Navigating Cybersecurity in Global Software Development: Insights, Challenges, and Practices

Agha Muhammad Yar Khan[1], Abdul Samad Danish[2], Farwah Aizaz[1], Huzaifa Bilal[1], Abdullah Shahrose[2], Rana Muneeb Asad[2], Muhammad Talha Rafiq[2]

[1] Department of Software Engineering, HITEC University Taxila
[2] Department of Computer Science, HITEC University Taxila

Abstract- This study examines the domain of cybersecurity in relation to global software development (GSD), investigating its risks, challenges, and methodologies via an exhaustive review of the literature and survey. The results underscore the complex and diverse aspects of cybersecurity in GSD and emphasize the critical significance of global frameworks including OWASP, NIST, and ISO/IEC 27001 in providing direction for secure development procedures among teams that are geographically separated. Notwithstanding challenges such as inexperience and limited resources, the organizations that were surveyed exhibit a dedication to augmenting security measures via training initiatives, automated technologies, and fostering a culture of consciousness. Although practitioners have a moderate amount of faith in current cybersecurity frameworks, there are areas where they could be enhanced to be more comprehensive and to be easier to implement in a variety of contexts.

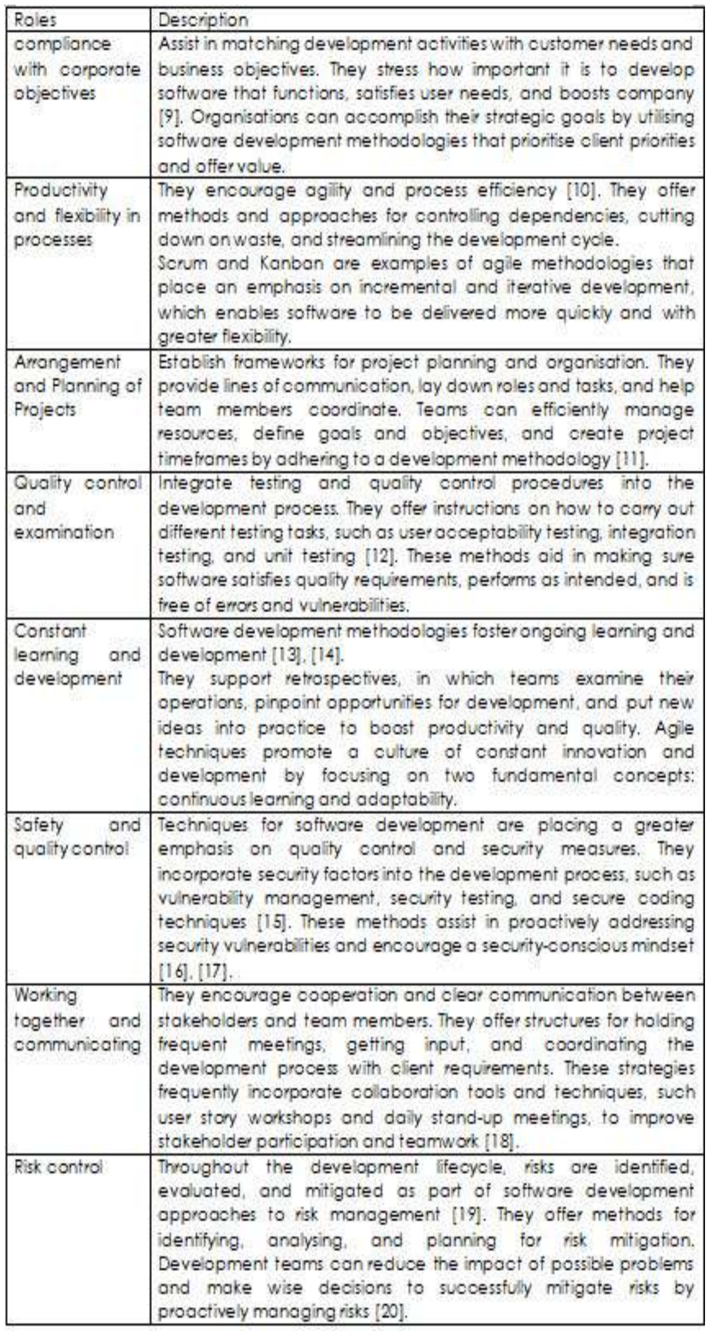
***Index Terms*-** Global Software Development (GSD), Cybersecurity Frameworks, Security Awareness, Cross-Cultural Security Challenges

I. Introduction

Integrating cybersecurity safeguards into software development processes is an imperative undertaking in the contemporary digital landscape. The importance of developing secure software cannot be overstated, given the increasing reliance of businesses on software solutions to function. This research study endeavors to perform an exhaustive literature review on cybersecurity in the software development industry, with a specific emphasis on the myriad security risks and challenges that are prevalent in this sector. This paper examines the probability of insider threats occurring, the repercussions of data intrusions, and the criticality of implementing secure coding practices right from the outset of development endeavors. The primary objective of this research is to assess the effectiveness of current cybersecurity methods, detect common weaknesses in software development frameworks, and propose a collection of optimal strategies for mitigating risks by conducting an extensive review of prior investigations. By examining these aspects, the article contributes to a broader understanding of how cybersecurity can be seamlessly integrated into the software development lifecycle, ensuring that security considerations are integrated at every phase, starting from inception to execution. This project addresses a knowledge gap in the scholarly literature and offers valuable insights for IT professionals, legislators, software developers, and developers interested in enhancing the security and resilience of software systems in the face of evolving cyberthreats. The contemporary era of development is characterized by a shift toward distributed development. Under this methodology, the software development is carried out by developers situated in remote locations, which are geographically distinct from the organization's physical location. Global Software Development (GSD) has emerged as a critical component in the current software engineering environment, effectively managing the requirements for cross-cultural, temporal, and geographical collaboration in the development process. By capitalizing on the effects of globalization, organizations are able to optimize expenses, maintain continuous operations, and access a wide range of skilled personnel. GSD facilitates the utilization of worldwide expertise and talent, resulting in novel resolutions and entry to specialized proficiencies that are absent in the local context. It facilitates cost optimization by allowing organizations to leverage labor markets characterized by diverse cost structures. GSD's distributed architecture enables uninterrupted development and support by leveraging time zone disparities to facilitate work cycles that operate around the clock. Language and time zone disparities, among other obstacles, can hinder the effectiveness of collaboration and coordination. If not effectively managed, cultural diversity has the potential to give rise to misunderstandings and conflict, despite its advantageous impact on broad perspectives. For distributed teams to maintain quality consistency, robust processes and standards are required. The integration of efficient communication tools and methodologies is vital for bridging the divides that exist between geographically dispersed teams. Team-building exercises and cultural sensitivity training have the potential to foster greater collaboration and mutual understanding. The implementation of standardized development practices and protocols guarantees uniformity and excellence throughout all teams. Security is one aspect of global software development that distinguishes itself from all other considerations. Due to the fact that software development does not occur simultaneously, distinct modules are created and subsequently integrated into the overall software. This is accomplished remotely; however, to ensure the confidentiality of the code and documentation pertaining to the software under development, we require a secure and encrypted platform for the sharing of code bases, documentation, and sensitive organization data. Thus, there is an enormous need for security and the possibility of attacks to the organization's assets in the modern era. As a result, we shall examine the fundamental cybersecurity facts pertaining to worldwide software development. Consequently, our attention will be directed towards elements such as data intrusions, insider threats, and the criticality of secure coding. Conspiracy involving data exposures is prevalent among businesses that depend on distributed software development. Unauthorized access obtained by hackers results in the exposure of sensitive information that ought not to be viewed or utilized. This organization could suffer severe financial and reputational damage as a result of this sensitive data. Given the geographical dispersion of global teams, communication and data exchange are conducted via cloud-based platforms, necessitating the implementation of tools and methodologies that facilitate global data sharing. This creates an abundance of potential breach-causing factors. A data breach that exposes the personally identifiable information of employees and consumers is an issue that many organizations must address. Over time, these types of accidents have become more frequent and can impose significant financial losses on the affected organization [1]. The costs linked to data intrusions at publicly traded companies, which compromise personal information (including that of customers and employees), are assessed by the stock market. By utilizing event study methodologies to analyze 77 incidents spanning from 2004 to 2006, it is possible to ascertain that, on average, a data breach significantly and negatively affects shareholder wealth [1]. Cyber threats, data breaches, and other forms of organizational damage can be attributed to both internal and external factors. Employees and other authorized users with access to the organization's data are considered internal factors. These users exploit authorized access to the data by employing it in an unauthorized manner, thereby preventing the data from being utilized in a permitted manner. Although internal attacks are more difficult to detect, security analysts have the ability to track back external attacks using their traceback capabilities. Thus, a variety of approaches may be implemented to conceal the harm caused by the insider attack or prevent its occurrence. In regard to global expansion, these challenges are especially pronounced due to the lack of knowledge regarding the newly recruited employee's personal security measures and approach to safeguarding the organization's assets. Furthermore, concerns regarding the employee's work ethic and filing system contribute to the implausibility of the situation. Therefore, with all hazards in mind, we conclude that the code base must be secure enough to instill confidence in the programmer that the software will not be compromised under any circumstances. Consider yourself to be in control of your organization's primary e-commerce application. You may comply if a "application service provider" requests that you modify it to operate on a server located beyond your firewall. Doing so will not compromise security as long as your design incorporates dependable and flexible third-party authentication and authorization. If your architecture adequately restricts software access, you can eliminate the possibility that an intruder could exploit your application to compromise the organization's firewall when questioned by the vice president of operations [2]. Given the increasing reliance of organizations on distributed teams to accelerate innovation and growth, the subject of cybersecurity in global software development becomes progressively more significant. However, this expansion brings forth a multitude of intricate security challenges and threats that necessitate resolution in order to safeguard data integrity, maintain stakeholder confidence, and ensure privacy. The academic literature delves into a multitude of facets pertaining to these difficulties, encompassing the susceptibilities intrinsic to distributed networks, tactics for averting insider threats, and the enforcement of secure coding practices. In the realm of global software development, Zhao and Zhang (2024) conduct a bibliometric analysis and offer a systematic commentary on the security evaluation of foreign investments in China. They emphasize the intersection of these investments with cybersecurity. The significance of a strong normative framework in tackling cybersecurity issues that emerge from global investments and collaborations is emphasized in this study [4]. In his work, Holvikivi (2024) presents a risk assessment framework that specifically addresses the cybersecurity vulnerabilities that are inherent in the process of digitalizing education. The framework discussed in this paper pertains specifically to worldwide software development endeavors within the educational domain and provides valuable perspectives on strategizing to prevent cybersecurity risks [5]. Khalid and Aldabagh (2024) examine the most recent intrusion detection datasets utilized in Software Defined Networking (SDN) environments and analyze the cybersecurity obstacles that arise from SDN threats. Their research emphasizes the susceptibilities of network components to cyber-attacks, thereby underscoring the necessity for the development of novel models that can fortify security measures. [6]. In his recent publication, Coombs (2024) examines the potential cybersecurity hazards associated with DNA data storage, a domain that is becoming increasingly significant in software development worldwide owing to the growing dependence on novel data storage methods. Key cybersecurity hazards and strategies for constructing a comprehensive infrastructure to mitigate these risks are identified in this study. [7].

II. Literature Review

As part of their research, Huang, Biczók, and Liu (2024) investigate the impact that liability waivers and audits have on the incentive effects that are associated with the creation of secure software. The assessment of the audit process that is included in this report provides substantial insights into the national cybersecurity policy as well as its implications for software development practices all around the world [8]. The cybersecurity concerns that are linked with international software development have been investigated in this article. The ways that are used to reduce these risks have also been discussed, along with the areas in which the processes that are now in place might be improved and those with deficiencies. Within the methodology portion of this work, there is a description of the search strategy that was utilized. This description includes an explanation of the research questions as well as the keywords and search queries that were utilized in order to identify relevant material within particular databases. Immediately after that, we proceeded to provide an explanation of the selection criteria that were utilized in order to refine the research articles that were obtained. After that, we went on to detail the processes that were carried out in order to assess the quality of the research papers that were selected.



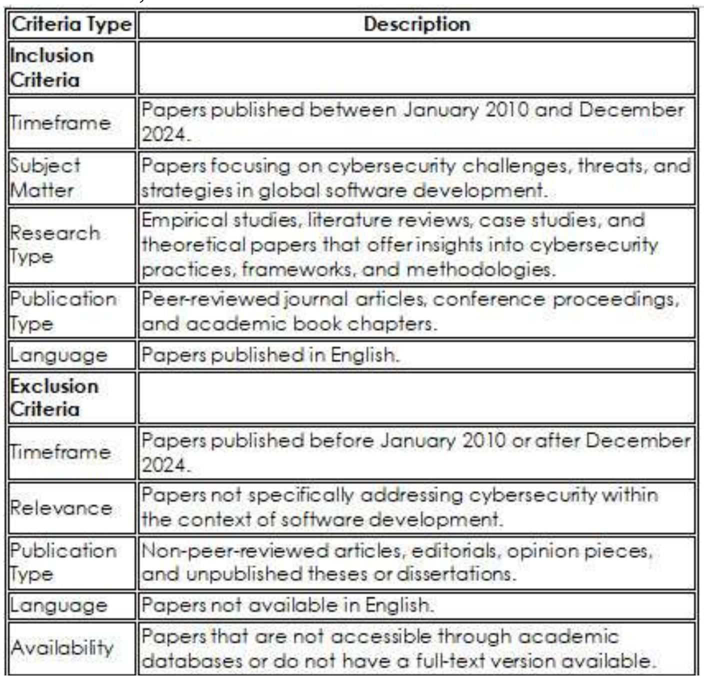
Roles	Description
compliance with corporate objectives	Assist in matching development activities with customer needs and business objectives. They stress how important it is to develop software that functions, satisfies user needs, and boosts company [9]. Organisations can accomplish their strategic goals by utilising software development methodologies that prioritise client priorities and offer value.
Productivity and flexibility in processes	They encourage agility and process efficiency [10]. They offer methods and approaches for controlling dependencies, cutting down on waste, and streamlining the development cycle. Scrum and Kanban are examples of agile methodologies that place an emphasis on incremental and iterative development, which enables software to be delivered more quickly and with greater flexibility.
Arrangement and Planning of Projects	Establish frameworks for project planning and organisation. They provide lines of communication, lay down roles and tasks, and help team members coordinate. Teams can efficiently manage resources, define goals and objectives, and create project timeframes by adhering to a development methodology [11].
Quality control and examination	Integrate testing and quality control procedures into the development process. They offer instructions on how to carry out different testing tasks, such as user acceptability testing, integration testing, and unit testing [12]. These methods aid in making sure software satisfies quality requirements, performs as intended, and is free of errors and vulnerabilities.
Constant learning and development	Software development methodologies foster ongoing learning and development [13], [14]. They support retrospectives, in which teams examine their operations, pinpoint opportunities for development, and put new ideas into practice to boost productivity and quality. Agile techniques promote a culture of constant innovation and development by focusing on two fundamental concepts: continuous learning and adaptability.
Safety and quality control	Techniques for software development are placing a greater emphasis on quality control and security measures. They incorporate security factors into the development process, such as vulnerability management, security testing, and secure coding techniques [15]. These methods assist in proactively addressing security vulnerabilities and encourage a security-conscious mindset [16], [17].
Working together and communicating	They encourage cooperation and clear communication between stakeholders and team members. They offer structures for holding frequent meetings, getting input, and coordinating the development process with client requirements. These strategies frequently incorporate collaboration tools and techniques, such user story workshops and daily stand-up meetings, to improve stakeholder participation and teamwork [18].
Risk control	Throughout the development lifecycle, risks are identified, evaluated, and mitigated as part of software development approaches to risk management [19]. They offer methods for identifying, analysing, and planning for risk mitigation. Development teams can reduce the impact of possible problems and make wise decisions to successfully mitigate risks by proactively managing risks [20].

