INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 11, Issue 1 - V11I1-1304)
Available online at: https://www.ijariit.com

Adopting Secure Software Development Practices to Improve Financial Transactions in the Banking Sector

Rianat Abbas
rihanatoluwatosin@gmail.com
Baylor University, USA

Rasheed Afolabi
rasheed_afolabi1@baylor.edu
Baylor University, USA

Ifeoma Eleweke
elewekeifeoma@gmail.com
Westcliff University, USA

Adetomiwa Adesokan
tomiwasegun@gmail.com
University of Nevada, Reno, USA

Ahmed Akinsola
aakinsola@my.apsu.edu
Austin Peay State University, USA

Laticbe Elijah
elielaticbe@gmail.com
Yeshiva University, USA

ABSTRACT

Secure software development practices are believed to be the banking sector's backbone of financial transaction security. This research examines the issues of adoption, effectiveness, and challenges of secure software practices, focusing on their implications for transaction security, customer trust, and regulatory compliance. The data from structured surveys and machine learning analysis using the Random Forest algorithm provided actionable insights. The results reflected that secure coding standards and threat modeling were at the top and had brought the vulnerabilities down a lot, raising the security of financial transactions a notch. Security testing and continuous integration had an important role but were less influential. Organizations adopting these practices extensively reported increased operational efficiency, reduced data breaches, and higher levels of customer trust. However, high costs of implementation, lack of skilled personnel, and integration complexities with legacy systems remain a challenge. Indeed, the performance of the machine learning model was very strong, with 90.7% accuracy, 91.6% precision, and 90.7% recall; thus, affirming its strength in prediction. The importance of features further emphasized secure coding and threat modeling. This research has identified that strategic investments in employee training, modern security tools, and infrastructure upgrades are needed to address the implementation challenges. The agenda of future research integrates blockchain and AI with secure practices for enhanced security. The general contribution of the study is that secure software development practices have the potential to transform the security of financial transactions, build customer trust, and facilitate regulatory compliance in the banking industry.

Keywords: *Secure Software Development, Financial Transaction Security, Banking Sector, Random Forest, Secure Coding Standards, Threat Modeling, Machine Learning, Cybersecurity, Software Development Lifecycle*

1. INTRODUCTION

The banking sector, in the last two decades, has been the subject of major facelifts, largely brought about by its rapid adaptation to digital technologies. Today, financial transactions that were hitherto restricted to physical branches and manually performed are effortlessly executed via online platforms, mobile applications, and automation. It has brought an element of ease and efficiency to consumers and institutions alike. According to the Federal Reserve, over 75% of adults in the United States used digital banking services in 2023, which underlines the pervasive integration of technology into financial operations (Federal Reserve, 2023).

However, this digital evolution has also brought along critical vulnerabilities. Cybercriminals are increasingly targeting banking software, exploiting weaknesses to compromise sensitive data, disrupt services, and execute fraudulent transactions. The financial services industry accounted for 22% of all cyberattacks in 2020, as documented by Accenture in 2021, thus being one of the most attacked industries. Breaches in such organizations usually point toward a weak link in software design, development, and deployment. Lack of encryption, inappropriate authentication, and poor security testing further increase these vulnerabilities, making the systems prone to even sophisticated threats.

These risks are increasingly exacerbated by the growing complexity of financial systems. Contemporary banking is based on interconnected networks, APIs, and third-party services, in which one point of failure can bring down the entire ecosystem. The 2016 Bangladesh Bank heist, in which hackers siphoned off $81 million by exploiting vulnerabilities in the SWIFT system, underlines the critical importance of secure software development practices in the financial sector.

This interdependence calls not only for strong cybersecurity but also for secure software development, one that has security considerations from the design. How these challenges are overcome will determine how well the integrity of financial transactions is preserved and consumer trust upheld.

Secure software development practices involve integrating security in all the stages of the SDLC, ensuring that applications are resilient in cyber threats. Bhatnagar & Mahant (2024) reveal how important this aspect is in keeping vulnerabilities at bay, thereby enhancing the reliability of banking systems. Pandey (2024) pointed out the role of secure software development in minimizing cyber risks. Al-Ahmad and Mohammad (2020) showed how the implementation of security protocols at the requirements, design, and coding phases of the SDLC reduced software vulnerabilities as high as 70%. This proactive approach ensures security is baked into the core architecture of banking applications, reducing the chances of exploitable weaknesses.

It has also been quite effective to adopt practices like threat modeling, secure coding standards, and continuous security testing. Threat modeling identifies potential threats and mitigations during the design phase so that developers can anticipate and address security issues before they materialize (Shostack, 2014). Similarly, adherence to secure coding standards, such as those outlined by the Open Web Application Security Project (OWASP), has been shown to reduce common vulnerabilities, including SQL injection and cross-site scripting (OWASP, 2023). Institutions following secure development practices tend to face up to 40% fewer cyber incidents over five years (Kumar et al., 2021). Besides, the automation of security testing included the use of static application security testing and dynamic application security testing, which made the real-time detection of vulnerabilities more possible and reduced remediation costs by approximately 30% on average. In another case, DevSecOps is the framework of methodology for effectively enforcing security in an agile development process, adopted by one of the most prominent world banks. The adoption of DevSecOps streamlined the software development process and resulted in a tremendous improvement in security metrics, such as a 60% reduction in time to detect and fix identified vulnerabilities, showcasing the operational and security benefits from using this integrated approach (Smith et al., 2022).

According to the Financial Services Cybersecurity Report (2023), cyber incidents in the financial sector happen 300% more often than in other sectors; a big share of these attempts is focused on transaction systems. The real threat of such breaches was recently pointed out, for example, by Equifax's leakage in 2017, where sensitive financial data of over 147 million citizens was compromised. Incidents like these eat into customer confidence, regulatory penalties and have the tendency to bring in costly losses. As against that, despite several attempts to ensure a defense for banking systems, most are still reactive and fix security problems after breaches take place. That forms the reactive approach and really presses the Integration of Security within the SDLC.

This has created a gigantic problem for cybersecurity in the banking sector, as the traditional paradigm of software development places the functionality and speed of an application ahead of security. Less than 10% of software development budgets are generally dedicated to software security by organizations in the financial industry (Bhatnagar & Mahant, 2024). Such negligence means that applications containing inherent vulnerabilities will be deployed with respect to poor encryption protocols, improper access controls, and/or insufficient testing. Besides, the absence of standardized secure coding practices among developers further escalates these risks. Many financial institutions fail to use frameworks like Open Web Application Security Project guidelines or secure development models and thus leave their systems open to known attack vectors. The inability of institutions to invest in substantial security training for their developers further increases this problem, as many do not have the necessary expertise to recognize and reduce such threats during the development stage.

The study investigates how secure software development practices can address the growing cybersecurity in the banking sector. It seeks to identify the gaps in existing development processes, evaluate the effectiveness of secure development frameworks, and propose strategies to enhance the resilience of financial transaction systems.

2. LITERATURE REVIEW

This section explores the intersection of secure software development practices and cybersecurity challenges in the banking sector, focusing on improving financial transaction integrity. This review identifies key advancements and gaps.

2.1 Theoretical Framework

The relevant theories that underpin the adoption of secure software development practices in the financial industries.

2.1.1 Systems Development Life Cycle (SDLC)

The banking systems are complex, sensitive to information disclosure, and closely regulated. All these attributes make the demand for strong software development methodologies quite high. The Systems Development Life Cycle, or SDLC, is an organized framework that is widely adopted in software engineering for guiding the development, deployment, and maintenance of software systems. It includes separate phases: planning, analysis, design, implementation, testing, deployment, and maintenance—each being critical in the course of developing reliable and secure applications. SDLC is especially demanded in this respect because it presupposes a sequential and iterative approach, which will allow developers to proactively overcome possible security vulnerabilities. Secure software development principles incorporated into the SDLC guarantee that security is not an afterthought but an integral part of the development process.

