AMERICAN INTERNATIONAL UNIVERSITY – BANGLADESH

Faculty of Science & Technology

Programs (√): √ CSE □ EEE □ IPE □ CoE □ BBA □ BA

Course Name & ID: RESEARCH METHODOLOGY CSC4197

Semester: Spring 2024-2025	**Date of Submission:** …01/06/2025……

SUBMITTED TO:

……*Dr. Md. Abdullah - Al - Jubair*……

SUBMITTED BY:

ID	NAME
22-49442-3	Ayesha Khatun
22-46868-1	Fardin-Al-Sezan
22-47243-1	Ananna Monjur
22-47278-1	Brishav Mondal
22-47249-1	Abu Nayem Md Arman

FOR FACULTY USE ONLY

Faculty comments: ……,

Signature: ……. **Date: …….**

Blockchain in Smart Contracts and Cybersecurity

1. Introduction

Blockchain technology has introduced new paradigms in trustless, decentralized systems. One of its most groundbreaking applications is the smart contract, which allows self-executing agreements without intermediaries. Smart contracts are used in industries like finance, logistics, and law, offering automation and cost-efficiency. However, high-profile security breaches—such as the DAO attack (Mehar et al., 2019)—have exposed their vulnerabilities. This proposal investigates cybersecurity in smart contracts, aiming to identify threats and propose robust security practices.

2. Background

Smart contracts on platforms like Ethereum have revolutionized digital agreements, but their immutable nature makes them unforgiving to bugs. Vulnerabilities such as reentrancy and integer overflows are commonly exploited by attackers (Atzei et al., 2017). Despite available tools like Oyente for static analysis (Luu et al., 2016), developers often fail to integrate secure coding practices, leaving contracts open to attack (Zhou et al., 2020). Auditing helps reduce risks (Wüst & Gervais, 2018), but audits vary in quality and cannot catch everything. A secure-by-design approach is critical to prevent exploits and establish trust in smart contracts.

3. Problem Statement

Although smart contracts provide automation and decentralization, they introduce serious cybersecurity challenges. Exploits such as the DAO hack (Mehar et al., 2019) and Parity wallet failures highlight how unchecked flaws can result in significant losses. Even basic programming errors can have devastating consequences due to the immutable and public nature of blockchain. Studies show that many developers skip formal verification and auditing, leading to persistent security flaws (Zhou et al., 2020; Delmolino et al., 2016). This research aims to identify these vulnerabilities and recommend tools and frameworks to prevent them.

4. Objectives

General Objective:

To reduce smart contract cybersecurity risks through the identification of vulnerabilities and promotion of secure development techniques.

Specific Objectives:

1. Identify and categorize critical vulnerabilities (Atzei et al., 2017).
2. Investigate tools like Oyente and Mythril for static analysis (Luu et al., 2016).
3. Explore secure development frameworks like OpenZeppelin (Delmolino et al., 2016).
4. Recommend best practices for coding, auditing, and deploying contracts securely.

5. Contribution of the Study

This research will deliver a current and structured analysis of smart contract vulnerabilities, expanding on past literature (Atzei et al., 2017; Zhou et al., 2020). It will assess the effectiveness of popular security tools, coding standards, and auditing methods. Additionally, it aims to offer actionable strategies for both novice and experienced developers, with guidelines rooted in real-world failures like the DAO hack (Mehar et al., 2019) and supported by academic insights.

6. Related Work

Atzei et al. (2017) provided a comprehensive classification of smart contract vulnerabilities, including reentrancy, timestamp dependency, and unchecked external calls. Their survey remains foundational for understanding core risks. Luu et al. (2016) developed Oyente, a static analyzer that scans smart contract code for common bugs before deployment, advocating for formal verification as a preventive measure.

Mehar et al. (2019) analyzed the DAO attack, illustrating how reentrancy bugs, poor oversight, and lack of formal reviews led to massive losses. Delmolino et al. (2016) highlighted educational gaps by showing how even technically skilled students frequently introduced serious bugs in smart contracts.

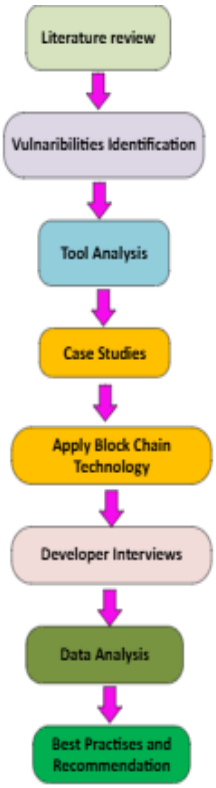
Zhou et al. (2020) conducted large-scale reviews of real-world smart contract exploits using data from Ethereum and GitHub. They found that bugs such as logic errors and integer overflows continued to appear despite available tools. Wüst & Gervais (2018) emphasized that while third-party audits help, their quality varies, and some vulnerabilities still escape detection. These works guide the present study's methodology and tool evaluation.

7. Research Methodology (Flowchart)

This research uses a **Mixed Methods** approach:

- **Qualitative:** Case studies (DAO, Parity), expert interviews.
- **Quantitative:** Smart contract data from GitHub and audit reports.

