PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Research
University of Mohamed Khider – BISKRA
Faculty of Exact Sciences, Science of Nature and Life
Computer Science Department

Order N°: IA15/M2/2021

Master Project

Presented to obtain the academic master's degree in

Computer Science

Option: **Artificial Intelligence (AI)**

Design and implementation of an approach based on Blockchain for big data

---

By:

BOUTARFAIA NEJIA

Defended on 06/27/2022 in front of the jury composed of:

Belouaar Houcine	Associate professor	President
Kazar Okba	Professor	Supervisor
Abdelli Belakcem	Assistant professor	Examiner

Academic year 2021-2022

Acknowledgements

First and foremost, I would like to thank God. He has given me strength and encouragement throughout all the challenging moments of completing this dissertation. I am truly grateful for His unconditional and endless love, mercy, and grace.

*I am so grateful to have professor **KAZAR Okba** as my supervisor. I appreciate how you have consistently encouraged and guided me, as well as how nice and supportive you have been of all of my efforts. Working under your supervision has been very enjoyable and I have learned a lot.*

*I would also like to express my gratitude to all of my **computer science teachers**. I would want to express my gratitude to the members of **the jury** for accepting to judge this work.*

*I am extremely grateful to my parents for their love, prayers, caring and sacrifices for educating and preparing me for my future. I am very much thankful to my sisters **Amira** and **Zineb** for their love, understanding, prayers and continuing support to complete this research work.*

***Meriem**, **Kouather**, **Khadidja**, and **Ibtisam**, as well as all my friends, deserve special thanks for always being a huge source of encouragement when things became a little depressing. Thank you for always being supportive of me.*

"This Master's project is dedicated to Mom and Dad, my two lovely sisters, and my friends. Thank you for always believing in me.."

ملخص

يتزايد حجم وتنوع البيانات التي يتم إنشاؤها كل عام من قبل كل من البشر والآلات. تستخدم البيانات الضخمة الآن في كل مجال تقريبًا، بما في ذلك التجارة والرعاية الصحية والخدمات المالية والحكومة والزراعة وما إلى ذلك. بشكل عام، هناك حاجة كبيرة للبيانات الضخمة في جميع الصناعات والشركات. تثير مشاركة البيانات الضخمة وتحليلها العديد من المشكلات الأمنية وتزيد من مخاطر الخصوصية، نتيجة للامركزية والأمن. على الرغم من النمو الهائل، فإن الأدوات والتكنولوجيا المطورة لإدارة أحجام البيانات الضخمة ليست مصممة للوفاء بمعايير الأمن وحماية البيانات. علاوة على ذلك، يتم توفير غالبية أنظمة أمن البيانات الضخمة الحالية من قبل طرف ثالث مركزي، مما يجعلها مفتوحة لمجموعة واسعة من المخاطر الأمنية. في السنوات الأخيرة، كان تكامل البيانات الضخمة مع تقنية سلسلة الكتل موضوعًا كبيرًا. بسبب اللامركزية والأمن، تتمتع سلسلة الكتل بالقدرة على تحسين خدمات وتطبيقات البيانات الضخمة. والمعلومات الموجودة في الشبكة آمنة ولا يمكن تعديلها؛ أيضًا، قد يؤدي استخدام سلسلة الكتل لتخزين كميات هائلة من البيانات إلى جعل البيانات أكثر تنظيمًا وقيمة. لإثبات تكامل البيانات الضخمة مع سلسلة الكتل، اخترنا مجال البنوك لهذا المشروع لأنه القطاع الأكثر تعرضًا للهجوم ويتطلب تدابير أمنية متقدمة. في هذا الصدد، الغرض من هذا المشروع هو اقتراح حل سلسلة الكتل لإدارة البيانات المالية من خلال استخدام آليات أمان سلسلة الكتل.

الكلمات الرئيسية: *البيانات الضخمة، سلسلة الكتل، الأمن، الشبكة، القطاع المصرفي، اللامركزية، الخصوصية.*

Abstract

The volume and diversity of data created each year by both humans and machines are increasing. Big Data now has uses in almost every area, including commerce, healthcare, financial services, government, agriculture, and so on. Overall, there is a high need for Big Data across all industries and enterprises. Sharing and analyzing Big Data raises several security issues and increases privacy risks, as a result of its decentralization and security. Despite the exponential growth, the tools and technology developed to manage huge data volumes are not designed to fulfill security and data protection standards. Furthermore, the majority of existing Big Data security systems are provided by a centralized third party, making them open to a wide range of security risks. In recent years, the integration of Big Data with Blockchain technology has been a big topic. Because of its decentralization and security, Blockchain has the potential to improve Big Data services and applications. The information in the network is safe and cannot be modified; also, using Blockchain to store vast amounts of data may make the data more structured and valuable. To demonstrate the integration of Big Data with Blockchain, we chose the banking sector for this project since it is the most attacked sector and requires advanced security measures. In this regard, the purpose of this project is to propose a Blockchain solution to financial data management by utilizing Blockchain security mechanisms.

Keywords: *Big Data, Blockchain, Security, Network, Banking sector, Decentralization, Privacy.*

Résumé

Le volume et la diversité des données créées chaque année par les humains et les machines augmentent. Le Big Data a maintenant des utilisations dans presque tous les domaines, y compris le commerce, la santé, les services financiers, le gouvernement, l'agriculture, etc. Dans l'ensemble, il existe un besoin élevé de Big Data dans tous les secteurs et entreprises. Le partage et l'analyse du Big Data soulèvent plusieurs problèmes de sécurité et augmentent les risques pour la vie privée, en raison de sa décentralisation et de sa sécurité. Malgré la croissance exponentielle, les outils et la technologie développés pour gérer d'énormes volumes de données ne sont pas conçus pour répondre aux normes de sécurité et de protection des données. De plus, la majorité des systèmes de sécurité Big Data existants sont fournis par un tiers centralisé, ce qui les rend ouverts à un large éventail de risques de sécurité. Ces dernières années, l'intégration du Big Data à la technologie Blockchain a été un sujet important. En raison de sa décentralisation et de sa sécurité, Blockchain a le potentiel d'améliorer les services et les applications de Big Data. Les informations du réseau sont sûres et ne peuvent pas être modifiées ; De plus, l'utilisation de Blockchain pour stocker de grandes quantités de données peut rendre les données plus structurées et plus précieuses. Pour démontrer l'intégration du Big Data avec Blockchain, nous avons choisi le secteur bancaire pour ce projet car c'est le secteur le plus attaqué et nécessite des mesures de sécurité avancées. À cet égard, le but de ce projet est de proposer une solution Blockchain pour la gestion des données financières en utilisant les mécanismes de sécurité Blockchain.

Mots clés : *Big Data, Blockchain, Sécurité, Réseau, Secteur bancaire, Décentralisation, Vie privée*

Contents

List of Figures

List of Tables

General Introduction

General Introduction

People are getting more digitized and they are analyzing even their regular behaviors with more data and information, nowadays, if a person was only going for a trip, he can know exactly the distance, duration, weather forecasts, traffic congestion, and so on. All of this and more is feasible with important data, and the Big Data landscape appears to be rapidly expanding and infiltrating a wide range of sectors and companies. Every day, we generate trillions of bytes of data. This data is derived from a number of sources, including sensors used to collect climatic data, social media posts, and a variety of others, and as a result, Big Data continues to grow in size.

Big Data applications promise interesting opportunities for many sectors. Extracting valuable insight and information from disparate large data sources enables organizations to improve their competitive advantage. Big Data sharing and analysis raise many security issues and increase privacy threats.

Blockchain is one of the most recent technologies in the domain of security, tractability, and transparency. Following the success of Bitcoin, Blockchain technology has gotten a lot of attention. There have been efforts to harness Blockchain's key properties for various applications and use cases. Blockchain is a decentralized ledger of transactions that is spread among all computers in a peer-to-peer network, with all transaction information available to everyone on the network. Blockchain provides enterprises and consumers with greater confidence in the integrity of the captured data.

Integrating Big Data with Blockchain technology is a certain approach to address and manage some, if not all, of Big Data's core issues and challenges. Due to its decentralization and security, Blockchain has the potential to improve Big Data services and applications. Unlike traditional methods, the information in the network is secure and cannot be duplicated; additionally, using Blockchain to store large amounts of data can make the data structured and valuable.

The financial sector is one of the most susceptible and it needs to be better protected. As a result, Blockchain might be one of the greatest options. The banks put up a trustworthy role in ensuring all the financial transactions performed digitally are sufficiently secured from any future threats. This enables the customers to carry out their banking operations efficiently as they can thoroughly monitor their account activity. The banking sector is yet to find a strong, single technical solution to combat all the security challenges and it is experiencing challenges with data volume and security. Because the volume of data is big, data protection is also required, for this reason, we attempt to employ the Blockchain technique to tackle security challenges.

In our work, we aim to solve banking security issues and storage problems with Blockchain as a solution by developing an application that simplifies bank operations and allows customers to access their data in full transparency and conduct financial transactions with a single click of a button, from anywhere and at any time. Our application's primary objectives are as follows:

- Save a large amount of data;
- Increasing security in the financial industry;
- Data may now be accessed more simply, rapidly, and in real-time;
- Improve data quality by keeping accurate and valuable information;
- Greater traceability and transparency.

Our work is organized in the following manner:

The first chapter describes the fundamental concepts and aspects of Big Data technology, as well as the data life cycle, various application areas, and the most important technologies that allow it to be employed. We describe specific security techniques and related activities, as well as the security difficulties it has faced.

The second chapter will go through the principles of Blockchain technology. First, some Blockchain definitions. The essential characteristics of the technology, as well as the models that may be employed, the process, and the architecture, are then outlined, followed by application areas. Finally, the integration of Blockchain and Big Data, as well as some associated research.

The third chapter describes the proposed architecture design and contribution of the system, as well as the stages of development. The UML sequence diagram is used to explain how processes are carried out.

The fourth chapter addresses the realization and implementation of our system with screenshots of the system.

Finally, we conclude our Master report with a general conclusion and perspectives.

Chapter 1
State of the art of Big Data

1.1 Introduction

"Big Data" refers to massive data sets gathered by businesses and governments that are so vast and complicated that typical data processing methods can't handle the computations required to make sense of the information. Because of the large amount of information buried inside the data structures, these data sets are extremely valuable. Big data can provide more precise insights into hidden patterns, trends, and associations when processed computationally, especially in the context of human decision-making. Furthermore, as the volume of data created by users grows as a result of improved internet connections throughout the world, the issues of handling that data grow as well. As a result, techniques like machine learning (ML) may assist firms in managing and utilizing data supplied by users.

The first chapter aims to introduce the general concepts of big data technology. We start with some well-known definitions, followed by a quick review of the main characteristics of the technology. Following that, the life cycle and different application domains where big data can play a significant role in improving the services have been discussed, as well as the most critical technologies that enable their usage. The focus then changes to the security difficulties it has faced, and we describe some security techniques and related work.

1.2 Big Data definition

It's hard to come up with a precise definition of the term **"Big Data".** For a better understanding, below are several well-known definitions of Big Data:

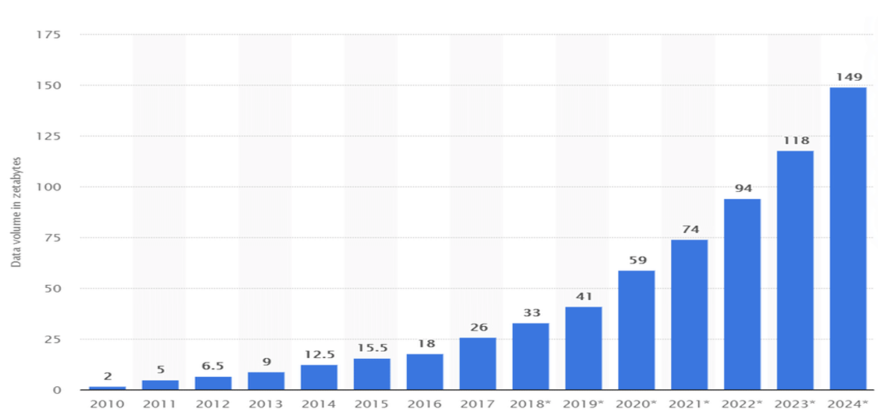


Figure 1. 1 : The volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2024 **[01]**

Big Data was initially defined by Laney **[02]** as: "high volume, high velocity, and high variety of information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making."

This defines Big Data as having three characteristics, or 3Vs, of volume, velocity, and variety. This massive volume of data, as well as its diversity and speed, leads to better decisions and business moves.

Big Data is defined by McKinsey Global Institute **[03]** as: "Big Data refers to data sets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze."

Because the volume and variety of data are so large and diverse, traditional methods and technologies can no longer be used to manage them.

John Gantz **[04]** has described Big Data as: "Big Data technologies describe a new generation of technologies and architectures designed to economically extract value from very large volumes of a wide variety of data by enabling high velocity capture, discovery, and/or analysis."

This definition more accurately describes Big Data as a process of discovering patterns and trends from large amounts of data. It is not just about storage or access to data, but aims to analyze data, understand it, exploit its value, and extract only useful information from the dataset for several uses.

After all these definitions, we can define Big Data as:

"Big Data" refers to the development and application of technology for extracting, analyzing, and interpreting large volumes of structured and unstructured data that are too complicated and expansive for humans or standard data management systems to understand. Companies may take their development to the next level by gathering data in a digital format. Analyzing digital data may help speed up the planning process while also revealing trends that can be utilized to better plan and obtain new insights. The analysis helps them understand patterns in the data that eventually lead to better business decisions. All of this helps to save time, effort, and money.

1.3 Big Data characteristics

Initially, Big Data was defined using a 3Vs model (volume, variety, and velocity). Other organizations extended this 3Vs model to a 4Vs model by including a new "V" value, extending it to 5Vs. More Vs have been developed (7Vs **[05]**, 10Vs **[06]**, 17Vs **[07]**, and even 56 Vs **[08]**) in order to cover the complicated nature of large datasets (validity, volatility, vulnerability, valence, variability, visualization, and many others). We can characterize Big Data using the five principal characteristics of 5 V's (volume, variety, velocity, veracity, and value) as shown in the figure **1.2**:

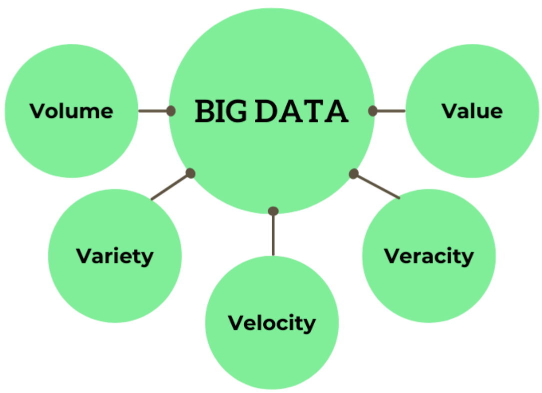


Figure 1. 2: The 5 Vs of Big Data

1.3.1 Volume

In big businesses, a vast volume of data is gathered and created every second. This data comes from a variety of sources, including IoT devices, social media, videos, financial transactions, and customer records. Previously, storing and analyzing such a large amount of data was difficult. However, distributed systems like Hadoop are increasingly being utilized to organize data from all of these sources. This massive volume of data aids in the gathering of reliable and precise information in order to acquire valuable knowledge.

1.3.2 Variety

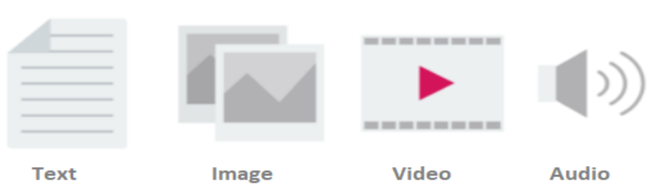
Over time, the data sources have changed. Previously, it could only be found in spreadsheets and databases. Data may now be found in the form of text, music, video, photos, data logs, and other types of data. This data can be classified as structured or unstructured. Many sorts of data may be used to obtain desired information or results.

- **Structured data:** this data is organized in a way that makes it simple to find and evaluate. The data is backed by a model that specifies the size of each field, including its type, length, and value limitations. An example: a bank statement containing the date, time, amount, etc.