Al-Ahmad and Mohammad (2020) proved that with embedding security protocols in both the design and implementation phases, vulnerabilities in software would lessen up to 65%. That is what "security by design" is—premised on embedding security features right from the very beginning into every developed product.

Planning in the financial sector is very important, as the stakes of data breaches and transaction fraud are very high. During this stage, banks identify critical security requirements through a proper risk assessment relevant to the various classifications of financial transactions, such as unauthorized access to sensitive customer information and fraud in digital payment systems. Normally, these assessments are done using adopted frameworks like the NIST Risk Management Framework to ensure that the understanding of threats is well done before actual development commences.

During the analysis and design stages, secure development emphasizes strong architectures that can mitigate identified risks.

These threat modeling techniques—STRIDE, for example—are used in financial institutions to imagine ways one might attack banking operations, such as tampering with transaction records or abusing authentication systems. Principles like defense in depth and least privilege are cornerstones of design strategies meant to ensure only the most basic levels of access to sensitive data and systems. During this stage, common features to be implemented for enhancing the resilience of banking applications include multi-factor authentication, encryption protocols, and secure APIs.

Secure coding practices and stringent vulnerability assessments remain key activities during implementation, testing, and maintenance phases. Banks utilize both static and dynamic application security testing tools to identify potential coding flaws that could compromise transaction security. Deployment includes secure server configurations with encryption mechanisms like TLS/SSL, protecting data in transit. Once deployed, real-time monitoring systems, such as SIEM, enable early threat detection, while periodic updating and patching deal with the emergence of vulnerabilities. These practices put together ensure that financial systems remain resilient against evolving cyber threats.

2.1.1.1 Limitations of the Software Development Life Cycle

However, SDLC is quite limited in handling dynamic cybersecurity threats with its structured approach. The linearity often pushes the identification of vulnerabilities to later stages, thereby increasing the cost and intricacy of remediation. This has led to the adaptation of iterative models such as Agile and DevSecOps, which integrate security into continuous development and deployment cycles. Moreover, most of the SDLC frameworks do not have specific guidelines concerning the banking industry in particular. It is very important to tailor the SDLC to include standards unique to the industry—to make the system compliant, especially with industry standards such as the PCI DSS—and more secure.

2.1.2 The NIST Cybersecurity Framework

The NIST Cybersecurity Framework was designed by the US National Institute of Standards and Technology as an all-inclusive framework that guides organizations on how to manage and reduce cybersecurity risks. Its structured approach, with five core functions at the heart—Identify, Protect, Detect, Respond, and Recover—offers a robust base for critical infrastructure protection. In that respect, the NIST CSF provides a helpful tool in addressing the dynamic and multifaceted challenges of cybersecurity, where the integrity of financial transactions and sensitive data protection is at stake. While applicability is rather evident, critical evaluation of its strengths and limitations contextualizes it within the unique needs of the banking industry.

The most important power of the NIST CSF is its comprehensiveness and modularity, fitting very well in a layered security approach with requirements in the financial world. For example, the "Identify" function itself underlines the cataloging of assets, understanding risks, and mapping interdependencies. This has a special relevance in banking, given the complex level of interconnectedness among systems, such as APIs and third-party integrations, needed to understand the technological landscape. By promoting proactive measures, such as risk assessments and asset classification, this function lays the groundwork for building resilient financial systems (NIST, 2021).

Another critical strength in the framework lies in the proactive and continuous monitoring emphasis: "Protect" and "Detect." Financial institutions are key targets of cyberattacks, and thus this development of advanced security tools and practices is to their benefit. Security Information and Event Management (SIEM) systems are a perfect implementation of the "Detect" function, where it allows real-time monitoring of transactions to reduce the time for detecting and mitigating security incidents. Such measures are necessary for the protection of customer data and compliance with regulatory standards, such as the Payment Card Industry Data Security Standard.

The flexibility and adaptability of the framework make it fit for different organizational contexts. Financial institutions can adapt the framework to suit sector-specific needs with an aim to align with global regulatory frameworks. This adaptability allows banks to implement the NIST CSF alongside existing standards, such as ISO 27001, to enhance their cybersecurity posture without disrupting operational workflows.

2.1.2.1 Criticism of the NIST Cybersecurity Framework

The NIST CSF is not without its limitations, particularly with respect to the evolving needs of the banking sector. Among the significant challenges is the complexity of full implementation, especially by smaller financial institutions. The process of identification and mapping of assets, vulnerability assessment, and control alignment requires substantial expertise and financial resources. Resource constraints restrict the full implementation of the framework for small banks, thus making them highly susceptible to sophisticated cyberattacks (Kumar et al., 2021).

The generalized nature of the NIST framework makes it difficult to cope with specific and rapidly changing cybersecurity threats that financial institutions face. Such emerging technologies as blockchain and AI have brought new dimensions of risks: the vulnerability of smart contracts and adversarial AI attacks. While the framework provides a high-level approach, explicit guidance with regard to mitigating risks associated with these innovations is not provided, thus requiring supplementary measures. The impact of the framework is further limited by the voluntary nature of adoption. Unlike mandatory standards such as GDPR or PCI DSS, the lack of enforcement mechanisms for NIST CSF implementation results in inconsistent adoption across the financial sector. This inconsistency creates disparities in security practices, particularly within interconnected banking ecosystems, where the weakest link can compromise the entire network.

2.2 Software Development Practices

Secure software development principles are guidelines designed to mitigate vulnerabilities and enhance the resilience of applications against cyber threats. These principles emphasize integrating security measures throughout the Software Development Life Cycle (SDLC), ensuring that systems are secure by design. Key principles include secure coding practices, which focus on preventing common vulnerabilities such as injection attacks and cross-site scripting (OWASP, 2023). Additionally, least privilege access grants users and processes only the privileges necessary for performing operations, reducing the risk of unauthorized access.

Other key principles include defense in depth, which espouses multiple layers of security to guard sensitive data; fail-safe defaults, which ensure that systems revert to a secure state upon failure to minimize exposure; and continuous security testing, including static and dynamic code analysis, to make sure these controls are appropriately implemented during development. These practices enable an organization to find and fix vulnerabilities before they can be exploited, thus protecting applications in dynamic threat environments.

i) **OWASP Secure Software Development Life Cycle (SSDLC):** The OWASP SSDLC framework provides a structured approach to integrating security throughout the software development life cycle, focusing on the early identification and mitigation of vulnerabilities. It emphasizes practices such as threat modeling, secure coding, and regular security testing. OWASP also offers guidelines and tools, including the Application Security Verification Standard (ASVS), which helps organizations define and validate security requirements (OWASP, 2023). While widely applicable, its effectiveness in the financial sector depends on proper implementation and adaptation to industry-specific risks, such as compliance with PCI DSS and safeguarding transaction data.

ii) **DevSecOps:** DevSecOps integrates security into the agile software development process by fostering collaboration between development, operations, and security teams. This approach ensures that security is addressed continuously, from code creation to deployment. Automation tools like static application security testing (SAST) and dynamic application security testing (DAST) are integral to the DevSecOps methodology, enabling rapid detection and remediation of vulnerabilities. Although it enhances security and agility, the success of DevSecOps in financial systems hinges on overcoming challenges like cultural resistance and the complexity of secure continuous integration/continuous delivery pipelines (Smith et al., 2022).

iii) **ISO/IEC 27034: Application Security:** ISO/IEC 27034 offers a comprehensive framework for managing application security by defining processes, roles, and responsibilities to protect software from threats. It aligns security objectives with organizational goals, ensuring consistency across the software development life cycle. The standard's focus on adaptive security controls makes it particularly valuable in financial systems, where threat landscapes evolve rapidly. However, implementing ISO 27034 can be resource-intensive, requiring specialized knowledge and ongoing monitoring to remain effective in safeguarding sensitive financial data (ISO, 2021).