Flowchart:



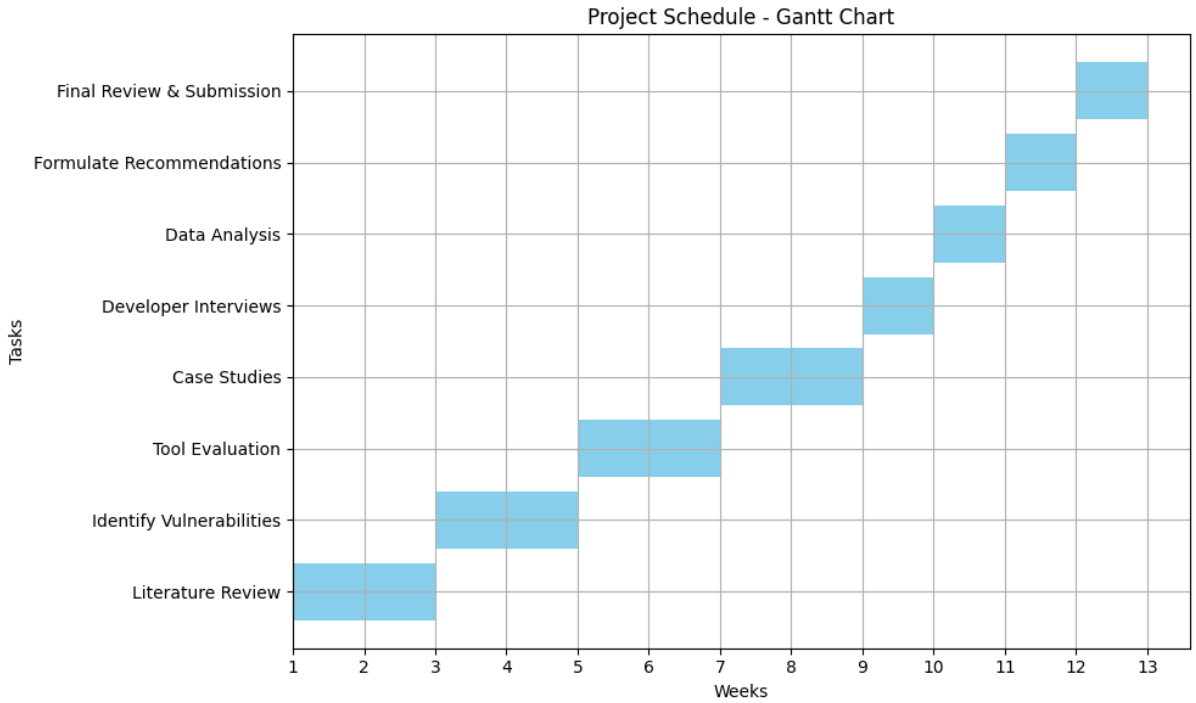
8. System Development Methodology & Justification

Method Chosen: Agile

Agile supports iterative development, where smart contract code is continuously tested, audited, and refined. According to Wüst & Gervais (2018), constant updates and code reviews are essential to catch security issues early. Agile allows better integration of static analyzers (Luu et al., 2016) and formal reviews at each sprint, ensuring faster identification and mitigation of vulnerabilities. Its adaptability makes it ideal for blockchain development, which is fast-paced and security-critical.

9. Schedule and Budget

Schedule (12 Weeks):



Budget:

Item	**Estimated Cost (BDT)**
Internet & Hosting Services	৳5,500
Developer Tools & Software	৳11,000
Research Article Access Fees	৳8,250
Miscellaneous (transport, printing, etc.)	৳2,750
Total	**৳27,500**

10. Data Collection Method

1. **Primary Sources:**
    - Expert interviews with blockchain developers and auditors.
    - Manual analysis of real attack cases like the DAO (Mehar et al., 2019).
2. **Secondary Sources:**
    - GitHub repositories and smart contract codebases (Zhou et al., 2020).
    - Research studies and whitepapers (Atzei et al., 2017; Luu et al., 2016).
    - Public audit reports and bug bounty results.

11. Significance of the Study

Smart contracts today manage billions of dollars in assets. If even a single vulnerability goes undetected, it can result in catastrophic losses. This research offers:

1. Educational guidance for secure coding (Delmolino et al., 2016).
2. Tool reviews that inform platform developers (Luu et al., 2016).
3. Policy input for organizations building DeFi protocols and DAOs (Mehar et al., 2019).
4. Contributions to standard-setting for blockchain security practices (Zhou et al., 2020).

12. Conclusion

Blockchain's power to decentralize and automate agreements is revolutionary, but smart contracts must be secure to maintain trust. By analyzing vulnerabilities, reviewing tools, and proposing best practices, this study aims to help developers and organizations create resilient smart contract systems. It builds on established research (Atzei et al., 2017; Mehar et al., 2019) while also generating original insights from real-world data and expert input.

13. References (APA 6)

Atzei, N., Bartoletti, M., & Cimoli, T. (2017). A survey of attacks on Ethereum smart contracts. *International Conference on Principles of Security and Trust*, 164–186. Springer.

Delmolino, K., Arnett, M., Kosba, A., Miller, A., & Shi, E. (2016). Step by step towards creating a safe smart contract. *International Conference on Financial Cryptography and Data Security*, 79–94. Springer.

Luu, L., Chu, D. H., Olickel, H., Saxena, P., & Hobor, A. (2016). Making smart contracts smarter. *ACM SIGSAC Conference on Computer and Communications Security*, 254–269.

Mehar, M. I., Shier, C., Giambattista, A., Gong, E., Fletcher, G., Sanayhie, R., Mclaren, M., & Laskowski, M. (2019). Understanding a revolutionary and flawed grand experiment in blockchain: The DAO attack. *Journal of Cases on Information Technology*, 21(1), 19–32.

Zhou, Y., Kumar, D., Mason, J., Saxena, P., & Bailey, M. (2020). Erays: Reverse engineering Ethereum's opaque smart contracts. *USENIX Security Symposium*, 1371–1385.