Structured Table Data

- **Semi-structured data:** since most semi-structured data seems to be unstructured at first glance, the distinction between unstructured and semi-structured data has always been unclear. Semi-structured data is information that is not in the standard database format as structured data but has some organizational qualities that make it easier to handle. Examples: log files, JSON files, Sensor data, CSV files, etc.

- **Unstructured data:** structured data is the total opposite of unstructured data. There is no pre-defined organizational property or conceptual description for it. The majority of the data is unstructured. Example: text files, emails, images, videos, voicemails, audio files, etc.



1.3.3 Velocity

Represents the speed at which data is generated, gathered, and distributed. The data is sent in a stream and must be examined immediately. The speed at which data is produced is also related to the speed at which it will be processed. This is because the data can only fulfill the expectations of the consumers once it has been analyzed and processed. Fast processing enhances efficiency since some tasks are extremely important and require instant replies. Big Data's main purpose is to supply high-demand data fast.

1.3.4 Veracity

It relates to ensuring the data's quality, integrity, trustworthiness, and accuracy. It is important to filter out the unnecessary information and use the rest for processing. As a result, discovering the right correlations in big data is critical for organizations to effectively use the data and make the best decisions. Building confidence in Big Data is becoming increasingly difficult as the quantity and variety of sources grow. Data quality and privacy are important for correct analysis.

1.3.5 Value

Refers to the process of extracting useful data from a large amount of data. This term describes the degree of importance of data because it provides useful information and knowledge for business. Raw data is initially converted into information by data scientists. The data is then cleansed to extract the most important information. This data collection is subjected to analysis and pattern recognition. The data can be regarded as useful if the procedure is successful.

1.4 Big Data life cycle

Since the emergence of Big Data, data lifecycle management (DLM) has become increasingly important. Massive amounts of data are being generated all around the world by an ever-increasing number of gadgets. To maximize the utility of data and reduce the risk of errors, proper data oversight is required throughout its life cycle. Finally, archiving or removing data when it has outlived its usefulness guarantees that it does not demand more resources than is required.

The "Big Data life cycle" refers to the steps that a piece of data goes through from the moment it is created or captured until it is archived and/or deleted at the end of its useful life. In

2020, the most recent research on the Big Data life cycle is divided into five phases (collection, storage, analytics, utilization, and destruction) by Koo et al **[09]**.

Figure 1. 3: Big Data life cycle **[09]**

1.4.1 Collection

Data is acquired from various sources in various formats, such as structured, semi-structured, and unstructured, during the data collection phase. In an ideal world, Big Data security technologies should be applied first and foremost to the data collection phase of the life cycle. To guarantee that this phase is properly guarded and protected, it is critical to get trustworthy data. Furthermore, extra security measures are required to prevent data leakage. During this phase, various security measures, such as limited access control and data encryption, can be implemented. Furthermore, data can be gathered without the agreement of the data source through software, social media, and the internet **[09].**

1.4.2 Storage

The obtained data is saved in the data storage phase for use in the following step. Because the obtained data may contain sensitive information, it is critical to implement effective data storage protections. By integrating physical security approaches with data protection technologies, the stored data must be safeguarded from different risks. Data integrity and confidentiality must be protected by privacy-preserving technology in circumstances when it is not totally dependable, such as in the cloud (e.g., encryption and masking). Because data is so large, data storage services must conform to a distributed storage model, and sensitive data must be made available only to authorized individuals via access control. Furthermore, if sensitive data is accidentally shared without consent during the collection process, it must be erased right away **[09].**

1.4.3 Analytics

The data is processed and evaluated once it has been collected and stored in order to provide relevant knowledge. Clustering, classification, and link rule mining are some of the data mining techniques utilized in this stage. It's critical to provide a secure environment for processing and analysis throughout the analytics phase. Data miners can use strong mining algorithms to find sensitive data, making their systems vulnerable to privacy invasion. As a result, the data mining process and analysis findings should be safeguarded against mining-based assaults, and only authorized individuals should be able to access them **[09].**

1.4.4 Utilization

The analytics phase provides decision-makers with fresh information and important insights. In a competitive climate, this knowledge is considered sensitive. In order to compete against their commercial rivals, companies usually take extra precautions with highly sensitive information. Furthermore, they diligently guarantee that important customer personal information is not made public. The major goal of the usage phase is to create new information through a mix of sensitive and non-sensitive information analytics. Even if no sensitive data is present, combining data from other areas can aid in identifying a specific individual or inferring sensitive data, and this information can be used for other reasons without authorization **[09].**

1.4.5 Destruction

Data utilized for analysis is destroyed during the data destruction phase. Unless otherwise indicated in other laws and regulations, private data should be erased immediately after the data retention period has expired. The data must be destroyed in accordance with the data's purpose and the user's withdrawal of consent. Despite fulfilling their primary objective and withdrawing their consent, some organizations continue to use the data. More specifically, privacy issues develop as a result of the act of selling data to third-party companies. Furthermore, because of the nature of Big Data architectures, data may be unable to be removed because it is destroyed in a distributed context **[09].**

1.5 Big data technologies

We have a variety of tools to process large amounts of data. Some of the tools we are discussing here are **Hadoop**, **MapReduce, Spark,** and **Storm**.

1.5.1 Hadoop

Apache Hadoop is a batch processing technology with fault tolerance and scalability. Hadoop can handle petabytes of data and execute programs across numerous nodes. In addition, the log data is divided into blocks and delivered to the Hadoop cluster's nodes. Hadoop is also popular because of its quick retrieval, searching log data, scalability, quick data insertion, and fault tolerance capabilities. The Hadoop Distributed File System (HDFS) is a GFS-based file system with a distributed architecture that is optimized for commodity hardware. Although it has many similarities to earlier distributed systems in terms of functionality, the main distinction is that HDFS is built on low-cost hardware and is meant to deliver high throughput to application data, making it particularly appropriate for applications with massive datasets **[10].**

1.5.2 Spark

In 2009, the University of California, Berkeley created Apache Spark as a unified architecture for distributed data processing. Spark adds a data sharing concept called "Resilient Distributed Dataset" to the MapReduce architecture (RDD). With this extension, Spark can capture and process workloads such as SQL, streaming, machine learning, and graph processing. It has many distinguishing characteristics, including in-memory processing and an in-memory data-caching function. For processing massive amounts of data, Spark supports multiple programming languages (Python, Java, and Scala). It can analyze large amounts of data in a short amount of time. It is 100 times quicker than Hadoop and MapReduce at processing data. One of Spark's drawbacks is that memory consumption issues are difficult to manage in a user-friendly manner. It needs a larger amount of RAM than other big data systems **[11].**

1.5.3 Storm

Nathan Marz had an idea in December 2010 to create a stream processing system that could be delivered as a single program. Storm, a new project based on this concept, was born as a result of this notion. On September 17, 2014, Apache Storm joined the Apache Family. Apache Storm is a real-time calculation system that is open source. Storm makes it simple to process data streams in real-time. Furthermore, each node is capable of processing millions of tuples per second. Storm is a user-friendly, scalable, fault-tolerant, and speedy system. It also has the ability to integrate databases into the processing **[12].**

1.5.4 MapReduce

Google defined MapReduce in 2004 as a method for distributing data workloads across thousands of nodes. It's a programming model that uses a parallel, distributed method to process and produce large data sets across a cluster of computers. It can be used with Hadoop and other similar platforms. The "map" component distributes the programming problem or task across a large number of systems while managing placement to balance the load and allow recovery from failures. After the distributed computation is complete, another function called "reduce" aggregates all the elements back together to provide a result. Multiple computers in a system can perform this process at the same time to quickly process data from the raw data lake to usable findings **[13].**

Table 1. 1: Comparison of Big Data analytics tools [14]

Key Functions	Tools	Features	Strengths	Weaknesses
Data storage management	**Hadoop distributed file system (HDFS)**	Used for storage for high volumes of data. It is reliable and faults tolerant	Enable data to be read once and written many times with less expensive data storage.	Lack of the ability to efficiently support random read of a small amount of data. In addition, it difficult to manage Hadoop clusters.
Big database management	**Apache Spark**	Hadoop tools for real-time processing and machine learning.	Efficient for a reading/write operation, batch processing, join streams and ability to handle failures of any worker nodes. Furthermore, Spark support implementation using multiple and commonly used programming languages with built-in App.	Challenging to provide real-time processing. Also, have a problem processing small dataset and require manual optimization for a specific dataset.
Big Data processing	**Apache Storm**	Tools for online machine learning, real-time data analytics for analyzing a large amount of real-time data, streaming and real-time processing	Efficient for non-complicated streaming operation, low latency, and high throughput streaming operation	Lack of advanced features for event time processing, data aggregation and implicit support for state management.
	MapReduce	Hadoop distributed programming framework for batch processing, resources scheduling and compute job management.	Highly scalable due to the ability to store a large volume of distributed data and also cost-effective	Inability to handle interactive, in-memory and graph processing. In addition, map reduce are not configured for small dataset.

1.6 Domains of application

Big Data has gained interest and application in a variety of industries in recent years. Every sector of research, whether in industry or academia, generates and analyzes vast amounts of data for a variety of purposes, such as business, healthcare, education, agriculture, research, transportation, and banking. The growing interest in Big Data is due to the fact that it gives companies significant knowledge upon which they may make judgments, suggestions, and decisions.

Above are a few big data application examples:

1.6.1 Business and commerce

The use of Big Data will become an important component of competition and growth for companies that require a comprehensive understanding of their customers. Businesses can function much more efficiently by analyzing vast amounts of data, which can then be used to raise income or create new revenue streams. Big Data provides organizations with the resources they need to make better decisions, future decisions, and decisions that matter, as well as to better understand their consumers and make appropriate suggestions based on that information. Companies will be able to build better goods or services to meet their demands **[15].**

1.6.2 Healthcare

In recent years, the healthcare sector has seen rapid growth. As a result, a large amount of data has grown. This data is difficult to manage using standard data management tools and methodologies. The goal of Big Data solutions in the health sector is to use these massive, complex data sets in order to obtain more focused information and insights. Using systems capable of dealing with such "Big Data" will provide the healthcare sector with several potential benefits, including the ability to save lives, reduce costs, and increase operational efficiency **[15].**

1.6.3 Education

Big Data may help to improve education by adding new learning methods to the system, making it more efficient and targeted. Teachers have useful and convenient tools with which they can assess their own performance, improve the classroom atmosphere, and greatly expand learning opportunities. Students can also better grasp prospective future occupations and sectors that fit their unique strengths and inclinations due to Big Data recommendations. Data about students and teachers can lead to new insights that may help schools improve their operations, and it is possible to use Big Data to obtain a much greater understanding of how to ensure students are successful **[15].**

1.6.4 Scientific Research

Many scientists now generate massive amounts of data, which makes scientific study and discovery more data-intensive. Big Data has prompted scientists to rethink their research methods, resulting in a revolution in scientific thought and action. Big Data's advent has produced a new research paradigm in which researchers may just need to locate or mine the needed information and knowledge from it. They don't even require direct access to the material they're studying **[15].**

1.6.5 Banking

The banking market and consumers who utilize financial products generate an enormous amount of data daily. Big Data analytics is currently being adopted throughout many areas of the banking industry, and it is assisting them in providing better services to their internal and external consumers, as well as improving their active and passive security systems **[16].**

1.6.6 Transportation

Big Data has the potential to change the transportation industry in a variety of ways. Traffic congestion consumes energy, contributes to global warming, and costs people time and money in many nations. Real-time traffic data may be evaluated and shared via distributed sensors on mobile devices, cars, and highways. This data, when combined with more autonomous capabilities in automobiles, can help drivers drive more safely and with less disturbance to traffic flow **[17].**

1.6.7 Agriculture

In the world of agriculture, Big Data is a pioneering development. Intuition, intelligence, and insights are all produced as a result of this process. Traditional tools are being replaced by sensor-equipped machines that can collect data from their environments to control their behavior, such as thermostats for temperature regulation or algorithms for implementing crop protection strategies. The rapid growth of smart farming is aided by the use of technology in conjunction with external Big Data sources such as meteorological data, market data, and farm standards **[18].**

1.6.8 Weather

The ability to understand changes in the frequency, intensity, and location of weather and climate can benefit millions of citizens and thousands of businesses that depend on the weather, including farmers, tourism, transportation, and insurance companies. Meteorological forecasters' capacity to predict the time and intensity of hurricanes, floods, snowstorms, and other weather disasters has substantially improved because of dramatic improvements in the ability to acquire and analyze data **[19].**

1.7 Challenges in big data security and privacy

It is clear that Big Data has had a significant impact on business development and success. It's even better because its good influence is noticed in every sector throughout the world. Big Data analytics is being used by businesses to find business opportunities, enhance performance, and guide decision-making. As a consequence of the massive increase in data consumption, many Big Data technologies are free and open-source, and they aren't created with security in mind. As a result, various data security issues exist.

Big Data challenges include the best way to handle large amounts of data, which includes the process of storing and analyzing large quantities of data across several data sources. When dealing with Big Data, there are several problems that must be addressed with agility. The following is a list of the three most prevalent Big Data security problems.

1.7.1 Storage and management challenges

Organizations and companies' storage systems are experiencing severe issues from massive amounts of data and the ever-increasing amount of created data as the rate of data explosion accelerates. Data, regardless of its size, is critical to the industry. Large data sets may be used to produce value. Facebook, for example, boosts ad income by mining users' personal preferences and constructing profiles that show marketers whose things they are most interested in. Google also profiles users' behavior using data from Google Search, Google Hangouts, YouTube, and Gmail accounts. Big Data has outgrown its present infrastructure, pushing the storage capacity and storage network to their limits. Due to the massive quantity of data, existing standard methodologies are unable to support and perform effective analysis **[20].**

1.7.2 Transmit and sharing challenges

The massive volume of Big Data, as stated previously, increases transmission time and raises the chance of personal data being compromised. Two strategies resolve themselves in order to limit data risk during transmission. Put the Big Data "in place" first, and then just provide the analysis results. In other words, rather than bringing data into code, the client delivers code to the data. Second, categorize data and send just the information needed for downstream analysis. Integral and originating metadata are meant to communicate alongside the actual data in these two cases **[20].**

1.7.3 Analytical challenges

The structure of the data, which comprises structured, semi-structured, and unstructured data, influences Big Data analysis. Big Data, regardless of its form, has volume characteristics. As a result, the scalability of Big Data poses the greatest barrier in data analysis. Data encryption is a good approach to protect privacy while working with Big Data. However, the massive volume of data only adds to the security burden. Furthermore, the danger of data analysis is increased by the dispersed storage of Big Data. The demand for timeliness for data stream analysis is a greater challenge to encryption techniques and security protection for real-time Big Data applications, such as social networks, intelligent transportation, navigation, stock analysis, and so on **[20].**

1.8 Techniques for big data security and privacy

Big Data has been increasingly adopted in almost all sectors of the industry over the past decade, and much work has focused on developing new technologies for Big Data analytics. Companies store numerous types of data from diverse sources; therefore, security must be prioritized since practically every organization that uses Big Data contains sensitive data that must be protected. Credit card numbers, financial information, and passwords are examples of sensitive data.

The security and privacy of Big Data concepts are some of the most pertinent issues associated with Big Data in the modern world. There have been concerned efforts geared towards the development of robust techniques aimed at solving the various security challenges in Big Data.

Research has mostly focused on the cloud computing environment, which can enable Big Data storage and analytics because of its cost-effective scalability, flexibility, resource pooling, and data processing properties. Cloud computing, on the other hand, raises security and privacy concerns **[21].**

Big Data environments add another layer of protection because security tools must operate during three data stages that are not all present in the network: data ingress (what comes in), stored data (what is saved), and data output (what goes out to apps and reports).

Encryption is a technique for converting understandable data (such as plaintext) into an unreadable one (e.g., ciphertext). This is done to guarantee that only authorized users have access to the information. It refers to the process of using a mathematical procedure to turn plaintext into ciphertext. Decryption, or converting ciphertext to plaintext, is only possible with a set of encryption keys **[09]**. The aim of encryption is to ensure that the stored data or the data in transit cannot be accessed by unauthorised parties. Strong encryption algorithms are required to ensure that data access is only performed according to the predefined control policies **[22]**. The most common encryption methods are: **ABE** and **RSA**.

