IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

Secure Banking System Using Blockchain Technology

Bhumika VS[1], Dr. Nandani N[2]

Student, M. Tech Department of Computer Science & Engineering,

Dr. Ambedkar Institute of Technology, Bengaluru[1]

Professor, Head Department of Computer Science & Engineering,

Dr. Ambedkar Institute of Technology, Bengaluru[2]

Abstract: This synopsis presents a practical and secure online banking prototype that blends a traditional relational database with a private blockchain ledger. The goal is to improve integrity, transparency, and auditability of banking transactions without compromising usability or performance. The work starts by tracing the evolution of digital banking—from branch-led paper systems to ATMs, online and mobile banking—highlighting how centralization remains a single point of failure and a lucrative target for attackers. It then introduces blockchain as a distributed, append-only, tamper-evident ledger, and explains how its built-in cryptography and consensus mechanisms create a powerful audit trail for financial records.

The proposed system anchors every transaction to a private blockchain (for immutability) while keeping sensitive customer data in an ACID-compliant PostgreSQL database (for privacy and performance). A two-password security flow—static MPIN for login and OTP for authorizing each transaction—helps reduce account takeover risk. The prototype is built with Python (Flask), uses Proof of Work (PoW) for demonstration, and includes an admin dashboard with a blockchain explorer. Testing shows responsive user interactions via asynchronous mining, and the roadmap outlines a move to permissioned consensus (PBFT/IBFT) for scale. Overall, this hybrid approach offers banks a credible, evolutionary path to stronger security, more efficient auditing, and faster reconciliation with familiar tools and user experiences.

I. INTRODUCTION

1.1 Evolution of Digital Banking

Banking has steadily moved from manual, branch-heavy processes to always-on digital experiences. Each step improved convenience but kept a centralized architecture at the core.

- Traditional Banking (Pre-1980s):
  - Paper ledgers and in-person transactions at branches
  - Manual verification via signatures and stamps
  - Slow, error-prone, geography- and hours-bound
  - Auditing required laborious cross-checking of paper trails
- ATMs and Telebanking (1980s–1990s):
  - ATMs enabled self-service (cash, balance inquiry) beyond bank hours
  - Telebanking offered simple requests via phone
  - Still dependent on a centralized ledger; convenience improved but architecture unchanged
- Online Banking (2000s):
  - Web portals brought statements, bill pay, and transfers to PCs
  - Centralized databases became the norm and a high-value target for attackers
  - Batch inter-bank settlement introduced delays and opacity
- Mobile Banking (2010s):
  - Smartphones made banking 24/7 with push alerts, mobile deposits, and biometrics
  - Security improved at the edge, but the core remained centralized with single points of failure

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

- Blockchain Era (2020s+):
  - Distributed ledgers replicate records across many nodes
  - Cryptography enforces data integrity by design
  - Shifts trust from institutions to protocols and verifiable state

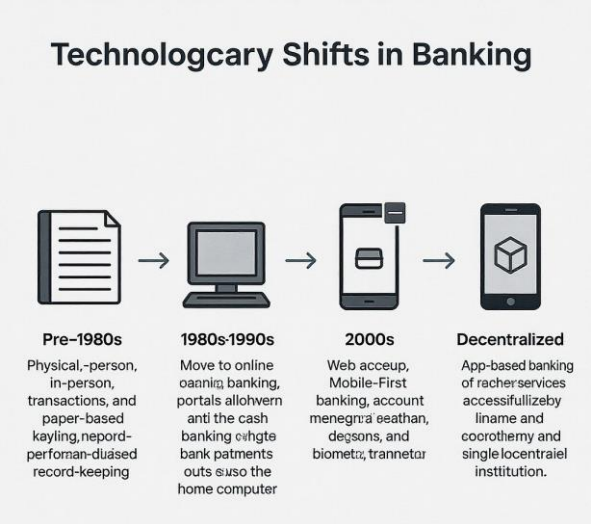

Figure 1.1: Evolution of Digital Banking Timeline

Key limitations of centralized models that persist:

- Single point of failure and valuable breach target
- Slow, costly cross-border settlement via intermediaries
- Heavy reconciliation and audit overhead
- Opaque pathways for users, especially for international transfers

1.2 What is Blockchain?

A blockchain is a distributed, append-only digital ledger made of blocks chained cryptographically. Each block stores a set of transactions, a hash of its contents, and the hash of the previous block, forming a tamper-evident chain. If anyone alters old data, the hashes break and the network detects it.