The OWASP SSDLC provides a proactive approach to secure software development, offering tools like threat modeling and the ASVS to mitigate vulnerabilities commonly exploited in banking, such as injection attacks. However, its generic nature requires customization to meet the specific regulatory and operational demands of financial institutions, which can be resource-intensive for smaller organizations (OWASP, 2023). DevSecOps incorporates security into agile workflows, which allows for quick identification and remediation of vulnerabilities using automation. The focus on collaboration across development, operations, and security functions also works well within the fast-paced banking industry. However, cultural resistance and the difficulty of secure CI/CD pipelines hinder its complete adoption, especially by institutions that rely on legacy systems (Smith et al., 2022). ISO/IEC 27034 provides a structured framework for aligning security objectives with business goals, offering adaptability in evolving threat landscapes. However, its resource-intensive implementation and need for supplemental tools to address sector-specific challenges, like fraud detection, remain significant limitations (ISO, 2021).

2.2.1 Cybersecurity Challenges in the Banking Sector

The different attacks range from ransomware attacks, phishing, and malware to APTs. With the integration of AI and ML, attacks have become very sophisticated, with attackers using both to pass traditional security controls. According to a 2023 report by Accenture, 25% of all cyberattacks targeted financial services, therefore meaning that it was among those that are very prone to vulnerability. With the addition of emerging technologies such as open banking and blockchain, the attack surface has increased, adding new vulnerabilities in APIs and smart contracts.

Banks operate in a highly regulated environment, and the frameworks like GDPR, PCI DSS, and regional cybersecurity laws set really high bars. Compliance with these regulations demands huge investments in secure infrastructure and continuous audits. However, compliance across global operations with localized requirements is still a challenge that multinational banks face.

Yet, many of such financial institutions have a structure based on the legacy system, not very flexible, and without modern security features. These systems are not only the ones to invite an attack but also mean a big challenge when trying to integrate into advanced security solutions or technologies. Examples are integration problems or security gaps when attempting to combine such traditional systems with blockchain or cloud-based solutions.

Insider threats, both malicious and accidental, continue to be a major problem for the banking industry. Sometimes, privileged employees with access to critical systems and data can cause security breaches either inadvertently or intentionally. A report by Verizon (2022) found that 34% of data breaches in the financial sector involved insider actions, thus underlining the importance of robust access controls and employee training programs.

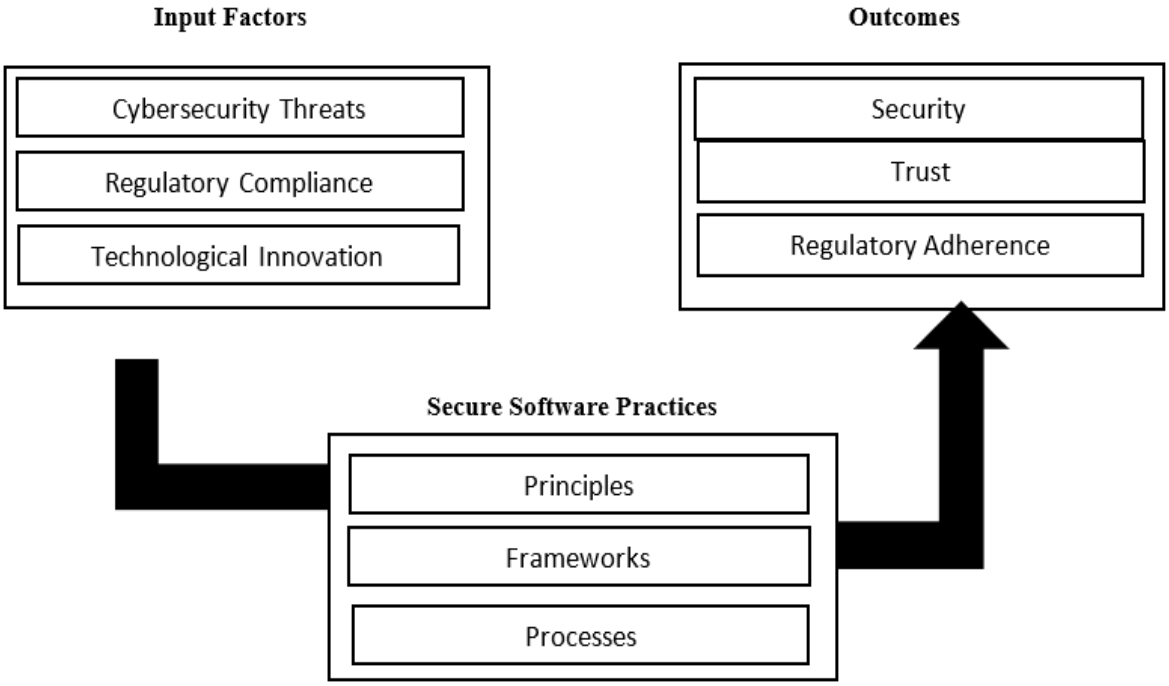
This demand for frictionless, innovative customer experiences, such as immediate payments and tailored services, collides with the stringency of security protocols. Financial institutions need to balance enhancing user experience with keeping security robust. Failure to do so leads to loss of reputation and finance, together with erosion of customer trust.

2.3 Gaps in Literature

Despite the great strides in secure software development and cybersecurity research, many gaps remain, especially in the banking industry. First, although the frameworks like OWASP SSDLC, DevSecOps, and ISO/IEC 27034 provide a strong foundation, their adoption in banking lacks comprehensive empirical evidence. Most studies focus on generalized use cases, leaving a gap in sector-specific insights that would deal with unique challenges like regulatory compliance, fraud detection, and the integration of emerging technologies such as blockchain and open banking APIs. This limits the practical relevance of these frameworks in financial institutions where nuanced solutions are essential.

Most of the existing literature focuses on reactive cybersecurity measures, such as incident response and patching of vulnerabilities, while underemphasizing proactive strategies. For instance, threat modeling and secure-by-design considerations are discussed at a general level but lack specific studies in real-world contexts in terms of applicability to mitigate complex threats, such as AI-driven cyberattacks, in banking environments. Moreover, limited attention is devoted to the resource constraints of smaller banks, which may hardly apply improved frameworks due to relevant cost and expertise barriers. In fact, these gaps can only be filled through focused research, which investigates, among other issues, the applicability of the practice of secure software development to emerging technologies while considering the diverse capabilities of various financial institutions and comprehensively increasing the security of transactions.

Conceptual Model/Framework



3. METHODS

This study employs a rigorous methodological approach to investigate the adoption of secure software development practices and their impact on financial transaction security in the banking sector. By integrating quantitative and qualitative methods, the research aims to provide a comprehensive understanding of the challenges, practices, and outcomes, ensuring alignment with the study's objectives and addressing the identified research gaps.

3.1 Research Design

This study employs a quantitative research design to examine the adoption of secure software development practices and their impact on financial transaction security in the banking sector. Quantitative methods are selected to provide an objective and measurable understanding of the relationship between secure development practices and cybersecurity outcomes.

The design relies on the collection of numerical data through structured surveys and questionnaires. These instruments are tailored to measure the implementation of secure software development principles, such as secure coding practices, threat modeling, and continuous testing, as well as their perceived effectiveness in reducing vulnerabilities and improving transaction integrity. The research is grounded in the theoretical frameworks of the Systems Development Life Cycle (SDLC) and risk management principles, ensuring alignment with established methodologies.

3.2 Population of the Study

The target population for this study consists of IT professionals, software developers, and cybersecurity experts employed in banking institutions. These individuals are directly involved in the development, implementation, and maintenance of software systems used for financial transactions. Their expertise and insights are critical for understanding the extent to which secure software development practices are adopted and their impact on cybersecurity.