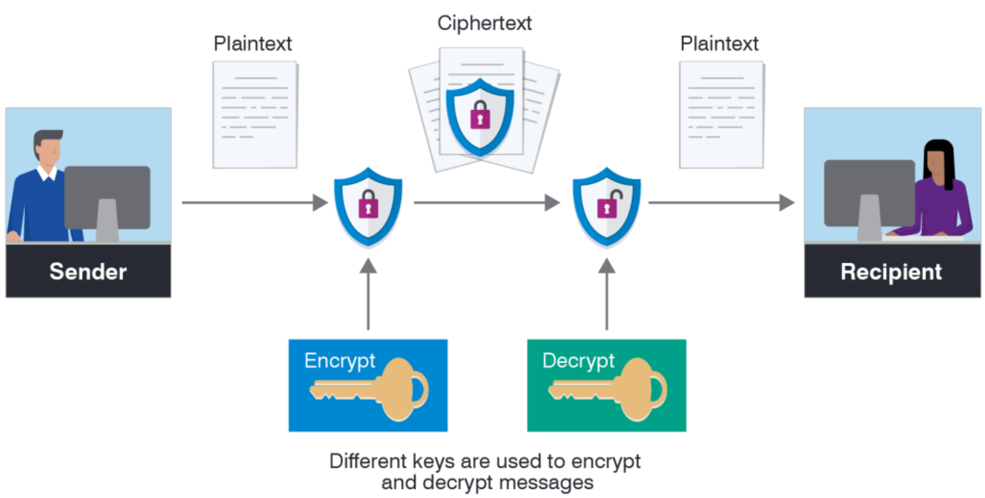


Figure 1. 4: The encryption and decryption process **[23]**

- **ABE:** Attribute-Based Encryption is a strong and promising public key encryption system in which the cipher text and key are determined by characteristics. Amit Sahai **[24]** was the first to propose the idea of attribute-based encryption. For writing log secrets, ABE will be utilized. Encrypting the properties that match receivers' attributes will eliminate the need to encrypt every element of the log **[25]**. Since its beginning, ABE has been a very active field of cryptographic analysis. Chi Lee **[26]** has made a good comparison of raw ABE, CP-

ABE, KP-ABE, HABE, and ABE with non-monotonic structure. To complete ABE, there are four essential phases **[25]**:

1. Setup: A master is generated by the authority mistreatment the attributes and public key's published.
2. Key Generation: The authority generates a personal key for users. The non-public key's generated with secret sharing. and therefore the secret attributes are embedded within the parts of user's non-public key. Conjointly the key's related to the random polynomial. So, each user's non-public key cannot be combined with a brand new non-public key to perform the collusion attack.
3. Encryption: Owner encrypts the info employing a set of attributes and therefore the message is revealed. A random variety is chosen to avoid user decrypting the info once the primary decrypting, when he infers the quantity.
4. Decryption: information user will rewrite the info with a non-public key.

- **RSA:** Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman developed a cryptographic algorithm in 1978 to replace the less secure National Bureau of Standards (NBS) method. The most important feature of RSA is that it includes a public-key cryptosystem as well as digital signatures. RSA was inspired by the earlier published works of Diffie and Hellman, who discussed the concept of such an algorithm but never fully implemented it **[27]**. RSA is a robust encryption method that has stood the test of time in part. The security of the RSA public-key cryptosystem, which permits secure communications and "digital signatures," is based in part on the difficulty of factoring huge numbers. Two secret keys are used in public-key cryptography (i.e., the public key and a private key). Anyone with a public key can encrypt data, but only a private key can decode it. It is not used to directly encrypt sensitive data due to its slow speed, but it is commonly used to transmit the secret key of symmetric key cryptography **[09]**. The RSA algorithm's phases **[28]**:

1. First of all, on receiver system, private key isgenerated by receiver. For this, he selects any twoprime Numbers. Receiver also choose RSA publickey and generate his private key. Now at receiverend private key is generated.
2. If any sender wants to send message to receiver, first of all he has to obtain his public key which receiver has used or sending message. This key is distributed by the receiver over the network.

3. Sender who sends the message use receiver' public key and decrypt the message. In this case, he is not encryption of text message, first of all plaintext message is converted into bits form i.e.0010000100010111001001 then sender encryptthe message using public key for encryption. After encryption, send the crypted file over network with authenticate receiver.
4. At receiver end if any unknown person tries to access the data, due to encryption, he is not able to encrypt the message because he does not know about the private key. Private key is not distributed over network. So not possible for unauthorized user to decrypt the message.
5. When authorized receiver tries to see the actual message from encrypted one then he has to use his private key for encryption. First it verifies the user then show him the encrypted message. Using private key when he tries to decrypt the message, first message is convert into bits form as security is done at physical layer. From that bits form, original data is extracted. Thus receiver is able toread the message.

1.9 Conclusion

Big Data applications promise interesting opportunities for many sectors. Extracting valuable insight and information from disparate large data sources enables organizations to improve their competitive advantage. However, Big Data sharing and analysis raise many security issues and increase privacy threats. This chapter presents some of the important Big Data security challenges and describes related solutions and recommendations. because securing huge data sets is almost impossible. For cloud storage, decentralization, distrust, and breaking information asymmetry are future directions of the cloud environment, such as the *Blockchain* scheme that is used for bitcoin transaction storage. While Blockchain technology is still relatively new, it is starting to have an impact on the world and how much data is processed. As more data is collected in real-time, it will be interesting to see how the Blockchain will continue to revolutionize different industries and bring better data privacy. In the next chapter, we'll take a look at what Blockchain is and how it can help secure massive amounts of data.

Chapter 2

Blockchain Technology

2.1 Introduction

The Blockchain's purpose is to produce a distributed, decentralized, and trustworthy record of the system's history. This record is used by the most well-known Blockchain, Bitcoin, to keep a history of transactions, allowing consumers to send and receive payments on the Bitcoin Blockchain while remaining certain that their money will not be lost or stolen. Following the success of Bitcoin, Blockchain technology has gotten a lot of attention.

There have been efforts to harness Blockchain's key properties for various applications and use cases. Integrating Blockchain with Big Data has many advantages because it enables better management of enormous volumes and varieties of information.

The general concepts of Blockchain technology will be presented in this chapter. We will start with some Blockchain definitions. Then we outline the technology's major characteristics, the models that may be used, the process, and the architecture, followed by application areas. The discussion then shifts to the relationship between Blockchain and Big Data, as well as some related research.

2.2 Blockchain Evolution Timeline

Blockchain technology made its public debut in 2008 when Satoshi Nakamoto published the whitepaper Bitcoin: A Peer to Peer Electronic Cash System, which described Bitcoin as a "purely peer-to-peer version of electronic cash." Bitcoin's Blockchain technology has evolved over the previous decade into one of today's most ground-breaking technologies, having the ability to touch every industry from finance to manufacturing to education. Here's a brief overview of the history of Blockchain technology **[29]**.

Table 2. 1: Blockchain Evolution Timeline [29]

Timeline	Blockchain	Bitcoin	Ethereum	NEO
1991-2008	Stuart Haber and Scott Stornetta Work on The First Blockchain			
2009		Satoshi Nakamoto Releases Bitcoin White Paper		
2010		The First Bitcoin Purchase 10,000BTC take place		
2013		Bitcoin Marketplace Surpasses $1 Billion	Vitalik Buterin Releases Ethereum White Paper	
2014			Ethereum Blockchain Is Funded By Crowdsale	
2014	Blockchain Technology R3 is Formed and forms a Consortium of Over 40 Legacy financial for implementing Blockchain Technology			
2014				NEO Project Is Launched as Antshares by Da Hongfei and Erik Zhang
2015			Ethereum Second Blockchain Is Unveiled	
2015	Linux Foundation Unveils Hyperledger To Enhance Blockchain development			
2017	EOS.IO is Unveiled by block.one as a new Blockchain protocol for the deployment of decentralized applications			
2015-2018	Blockchain Technology Continues To Evolve Depicted by increased number of cryptocurrencies as well as Companies leveraging the Technology To enhance Efficiency			

2.3 Blockchain technology

Blockchain is a decentralized ledger of transactions that is spread among all computers in a peer-to-peer network, with all transaction information available to everyone on the network. This is a developing list of interconnected blocks. Valid transactions, a timestamp, and a hash pointer as a link to the preceding block in the chain make up the blocks. A secure, interdependent chain is formed by a series of connected units. To be included in the Blockchain, blocks must first be verified. When a block is validated, it is distributed (added to the current chain) across the network, with each node adding the block to the majority Blockchain **[30].**

Blockchain is a decentralized, distributed directory that powers smart contracts and allows for traceability, record management, supply chain automation, payment applications, and other types of commercial operations. Blockchain is an immutable record of virtually real-time replication amongst a network of business partners **[31].** Because of the public confidence it fosters, Blockchain allows for the removal of intermediary third parties. Transactions may also be fine-tuned and confirmed at a granular level. This facilitation allows for a substantial shift in the transaction verification architecture. Such transactions provide for more accurate information trails in the various information systems **[32].**

Three data properties are available in Blockchain that are rarely available in traditional centralized data management **[30]:**

- **Security:** when Blockchain is employed, it is nearly difficult for data to be modified or damaged.
- **Integrity:** Blockchain data offers audit trails, certainty of origin, and digital signature of the message, also provides integrity through the transfer.
- **Value:** the data generated by Blockchain is complete, and its worth is undeniable.

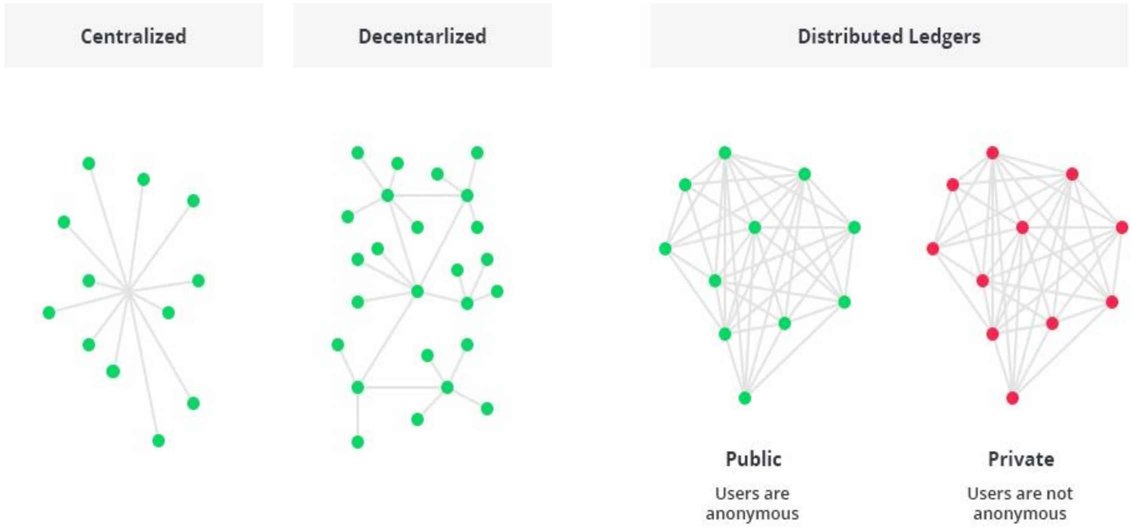


Figure 2. 1: Centralized, Decentralized, and Distributed Ledger **[33]**

2.4 Characteristics of Blockchain

Despite its original association with Bitcoin, Blockchain technology may be used in a wide range of applications and businesses, including insurance and healthcare. A Blockchain may be used in almost any sector that manages assets and conducts transactions. Its functional characteristics that support transactions through trust, consensus, security, and smart contracts can create a safe chain of custody for both digital and physical assets.

The next sections go through these characteristics of Blockchain which was mentioned in the article **[34].**

- **Multicenter**

Because the Blockchain uses distributed, decentralized storage, data may be recorded, stored, and updated without the use of a single central point. Because there is no centralized hardware or management organization, any node can act on the Blockchain's data in accordance with the rules.

- **Transparency**

The Blockchain's system data is open and transparent, and any node may see the network's general ledger. The Blockchain data are available to all nodes, with the exception of the private information of the data's directly linked parties, which is protected using asymmetric encryption technology. As a result, the entire system information is very open and transparent.

- **Autonomy**

There are several users in the Blockchain system, and they have automatically established specifications and protocols based on open rules and algorithms. During operation, each node in the system follows these requirements and protocols. This means that in a trustless system, every transaction can be guaranteed to be correct and legitimate. Data can be safely exchanged, recorded, and updated between the nodes, and actions that do not meet the standards and protocols will fail.

- **Immutability**

There is a complete backup locally on each node once the Blockchain transaction information passes through the consensus of all nodes and is recorded in the block. The hash algorithm performs the correlation between blocks at the same time. If you wish to change a piece of data, you must change all the following blocks as well, which is highly expensive.

- **Traceability**

Each Blockchain node preserves all of the previous entries. By traversing the local Blockchain data, any piece of data may be located, making every data on the Blockchain chain traceable.

- **Programmability**

The Blockchain's nature enables a trusted application environment for the execution of smart contracts, allowing users to manipulate data in a programmable manner. Smart contract rules can be customized to match the demands of the user. At the same time, it ensures the security of assets and data on the chain owing to its open and autonomous execution features.

2.5 Types of Blockchain

Private and public Blockchain are the two main types of Blockchain. There are, however, certain variants, such as consortium and hybrid Blockchain.

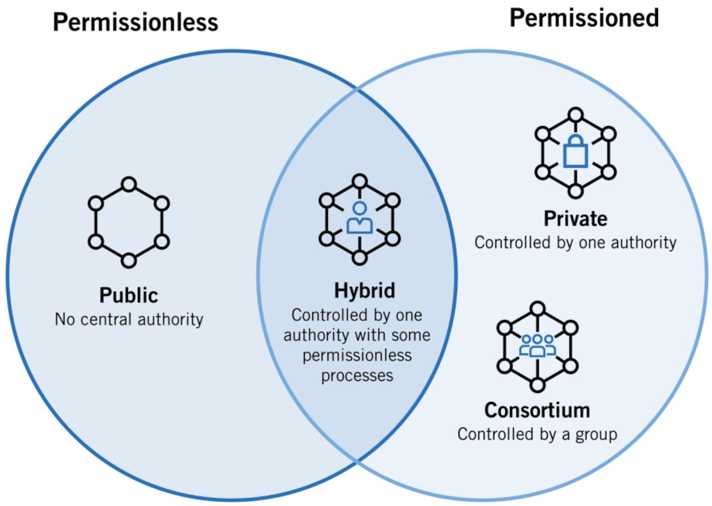


Figure 2. 2: Blockchain types **[35]**

2.5.1 Public Blockchain

A public Blockchain is a non-restrictive, permissionless distributed ledger system. Anyone with an internet connection may sign up for a Blockchain platform and join the network as an authorized node. Viewing current and historical information, validating transactions, doing proof-of-work for an incoming block, and mining are all permitted by a public Blockchain node or user. The most basic use case for public Blockchain is bitcoin mining and trade. As a result, the public Blockchain Bitcoin and Litecoin are the most extensively utilized. Public Blockchain are relatively safe if users follow security rules and standards. It is only dangerous when the participants do not follow the security guidelines **[36]**, as examples: Bitcoin, Ethereum, Litecoin.

2.5.2 Private Blockchain

A permissioned or limited Blockchain that may only be utilized in a closed network is known as a private Blockchain. Private Blockchain are often used within a corporation or organization where only a small number of people are permitted to join in a Blockchain network. The level of security, authorizations, permissions, and accessibility is determined by the controlling organization. As a result, private Blockchain are functionally equivalent to public Blockchain, but their network is smaller and more restricted. Private Blockchain networks are used in voting, supply chain management, asset ownership, and other applications **[36]**, as examples: Multichain, Hyperledger projects, Corda, etc.