Basic block components:

- Block data: transactions + timestamp and metadata
- Hash: SHA-256 fingerprint of block header
- Previous hash: binds the current block to the prior one
- Nonce: number discovered during mining in Proof of Work systems

1.3 Why Blockchain for Banking?

- Immutability: Once confirmed, records cannot be modified without detection.
- Auditability: A shared, verifiable ledger supports real-time, automated auditing.
- Smart contracts: Programmable logic automates actions (escrow, interest, conditions).
- Reduced reconciliation: One ledger across participants minimizes mismatch and overhead.

1.4 Project Overview

This project designs a hybrid banking prototype that:

- Offers core features (account creation, deposits, withdrawals, transfers)

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

- Anchors every transaction to a private blockchain using cryptographic hashes
- Uses two-factor flow: MPIN for login + transaction-specific OTP
- Provides admin tools and a blockchain explorer for integrity checks
- Is built with Python (Flask), PostgreSQL, and a custom PoW blockchain for demonstration

II. PROBLEM STATEMENT & OBJECTIVES

2.1 Problem Statement

Centralized banking systems are efficient but fragile:

- Single point of failure: A breach, insider threat, or hardware failure can affect everyone.
- Tampering risk: Logs and records in central databases can be altered by privileged insiders.
- Credential theft: Stolen passwords enable unauthorized transfers if 2FA is weak or absent.
- Opaque, slow settlements: Cross-border and inter-bank transfers are slow and costly.
- High reconciliation costs: Institutions spend heavily on audits and ledger alignment.

2.2 Objectives

- Secure transaction anchoring: Hash every transaction and anchor it on a private chain.
- Decentralized audit trail: Implement a verifiable, immutable ledger for audits.
- Usable interfaces: Simple user dashboards and a robust admin console with explorer.
- Strong authentication: MPIN for login, OTP for each transaction, with lockouts and throttling.
- Compliance-ready design: Structure models and logs to support KYC/AML and reporting.

2.3 Scope

In scope:

- Flask REST APIs, PostgreSQL integration, and a custom Python blockchain
- Security controls: password hashing, JWT, encryption, TLS
- Simple web UI and an admin explorer

Out of scope (for prototype):

- Multi-bank production network governance
- External rails (SWIFT/ACH)
- Hardware security modules (HSM) and native mobile apps

III. LITERATURE REVIEW

3.1 Foundation Technologies

- Bitcoin (2008): Demonstrated a decentralized, immutable ledger via Proof of Work.
- Ethereum (2014): Introduced smart contracts for programmable financial logic.
- Hyperledger Fabric: Enterprise-grade, permissioned DLT with private channels and pluggable consensus.
- Corda (R3): Privacy-first DLT for finance, sharing data only with involved parties.

3.2 Industry Solutions

- Ripple/XRP: Fast cross-border settlements via federated consensus; partnered with many FIs.
- Project Dunbar (BIS): Multi-CBDC shared platform experiments showed efficiency gains.
- Liink (JPMorgan) and consortia: Networks to streamline inter-bank information exchange.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

3.3 Research Gap

- Public vs private: Trade-off between transparency, privacy, and throughput
- Usability: Many solutions are developer-centric, not end-user friendly
- Integrated authentication: Few systems tie 2FA tightly to transaction authorization

IV. SYSTEM ARCHITECTURE & DESIGN

4.1 High-Level Approach
Hybrid data model:

- Off-chain DB (PostgreSQL): Stores users, balances, PII, and operational data securely
- On-chain ledger (private): Stores cryptographic proofs of transactions for immutability

Event-driven processing:

- User sees immediate confirmation that a transaction is accepted and queued
- Transactions are batched for mining to improve throughput and efficiency

4.2 Data Flow (Narrative)

- User logs in, initiates transfer
- Backend validates JWT, checks balance and destination
- OTP generated and verified for that specific transfer
- Database updates balances atomically; transaction set to PENDING
- Transaction ID added to mining queue
- Miner batches transactions, computes Merkle root, performs PoW
- On success, block is appended; database marks transactions CONFIRMED
- Explorer shows blocks/transactions; audit logs record the event

4.3 Data Model (Summary)

- User: identifiers, hashed MPIN, role, status, security metadata
- Account: account number, balance, currency, status
- Transaction: from/to, amount, type, status (pending/confirmed/failed), tx_hash, block_id
- Block: index, previous_hash, merkle_root, hash, nonce, timestamp
- OTP: code hash, purpose, expiry, used_at, attempts

