Master's Thesis

Design and Implementation of Blockchain-based Central Bank Digital Currency

본 논문작성자는 한국정부초청장학금 (Global Korea Scholarship)을 지원받은 장학생임

Sajan Maharjan (머허르전 사전)

Department of Computer Science and Engineering

Pohang University of Science and Technology

2020

블록체인 기반 중앙은행 디지털 화폐의 설계 및 구현

Design and Implementation of Blockchain-based Central Bank Digital Currency

Design and Implementation of Blockchain-based Central Bank Digital Currency

by

Sajan Maharjan
Department of Computer Science and Engineering
Pohang University of Science and Technology

A thesis submitted to the faculty of the Pohang University of Science and Technology in partial fulfillment of the requirements for the degree of Master of Science in the Computer Science and Engineering

Pohang, Korea
06. 27. 2020
Approved by
James Won-Ki Hong (Signature)
Academic advisor

Design and Implementation of Blockchain-Based Central Bank Digital Currency

Sajan Maharjan

The undersigned have examined this thesis and hereby certify that it is worthy of acceptance for a master's degree from POSTECH

06. 30. 2020

Committee Chair	James Won-Ki Hong	(Seal)
Member	Jong Kim	(Seal)
Member	Jongsoo Woo	(Seal)

MCSE 머허르전 사전. Sajan Maharjan
20182095
Design and Implementation of Blockchain-based Central Bank Digital Currency,
블록체인 기반 중앙은행 디지털 화폐의 설계 및 구현
Department of Computer Science and Engineering , 2020, 67p, Advisor : James Won-Ki Hong. Text in English.

ABSTRACT

The emergence of Bitcoin and henceforth the blockchain technology have inspired the development of thousands of altcoins and cryptocurrencies. Everyday, thousands of transactions worth millions of dollars are being conducted on public blockchain platforms using these cryptocurrencies and altcoins. While such public blockchains provide the features of peer-to-peer, immutable, disintermediated and decentralized record of transactions, they are often less regulated and susceptible to illegal transactions. Due to the lack of regulations and control surrounding different cryptocurrencies, banks and governments are concerned over the valuation of such cryptocurrencies and their activities. Also, on the other hand, banks and governments are also keen to learn the potential use of blockchain technology in legacy banking services for their benefits. Particularly, banks are looking forward to issuing digital currencies that are well regulated and controlled. This has led to the concept of central bank digital currency (CBDC).

In this thesis, the author explains the concepts of CBDC, its motivation and different types of CBDC. The author also presents the design and implementation of blockchain-based direct CBDC model, indirect CBDC model and hybrid CBDC model. Lastly, we also discuss the validity of our work and compare the different CBDC model implementations.

Contents

List of Tables

List of Figures

I. Introduction

1.1 Motivation

Bitcoin [1] was introduced to the world in 2009 by its anonymous creator Satoshi Nakamoto. Bitcoin, a peer-to-peer decentralized electronic payment system, allows trustless payments to be made between parties without the need for a financial intermediary. Payments made in Bitcoin are cryptographically secure by the use of public key cryptography and verified by a distributed set of network participants. In Bitcoin, transactions are included into a block, such that blocks are linked with their previous blocks, forming the history as a chain of blocks and providing tamper-proof immutable record of transactions.

Interestingly, the coinbase message in the genesis block of Bitcoin contains the message - "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks", referring to a newpaper article published in the Times, providing a grim reminder of the financial crisis of 2008 and how the financial system had failed [2, 3]. Since, its emergence in 2009, Bitcoin has inspired the blockchain [4] technology and led to the development of thousands of altcoins [5]. Recent developments in the blockchain and cryptocurrency space has brought upon novel, innovative and revolutionary services in the financial area.

There has been a gradual increase in the number of participants in the cryptocurrency ecosystem as evident by the increase in the number of daily confirmed

transactions in Bitcoin (Fig. 1.1) and Ethereum [6] (Fig. 1.2). Another metric to evaluate the growth of the cryptocurrency ecosystem is the fiat value at which cryptocurrencies are exchanged. It is possible to surmise that initial doubters of the blockchain technology have turned to avid aficionados amid the surge in the price of Bitcoin, Ethereum and other cryptocurrencies. While new entrants in the financial system are enjoying the novelty and innovation offered by blockchain technology, incumbents in legacy financial institutions like banks and governments are growing concerned over lack of authority, issues on monetary sovereignty and financial regulations on such cryptocurrencies.

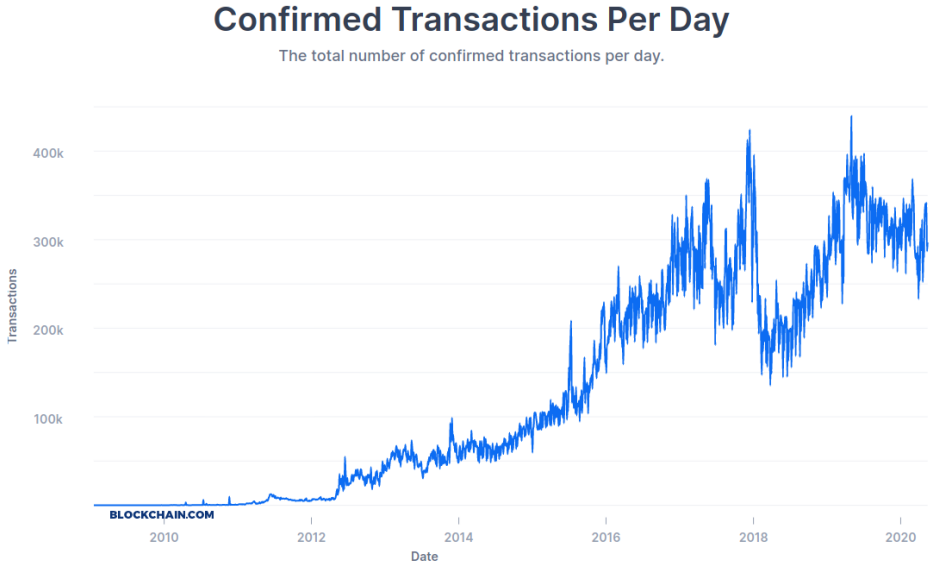

Figure 1.1: Number of daily confirmed transactions in the Bitcoin network (source: blockchain.com)

The exponential growth of blockchain-based financial services, and the banks'

Figure 1.2: Number of daily transactions in the Ethereum network (source: etherscan.io)

inability to govern them is not their only problem at hand. Fiat currencies printed by the central banks have their own inherent problems. Central bank money in the form of cash is expensive to produce and circulate. It is reported that the Bank of Korea(BOK) spends 20 KRW to yield a coin worth 10 KRW. The BOK also reports that bills worth 20 million KRW are subject to damage and 60 billion KRW are invested in re-issuance costs annually [7]. Also, the US Federal Reserve spends twice as much to produce coins worth a penny [8]. For the financial year of 2020, it has also allocated a budget of $827.7 million in costs for printing bills and engraving coins [9]. Not just the production of fiat cash, its distribution is expensive as well. Cash is moved from one place to another by using armored vehicles guarded by multiple personnel. Other problems with fiat cash are physical storage requirement, susceptibility to loss (eg. theft, fire,

damage, etc.) and lack of transparency (as cash is used in money-laundering, financing of terrorism and other illegal activities).

Cash is still the only central bank liability available to the public, however, cash is being less used in the current payment landscape. The public's preference lies in using credit cards and mobile applications for payments rather than holding cash. The payments market is moving towards a cashless society as seen in Sweden and China. In Sweden, retail payments are carried out using the application Swish [10]. In China, Alibaba and Tencent have dominated the payments market with the widespread use of AliPay [11] and WeChat Pay [12] respectively. Although Swish was created by a consortium of Nordic banking institutions, AliPay and WeChat Pay are privately owned. Such private payments service providers (PSPs) are less regulated compared to banks, and are exposed to fungibility and liquidity risks.

The above mentioned are the most notable problems that central banks and governments are facing currently. Central banks and governments are exploring the blockchain technology and pondering upon the launch of a digital currency with the intention of regaining control and maintaining regulations in the financial market.

1.2 Birth of Central Bank Digital Currencies

The long-standing interest of central banks in cryptocurrencies and digital payment services and their desire to launch a virtual electronic currency has led to the birth of central bank digital currency (CBDC). CBDC can be simply un-

derstood as a digital currency issued by a central bank. Due to the proliferation of cryptocurrencies and mobile payment applications as well as the recent announcement of Facebook launching its own cryptocurrency, Libra [13], there has been a surge of research interest in CBDC.

Amid the rising interest in CBDC, in this thesis, we introduce the concepts related to CBDC and, present the design and implementation of different models of CBDC based on blockchain platforms. Our choice of interest for the blockchain platform in the implementation of CBDC application lies in using Quorum blockchain. The remainder of the thesis is organized as follows. Chapter II describes the current literature on CBDC. Chapter III presents the design of various CBDC models based on blockchains. Chapter IV explains the implementation of the proposed designs by using Quorum blockchain environment. Chapter V discusses the validity of our work, presents the pros and cons of our implementation and compares different models of CBDC implementation with the Quorum blockchain environment. Finally, Chapter VI concludes the thesis with possible future work.

II. Background and Related Work

2.1 Background

CBDCs are electronic form of money issued by a central bank. CBDCs are monetary value stored electronically (digitally, or as an electronic token) that represents liability of the central bank and can be used to make payments [14]. Like cash which is a physical form of money issued by central banks, a CBDC in its electronic form serves as a method of payment, medium of exchange and a store of value. The money flower [15] portrays different forms of money currently in use and highlights the major features of CBDC (Fig. 2.1).

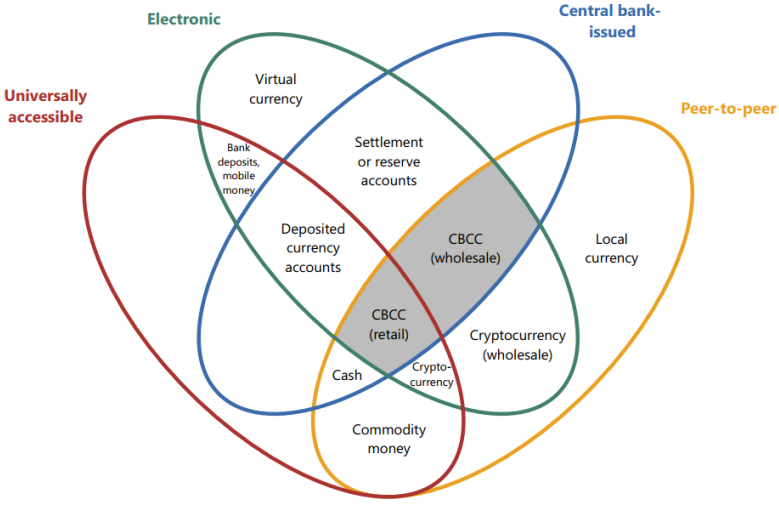

Figure 2.1: Money Flower. Shaded Area represents CBDCs

In the money flower representation, CBDCs are defined as an electronic form of central bank-issued money that allows peer-to-peer transfers which may or may not be universally accessible to the general public. This stimulates the idea that different forms of CBDC can be possible. Beside accessibility, other factors such as presence of intermediary, level of anonymity, possibility of cross-border transactions, interest-bearing ability, etc. define different forms of CBDC.

Based on accessibility, CBDCs are classified as retail CBDC and wholesale CBDC. CBDCs that are accessible by the general public and which are used in daily payment activities are called retail CBDC. They are also known as "general-purpose" CBDC. Usually, small-valued transactions are involved in retail CBDC. Wholesale CBDCs are CBDCs that are limited to financial institutions like commercial banks. Usually, large-valued transactions are involved in wholesale CBDC. It can be argued that wholesale CBDC is already in existence in the form of central bank electronic settlement accounts offered to commercial banks and other financial institutions.

Based on the presence of an intermediary, CBDCs can be categorized into direct CBDC and indirect CBDC. Under direct CBDC, digital currency can be issued and distributed from issuer to end-users directly, without the need of an intermediary. Such forms of CBDC usually incorporate overhead on the central banks as all operations and management activities are to be handled by it. Indirect CBDC (*aka* two-tier CBDC) features the presence of a financial intermediary like exchanges, banks, etc. that interface end-users. The presence of such intermediaries eases the burden of responsibilities on central banks by delegat-

ing certain responsibilities to the intermediaries. Hybrid CBDC is an initiative to CBDC development in which central banks collaborate with private PSPs to offer central bank-backed digital money in widely used mobile payment applications. Table 2.1 highlights the differences between direct, indirect and hybrid CBDCs.

Table 2.1: Direct CBDC vs Indirect CBDC vs Hybrid CBDC.

Features	Direct CBDC	Indirect CBDC	Hybrid CBDC
Presence of Intermediary	No	Yes	Yes
Claim on Central Bank	Yes	No	Yes
Division of Responsibilities	No	Yes	Yes
Deferred Settlements	No	Yes	Yes

Based on the level of anonymity, CBDCs can be separated as token-based CBDC and account-based CBDC. Token-based CBDC uses cryptographic tokens held in digital wallets like in Bitcoin. Ownership of tokens is verified by proving the ownership of digital wallet addresses (i.e., private keys). Account-based CBDC holds user balances similar to accounts held at banks. Ownership of users' account can be verified by presenting related documents (like bank book, social security card, etc.). Token-based CBDC offers greater anonymity and has fewer user-identity constraints compared to account-based CBDC. CBDCs are also classified as cross-border CBDC or domestic CBDC based on the possibility of cross-border transactions. Similar to cash deposits at banks offering interest, CBDCs may also be labelled as interest-bearing or not when deposited at banks.

