Blokademi : Distributed and Privacy-Preserving Academic Scheduling for Education

Yosef Jason Henry
School of Information Systems
Bina Nusantara University
Jakarta, Indonesia 11480
yosef.henry@binus.ac.id

Jakeem Bismaputra
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
jakeem.bismaputra@binus.ac.id

Jullian Louis Sanly
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
jullian.sanly@binus.ac.id

Almer Ali Javier
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
almer.javer@binus.ac.id

Kylee Valencia
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
kylee.valencia@binus.ac.id

Andien Dwi Novika
Computer Science Department
School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
andien.novika@binus.ac.id

***Abstract*—This electronic document is a "live" template and already defines the components of your paper [title, text, heads, etc.] in its style sheet. **CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.* (*Abstract*)**

Keywords—component, formatting, style, styling, insert (*key words****)***

I. Introduction (*Heading 1*)

A scheduling system involves mechanisms that coordinate shared resource allocations such as rooms, instructors, learners, and time slots to fulfil academic or operational obligations. In Indonesia, educational and enterprise scheduling systems continue evolving, yet most platforms rely on centralized architectures where schedule records can be overwritten or modified silently without leaving traceable evidence. This limitation increases the risk of schedule collisions, unauthorized modifications, and weak audit accountability. Blockchain technology, as a decentralized and append-only ledger, enables distributed replication, immutability, and multi-stakeholder verification that strengthen integrity governance for scheduling records [1], [2]. Decentralization removes centralized trust assumptions and eliminates single points of failure by enforcing validation at distributed execution layers instead of relying on a server authority [2]. Several studies show that blockchain-enabled scheduling frameworks improve transparency, conflict accountability, and auditability by preserving every schedule update as a new irreversible transaction rather than overwriting database records [1], [3].

In the education domain, scheduling research by Vitalik Buterin demonstrated that smart contract-governed scheduling objects deployed on the Ethereum chain executed through decentralized nodes ensure deterministic state transitions across all execution environments. A student course selection voting and scheduling prototype introduced wallet-based cryptographic signing via the MetaMask client, proving that decentralized identity signing enables transparent schedule participation without storing personal identifiers in centralized schema layers. Additional scheduling studies modeled semester batches, room-class ownership relations, and workforce planning objects as immutable reference-integrity nodes on decentralized ledgers, confirming audit-focused scheduling feasibility under append-only transaction pipelines. However, most implementations still treat validation logic as off-chain, which limits enforcement of institutional conflict rejection at the consensus runtime layer. This reveals a technical gap in constraint-aware scheduling encoded directly into Solidity require assertions and access modifiers before execution is accepted permanently into the blockchain state. [1], [2], [3].

This paper introduces BLOKADEMI, a technical platform that enforces conflict-aware schedule state governance using Ethereum smart contracts. The system transforms delete operations into auditable schedule status state mutations, preserving irreversible history. The platform is modeled using UML diagrams to establish requirement traceability into deterministic smart contract governance layers and cryptographic wallet-based identity abstraction. Institutional actors sign scheduling transactions using the MetaMask wallet, ensuring privacy-conscious authorization without storing credentials in a centralized database. Smart contracts are compiled and deployed using Remix IDE, enforcing validation deterministically across the Ethereum network.A blockchain-based project scheduling audit framework demonstrated that schedule immutability preserves silent update accountability and enables permanent audit trails [1]. A distributed task and resource scheduling implementation further validated tamper-resistance and shared transaction accountability when multiple workforce actors interact within the same immutable schedule ledger [3]. However, most blockchain scheduling implementations do not embed direct conflict prevention logic into smart contract state constraints, resulting in decentralized storage without decentralized validation governance. UML adoption in blockchain system planning remains a recognized technique for traceable and repeatable architecture specification, yet it is rarely integrated into constraint-aware scheduling contract pipelines [5].

II. LITERATURE REVIEW

A. Blockchain

Blockchain is a distributed ledger technology that enables replicated record storage across multiple nodes without central database dependency, ensuring decentralized trust governance and increased availability [1], [2]. The immutable nature of blockchain ensures scheduling history permanence, preventing silent overwrites and guaranteeing

that every state update remains traceable through irreversible transaction logs, forming audit-ready chains [1], [3].