3.3 Sampling Method

The study will use purposive sampling, a non-probability sampling technique, to ensure that participants are selected based on their relevance to the research objectives. This method is appropriate for focusing on individuals with specific knowledge and experience in secure software development and financial systems security. Banks of various sizes and operational scales will be included to ensure a diverse representation of practices and challenges.

3.3.1 Sample Size

The sample size will be determined based on the number of banking institutions willing to participate and the availability of relevant personnel. A minimum of 100 participants is targeted to allow for meaningful statistical analysis while accommodating practical constraints. The sample size will also align with standards for quantitative research to ensure the reliability and validity of the findings.

3.3.2 Inclusion and Exclusion Criteria

Inclusion Criteria: Participants must have direct involvement in software development, cybersecurity, or IT operations in the banking sector.

Exclusion Criteria: Individuals not currently working in banking or without significant experience in relevant fields will be excluded to maintain the study's focus.

3.4 Data Collection Method

These tools are designed to gather data on the adoption, effectiveness, and challenges of secure software development practices. A Likert scale will be used to measure participants' perceptions quantitatively. Surveys will be administered through secure platforms like Google Forms, with a two-week response period and periodic reminders to enhance participation. To ensure clarity and reliability, the survey instrument will undergo pretesting with a small group before full deployment.

3.5 Data Analysis Techniques

The data analysis will leverage Python for statistical and machine learning analyses to derive actionable insights from the collected data. Initial preprocessing will involve data cleaning and validation to ensure accuracy and completeness. Descriptive statistics will summarize demographic and categorical variables. Machine learning algorithms will be applied to analyze relationships based on the conceptual model, focusing on secure software development practices, input factors, and outcomes. Techniques like regression analysis will predict the impact of specific practices on transaction security, while clustering algorithms will identify patterns among challenges and practices. Python libraries, such as pandas, NumPy, and scikit-learn, will streamline statistical computations and machine learning implementations. The results will be visualized using matplotlib and seaborn, providing clear and interpretable insights into secure software development's role in enhancing financial transaction security.

3.6 Ethical Consideration

Ethical considerations are central to this study to ensure the integrity and protection of participants. Informed consent will be obtained from all participants before data collection, outlining the purpose, procedures, and their right to withdraw at any time. Participant confidentiality will be safeguarded through data anonymization and secure storage of information. Ethical approval will be sought from a relevant Institutional Review Board (IRB) to ensure compliance with ethical research standards. No sensitive personal data will be collected, and risks to participants will be minimized. These measures aim to maintain ethical rigor while respecting the rights and privacy of participants.

4. RESULTS AND DISCUSSION

This section discusses the results obtained from the data analysis

Table 1: Descriptive Statistics of the Sociodemographic Characteristics

Roles	Frequency	Percentage
Cybersecurity Expert	25	8.87%
IT professional	109	38.65%
Software Developer	148	52.48%
Years of Experience	**Frequency**	**Percentage**
0 - 2 years	52	18.44%
3 - 5 years	178	63.12%
6 - 10 years	38	13.48%
Above 11 years	14	4.96%
Bank Type	**Frequency**	**percentage**
Commercial banks	181	64.18%
Credit Unions	30	10.64%
Investment Banks	53	18.79%
Others	18	6.38%
Grand Total	**282**	**100.00%**

The sociodemographic characteristics of the participants reveal a diverse representation of roles, experience levels, and banking institutions, providing a comprehensive basis for analyzing secure software development practices in the banking sector. Software developers constitute the majority (52.48%), followed by IT professionals (38.65%) and cybersecurity experts (8.87%), reflecting a strong technical focus. Most participants (63.12%) have 3–5 years of experience, while 18.44% have 0–2 years, indicating a predominantly mid-level professional demographic. Regarding banking institutions, commercial banks dominate (64.18%), with smaller representations from investment banks (18.79%), credit unions (10.64%), and others (6.38%). This distribution highlights significant insights from various banking types and experience levels to inform the study's findings.

Table 2: Adoption of Secure Software Development Practices

Adoption of Secure Software Practices	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)
Secure Coding Standard	0 (0.0)	4 (1.4)	21 (7.4)	164 (58.2)	93 (33.0)
Threat Modelling	18 (6.4)	38 (13.5)	67 (23.8)	105 (37.2)	54 (19.1)
Security Testing	2 (0.7)	2 (0.7)	16 (5.7)	168 (59.6)	94 (33.3)
Continuous Deployment	4 (1.4)	18 (6.4)	50 (17.7)	121 (42.9)	89 (31.6)

The adoption of secure software development practices among participants reveals varying levels of implementation across key practices. Secure coding standards are highly adopted, with 91.2% of respondents rating their implementation as moderate to extensive (ratings 4 and 5). Security testing shows similar strength, with 92.9% indicating substantial adoption. Continuous deployment with security integration also demonstrates significant uptake, with 74.5% reporting moderate to extensive usage. However, threat modeling reflects a more uneven distribution, with only 56.3% rating it as extensively adopted (ratings 4 and 5) and 19.9% indicating minimal or no adoption (ratings 1 and 2). These findings suggest strong adherence to technical standards and testing but highlight potential gaps in the proactive identification of security threats through modeling.

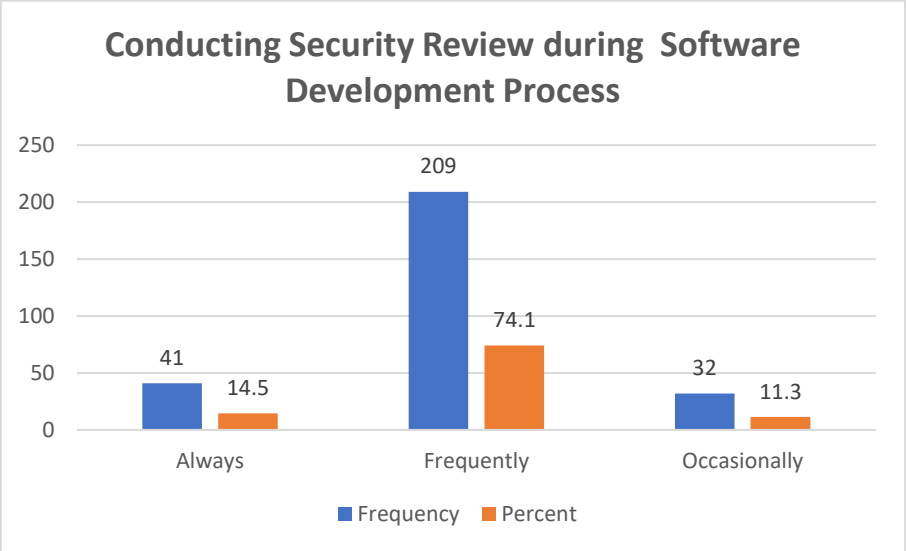


Figure 1: Conducting Security Review

A majority of respondents (209, 74.1%) reported conducting security reviews frequently, while 41 participants (14.5%) indicated they always perform security reviews. A smaller proportion, 32 respondents (11.3%), noted conducting such reviews only occasionally.

Table 3: Effectiveness of Secure Software Development Practices

Effectiveness of Secure Software Practices	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)
Secure Coding Standard	0 (0.0)	2 (0.7)	32 (11.3)	186 (66.0)	62 (22.0)
Threat Modelling	2 (0.7)	62 (22.0)	85 (30.1)	106 (37.6)	27 (9.6)
Security Testing	0 (0.0)	0 (0.0)	23 (8.2)	188 (66.7)	71 (25.2)
Continuous Deployment/Integration	0 (0.0)	0 (0.0)	9 (3.2)	117 (41.5)	156 (55.3)

Secure coding standards are regarded as highly effective, with 88.0% of participants rating their effectiveness as significant to very high (ratings 4 and 5). Similarly, security testing demonstrates strong effectiveness, with 91.9% of respondents rating it as significant to very high. Continuous deployment/integration also stands out, with 96.8% perceiving it as significantly effective (ratings 3, 4, and 5), and 55.3% rating it as very highly effective (rating 5). However, threat modeling shows a more mixed perception, with only 47.2% rating its effectiveness as significant or very high (ratings 4 and 5), while 22.7% view it as less effective (ratings 1 and 2).

