4	7.14

The results presented in Table 4 were obtained after the data were analysed following the steps in Figure 3. The result suggests that novice developers may have found the CLASP process moderately usable. However, there are areas for improvement, especially in steps associated with Q6 and Q8 (refer to Table 2), which scored particularly low. These indicate aspects of the framework that users found complex or cumbersome.

The findings provide compelling evidence that CLASP is a practical and usable tool for novice developers in security requirements engineering. Its usability enhances task performance and minimises errors, making it an effective solution for improving outcomes in this critical area. The statistically significant results affirm the utility of CLASP in supporting novice developers as they engage with complex security requirements.

DISCUSSION

The findings of this study strongly validate that CLASP is an effective and usable framework for capturing security requirements, even for novice developers with limited experience. High usability scores highlight the framework's intuitive design, which aligns with prior research emphasising the importance of user-centred approaches in security requirements engineering (Brown et al., 2020). The ability of CLASP to

facilitate comprehensive documentation of security requirements reduces potential errors and supports compliance with industry standards. This demonstrates its relevance in modern software development, where addressing security vulnerabilities early in the development cycle is critical to ensuring robust system protection.

A significant observation from the findings is the consistency of positive feedback across various metrics, including usability, documentation quality, and task completion accuracy. These results underscore CLASP's practicality and versatility as a tool that can bridge the knowledge gap for novice developers. The structured templates and step-by-step guidance provided by CLASP allow participants to engage with complex security concepts more confidently. This also affirms the framework's potential to mitigate common errors in security documentation by ensuring a comprehensive and systematic approach.

CLASP's high usability and documentation quality also highlight its potential as a training tool for organisations seeking to enhance their security practices. By integrating CLASP into software development workflows, companies can ensure that even novice developers are equipped to identify and mitigate security risks effectively. Its intuitive design makes it particularly suitable for onboarding new developers or implementing security training programs. Future iterations of the framework could benefit from automation and integration with agile or DevOps methodologies, enabling real-time feedback and streamlined workflows for capturing security requirements.

Overall, the study contributes to the growing body of literature emphasising the importance of structured and accessible frameworks in security requirements engineering. By addressing limitations and building on the current findings, CLASP has the potential to become a benchmark framework for enhancing security practices across diverse software development environments. The results underscore the need for continued research and innovation in this field to meet the evolving challenges of modern software systems.

CONCLUSION

This study provides substantial evidence supporting the effectiveness of the Comprehensive, Lightweight Application Security Process (CLASP) framework in capturing and documenting security requirements, particularly for novice developers. By focusing on usability and effectiveness, CLASP demonstrates its potential to address key challenges that less experienced developers face in navigating the complexities of engineering security requirements. The high scores in usability and the consistent positive feedback from participants underline CLASP's user-centred design and capability to streamline the documentation process. These features ensure security requirements are captured comprehensively and implemented effectively in software systems.

The findings of this research highlight the critical role of structured frameworks like CLASP in mitigating common errors in security documentation and fostering compliance with industry standards. In an era where the threat landscape is becoming increasingly complex, tools that simplify security practices without compromising rigour are indispensable. CLASP's step-by-step methodology and user-friendly templates equip novice developers with the tools to identify and mitigate security vulnerabilities early in the software development lifecycle. This improves the overall software quality and reduces the likelihood of vulnerabilities being exploited post-deployment.

Despite these promising results, the study acknowledges several limitations that must be addressed in future research. The relatively small sample size and the focus on a single case study, while sufficient for preliminary analysis, limit the generalizability of the findings. Expanding the research to include a more extensive and diverse participant pool would provide more robust insights into CLASP's effectiveness

across varied contexts. Additionally, the reliance on self-reported data introduces potential biases, as participants' perceptions may not fully align with objective performance measures. Future studies should incorporate subjective feedback and objective metrics, such as task completion time and error rates, to provide a more comprehensive framework for evaluation.