2.5.3 Consortium Blockchain

A consortium Blockchain is a semi-decentralized kind in which many entities control a Blockchain network. A private Blockchain, on the other hand, is controlled by a single entity. More than one company can act as a node in this type of Blockchain, sharing information or mining. Consortium Blockchain are often used by banks, government organizations, and other entities **[36]**, as examples: Energy Web Foundation, R3, etc.

2.5.4 Hybrid Blockchain

The benefits of both private and public Blockchain are combined in a hybrid Blockchain. It combines the benefits of both private and public Blockchain, allowing for both private and public permission-based systems. A hybrid network like this allows users to select who has access to which data stored on the Blockchain. Only a part of the Blockchain's data or records can be made public, with the rest of the network remaining private. Because of the hybrid Blockchain system's versatility, users may easily combine a private Blockchain with several public Blockchain. A transaction on a private network of a hybrid Blockchain is normally confirmed inside that network. Users can, however, validate it by posting it on the public Blockchain. The hashing power of public Blockchain has increased, and the verification process now includes more nodes. This increases the security and transparency of the Blockchain network **[36]**.

Table 2. 2: Public Vs Private Blockchain [37]

Property	Public Blockchain	Consortium Blockchain	Private Blockchain
Consensus determination	All miners	Selected set of nodes	One organization
Read permission	Public	Could be public or restricted	Could be public or restricted
Immutability	Nearly impossible to tamper	Could be tampered	Could be tampered
Efficiency	Low	High	High
Centralized	No	Partial	Yes
Consensus process	Permissionless	Permissioned	Permissioned

2.6 How does Blockchain technology work

Individual transactions and blocks are the two types of records that make up a Blockchain ledger. The first block contains a header and details about transactions that occurred within a specific time frame. The timestamp of the block is used to construct a hash, which is an alphanumeric string. Following the creation of the initial block, each successive block in the ledger calculates its hash using the previous block's hash. A computational method known as validation or consensus must be used to verify the validity of a new block before it can be added to the chain. A majority of nodes in the network must agree that the new block's hash has been calculated correctly at this point in the Blockchain process. Consensus guarantees that all copies of the distributed ledger Blockchain are in the same state. A block can be referenced in succeeding blocks, but it cannot be changed after it has been added. If someone attempts to swap out a block, the hashes for the previous and following blocks will change as well, causing the shared state of the ledger to be disrupted. When consensus is no longer achievable, other computers in the network are alerted, and no new blocks will be added to the chain until the issue is resolved. The block that caused the problem is usually removed, and the consensus process is restarted **[38].**

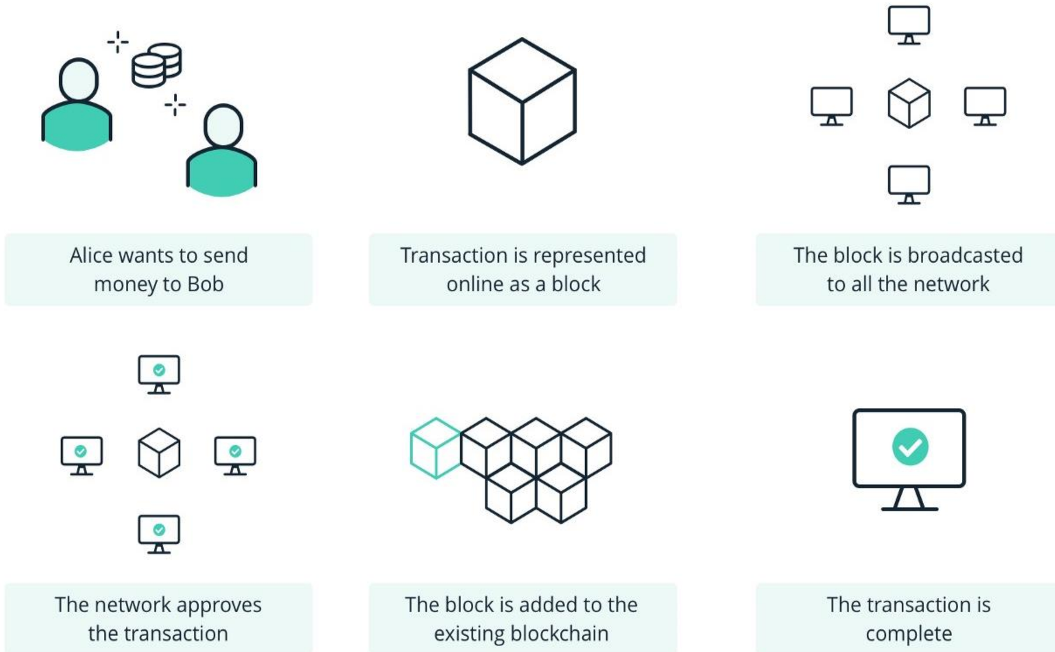
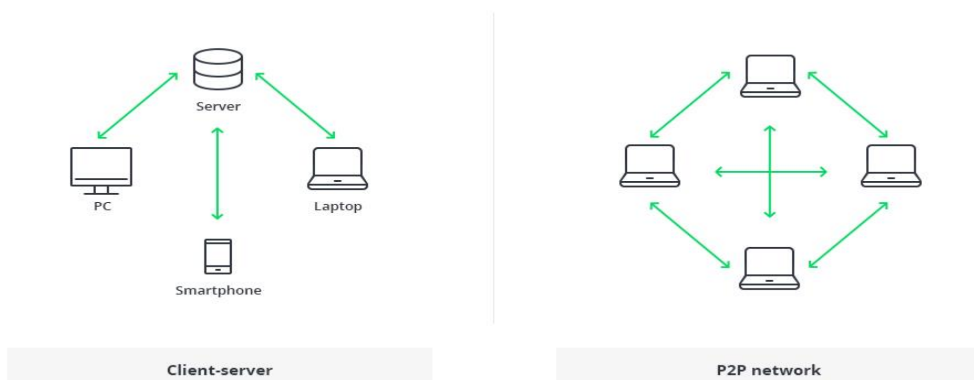


Figure 2. 3: The process of Blockchain **[39]**

2.7 Blockchain architecture

In the financial industry, Blockchain architecture is widely employed. These days, however, this technology aids in the creation of software for cryptocurrency and record keeping, as well as digital notaries and smart contracts



.

Figure 2. 4: Client-Server vs P2P Network **[33]**

The World Wide Web's conventional architecture is based on a client-server network. Because the server is a centralized database maintained by a number of administrators with rights, it stores all of the needed information in one place and makes it easy to update. In the case of Blockchain systems, the trust is decentralized. Users just need to trust the system and the smart code that is shared between all the participants. From a technical point of view, a Blockchain is a distributed database that exists on a P2P network. This P2P network is the backbone of the system because every node in the network is on the same level as all the other nodes. Although nodes can come in many forms, there is no central node that is an authority. Every node stores a local copy of the Blockchain. If a consensus of nodes agrees upon the transaction's validity, then the transaction is considered valid **[33]**.

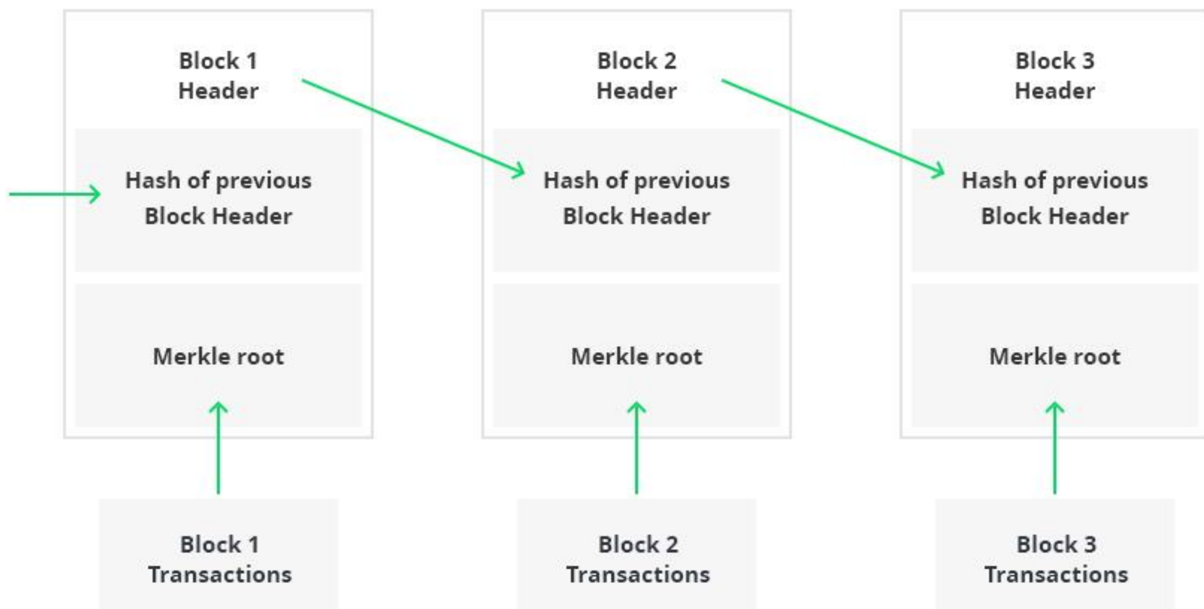


Figure 2. 5: Blockchain Structure **[33]**

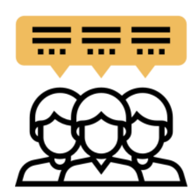
The data in a Blockchain is formatted differently from that of a traditional database. A Blockchain organizes data into blocks, whereas a database organizes data into tables. Each transaction on the network is added to these blocks, which have a limited storage capacity.

Once a block is full, a new block is appended to the previously filled block to include new transactions, producing the Blockchain, which is a chain of blocks.

As a result, Blockchain are decentralized and do not store data in a single location. Instead, they are stored on network nodes or computers. Every node has a copy of the Blockchain, which contains all of the network's transactions. As a result, every single node in the network is backing up the system **[33]**.

The following table will outline the basic components of a Blockchain:

Table 2. 3: Core Blockchain component

Component	Definition	
Node	Every entity or machine that processes and validates transactions is a node in this network. Every node has a copy of the whole Blockchain database, which includes all transaction details.	
Transactions	A transaction in a Blockchain network is a small unit of labor that is recorded in public records. The majority of the network's members must agree in advance to the execution and storage of these entries on the Blockchain. Any previously completed transaction can be seen but not modified at any time. The transactions that are continuously made by nodes and aggregated in blocks indicate the current status of the Blockchain.	
Block	Blocks are the elements of a Blockchain network that package and distribute a group of transactions to each node. Miners are the ones that create these blocks. Every block contains data as well as a few transactions that miners have selected. The block header contains metadata about a block that nodes use to validate the data in that block.	
Miners	Blockchain mining is the process of miners on the network adding new blocks to the Blockchain. In other words, "mining" is the process of creating Blockchain in a secure and decentralized manner.	
Consensus	The nodes will have no centralized party to monitor and resolve disputes or protect against attacks once data transmission over a Blockchain network begins. As a result, a mechanism to trace the transfer of funds and ensure an incontrovertible fund exchange is required to prevent fraud. All nodes must agree on a similar content update process for this ledger to maintain consistency, and blocks should not be permitted into the Blockchain without majority consent.	

2.8 Applications and use cases of Blockchain

After the successful implementation of Blockchain in Bitcoin because of its unique features, while the concept works extremely well for Bitcoin and other cryptocurrencies, there are loads of other useful applications of Blockchain technology. In the next part, we will give a quick summary of each domain which was mentioned in the article **[40]**.

2.8.1 Finance

Traditionally, financial transactions are verified and processed by an intermediary, such as a bank. Having such a centralized system places a lot of labor in the hands of intermediaries, and the transactions are prone to mistakes since several uncoordinated parties must keep track of and change the records. As a result, the entire procedure is time-consuming and expensive. By creating a distributed public ledger, where transactions are validated by miners using "proof-of-work," the Blockchain lowers the complexities involved with financial services **[40]**.

2.8.2 Blockchain Government

Different government organizations and units may use Blockchain technology to develop trustworthy and successful government operations through collaborative and transparent networks. The key qualities of Blockchain technology will aid accountability, transparency, and confidence among stakeholders such as people, leaders, and government officials, as well as their various activities. To address the accountability of its bodies, the government must make its operations open. To do so, the government may have to make a large quantity of data available to the public **[40]**.

2.8.3 Internet of Things (IoT)

Every year, the number of electronic devices linked to the Internet grows rapidly. The Internet-of-Things is created when a large number of devices are connected. The Internet of Things is expected to alter people's lives, making notions like smart houses a reality. While having a large number of heterogeneous devices connected to the Internet is expected to make life simpler, it also raises serious concerns about cyber security and privacy. Blockchain has the potential to be a significant technology for securing the IoT. With millions of devices linked and interacting, it's critical to guarantee that the data flowing through IoT is safe and that participants are held accountable **[40]**.

2.8.4 Cybersecurity

Another use of Blockchain is in cybersecurity to prevent future attacks, where dangerous information may be shared across participants/organizations utilizing Blockchain to prevent future cyber-attacks. For example, different companies or governments are reticent to share cyber-attacks or threat intelligence with one another since rivals might use the knowledge unilaterally if it is supplied with identifying data. However, by using Blockchain and a public-private key combination, information may be transferred without disclosing any personally-identifying information other than the public key. This allows businesses or governments to exchange threat information without fear of rivals abusing the knowledge to gain an unfair advantage unilaterally **[40]**.

2.8.5 Intellectual Property

Blockchain technology might be used in an intellectual property management system to enforce demonstrable intellectual property rights, where verifiable, irreversible, and secure activities in Blockchain could aid with any conflicts **[40]**.

2.8.6 Blockchain Health-Care

Personal health records are delicate data that must be handled with extreme caution. Personal records may be encoded and saved using Blockchain, with a private key allowing only particular users to access the information. Similarly, the same process may be utilized to conduct research in which personal records are used to protect data confidentiality under the Health Insurance Portability and Accountability Act (HIPAA) requirements. Patients' records on the Blockchain can be automatically forwarded to insurance providers, or the doctor can securely communicate the patient's health records to the appropriate parties **[40]**.

2.8.7 Voting

In the next years, Blockchain might provide numerous real benefits for a verified and secure voting system. The current voting method has problems, making votes difficult to verify. As a result, Blockchain's characteristics may be able to create an immutable, verifiable, and secure voting system in which voters may cast their ballots with confidence from anywhere on the planet **[40]**.

2.8.8 Smart Contracts

Ethereum Virtual Machine (EVM) bytecode is a Turing-complete byte language used to write smart contracts. They are effectively a collection of functions, each one is a set of instructions. Conditional statements are included in these contracts, allowing them to self-execute. Smart contracts can be used in place of intermediaries to ensure that all parties are bound by the terms of the agreement. As a result of Blockchain, such regulating agencies are rendered obsolete. Smart contracts based on Blockchain guarantee that all participants are aware of the contract terms and that the agreement is automatically implemented after all requirements have been met. In order for smart contracts to function, a set of mutually "untrusted" peers known as miners must validate the contract's transactions. Miners gather and verify each transaction broadcast to the Blockchain network before encoding it into a new block and appending it to the Blockchain **[40]**.

2.9 Big Data & Blockchain technology

Integrating Big Data with Blockchain technology is a certain approach to address and manage some, if not all, of Big Data's core issues and challenges. With the rise of bitcoin and other cryptocurrencies, Blockchain technology has gained widespread attention. Every Bitcoin transaction must traverse through the Blockchain in order to be validated by the millions of other peer-to-peer users. This means that every transaction is public, and the activities of every Bitcoin user are visible to the Blockchain's millions of other users. The system enables extremely secure digital transactions and data storage. Even though Blockchain was originally used in the context of cryptocurrency, the concept may be used for a wide range of activities, including agricultural transactions **[41].**

In the first epoch of Big Data, power resided with those who owned the data. In the Blockchain epoch of Big Data, power will reside with those who can access the most data (where public Blockchain will ultimately defeat private Blockchain) and those who can gain the most insights most rapidly. This has two major consequences, which are as follows **[42]**:

- Organizations will no longer own customer data, which will be stored in corporate databases. It will be represented as tokens or currencies on an identity Blockchain and will belong to each individual. As needed, the future customer will grant access to others.