Additional to the motivation behind the development of CBDCs explained in section 1.1, the following are some of the reasons for banks' and governments' initiative to CBDC: [14]

1. Improve central bank seigniorage revenue.

2. Reduce the lower bound on interest rates.

3. Support unconventional monetary policy.

4. Reduce aggregate risk and improve financial stability.

5. Increase contestability in payments.

6. Promote financial inclusion.

7. Inhibit criminal activity.

2.2 Related Work

Consensys reports that according to the Bank for International Settlements, among a surveyed group of central banks, over 70% of central banks are looking at issuing a digital currency on a blockchain platform [16]. These central banks have been approaching CBDC through different tests, pilots or even just feasibility analysis projects. Engert and Fung point out the motivation behind CBDCs and their implications on the financial system [14]. The Bank for International Settlements presents a taxonomy on different forms of CBDC, design considerations and challenges in implementation [17]. The International Monetary Fund (IMF) categorizes different forms of money currently in use in the global payment landscape [18]. The IMF argues that e-money offered by third-party private PSPs are the most widely used forms of payment and that banks must either cooperate

or compete with them to dominate the payments market. In the paper, IMF introduces the concept of hybrid CBDC whereby central banks and PSPs collaborate together to issue CBDC. Likewise, VISA has recently registered a cryptocurrency patent similar to the concept of hybrid CBDC in which private transaction processing networks (like VISA) collaborate with central banks in replacing existing physical fiat currency for a digital currency [19]. Additionally, the World Economic Forum provides a toolkit for guiding policy-makers to undertake/deploy a particular form of CBDC given different design requirements [20].

While there are numerous undertakings on surveying the feasibility, application benefits, design considerations and challenges to the development of CBDCs, there are only a handful of papers describing the technical details of implementation. Danezis and Meiklejohn have proposed RSCoin, a cryptocurrency framework in which central banks maintain complete control over the monetary supply, but rely on a distributed set of authorities or mintettes to prevent double-spending [21]. RSCoin builds upon the limitations of Bitcoin - wasteful hashing and lack of governance, and proposes a centralized control of monetary supply henceforth increasing the scalability and stability in the system. Wust et al. have proposed PRCash, a blockchain based currency with central governance which makes use of zero-knowledge proofs and homomorphic encryption to hide the details of transactions [22]. As opposed to RSCoin which focuses on scalability of transaction processing, PRCash focuses on user anonymity while also guaranteeing governance and regulation.

Other implementations of CBDC include different undertakings by banks

across the globe. Since 2016, the Bank of Canada has undertaken Project Jasper with the aim of understanding how distributed ledger technology(DLT) could transform the future of payments in Canada [23]. In its initial phase, Project Jasper implemented a wholesale CBDC architecture based on DLT while in the latter phases, it is now exploring the possibilities of retail CBDC. Similarly, Project Ubin is an undertaking by the Monetary Authority of Singapore(MAS) to explore the use of blockchain and DLT for clearing and settlement of payments and securities [24]. Other notable central bank initiatives include Project Stella [25], Project Inthanon [26], Project Khokha [27] and Project Bakong [28]. Also, Don and Alex Tapscott have discussed how the digital dollar can be impactful in the face of a global pandemic like COVID-19 in a webinar with J. Christopher Giancarlo [29].

As there have been arguments that wholesale CBDC is already in existence in the form of central bank electronic settlement accounts as well as different blockchain-based undertakings by central banks around the globe, thus the focus of this thesis is on retail CBDC. In this thesis, we present the design of different blockchain-based CBDC models i.e., direct CBDC model, indirect CBDC model and hybrid CBDC model. Furthermore, we implement the three models— direct model, indirect model and hybrid model, by using the Quorum blockchain environment. The details of the proposed model is discussed in the following chapters.

III. Design

CBDCs do not necessarily have to be designed by using blockchain and DLT-based solutions, as centralized database solutions are also possible. However, blockchain-based CBDCs offer greater transparency, cryptographic security and robustness compared to centralized database solutions. Additionally, blockchain-based CBDCs offer the feature of programmability of money via smart contracts [30]. Hence, the proposed CBDC models have been designed using blockchain environments. Understanding the roles of different participants and entities in a system is a pre-requisite for efficient functioning of any system. Thus, we first list out the roles and requirements of stakeholders in the system and then design the system which can be later implemented effectively.

3.1 Requirements

Different CBDC systems try to emulate banking system functionalities by providing well-controlled and regulated digital currency. Thus, deriving from analogy, the major participants in any CBDC system resembles the participants in a banking system i.e., end-users, central banks, commercial banks and PSPs. In this section, we define the roles and requirements of these entities specific to each CBDC model. There may be other indirect participants in the banking system, e.g., regulators or governmental bodies who mainly seek to verify and validate the data in the system. They occasionally interact with the system and are not

active participants in the system, so we skip defining their roles. However, our system ensures that their request for data access is addressed. Also the general non-functional requirements of any CBDC system are as follows:

- Payment system should be simple and easy to use.
- Transaction finality should be near-immediate.
- Ensure high-availability of the system.
- Offer API access to third party applications.
- Issued digital currency should be stable and pegged at a specified rate.

3.1.1 Requirements for Direct CBDC Model

Under the direct CBDC model, central banks and end-users are the only active participants. Their roles and requirements are listed below:

Functional Requirements of Central Banks

- Initialize or setup blockchain.
- Issue digital currency regularly (fixed amount in a fixed period of time).
- Create wallet address for end-users and distribute digital currency.
- View transactions of wallet addresses.
- Burn digital currency owned by it, as per necessity.

Non-Functional Requirements of Central Banks

- Monetary supply should be well regulated.
- Deploy multiple nodes throughout the country to enhance security.

Functional Requirements of End-Users

- Access user wallets to view current balance.
- Conduct P2P transfer among different users.
- View transactions corresponding to their accounts or wallet addresses.

Non-Functional Requirements of End-Users

- Ensure safety of private keys of corresponding wallet addresses.
- Provide relevant information or documents to stakeholders when required.

Commercial bank roles are not explicitly defined under direct CBDC model. However, commercial banks are reduced to roles in which they may operate implicitly as an end-user holding large amount of digital currency, offering P2P lending services.

3.1.2 Requirements for Indirect CBDC Model

The indirect CBDC model comprises of central banks, commercial banks and end-users, and its operation is similar to the current banking environment. The expected roles of the participants are as follows:

Functional Requirements of Central Banks

- Initialize or setup blockchain.
- Issue digital currency regularly (fixed amount in a fixed period of time).
- Create wallet addresses for banks and distribute digital currency.
- View transactions of wallets addresses under its domain.
- Burn digital currency owned by it, as per necessity.

Non-Functional Requirements of Central Banks

- Monetary supply should be well-regulated.
- Assessment of digital currency to lend to commercial banks.

Functional Requirements of Commercial Banks

- Create wallet addresses for end-users under its domain.
- Distribute digital currency to end-users under its domain
- View transactions of wallets addresses under its domain
- Offer banking services via smart contracts e.g., loans, deposits, etc.

Non-Functional Requirements of Commercial Banks

- Maintain a node locally to minimize transaction propagation latency.
- Settlement of transactions with central banks at specified period.

Functional Requirements of End-Users

- Access user wallets to view current balance.
- Conduct P2P transfer among users.
- View transactions corresponding to their wallet addresses.

Non-Functional Requirements of End-Users

- Ensure safety of private keys of corresponding wallet addresses.
- Provide relevant information or documents to stakeholders when required.

3.1.3 Requirements for Hybrid CBDC Model

Under the hybrid model, the main participants are central banks, end-users and private PSPs. Commercial banks can also co-exist in the hybrid model but their roles are not as significant as PSPs. Commercial banks will exhibit similar roles of digital currency distribution in Hybrid model as well. The roles and requirements of participants are listed below:

Functional Requirements of Central Banks

- Create or initialize blockchain
- Issue digital currency regularly (fixed amount in a fixed period of time).
- Distribute digital currency to intermediaries and PSPs.

Non-Functional Requirements of Central Banks

- Monetary supply should be well-regulated.
- Conduct due diligence checks on PSPs and intermediaries.

Functional Requirements of Commercial Banks

- Create wallet addresses for end-users under its domain.
- Distribute digital currency to end-users under its domain
- View transactions of wallets addresses under its domain
- Offer banking services via smart contracts e.g., loans, deposits, etc.

Non-Functional Requirements of Commercial Banks

- Maintain a node locally to minimize transaction propagation latency.
- Settlement of transactions with central banks at specified period.

Functional Requirements of Payment Service Providers

- Offer payment services to end-users via payment applications.
- Distribute CBDC token to end-users through payment applications.

Non-Functional Requirements of Payment Service Providers

- Facilitate financial services in the network.
- Comply with regulations sets by central banks.

Functional Requirements of End-Users

- Access user wallets to view current balance.
- Conduct P2P transfer among users.
- View transactions corresponding to their wallet addresses

Non-Functional Requirements of End-Users

- Ensure safety of private keys of corresponding wallet addresses.
- Provide relevant information or documents to stakeholders when required.

All of the three models share common functionalities i.e., digital currency is minted by central banks, distributed by commercial banks and PSPs, and transactions are conducted by end-users. There are subtle differences in terms of presence of intermediaries and offering of secondary services. Particularly, in the case of hybrid CBDC, the roles and requirements of entities are not clearly defined. For example, either central banks or private PSPs can initialize the blockchain. Likewise, intermediaries like commercial banks maybe involved in the payment system or could be simply ignored. While the concept of hybrid CBDC model is relatively new, however in this chapter we present the design of all three models.

3.2 Design

3.2.1 Design of Direct CBDC Model

Based on the requirements listed in section 3.1.1, we designed a model for direct CBDC architecture (Fig. 3.1). Under the proposed design, the direct CBDC model is setup as follows:

1. Central bank initializes the blockchain platform and issues digital currency. Transactions are recorded on the blockchain.

2. Central bank generates key-pairs for end-users and transfers digital currency to end-users. Transactions are recorded on the blockchain.

3. End-users conduct P2P transfers among themselves. Transfer transactions are recorded on the blockchain.

4. Regulatory bodies and third-party applications may access the blockchain for verification, regulatory purposes, etc. when required.

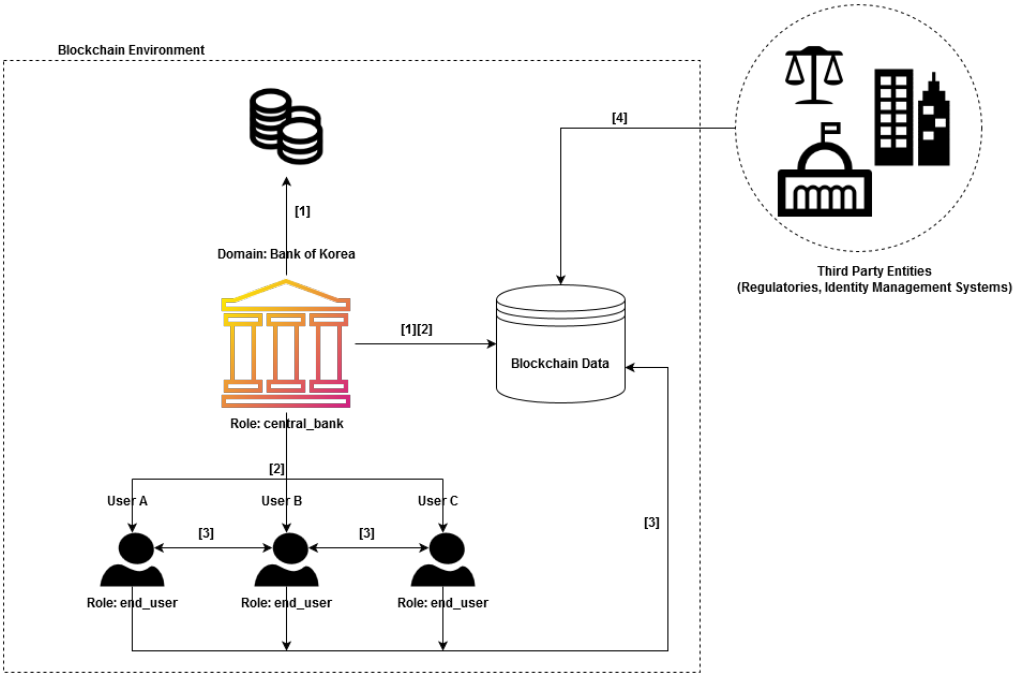

Figure 3.1: Proposed design of blockchain-based direct CBDC model

The sequence flow diagram better explains the step-by-step process of token issuance, user account address creation and transfer of balance between end-users (Fig. 3.2).