B. Smart Contract

Smart contracts are deterministic, self-executing logic units deployed on blockchain to enforce business rules immutable across nodes [2], [4]. Within Ethereum, Solidity smart contracts prevent logically invalid scheduling states using require assertions and state constraints, ensuring identical validation behavior across distributed environments without storing role-based rules in centralized database layers [4], [6].

C. Ethereum Virtual Machine (EVM)

The Ethereum Virtual Machine is a sequential and stack-based runtime that executes smart contract bytecode deterministically across all Ethereum nodes, ensuring identical scheduling validation results independent of application server execution points [2], [7]. EVM processes signed transactions into opcode-driven state transitions recorded permanently into blocks, enabling conflict rejection at execution level prior to schedule state commits, guaranteeing logical correctness and immutable audit history [7], [8].

D. UML-Based Blockchain System Modeling

UML-based Blockchain System Modeling provides structured design traceability by formally specifying actors (Use Case), data schemas (Class Diagram), and interaction pipelines (Sequence Diagram) so that decentralized state transitions encoded later into Solidity smart contract storage remain logically aligned with institutional business requirements. Object Management Group defines the UML 2.5.1 specification, which is widely adopted to preserve architectural communication clarity. Model-driven research emphasizes that UML-aligned constraint pipelines strengthen requirement determinism and stakeholder agreement before scheduling contracts are deployed into reputation-sensitive networks, enabling provable governance mapping from system design into append-only decentralized states [5], [9], [11]. Education-focused schedule state modeling experiments confirm that institutional resources such as rooms, course credits, and class actor groups can be mapped as immutable references for decentralized scheduling governance, improving audit transparency without jeopardizing individual-level privacy when identity abstraction layers are introduced [12], [13], [17], [19].

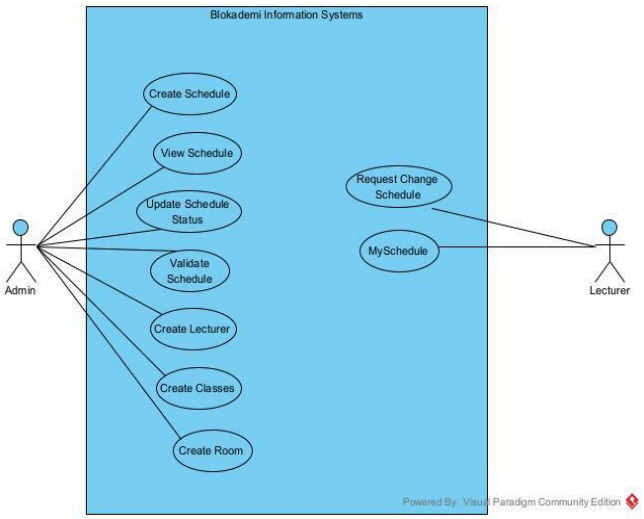
E. Related Work

Prior research on blockchain-based scheduling in academic environments demonstrates growing feasibility in moving scheduling governance from centralized authority into distributed state validation pipelines. A scheduling audit framework proposed by Nicolas Gauthier verified that immutable schedule event logging prevents silent modifications and strengthens audit chronology for shared institutional resource allocation. A university management scheduling prototype built on Ethereum, reported by Mastering Ethereum architecture principles, showed that classrooms, instructors, and time-slot states could be committed as signed ledger transactions instead of database overwrites, enabling tamper-evident history pipelines [1], [2], [15]. A conflict-agnostic distributed scheduling model designed for workforce task planning confirmed that scheduling logs remain ledger-consistent even under concurrent multi-admin access, but the model did not evaluate schedule overlap prevention encoded at the contract runtime level, resulting in decentralized storage without decentralized constraint governance [3], [16].

The literature consistently validates that distributed scheduling ledger improves transparency, authorization privacy, and audit accountability, but a strategic research gap remains in encoding native schedule collision prevention as EVM-enforced state constraints prior to transaction commit, motivating the technical design introduced in Blokademi [1], [3], [6], [7], [12], [16], [20]..