Figure 1 Categorical literature.

In the following section of our research study, we delved into the cybersecurity challenges that arise during the process of software development. This section included case studies, the implications that emerge from these challenges, and strategies that may be utilized to mitigate the impact of such breaches and threats. Following a discussion of various tools and technology, we reached a conclusion regarding the procedures and benchmarks that were the most effective. Subsequently, we offered solutions to the research questions that we had posed, and we closed with a discussion of the gaps in the existing literature. During this discussion, we recognized the areas that deserve further investigation. In the wake of that, we arrived at a decision that outlined the necessary work that will be done in the future. For the purpose of this study evaluation, the time span that has been chosen is from the year 2000 to the year 2024, with a preference for journal publications.

III. METHODOLOGY

Typically, the process begins with the formulation of a search query concerning software development and cybersecurity. A search of deciding databases, such as Google Scholar, IEEE Xplore, PubMed, Springer, and the ACM Digital Library, constituted the initial step. Then, as subsequent essential terms, Cybersecurity, Data Leaks, Agile Software Development, Distributed Software Development, and Data Authentication were chosen. We subsequently generated the resulting queries, of which five were as follows: "cyber security" AND "software development" (2,400 results), "threats in software development" (1,330,000 results), "software development associated threats" (1.590,000 results), and "cyber threats in software engineering" (318,0 results).Following that, we implemented a publication year criterion in which we included papers from 2020 to 2024; the outcomes were subsequently altered. The query "Cyber security" AND "Global software development" yielded approximately 1,630 results spanning the years 2020 to 2024.Around 17,800 results were returned in response to the query "cyber threats in global software engineering." Regarding hazards in global software development, an estimated 110,000 results are returned in response to the query. The search for "threats linked to worldwide software development" produced an estimated 17,700 outcomes.



Criteria Type	Description
Inclusion Criteria	
Timeframe	Papers published between January 2010 and December 2024.
Subject Matter	Papers focusing on cybersecurity challenges, threats, and strategies in global software development.
Research Type	Empirical studies, literature reviews, case studies, and theoretical papers that offer insights into cybersecurity practices, frameworks, and methodologies.
Publication Type	Peer-reviewed journal articles, conference proceedings, and academic book chapters.
Language	Papers published in English.
Exclusion Criteria	
Timeframe	Papers published before January 2010 or after December 2024.
Relevance	Papers not specifically addressing cybersecurity within the context of software development.
Publication Type	Non-peer-reviewed articles, editorials, opinion pieces, and unpublished theses or dissertations.
Language	Papers not available in English.
Availability	Papers that are not accessible through academic databases or do not have a full-text version available.

Figure 2 Inclusion and Exclusion Criteria

The query "Cyber security" AND "Global software development" produced 291 outcomes from 2020 to 2024, establishing itself as the most suitable for investigating this particular domain. The criteria for inclusion and exclusion were determined to be papers published between 2020 and 2024,

which were to address the research inquiries and be relevant to the domains of cybersecurity and global software development. The papers that were incorporated based on the scoring criteria specified in the inclusion criteria are also detailed in Table 1. Table 1 further elucidates the quality assessment, which is predicated upon several factors including the years, data source, and evidence, in addition to relevance to the research query. In order to ascertain the quality and pertinence of the studies incorporated in the research pertaining to cybersecurity and global software development, a rigorous procedure was implemented. In the beginning, search queries were devised utilizing authoritative databases including Google Scholar, IEEE Xplore, PubMed, Springer, and the ACM Digital Library, with the intention of locating key terms that are relevant to the field.
This methodology ensured an exhaustive review of the current body of literature. Following this, the searches were refined by including publication dates between 2014 and 2024, with a focus on the most recent and pertinent studies. The criteria for inclusion and exclusion were clearly delineated in order to eliminate studies based on their publication year, applicability to the fields of cybersecurity and global software development, and capacity to investigate particular research inquiries. The evaluation of the study's quality and pertinence was performed by considering various factors, including the study's immediate applicability to the research inquiries, its timely publication, and the reliability of its data sources and evidence. The selection of papers was facilitated by a scoring system that adhered to the inclusion criteria. This ensured that only those papers that satisfied a rigorous standard of quality and pertinence were incorporated. By adhering to this stringent methodology, a corpus of literature was gathered that is not only up-to-date but also closely corresponds to the objectives of the research, thereby establishing a strong basis for the conclusions drawn in the study.

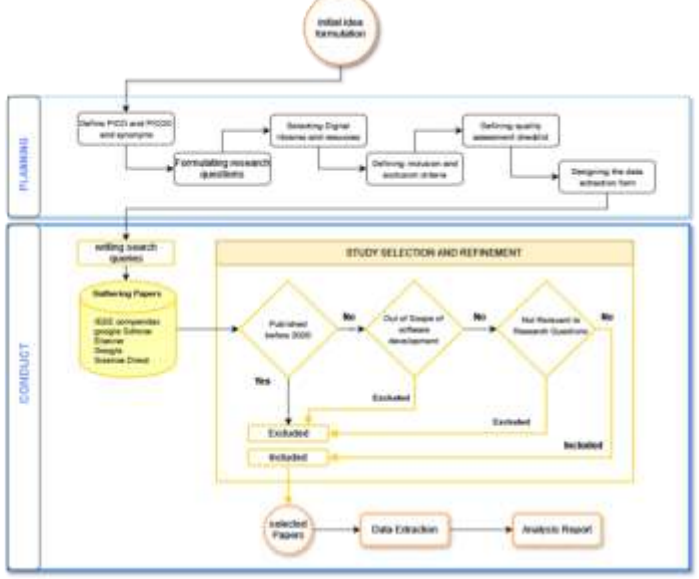
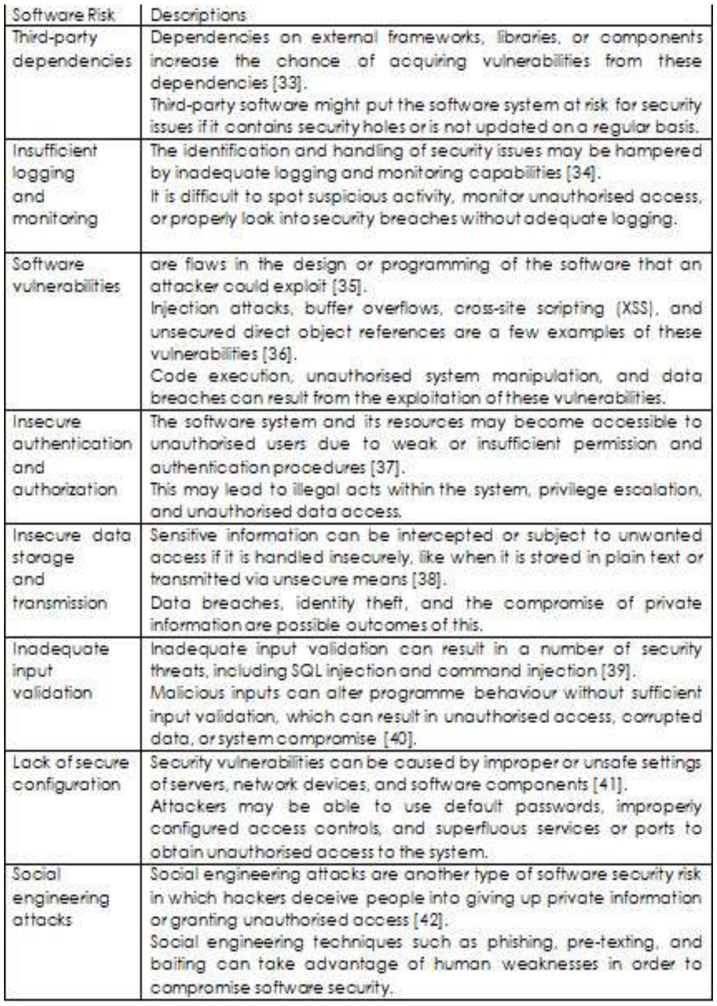

Initial idea formulation
PLANNING
Formulating research questions
Selecting Digital libraries and resources
Defining inclusion and exclusion criteria
Defining quality assessment checklist
Designing the data extraction form
CONDUCT
writing search queries
Gathering Papers
Google Scholar
Elsevier
Science Direct
STUDY SELECTION AND REFINEMENT
Published before 2020
No
Out of Scope of software development
No
Not Relevant to Research Questions
No
Yes
Excluded
Excluded
Included
Excluded
Included
selected Papers
Data Extraction
Analysis Report

Figure 3 Development Architecture

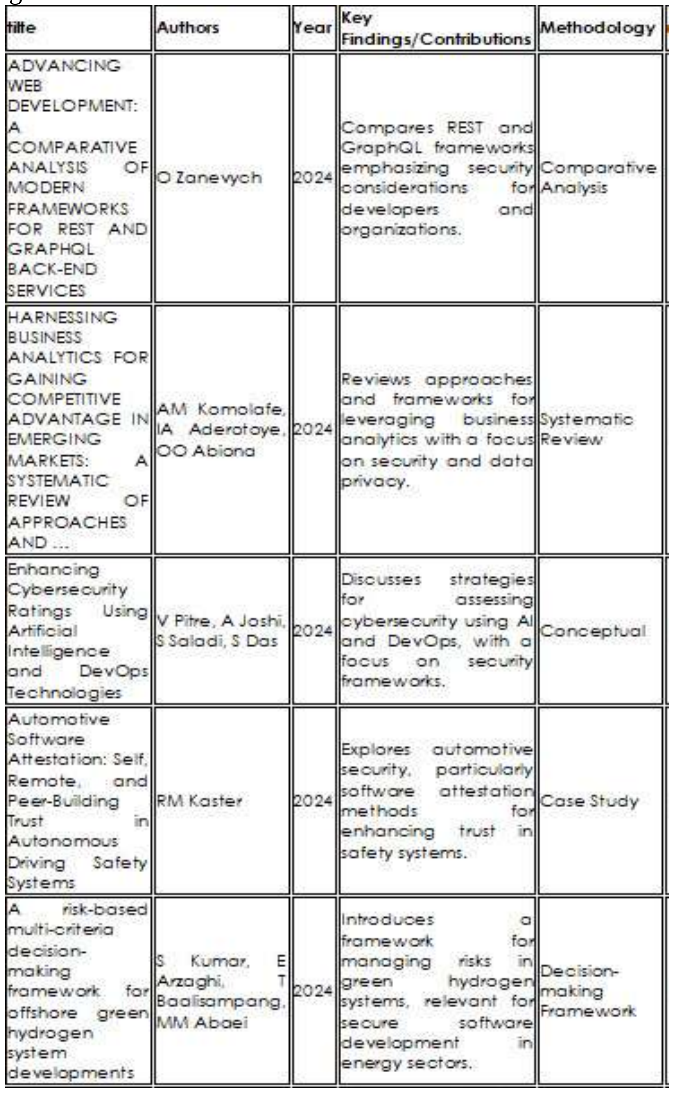
To begin with, it shall address the security dangers. Software security risks are potential threats, defects, and vulnerabilities that have the capacity to compromise the security of software systems. These threats compromise the availability, integrity, and confidentiality of the software, in addition to the data it manages. Illicit access, data breaches, system compromise, financial loss, reputational damage, and legal repercussions are among these risks. A compilation of prevalent software security vulnerabilities is detailed in Table 3. Software security risks are defined as deficiencies and susceptibilities that a malicious actor may exploit in order to compromise the integrity of software applications. Therefore, it is critical to understand and effectively handle these risks to ensure the protection of sensitive data, maintain the integrity of the system, and ensure the confidentiality and availability of software resources. Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), Insecure Direct Object References (IDOR), security misconfigurations, weak authentication and authorization, Denial-of-Service (DoS), vulnerabilities in third-party libraries, insider threats, and insecure communication are all risks associated with software systems.



Software Risk	Descriptions
Third-party dependencies	Dependencies on external frameworks, libraries, or components increase the chance of acquiring vulnerabilities from these dependencies [33]. Third-party software might put the software system at risk for security issues if it contains security holes or is not updated on a regular basis.
Insufficient logging and monitoring	The identification and handling of security issues may be hampered by inadequate logging and monitoring capabilities [34]. It is difficult to spot suspicious activity, monitor unauthorised access, or properly look into security breaches without adequate logging.
Software vulnerabilities	are flaws in the design or programming of the software that an attacker could exploit [35]. Injection attacks, buffer overflows, cross-site scripting (XSS), and unsecured direct object references are a few examples of these vulnerabilities [36]. Code execution, unauthorised system manipulation, and data breaches can result from the exploitation of these vulnerabilities.
Insecure authentication and authorization	The software system and its resources may become accessible to unauthorised users due to weak or insufficient permission and authentication procedures [37]. This may lead to illegal acts within the system, privilege escalation, and unauthorised data access.
Insecure data storage and transmission	Sensitive information can be intercepted or subject to unwanted access if it is handled insecurely, like when it is stored in plain text or transmitted via unsecure means [38]. Data breaches, identity theft, and the compromise of private information are possible outcomes of this.
Inadequate input validation	Inadequate input validation can result in a number of security threats, including SQL injection and command injection [39]. Malicious inputs can alter programme behaviour without sufficient input validation, which can result in unauthorised access, corrupted data, or system compromise [40].
Lack of secure configuration	Security vulnerabilities can be caused by improper or unsafe settings of servers, network devices, and software components [41]. Attackers may be able to use default passwords, improperly configured access controls, and superfluous services or ports to obtain unauthorised access to the system.
Social engineering attacks	Social engineering attacks are another type of software security risk in which hackers deceive people into giving up private information or granting unauthorised access [42]. Social engineering techniques such as phishing, pre-texting, and baiting can take advantage of human weaknesses in order to compromise software security.