- Anyone will be able to see transaction details. The data regarding the transactions that take place on a Blockchain may be accessed by anybody.

Blockchain appear to have significant ramifications for large data security and privacy. On the one hand, Blockchain-based solutions, when paired with other cryptographic primitives, may enable more secure financial transactions, data exchange, and provenance storage. The data saved on Blockchain, on the other hand, may be analyzed to give new insights into developing data security issues.

Cryptocurrencies appear to be used in payments for human trafficking, ransomware, personal blackmail, and money laundering, among other things. Law enforcement authorities can employ Blockchain Data Analytics technologies and large data analysis algorithms to detect such misuse.

2.10 Blockchain Big Data applications

Maintaining data ownership, data transparency, and access control management has always been a major challenge. Blockchain technology solves this problem by keeping access controls to personal data in the Blockchain architecture. A decentralized personal data management system is built by creating a protocol that allows people to own and control their data using Blockchain technology. Organizations may focus more on data use rather than security management and compartmentalization because their dependency on third parties is fully removed **[43].**

Data management and data analytics are two areas where Blockchain is being used in combination with Big Data. In the case of data management, Blockchain technologies are used to store important data since they are secure and distributed. It may also verify data validity and protect sensitive data from being tampered with. In the applications of data analytics, Blockchain is used to analyze trading trends, predict potential customers, diseases, or business partners **[44].**

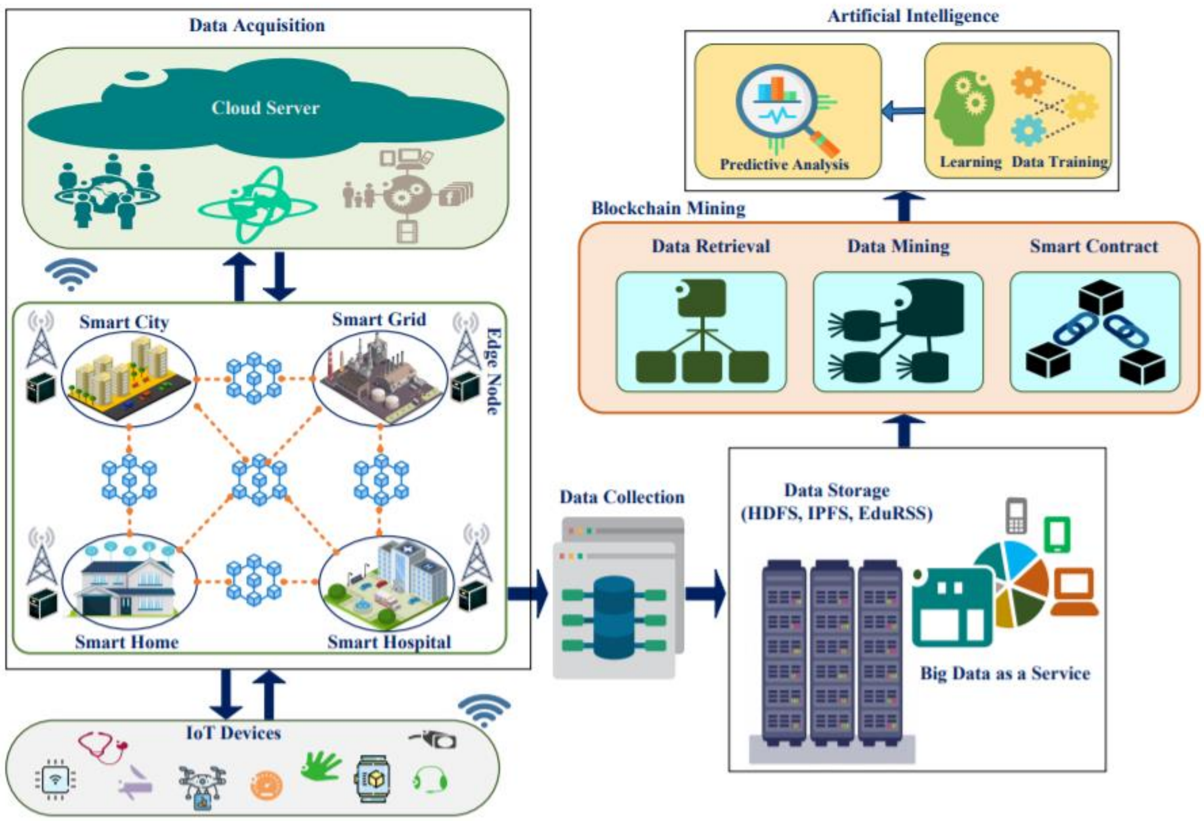


Figure 2. 6: An overview of Blockchain services in Big Data environment **[45]**

2.10.1 Blockchain Big Data in Smart City

A massive amount of data will be generated by the crowd and IoT devices on a daily basis. Rahman et al. they propose a Blockchain-based infrastructure for a long-term sharing economy in megacities that supports IoT and secures spatio-temporal smart contract services. Sharing economy services where two parties can securely execute any number of transactions without requiring any trusted third party can take advantage of this recommended structure by utilizing intelligence at the edge to coordinate seamlessly with the IoT data processing framework **[46].**

2.10.2 Blockchain Big Data in Smart Healthcare

Healthcare data is a vital source of information in the field of medicine. Sharing healthcare data is an important step in making the healthcare system smarter and improving service quality. Instead of being distributed among numerous healthcare systems, which prevents data sharing and puts patient privacy at risk, healthcare data, which is a personal asset of the patient, should be owned and controlled by the patient.

Yue et al. They presented a Blockchain-based app (Healthcare Data Gateway (HGD)) architecture that offers a new way to boost the intelligence of healthcare systems while maintaining patient privacy. The suggested purpose-centric access architecture assures that patients own and control their healthcare data; the basic unified Indicator Centric Schema (ICS) allows for the practical and easy organization of all types of personal healthcare data.

They also mention that MPC (Secure Multi-Party Computing) is one promising option for allowing untrusted third-parties to perform computations over patient data while maintaining privacy **[47].**

2.10.3 Blockchain Big Data in Smart Transportation

Cocîrlea et al. stores trustworthy traffic data that drivers and other intelligent transportation systems (ITS) can use in a variety of situations using Blockchain technology. Because of the structure of the Blockchain, tampering with the data is difficult, ensuring its integrity. In this study, a reputation system was defined with the purpose of assisting in the creation of a consensus within a decentralized network of untrustworthy nodes, which were represented by the drivers.

Region nodes take data from users in their region, run the consensus algorithm on that data, and generate blocks in the region's Blockchain, while master nodes collect reliable data from region nodes and create new blocks in the master Blockchain **[48].**

The following table presents the problem in each of the aforementioned domains, as well as the proposed solution utilizing Blockchain technology in each work:

Table 2. 4: Summary of Blockchain big data applications

Reference	Application	Problem	Blockchain Solution
[46] Rahman et al.	**Smart City**	The issue in smart cities is how to securely exchange the massive amounts of data produced by IoT devices.	Blockchain-based smart city sharing economy solutions allow two parties to do transactions in a completely decentralized way, without the need for a third-party trusted intermediary.
[47] Yue et al.	**Smart Healthcare**	How to obtain, store, and analyze personal healthcare data without causing privacy violations is a major difficulty for healthcare data systems as they strive to grow smarter.	Blockchain is a public, secure, and verifiable ledger that may be thought of as a storage supply chain in which every action is validated, responsible, and immutable. Because of these individual characteristics, it might be a viable option for healthcare data systems that are concerned about both patient privacy and data exchange. The suggested architecture is independent of any other parties, and no one entity has complete control over the processing.
[48] Cocîrlea et al.	**Smart Transportation**	A proactive safety system might provide information on typical speeds on a certain stretch of road, if there is a traffic delay, and alerts for potential risks such as potholes, roadkill, storms, or hazardous curves that users may encounter. Other users advertise the system or guarantee the information supplied by other users, therefore the data comes from them.	After collecting user data, which is subjected to a consensus method, both drivers and Intelligent Transportation Systems may utilize the data to build a safer, more streamlined, and healthier environment.

2.11 Conclusion

Over the last few years, Big Data has generated a lot of interest in a variety of scientific and technical fields. Despite its numerous benefits and applications, Big Data presents a number of issues that must be addressed in order to improve service quality. Due to its decentralization and security, Blockchain has the potential to improve Big Data services and applications. The first layer of security can be attributed to the network architecture of the Blockchain. Unlike with traditional methods, the information in the network is secure and cannot be duplicated. Secondly, using Blockchain to store Big Data can make the information structured and valuable. Performing analysis on data becomes much more efficient and easier. Using Blockchain technology can also help with fraud detection. Banks and financial institutions can trace a transaction to its origins and detect any anomalies. In the next chapter we will introduce the design of the new approach using Blockchain for Big Data in banking.

Chapter 3

A Blockchain-based approach for Big Data security

3.1 Introduction

Different sectors have made extensive use of electronic platforms as a result of the exponential rise of digital technology. Customers can make use of a variety of online services provided by banks. Fund transfers, bill payments, loan application processing, and distribution through the internet are among the several transactional operations. A large number of customers have chosen the following services since they are simple to use and handle. Banks have an important role in ensuring that all digital financial transactions are properly protected against potential threats. Customers can effectively carry out their banking activities since they can closely monitor their account activity. As a result, Blockchain may be one of the greatest methods for ensuring high security while decreasing fraud and administrative costs.

In this chapter, we will describe our ***"Aman"*** system, which guarantees that banking transactions are transferred in a safe and secure way and that the storage phase is secured by using Blockchain technology. We will discuss why we chose Blockchain as a solution for bank security issues, and then we will describe the global architecture of our system, as well as how our system works and how we use the Blockchain.

3.2 Benefits of Blockchain in Banking

Below, we'll go over the advantages of adopting Blockchain:

- **Privacy:** it enables users to own their data and prevents third-party middlemen from misusing and obtaining data.
- **Transparency:** because the transactions are distributed across the network, transparency is enhanced, making fraud detection and prevention easier and because banks utilize a shared digital ledger to record each transaction, they are more visible to Blockchain members.
- **Immutability:** transaction history on the Blockchain cannot be changed since the Blockchain ledger is permanent and unmodified.
- **Visibility and traceability:** customers may verify their chain and trace any changes in data or fraud.

- **Faster Transactions:** when compared to traditional financial institutions, Blockchain transactions are speedier. This allows a user to transfer money more quickly, saving time in the long term.

3.3 The Global System Architecture

As mentioned in the figure **(Figure 3.1)**, we divided our proposed architecture into four layers that interact with each other and exchange data in real time. As a result, each layer performs a specific role within the application. Through the presentation layer up to the storage layer, it also shows how the data passes until be stored.

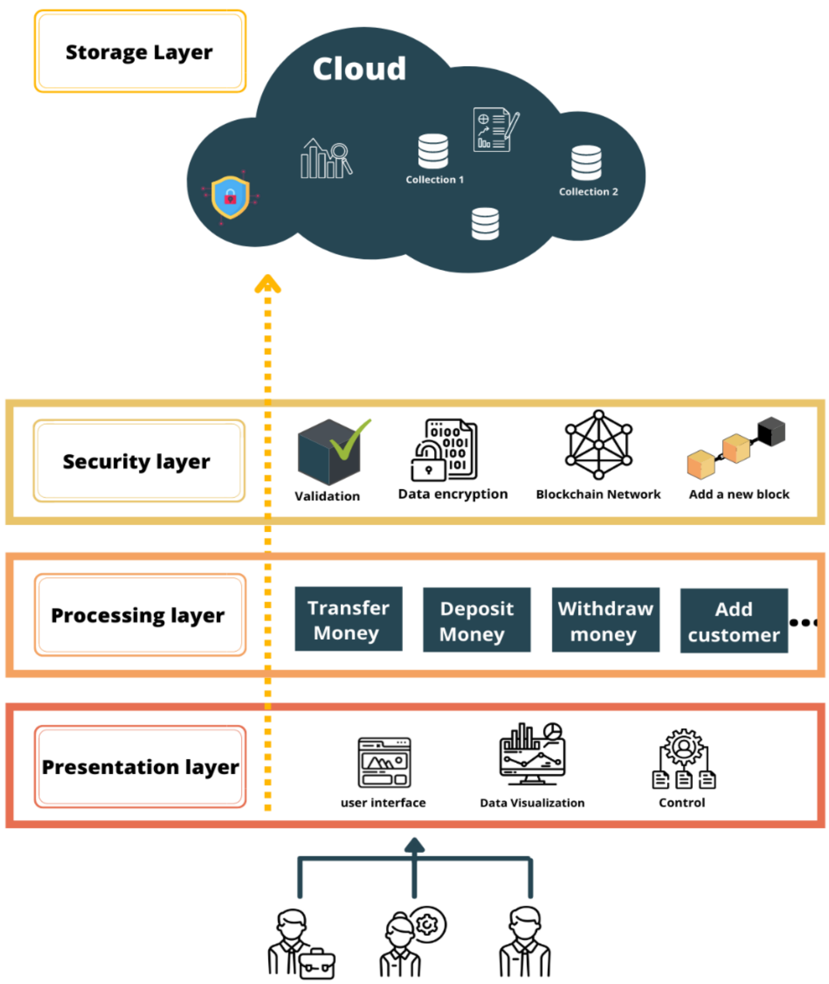


Figure 3. 1: The Global System Architecture

The next subsection will go over the components of the suggested architecture.

3.3.1 The Actors

The actors are the users who interact with the system. An actor is a person, a group, or an external system that interacts with the application. They must be external data producers or consumers. In our case, they may be agency employees, clients as users, and as controllers the administrators.

3.3.2 Presentation Layer

Presentation layer is the user interface which is the point at which the actors interact with the application. Its goal is to make the user's experience easy, requiring minimal effort, so that the user can simply monitor and handle his data while also facilitating his job.

3.3.3 Processing Layer

The Processing layer is an application layer that is in charge of coordinating components, service calls, and all of the technology necessary to process the incoming request and produce and provide the response to the client. It refers to the sections or code of a program that allow it to run but cannot be viewed by the user.

3.3.4 Security Layer

All financial information, including deposits, withdrawals, and transfers, user data must be protected. Sharing this information with relevant clients and bank staff for each agency or central bank increases operations' credibility and transparency while also providing clients with a safe space to keep their data. As a consequence, we employ the Blockchain technology to protect this data, as well as the data encryption technique for some information. We will go through all of the techniques we used to secure financial data with a high level of security.

3.3.4.1 Blockchain Network

Each client has its own Blockchain, which can only be accessed by authorized users. As shown in the figure **(Figure 3.2)**, the client's Blockchain is shared with staff from the same agency to which he belongs and the central bank.

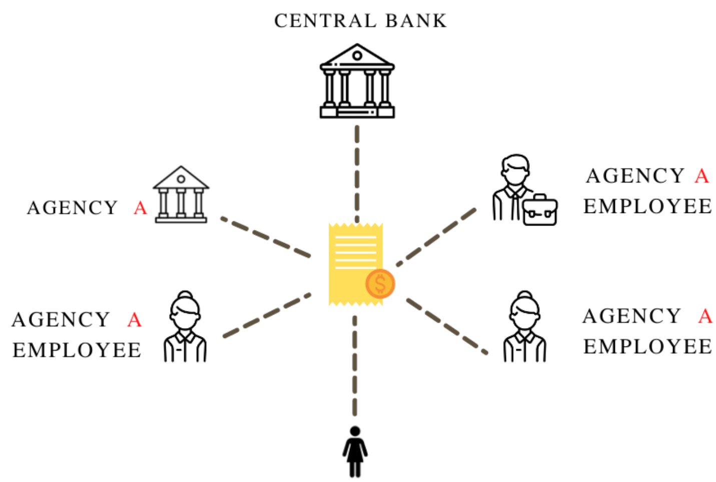


Figure 3. 2: Blockchain Network

3.3.4.2 Validation

Each transaction must be confirmed before it can be completed or a new block added to the chain; there are two possibilities here. The first is the case of money transfer, which may be done by agency staff or the client himself, so we must first check the identification of the sender and its balance to determine if it is sufficient or not, then verify the identity of the recipient and finally check the identity and password of the executor.