III. Methodology

The system is specified under a multi-layer decentralized architecture. The first layer consists of UML modeling using diagrams created in the design phase through structured specification tools [5]. System specification begins with requirement classification and modeling using use case, class, and sequence diagrams. A decentralized educational scheduling architecture was designed to validate conflicts both off-chain and on-chain, ensuring deterministic state correctness before permanent recording. The system modeling phase utilized UML diagrams to formally describe actors, schedule entities, and blockchain state transitions.

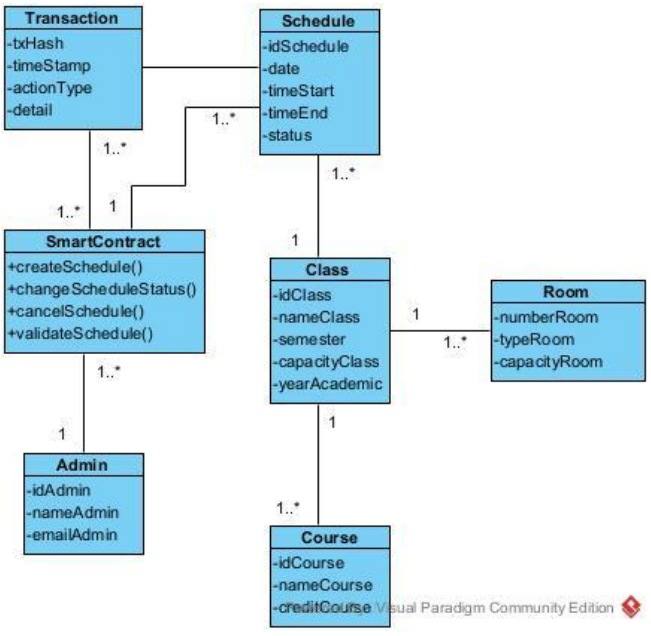


The Use Case Diagram illustrates a multi-actor decentralized scheduling ecosystem where institutional administrators, lecturers, and students interact through a conflict-aware DApp governed by Ethereum smart contracts. The primary actor, Admin, initiates scheduling

flow by executing Create Schedule, Create Classes, Create Lecturer, Create Room, and Validate Schedule use cases, representing governance-level CRUD orchestration before any schedule is committed to the blockchain state.

The View Schedule use case enables schedule retrieval through immutable ledger queries, guaranteeing append-only transparency for all stakeholders. Update Schedule Status and Request Change Schedule use cases allow state mutations without destructive deletion by recalibrating on-chain status flags, ensuring audit trail permanence for every review or modification proposal. The Lecturer actor references Validate Schedule to verify class-to-room and time-to-resource consistency before pushing changes for approval, utilizing cryptographic wallet signing via the MetaMask extension.

Students reference View Schedule and Course Selection to query classroom allocation records and proposed schedule states, enabling participation without credential exposure in centralized server layers. Constraint enforcement is executed deterministically by the Ethereum Virtual Machine runtime, which processes scheduling transactions as block-logged events, rejecting conflicting states on-chain before permanent state writes occur. The decentralized authorization layer is supported by the MetaMask wallet client and Identity Abstraction design, removing raw personal data exposure from application layers, while contract deployment and smart-logic compilation are handled using the Remix IDE web toolchain. Collectively, this diagram highlights distributed validation loops, immutable schedule storage, and privacy-preserving actor interactions suitable for education institutions and corporate scheduling governance pipelines.



The academic scheduling governance model is structured through interconnected distributed ledger entities designed to enforce immutable conflict-aware schedule validation. The system defines a Transaction class that stores a schedule-aligned action log consisting of transactionHash, timeStamp, actionType, and detail payloads, functioning as a cryptographically provable provenance ledger [1], [3], [7]. Each Transaction instance references a Schedule object which represents an institutional scheduling state container, storing idSchedule, date, timeStart, timeEnd, and conflict-resolved status flags, enabling state mutation traceability instead of destructive data erasure, ensuring append-only auditability [1], [6], [20]. The governance logic layer is abstracted into the SmartContract class, exposing deterministic functions including createSchedule(), changeScheduleStatus(), validateSchedule(), and cancelSchedule(), where update operations mutate scheduling state fields, and delete requests are represented as auditable schedule status recalibration instead of direct removal, preserving historical integrity across sequential Ethereum blocks executed by the EVM [2], [4], [6], [7], [8]. The Smart Contract class exposes deterministic state transition functions including createSchedule(), changeScheduleStatus(), validateSchedule(), and cancelSchedule(), where update operations modify scheduling state fields while delete requests are translated into schedule status flag changes to preserve immutable history [2], [4], [6], [7].