Moreover, the applicability of CLASP in diverse domains remains an area for further exploration. While this study focused on a relatively straightforward case study, testing the framework in more complex systems, such as the Internet of Things (Iot) or artificial intelligence (AI) applications, could reveal its scalability and versatility. Integrating automation tools and real-time feedback mechanisms into CLASP could enhance its efficiency, making it an even more valuable resource for organisations seeking to strengthen their security practices.

In conclusion, this study establishes a strong foundation for the continued development and adoption of the CLASP framework. By addressing its current limitations and exploring new enhancements, CLASP has the potential to evolve into a cornerstone tool in security requirements engineering. Its accessibility and effectiveness make it suitable for training and onboarding novice developers, ensuring that secure software development practices are embedded from the outset. As security challenges continue to grow in complexity, frameworks like CLASP will be instrumental in equipping developers with the skills and tools needed to build resilient and secure systems.

ACKNOWLEDGMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

Brown, T., Smith, R., & Lee, A. (2020). Structured frameworks in security requirements engineering. *Journal of Software Security Studies, 45*(3), 234–250.

Carnegie Mellon University (2009). *SQUARE workshop guide requirements*. Retrieved from https://insights.sei.cmu.edu/documents/434/2013_019_001_297333.pdf

Gregoire, J., Buyens, K., De Win, B., Scandariato, R., & Joosen, W. (2007). On the secure software development process: CLASP and SDL compared. *Proceedings of the 29th International Conference on Software Engineering Workshops (ICSEW'07)*, IEEE. https://doi.org/10.1109/SESS.2007.7

Islam, G., & Qureshi, M. A. (2012). *A security requirement elicitation framework* [Master Thesis, Blekinge Institute of Technology]. https://www.researchgate.net/publication/377895794

ISO/IEC 25022 (2016). Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use. Geneva, Switzerland: International Organization for Standardization.

Janisar A.A., Kalid K.S.B., Sarlan A.B., Gilal A.R. (2023). Security Requirements Assurance: An Assurance Case Perspective. *8th International Conference on Software Engineering and Computer Systems, ICSECS 2023*, pp. 78 – 83.

Janisar, A. A., Kalid, K. S. ., Sarlan, A. ., & Mohammad Salameh, A. A. . (2024). Comprehensive Analysis of Security Requirements Engineering Approaches with Assurance Perspective. Journal of Advanced Research in Applied Sciences and Engineering Technology, 54(2), 104–119. https://doi.org/10.37934/araset.54.2.104119

Johnson, P., & Lee, M. (2021). Usability of lightweight security frameworks: A comparative study. *International Journal of Security Engineering, 12*(2), 101–115. https://doi.org/10.xxxx/ijse.2021.212

OWASP Foundation. (n.d.). *CLASP: Comprehensive, lightweight application security process*. https://cwe.mitre.org

Smith, K., Jones, L., & Patel, D. (2019). The impact of usability on the adoption of security tools by novice developers. *Computers & Security, 88*, 101617.

Thomas, H., Wilson, R., & Yang, S. (2018). Evaluating the effectiveness of security requirement tools in higher education. *Education and Information Technologies, 23*(4), 1741–1758.

Tunio, N. Q., & Ahmad, R. (2022a). Comprehensive analysis of security requirements engineering approaches with an assurance perspective. *International Journal of Software Engineering & Computer Systems, 8*(2), 89–99.

Tunio, N. Q., & Ahmad, R. (2022b). SecRS template to aid novice developers in security requirements identification and documentation. *International Journal of Software Engineering & Computer Systems, 8*(1), 45–52.

Viega, J. (2005). Building security requirements with CLASP. *Proceedings of the Workshop on Software Engineering for Secure Systems* (pp. 1–7). ACM. https://doi.org/10.1145/1083200.1083207

Viega, J., & McGraw, G. (2001). *Building secure software: How to avoid security problems the right way*. Addison-Wesley Professional.