3.2.2 Design of Indirect CBDC Model

On the basis of requirements listed in section 3.1.2, we designed a model for indirect CBDC architecture (Fig. 3.3). Under the proposed design, the indirect CBDC model is setup as follows:

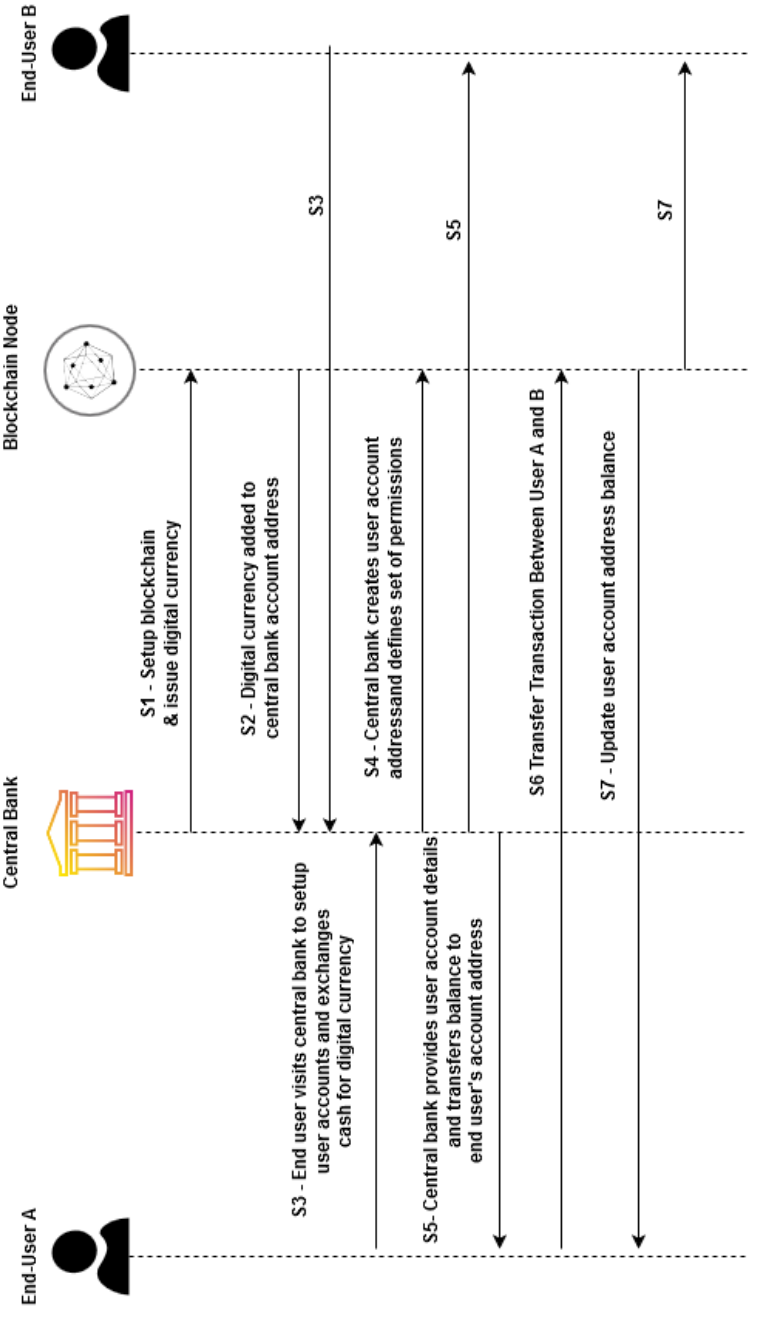

Figure 3.2: Sequence flow diagram while issuing CBDC tokens in direct model

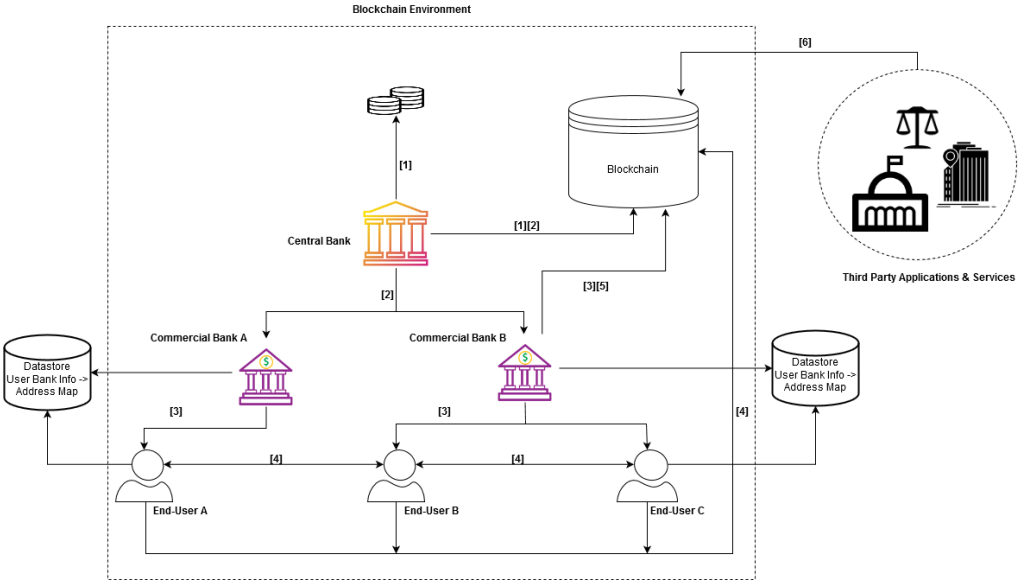

Figure 3.3: Proposed design of blockchain-based indirect CBDC model

1. Central bank initializes blockchain and issues digital currency. Transactions are recorded on the blockchain.

2. Commercial banks operate individual nodes. Central bank distributes digital currency to commercial banks. Transactions are recorded on the blockchain.

3. Commercial banks create wallet addresses for end-users and distributes digital currency. Transactions are recorded in the blockchain.

4. End-users conduct P2P transfer among themselves. Transfer transactions are recorded on the blockchain.

5. Commercial banks deploy smart contracts on the blockchain as an offering of financial services. e.g., deposits, loans, etc.

6. Regulatory bodies and third-party applications may access the blockchain (via APIs) for verification, regulatory purposes, etc. when required.

The sequence flow diagram better explains the step-by-step process of token issuance, user account address creation, transfer of balances and other service offerings of commercial banks (Fig. 3.5). The innovation offered by the proposed design is the introduction of financial services via smart contracts. Smart contracts function in tandem with cryptocurrencies. Thus, with the initiative of central-bank led CBDCs, commercial banks can offer financial services like interest on deposits, installment services, loans, etc. by deploying smart contracts on top of the blockchain-based CBDC platform (Fig. 3.4).

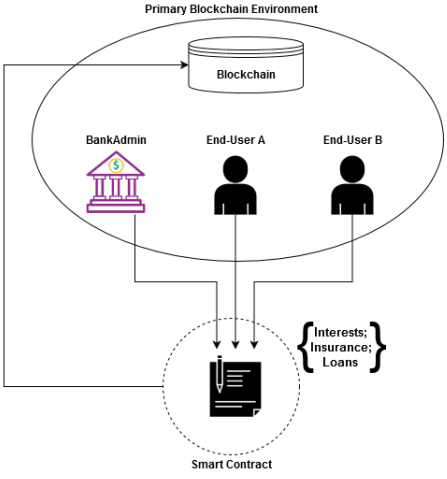

Figure 3.4: Financial services offered by commercial banks via smart contracts

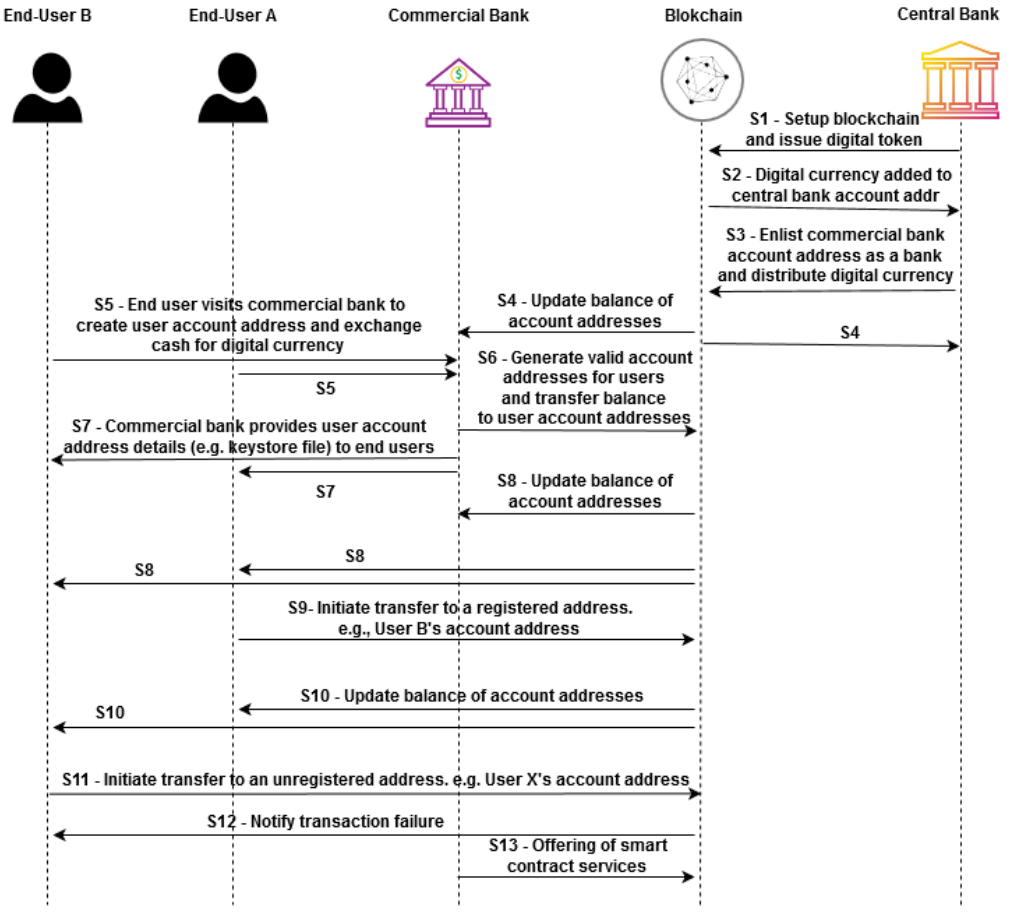

Figure 3.5: Sequence flow diagram while issuing CBDC tokens in indirect model

3.2.3 Design of Hybrid CBDC Model

On the basis of requirements listed in section 3.1.3, we designed a model for hybrid CBDC architecture (Fig. 3.6). Under the proposed design, the hybrid CBDC model is setup as follows:

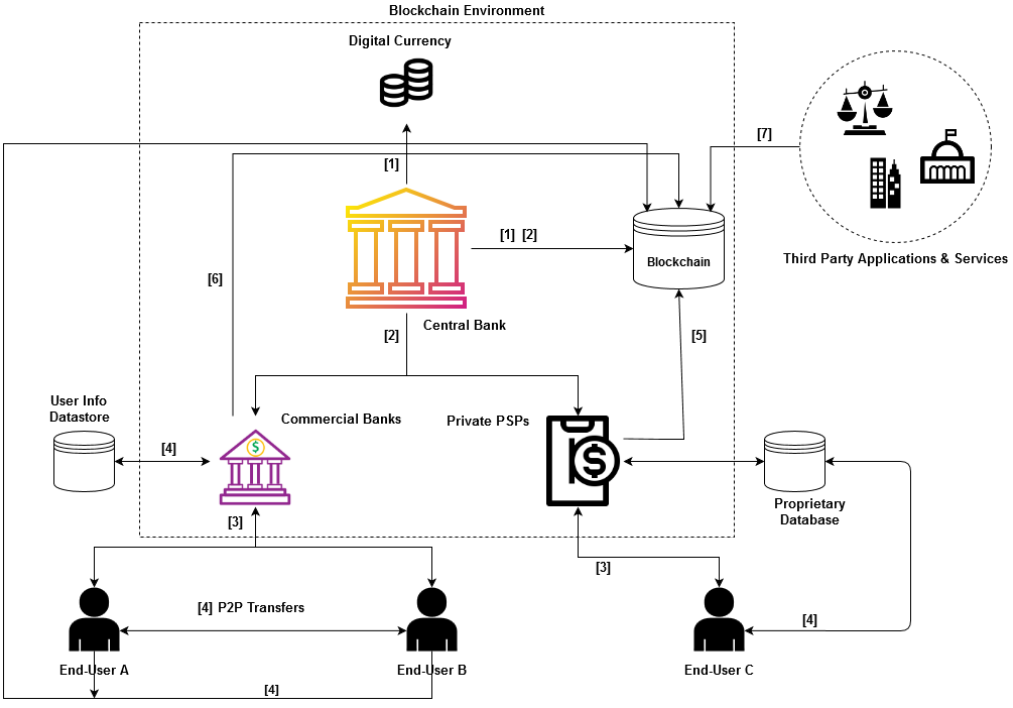

Figure 3.6: Proposed design of blockchain-based hybrid CBDC model

1. Central bank initializes the blockchain and issues digital currency. Transactions are recorded on the blockchain.

2. Central bank distributes digital currency to commercial banks and private PSPs. Transactions are recorded into the blockchain.

3. Commercial banks and private PSPs distribute digital currency to the end-users through proprietary wallets and applications.

4. End-users conduct P2P transactions among themselves through payment applications provided by PSPs and wallet addresses verified by commercial banks. Transactions initiated by using payment application of PSPs are recorded off-chain while transactions from verified wallet addresses provided by commercial banks are sent to blockchain.

5. Private PSPs verify and process transactions. PSPs report aggregate transactions to the central bank by submitting transactions to the blockchain at periodic intervals.

6. Commercial banks offer additional financial services by deploying smart contracts on the blockchain platform.

7. Regulatory bodies and third-parties may access the blockchain (via APIs) for verification, regulatory purposes, etc. when required.

The sequence flow diagram depicts the entire process in the hybrid CBDC model (Fig. 3.7). Clearly, there is a separation of concern and operation in hybrid model. Unlike direct and indirect model, end-users' transactions initiated from payment applications provided by PSPs are not reflected in the blockhain but stored off-chain in proprietary databases owned by the PSPs. Also, additional service offerings by the service provider may be specific to the vendor and not recorded on the blockchain. However, it is required of the PSPs to report on the use of CBDC tokens at periodic interval to the central bank.