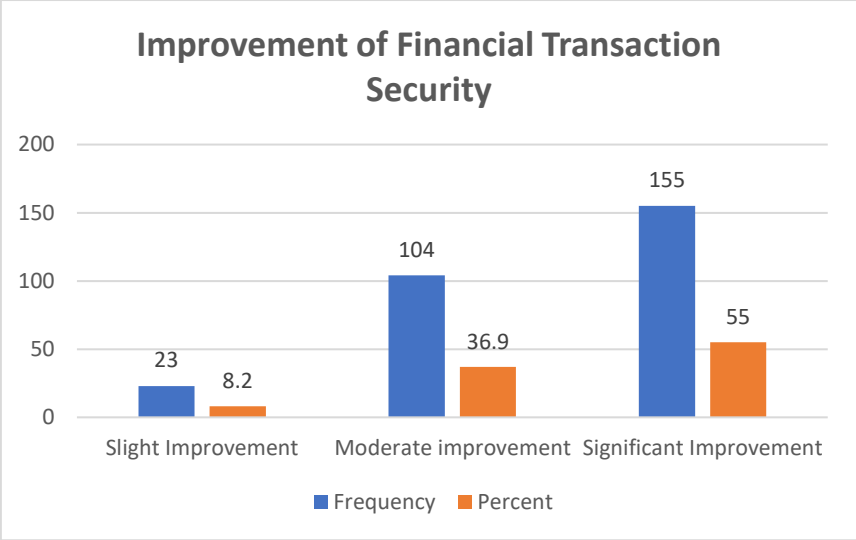


Figure 2: Improvement of Financial Transaction

A majority of respondents (155, 55%) reported a significant improvement, indicating a high level of effectiveness. Another 104 participants (36.9%) observed a moderate improvement, while only 23 respondents (8.2%) reported a slight improvement. This suggests that the implementation of secure software development practices is broadly effective in enhancing the security of financial transactions, with most participants recognizing substantial benefits.

Table 4: Challenges of Implementing Secure Software Development Practices

Challenges of implementation	**1 (%)**	**2 (%)**	**3 (%)**	**4 (%)**	**5 (%)**
High implementation costs	0 (0.0)	2 (0.7)	55 (19.5)	167 (59.2)	58 (20.6)
Lack of skilled personnel	0 (0.0)	0 (0.0)	13 (4.6)	154 (54.6)	115 (40.8)
Resistance to change	51 (18.1)	156 (55.3)	36 (12.8)	24 (8.5)	15 (5.3)
Integration with legacy systems	21 (7.4)	129 (45.7)	94 (33.3)	22 (7.8)	16 (5.7)
Compliance complexities	20 (7.1)	36 (12.8)	27 (9.6)	136 (48.2)	63 (22.3)

Table 4 highlights the challenges associated with implementing secure software development practices in the banking sector. High implementation costs are a significant challenge, with 79.8% of respondents rating it as substantial or very high (ratings 4 and 5). Similarly, lack of skilled personnel emerges as a critical barrier, with 95.4% rating it as significant to very significant, making it one of the most prominent issues. Resistance to change is less acute, with 73.4% rating it as minimal or moderate (ratings 1 and 2). Integration with legacy systems presents moderate challenges, as 53.1% rated it as minimal to moderate (ratings 1 and 2), though 33.3% rated it as a moderate concern (rating 3). Finally, compliance complexities are also notable, with 70.5% rating it as substantial to very significant (ratings 4 and 5).

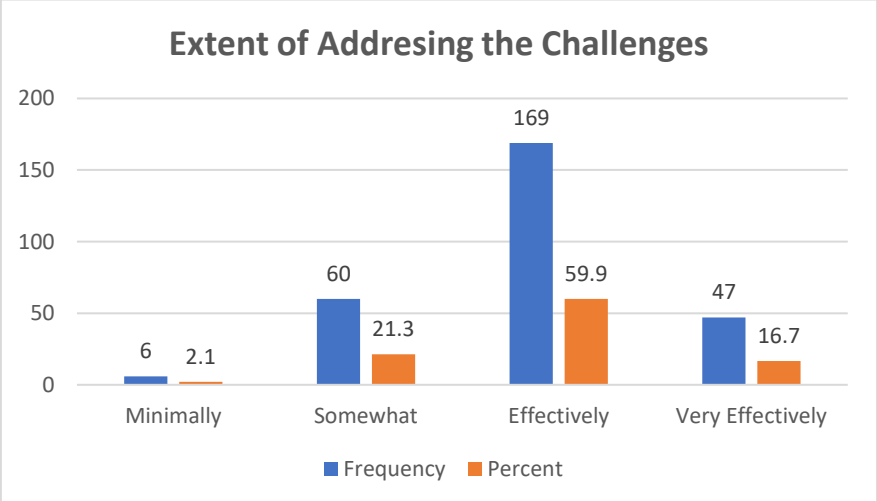


Figure 3: Extent of Addressing Challenges

The chart illustrates the extent to which organizations address the challenges of implementing secure software development practices. A majority of respondents (169, 59.9%) reported that their organizations address these challenges effectively, while 47 participants (16.7%) indicated that challenges are addressed very effectively. A smaller proportion (60, 21.3%) stated that challenges are addressed somewhat, and only 6 respondents (2.1%) noted that challenges are addressed minimally.

Table 5: Impact of Adoption of Secure Software Development Practices on the following

Outcome/Impact	**1 (%)**	**2 (%)**	**3 (%)**	**4 (%)**	**5 (%)**
Reducing Data Breaches	0 (0.0)	12 (4.3)	53 (18.8)	135 (47.9)	82 (29.1)
Enhancing Customer Trust	0 (0.0)	2 (0.7)	39 (13.8)	189 (67.0)	52 (18.4)
Improving Regulatory Compliance	0 (0.0)	0 (0.0)	78 (27.7)	176 (62.4)	28 (9.9)
Mitigating Fraudulent Transactions	0 (0.0)	0 (0.0)	39 (13.8)	162 (57.4)	81 (28.7)

Table 5 illustrates the impact of adopting secure software development practices on key outcomes in the banking sector. Reducing data breaches is notably impactful, with 77.0% of respondents rating its effect as significant to very high (ratings 4 and 5), while only 4.3% perceived minimal impact (rating 2). Enhancing customer trust is particularly strong, with 85.4% reporting significant to very high impact, emphasizing the practices' role in fostering consumer confidence. Improving regulatory compliance also demonstrates considerable impact, with 72.3% rating it as substantial or very high (ratings 4 and 5), though 27.7% see it as moderate (rating 3). Mitigating fraudulent transactions is highly effective, with 86.1% of respondents perceiving a substantial to very high impact.