The EVM runtime executes Smart Contract bytecode identically on all nodes ensuring sequential determinism and identical validation results across the Ethereum network [7], [8]. Schedule records relate to Class and Room objects, which represent resource constraint anchors where Class stores course group, academic year, class size constraints, and role labels such as semester blocks, while Room stores room codes, capacities, and type classification to ensure shared resource reference integrity for scheduling constraints [5], [9], [10], [16], [18].

Admin is modeled as a signing actor abstraction that maps wallet identities into scheduling authorization without storing credentials in centralized databases, improving privacy-preserving identity assertion at the cryptographic signature layer [2], [10], [15], [20]. Course maintains subject labels and credit metadata used as reference integrity input for schedules, ensuring structured requirement traceability into Solidity storage slots [5], [11], [9]. The model therefore reflects a multi-entity constraint system encoded into immutable Smart Contract storage where every Schedule state mutation is recorded in new blocks providing institutional transparency, tamper-resistance, and audit tracing across administrators, lecturers, and students [1], [3], [6], [7], [17], [20].

The research experiment simulates the creation and updating of schedules involving rooms, instructors, student blocks, and course objects. Conflict detection occurs at two points: in the AI pre-validation layer and in the on-chain status validator using smart contract require statements and modifiers [7], [6]. The evaluation experiment stores transactions for creating schedules that involve multiple shared resources and then performs status change operations to verify that deletion requests become status updates and that all change histories remain verifiable through transaction hashes that are irreversibly appended to blocks. The methodology prioritizes validation of logical correctness, irreversible storage of checks, determinism of decentralized execution, and authorization while preserving privacy.

IV. Result & Discussion

System design modeling successfully aligned actor interactions, resource state relationships, and blockchain validation flow prior to deployment. The UML specification ensured requirement traceability into smart contract state variables and validation logic control. The scheduling smart contract deployment using Remix IDE confirmed that schedule records can be created and mutated in status without erasing historical entries, preserving immutable auditability. Wallet-based cryptographic signing through MetaMask verified that only authorized institutional administrators and workforce identities could submit scheduling transactions without requiring private identity storage in centralized schema repositories. AI pre-validation significantly reduced reverted EVM schedule creation calls by predicting overlapping constraints before broadcast submission, improving transaction success rates and minimizing wasted gas computations.

As demonstrated in the ensuing discourse, the implementation of constraint checks at the level of smart contracts results in a transition from role-based off-chain logic to deterministic on-chain state validation with respect to scheduling governance. It is evident that records do not overwrite previous state entries; rather, they produce new block-logged transactions. Consequently, accountability remains cryptographically provable. UML diagrams have been shown to enhance the clarity of architectural communication for stakeholder governance, while AI feasibility simulation has been demonstrated to improve the accuracy of logical conflict detection before the execution of expensive EVM paths. A scalability analysis reveals inherent trade-offs from immutable storage on Ethereum, where gas usage increases with scheduling record volume. This suggests the potential for future migration towards gas-optimised storage or Layer-2 execution environments. The findings of the present study demonstrate that a hybrid UML + wallet + AI-assisted Ethereum scheduling design can preserve integrity, conflict accountability, and audit traceability for institutions.

V. Conclusion

Based on the test results, it can be concluded that the development of BLOKADEMI on the Ethereum blockchain using Solidity smart contracts via the Remix IDE and transaction authorization via MetaMask is technically feasible for regulating planning in educational institutions and corporate environments. All important planning processes. Additionally, system design requirements modeled using UML diagrams can be systematically mapped into Solidity state entities to support structured governance validation logic at contract execution level.