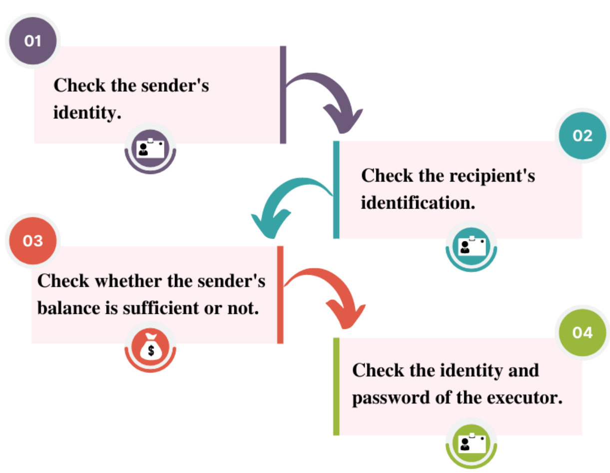


Figure 3. 3: Verification process in case of money transfer

In the case of withdrawing and depositing funds, in addition to validating the executor's identification and password, we will verify the client's identity, whether he previously had an account, and if he is a new client, a new account will be created. Finally, in the case of a withdrawal, we will verify the balance to determine if it is sufficient; in the case of a deposit, we do not need to check the balance.

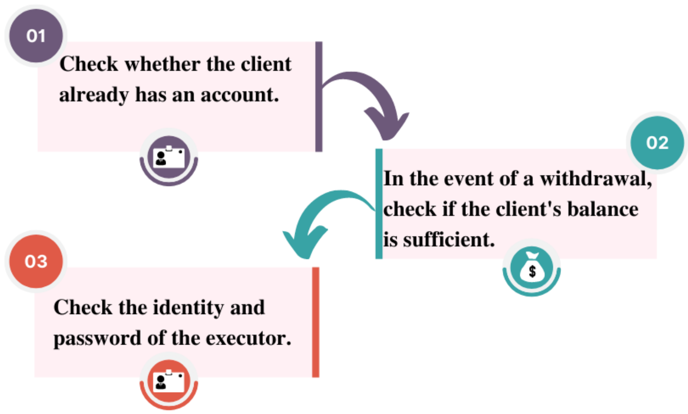


Figure 3. 4: Verification process in case of withdrawal and deposit of money

3.3.4.3 Data Encryption

When users' data is stored in the database, there are some sensitive data that must be stored in a secure manner and only the user can know about this information because the database may be compromised and its data may be obtained by someone else. One such piece of data is the "password", if the passwords are stored in what we call plain text it will be easy for a hacker to get the user's passwords. For this reason, we opted to hash passwords in our application using a one-way hashing mechanism rather than plain text. "Hashing" a password refers to taking a plain text password and putting it through a hash algorithm. They take any string and convert it into a fixed-length string that cannot be reversed.

The hacker can determine out the original password if we only hash the password, we add a ***salt*** to the hashed password, which is a random text. Salts generate unique passwords even if two people choose the same password. The salt is automatically added to the hash.

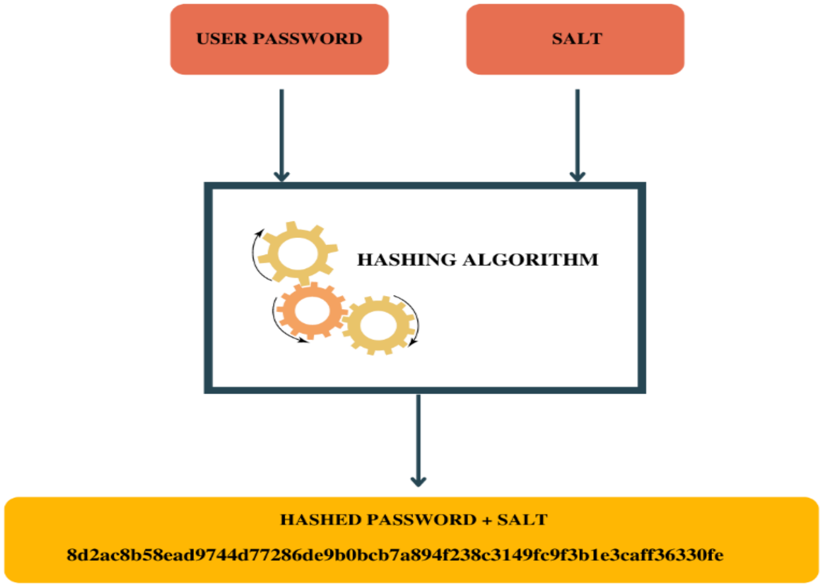


Figure 3. 5: Password hashing process

3.3.4.4 The process of adding a new block

As previously stated, we adopted a Blockchain architecture for deposit storage, withdrawal, and transfer. We separated the storage spaces into two sections: one for keeping withdrew and deposited money data and the other for money transfers.

- **Case of fund deposit and withdrawal**

When a new request is received, the system verifies its validity first. After that, it checks to see if there is a previous block or if this is the first time. If the previous block does not exist, the system obtains the genesis block. The hash of the previous block will then be appended to the new block as the previous hash, and the hash of the block will be generated using data, the previous hash, timestamp, and nonce.

After generating the hash, the new block will be added to the client chain and this information is shared with the client as well as all employees of the agency to which he belongs.

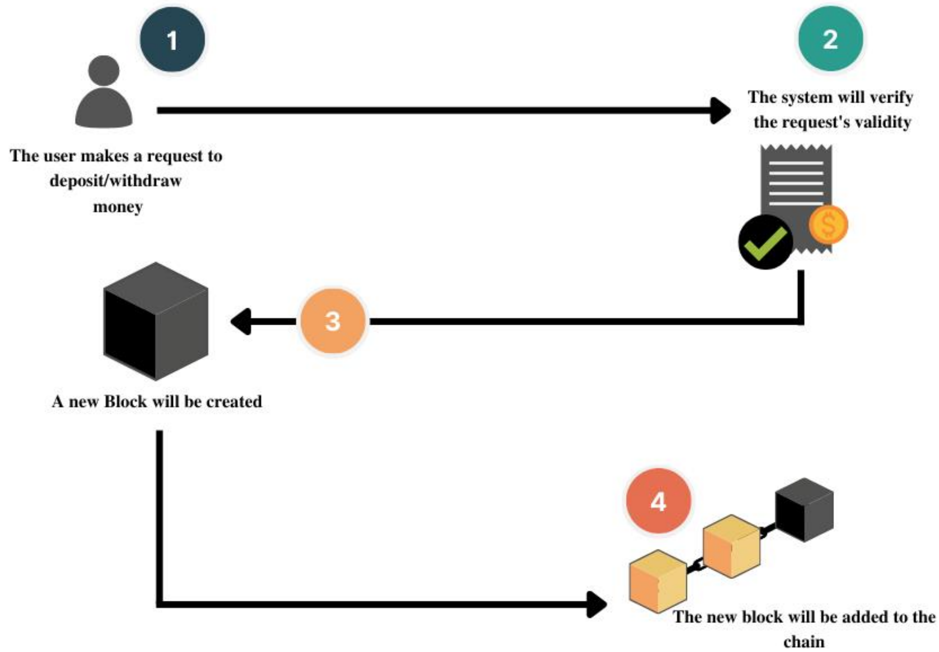


Figure 3. 6: The process of creating a new block in case of withdrawal and deposit of funds

- **Case of money transfer and reception**

In this situation, two blocks will be generated at the same time, one for the sender and one for the recipient. The two additional blocks will be created in the same way as money is deposited and withdrawn.

Here, we will generate two new blocks by obtaining the last block for each, that is, obtaining the chains of both (sender and receiver) and then adding the two blocks to each of them. The following example will aid our comprehension **(Figure 3.8)**.

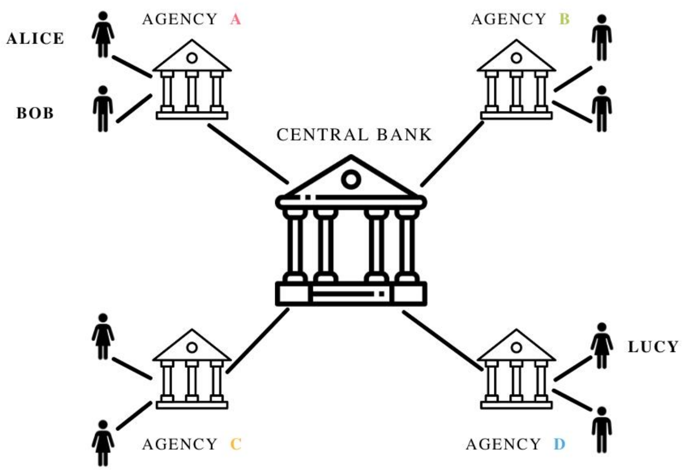


Figure 3. 7: The national bank's general structure

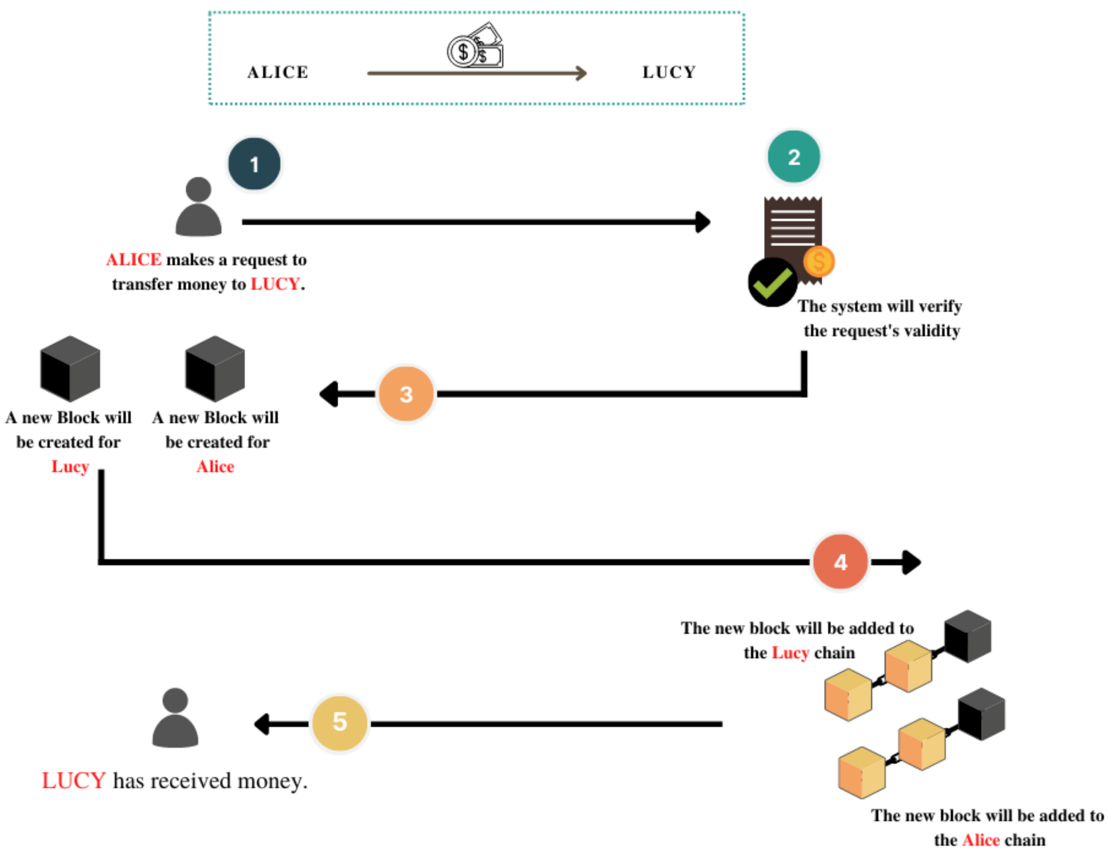


Figure 3. 8: Money transfer example

3.3.5 Blockchain based cloud storage Layer

A database is any data structure that is used to store information. As a result, we may refer to Blockchain as a database because it is a ledger for securely recording transaction data. When constructing a Blockchain, each block must be stored in a centralized location so that the Blockchain can be searched and new blocks can be added to the chain. After extensive research, we determined that the MongoDB Atlas, a database-as-a-service cloud, is the best solution for storing a Blockchain ledger. We now wonder what this has to do with big data and how this method may solve big data issues.

As stated in the first chapter the Big Data life cycle is divided into five stages, our solution will handle the security issues of ***'the storage phase'***, which is also the most essential phase. The suggested solution will be the largest place to store Big Data while also being the strongest solution for safeguarding big data through the use of Blockchain and cryptography **(Figure 3.9)**.

Once the data has been protected, the analytics phase will be able to give decision-makers with true and fresh information as well as valuable insights.

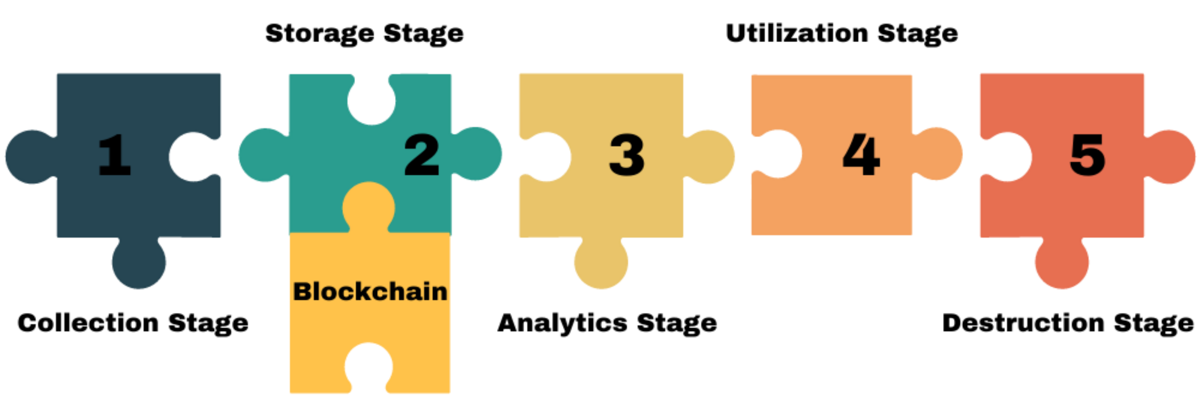


Figure 3. 9: Big Data lifecycle combined with Blockchain

Blockchain-based cloud storage will also provide the key characteristics of Big Data:

- **Volume:** Mongodb Atlas will provide sufficient storage capacity for massive amounts of data.
- **Value:** this word illustrates the value of data since it gives essential information and knowledge for businesses, thus when we ensure that the data saved is correct and secure, we may have trust in the results obtained after analyzing this data.
- **Velocity:** queries can be substantially quicker than in a relational database since related data is kept together in documents.
- **Veracity:** securing data with Blockchain will assure data quality and accuracy, as data quality and privacy are critical for accurate analysis.

3.4 System operations

This section will be divided into two parts, the first section discusses the system's functional flowchart, while the second section discusses the sequence diagrams that specify the interactions.

3.4.1 Blockchain functional flowchart

First, we must determine whether or not the first block, ***"Genesis Block,"*** has been generated. The genesis block is the initial block with no contents and a previous hash of 0. When a new request comes in, whether for a transfer, deposit, or withdrawal, we must validate it before adding a new block. When depositing funds, we must first verify whether or not the customer exists, then check who did the transaction, whether or not he exists, then verify his password.

In the case of withdrawing funds, we must also verify if the customer's balance is sufficient. Similarly, in addition to the previous, in order to transfer funds, we must verify the receiver and sender.