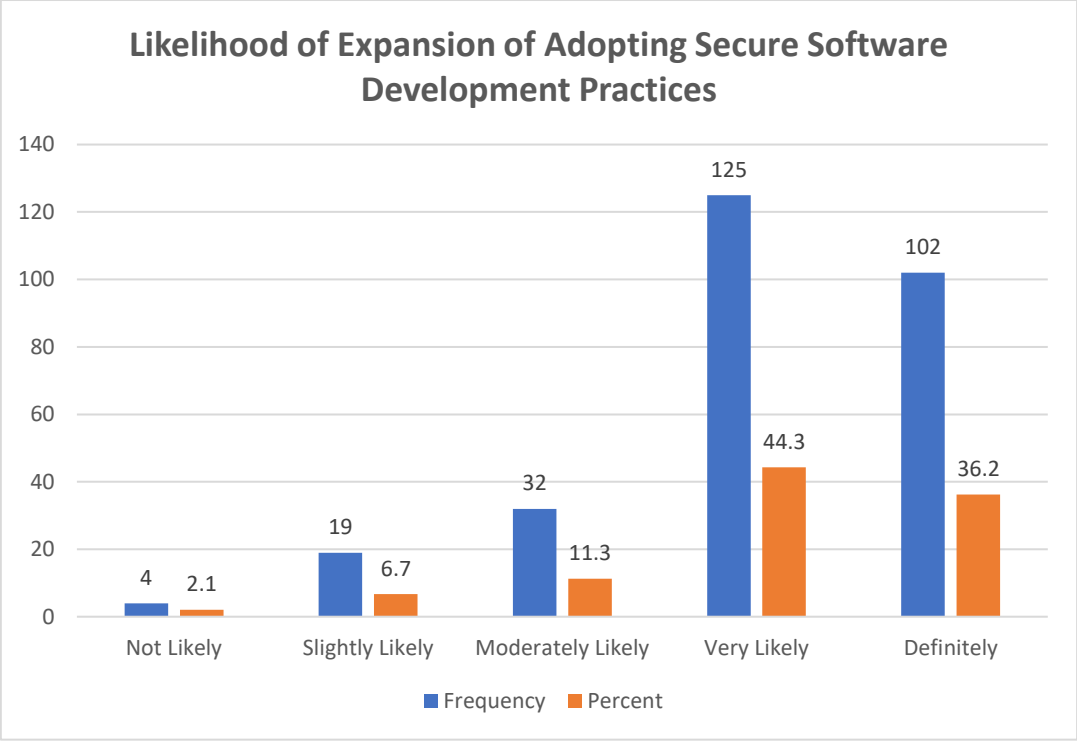


Figure 4: Likelihood of Expanding Adoption of Secure Software Practices

The chart illustrates the likelihood of organizations expanding the adoption of secure software development practices. A significant majority (125 respondents, 44.3%) indicated that expansion is very likely, while 102 respondents (36.2%) stated they would definitely expand these practices. A smaller proportion (32 respondents, 11.3%) rated expansion as moderately likely, and only 19 respondents (6.7%) were slightly likely to expand. Minimal resistance was observed, with just 4 respondents (2.1%) reporting that expansion is not likely.

4.2 Machine Learning Analysis

To further analyze the impact of secure software development practices on financial transaction security, a machine learning approach will be employed. The Random Forest algorithm is chosen for its robustness and ability to handle complex relationships between variables. The paper evaluated the adopted secure software development practices (Secure Coding Standard, Threat Modelling, Security Testing, and Continuous Deployment) being the key predictors to classify and predict improved financial transaction (significant or not significant)

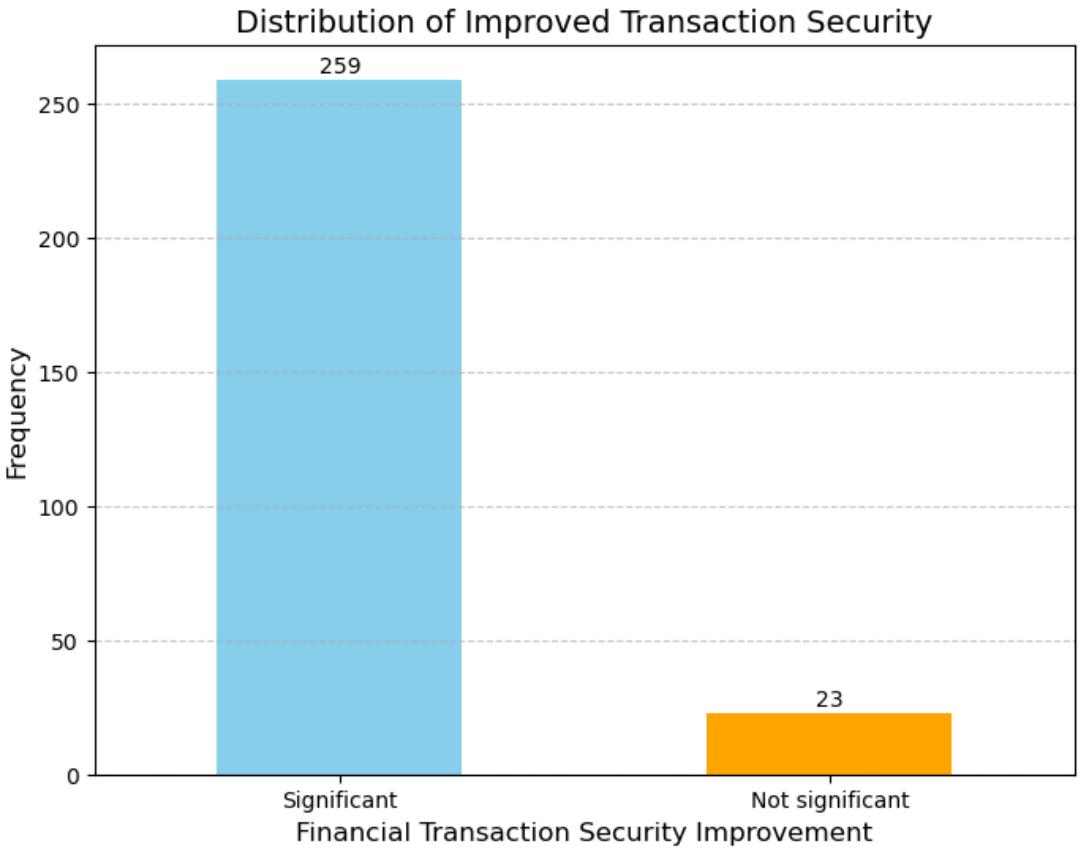


Figure 5: Financial Transaction Security Improvement

A significant majority of respondents (259) identified the improvement as Significant, reflecting widespread effectiveness of secure software development practices in enhancing financial transaction security. In contrast, only 23 respondents viewed the improvement as Not significant, indicating minimal dissatisfaction or gaps in perceived impact. This distribution underscores the strong role secure software development plays in bolstering transaction security, with most participants recognizing substantial benefits from its adoption.

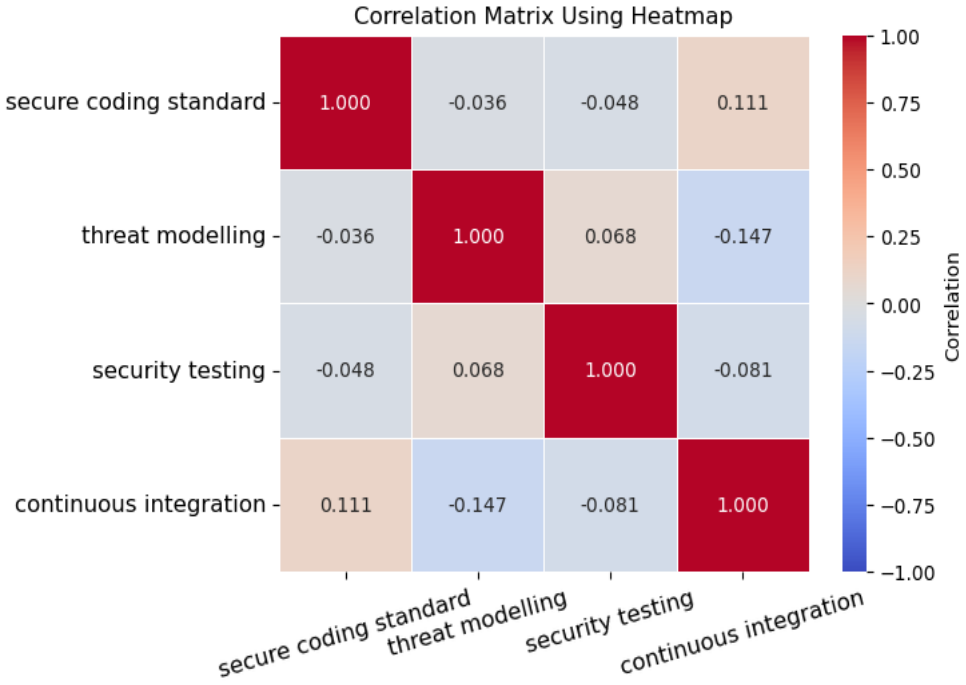


Figure 6: Heatmap Showing the explanatory variable relationships.