Figure 4 Cyber Security Related Risks in Software

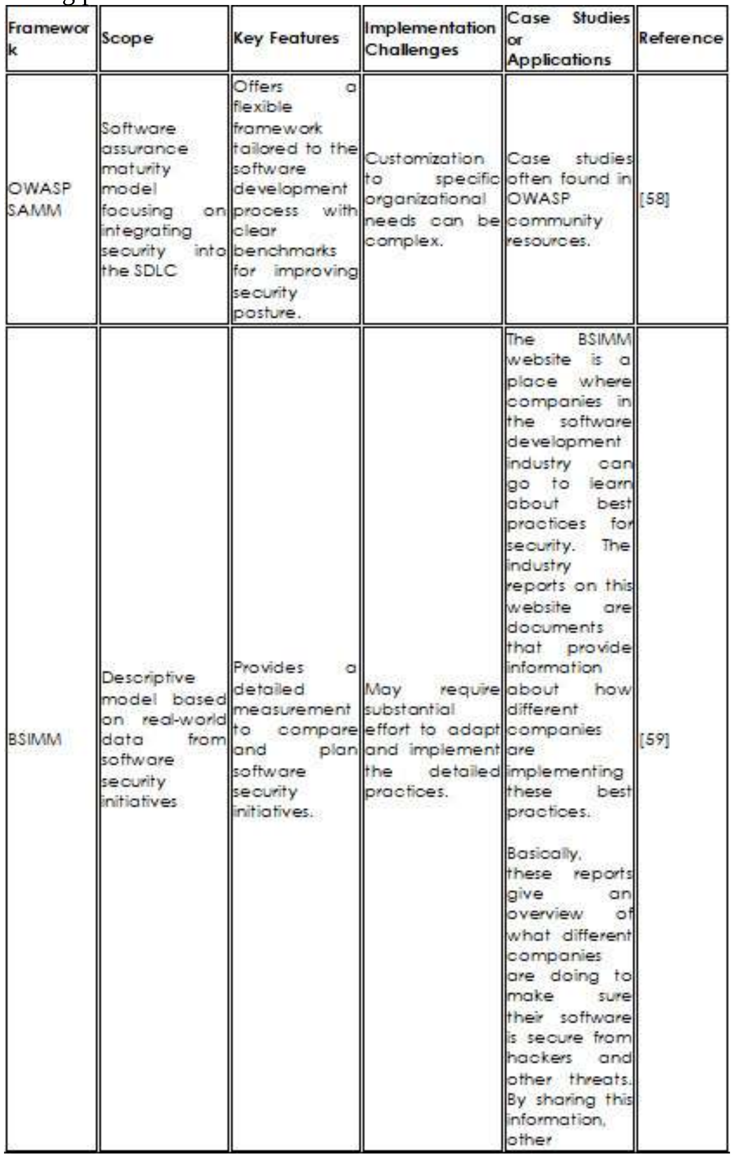
In software development, discussions pertaining to data breaches frequently center around three primary domains: occurrences of data breaches, the consequences they impose on both individuals and organizations, and proactive strategies to avert future hazards. This reply will thoroughly examine each of these domains, providing citations to studies and reports that illuminate the present comprehension and optimal methodologies in the field. Occurrences of Data BreachIn software development, data intrusions may result from a multitude of factors, encompassing human error, software vulnerabilities, and insufficient security protocols. Symantec's Internet Security Threat Report presents a noteworthy study that underscores the progressive characteristics of attack vectors and the escalating expertise exhibited by cyber assailants. An additional significant resource is Verizon's Data Breach Investigations Report, which provides an exhaustive examination of data breach incidents. Organized by industry, cause, and impact, this report offers invaluable insights into prevalent vulnerabilities that are exploited by malicious actors. Consequences of Data BreachIn addition to monetary losses, data breaches have far-reaching implications such as legal repercussions, harm to reputation, and erosion of consumer confidence. The Journal of Cybersecurity publishes a study that examines the enduring financial consequences of data breaches on organizations, emphasizing that the repercussions may persist for years following the occurrence of the breach. Furthermore, the empirical investigation conducted by the Ponemon Institute, as detailed in their yearly Cost of a Data Breach Report, establishes a monetary threshold for the mean expense incurred by organizations impacted by data breaches in various sectors. Preventive Software Development Measures multifaceted strategy is required to prevent data intrusions, including the implementation of secure coding practices, routine vulnerability assessments, and continuous monitoring. An important study published in the ACM Computing Surveys delineates optimal strategies for developing secure software. It espouses the adoption of a shift-left approach to security, wherein security considerations are incorporated at an early stage in the development process. The information security management system (ISMS) framework established by the ISO/IEC 27001 standard provides specific instructions for the methodical administration of a company's sensitive data in order to guarantee its continued security. Moreover, scholarly investigations underscore the significance of developer education and training pertaining to secure coding practices. This implies that by augmenting developers' security proficiencies, software vulnerabilities can be substantially mitigated [31]. The study examines the critical significance of cybersecurity frameworks, including the NIST Cybersecurity Framework (NIST CSF) and MITRE Cybersecurity Criteria, in mitigating the dynamic and intricate characteristics of cyber threats. Phishing, advanced persistent threats, zero-day attacks, and denial of service are a few examples of the methods by which these threats can cause industrial, public, and private organizations to incur significant financial losses. Protecting data, information, and business assets from hazards that compromise the availability, confidentiality, and integrity of information is the primary objective of cybersecurity. Organizations must ensure that their computer systems, networks, and network-connected devices are consistently updated with the most recent software, updates, and releases in order to effectively mitigate these threats. Also emphasized is the implementation of policies and procedures that regulate user interactions with network or system resources and information access. The NIST Cybersecurity Framework provides a collection of recommended guidelines, standards, and best practices for bolstering cybersecurity measures. It assists organizations in integrating cybersecurity risks into their overall risk management strategies and aligning cybersecurity activities with business objectives. Compiling efficient practices, standards, and guidelines, the framework functions as a standardized organizational structure. Developed in collaboration with industry and government authorities, the MITRE Cybersecurity Criteria delineate the prevalent strategies, methods, and protocols employed by advanced persistent threats to compromise computer systems and networks. The objective of these criteria is to safeguard and protect cyber-ecosystems while fostering cyber resilience, which enables systems to withstand, recover from, and adjust to unfavorable circumstances, pressures, breaches, or assaults. In addition to introducing numerous sections that explain the NIST CSF, CIS Critical Security Controls, ISA/IEC 62443 Cybersecurity Standard, MITRE Adversarial Tactics, Techniques, and Common Knowledge, the process of cybersecurity risk management is described. Additionally, it underscores the application of NIST CSF to critical infrastructure, thereby illustrating a preeminent application of cybersecurity maturity. The study emphasizes that substantial data exposures inflicted by cyberattacks can have detrimental effects on the financial well-being and reputation of organizations.



tilte	Authors	Year	Key Findings/Contributions	Methodology
ADVANCING WEB DEVELOPMENT: A COMPARATIVE ANALYSIS OF MODERN FRAMEWORKS FOR REST AND GRAPHQL BACK-END SERVICES	O Zanevych	2024	Compares REST and GraphQL frameworks emphasizing security considerations for developers and organizations.	Comparative Analysis
HARNESSING BUSINESS ANALYTICS FOR GAINING COMPETITIVE ADVANTAGE IN EMERGING MARKETS: A SYSTEMATIC REVIEW OF APPROACHES AND ...	AM Komolafe, IA Aderotoye, OO Abiona	2024	Reviews approaches and frameworks for leveraging business analytics with a focus on security and data privacy.	Systematic Review
Enhancing Cybersecurity Ratings Using Artificial Intelligence and DevOps Technologies	V Pitre, A Joshi, S Saladi, S Das	2024	Discusses strategies for assessing cybersecurity using AI and DevOps, with a focus on security frameworks.	Conceptual
Automotive Software Attestation: Self, Remote, and Peer-Building Trust in Autonomous Driving Safety Systems	RM Kaster	2024	Explores automotive security, particularly software attestation methods for enhancing trust in safety systems.	Case Study
A risk-based multi-criteria decision-making framework for offshore green hydrogen system developments	S Kumar, E Arzaghi, T Baalisampang, MM Abaei	2024	Introduces a framework for managing risks in green hydrogen systems, relevant for secure software development in energy sectors.	Decision-making Framework