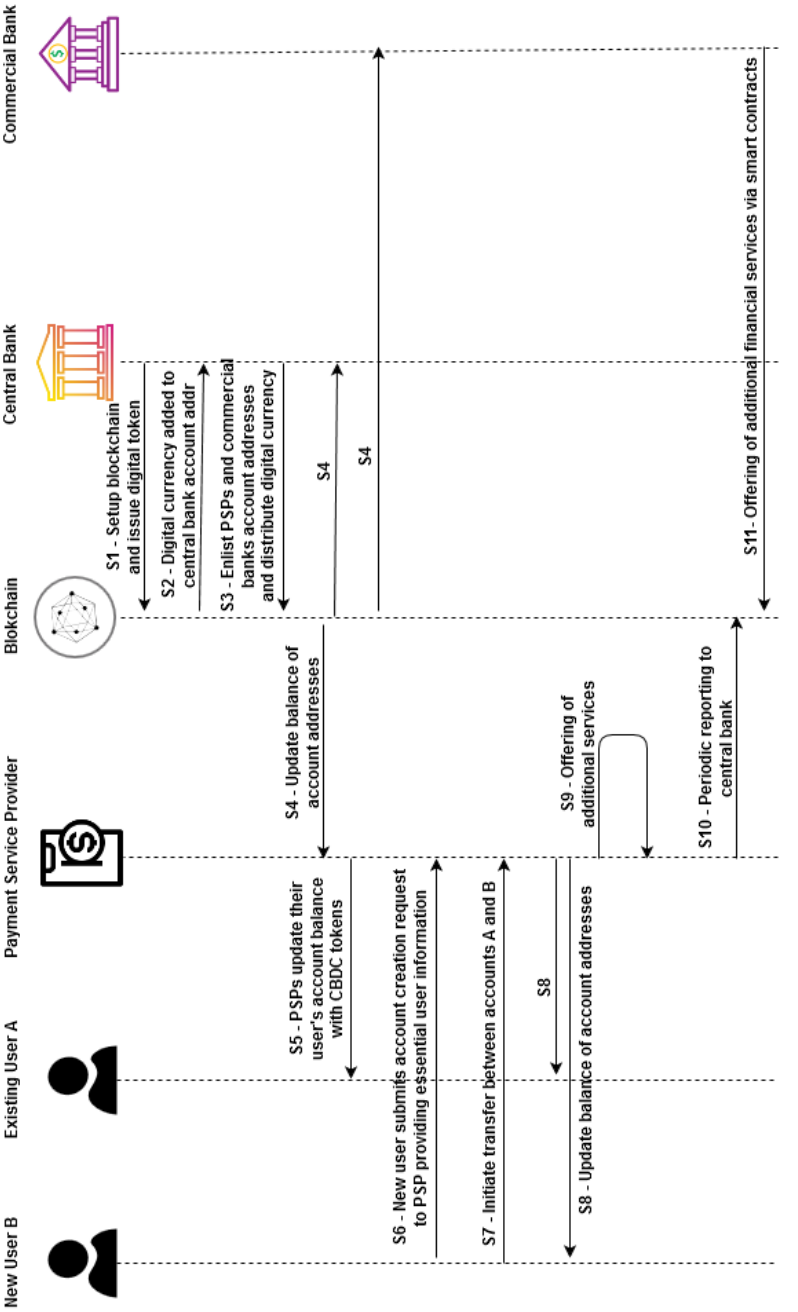

Figure 3.7: Sequence flow diagram while issuing CBDC tokens in hybrid model

Multiple nodes will be required to be deployed in blockchain environments for data redundancy. When multiple nodes are distributed geographically, users (in direct and indirect model) and PSPs (in hybrid model) can access the nearest node and send transactions via user interfaces, e.g., a web application. The blockchain platform will be responsible for providing APIs for blockchain interaction (Fig. 3.8).

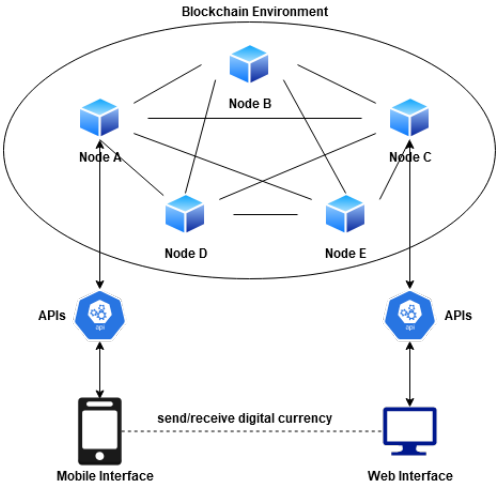

Figure 3.8: User interaction with the blockchain via web or mobile apps

IV. Implementation

Initially, we explored different blockchain initiatives applicable to CBDC development. Particularly, among the projects under HL foundation, HL Iroha [31] as a permissioned blockchain environment offers easy management of digital assets and identities. Iroha is being deployed in applications relating to interbank settlements, payment systems, logistics and CBDCs. The Iroha blockchain comes with built-in objects like accounts, assets and domains. Different methods (commands and queries) are defined on these objects such as create asset, add/subtract asset quantity, transfer assets, create domain, create accounts, view list of transactions, view list of accounts, etc. The privilege to execute a command or a query is termed as a permission. Furthermore, Iroha allows custom definition of roles (a set of permissions) that can be applied to different users in the network. Iroha also offers its APIs for Python, Java, Javascript, Android and iOS programming languages. Despite all these offerings in functions relating to CBDC development, customizability in Iroha is limited as user-defined functions cannot be implemented yet.

Likewise, Quorum [32] is a private/permissioned blockchain based on the official Go implementation of the Ethereum protocol. Quorum was developed by J.P. Morgan as an initiative for the adoption of blockchain among financial industries. Quorum is an enterprise version of Ethereum developed specifically for financial applications. Quorum is being used in supply chain management,

health, insurance, business and banking industries. Quorum achieves data privacy through the introduction of "private" transactions type. Contrary to Ethereum in which consensus is established through probabilistic algorithms like Proof-of-Work, consensus is established among the distributed nodes through deterministic algorithms like I-BFT and RAFT. This guarantees faster block generation time and immediate transaction finality in Quorum. Also, the feature of smart contract execution in Quorum allows building custom program logic into blockchain applications. Therefore, we chose the Quorum blockchain environment for the implementation of the three CBDC models due to its offering of smart contracts functionality and a private permissioned network. In this chapter, we present prototype implementations of direct, indirect and hybrid CBDC models based on the Quorum blockchain. Since this is only a prototype implementation, not all functions have been realized.

4.1 Implementation of Direct CBDC Model

4.1.1 Implementation Architecture

The details of implementation for direct CBDC model using the Quorum blockchain network is shown in Fig. 4.1. Under the direct CBDC model, central banks and end-users are the only major participants in the system. Initially, central bank writes a smart contract containing ERC20 token functionalities such as create token, add new tokens, transfer tokens, burn tokens, etc. and deploys it to the private Quorum network. End-users and central bank interact with the blockchain by sending requests via HTML forms hosted by a web-application

server. The web-server interacts with the blockchain through the Web3.js [33] library. Web3.js is a collection of libraries which allows interaction with a local or remote ethereum node, using an HTTP or IPC connection. Since Quorum is based on Ethereum and supports web3 interactions for public transactions, we used web3 library in our implementation. Users' HTTP requests for transactions and blockchain variables are handled by Web3.js. The Web3.js library reads and writes data to and from the blockchain node, and response is returned to the user via web interface. Wallet addresses are used to conduct transactions in the given network. For easy management of wallet addresses, we used Metamask [34] in our implementation.

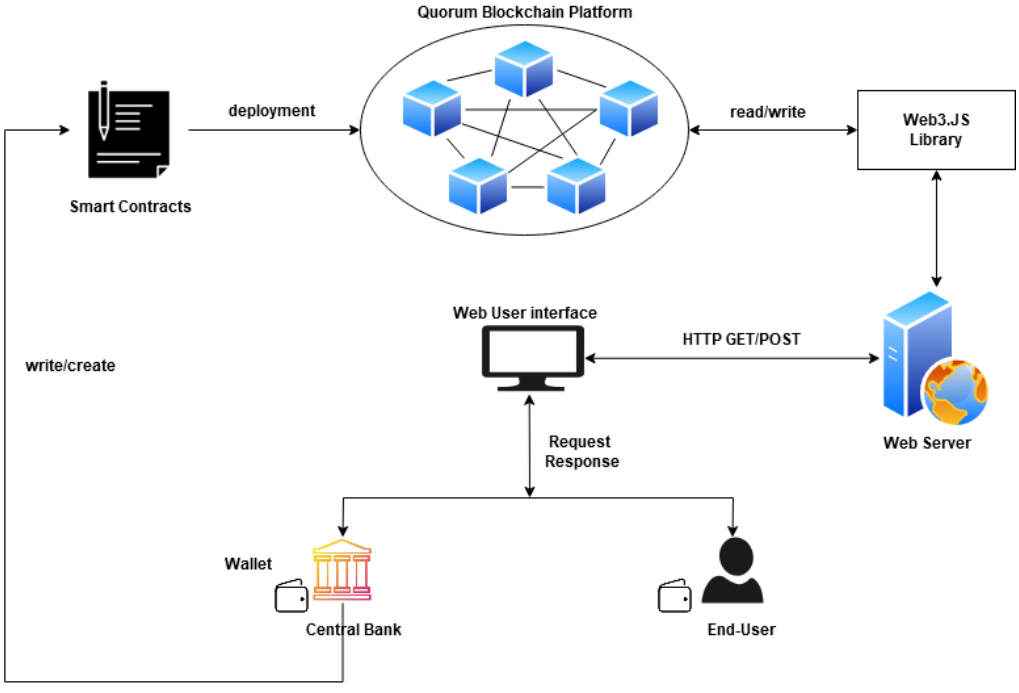

Figure 4.1: Implementation of the direct CBDC model using Quorum blockchain

4.1.2 Smart Contract

Central bank is expected to deploy a smart contract containing ERC20 token functionality in the Quorum network. Function execution of adding and burning token is limited to central bank (as per the requirements listed in section 3.1.1) by using modifiers. To enable mapping of end-users' account addresses to real-world identities, we defined an additional struct object that maps account addresses to user identities. Also, to track the flow of funds, we also created mappings that labels a particular account address as valid and only these valid addresses can hold CBDC tokens. Listing IV.1 shows the source code snippet of the smart contract deployed into the Quorum network.

```
1  contract BOKCoin {
2      mapping(address=>uint256) public balanceOf;
3      mapping(address=>bool) public validAddresses;
4      mapping(address=>AccountInfo) public accountDetailsByAddr;
5      uint256 public userAccountsNum;
6      constructor(uint256 _initialSupply) public {
7          totalSupply += _initialSupply;
8          creator = msg.sender;
9          burner = 0x0;
10         balanceOf[msg.sender] = totalSupply;
11         validAddresses[msg.sender] = true;
12         userAccountsNum = 0;
13     }
14     struct AccountInfo {
15         address userAddress;
16         string name;
17         string identification;
18         uint256 idnum;
19     }
20     modifier onlyCreator() {
21         require(msg.sender == creator);
```

```
22      }
23      function addUserAddressInfo(address _addr, string memory _name,
            string memory _id) public onlyCreator returns (bool success
        ) {
24          userAccountsNum++;
25          accountDetailsByAddr[_addr] = AccountInfo(_addr, _name, _id
                , userAccountsNum);
26          validAddresses[_addr] = true;
27          return true;
28      }
29      function transfer(address _to, uint256 _value) public returns (
            bool success) {
30          require(balanceOf[msg.sender] >= _value);
31          require(validAddresses[_to] == true);
32          balanceOf[msg.sender] -= _value;
33          balanceOf[_to] += _value;
34      }
35      function addTokens(uint256 _quantity) public onlyCreator
            returns (bool success) {
36          totalSupply += _quantity;
37          balanceOf[creator] += _quantity;
38          return true;
39      }
40      function burnTokens(uint256 _quantity) public onlyCreator
            returns (bool success) {
41          transfer(burner, _quantity);
42          totalSupply -= _quantity;
43          return true;
44      }
45  }
```

Listing IV.1: Smart Contract Code Snippet for ERC20 token functionality

In the above listing, the struct '*AccountInfo*' holds user-related information such as name and identification to a particular account address generated by our application. The mapping '*accountDetailsByAddr*' is used to map an address to

the '*AccountInfo*' object. Also, '*validAddresses*' is another mapping that is used to track the flow of funds. The process of user account address creation was explained in Fig. 3.2. During user account address creation the smart contract function '*addUserAddressInfo*' is invoked which maps user account addresses to real world identities and keystore files are provided to the users. Program logic for creation, transfer and addition of tokens is implemented as shown above.

4.1.3 User Interfaces

Users interact with the blockchain through a web-application (Fig. 4.2). Users are enrolled into the CBDC application upon authorization by the administrator of central bank and are mapped to real-world name and identities (Fig. 4.3). The blockchain generates a keystore file secured by a password entered during account creation, which is then imported into a wallet. e.g., Metamask (Fig. 4.4). This newly created account address, mapped to real-world identity, is listed among registered list of account addresses (Fig. 4.5). A user in possession of CBDC tokens can initiate transfer to a registered address filling up a form consisting of recipient address and amount to transfer (Fig. 4.6). Upon successful transfer, the new user can confirm equivalent tokens in its account (Fig. 4.7). Only the central bank is allowed for token addition (Fig. 4.8) which is confirmed upon successful transaction (Fig. 4.9). Also, only central bank is allowed for burning of tokens (Fig. 4.10) and corresponding change in total supply is observed. (Fig. 4.11.)