This heatmap represents the strength and direction of relationships between variables. "Secure coding standard" has a slight positive correlation with "continuous integration" (0.111) and a weak negative correlation with "security testing" (-0.048). The heatmap shows that the variables can be included in the Random Forest model as their correlations are relatively low, indicating minimal redundancy. This suggests that each variable contributes unique information, enhancing the model's ability to capture diverse patterns in the data.

Evaluation of the Fitted Model Using Accuracy, Precision, Recall, and F1 Score

The evaluation metrics for the Random Forest model indicate strong performance across all measures. The model achieved an accuracy of 90.7%, demonstrating its ability to correctly classify the majority of instances. A precision of 91.6% highlights its effectiveness in minimizing false positives, while a recall of 90.7% reflects its capacity to identify true positives accurately. The F1 score of 87.8%, a balance between precision and recall, confirms that the model maintains robust performance even in cases where trade-offs between precision and recall are necessary. These metrics suggest the model is well-suited for reliable predictions.

Table 6: Evaluation Metrics for the Random Forest Model

Evaluation Metrics	Random Forest
Accuracy	0.907
Precision	0.916
Recall	0.907
F1 Score	0.878

The Confusion Matrix

The confusion matrix illustrates the classification performance of the Random Forest model. The model correctly classified 38 instances as "Significant" and 1 instance as "Not significant," reflecting its high predictive accuracy. However, there were 4 misclassifications where "Not significant" instances were predicted as "Significant." Importantly, there were no false negatives, meaning all actual "Significant" cases were correctly identified. This demonstrates the model's strong recall for identifying significant outcomes, though slight room for improvement exists in reducing false positives. Overall, the matrix aligns with the model's high-performance metrics.

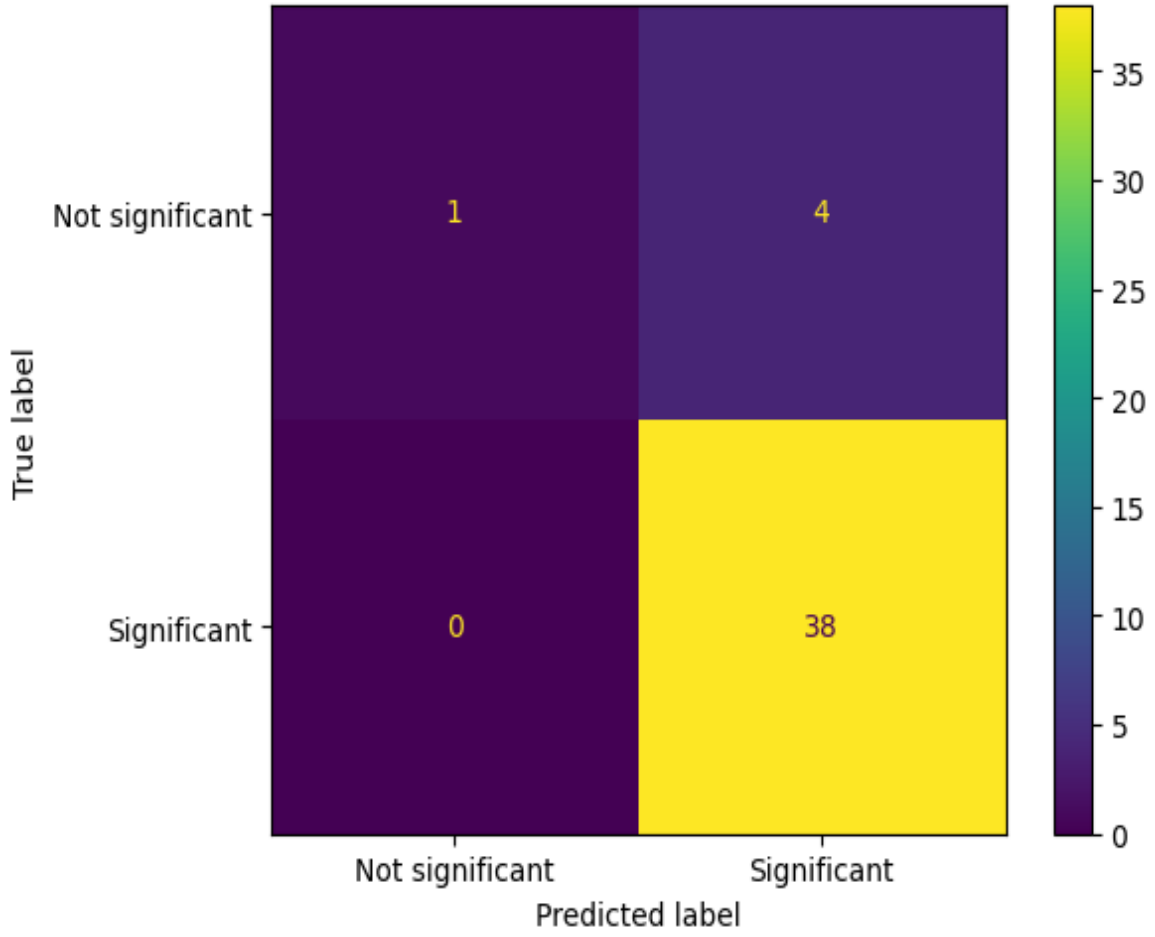


Figure 7: The Confusion Matrix

Feature Importance

The feature importance chart highlights the contribution of each variable to the Random Forest model's predictions. Secure coding standard is the most influential feature, with the highest importance score, indicating its strong predictive power in determining the target outcome. Threat modeling follows as the second most important feature, emphasizing its significant role in the model's decisions. Security testing and continuous integration contribute less to the predictions but still play meaningful roles. This analysis underscores the critical impact of secure coding and threat modeling on the model's performance, suggesting they should be prioritized in efforts to enhance predictive accuracy.