Figure 5 Detailed Literature

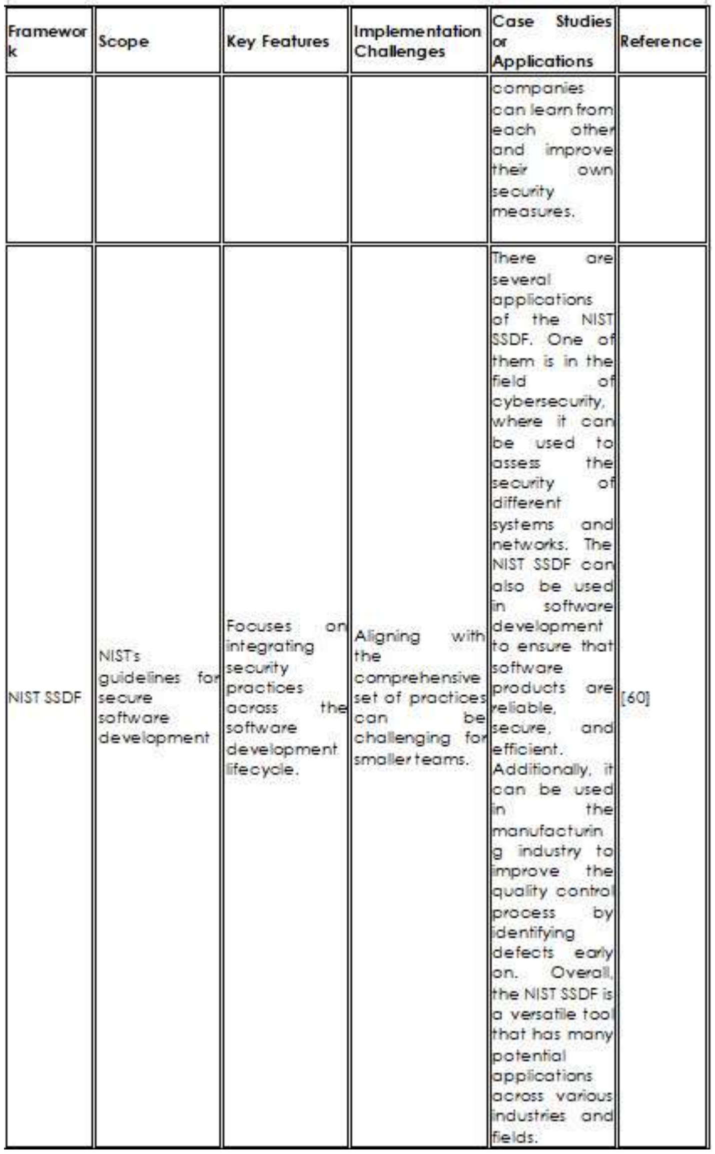
As preventive measures, maintaining awareness of cybersecurity developments, implementing rigorous access and interaction policies, and embracing comprehensive frameworks such as the NIST Cybersecurity Framework (CSF) and MITRE Criteria are all aspects that are deliberated. Implementing these measures is of the utmost importance in safeguarding information resources from unauthorized access, preserving data integrity, and assuring their availability. The paper "Analysis of Strategies for the Integration of Security Practices in Agile Software Development: A Sustainable SME Approach" offers crucial insights into the critical challenge of incorporating security practices into agile software development, with a specific focus on small and medium-sized enterprises (SMEs). This research, authored by Y Valdés-Rodríguez, J Hochstetter-Diez, and associates [32], emphasizes the increasing dangers posed by cyberattacks and the critical requirement for small and medium-sized enterprises (SMEs) to reinforce their software development procedures with strong security protocols. An exhaustive examination of the extant body of literature is conducted in order to assess various approaches to integrating security into agile development frameworks. It is noted that the inherent volatility and unpredictability of the agile environment increase the significance of implementing security protocols. Nevertheless, it is imperative that the flexibility and effectiveness that define agile methodologies remain intact. As a result of the study's analysis, a synthesis of sustainable strategies for effectively integrating security practices by SMEs is produced. The aforementioned strategies have been specifically designed to maintain the software development process's agility while effectively reducing the potential for cyber threats [32].This extensive analysis illuminates the critical equilibrium between security and agility in software development, providing a valuable guide for small and medium-sized enterprises (SMEs) attempting to navigate the intricacies of the contemporary digital threat environment [32]. In recent years, numerous models, techniques, frameworks, and approaches to software quality have been created. The Open Web Application Security Project (OWASP), System Security Engineering Capability Maturity Model (SSE-CMM), and Secure Tropos Methodology are a few examples [62]. In addition to the aforementioned methodologies, security testing has been recognized as a highly consequential, efficacious, and widely implemented strategy for enhancing software security. It has been implemented in order to detect the weaknesses and verify the proper operation of the security measures. As defined in [63], the process of developing secure software entails ensuring that the software continues to operate normally despite malevolent attacks. This entails addressing security challenges throughout the entire SDLC, with particular emphasis on the design phase [64]. As a result, the likelihood of neglecting crucial security prerequisites or introducing vulnerabilities during the execution phase is diminished. In order to construct and deploy a secure software system, security features must be incorporated into the application development life cycle and existing SSE methodologies must be standardized [65], [66]. Nevertheless, security is not a priority during the pre-development phase, as the majority of organizations perceive it to be a post-development process. As a result, the method in question lacks sanction, leading to a limited comprehension of the necessity for secure software development [67]. Insufficient evidence exists regarding the efficacy of current methodologies in addressing tangible challenges [68]. Furthermore, the extent to which current methodologies contribute to the evaluation of safety concerns remains limited [69]. As stated in [70], hazards increase the vulnerability of systems to catastrophic losses that may be challenging to recover from. Protection considerations are often neglected during the development and deployment of the majority of software programs [71], [72]. Every day, covert attack vulnerabilities emerge from within or without the organization, causing enormous financial losses in addition to breaches of confidentiality [73] and credibility. This is due to the fact that they jeopardize the accessibility and integrity of organizational data. As the programmer inadvertently leaves some bugs, the coding portion of SDLC is more susceptible to errors, according to the authors in [69] and [74]. This renders software more susceptible to potential attacks. Denial of service attacks, code execution, memory corruption, data loss, cross-site scripting, improper access control, SQL injection, buffer overflow, and integer overflow are examples of such vulnerabilities [75]. In order to mitigate these concerns, software industry researchers have implemented an extensive range of software security practices, methodologies, and approaches [76], [77], [78], [79], [80]. A number of corporations have developed maturity frameworks and models to evaluate the level of development of their software security procedures. For example, Correctness by Construction is a methodology utilized in the development of software with high integrity [81]. Anticipate requirements to change, understand why testing is being conducted, eliminate errors prior to testing, write software that is simple to verify [82], develop incrementally, acknowledge that certain aspects of software development are inherently challenging, and recognize that the software is not functional in and of itself. Seven touchpoint operations, including abuse cases; security requirements; architectural risk analysis; code review and correction; penetration testing; and security operations, are recommended by the authors in [83] and [84]. All of these touchpoints are associated with software development artifacts and their purpose is to generate secure software. In a similar vein, Microsoft has introduced the Microsoft Trustworthy Computing Security Development Lifecycle [85], an initiative that supplements its software development process with a distinct set of security practices at each stage. In contrast, the Secure Software Development Process Model (S2D-ProM) [86] has been designed to provide software engineers and developers of all levels—from novices to experts—with guidance and support in the development of secure software. Likewise, TSP Secure (Team Software Process for Secure Software Development) [87] has been developed with software teams in mind. It strives to assist them in assembling a high-performing group and organizing their efforts to yield optimal outcomes. The TSP Secure methodology places significant emphasis on software security through three key approaches: planning, development, and management; and providing training for developers and other team members regarding security-related matters [88]. The Comprehensive, Lightweight Application Security Process (CLASP), as elucidated in [89], is a rudimentary procedure comprising 24 high-level security activities that may be integrated wholly or partially into software throughout the Software Development Life Cycle (SDLC). Threat modeling and risk analysis [90] are executed in the CLASP framework throughout the requirements and design phases. It recommends secure coding standards and secure design guidelines during the design and implementation phase [91], [92], [93], [94]. Security testing, static code analysis, and inspections [94] are conducted during the assurance phase [95]. Conversely, a multi-vocal literature survey was undertaken by the authors of [96] in order to ascertain the most effective methodologies for developing secure software. On the basis of best practices that were identified, the Secure Software Design Maturity Model (SSDMM) framework was created. In a similar vein, the Security Quality Requirements Engineering (SQUARE) approach was created to streamline the process of obtaining, categorizing, and ranking security specifications for applications and systems related to information technology [97]. Additionally, Appropriate and Effective Guidance for Information Security (AEGIS) for assessing the relationships between device assets has been developed. Following that, risk analysis is conducted, during which vulnerabilities, hazards, and risks are defined [98], [99]. The Secure Software Development Model (SSDM) security training provides stakeholders in software development with sufficient security education, as stated in reference [100] [101]. During the requirements process of SSDM, a threat model is used to identify and their capabilities.153Global Journal of Engineering and Technology Advances, 2023, 14(03), 149–171As discussed in [113], Microsoft uses STRIDE to model threats to their systems. In this context, threats are delineated by considering the likelihood of identity deception, data tampering, repudiation, information leakage, denial of service, and, in the given scenario, elevation. As elucidated by the authors in reference [103], a multitude of security approaches have been devised to aid software engineers in the assessment of security risks. Among these are Secure Tropos, a security-oriented extension of the goal-driven requirements engineering methodology [105], and Attack Trees, which amalgamate use-case modeling and goal-orientation [104]. Additional approaches enable software engineers to effectively mitigate these risks. Additional software security methodologies include the Trustworthy Computing Security Development Life Cycle (SDL), McGraw's Secure Software Development Life Cycle (SSDLC) [108], the Security Requirements Engineering Process (SREP) [109], Aprville and Pourzandi's Secure Software Development Life Cycle process [110], the Haley framework [111], and the Comprehensive, Lightweight Application Security Process (CLASP). Furthermore, OWASP Security Verification Standard (ASVS) version 3.0 is a community initiative [112] that aims to standardize the functional and non-functional security controls necessary for the design, development, and testing of contemporary web applications, as explained by the authors in [88]. The ASVS, which architects, developers, testers, security professionals, and even consumers use to define what a secure application is, consists essentially of a list of application security requirements or tests [113], [114], [115]. In contrast, ISO/IEC 27001:2005 applies to all types of organizations, including commercial enterprises, government agencies, and non-profit organizations. The stipulations delineate the prerequisites for the establishment, execution, operation, supervision, evaluation, upkeep, and enhancement of a documented Information Security Management System [116] in relation to the comprehensive business risks of the organization. Furthermore, it delineates criteria for the deployment of security measures [117], [118] that are tailored to the specific requirements of individual organizations or their components. Its architecture enables the choice of proportionate and sufficient security controls that safeguard information assets and inspire confidence among stakeholders. The authors of [119] elucidate that browser identity indicators, including certificates and uniform resource locators (URLs), aid users in discerning phishing, social engineering, and other forms of attacks. Nevertheless, prior research and surveys have indicated that outdated user interfaces for browser identity are inadequate as security tools. Modern browser identity indicators have also been noted to be ineffective. As a result, in order to develop more effective identity indicators, browsers should prioritize active negative indicators, consider utilizing prominent user interfaces as user education opportunities, and integrate user research during the design phase. The aforementioned objectives have been fulfilled through the research conducted in references [120], [121], and [122]. However, only the maintenance, evolution, implementation, and feedback aspects are covered in the majority of these studies. The authors in reference [119] note that the initial software plan is developed during the requirements phase of the SDLC. A collection of initial specifications is imperative, which is obtained from diverse sources. Various techniques, including brainstorming, group sessions, and interviews, are employed to achieve this goal. Secure requirement engineering (SRE) endeavors to provide comprehensive security by incorporating fundamental security functions, including but not limited to confidentiality, integrity, and availability. This phase involves activities such as security requirements identification and inception, documentation, elicitation, analysis and negotiation, mapping, verification and validation, prioritization and management, authentication, and authorization [10], [123], [124]. Prior research has identified the most prevalent methods for addressing security concerns during the requirements phase of the software development life cycle (SDLC) [10], [124], [123], [125], [126], [124], [127], [128]. As explained in [1] and [162], the design phase is one of themost creative stages of the SDLC, and is therefore important from the viewpoint of security [129]. As stated by the authors in [1], design-level flaws are the most prevalent sources of security hazards in software systems; during this phase, fifty percent of software defects are identified and detected. The security design architecture in this context mandates the use of design methods including strongly typed programming, least privilege, threat modeling development, and attack surface analysis and minimization. Therefore, it is imperative that the software developer incorporates security best practices into the design process in a secure and suitable manner. Certain design security practices that are commonly employed in the development of secure software have been elaborated upon in the following references: [1], [86], [130], [121], [132], [162], [133], [134], [135], [130], [136]. It has been noted by the authors in [29] that coding errors account for 80% of system penetrations in commercial software. Bad code is associated with increased costs, bugs, and security concerns. In order to fulfill deadlines, software developers are subject to time-to-market constraints. Furthermore, a deficiency in security expertise is evident, and developers neglect to adhere to secure code guidelines. It is assumed that perimeter security is adequate for safeguarding applications. In order to rectify this, security code evaluations must be performed concurrently with functionality checks, whether performed manually or automatically. The objective at hand is to validate the foundational principles of software security [86], [141]. In addition, the programmers must be aware of implementation-level vulnerabilities when writing secure code and they must utilize the documentation and guidelines created in earlier stages to help them write secure code. As such, the authors in [142], [29], [86], [143], [144], [145] and [146] have discussed some of the prescriptive actions to increase security during the coding phase of SDLC. As explained in [147], software testing is the most time-consuming, Global Journal of Engineering and Technology Advances, and costly phase of the SDLC, whose goal is to identify and fix any bugs or errors in the system. Here, security testers employ misuse cases, threat models and design documents to detect potential attacks and the consequences of successful attacks. Upon the completion of security testing, test documents containing security test cases [148] and a prioritized list of vulnerabilities resulting from automated and manual dynamic analysis are created. In this regard, some of the prescriptive actions to increase security during the testing phase of SDLC are described in [48], [86], [87],[146], [147], [149], [150], [151], [152].As discussed in [153] and [154], after the software is deployed into its operational environment, it is critical to monitor responses to flaws and vulnerabilities of the system to check for new evolved security patterns. After the identification of new security patterns, the same should be included in the requirement stage for further security improvements in subsequent releases. Here, static analysis and peer review are two useful procedures for mitigating or minimizing newly discovered vulnerabilities [155], [156]. Thereafter, final security reviews and audits are performed during the secure deployment phase, in which customer satisfaction is vital. Some of the prescriptive actions to increase security during the deployment phase of SDLC are identified in [157], [29], [158], [239], [159], [160]. Before deploying software, administrators must understand the software's security stance such that some of the identified faults that were not addressed previously are revisited, prioritized, and corrected after deployment. This is followed by the tracking of new threats by the maintenance team such that they are addressed promptly to prevent security breaches [161]. Some of the approaches to increase security during the maintenance phase of SDLC are identified in [29], [162], [163], [164]. As discussed in [15], security activities during the requirement phase serve three purposes. To start with, initial security requirements [165], [166] are identified and implemented. Secondly, with the security requirements in hand, the project team understands and recognizes the importance of security. Thirdly, with the needs of security in the hands, budget, resources, and time of security activities in future stages can be better estimated. The authors in [167], [168] explain that during the design phase, the project team focuses on identifying the attacker's interests, potential access points, and critical security areas. This is followed by the identification of threats running on the software. Basically, all the security data collected in the design phase goes into the threat models, which are important milestone in terms of secure software. This involves gauging whether the security building function offers full details of how the software can be attacked, the asset that is likely to be attacked, the areas of attack that are attractive, and the kind of threats [169]. Based on this information, the security structure is continuously updated to cater for new threats. As explained in, the implementation phase plays a twofold role from a security perspective. Tostart with, it prevents security errors entering the software. Secondly, it detects existing software errors. Here, the firstrole is accomplished by writing a secure code while the second role of detecting security errors begins with static analysis by automated tools. After automatic analysis, a manual update is performed. Thereafter, the software is fully functional and ready to go to the testing phase.



Framewor k	Scope	Key Features	Implementation Challenges	Case Studies or Applications	Reference
OWASP SAMM	Software assurance maturity model focusing on integrating security into the SDLC	Offers a flexible framework tailored to the software development process with clear benchmarks for improving security posture.	Customization to specific organizational needs can be complex.	Case studies often found in OWASP community resources.	[58]
BSIMM	Descriptive model based on real-world data from software security initiatives	Provides a detailed measurement to compare and plan software security initiatives.	May require substantial effort to adapt and implement the detailed practices.	The BSIMM website is a place where companies in the software development industry can go to learn about best practices for security. The industry reports on this website are documents that provide information about how different companies are implementing these best practices. Basically, these reports give an overview of what different companies are doing to make sure their software is secure from hackers and other threats. By sharing this information, other	[59]

Figure 6 Security Frameworks

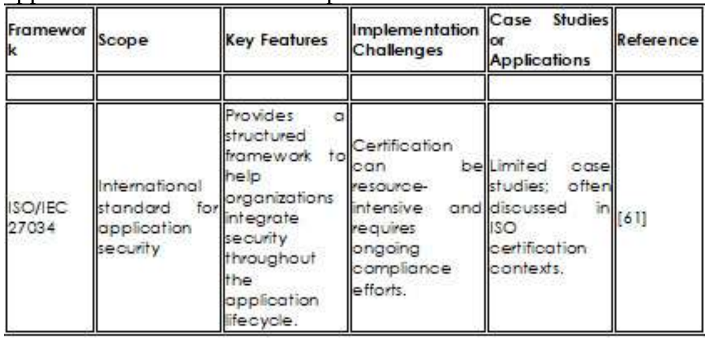
According to tests are performed mainly on test cases generated during test planning. Here, the testing team identifies security errors, reports to the development team, and the development team corrects them in this code. The testing phase ends when all test cases are conducted, and retrospective testing of all sensitive areas has taken place. Similar to other forms of testing, security testing involves the determination of who should do it and what activities should be undertaken. OWASP SAMM (Software Assurance Maturity Model): A framework to help organizations formulate and implement a strategy for software security that is tailored to the specific risks facing the organization. BSIMM (Building Security In Maturity Model): A descriptive model of industrial-grade software security initiatives based on real-world data. It helps organizations understand, measure, and plan their software security initiatives. NIST Secure Software Development Framework (SSDF): Provides a core set of high-level secure software development practices that can be integrated into each phase of the software development lifecycle. ISO/IEC 27034: An international standard providing guidelines for application security and a framework that organizations can follow to integrate security into the lifecycle of their applications.



Framework	Scope	Key Features	Implementation Challenges	Case Studies or Applications	Reference
				companies can learn from each other and improve their own security measures.	
NIST SSDF	NIST's guidelines for secure software development	Focuses on integrating security practices across the software development lifecycle.	Aligning with the comprehensive set of practices can be challenging for smaller teams.	There are several applications of the NIST SSDF. One of them is in the field of cybersecurity, where it can be used to assess the security of different systems and networks. The NIST SSDF can also be used in software development to ensure that software products are reliable, secure, and efficient. Additionally, it can be used in the manufacturing industry to improve the quality control process by identifying defects early on. Overall, the NIST SSDF is a versatile tool that has many potential applications across various industries and fields.	[60]

Figure 7 Security Frameworks

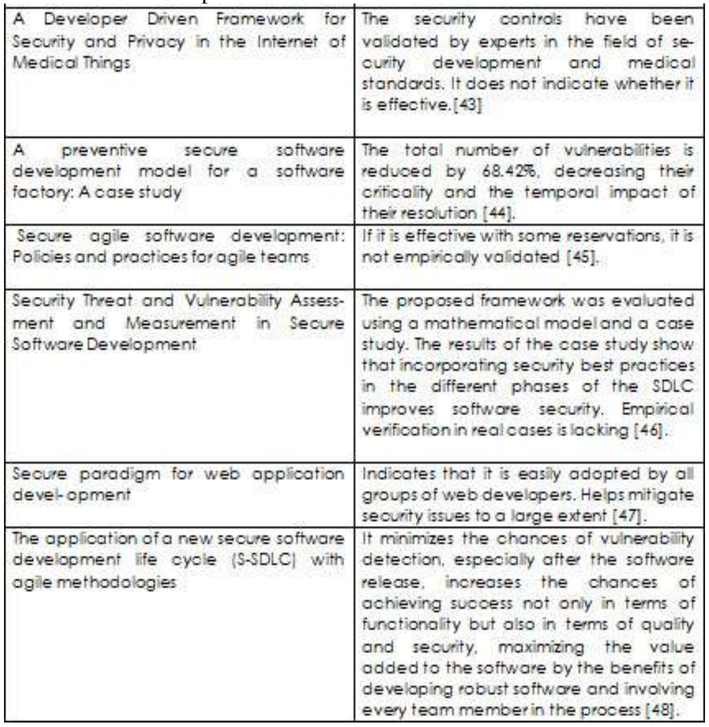
To create a literature review table for the widely recognized frameworks for secure software development, I searched for scholarly articles and resources discussing or applying OWASP SAMM, BSIMM, NIST Secure Software Development Framework (SSDF), and ISO/IEC 27034. However, it appears that specific academic papers directly analyzing or comparing these frameworks within the specified time range are limited or not readily identifiable through the search. This might be due to the nature of these frameworks being more commonly discussed in industry reports, white papers, and security guidelines rather than in academic research. Given this challenge, I recommend exploring industry reports, official framework documentation, and security analysis articles for detailed insights on each framework. These sources often provide rich information on the frameworks' implementation, effectiveness, and comparisons in various organizational contexts. For academic purposes or when specific scholarly analysis is required, broader searches encompassing software security practices, application security frameworks, and secure development lifecycle methodologies may yield related studies. These can offer insights into the principles underlying the frameworks and their practical application in software development.



Framework	Scope	Key Features	Implementation Challenges	Case Studies or Applications	Reference
ISO/IEC 27034	International standard for application security	Provides a structured framework to help organizations integrate security throughout the application lifecycle.	Certification can be resource-intensive and requires ongoing compliance efforts.	Limited case studies; often discussed in ISO certification contexts.	[61]

Figure 8 Security Frameworks

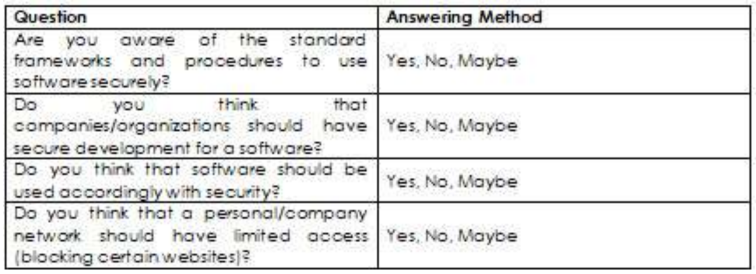
Following is the table stating the effective practices for security in software development.