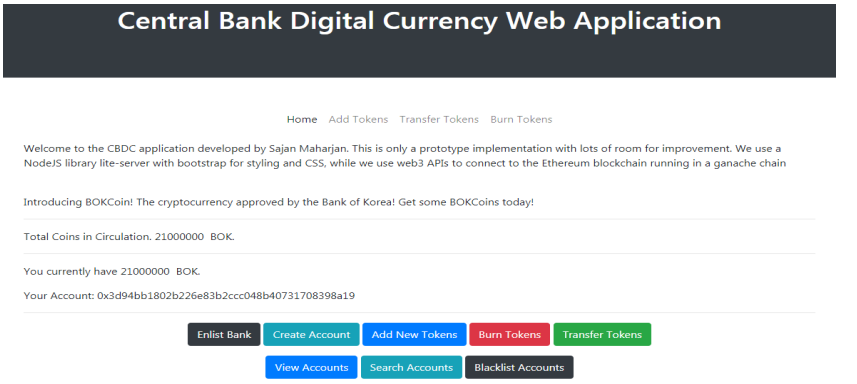

Figure 4.2: Landing page of direct CBDC implementation

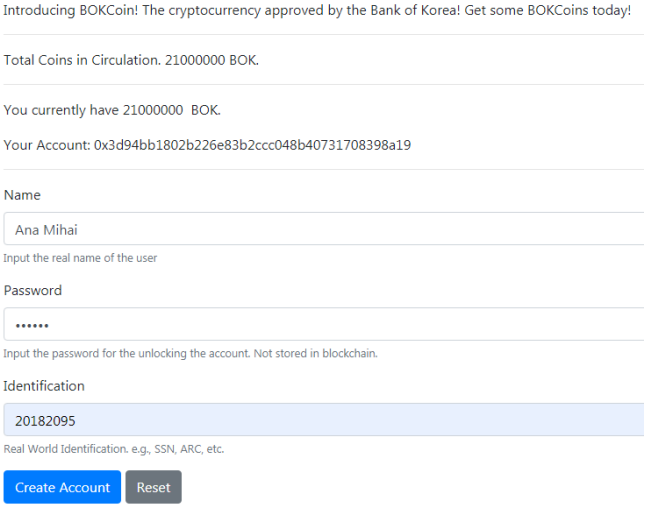

Figure 4.3: Account address creation process secures keystore file with password

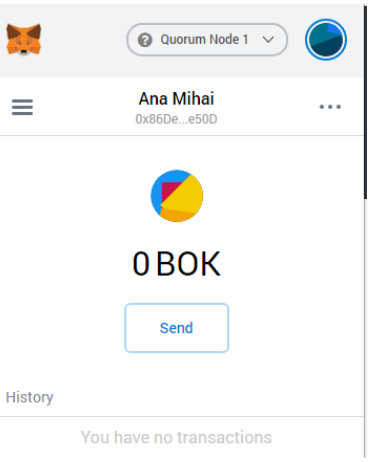

Figure 4.4: User account address imported into Metamask using keystore file

Introducing BOKCoin! The cryptocurrency approved by the Bank of Korea! Get some BOKCoins today!

Total Coins in Circulation. 21000000 BOK.

You currently have 21000000 BOK.

Your Account: 0x3d94bb1802b226e83b2ccc048b40731708398a19

ID	User Address	Creator Address	Real World Name	Identification
1	0x86dec698f9e390a00fb06fa07c5ca719f0dee50d	0x3d94bb1802b226e83b2ccc048b40731708398a19	Ana Mihai	20182095

Figure 4.5: List of registered account addresses mapped to real-world identity

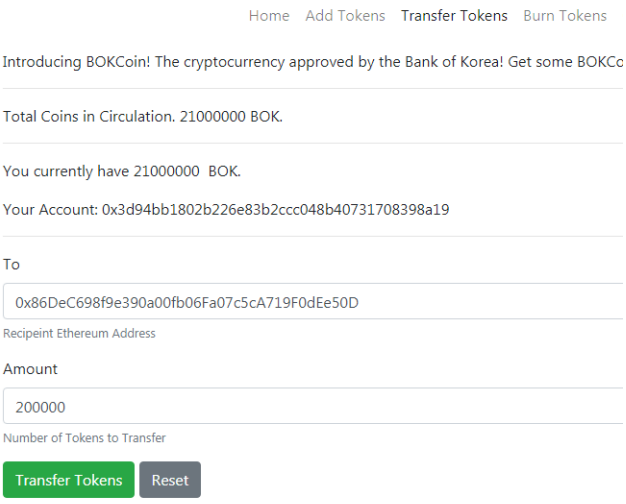

Figure 4.6: Transfer of tokens to a registered address

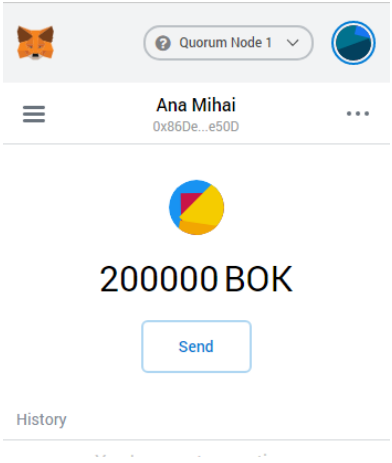

Figure 4.7: Balance confirmation upon successful transfer

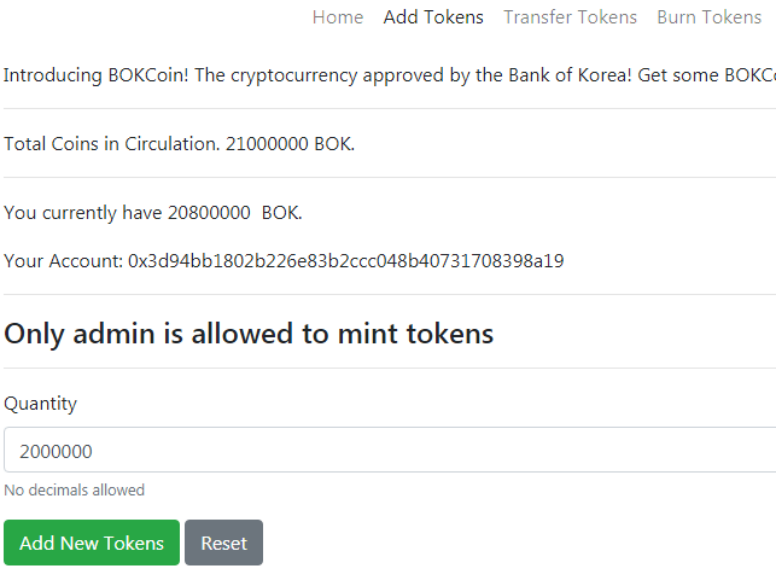

Figure 4.8: Token addition by central bank account

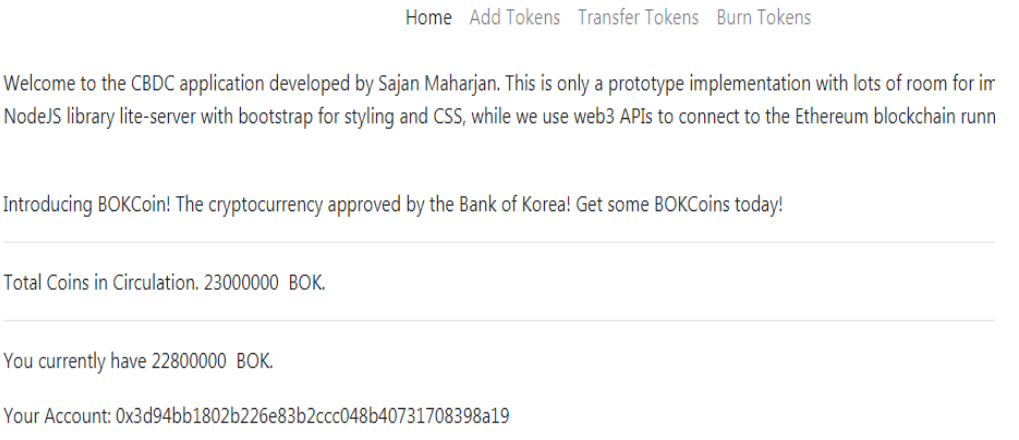

Figure 4.9: Balance confirmation after token addition

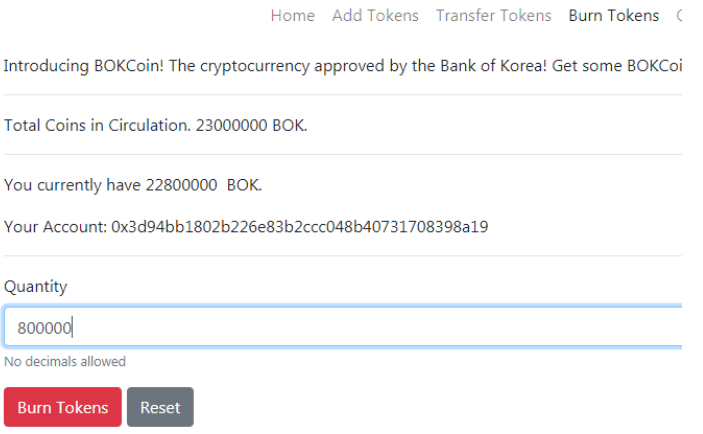

Figure 4.10: Burning of tokens by central bank account

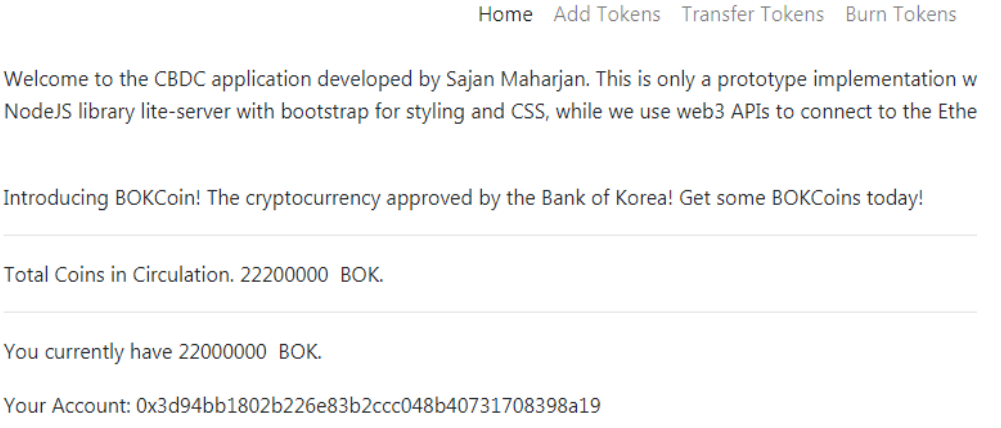

Figure 4.11: Balance and total supply confirmation after token burning

4.2 Implementation of Indirect CBDC Model

4.2.1 Implementation Architecture

In the indirect CBDC model, commercial banks are integral part of the CBDC environment responsible for the distribution of CBDC tokens to end-users and offering additional financial services via smart contracts. The details of our implementation is shown in Fig. 4.12. Under our implementation, we used a Quorum network operating with RAFT consensus algorithm with 2 nodes at least (one for central bank and one for commercial bank).

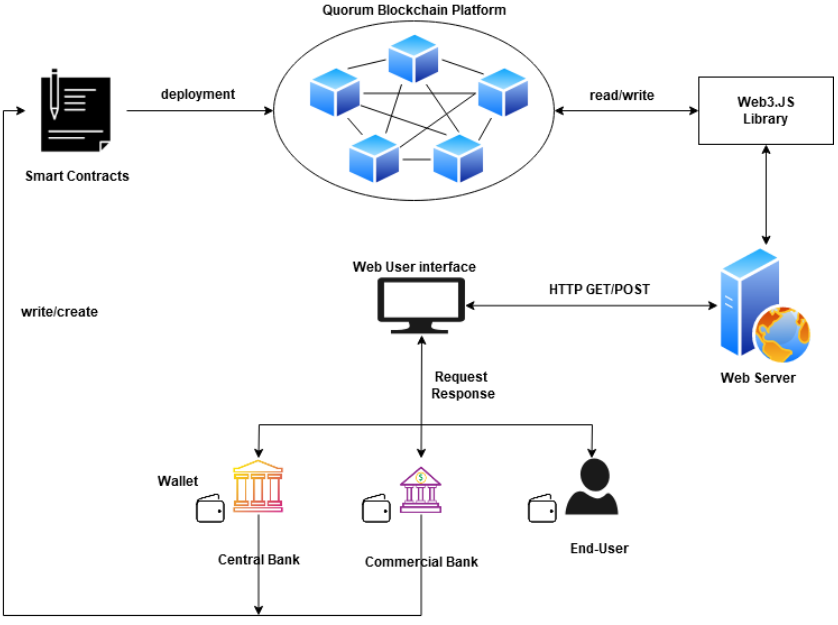

Figure 4.12: Implementation of indirect CBDC model using Quorum blockchain

Initially, the central bank deploys a smart contract containing ERC20 token functionalities into the Quorum network. Next, commercial bank deploys an-

other smart contract inheriting the CBDC token deployed by central bank and defines additional functionalities for financial services. In our implementation, we developed smart contract logic for interest on deposits offered by commercial banks and deployed it to the blockchain. Similar to direct CBDC implementation, users (central bank, commercial bank and end-users) interact with the blockchain by sending requests via HTML forms hosted by a web-application server. The web-server then interacts with the blockchain through the Web3.js library. Users' HTTP requests for transactions and blockchain variables are handled by Web3.js. The Web3.js library reads and writes data to and from the blockchain node, and response is returned to the user via web interface. Wallet addresses are used to conduct transactions in the given network. Likewise, we also used Metamask for easy management of wallet addresses in the indirect CBDC implementation.