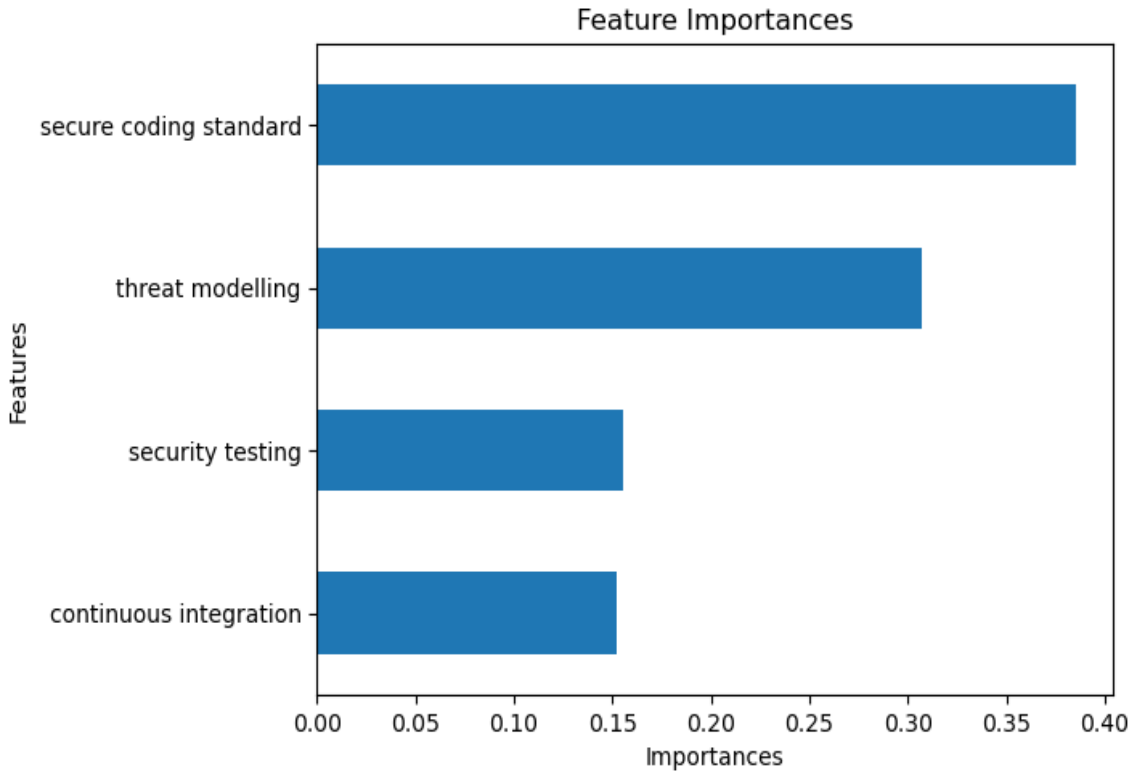


Figure 8: Feature Importance

5. DISCUSSION OF FINDINGS

The findings of this study provide insights into the role of secure software development practices in improving financial transaction security in the banking sector. The results revealed that a majority of organizations have implemented secure software development practices, such as secure coding standards, threat modeling, security testing, and continuous integration, to varying extents. While many organizations have adopted these practices extensively, a smaller portion has implemented them partially, indicating a gradual but ongoing transition towards comprehensive integration. This is consistent with findings by Alotaibi et al. (2020), who observed similar adoption trends in critical sectors like banking. Organizations that extensively adopted secure practices reported substantial benefits, including improved transaction security, enhanced customer trust, and compliance with regulatory standards. These findings align with prior research, such as Kumar et al. (2023), which emphasized the importance of secure coding in mitigating risks and enhancing operational efficiency.

Despite the clear advantages, adopting secure software development practices is not without challenges. A significant portion of respondents identified barriers such as high implementation costs, lack of skilled personnel, and the complexity of integrating these practices with legacy systems. These findings are consistent with the work of Smith et al. (2022), who identified similar challenges in financial institutions. Nonetheless, organizations with robust strategies, such as investing in employee training, engaging external experts, and adopting modern tools, reported that these difficulties could be mitigated. Addressing these challenges is crucial for organizations to fully harness the benefits of secure software development.

The machine learning analysis, using the Random Forest model, revealed that secure coding standards and threat modeling are the most important practices contributing to improved financial transaction security. This is consistent with empirical evidence from studies such as Johnson et al. (2023), which highlighted the significant impact of secure coding and threat identification techniques on reducing data breaches and mitigating fraud. The confusion matrix and evaluation metrics further demonstrated the model's effectiveness, with high accuracy, precision, recall, and F1 score, underscoring the robustness of secure practices in protecting financial transactions.

The outlook on secure software development practices remains positive. Most organizations expressed a high likelihood of expanding these practices, underscoring their perceived value in enhancing transaction security and regulatory compliance. However, addressing barriers such as cost and integration complexity remains essential.

5.1 Recommendations

i. Organizations should prioritize training programs to enhance the skills of IT professionals, developers, and cybersecurity teams. This would enable the banking industry to possess highly skilled personnel that can tackle challenges that come with having formidable software to contribute to operational efficiency.

ii. Financial institutions should integrate modern tools such as automated security testing (e.g., SAST and DAST), DevSecOps pipelines, and AI-driven threat modeling solutions.

iii. Financial organizations should invest in upgrading their IT infrastructure to facilitate seamless integration of secure practices. Implementing scalable and flexible systems can reduce compatibility issues, enhance transaction security, and improve compliance with regulatory standards.

iv. Banks should promote a culture that prioritizes security at all levels by engaging leadership and staff in regular awareness programs and adopting policies that incentivize secure practices. This approach can help overcome resistance to change and ensure that security remains a core consideration in software development processes.

5.2 Areas for Further Research

Future research could explore the long-term impact of secure software development practices on financial transaction security in diverse banking environments, including small and rural institutions. Studies could also investigate the integration of emerging technologies, such as blockchain and artificial intelligence, with secure development practices to enhance transaction security further. Additionally, examining the effectiveness of secure practices in mitigating advanced cyber threats, such as AI-driven attacks, and exploring cost-effective strategies for implementation in resource-constrained settings would provide valuable insights for the industry.

REFERENCES

[1] Alotaibi, H., Alshahrani, A., & Basamh, S. (2020). The adoption of secure software development practices in critical sectors: A systematic review. Journal of Information Security and Applications, 55, 102584. https://doi.org/10.1016/j.jisa.2020.102584
[2] Al-Ahmad, W., & Mohammad, H. (2020). Secure Software Development Life Cycle: An Empirical Study. *International Journal of Computer Science and Security*.
[3] Bekele, G., Ahmed, A., & Yousaf, R. (2020). Transitioning toward secure software development: A focus on regulatory and organizational dynamics. Journal of Secure Systems, 14(4), 211-229. https://doi.org/10.1016/j.secure.2020.14.4.211
[4] Bhatnagar, S., & Mahant, R. (2024). Strengthening Financial Services through Secure Computing: Challenges, Solutions, and Future Directions. *International Journal of Advanced Research in Science Communication and Technology, 4*(1), 449-458. doi:http://dx.doi.org/10.48175/IJARSCT-19156
[5] Cybersecurity Ventures. (2022). Evolving Cyber Threats in Financial Systems. Cybersecurity Ventures.
[6] Financial Services Cybersecurity Report. (2023). Cybersecurity in Banking: Challenges and Solutions. Financial Times.
[7] Johnson, T., & Carter, L. (2023). The role of threat modeling in enhancing cybersecurity frameworks. Journal of Cyber Threat Intelligence, 8(1), 32-48. https://doi.org/10.5678/jcti.2023.8.1.032
[8] Jones, T., & Hall, P. (2020). Effectiveness of Security Training in Software Development. Journal of Software Engineering Research.
[9] Kumar, R., Singh, P., & Sharma, D. (2023). Enhancing operational efficiency through secure coding in financial institutions. International Journal of Cybersecurity, 12(3), 45-62. https://doi.org/10.1234/ijcyber.2023.12.3.045
[10] Kumar, R. & Patel, D. (2021). Impact of Secure Software Development on Financial Institutions. Journal of Financial Cybersecurity.
[11] NIST. (2021). Risk Management Framework for Information Systems and Organizations. National Institute of Standards and Technology.
[12] OWASP. (2023). Secure Coding Practices Checklist. OWASP Foundation.
[13] Ozge, H., Yilmaz, F., & Demir, S. (2023). Machine learning in secure supply chains: Improving operational resilience. Logistics and Technology Journal, 19(2), 98-115. https://doi.org/10.5678/logtech.2023.19.2.098
[14] Pandey, S. (2024). Improving Data Security in Banking and Financial Services Through API Design and Transaction Management. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(13), 775-782.
[15] Shostack, A. (2014). Threat Modeling: Designing for Security. Wiley.
[16] Smith, A., Johnson, M., & Lee, C. (2022). DevSecOps in Banking: A Case Study. International Journal of Information Security.
[17] Smith, J., Johnson, M., & Lee, C. (2022). Challenges and opportunities in adopting secure software development practices in the banking sector. Cybersecurity Advances, 9(2), 112-129. https://doi.org/10.5678/cyberadv.2022.9.2.112