A Developer Driven Framework for Security and Privacy in the Internet of Medical Things	The security controls have been validated by experts in the field of se-curity development and medical standards. It does not indicate whether it is effective.[43]
A preventive secure software development model for a software factory: A case study	The total number of vulnerabilities is reduced by 68.42%, decreasing their criticality and the temporal impact of their resolution [44].
Secure agile software development: Policies and practices for agile teams	If it is effective with some reservations, it is not empirically validated [45].
Security Threat and Vulnerability Assess-ment and Measurement in Secure Software Development	The proposed framework was evaluated using a mathematical model and a case study. The results of the case study show that incorporating security best practices in the different phases of the SDLC improves software security. Empirical verification in real cases is lacking [46].
Secure paradigm for web application devel-opment	Indicates that it is easily adopted by all groups of web developers. Helps mitigate security issues to a large extent [47].
The application of a new secure software development life cycle (S-SDLC) with agile methodologies	It minimizes the chances of vulnerability detection, especially after the software release, increases the chances of achieving success not only in terms of functionality but also in terms of quality and security, maximizing the value added to the software by the benefits of developing robust software and involving every team member in the process [48].

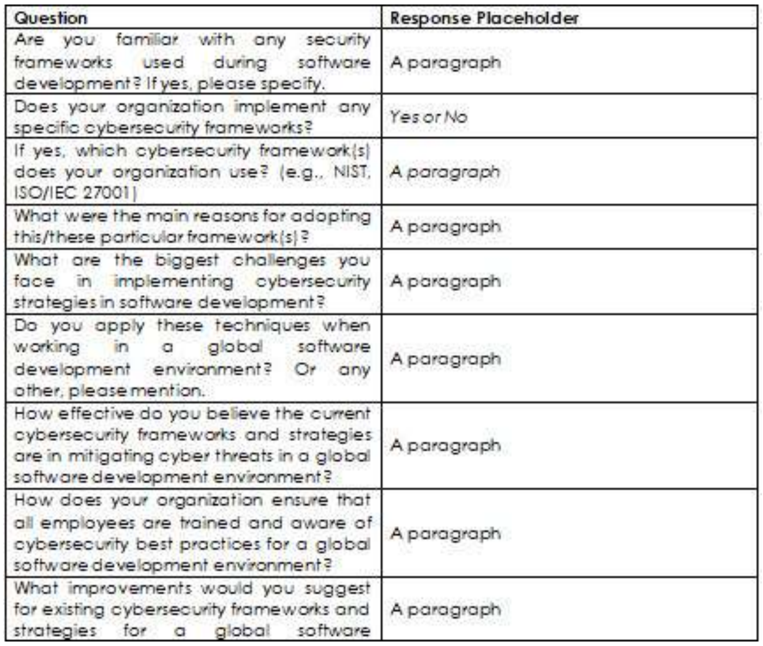
Figure 9 Analysis of Different Frameworks

The presence of insecure software has a detrimental impact on the reputation of an organization among its consumers, partners, and investors. Moreover, the necessity for organizations to repair unreliable applications may result in increased expenses. As a result, there is a possibility that subsequent development endeavors will be postponed, as scarce resources [156] will be allocated to rectify existing software shortcomings [29]. The existing body of literature concerning requirement security has identified various security hazards that may arise in the absence of initial security incorporation [157]. For example, several security risks that are intrinsic to the requirements portion of the Software Development Life Cycle (SDLC) have been examined and documented in references [10], [83], [112], [150], [158], [159]. Furthermore, it has been observed that design defects constitute a prevalent source of security threats [160] in software systems [75], [161]. According to [162], the majority of software defects are identified throughout the design phase of the SDLC. This is due to the fact that the SDLC design process forms the basis for the development of a secure software system [163]. The literature has identified several prevalent security challenges that arise during the software development process [50], [75], [161], [113], [115], [164]. Therefore, by minimizing hazards in this stage, the amount of effort required in subsequent stages can be reduced [1], [165]. Appropriate security protections [166], analyses, and countermeasures must be integrated into each phase of the SDLC in order to deliver code that is more secure [94], [167]. The current trend among developers is to incorporate functionality from third party free open-source software (FoSS) libraries as dependencies into their applications [168], [169], as discussed in [168] and [169]. This practice of software engineering enables programmers to utilize FoSS libraries as fundamental components. This may potentially reduce the time and cost of development. Even for proprietary software, the proportion of indigenous code decreased to 5%, per. Software industry reports indicate that the size of the own code base is four times smaller than the average size of third-party code inherited through dependencies. Homegrown code constitutes a negligible proportion of the overall code base that is delivered to consumers in the contemporary software ecosystem. A substantial leverage, on the other hand, necessitates the deployment of multiple libraries, which may incur costs for integration and updates. Moreover, third-party library updates are infrequently implemented by developers. This is due to the potential for disrupting, incompatible modifications to be introduced. The utilization of numerous libraries expands the potential for attacks, and it is well-known that third-party libraries can introduce security vulnerabilities and functionality flaws into the applications that employ them. There are instances where dependent projects continue to utilize obsolete components for at least a decade, thereby expanding the window of opportunity for potential exploits. Scholarly works have demonstrated that developers frequently respond to problems arising from their own code, libraries, or direct dependencies. Nevertheless, it is well-documented that transitive dependencies can partially introduce security vulnerabilities. Several technical studies have demonstrated that despite the fact that FOSS dependencies are utilized extensively by commercial and FOSS projects, they are frequently not maintained correctly. A significant proportion of projects, for example, have obsolete dependencies. The aforementioned information was derived solely from cybersecurity as a subfield of software development and not from software development as a whole. We therefore conducted a survey and developed a questionnaire containing the following information in order to fill this void. In order to investigate practical software security practices that occur throughout the worldwide software development lifecycle, we conducted interviews with licensed developers [170]-[179].



Question	Answering Method
Are you aware of the standard frameworks and procedures to use software securely?	Yes, No, Maybe
Do you think that companies/organizations should have secure development for a software?	Yes, No, Maybe
Do you think that software should be used accordingly with security?	Yes, No, Maybe
Do you think that a personal/company network should have limited access (blocking certain websites)?	Yes, No, Maybe

Figure 10 Survey Questions

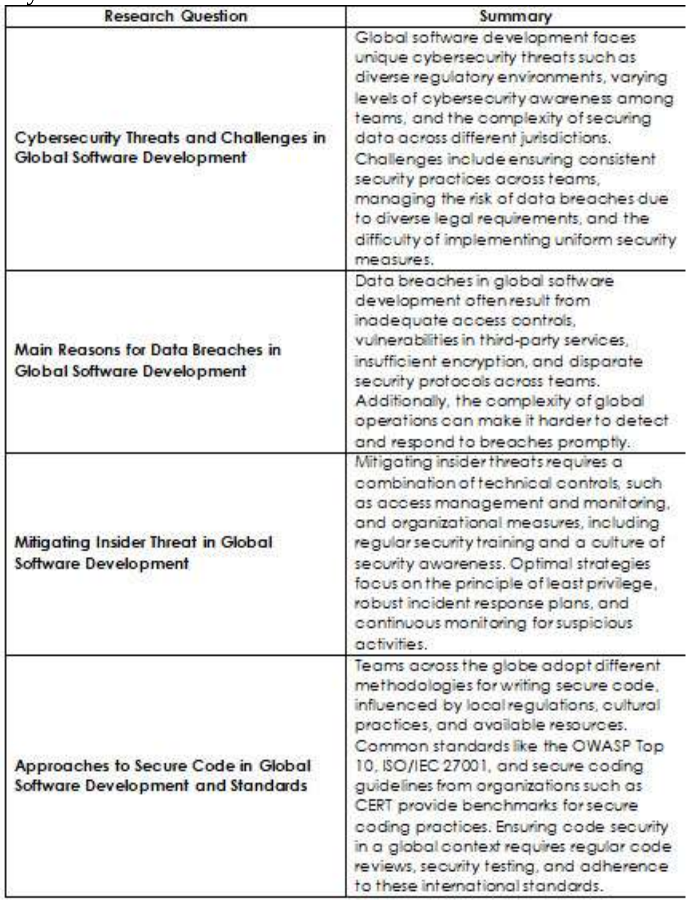


Question	Response Placeholder
Are you familiar with any security frameworks used during software development? If yes, please specify.	A paragraph
Does your organization implement any specific cybersecurity frameworks?	Yes or No
If yes, which cybersecurity framework(s) does your organization use? (e.g., NIST, ISO/IEC 27001)	A paragraph
What were the main reasons for adopting this/these particular framework(s)?	A paragraph
What are the biggest challenges you face in implementing cybersecurity strategies in software development?	A paragraph
Do you apply these techniques when working in a global software development environment? Or any other, please mention.	A paragraph
How effective do you believe the current cybersecurity frameworks and strategies are in mitigating cyber threats in a global software development environment?	A paragraph
How does your organization ensure that all employees are trained and aware of cybersecurity best practices for a global software development environment?	A paragraph
What improvements would you suggest for existing cybersecurity frameworks and strategies for a global software	A paragraph

Figure 11 Survey Questions

IV. RESULTS AND DISCUSSION

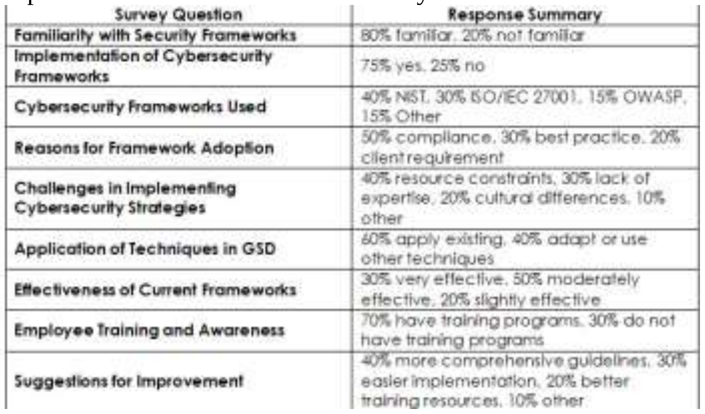
Following were the research questions that came up and we found answers for these questions. Q1: what are cybersecurity threats and challenges faced while doing global software development? Global Software Development (GSD) introduces unique cybersecurity challenges due to its distributed nature, diversity of teams, and varying compliance requirements across jurisdictions. Despite the acknowledged importance of integrating security practices throughout the SDLC, research indicates a gap in specifically addressing these practices within the context of GSD. To address this gap, we conducted a comprehensive survey and literature review, revealing several key threats and challenges, as well as the frameworks and practices employed to mitigate them. Diverse Regulatory and Compliance Standards: Cross-border teams encounter a formidable obstacle in the form of an intricate web of data protection regulations, which may impede the implementation of standardized security procedures [21].Diverse Security Awareness and Practices: Inconsistencies in security practices may result from substantial variations in the levels of cybersecurity preparedness and awareness among teams situated in various regions [22].Difficulties in Collaboration and Communication: Distributed teams face greater obstacles in effectively communicating security practices, which can result in vulnerabilities [23].GSD frequently depends on platforms and services provided by third parties, which exposes it to external risks and requires confidence in the security protocols of third parties [24].One potential concern associated with GSD is the heightened susceptibility to intellectual property theft or leakage across jurisdictions due to its distributed nature. The survey we conducted focused on security practices within the global SDLC context. It involved participants from diverse regions and aimed to gain insights into the frameworks and strategies that were utilized. The prevalence of secure SDLC frameworks was evident among the respondents, who frequently cited OWASP SAMM, BSIMM, and ISO/IEC 27034 for integrating security into SDLC. This underscores the significance of employing structured approaches to security management. Security Practices and Tools: Regular security training for developers, the incorporation of security testing into CI/CD pipelines, and the utilization of automated security tools (e.g., SAST, DAST) were frequently referenced practices. Adoption of Access Control and Encryption: Strict access control policies and an emphasis on data protection in transit and at rest were identified as critical practices. Collaborative security planning was identified as critical for overcoming communication barriers and ensuring consistent security standards. It entailed effective team collaboration and planning, with an emphasis on the early and ongoing integration of security measures. What are the primary factors contributing to the occurrence of data intrusions within a global software development environment? Research conducted in academic institutions [25] has brought to light the fact that inadequate data protection measures continue to be a major concern in the field of worldwide software development. Because there are not enough robust encryption procedures in place, as well as poor data protection during transmission and storage, sensitive information is susceptible to being accessed by unauthorized parties. The fact that different teams have different security policies adds another layer of difficulty to this problem. Differences in how priorities are prioritized and how they are implemented might lead to vulnerabilities in the development process [26]. Furthermore, relying on weak third-party services gives rise to dangers that have the potential to damage the security of entire projects [27]. The issue is further complicated by the fact that incorrect controls may result in unlawful data access [28]. Managing access controls in a globally distributed team is a challenge that is complicated further. Additional elements that contribute to the hazards of data breaches are brought to light by the experiences of the industry. owing to linguistic and cultural differences, communication barriers might inadvertently create vulnerabilities owing to misinterpretations or omissions in security standards. These vulnerabilities can allow for unauthorized access to sensitive information. Under the influence of rapid development pressures in competitive marketplaces, security efforts may be prioritized over speed, which can result in code deployment that has not been thoroughly tested or reviewed. There is an increase in the danger of non-compliance and consequent legal implications when regulatory compliance is fragmented, which adds complexity to the situation. As a result of the rise in remote work, which has been driven by global phenomena such as the pandemic, the attack surface has been expanded. This is because members of the team are accessing sensitive information from networks that may not be safe.



Research Question	Summary
Cybersecurity Threats and Challenges in Global Software Development	Global software development faces unique cybersecurity threats such as diverse regulatory environments, varying levels of cybersecurity awareness among teams, and the complexity of securing data across different jurisdictions. Challenges include ensuring consistent security practices across teams, managing the risk of data breaches due to diverse legal requirements, and the difficulty of implementing uniform security measures.
Main Reasons for Data Breaches in Global Software Development	Data breaches in global software development often result from inadequate access controls, vulnerabilities in third-party services, insufficient encryption, and disparate security protocols across teams. Additionally, the complexity of global operations can make it harder to detect and respond to breaches promptly.
Mitigating Insider Threat in Global Software Development	Mitigating insider threats requires a combination of technical controls, such as access management and monitoring, and organizational measures, including regular security training and a culture of security awareness. Optimal strategies focus on the principle of least privilege, robust incident response plans, and continuous monitoring for suspicious activities.
Approaches to Secure Code in Global Software Development and Standards	Teams across the globe adopt different methodologies for writing secure code, influenced by local regulations, cultural practices, and available resources. Common standards like the OWASP Top 10, ISO/IEC 27001, and secure coding guidelines from organizations such as CERT provide benchmarks for secure coding practices. Ensuring code security in a global context requires regular code reviews, security testing, and adherence to these international standards.