4.4 API Overview

- /auth/login: Authenticate and obtain JWT
- /auth/otp/request: Request OTP for transaction
- /transactions/transfer: Initiate transfer with OTP
- /accounts/{id}: Get account details and recent activity
- /chain/blocks, /chain/blocks/{index}, /chain/tx/{hash}: Explorer endpoints

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

- /admin/*: Admin utilities (metrics, reset MPIN, chain revalidation)

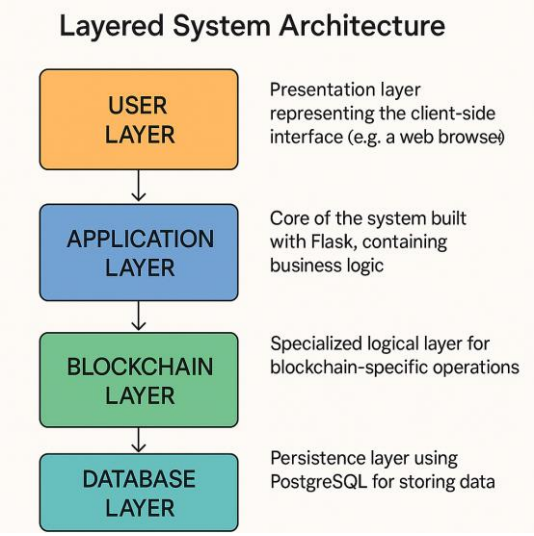

Figure 4.1: Layered System Architecture Diagram

V. SECURITY ARCHITECTURE

5.1 Threat Model (STRIDE Summary)

- Spoofing: MPIN + OTP + TLS + short-lived JWT reduce impersonation risk
- Tampering: Blockchain immutability and signed audit logs detect changes
- Repudiation: Immutable records + OTP authorization provide non-repudiation
- Information disclosure: PII kept off-chain; AES-256 at rest, TLS in transit
- DoS: Rate limiting, CAPTCHAs, and scalable stateless services
- Privilege escalation: Strict RBAC, admin MFA, least privilege for services

5.2 Two-Password Flow

- MPIN (first factor): Salted and hashed via PBKDF2-HMAC-SHA256; used for login
- OTP (second factor): One-time, short-lived code for each transaction; includes transaction details in the message; lockout after repeated failures

5.3 Cryptography

- Hashing: SHA-256 for blocks, tx, Merkle roots
- Passwords: PBKDF2-HMAC-SHA256 (future: Argon2id)
- Encryption: AES-256-GCM for sensitive data at rest
- Transport: TLS 1.2/1.3 with modern ciphers and HSTS
- Tokens: JWT (HS256), minimal claims, short TTL, refresh tokens
- Key management: KMS/HSM in production for secure storage and rotation

VI. BLOCKCHAIN IMPLEMENTATION

6.1 Block Structure

Each block includes index, timestamp, transactions, previous_hash, merkle_root, nonce, and hash. A Merkle tree summarizes transactions efficiently and supports inclusion proofs.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

6.2 Mining & Consensus (Prototype)

- Proof of Work (PoW): Miner finds a nonce so that block hash meets difficulty
- Difficulty tuning: Adjustable; low in prototype for speed
- Upgrade path: Replace PoW with permissioned consensus like PBFT/IBFT/Raft for throughput and finality

6.3 Validation & Recovery

- validate_chain():
  - Verify previous_hash linkage
  - Recompute block hash
  - Recompute Merkle root
  - Check PoW target
- Recovery workflow:
  - Quarantine from first invalid block
  - Rebuild from last valid state by replaying confirmed DB transactions
  - Admin review and approval before returning to normal

6.4 Blockchain Explorer

- Browse blocks and transactions
- Search by index or hash
- Visual integrity indicator (validation status)
- CSV/JSON export for audits

VII. IMPLEMENTATION DETAILS

7.1 Technology Stack

- Backend: Python, Flask, SQLAlchemy
- Database: PostgreSQL (prod), SQLite (dev)
- Security: PBKDF2, JWT (PyJWT), AES-256 via cryptography library
- DevOps: Docker, Gunicorn, Nginx, Pytest, CI/CD (GitHub Actions)
- Optional: Celery/RQ for robust background mining

7.2 Deployment Topologies

- Development: Single-node, SQLite, debug explorer
- Staging: Docker Compose, TLS, basic replication, UAT
- Pilot Production: Multiple stateless API nodes behind load balancer, managed PostgreSQL, KMS, backups, monitoring