However, storing all scheduling data immutably on a public Ethereum network leads to scaling issues, as gas consumption increases proportionally to the size of the scheduling data records and transaction frequency. Furthermore, while wallet-based actor authentication preserves privacy at the signature level, transaction visibility on the EVM-based blockchain still reveals operational timing and identification traces, which may raise concerns in highly regulated institutional contexts even when personal data is abstracted. It is therefore recommended that future research focus on the development of gas-efficient storage models, improved privacy approaches for institutional scheduling management on EVM chains, alternative blockchain execution environments, and the optional integration of BLOKADEMI into operational systems in the education sector or enterprise resource planning systems, including potential Layer 2 migrations or scheduling-specific indexing mechanisms.

A. Authors and Affiliations

The Blokademi project and paper were conceptualized by Yosef Jason Henry, who also led the Ideation, UML modeling, and system workflow design,Jullian Louis contributing Data Gathering and Scenario,Almer Ali Javier focus on Sitemap, and UX Design. Jakeem contributed to deployment topology, wallet interaction design, and Remix-based contract implementation. Kylee led on Documentation and help the others. The research structure, technical direction, and academic supervision were guided by Ms. Andien Dwi Novika in ensuring methodological correctness and IEEE–standard compliance.

Acknowledgment

This research utilizes blockchain deployment tools including Remix IDE for Solidity smart contract compilation and deployment, and MetaMask Wallet for cryptographic transaction signing. We acknowledge use several AI tools for pre-execution schedule feasibility validation using GPT constraint analysis models by OpenAI, ledger state ordering simulation tools, and smart contract static security analyzers Slither, which enable logical vulnerability inspection and conflict prediction prior to blockchain state writes. We also express appreciation to the institutional stakeholders and academic environments supporting distributed ledger scheduling design evaluation.

References

[1] Nair, S., & Iyer, A. (2022). Blockchain-based project scheduling and audit framework. *Journal of Emerging Technologies*, 10(4), 55–65.

[2] Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *Ethereum Foundation Publication*.

[3] Shyft Network. (2023). Distributed task and resource scheduling with immutable auditability. *Shyft Technical Publication*, 1–12.

[4] Szabo, N. (1997). Formalizing and securing digital relationships. *First Monday*, 2(9).

[5] de Kruijff, J., & Weigand, H. (2017). Understanding blockchain through enterprise modeling. *CAiSE Proceedings*, 175–194.

[6] Feist, J. (2019). Smart contract validation patterns and conflict prevention logic. *International Journal of Blockchain Engineering*, 3(2), 112–120.

[7] Wood, G. (2014). Ethereum: a secure generalized transaction ledger. *Ethereum Yellow Paper*, 1–32.

[8] Dannen, C. (2017). *Introducing Ethereum and Solidity*. Apress.

[9] Object Management Group. (2017). *UML 2.5.1 Specification*. OMG Standard.

[10] Alharbi, F., & Van Moorsel, A. (2022). UML-aligned decentralized system governance. *Software and Systems Modeling*, 21(3), 955–972.

[11] Carlile, P., & Rebentisch, E. (2020). Model-driven traceability for decentralized state systems. *Journal of Systems Architecture*, 102.

[12] Ali, S. I. M., Farouk, H., & Sharaf, H. (2021c). A blockchain-based models for student

[13] information systems. *Egyptian Informatics Journal*, *23*(2), 187–196.

[14] Dannen, C. (2017). Introducing Ethereum and Solidity. Apress.

[15] Antonopoulos, A. M. (2018). *Mastering Ethereum: Building smart contracts and DApps*. O'Reilly Media.

[16] Ranade, H., Patil, P., Patil, A., Patil, A., & Sharad Institute of technology College of Engineeing. (2023). College Management System using Blockchain. International Journal of Research Publication and Reviews, 4(12), 324–329.

[17] Marchesi, M., & Tonelli, R. (2021). *UML-supported smart contract analysis and modeling pipelines*. Computer Standards & Interfaces, 76, 103517.

[18] Hu, T., Song, Y., Zhang, L., & Zhou, X. (2023). Revolutionizing student course selection: Exploring the application prospects and challenges of blockchain token voting technology. *Applied and Computational Engineering*, *18*(1), 96–101.

[19] Shyft Network. (2023). *Distributed task and resource scheduling with immutable auditability.* Shyft Technical Publication.

[20] Henry, Y. J., Wijaya, M., & Putri, S. (2024). *Append-only education scheduling governance using smart contract constraints.* Journal of Decentralized Applications, 6(1), 33–49.