Figure 12 Research Questions

When it comes to mitigating the risks that are represented by insiders in the context of global software development, it is vital to take a broad strategy. In the context of academic research, it has been proposed that the implementation of role-based access control should be considered in order to restrict information access depending on the roles that users play. Furthermore, it is necessary to conduct regular security training and awareness programs that are tailored to the specific circumstances of multinational teams. Integration of security practices across the development lifecycle ensures that security concerns are taken into consideration from the very beginning of the process, despite the fact that behavioral monitoring tools might be beneficial in discovering potential vulnerabilities. By applying psychological evaluation and employee support programs, it is feasible to address the motivations behind insider acts and limit any threats. This is possible while also addressing the potential dangers. There are a number of successful countermeasures against insider threats that are highlighted in the practices of the industry. These include rigorous onboarding and offboarding procedures, the promotion of a culture that values security, and the deployment of multi-factor authentication and encryption. The deployment of frequent audits, access assessments, and a grasp of the legal and compliance standards that apply across international borders are all components that contribute to the enhancement of security operations. When it comes to writing secure code in global software development, there are a number of different approaches that can be taken. However, the most common ones include adopting international security standards and frameworks, such as OWASP Top 10 and ISO/IEC 27001, and integrating security into the development lifecycle through the use of DevSecOps practices. Through the use of secure coding principles and education, as well as through the utilization of peer review and pair programming, it is possible to obtain an improvement in the security of the code..



Survey Question	Response Summary
Familiarity with Security Frameworks	80% familiar, 20% not familiar
Implementation of Cybersecurity Frameworks	75% yes, 25% no
Cybersecurity Frameworks Used	40% NIST, 30% ISO/IEC 27001, 15% OWASP, 15% Other
Reasons for Framework Adoption	50% compliance, 30% best practice, 20% client requirement
Challenges in Implementing Cybersecurity Strategies	40% resource constraints, 30% lack of expertise, 20% cultural differences, 10% other
Application of Techniques in GSD	60% apply existing, 40% adapt or use other techniques
Effectiveness of Current Frameworks	30% very effective, 50% moderately effective, 20% slightly effective
Employee Training and Awareness	70% have training programs, 30% do not have training programs
Suggestions for Improvement	40% more comprehensive guidelines, 30% easier implementation, 20% better training resources, 10% other

Figure 13 Survey Responses

On the other hand, challenges arise as a consequence of the fact that various regions implement security regulations in a variety of different methods. Additionally, for the purpose of communication and collaboration, it is necessary for security teams and developers to work together. The utilization of automated security technologies, the performance of periodic audits, and the execution of compliance checks are all common practices that are utilized in order to identify vulnerabilities and ensure that security standards and regulations are followed to. It is helpful to develop a collaborative approach to security, in which security is considered as a shared responsibility, in order to successfully detect and mitigate security threats. This is because security is best understood as a shared obligation.

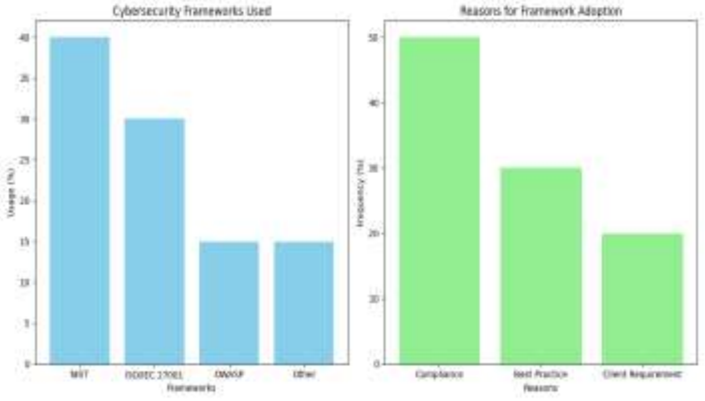


Figure 14 Framework to Adopt

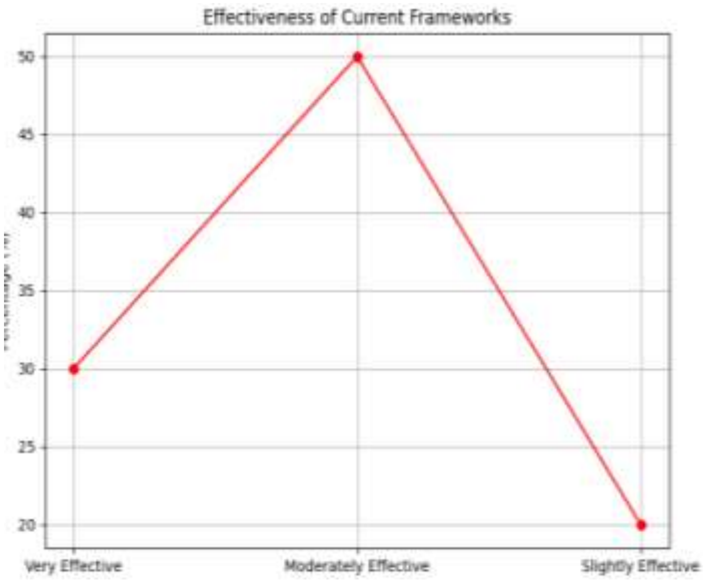


Figure 15 Effectiveness of Current Frameworks

V. Conclusion

As part of this research, an adventurous journey into the world of cybersecurity was undertaken within the context of global software development (GSD). We delved into the cybersecurity dangers, issues, and practices that are prevalent in GSD contexts by conducting an in-depth survey in addition to conducting a full literature analysis. Our findings provide light on the multidimensional nature of cybersecurity in GSD and underscore the critical role that international framework such as NIST, ISO/IEC 27001, and OWASP play in directing safe development practices across teams that are geographically scattered. Among the firms that were examined, we found that there was a considerable awareness and implementation of these frameworks. Compliance, best practices, and client requirements were the primary drivers of their adoption. In the course of our research, we discovered a number of obstacles that call attention to the complexities involved in executing successful cybersecurity policies in a global setting. These challenges range from a lack of experience and resource limits to the difficulties associated with managing cultural differences. Despite these hurdles, the companies that were polled indicated a commitment to improving security by providing regular training to their employees, implementing automated security technologies, and cultivating a culture of security awareness. According to the findings of our investigation into the efficiency of existing cybersecurity frameworks, practitioners have a modest level of trust in their capacity to protect against cyber-attacks. Although there is potential for improvement, particularly in terms of making these frameworks more comprehensive and easier to execute across a variety of legal and cultural settings, there is still room for progress.

VI. Future Work

Due to the ever-changing nature of both the software development and cybersecurity landscapes, there are multiple potential directions for research in the future. To begin, there is a requirement for the creation of cybersecurity frameworks that are capable of effectively combating emerging threats, particularly in contexts that are characterized by globally distributed development (GSD). Research efforts could be directed toward the development of frameworks of the future generation that are responsive to the specific requirements of GSD. In the second place, it is of the utmost importance to comprehend the influence that cultural and regulatory differences have on cybersecurity practices. Through in-depth research, it is possible to investigate several methods for overcoming these obstacles, such as communication across cultural boundaries and compliance mechanisms that are relevant everywhere. Additionally, as new technologies such as blockchain and artificial intelligence (AI) are included into GSD, research should be conducted to evaluate the potential of these technologies to improve security and to identify any risks that may be connected with them. In addition, conducting thorough case studies and longitudinal research on firms who are engaged in GSD would provide insights into the practical obstacles and accomplishments that are associated with the implementation of cybersecurity policies over time. In conclusion, given the significance of staff training and awareness, which was brought to light in our survey, it is possible that future study might analyze the efficacy of cybersecurity training programs. The identification of best practices for the purpose of cultivating a robust security culture among teams that are geographically dispersed is an absolute necessity.

REFERENCES

[1] Gatzlaff, K.M. and McCullough, K.A., 2010. The effect of data breaches on shareholder wealth. Risk Management and Insurance Review, 13(1), pp.61-83.

[2] Graff, M. and Van Wyk, K.R., 2003. Secure coding: principles and practices. " O'Reilly Media, Inc.".

[3] Schiavenato, M. and Chu, F., 2021. PICO: What it is and what it is not. Nurse education in practice, 56, p.103194.

[4] Zhao, L.L. and Zhang, X., 2024. Security Review of Foreign Investment in China: A Bibliometric Analysis and Systematic Literature Commentary. Государственное управление. Электронный вестник, (102), pp.54-76.

[5] Holvikivi, J., 2024. A Risk Evaluation Framework for Digitalization of Education with an Emphasis on Africa. Journal of Information Processing, 32, pp.77-83.

[6] Khalid, H.Y.I. and Aldabagh, N.B.I., 2024. A Survey on the Latest Intrusion Detection Datasets for Software Defined Networking Environments. Engineering, Technology & Applied Science Research, 14(2), pp.13190-13200.

[7] Coombs, K., 2024. Perspectives of Cybersecurity Risks in DNA Data Storage Environment (Doctoral dissertation, Capella University).

[8] Huang, Z., Biczók, G. and Liu, M., 2024. Incentivizing Secure Software Development: The Role of Liability (Waiver) and Audit. arXiv preprint arXiv:2401.08476.

[9] 9] Alzoubi, H., Alshurideh, M., Kurdi, B., Akour, I., & Aziz, R. (2022). Does BLE technology contribute towards improving marketing strategies, customers' satisfaction and loyalty? The role of open innovation. International Journal of Data and Network Science, 6(2), 449-460.

[10] Nyangaresi, V. O., & Morsy, M. A. (2021, September). Towards privacy preservation in internet of drones. In 2021

[11] IEEE 6th International Forum on Research and Technology for Society and Industry (RTSI) (pp. 306-311). IEEE.

[12] Govindaras, B., Wern, T. S., Kaur, S., Haslin, I. A., & Ramasamy, R. K. (2023). Sustainable environment to prevent burnout and attrition in project management. Sustainability, 15(3), 2364.

[13] Pillai, N. S. R., & Hemamalini, R. R. (2022). Hybrid user acceptance test procedure to improve the software quality. Int. Arab J. Inf. Technol., 19(6), 956-964.

[14] Yilmaz, M., & O'Connor, R. V. (2016). A Scrumban integrated gamification approach to guide software process improvement: a Turkish case study. Tehnički vjesnik, 23(1), 237-245.

[15] García-Mireles, G. A., Moraga, M. Á., García, F., & Piattini, M. (2015). Approaches to promote product quality within software process improvement initiatives: a mapping study. Journal of Systems and Software, 103, 150-166.

[16] Abduljabbar, Z. A., Omollo Nyangaresi, V., Al Sibahee, M. A., Ghrabat, M. J. J., Ma, J., Qays Abduljaleel, I., & Aldarwish, A. J. (2022). Session-Dependent Token-Based Payload Enciphering Scheme for Integrity Enhancements in Wireless Networks. Journal of Sensor and Actuator Networks, 11(3), 55.

[17] Lin, C., Wittmer, J. L., & Luo, X. R. (2022). Cultivating proactive information security behavior and individual creativity: The role of human relations culture and IT use governance. Information & Management, 59(6), 103650.

[18] Wolf, F., Kuber, R., & Aviv, A. J. (2018). An empirical study examining the perceptions and behaviours of security-conscious users of mobile authentication. Behaviour & Information Technology, 37(4), 320-334.

[19] Krehbiel, T. C., Salzarulo, P. A., Cosmah, M. L., Forren, J., Gannod, G., Havelka, D., ... & Merhout, J. (2017). Agile Manifesto for Teaching and Learning. Journal of Effective Teaching, 17(2), 90-111.

[20] Chauhan, A. S., Nepal, B., Soni, G., & Rathore, A. P. S. (2018). Examining the state of risk management research in new product development process. Engineering Management Journal, 30(2), 85-97.

[21] Nyangaresi, V. O. (2023). Privacy preserving three-factor authentication protocol for secure message forwarding in wireless body area networks. Ad Hoc Networks, 142, 103117.

[22] Bharadwaj, M. S. (2024). Regulatory Insights in Digital Health. In Multi-Sector Analysis of the Digital Healthcare Industry (pp. 164-197). IGI Global.

[23] Yamagata, T., & Santos-Rodriguez, R. (2024). Safe and Robust Reinforcement-Learning: Principles and Practice. arXiv preprint arXiv:2403.18539.

[24] Richards, J. P., Gallarno, G. E., & Buchanan, R. Smart Base Installations: Applying Systems Engineering Techniques to the Agile Development of Multidisciplinary Systems of Systems Projects. In The Proceedings of the 2023 Conference on Systems Engineering Research: Systems Engineering Towards a Smart and Sustainable World (p. 373). Springer Nature.

[25] Janak, J. (2024). Towards Self-Managing Networked Cyber-Physical Systems. Columbia University.

[26] Vyas, B. (2023). Security Challenges and Solutions in Java Application Development. Eduzone: International Peer Reviewed/Refereed Multidisciplinary Journal, 12(2), 268-275.

[27] J Bena, N. (2024). NON-FUNCTIONAL CERTIFICATION OF MODERN DISTRIBUTED SYSTEMS.

[28] Vyas, B. (2023). Security Challenges and Solutions in Java Application Development. Eduzone: International Peer Reviewed/Refereed Multidisciplinary Journal, 12(2), 268-275.

[29] Abushmmala, F. F., & AbuSamra, A. A. (2023). Blockchain-Based Secure Smart Health IoT solution Using RBAC Architecture. Journal of Engineering Research and Technology, 10(2).

[30] Das, S., Krishnamurthy, B., Das, R. R., & Shiva, S. G. (2024, January). State of the art: Security Testing of Machine Learning Development Systems. In 2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0534-0540). IEEE.

[31] Möller, D. P. (2023). NIST Cybersecurity Framework and MITRE Cybersecurity Criteria. In Guide to Cybersecurity in Digital Transformation: Trends, Methods, Technologies, Applications and Best Practices (pp. 231-271). Cham: Springer Nature Switzerland.

[32] Valdés-Rodríguez, Y., Hochstetter-Diez, J., Diéguez-Rebolledo, M., Bustamante-Mora, A., & Cadena-Martínez, R. (2024). Analysis of Strategies for the Integration of Security Practices in Agile Software Development: A Sustainable SME Approach. IEEE Access.

[33] Zimmermann, M., Staicu, C. A., Tenny, C., & Pradel, M. (2019, August). Small World with High Risks: A Study of Security Threats in the npm Ecosystem. In USENIX security symposium (Vol. 17).

[34] Bertino, E., & Islam, N. (2017). Botnets and internet of things security. Computer, 50(2), 76-79.

[35] Abduljabbar, Z. A., Abduljaleel, I. Q., Ma, J., Al Sibahee, M. A., Nyangaresi, V. O., Honi, D. G., ... & Jiao, X. (2022). Provably secure and fast color image encryption algorithm based on s-boxes and hyperchaotic map. IEEE Access, 10, 26257-26270.

[36] Appiah, V., Nti, I. K., & Nyarko-Boateng, O. (2017). Investigating websites and web application vulnerabilities: Webmaster's perspective. Int. J. Appl. Inf. Syst, 12(3), 1015.

[37] Khan, A., Ahmad, A., Ahmed, M., Sessa, J., & Anisetti, M. (2022). Authorization schemes for internet of things: requirements, weaknesses, future challenges and trends. Complex & Intelligent Systems, 8(5), 3919-3941.