4.2.2 Smart Contract

Central bank first deploys a smart contract containing ERC20 token functionalities into the blockchain. The program logic for this contract is almost similar to the one deployed by the central bank in the direct model with minor modifications which includes mappings for commercial bank addresses. Listing IV.2 shows the smart contract deployed by central bank for indirect model.

```
46 contract BOKCoin {
47     mapping(address=>uint256) public balanceOf;
48     mapping(uint256=>AccountInfo) public accountDetailsById;
49     mapping(address=>AccountInfo) public accountDetailsByAddr;
50     uint256 public userAccountsNum;
51     address[] public banks;
52     uint256 public bankNum = 0;
53     mapping(address=>bool) public validAddresses;
54     mapping(address=>bool) public validBankAddresses;
55
56     constructor(uint256 _initialSupply) public {
57         ...
58         //same as direct model
59     }
60     modifier onlyCreator() {
61         require(msg.sender == creator);
62     }
63
64     struct AccountInfo {
65         address userAddress;
66         address creatorAddress;
67         string name;
68         string identification;
69         uint256 idnum;
70     }
71     function addUserAddressInfo(...) public bool (returns success)
           {
72         require(validBankAddresses[msg.sender] == true);
73         ...
74     }
75     function addBank(address _bankAddr) public onlyCreator returns
           (bool success) {
76         banks.push(address(_bankAddr));
77         bankNum++;
78         validBankAddresses[_bankAddr] = true;
79         setValidAddress(_bankAddr);
80         return true;
81    }
```

```
82      function transfer(address _to, uint256 _value) public returns (
            bool success) {
83          ...
84          //same as direct model
85      }
86      function addTokens(uint256 _quantity) public onlyCreator
            returns (bool success) {
87          ...
88          //same as direct model
89      }
90      function burnTokens(uint256 _quantity) public onlyCreator
            returns (bool success) {
91          ...
92          //same as direct model
93      }
94  }
```

Listing IV.2: Source code snippet of smart contract deployed by central bank in indirect model

Modifications include the addition of different variables and functions for tracking banks. Central bank enlists an address as a bank by invoking the '*addBank*' function which then provides necessary privilege to a bank address to add user accounts and offer other financial services. Mapping of users' account addresses to real-world identities is done by using the struct '*AccountInfo*'. Since both commercial and central banks can create user account addresses in indirect model, we use an additional field '*creatorAddress*' which signifies the address of the bank that created the user account address. Flow of funds is controlled through the mapping '*validAddresses*'. User addresses are checked for validity when executing token related functions. The mappings also allow blacklisting of banks and user account addresses i.e., *validAddresses[addr] = false*

and*validBankAddresses[addr] = false*.

Next, commercial bank deploys another smart contract containing logic for interests on deposits. Listing IV.3 shows the source code snippet of the smart contract deployed by commercial bank for deposit services.

```
95  uint256 public depositCount = 0;
96  mapping(uint256=>Deposit) public deposits;
97
98  struct Deposit {
99          uint256 id;
100         address payable depositor;
101         address payable provider;
102         uint256 rate;
103         uint256 timeperiod;
104         uint256 principal;
105         uint256 maxDepositAmt;
106         uint256 endtime;
107         uint256 starttime;
108         uint256 interest;
109         uint256 amt;
110         bool isAvailable;
111         bool notWithdrawn;
112 }
113
114 function offerDeposit(uint256 _rate, uint256 _time, uint256
        _maxDepositAmt) public returns (bool success) {
115     require(balanceOf[msg.sender] >= _maxDepositAmt);
116     require(validBankAddresses[msg.sender] == true;
117     depositCount++;
118     deposits[depositCount] = Deposit(depositCount, 0x0, msg.sender,
            _rate, _time, 0, _maxDepositAmt, 0, 0, 0, 0, true, false);
119       return true;
120 }
121
122 function makeDeposit(uint256 _id, uint256 _principal) public
        payable returns (bool success) {
```

```
123     require(validAddresses[msg.sender] == true
124     require(balanceOf[msg.sender] >= _principal);
125     require(_principal <= deposits[_id].maxDepositAmt);
126     require(deposits[_id].isAvailable == true);
127     deposits[_id].depositor = msg.sender;
128     deposits[_id].principal = _principal;
129     deposits[_id].starttime = now;
130     deposits[_id].endtime = deposits[_id].starttime + deposits[_id
            ].timeperiod * 1 minutes;
131     deposits[_id].notRedeemed = true;
132     deposits[_id].isAvailable = false;
133     address payable _provider = deposits[_id].provider;
134     balanceOf[msg.sender] -= _principal;
135     balanceOf[_provider] += _principal;
136     uint256 _ptr;
137     uint256 _rate = deposits[_id].rate;
138     uint256 _timeperiod = deposits[_id].timeperiod;
139     _ptr = (_principal.mul(_rate)).mul(_timeperiod);
140     deposits[_id].interest = _ptr.div(100);
141     deposits[_id].amt = _principal.add(deposits[_id].interest);
142         return true;
143 }
144
145 function withdrawDeposit(uint256 _id) public payable returns (bool
        success) {
146     require(now >= deposits[_id].endtime);
147     require(msg.sender == deposits[_id].depositor);
148     require(deposits[_id].notRedeemed == true);
149     uint256 _amount = deposits[_id].amt;
150     address payable _provider = deposits[_id].provider;
151     balanceOf[msg.sender] += _amount;
152     balanceOf[_provider] -= _amount;
153     deposits[_id].notWithdrawn = false;
154     return true;
155 }
```

Listing IV.3: Smart contract code for deposit services deployed by a Commercial bank in indirect model

Commercial banks and end-users interact with the *Deposit* object by invoking different functions defined in the smart contracts, namely: *offerDeposit*, *makeDeposit* and *withdrawDeposit*.

Initially, a commercial bank interested in providing deposit services to end-users invokes the *offerDeposit* function. This step initializes a new *Deposit* object and stores the details of service offered into the blockchain i.e., service provider address, rate of interest, required time period for deposit and maximum amount upto which deposits can be made. Next, a prospective customer interested in making deposit to the offered service, invokes the *makeDeposit* function. The *makeDeposit* function references a particular *Deposit* object stored in the blockchain and updates information such as depositor address, principal value, timeperiod and other flags, and transfers balance between provider and depositor. Once the maturity of the deposit is reached, the depositor then invokes the *withdrawDeposit* function which then transfers corresponding amount (with interest) back to the depositor from the service provider.

Thus, our implementation of indirect CBDC model requires two separate smart contracts to be deployed on the Quorum blockchain. The first smart contract deployed by central bank provides only basic token functionalities while the second contract deployed by commercial bank offers additional financial services.

4.2.3 User Interfaces

Users are identified in the indirect CBDC implementation by their wallet addresses. The process for account address creation has been explained in the Design chapter of this thesis (Fig. 3.5). Accordingly, account addresses for central

bank, commercial bank and end-user were created and imported into Metamask (Fig. 4.13). Users interact with the blockchain via a web-interface which has buttons for transactions like create tokens, transfer tokens, burn tokens, offer deposits, view deposits, etc. (Fig. 4.14). Since implementation of transactions like create tokens, transfer tokens, burn tokens are similar to the direct CBDC implementation, we skip the details of these user interfaces. Instead, we focus on the user interfaces for deposit services. Any user can view the list of deposits by clicking on the 'View Deposit' button (Fig. 4.15). A commercial bank user can ask for deposits from end-users by clicking on the 'Offer Deposit' button. In doing so, the commercial bank fills up a form specifying the time period, maximum deposit amount and offered rate of interest (Fig. 4.16). Upon submission, the form data is submitted as a transaction to the blockchain calling the *offerDeposit* function of the contract.

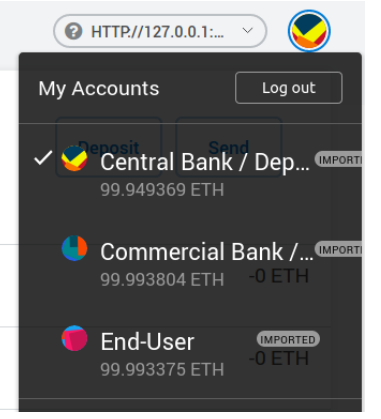

Figure 4.13: List of accounts created for indirect CBDC implementation

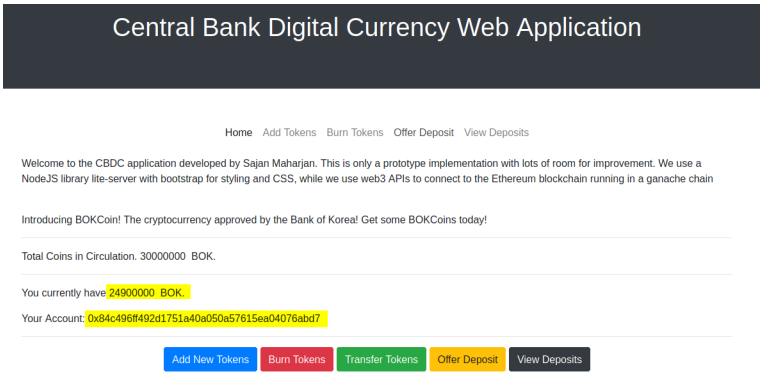

Figure 4.14: Landing page for indirect CBDC implementation

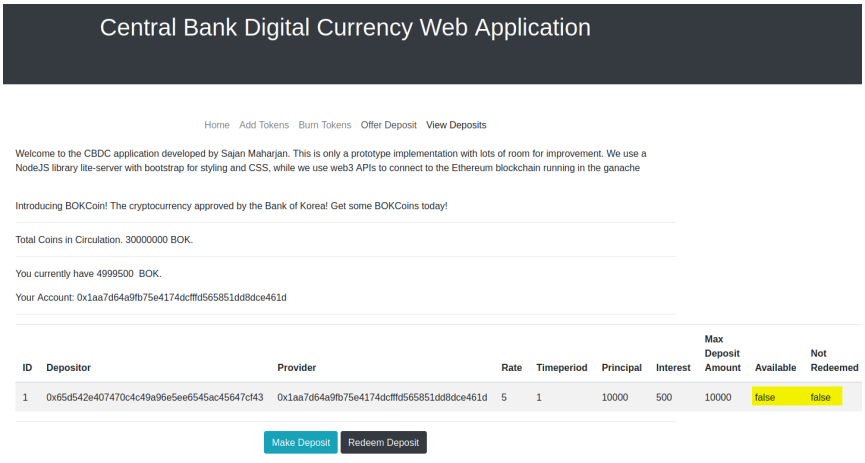

Figure 4.15: Initial list of deposits in indirect CBDC implementation

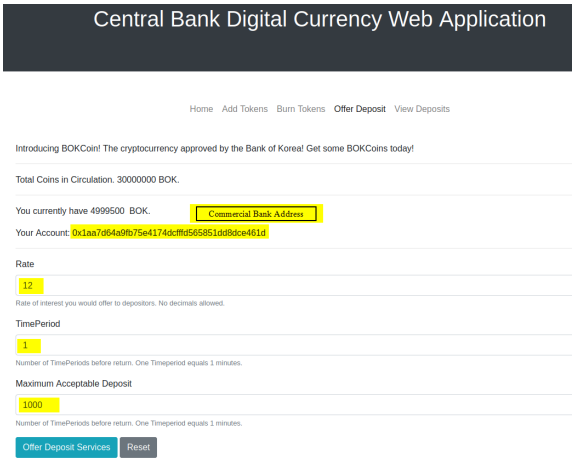

Figure 4.16: Commercial banks offer deposit services specifying rate, timeperiod and max deposit amount

This new offered deposit service is now listed amongst the list of deposits (Fig. 4.17). Notice that the latest deposit is identified by the highest deposit ID and is labelled available. Interested end-users can make deposits into a particular deposit service by clicking on the 'Make Deposit' button. End-users need to submit deposit id and principal value via HTML forms to make deposit (Fig. 4.18). This action invokes the *makeDeposit* function of the smart contract and results in the change of balance. Viewing the list of deposits shows update to the deposit made i.e., depositor address, principal amount and is labelled unavailable to prevent other users from making deposit (Fig. 4.19). To withdraw the deposited amount with interest, required time period must have elapsed. The depositor submits deposit id and clicks the 'Redeem Deposit' button (Fig. 4.20) which invokes *withdrawDeposit* function. Transfer of balance takes place and *notWithdrawn* flag is updated to prevent double spending (Fig. 4.21).

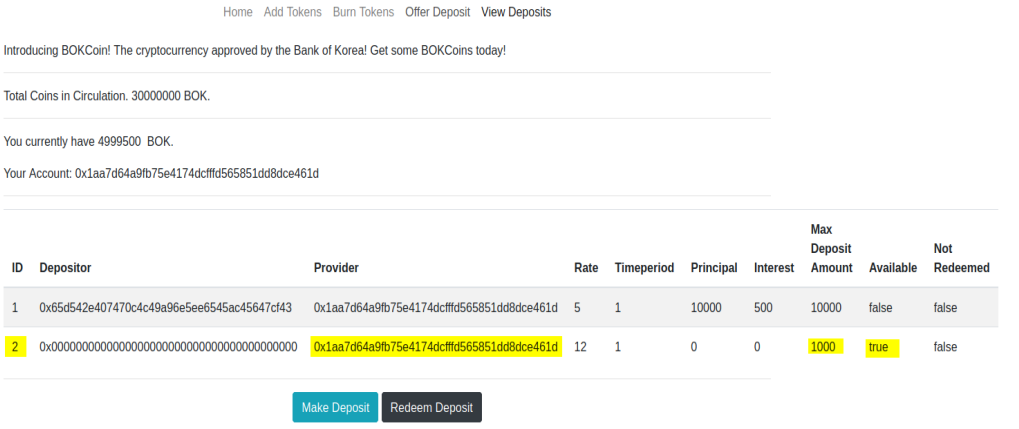

Figure 4.17: Updated list of deposit when commercial banks submit new deposit service