7.3 Configuration Management

- ENV-driven parameters:
  - BLOCK_DIFFICULTY, BLOCK_MAX_TX, OTP_EXPIRY_SECONDS
  - PBKDF2_ITERATIONS, JWT_ACCESS_TTL, JWT_REFRESH_TTL
  - RATE_LIMITS, DATABASE_URL

7.4 Logging & Telemetry

- Structured JSON logs with correlation IDs
- Prometheus metrics: TPS, mining time, p95 latency, validation time

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

- Alerts: spikes in failed logins, chain validation failures, latency breaches, queue growth

7.5 Backup & Disaster Recovery

- PostgreSQL PITR and daily snapshots stored offsite
- Regular blockchain state backups
- Defined RPO/RTO targets and routine DR drills

VIII. MODULES & CORE FEATURES

8.1 User Management

- Registration with email verification
- Login with MPIN, rate limits, CAPTCHA on repeated failures
- Profile management (admin approvals for sensitive changes)
- MPIN reset via secure, time-limited link

8.2 Account Services

- Account opening/closure requests
- Real-time balance view
- Statement generation (e.g., monthly PDFs)

8.3 Transaction Processing

- Deposit, withdrawal (admin), and user transfers
- Idempotency keys to prevent duplicates
- Status states: PENDING → CONFIRMED → FAILED
- Reversals via compensating transactions (immutability preserved)

8.4 Security Layers

- Two-password auth, lockouts, encryption, JWT
- Non-repudiation via immutable ledger entries

8.5 Admin & Dashboard

- User and role management
- Chain health: height, difficulty, average block time
- System performance: throughput, latency, DB metrics
- Policy tuning for risk (lockout thresholds, rate limits)

8.6 Use Cases

- Retail banking for individuals
- Remittances within a permissioned consortium
- Trade finance document/state tracking
- KYC data sharing across trusted institutions

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

IX. TESTING, RESULTS & ANALYSIS

9.1 Test Strategy

- Unit tests: Block hashing, validations, service utilities
- Integration tests: End-to-end flows (API → DB → mining → explorer)
- Security tests: AuthZ checks, rate limiting, SQLi/XSS, OTP expiry/lockouts
- Performance tests: Load generation for concurrency, API latency, throughput
- UAT: Usability checks by pilot users

9.2 Sample Test Cases

- Valid/invalid login flows
- Transfer with valid OTP → confirmed after mining
- Expired/incorrect OTP → transaction blocked
- Chain tampering → validation failure and alert
- Unauthenticated access → HTTP 401

9.3 Metrics (Prototype Hardware)

- API p95 (read ops): ~250–400 ms
- Transfer API p95 (pre-mining): ~1.5–1.8 s
- Mining time (difficulty=4): ~2–5 s
- Throughput (batched): ~30–50 TPS theoretical
- Chain validation: ~0.5–0.9 s per 1,000 blocks

9.4 Analysis

- Asynchronous mining keeps UX responsive
- Difficulty and batch size offer tunable trade-offs
- Security controls (OTP, RBAC, validation) behave as intended

X. SCALABILITY & NETWORK PERFORMANCE

Bottlenecks:

- PoW is slow and non-scalable
- Single miner limits throughput
- Database write contention under load
- In-memory queues prevent horizontal scaling

Strategies:

- Stateless API servers behind a load balancer
- Read replicas for query-heavy operations
- Distributed queues (Kafka/RabbitMQ) and parallel miners
- Layer 2/off-chain techniques (state channels, batch settlement)
- Database sharding at extreme scale
- Transition to permissioned consensus (PBFT/IBFT) for high throughput and finality
- Caching with careful invalidation for non-critical reads

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

XI. REGULATORY COMPLIANCE & GOVERNANCE

KYC/AML:

- Extensible data models for KYC fields stored off-chain (encrypted)
- Risk scoring and third-party verification
- Rules-based transaction monitoring with immutable audit evidence

Privacy (GDPR/CCPA):

- PII minimized and kept off-chain
- AES-256 at rest, TLS in transit
- Right to be forgotten applied to off-chain identity; on-chain data remains pseudonymous

Standards Mapping:

- ISO 27001: Access control, logging, encryption, DR plan
- NIST CSF: Identify, Protect, Detect, Respond, Recover alignment
- NIST SP 800-63B: Meets AAL2 through MPIN + OTP
- PCI DSS (if applicable): Tokenization and segmentation patterns supported

XII. APPLICATIONS, ADVANTAGES & DISADVANTAGES

Applications:

- Retail banking ledger integrity and transparency
- Cross-border payments within a consortium
- Trade finance lifecycle tracking
- Syndicated loans and shared ownership records
- Asset tokenization (future extension)

Advantages:

- Strong integrity and non-repudiation
- Real-time, automated auditability
- Reduced reconciliation costs across parties
- Faster settlement in permissioned networks
- Higher customer trust via transparency
- Programmable automation with smart contracts

Disadvantages:

- Lower single-ledger throughput vs centralized DBs
- Irreversibility requires compensating transactions
- Extra step (OTP) adds user friction
- Governance is complex in multi-organization networks
- Ever-growing chain requires storage strategies

XIII. CHALLENGES, RISKS & MITIGATIONS

- Throughput and latency:
  - Mitigation: Permissioned consensus, batching, async processing
- Key management:

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

    - Mitigation: KMS/HSM, strict access control, rotation policies
- Forks and synchronization:
    - Mitigation: BFT consensus, clear fork rules, frequent validation
- User onboarding:
    - Mitigation: Intuitive UI, clear confirmations, threshold-based OTP prompts
- Legacy integration:
    - Mitigation: API-first design, middleware adapters
- Cryptographic agility:
    - Mitigation: Modular crypto to enable post-quantum upgrades when ready

XIV. FUTURE ENHANCEMENTS & ROADMAP

Phase 1 (0–6 months):

- Replace PoW with PBFT/IBFT
- Integrate KMS/HSM
- Improve CI/CD with SAST/DAST
- Mobile apps with biometrics and push-OTP

Phase 2 (6–12 months):

- Smart contracts for recurring payments, escrow, interest

Phase 3 (12–24 months):

- Real-time fraud detection via ML
- Zero-knowledge techniques for privacy
- Interledger connections for cross-system transfers

Phase 4 (24+ months):

- CBDC/stablecoin integration
- Decentralized identity (DID) for user control
- Post-quantum readiness

XV. COMPARISON WITH EXISTING SYSTEMS

Traditional Online Banking:

- Centralized, mutable database; high privacy but opaque to users
- Fast internal, slow cross-border; periodic manual audits
- Reversals possible by institution

Public Crypto Wallets:

- Fully decentralized and transparent; user-managed keys
- Finality with no reversals; high privacy through pseudonyms
- Fees and throughput vary with network conditions

This Project (Hybrid):

- Off-chain state + on-chain proofs; permissioned visibility
- Fast user feedback, minutes to confirmation on private chain

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 8, August 2025

DOI: 10.17148/IARJSET.2025.12816

High auditability, reduced reconciliation; privacy preserved by keeping PII off-chain

XVI. CONCLUSION

This work shows a practical way to raise the security and trust of online banking by combining a traditional database with a private blockchain. Transactions are quick for users, yet permanently anchored to an immutable ledger for auditors and regulators. The two-password flow (MPIN + OTP) targets one of the biggest real-world risks: credential compromise leading to unauthorized transfers. The prototype demonstrates core concepts—block creation, mining, validation, and explorer tooling—using Flask, PostgreSQL, and a custom PoW chain. Tests confirm responsiveness thanks to asynchronous design, and results underscore the need to move from PoW to a permissioned consensus for scale and energy efficiency. Overall, this hybrid approach offers a realistic, evolutionary path for banks to adopt blockchain benefits without sacrificing usability, privacy, or operational control.

REFERENCES

[1]. Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf
[2]. Buterin, V. (2014). A Next-Generation Smart Contract and Decentralized Application Platform. https://ethereum.org/en/whitepaper/
[3]. Hyperledger Fabric Documentation. https://hyperledger-fabric.readthedocs.io/
[4]. R3 Corda Documentation. https://docs.r3.com/
[5]. ConsenSys. Quorum: Enterprise Ethereum. https://consensys.io/quorum
[6]. NIST SP 800-63B: Digital Identity Guidelines. https://doi.org/10.6028/NIST.SP.800-63b
[7]. NIST SP 800-132: Password-Based Key Derivation. https://doi.org/10.6028/NIST.SP.800-132
[8]. NIST Cybersecurity Framework v1.1. https://doi.org/10.6028/NIST.CSWP.04162018
[9]. ISO/IEC 27001: Information security management systems — Requirements
[10]. Castro, M., & Liskov, B. (2002). Practical Byzantine Fault Tolerance. ACM TOCS, 20(4), 398–461.
[11]. RFC 6238: TOTP: Time-Based One-Time Password Algorithm. https://doi.org/10.17487/RFC6238