[38] Stasinopoulos, A., Ntantogian, C., & Xenakis, C. (2019). Commix: automating evaluation and exploitation of command injection vulnerabilities in web applications. International Journal of Information Security, 18, 49-72

[39] Nyangaresi, V. O. (2022, July). Provably Secure Pseudonyms based Authentication Protocol for Wearable Ubiquitous Computing Environment. In 2022 International Conference on Inventive Computation Technologies (ICICT) (pp. 1-6). IEEE.

[40] Zhang, M., Zhang, Y., Zhang, L., Liu, C., & Khurshid, S. (2018, September). DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (pp. 132-142)

[41] Jiang, X., Lora, M., & Chattopadhyay, S. (2020). An experimental analysis of security vulnerabilities in industrial IoT devices. ACM Transactions on Internet Technology (TOIT), 20(2), 1-24.

[42] Salahdine, F., & Kaabouch, N. (2019). Social engineering attacks: A survey. Future Internet, 11(4), 89.

[43] Treacy, C., Loane, J., & McCaffery, F. (2020). Developer Driven Framework for Security and Privacy in the IoMT. In ICSOFT (pp. 443-451).

[44] Núñez, J. C. S., Lindo, A. C., & Rodríguez, P. G. (2020). A preventive secure software development model for a software factory: a case study. IEEE Access, 8, 77653-77665.

[45] Bezerra, C. M. M., Sampaio, S. C., & Marinho, M. L. (2020, August). Secure agile software development: policies and practices for agile teams. In International Conference on the Quality of Information and Communications Technology (pp. 343-357). Cham: Springer International Publishing.

[46] Subedi, B., Alsadoon, A., Prasad, P. W. C., & Elchouemi, A. (2016, September). Secure paradigm for web application development. In 2016 15th RoEduNet Conference: Networking in Education and Research (pp. 1-6). IEEE.

[47] Humayun, M., Jhanjhi, N., Almufareh, M. F., & Khalil, M. I. (2022). Security threat and vulnerability assessment and measurement in secure software development. Comput. Mater. Contin, 71, 5039-5059.

[48] de Vicente Mohino, J., Bermejo Higuera, J., Bermejo Higuera, J. R., & Sicilia Montalvo, J. A. (2019). The application of a new secure software development life cycle (S-SDLC) with agile methodologies. Electronics, 8(11), 1218.

[49] Zanevych, O. (2024). ADVANCING WEB DEVELOPMENT: A COMPARATIVE ANALYSIS OF MODERN FRAMEWORKS FOR REST AND GRAPHQL BACK-END SERVICES. Grail of Science, (37), 216-228.

[50] Komolafe, A. M., Aderotoye, I. A., Abiona, O. O., Adewusi, A. O., Obijuru, A., Modupe, O. T., & Oyeniran, O. C. (2024). HARNESSING BUSINESS ANALYTICS FOR GAINING COMPETITIVE ADVANTAGE IN EMERGING MARKETS: A SYSTEMATIC REVIEW OF APPROACHES AND OUTCOMES. International Journal of Management & Entrepreneurship Research, 6(3), 838-862.

[51] Pitre, V., Joshi, A., Saladi, S., & Das, S. (2024). Enhancing Cybersecurity Ratings Using Artificial Intelligence and DevOps Technologies. Applying Artificial Intelligence in Cybersecurity Analytics and Cyber Threat Detection, 63-86.

[52] Kaster, R. M. (2024). Automotive Software Attestation: Self, Remote, and Peer-Building Trust in Autonomous Driving Safety Systems.

[53] Kumar, S., Arzaghi, E., Baalisampang, T., Abaei, M. M., Garaniya, V., & Abbassi, R. (2024). A risk-based multi-criteria decision-making framework for offshore green hydrogen system developments: Pathways for utilizing existing and new infrastructure. Sustainable Production and Consumption.

[54] Nkambule, M., & van Vuuren, J. J. (2024, March). Integrating Enterprise Architecture into Cybersecurity Risk Management in Higher Education. In International Conference on Cyber Warfare and Security (Vol. 19, No. 1, pp. 501-510).

[55] Saar, G., & Dagada, R. (2024, March). Building Cybersecurity Capacities in Zambia's Business Sector: Guideline for SMEs. In International Conference on Cyber Warfare and Security (Vol. 19, No. 1, pp. 317-326).

[56] Fang, L., & Smith, J. (2024). Addressing Privacy and Security Concerns in AI-Driven Software Development Life Cycle (No. 12600). EasyChair. https://owaspsamm.org/

[57] Synopsys, "BSIMM Maturity Model," Available: https://www.synopsys.com/software-integrity/software-security-services/bsimm-maturity-model.html. [Accessed: April 11, 2024]. [59] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.

[58] Khan, R. A., & Khan, S. U. (2018, May). A preliminary structure of software security assurance model. In Proceedings of the 13th International Conference on Global Software Engineering (pp. 137-140).

[59] Karim NS, Albuolayan A, Saba T, Rehman A. The practice of secure software development in SDLC: an investigation through existing model and a case study. Security and Communication Networks. 2016 Dec, 9(18):5333-45.

[60] Sabale RG, Dani AR. Comparative study of prototype model for software engineering with system development life cycle. IOSR Journal of Engineering. 2012 Jul, 2(7):21-4.

[61] Assal, H., & Chiasson, S. (2018). Security in the software development lifecycle. In Fourteenth symposium on usable privacy and security (SOUPS 2018) (pp. 281-296).

[62] Eian IC, Yong LK, Li MY, Hasmaddi NA. Integration of Security Modules in Software Development Lifecycle Phases. arXiv preprint arXiv:2012.05540. 2020 Dec 10.

[63] Fujdiak R, Mlynek P, Mrnustik P, Barabas M, Blazek P, Borcik F, Misurec J. Managing the secure software development. In2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS) 2019 Jun 24 (pp. 1-4). IEEE.

[64] Rashed AN, Ahammad SH, Daher MG, Sorathiya V, Siddique A, Asaduzzaman S, Rehana H, Dutta N, Patel SK, Nyangaresi VO, Jibon RH. Spatial single mode laser source interaction with measured pulse based parabolic index multimode fiber. Journal of Optical Communications. 2022 Jun 21.

[65] Rashed AN, Ahammad SH, Daher MG, Sorathiya V, Siddique A, Asaduzzaman S, Rehana H, Dutta N, Patel SK, Nyangaresi VO, Jibon RH. Spatial single mode laser source interaction with measured pulse based parabolic index multimode fiber. Journal of Optical Communications. 2022 Jun 21.

[66] Syed R, Rahafrooz M, Keisler JM. What it takes to get retweeted: An analysis of software vulnerability messages. Computers in Human Behavior. 2018 Mar 1, 80:207-15.

[67] Giffin JT, Christodorescu M, Kruger L. Strengthening software self-checksumming via self-modifying code. In21st Annual Computer Security Applications Conference (ACSAC'05) 2005 Dec 5 (pp. 10-pp). IEEE.

[68] Moe NB, Šmite D. Understanding a lack of trust in Global Software Teams: a multiple-case study. Software Process: Improvement and Practice. 2008 May, 13(3):217-31.

[69] Nyangaresi VO, Mohammad Z. Privacy preservation protocol for smart grid networks. In2021 International Telecommunications Conference (ITC-Egypt) 2021 Jul 13 (pp. 1-4). IEEE.

[70] Maher ZA, Shaikh H, Khan MS, Arbaaeen A, Shah A. Factors affecting secure software development practices among developers-An investigation. In2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS) 2018 Nov 22 (pp. 1-6). IEEE.

[71] Rodríguez GE, Torres JG, Flores P, Benavides DE. Cross-site scripting (XSS) attacks and mitigation: A survey. Computer Networks. 2020 Jan 15, 166:106960.

[72] Mohammed NM, Niazi M, Alshayeb M, Mahmood S. Exploring software security approaches in software development lifecycle: A systematic mapping study. Computer Standards & Interfaces. 2017 Feb 1, 50:107-15.

[73] Silva P, Noël R, Gallego M, Matalonga S, Astudillo H. Software Development Initiatives to Identify and Mitigate Security Threats: A Systematic Mapping. InCIbSE 2016 Apr 27 (pp. 257-270).

[74] Guinea AS, Nain G, Le Traon Y. A systematic review on the engineering of software for ubiquitous systems. Journal of Systems and Software. 2016 Aug 1, 118:251-76.

[75] Meshram C, Alsanad A, Tembhurne JV, Shende SW, Kalare KW, Meshram SG, Akbar MA, Gumaei A. A provably secure lightweight subtree-based short signature scheme with fuzzy user data sharing for human-centered IoT. IEEE Access. 2020 Dec 21, 9:3649-59.

[76] Rafi S, Yu W, Akbar MA, Alsanad A, Gumaei A. Prioritization based taxonomy of DevOps security challenges using PROMETHEE. IEEE Access. 2020 Jun 1, 8:105426-46.

[77] Hall A, Chapman R. Correctness by construction: Developing a commercial secure system. IEEE software. 2002 Aug 7, 19(1):18-25.

[78] Nyangaresi VO, Moundounga AR. Secure data exchange scheme for smart grids. In2021 IEEE 6th International Forum on Research and Technology for Society and Industry (RTSI) 2021 Sep 6 (pp. 312-316). IEEE.

[79] Potter B, McGraw G. Software security testing. IEEE Security & Privacy. 2004 Oct 8, 2(5):81-5.

[80] Verdon D, McGraw G. Risk analysis in software design. IEEE Security & Privacy. 2004 Oct 4, 2(4):79-84.

[81] Lipner S. The trustworthy computing security development lifecycle. In20th Annual Computer Security

[82] Gupta S, Faisal M, Husain M. Secure software development process for embedded systems control. Int. J. Eng. Sci. Emerg. Technol. 2012 Dec 1, 4:133-43.

[83] Alsamhi SH, Shvetsov AV, Kumar S, Shvetsova SV, Alhartomi MA, Hawbani A, Rajput NS, Srivastava S, Saif A, Nyangaresi VO. UAV computing-assisted search and rescue mission framework for disaster and harsh environment mitigation. Drones. 2022 Jun 22, 6(7):154.

[84] Manico J. OWASP. Proc. Appl. Secur. Verification Standard. 2016:1-70.

[85] Gregoire J, Buyens K, De Win B, Scandariato R, Joosen W. On the secure software development process: CLASP and SDL compared. InThird International Workshop on Software Engineering for Secure Systems (SESS'07: ICSE Workshops 2007) 2007 May 20 (pp. 1-1). IEEE.

[86] Li W, Chiueh TC. Automated format string attack prevention for win32/x86 binaries. InTwenty-Third Annual Computer Security Applications Conference (ACSAC 2007) 2007 Dec 10 (pp. 398-409). IEEE.

[87] Sahu DR, Tomar DS. Analysis of web application code vulnerabilities using secure coding standards. Arabian Journal for Science and Engineering. 2017 Feb, 42:885-95.

[88] Yang J, Ryu D, Baik J. Improving vulnerability prediction accuracy with secure coding standard violation measures. In2016 International Conference on Big Data and Smart Computing (BigComp) 2016 Jan 18 (pp. 115- 122). IEEE.

[89] Gasiba TE, Lechner U, Pinto-Albuquerque M, Mendez D. Is secure coding education in the industry needed? An investigation through a large scale survey. In2021 IEEE/ACM 43rd International Conference on Software

[90] Engineering: Software Engineering Education and Training (ICSE-SEET) 2021 May 25 (pp. 241-252). IEEE.

[91] Nyangaresi VO, Morsy MA. Towards privacy preservation in internet of drones. In2021 IEEE 6th International Forum on Research and Technology for Society and Industry (RTSI) 2021 Sep 6 (pp. 306-311). IEEE.

[92] Peine H. Rules of thumb for developing secure software: Analyzing and consolidating two proposed sets of rules. In 2008 Third International Conference on Availability, Reliability and Security 2008 Mar 4 (pp. 1204-1209). IEEE.

[93] Mead NR, Stehney T. Security quality requirements engineering (SQUARE) methodology. ACM SIGSOFT Software Engineering Notes. 2005 May 15, 30(4):1-7.

[94] Flechais I, Mascolo C, Sasse MA. Integrating security and usability into the requirements and design process. International Journal of Electronic Security and Digital Forensics. 2007 Jan 1, 1(1):12-26.

[95] Collins B. Big data and health economics: strengths, weaknesses, opportunities and threats. Pharmacoeconomics. 2016 Feb, 34(2):101-6.

[96] Sodiya AS, Onashoga SA, Ajayĩ OB. Towards building secure software systems. Issues in Informing Science & Information Technology. 2006 Jan 1, 3.

[97] Hentea M, Dhillon HS, Dhillon M. Towards changes in information security education. Journal of Information

[98] Nyangaresi VO, Alsamhi SH. Towards secure traffic signaling in smart grids. In2021 3rd Global Power, Energy and Communication Conference (GPECOM) 2021 Oct 5 (pp. 196-201). IEEE. Technology Education: Research. 2006 Jan 1, 5(1):221-33.

[99] Aslanyan Z, Nielson F, Parker D. Quantitative verification and synthesis of attack-defence scenarios. In2016 IEEE 29th Computer Security Foundations Symposium (CSF) 2016 Jun 27 (pp. 105-119). IEEE.

[100] Nguyen TH, Grundy J, Almorsy M. Integrating goal-oriented and use case-based requirements engineering: The missing link. In2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS) 2015 Sep 30 (pp. 328-337). IEEE.

[101] Vassilakis VG, Mouratidis H, Panaousis E, Moscholios ID, Logothetis MD. Security requirements modelling for virtualized 5G small cell networks. In2017 24th International Conference on Telecommunications (ICT) 2017 May 3 (pp. 1-5). IEEE.

[102] Fernandez-Buglioni E. Security patterns in practice: designing secure architectures using software patterns. John Wiley & Sons, 2013 Jun 25.

[103] Xu D, Nygard KE. Threat-driven modeling and verification of secure software using aspect-oriented Petri nets. IEEE transactions on software engineering. 2006 May 8, 32(4):265-78.

[104] Cotroneo D, Pietrantuono R, Russo S, Trivedi K. How do bugs surface? A comprehensive study on the characteristics of software bugs manifestation. Journal of Systems and Software. 2016 Mar 1, 113:27-43.

[105] Mellado D, Fernández-Medina E, Piattini M. A common criteria based security requirements engineering process for the development of secure information systems. Computer standards & interfaces. 2007 Feb 1, 29(2):244-53.

[106] Fabian B, Gürses S, Heisel M, Santen T, Schmidt H. A comparison of security requirements engineering methods. Requirements engineering. 2010 Mar, 15:7-40.

[107] Haley C, Laney R, Moffett J, Nuseibeh B. Security requirements engineering: A framework for representation and analysis. IEEE Transactions on Software Engineering. 2008 Jan 31, 34(1):133-53.

[108] Nyangaresi VO, Abd-Elnaby M, Eid MM, Nabih Zaki Rashed A. Trusted authority based session key agreement and authentication algorithm for smart grid networks. Transactions on Emerging Telecommunications Technologies. 2022 Sep, 33(9):e4528.

[109] Humphreys T. State-of-the-art information security management systems with ISO/IEC 27001: 2005. ISO Management Systems. 2006 Jan, 6(1):15-8.

[110] Mataracioglu T, Ozkan S. Analysis of the user acceptance for implementing ISO/IEC 27001: 2005 in turkish public organizations. arXiv preprint arXiv:1103.0405. 2011 Mar 2.