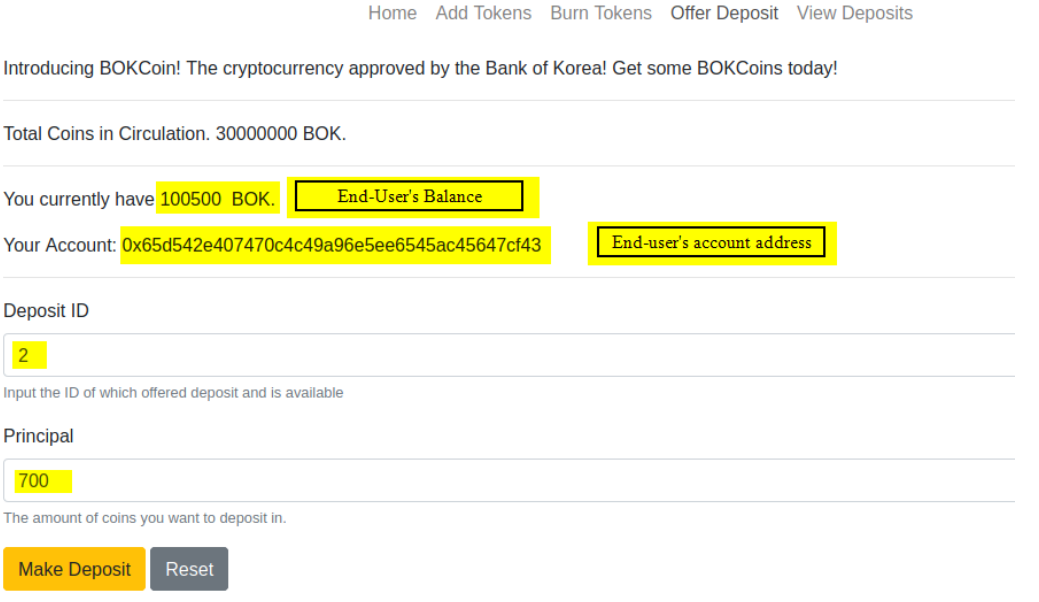

Figure 4.18: Interested end-user making a deposit to the offered deposit service

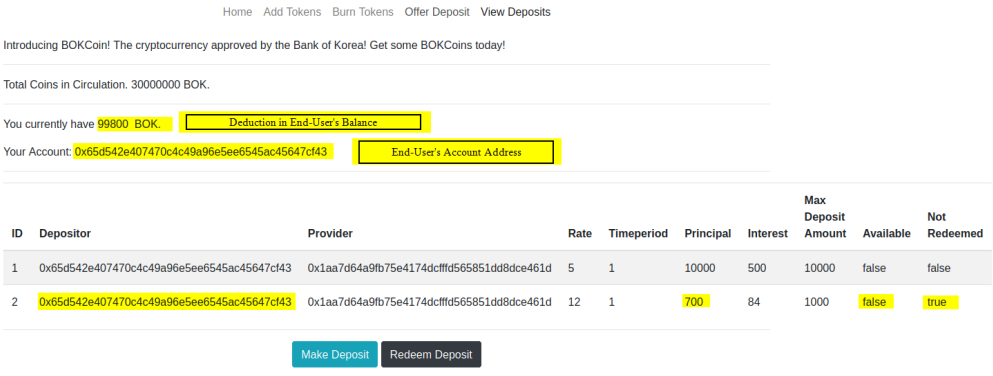

Figure 4.19: Deposit data is updated on user making deposits

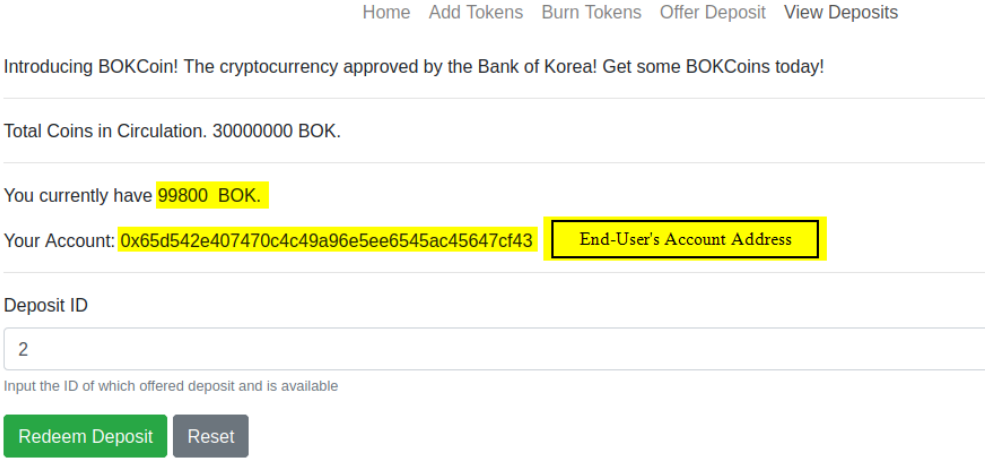

Figure 4.20: Deposit redemption by end-user upon time period elapsed

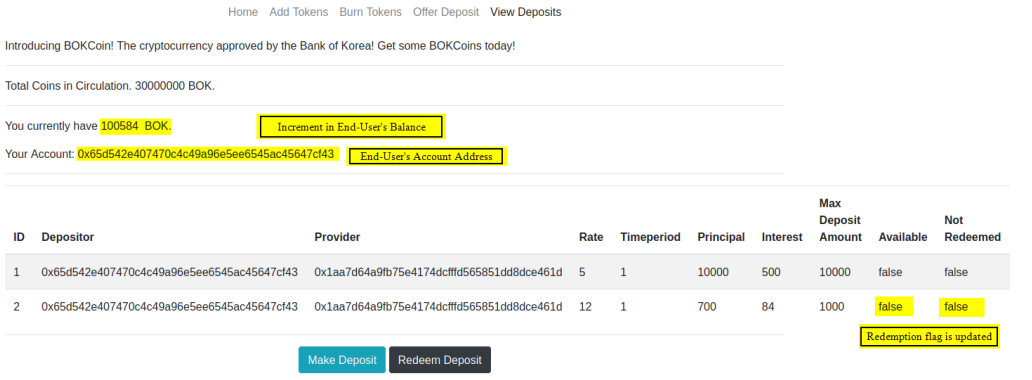

Figure 4.21: Increase in End-User's balance and change in redemption flag

4.3 Implementation of Hybrid CBDC Model

4.3.1 Implementation Architecture

Under the hybrid CBDC model, central banks, commercial banks, private PSPs and end-users are the major participants. The details of our implementation is shown in Fig. 4.22. The implementation architecture is very similar to the implementation of indirect CBDC. Central banks and commercial banks deploy smart contracts onto the Quorum blockchain containing basic token functionalities and additional financial services. End-users can choose to transact in CBDC tokens either through wallet addresses provided by commercial banks or use payment applications provided by private PSPs. User generated transactions from payment applications are recorded off-chain in private databases, so we do not concern with the implementation logic for such transactions. However, private PSPs report aggregate transactions to the central bank at periodic intervals. Central bank, commercial bank and private PSPs interact with the blockchain by

sending requests via HTML forms hosted by a web-application server. Web3.js library is used for reading/writing data, submitted via HTTP requests, into the blockchain. Similarly, response data from the blockchain is returned in the reverse direction via the web application.

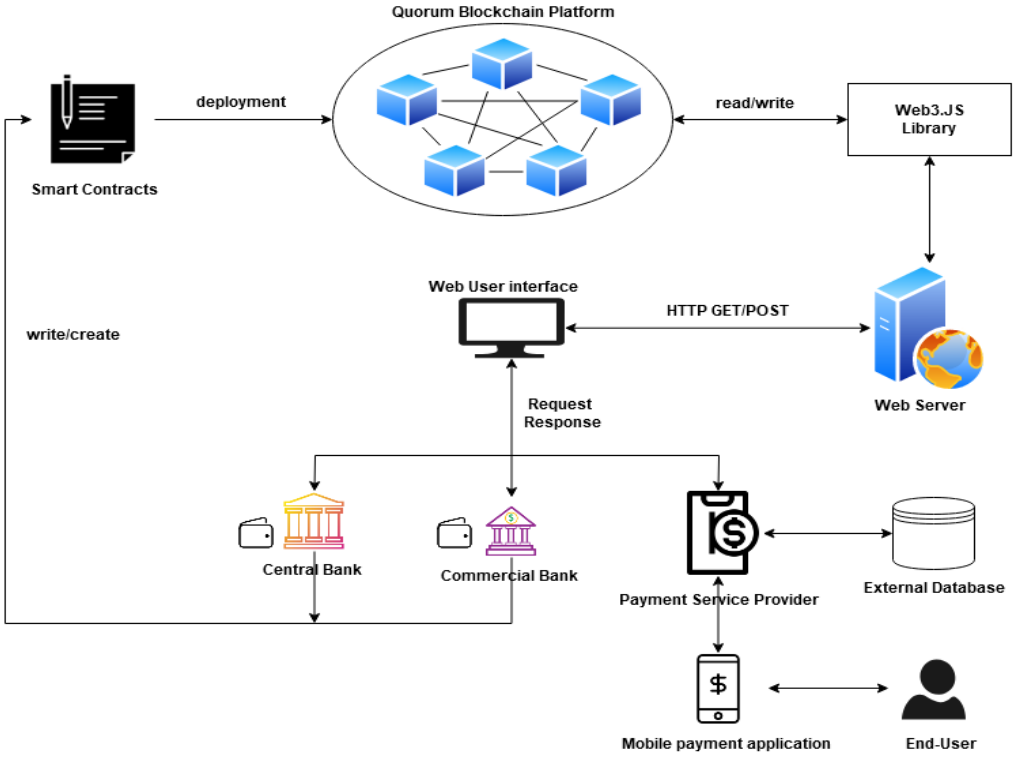

Figure 4.22: Implementation of Hybrid CBDC model using Quorum blockchain

4.3.2 Smart Contract

The smart contract codes deployed by central bank and commercial banks in the hybrid model is almost same as smart contract code in indirect model. Central bank will deploy smart contract code containing basic token functionali-

ties. An additional modification may include the definition of addresses as valid PSP addresses. Listing IV.4 highlights the major code addition in hybrid CBDC model.

```
156 contract BOKCoin {
157     //additional portion for PSP information
158     uint256 public pspCount = 0;
159     mapping(address=>bool) validPSPAddresses;
160     address[] public psps;
161     function addPSP(address _PSPAddr) public onlyCreator returns (
            bool success) {
162         psps.push(address(_bankAddr));
163         pspCount++;
164         validPSPAddresses[_bankAddr] = true;
165         return true;
166     }
167 }
```

Listing IV.4: Smart contract code snippet for adding PSP information

The smart contract deployed by commercial banks require no changes since there is no direct interaction between commercial banks and private PSPs in the hybrid model.

4.3.3 User Interfaces

User interfaces describing the creation, addition and transfer of tokens in the hybrid CBDC implementation is same as the interaction between central bank and commercial bank in indirect CBDC model. Hence, for brevity, we skip the details of the user interfaces.

V. Evaluation

As demonstrated in chapter IV, we used Quorum blockchain platform to implement direct, indirect and hybrid CBDC models. During initial exploration of CBDC application development, Hyperledger Iroha was also studied. Ethereum (the base fork of Quorum) and hyperledger are among the most popular blockchain platforms for decentralized application development. Popular CBDC initiatives like Project Ubin, Project Bakong and Project Khokha have implemented CBDC applications by using Quorum and HL Iroha blockchain platforms. Project Khoka and Project Ubin were targeted towards inter-bank settlement applications while only Project Bakong is aimed towards retail CBDC implementation.

In this chapter, we compare Quorum and HL Iroha blockchain platforms for features relating to CBDC implementation. Next, we evaluate the validity of our work and describe the pros and cons of our implementation. Also, since Quorum was used in the implementation of all three models, we discuss which model is best suited for CBDC development using Quorum blockchain.

Table 5.1 summarizes the major differences between HL Iroha and Quorum. Clearly, the abundance of features in Quorum blockchain like smart contract functionality, multiple choices of consensus algorithms, enhanced privacy, third-party wallet application compatibility, etc. led to the selection of Quorum blockchain over HL Iroha for CBDC implementation.

Table 5.1: Feature differences in HL Iroha and Quorum

Features	Hyperledger Iroha	Quorum
Use Cases	Payment-specific	Generic, Enterprise
Consensus Mechanism	YAC	IBFT and RAFT
Support for Private Transactions	No	Yes
Smart Contract Execution	No	Yes
Built-In Commands and Queries	Yes	No
Native Cryptocurrency	No	Yes
Decimal Precision Support	Yes	No
Metamask Compatibility	No	Yes

While previous work on CBDC have mainly focused on wholesale CBDCs for inter-bank settlements and cross-border transfers, under this project we designed and implemented prototypes for retail CBDC. Although our implementation of direct, indirect and hybrid CBDC models are less sophisticated compared to industrial projects and is only a prototype implementation, insights from this project could be used in further research and development of retail CBDC.

We stress that our implementation of retail CBDC model, particularly indirect CBDC implementation, fits well with the current banking environment at the customer level i.e., our implementation of indirect CBDC model mimics basic banking functionalities of digital currency distribution, offering of financial services, mapping of account addresses to real-world identities and control of addresses (blacklisting and white-listing). All of the suggested features were implemented on the blockchain without using any external database application. Although, sensitive personal information maybe stored on the blockchain, access to such blockchain data is limited to banks only. However, it is still necessary

for banks to maintain secondary external databases for cross-verification of user identities uploaded to the blockchain.

Also, our implementation of CBDC is generic in nature. Although, we named the created token as 'BOKCoin' (as in 'Bank of Korea Coin'), our implementation does not relate in any way to the CBDC initiatives taken by the Bank of Korea. Features developed for the implementation are generic to the banking industry. However, it is essential to realize that CBDC initiatives taken by central banks around the globe differ depending on a variety of factors. e.g., digital payment landscape within a country, financial laws, desired goals/outcomes, etc. Thus, our implementation is not totally relevant to any country pursuing CBDC initiatives. However, insights can be drawn from our implementation. Also, the comparison made between two blockchain platforms— Hyperledger Iroha and Quorum, can serve as a guide for undertaking CBDC development in respective platforms.

Since our implementation of direct and indirect CBDC models is only a prototype, the user interfaces developed are primitive in nature. User experience can be improved by providing adequate user-interfaces in response to actions performed by the user like data submission, page redirection, etc. at the web-application layer. We also carried out smoke tests on our smart contract code using the Truffle [35] framework to prevent any unwanted execution of program logic. While our code does not execute unwanted operations, it is still not optimal in design. Thus, code refactoring can be done to improve code quality. Furthermore, exhaustive testing against known smart contract vulnerabilities can also be carried out later.