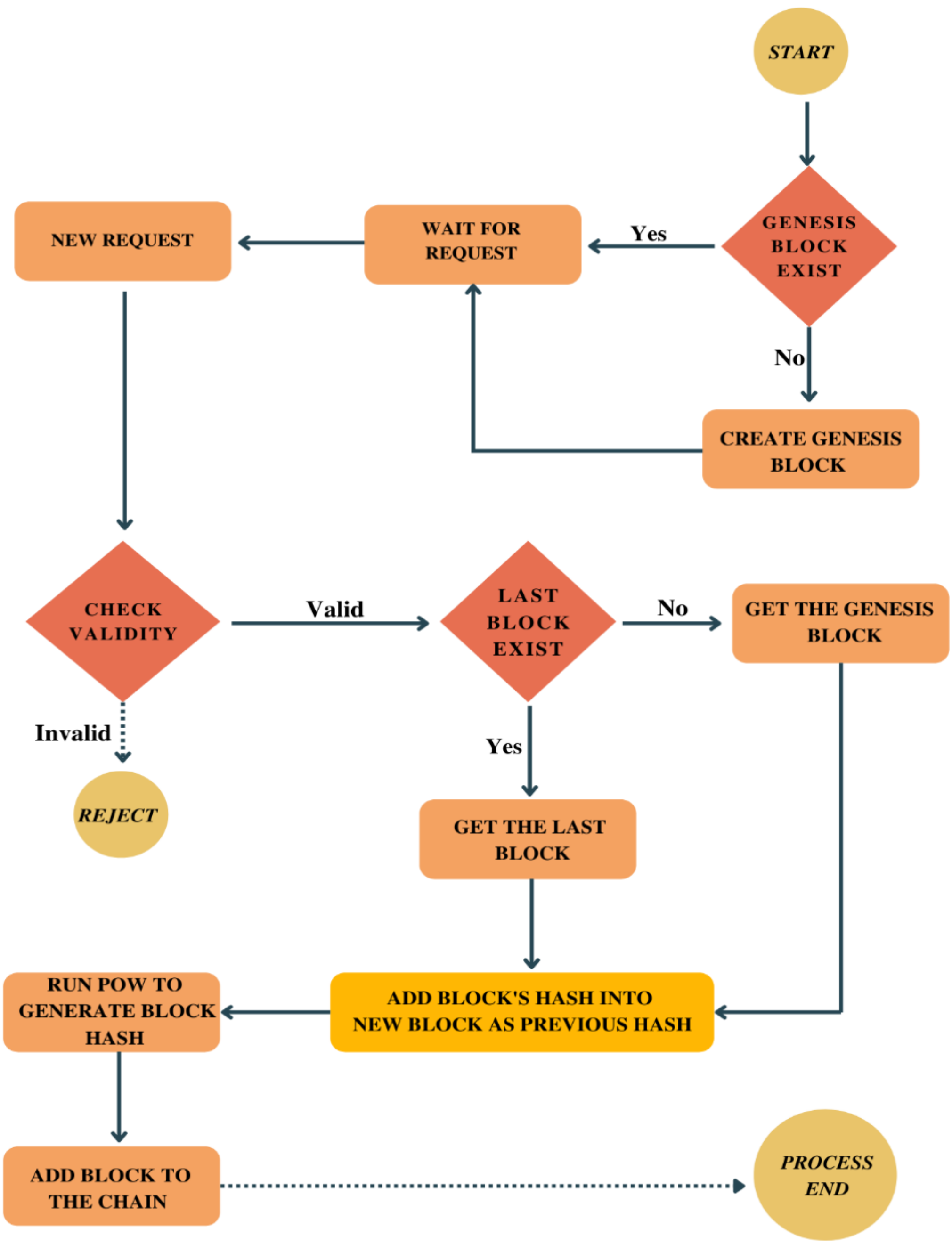


Figure 3. 10: Blockchain functional flowchart

Each client in the network has his own private chain; to add a new block, we utilize the most recent block in his chain; if it does not exist, we obtain the Genesis block; in both cases, we append the block's hash into the new block as the previous hash. Then, utilizing POW, we generate a new block hash based on data, nonce, previous hash, and timestamp. Finally, a new block will be successfully added to the chain.

3.4.2 Sequence diagrams

We provide a detailed description of our system by using sequence diagrams to highlight the several interactions between each of the actors and the system's components.

3.4.2.1 Sequence Diagram "Employee"

The employee is in responsible of creating client accounts, as well as having access to and seeing all customer data and information. He can edit data or delete a customer, as well as check his balance. He has access to any customer's deposit and withdrawal history, as well as transfer and receive list.

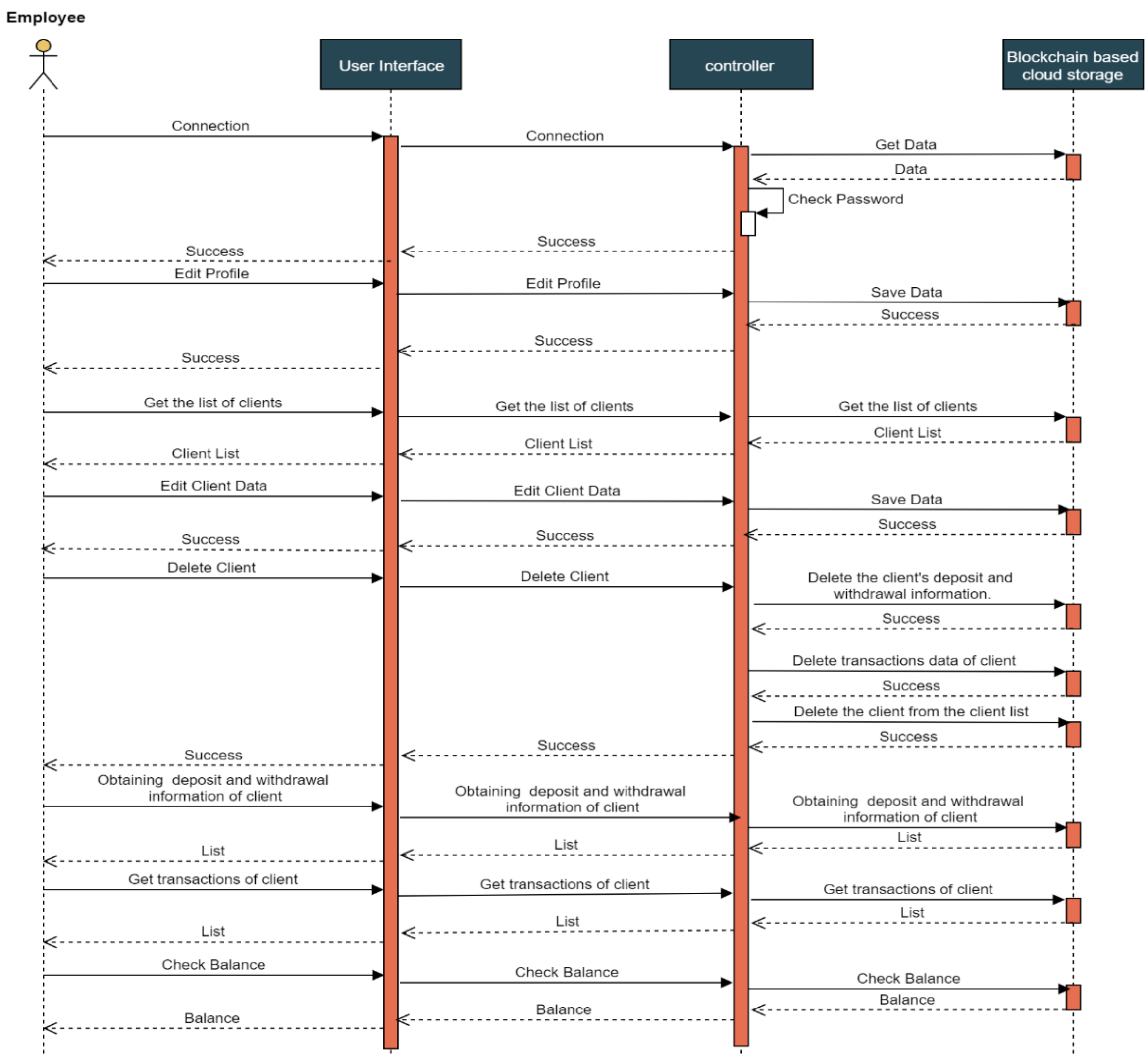


Figure 3. 11: Sequence Diagram **"Employee"**

3.4.2.2 Sequence Diagram "Administrator"

The administrator has the same powers as the employee, but he has the capacity to modify his profile, add new administrators, and add or update employee data.

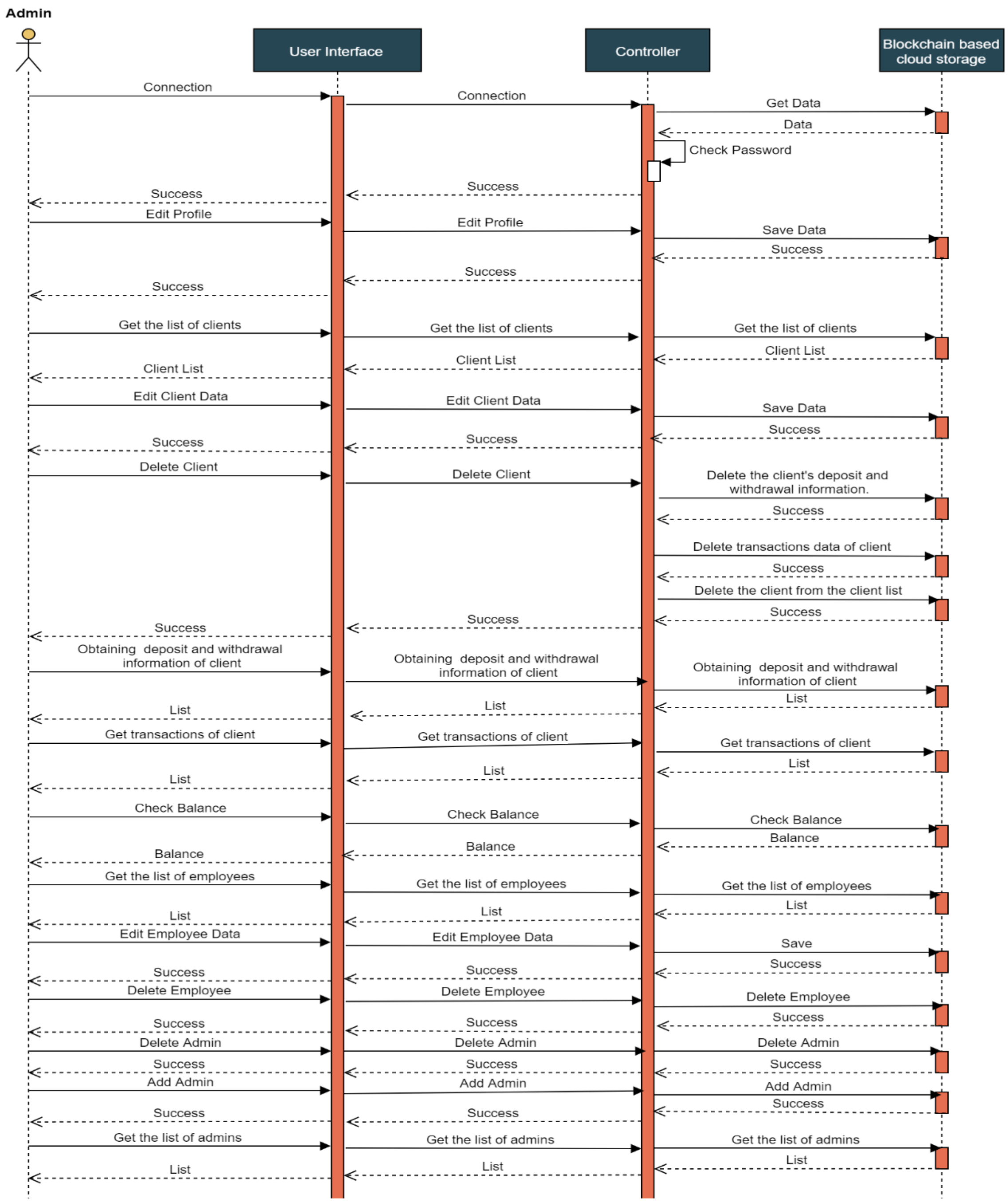


Figure 3. 12: Sequence Diagram **"Administrator"**

3.4.2.3 Sequence Diagram "Customer registration"

The customer cannot register on their own; this must be done by staff members. once account is created, password will be encrypted and no one can know it, only customer as shown in **(Figure 3.13)**.

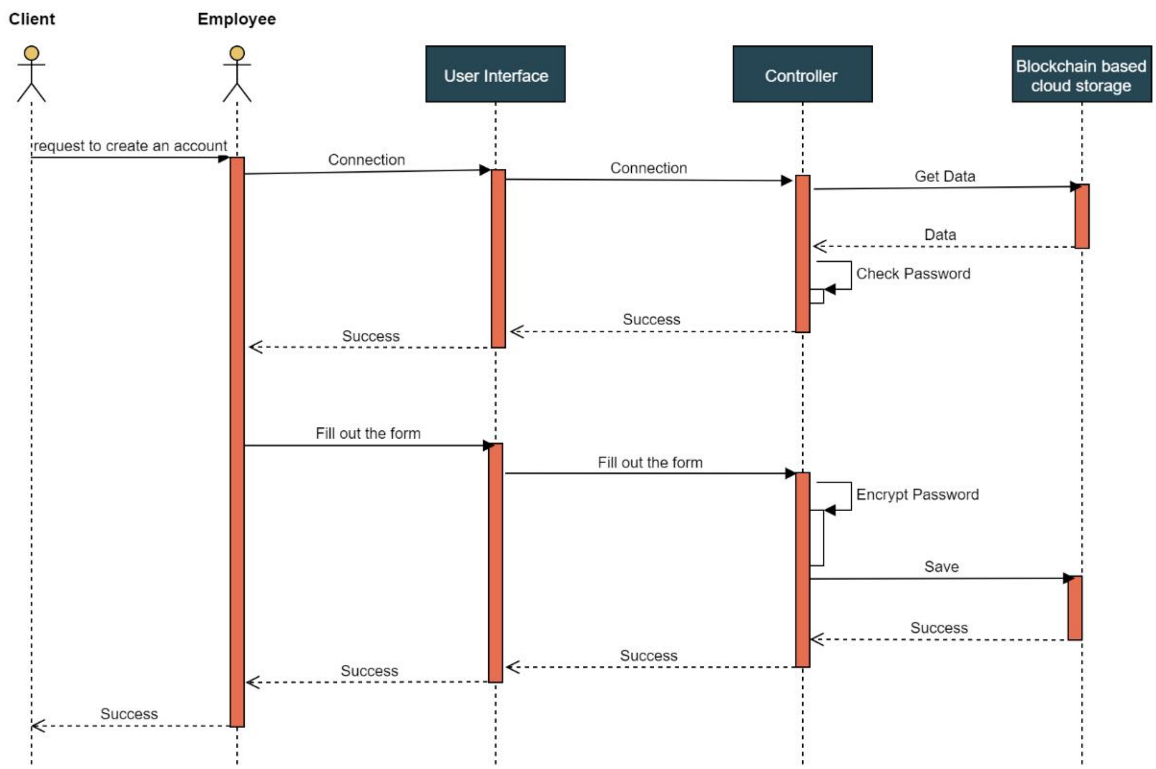


Figure 3. 13: Sequence Diagram **"Customer registration"**

3.4.2.4 Sequence Diagram "Deposit and withdraw funds" and "Transfer funds"

Deposit and withdraw funds, these two actions can only be performed by agency staff, once the request is made, the system will first check if the request is valid, if it is invalid, an error will be sent as a response otherwise a new block will be created and added to the chain of customer successfully, figure **3.14** illustrates this. Another thing the agency personnel may do is transfer money, as seen in figure **3.15**.