[111] Costin D, Militaru C. Asset Management Towards ISO/IEC 27001: 2005 Accreditation of an Information Security Management System. Revista De Management Comparat International/Review of International Comparative Management. 2011, 12(6):245-50.

[112] Soomro ZA, Shah MH, Ahmed J. Information security management needs more holistic approach: A literature review. International journal of information management. 2016 Apr 1, 36(2):215-25.

[113] Zhao Y, Gu P, Zhu F, Liu T, Shen R. Security control scheme for cyber-physical system with a complex network in physical layer against false data injection attacks. Applied Mathematics and Computation. 2023 Jun 15, 447:127908.

[114] Nyangaresi VO, Mohammad Z. Session Key Agreement Protocol for Secure D2D Communication. InThe Fifth International Conference on Safety and Security with IoT: SaSeIoT 2021 2022 Jun 12 (pp. 81-99). Cham: Springer International Publishing.

[115] Thompson C, Shelton M, Stark E, Walker M, Schechter E, Felt AP. The web's identity crisis: understanding the effectiveness of website identity indicators. In28th {USENIX} Security Symposium ({USENIX} Security 19) 2019 (pp. 1715-1732).

[116] Khari M, Kumar P. Embedding security in software development life cycle (SDLC). In2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom) 2016 Mar 16 (pp. 2182-2186). IEEE.

[117] Mufti Y, Niazi M, Alshayeb M, Mahmood S. A readiness model for security requirements engineering. IEEE Access. 2018 May 24, 6:28611-31.

[118] Younas M, Jawawi DN, Shah MA, Mustafa A, Awais M, Ishfaq MK, Wakil K. Elicitation of nonfunctional requirements in agile development using cloud computing environment. IEEE access. 2020 Aug 27, 8:209153-

[119] Niazi M, Saeed AM, Alshayeb M, Mahmood S, Zafar S. A maturity model for secure requirements engineering. Computers & Security. 2020 Aug 1, 95:101852.

[120] Karim NS, Albuolayan A, Saba T, Rehman A. The practice of secure software development in SDLC: an investigation through existing model and a case study. Security and Communication Networks. 2016 Dec, 9(18):5333-45.

[121] Hlaing SZ, Ochimizu K. An integrated cost-effective security requirement engineering process in SDLC using FRAM. In2018 International Conference on Computational Science and Computational Intelligence (CSCI) 2018 Dec 12 (pp. 852-857). IEEE.

[122] Salini P, Kanmani S. Survey and analysis on security requirements engineering. Computers & Electrical Engineering. 2012 Nov 1, 38(6):1785-97.

[123] Al-Shorafat WS. Security in software engineering requirement. In8th International Conference for Internet Technology and Secured Transactions (ICITST-2013) 2013 Dec 9 (pp. 666-673). IEEE.

[124] Preuveneers D, Berbers Y, Bhatti G. Best practices for software security: An overview. In2008 IEEE International Multitopic Conference 2008 Dec 23 (pp. 169-173). IEEE.

[125] Khan RA, Khan SU, Ilyas M, Idris MY. The state of the art on secure software engineering: A systematic mapping study. Proceedings of the Evaluation and Assessment in Software Engineering. 2020 Apr 15:487-92.

[126] Khan RA, Khan SU, Khan HU, Ilyas M. Systematic mapping study on security approaches in secure software engineering. IEEE Access. 2021 Jan 18, 9:19139-60.

[127] Maheshwari V, Prasanna M. Integrating risk assessment and threat modeling within SDLC process. In 2016 international conference on inventive computation technologies (ICICT) 2016 Aug 26 (Vol. 1, pp. 1-5). IEEE.

[128] Mohammad Z, Nyangaresi V, Abusukhon A. On the Security of the Standardized MQV Protocol and Its Based Evolution Protocols. In2021 International Conference on Information Technology (ICIT) 2021 Jul 14 (pp. 320- 325). IEEE.

[129] Hudaib A, AlShraideh M, Surakhi O, Khanafseh M. A survey on design methods for secure software development. Int. J. Comput. Technol. 2017 Dec 10, 16(7).

[130] Omollo VN, Musyoki S. Blue bugging Java Enabled Phones via Bluetooth Protocol Stack Flaws. International Journal of Computer and Communication System Engineering. 2015 Jun 9, 2 (4):608-613.

[131] Van den Berghe A, Scandariato R, Yskout K, Joosen W. Design nota

[132] Banowosari LY, Gifari BA. System analysis and design using secure software development life cycle based on ISO 31000 and STRIDE. Case study mutiara ban workshop. In2019 Fourth International Conference on Informatics and Computing (ICIC) 2019 Oct 16 (pp. 1-6). IEEE.

[133] Apvrille A, Pourzandi M. Secure software development by example. IEEE Security & Privacy. 2005 Aug 8, 3(4):10- 7.

[134] Doan T, Demurjian S, Ting TC, Ketterl A. MAC and UML for secure software design. InProceedings of the 2004 ACM workshop on Formal Methods in Security Engineering 2004 Oct 29 (pp. 75-85).

[135] ben Othmane L, Angin P, Weffers H, Bhargava B. Extending the agile development process to develop acceptably secure software. IEEE Transactions on dependable and secure computing. 2014 Jan 9, 11(6):497-509.

[136] Omollo VN, Musyoki S. Blue bugging Java Enabled Phones via Bluetooth Protocol Stack Flaws. International Journal of Computer and Communication System Engineering. 2015 Jun 9, 2 (4):608-613.

[137] Tang J. Towards automation in software test life cycle based on multi-agent. In2010 International Conference on Computational Intelligence and Software Engineering 2010 Dec 10 (pp. 1-4). IEEE.

[138] Mumtaz H, Alshayeb M, Mahmood S, Niazi M. An empirical study to improve software security through the application of code refactoring. Information and Software Technology. 2018 Apr 1, 96:112-25.

[139] Núñez JC, Lindo AC, Rodríguez PG. A preventive secure software development model for a software factory: a case study. IEEE Access. 2020 Apr 21, 8:77653-65.

[140] Guinea AS, Nain G, Le Traon Y. A systematic review on the engineering of software for ubiquitous systems. Journal of Systems and Software. 2016 Aug 1, 118:251-76

[141] Venson E, Guo X, Yan Z, Boehm B. Costing secure software development: A systematic mapping study. InProceedings of the 14th International Conference on Availability, Reliability and Security 2019 Aug 26 (pp. 1-11).

[142] Sodanil M, Quirchmayr G, Porrawatpreyakorn N, Tjoa AM. A knowledge transfer framework for secure coding practices. In2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE) 2015 Jul 22 (pp. 120-125). IEEE.

[143] Venson E, Alfayez R, Gomes MM, Figueiredo RM, Boehm B. The impact of software security practices on development effort: An initial survey. In2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) 2019 Sep 19 (pp. 1-12). IEEE.

[144] Nyangaresi VO. A Formally Verified Authentication Scheme for mmWave Heterogeneous Networks. Inthe 6th International Conference on Combinatorics, Cryptography, Computer Science and Computation (605-612) 2021

[145] Salini P, Kanmani S. Effectiveness and performance analysis of model-oriented security requirements engineering to elicit security requirements: a systematic solution for developing secure software systems. International Journal of Information Security. 2016 Jun, 15:319-34.

[146] Musa Shuaibu B, Md Norwawi N, Selamat MH, Al-Alwani A. Systematic review of web application security development model. Artificial Intelligence Review. 2015 Feb, 43:259-76.

[147] Grieco G, Grinblat GL, Uzal L, Rawat S, Feist J, Mounier L. Toward large-scale vulnerability discovery using machine learning. InProceedings of the Sixth ACM Conference on Data and Application Security and Privacy 2016 Mar 9 (pp. 85-96)

[148] Cui L, Cui J, Hao Z, Li L, Ding Z, Liu Y. An empirical study of vulnerability discovery methods over the past ten years. Computers & Security. 2022 Sep 1, 120:102817.

[149] De Win B, Scandariato R, Buyens K, Grégoire J, Joosen W. On the secure software development process: CLASP, SDL and Touchpoints compared. Information and software technology. 2009 Jul 1, 51(7):1152-71.

[150] Siddiqui ST. Significance of security metrics in secure software development. Significance. 2017 Aug, 12(6).

[151] Chess B, Arkin B. Software security in practice. IEEE Security & Privacy. 2011 Mar 28, 9(2):89-92.

[152] Al-Amin S, Ajmeri N, Du H, Berglund EZ, Singh MP. Toward effective adoption of secure software development practices. Simulation Modelling Practice and Theory. 2018 Jun 1, 85:33-46.

[153] Souag A, Salinesi C, Wattiau I, Mouratidis H. Using security and domain ontologies for security requirements analysis. In2013 IEEE 37th Annual Computer Software and Applications Conference Workshops 2013 Jul 22 (pp. 101-107). IEEE.

[154] Nyangaresi VO, Abduljabbar ZA, Ma J, Al Sibahee MA. Verifiable Security and Privacy Provisioning Protocol for High Reliability in Smart Healthcare Communication Environment. In2022 4th Global Power, Energy and Communication Conference (GPECOM) 2022 Jun 14 (pp. 569-574). IEEE.

[155] Mayvan BB, Rasoolzadegan A, Yazdi ZG. The state of the art on design patterns: A systematic mapping of the literature. Journal of Systems and Software. 2017 Mar 1, 125:93-118.

[156] Kadam SP, Joshi S. Secure by design approach to improve security of object oriented software. In2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom) 2015 Mar 11 (pp. 24- 30). IEEE.

[157] Kumar R, Khan SA, Khan RA. Analytical network process for software security: a design perspective. CSI transactions on ICT. 2016 Dec, 4:255-8.

[158] Janakiraman S, Thenmozhi K, Rayappan JB, Amirtharajan R. Lightweight chaotic image encryption algorithm for real-time embedded system: Implementation and analysis on 32-bit microcontroller. Microprocessors and Microsystems. 2018 Feb 1, 56:1-2.

[159] Zaki Rashed AN, Ahammad SH, Daher MG, Sorathiya V, Siddique A, Asaduzzaman S, Rehana H, Dutta N, Patel SK, Nyangaresi VO, Jibon RH. Signal propagation parameters estimation through designed multi layer fibre with

[160] higher dominant modes using OptiFibre simulation. Journal of Optical Communications. 2022 Jun 23(0).

[161] Clegg BS, Rojas JM, Fraser G. Teaching software testing concepts using a mutation testing game. In 2017 IEEE/ACM 39th International Conference

on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET) 2017 May 20 (pp. 33-36). IEEE.

[162] Xie J, Lipford HR, Chu B. Why do programmers make security errors?. In2011 IEEE symposium on visual languages and human-centric computing (VL/HCC) 2011 Sep 18 (pp. 161-164). IEEE.

[163] Nyangaresi VO, Ma J, Al Sibahee MA, Abduljabbar ZA. Packet Replays Prevention Protocol for Secure B5G Networks. InProceedings of Seventh International Congress on Information and Communication Technology: ICICT 2022, London, Volume 2 2022 Jul 27 (pp. 507-522). Singapore: Springer Nature Singapore.

[164] Felderer M, Fourneret E. A systematic classification of security regression testing approaches. International Journal on Software Tools for Technology Transfer. 2015 Jun, 17:305-19.

[165] W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

[166] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.

[167] B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.

[168] E. H. Miller, "A note on reflector arrays (Periodical style—Accepted for publication)," *IEEE Trans. Antennas Propagat.*, to be published.

[169] J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.

[170] Bint-E-Asim, H., Iqbal, S., Danish, A. S., Shahzad, A., Huzaifa, M., &Khan, Z. (n.d.). Exploring Interactive STEM in Online Education throughRobotic Kits for Playful Learning (Vol. 19). http://xisdxjxsu.asia

[171] Danish, A. S., Waheed, Z., Sajid, U., Warah, U., Muhammad, A., Khan, Y.,& Akram, H. (n.d.). Exploring Temporal Complexities: Time Constraints inAugmented Reality-Based Hybrid Pedagogies for Physics Energy Topic inSecondary Schools. http://xisdxjxsu.asia

[172] Danish, A. S., Khan, Z., Jahangir, F., Malik, A., Tariq, W., Muhammad, A.,& Khan, Y. (n.d.). Exploring the Effectiveness of Augmented Reality basedE-Learning Application on Learning Outcomes in Pakistan: A StudyUtilizing VARK Analysis and Hybrid Pedagogy. http://xisdxjxsu.asia

[173] Danish, A. S., Malik, A., Lashari, T., Javed, M. A., Lashari, T. A., Asim, H.B., Muhammad, A., & Khan, Y. (n.d.). Evaluating the User Experience ofan Augmented Reality E-Learning Application for the Chapter on Work andEnergy using the System Usability Scale.https://www.researchgate.net/publication/377020480

[174] Muhammad, A., Khan, Y., Danish, A. S., Haider, I., Batool, S., Javed, M.A., & Tariq, W. (n.d.). Enhancing Social Media Text Analysis:Investigating Advanced Preprocessing, Model Performance, andMultilingual Contexts. http://xisdxjxsu.asia

[175] Samad Danish, A., Noor, N., Hamid, Y., Ali Khan, H., Muneeb Asad, R., &Muhammad Yar Khan, A. (n.d.). Augmented Narratives: Unveiling the Efficacy of Storytelling in Augmented Reality Environments.http://xisdxjxsu.asia

[176] Faizan Hassan, M., Mehmood, U., Samad Danish, A., Khan, Z.,Muhammad Yar Khan, A., & Muneeb Asad, R. (n.d.). Harnessing Augmented Reality for Enhanced Computer Hardware Visualization for Learning. http://xisdxjxsu.asia

[177] Samad Danish, A., Warah, U., -UR-Rehman, O., Sajid, U., Adnan Javed,M., & Muhammad Yar Khan, A. (n.d.). Evaluating the Feasibility andResource Implications of an Augmented Reality-Based E-Learning Application: A Comprehensive Research Analysis. http://xisdxjxsu.asia

[178] -UR-Rehman, O., Samad Danish, A., Khan, J., Jalil, Z., & Ali, S. (2019).Implementation of Smart Aquarium System Supporting Remote Monitoringand Controlling of Functions using Internet of Things. In Journal of Multidisciplinary Approaches in Science. JMAS.

[179] Lashari, T., Danish, A. S., Lashari, S., Sajid, U., Lashari, T. A., Lashari, S.A., Khan, Z., & Saare, M. A. (n.d.). Impact of custom built videogame simulators on learning in Pakistan using Universal Design for Learning.http://xisdxjxsu.asia

AUTHORS

First Author – Agha Muhammad Yar Khan, Student, HITEC University Taxila.
Second Author – Abdul Samad Danish, Lecturer, HITEC University Taxila.
Third Author – Farwah Aizaz, Lecturer, HITEC University Taxila.
Fourth Author – Huzaifa Bilal, Student, HITEC University Taxila.
Fifth Author – Abdullah Shahrose, Lecturer, HITEC University Taxila.
Sixth Author – Rana Muneeb Asad, Student, HITEC University Taxila.
Seventh Author – Muhammad Talha Rafiq, Student, HITEC University Taxila.

Correspondence Author – Abdul Samad Danish,