Additional point of concern in our implementation is the scalability of the system. Concern for scalability is a valid argument in all blockchain-based retail CBDC implementations as the volume of transactions is exceedingly high. Although CFT and BFT based consensus mechanism provides immediate transaction finality, retail CBDC application could still suffer from delay in transaction processing due to large volume of transactions. Thus, it is essential to conduct analysis on the achieved throughput (number of transactions processed per second). Also, our implementation of indirect CBDC model is a Quorum network with RAFT consensus mechanism. RAFT is a crash fault tolerant consensus algorithm which fails in the presence of a malicious node being elected as 'leader'. Therefore, our system cannot currently handle malicious nodes. Improvements can also be made in the current implementation while storing sensitive personal information on the blockchain i.e., we can use enhanced privacy features offered in Quorum like zero-knowledge proofs and homomorphic encryption.

Furthermore, in this thesis, we designed and implemented three different CBDC models— direct, indirect and hybrid CBDC using Quorum blockchain. Based on our findings and implementation of each model using Quorum, we present a comparison of CBDC features for each implemented model using Quorum (Table. 5.2).

In all three implementations, basic ERC20 token functionalities were implemented. However, only indirect model properly emulates the current banking environment. Private transactions feature supported by Quorum can be applicable in indirect model e.g., a bank's internal transactions. Additional financial

services can be built on top of Quorum using smart contracts and such responsibilities can be delegated to banks. In hybrid model, PSPs operate separate application for user transactions, so concern for scalability is minimum while ensuring support for third party applications. However, end-user transactions are not completely visible to central bank in hybrid model.

Table 5.2: Comparison of CBDC features in direct, indirect and hybrid model using Quorum

CBDC Features	Direct	Indirect	Hybrid
ERC20 token functionalities	Yes	Yes	Yes
Emulates banking environment	No	Yes	No
Usability of private transactions	No	Yes	Yes
Offering of Financial Services	No	Yes	Yes
Concern for scalability	High	Medium	Low
Third-party Integration	Low	Medium	High
Central Bank Txn Visibility	High	Medium	Low

Thus, we argue that the Quorum blockchain is best suited for implementation in indirect model as most of the features required for indirect CBDC implementation can be implemented easily. Although implementing hybrid CBDC in Quorum guarantees ubiquity through widely used mobile payment applications, it requires PSP specific implementation of an additional layer (for porting CBDC tokens into such applications). On the other hand central banks can guarantee maximum visibility and regulation in direct model, however greater obligations and responsibilities are born by central banks. The direct model may also not be best suited with the current banking environment.

VI. Conclusion

6.1 Summary

Banks and governments around the world are interested in issuing a well-regulated digital currency to overcome the inherent problems of fiat cash and curb illegal activities on public blockchain platforms. This has led to the birth of CBDC. In this thesis, we explained the concepts of CBDC. Particularly, we explained the motivation behind CBDC and its taxonomy. We also presented on the different CBDC initiatives taken by banks and governments around the world. Recent developments in CBDC have involved payments at the wholesale level (e.g., inter-bank settlements) but developments in CBDC at the retail level is rare and a few.

We presented the design and implementation of CBDC at the retail level based on blockchain environments. Three models of CBDC— direct model, indirect model and hybrid model were designed and implemented by using Quorum blockchain platform. The innovation in our implementation involves the offering of financial services (e.g., interest on deposit) at the retail level on top of CBDC platform through smart contract execution. We also ensured that user accounts and identities are well-regulated and controlled by defining mappings in the smart contract. Also, we compared the features offered by Quorum and HL Iroha in terms of CBDC implementation. Finally, we discussed the validity of our work

and the pros and cons of our implementation. We also came to the conclusion that features available in the Quorum blockchain environment is most suitable for the implementation of indirect CBDC model.

6.2 Future Work

Our implementation of retail CBDC is only a prototype implementation and is far from production. Further developments can be made to our implementation by developing mobile interface for our application. We can also conduct research on the realized throughput and scalability of retail CBDC applications in the future. We can also enhance privacy in our implementation by hiding user sensitive data. In our implementation, while mapping account addresses to user identities, unique personal identification data is stored on the blockchain without any encryption. To prevent the leakage of such sensitive information, we can use zero-knowledge proofs and homomorphic encryption on such data. Deployment of the Quorum blockchain network with byzantine fault tolerant algorithms such as I-BFT can tolerate adversarial nodes and thus improve the dependability of our blockchain system. We used Truffle framework to conduct smoke tests on our smart contracts. While our smart contract does not execute maliciously, it is not optimal in design. Smart contract code refactoring and exhaustive testing against known smart contract vulnerabilities before deployment can be done in future implementations. Other avenues to conduct research in this field includes exploration of other blockchain platforms such as Hyperledger Fabric and R3 Corda for CBDC implementation at the retail level.

요 약 문

비트코인의 출현과 더불어 블록체인 기술은 수천 개의 알트 코인과 암호화폐 개발에 영감을 주었다. 수천 개의 알트 코인 및 암호화폐를 이용한 수백만 달러 상당의 거래가 퍼블릭 블록체인 플랫폼 상에서 매일 이루어지고 있다. 퍼블릭 블록체인은 P2P, 불변성, 거래 기록의 분산화의 특징을 제공하지만, 규제가 부족하고, 불법 거래에 취약하다. 서로 다른 암호화폐에 대한 규제의 부족으로 인해 은행과 정부는 암호화폐의 가치 평가와 활동 내역에 대한 검증 방안에 대해 우려하고 있다. 반면 은행과 정부는 이익을 창출하기 위해 기존의 은행 서비스에 블록체인 기술을 적용할 수 있는 방향에 대해 고려하고 있다. 특히 은행은 규제와 통제가 가능한 디지털 통화의 발행을 기대하고 있다. 이러한 기대로 중앙은행 디지털 화폐(CBDC)라는 개념이 등장하였다. 본 논문에서는 CBDC의 개념과 동기, 그리고 서로 다른 유형의 CBDC에 대해 설명한다. 또한 블록체인 기반의 direct CBDC와 indirect CBDC 모델의 디자인 및 구현 방식에 대해 소개한다. 마지막으로 CBDC 구현을 위해, Hyperledger와 Quorum라는 잘 알려진 두 개의 블록체인 플랫폼인을 비교 분석하였다.

References

[1] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.

[2] Bitcoin Wiki. Genesis block. Available at https://en.bitcoin.it/wiki/Genesis_block.

[3] The Times 2009. Chancellor on brink of second bailout for banks. Available at https://www.thetimes03jan2009.com/.

[4] Michael Crosby, Pradan Pattanayak, Sanjeev Verma, Vignesh Kalyanaraman, et al. Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2(6-10):71, 2016.

[5] Yuval Gov. Altcoins – the complete guide. Available at https://cryptopotato.com/altcoins-the-complete-guide.

[6] Vitalik Buterin et al. Ethereum white paper. *GitHub repository*, 1:22–23, 2013.

[7] Block Media. [what is block] central bank digital currency 'cbdc'. Available at https://www.blockmedia.co.kr/archives/114942.

[8] Mike Unser. Penny costs 2.06 cents to make in 2018, nickel costs 7.53 cents; us mint realizes $321.1m in seignior-

age. Available at https://www.coinnews.net/2019/05/24/penny-costs-2-06-cents-to-make-in-2018/.

[9] Board of Governors of the Federal Reserve System. 2020 currency budget. Available at https://www.federalreserve.gov/foia/files/2020currency.pdf.

[10] GetSwish AB. Swish payments. Available at https://www.swish.nu/.

[11] Alibaba. Alipay. Available at https://intl.alipay.com/.

[12] Tenpay Tencent. Wechat pay. Available at https://pay.weixin.qq.com/index.php/public/wechatpay.

[13] Libra Association et al. Libra white paper. *Internet access:,[accessed June 19, 2019]*, 2019.

[14] Walter Engert and Ben Siu-Cheong Fung. Central bank digital currency: Motivations and implications. Technical report, Bank of Canada Staff Discussion Paper, 2017.

[15] Morten L Bech and Rodney Garratt. Central bank cryptocurrencies. *BIS Quarterly Review September*, 2017.

[16] Matthieu Bouchaud, Tom Lyons, Matthieu Saint Olive, and Ken Timsit. Central banks and the future of digital money, 2020.

[17] B Coeuré and J Loh. Central bank digital currencies. bank for international settlements (2018), 2019.

[18] Mr Tobias Adrian and Mr Tommaso Mancini Griffoli. *The rise of digital money*. International Monetary Fund, 2019.

[19] Simon J. Hurry and Alexandre Pierre. Digital fiat currency, 2020.

[20] Centre for the Fourth Industrial Revolution. Central bank digital currency policy-maker toolkit, 2020. Available at http://www3.weforum.org/docs/WEF_CBDC_Policymaker_Toolkit.pdf.

[21] George Danezis and Sarah Meiklejohn. Centrally banked cryptocurrencies. *arXiv preprint arXiv:1505.06895*, 2015.

[22] Karl Wüst, Kari Kostiainen, Vedran Čapkun, and Srdjan Čapkun. Prcash: Fast, private and regulated transactions for digital currencies. In *International Conference on Financial Cryptography and Data Security*, pages 158–178. Springer, 2019.

[23] James Chapman, Rodney Garratt, Scott Hendry, Andrew McCormack, and Wade McMahon. Project jasper: Are distributed wholesale payment systems feasible yet. *Financial System*, 59, 2017.

[24] Darshini Dalal, Stanley Yong, and Antony Lewis. The future is here–project ubin: Sgd on distributed ledger. *Monetary Authority of Singapore & Deloitte*, 2017.

[25] Michinobu Kishi. Project stella and the impacts of fintech on financial infrastructures in japan. 2019.

[26] Chananun Supadulya, Kasidit Tansanguan, and Vijak Sethaput. Project inthanon and the project dlt scripless bond. 2019.

[27] South African Reserve Bank. Project khokha: Exploring the use of distributed technology for interbank payments settlement in south africa. *Pretoria, South Africa*, 2018.

[28] The next-generation mobile payments and banking. Available at https://bakong.nbc.org.kh/.

[29] Alex Tapscott J. Christopher Giancarlo, Don Tapscott. Going cashless: The digital dollar in the face of covid-19, 2020. Available at https://briwebinars.s3.us-east-2.amazonaws.com/Research/Giancarlo-Tapscott_Going+Cashless_Digital+Dollar_Blockchain+Research+Institute.pdf.

[30] Nick Szabo. Smart contracts: building blocks for digital markets. *EXTROPY: The Journal of Transhumanist Thought,(16)*, 18:2, 1996.

[31] Hyperledger iroha. Available at https://www.hyperledger.org/use/iroha.

[32] Quorum whitepaper. Available at https://github.com/jpmorganchase/quorum/blob/master/docs/QuorumWhitepaperv0.2.pdf.

[33] Web3.js - ethereum javascript api, 2016. Available at https://web3js.readthedocs.io/en/v1.2.6/.

[34] Metamask - a crypto wallet & gateway to blockchain apps. Available at https://metamask.io/.

[35] Truffle. Available at https://github.com/trufflesuite/truffle.

Acknowledgements

First of all, I would like to express my humble gratitude to the government of Korea for providing me with the scholarship opportunity to study at POSTECH which is one of the finest universities in the world. I would like to thank my supervisor, Prof. James Won-Ki Hong, for selecting me as a graduate student into his lab DPNM, and for the continuous support and guidance throughout the two years of masters degree program. I would like to thank him for providing me with different research and travel opportunities through academic conferences and seminars. Particularly, in the past few months, he has provided me with immense support and morale to complete this thesis project.

I am also grateful for having wonderful labmates at DPNM. They have created a wonderful lab culture and friendly environment for a foreign student like me. I would like to remember and thank Kyungchan Ko, Chaehyeon Lee and Changhoon Kang for helping me technically with this project and providing me with assistance when required. I would also like to thank Prof. Jongsoo Woo and Dr. Taeyeol Jeong for their valuable comments in improving this thesis.

I would like to thank my father, Radheshyam Maharjan, my mother, Asha Maya Maharjan, and my family for providing me the encouragement and inspiration to pursue master's degree. I would also like to remember my girlfriend, Ana Mihai, for taking care of me during difficult times of my study. I would also like to thank my friends Asmith and Anish for their help in completing this thesis.

Curriculum Vitae

Name : Sajan Maharjan

Research Interest

Central Bank Digital Currency; Machine Learning; Blockchain-based Applications;

Education

2011 – 2015	Department of Computer Science and Engineering, Kathmandu University (B.E. in Computer Engineering)
2018 – 2020	Department of Computer Science and Engineering, Pohang University of Science and Technology (M.S.)

Research/Project Experience

2015. 8. – 2017. 8. Application Developer at LISNepal Pvt. Ltd., Kathmandu, Nepal.

Publications: International Conference

1. Chaehyeon Lee, **Sajan Maharjan**, Kyungchan Ko, James Won-Ki Hong "Toward Detecting Illegal Transactions on Bitcoin using Machine-Learning Methods", 2019 International Conference on Blockchain and Trustworthy Systems (BlockSys'2019), Guangzhou, China, Dec. 7-8, 2019, pp. 520-533.
2. Kyungchan Ko, Taeyeol Jeong, **Sajan Maharjan**, Chaehyeon Lee, James Won-Ki Hong, "Prediction of Bitcoin Transactions Included in the Next Block", 2019 International Conference on Blockchain and Trustworthy Systems (BlockSys'2019), Guangzhou, China, Dec. 7-8, 2019, pp. 591-597.
3. Chaehyeon Lee, Heegon Kim, **Sajan Maharjan**, Kyungchan Ko, James Won-Ki Hong, "Blockchain Explorer based on RPC-based Monitoring System", 1st IEEE International Conference on Blockchain and Cryptocurrency (ICBC 2019), Seoul, Korea, May. 14-17, 2019.

Publications: Domestic Conference

1. **Sajan Maharjan**, Kyungchan Ko, Changhoon Kang, Jongsoo Woo, James Won-Ki Hong, "A Study of CBDC Model Applicable for theCurrent Banking Environment", KNOM Conference 2020, Online, May 15, 2020, pp. 55-60.