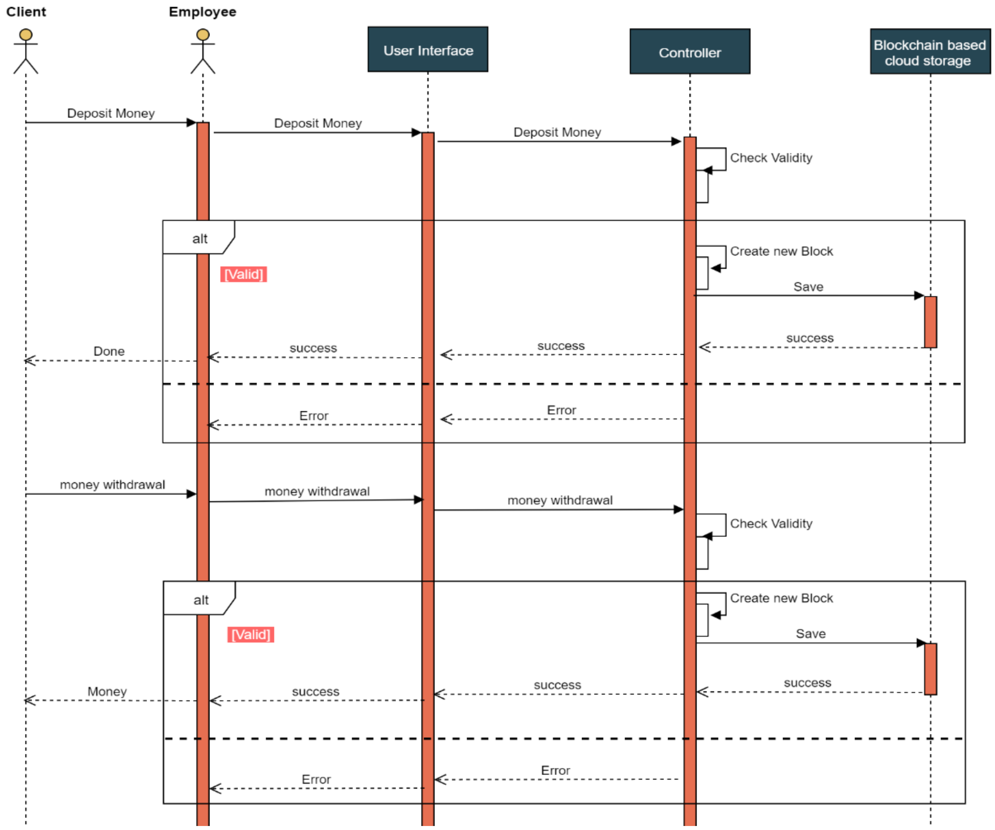


Figure 3. 14: Sequence Diagram **"Deposit and withdraw funds"**

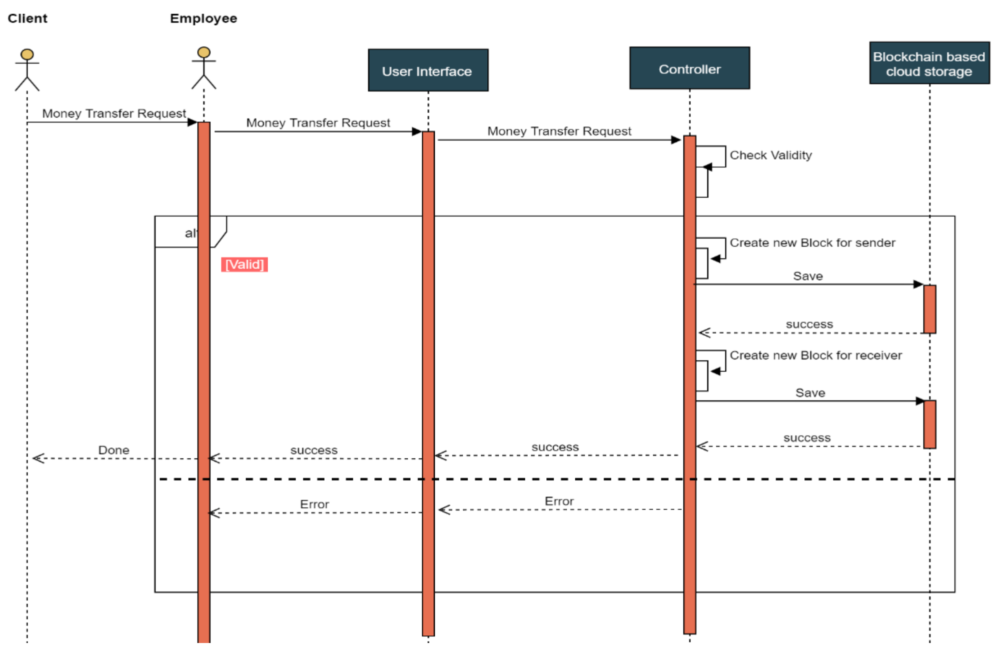


Figure 3. 15: Sequence Diagram **"Transfer funds"**

3.4.2.5 Sequence Diagram "Customer"

The customer may view his balance, transaction history, and the authenticity of the Blockchain, as well as transfer funds immediately from his account.

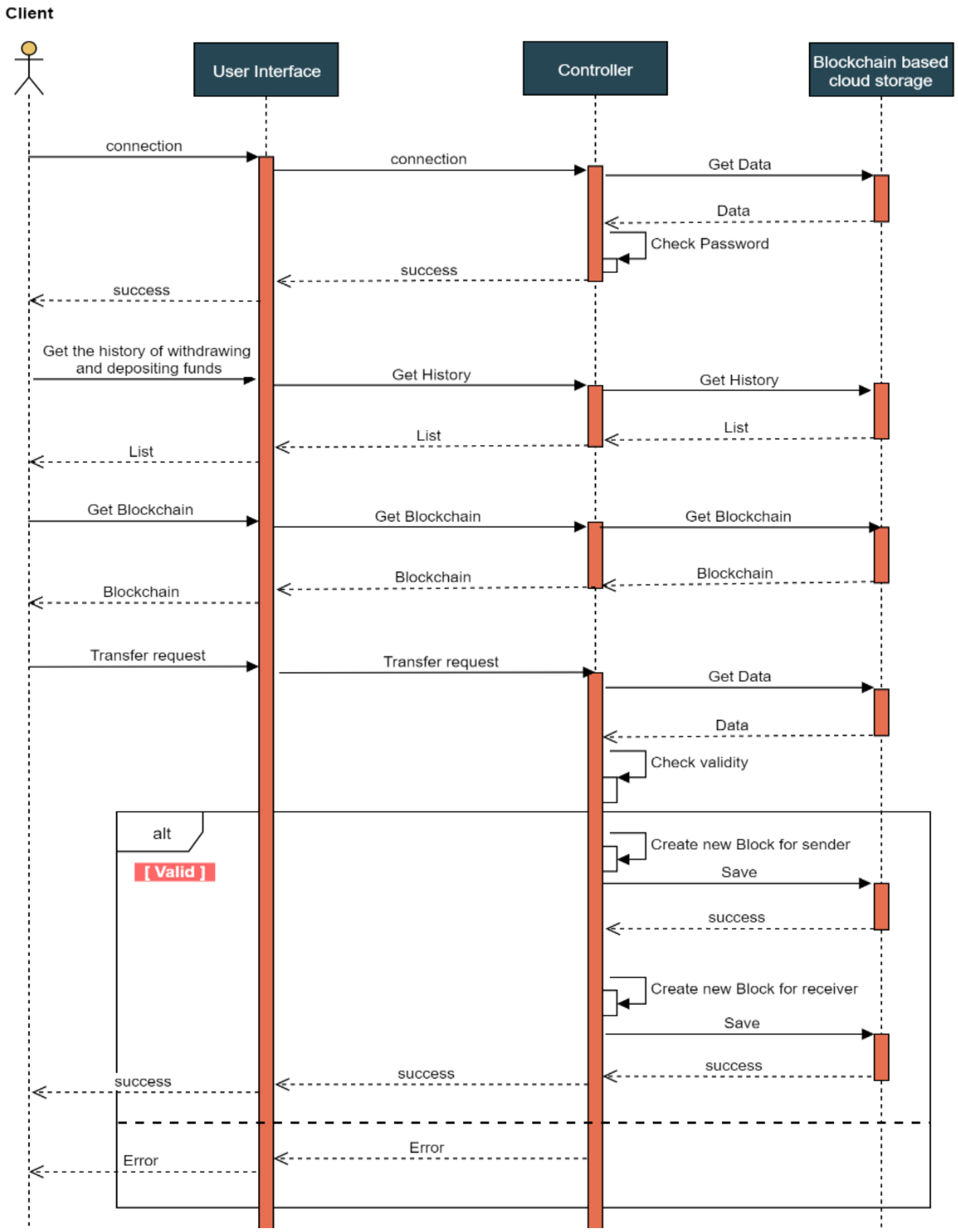


Figure 3. 16: Sequence Diagram **"Customer"**

3.5 Conclusion

In this chapter, we describe how we built our system using Blockchain technology, which allows us to securely store banking data. We've covered the benefits of using a Blockchain, then the system functionality, and finally how the process of creating a new block works. The implementation of our system will be presented in the next chapter.

Chapter 4
Implementation

4.1 Introduction

In the previous chapter, we presented a full explanation of our system's general architecture as well as covered the different design processes where we defined each layer in depth and discussed how to apply the Blockchain approach in and which layer exactly.

We begin this chapter by discussing the platforms and tools that were employed in the development of our system. Following that, we will provide an overview of our ***"Aman"*** application and define the essential components, including a detailed explanation of each.

4.2 Development tools and used platforms

In this section, we will cover the tools and platforms that we utilized in our system, beginning with hardware tools and then moving on to software tools.

4.2.1 Hardware tools

Our system was built on a graphics station running Windows 64-bit, an Intel(R) Core(TM) i7-7500U CPU @ 2.70 GHz, 2.90 GHz, and 8 GB of RAM.

4.2.2 Software tools

We divide this section into front-end and back-end tools as well as libraries and platforms:

4.2.2.1 Platforms

- **Visual Studio**

The platform used for coding is "visual studio", it is a desktop-based source code editor that is lightweight but powerful. It has built-in support for JavaScript, TypeScript, and Node.js, as well as a robust ecosystem of extensions for additional languages including C++, Java, Python, and PHP.

- **MongoDB Atlas**

We use MongoDB Atlas to store enormous volumes of data. It is a cloud-based, open-source NoSQL database that employs JSON documents with dynamic schemas as an alternative to table databases. Atlas delivers all of MongoDB's functionality while automating database administration activities like database configuration, infrastructure provisioning, patches, scaling events, backups, and more, allowing developers to focus on what matters most to them.

4.2.2.2 Front end tools

- **HTML**

HTML, or HyperText Markup Language, allows online users to employ elements, tags, and attributes to build and organize sections, paragraphs, and connections.

- **CSS**

CSS is a style sheet language used to describe the presentation of a document written in a markup language such as HTML.

- **JS**

JavaScript is a dynamic programming language used for web development, web applications, game creation, and many other things. It enables you to add dynamic features on web pages that would be impossible to achieve with simply HTML and CSS.

- **Bootstrap**

For the design, we utilize Bootstrap, a free and open-source web development framework. It's designed to make responsive, mobile-first website building easier by offering a set of syntax for template designs.

4.2.2.3 Back end tools

- **Node.js**

Node.js is a back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside of a web browser. JavaScript is a dynamic programming language that is used for web development, web applications, game development, and other purposes.

- **Express**

Express is a NodeJS web application framework that includes many functionalities in addition to the basic Node modules to speed up and simplify web development.

- **Npm**

Npm is a JavaScript package management developed by npm, Inc. The Node.js JavaScript runtime environment's default package manager is npm.

- **Nodemon**

Nodemon is a program that assists with the development of Node.js-based applications by restarting the node application when file changes in the directory are detected.

4.2.2.4 Libraries

- **EJS**

EJS or Embedded Javascript Templating is a templating engine used by Node.js. Template engine helps to create an HTML template with minimal code. Also, it can inject data into HTML template at the client side and produce the final HTML.

- **bcrypt.js**

The bcrypt NPM package is a JavaScript implementation of the bcrypt password hashing algorithm, which makes it simple to generate a hash from a password text. Hashing is a one-way function that cannot be reversed once completed, unlike encryption, which may be decoded to recover the original password.

- **CryptoJS**

CryptoJS is a growing collection of standard and secure cryptographic algorithms implemented in JavaScript using best practices and patterns. They are fast, and they have a consistent and simple interface.

- **Mongoose**

Mongoose is a MongoDB Object Data Modeling library based on Node.js. At the application layer, we utilize Mongoose to enforce a specific schema. In addition to enforcing a schema, Mongoose has a number of hooks, model validation, and other features that make working with MongoDB easier.

- **Morgan**

Morgan is a Node.js and Express middleware which is a function in Node.js and Express that provides access to the request and response lifecycle methods, which allows you to continue processing in your Express server.

- **Lodash**

Instead of creating basic functions again and again, we utilize Lodash, a JavaScript library, to accomplish the task with a single line of code.

4.3 System Implementation

In this section of the chapter, we provide a full overview of our system, accompanied by screenshots of its various web pages.

4.3.1 Description of the system

Our system's purpose is to manage all of the most frequent bank actions in order to make employees' tasks easier and to provide clients with access to their data at any time and from any location. The system allows the administrator to access all client data and offer an overview of the bank, from staff to clients. It also enables each agency's personnel to carry out all banking operations. It allows each client to see their entire deposit and withdrawal history. It also enables him to transfer money as securely and quickly as possible. Furthermore, it enables him to check the chain of transfers, allowing him to assess whether the chain is flawed. Figure **4.1** shows the system's logo.

Figure 4. 1: The system's logo

As previously stated, the primary goal of our system is to secure data; the strategies employed are detailed below:

- **How to store user data?**

We aim to keep user data private in our system; not all data needs to be secured; for this we chose to secure simply the password, as shown in the Figure **4.2** below, which shows how the data is saved and how the password is stored so that even administrators and employees cannot guess the password or, in other words, it is impossible to know the password.


```
fullname: "Nadjia Boutarfaia"
dateofbirth: "08/09/1999"
gender: "Female"
dateofhiring: "08/09/2022"
AgencyID: "07"
city: "07"
mobile: "06-------15"
email: "example@gmail.com"
password: "$2a$10$EVr/SBnKiavokQ2BT4vRc.WTDy8L20C3CcTi4s9TqJ.BdG/yGCka2"
createdAt: 2022-04-18T08:30:45.061+00:00
updatedAt: 2022-05-29T16:26:26.501+00:00
```


Figure 4. 2: Registered user data

- **Transaction data storage**

Since transfer and deposit (or withdrawal) data are different, we need two different schemes, so we decided to create a space to store transfer data and another for deposit and withdraw money. Each client has two transaction chains: one for transfers and one for deposits and withdrawals. For both situations, the system begins by creating a genesis block.

Figure **4.3** illustrates the genesis block for withdrawal and deposit funds, while Figure **4.4** represents the genesis block for transfer money.

```
index: "0"
Hash: "003a2e7292211f3dbabd0af3c810e3a0a866e7c1d051967ce48924440571b498"
PreviousHash: "0000"
Data: Array
  0: Object
    idemployee: "null"
    idclient: "null"
    oldbalance: "null"
    newbalance: "null"
    Amount: "null"
    ttype: "null"
    _id: ObjectId("62994e8877199aeae301f6eb")
Nonce: 0
createdAt: 2022-06-02T23:58:00.775+00:00
updatedAt: 2022-06-02T23:58:00.775+00:00
```


Figure 4. 3: In the case of withdrawal and deposit money, the genesis block


```
_id: ObjectId("62994e5b77199aeae301f6e6")
index: "0"
Hash: "00d3c894956f9107764a324c42a8727b1d7ba21d7e731429ef828eb6956f1fea"
PreviousHash: "0000"
ChaineID: "null"
Data: Array
  0: Object
    Whodid: "null"
    From: "null"
    To: "null"
    Amount: "null"
    _id: ObjectId("62994e5b77199aeae301f6e7")
Nonce: 0
createdAt: 2022-06-02T23:57:15.513+00:00
updatedAt: 2022-06-02T23:57:15.513+00:00
```


Figure 4. 4: In the case of transfer money, the genesis block

The block is defined as a unit containing the information of each transaction. The block components in the case of depositing and withdrawing funds are indicated below:

1. **Index**: indicates the block number.
2. **Hash**: the hash of the current block, which is the hash of all the information in this block.
3. **PreviousHash**: It is the hash of the previous block, which is "0" in the case of the genesis block.
4. **Data**: the data is an array containing transaction information such as the ID of the employee who performed the action, the ID of the client, the old balance of the client, the amount, the

new balance after adding the amount to the old balance, and finally the type, which determines whether the transaction is a withdrawal or a deposit of funds.

5. **Nonce**: a nonce is an abbreviation for "number only used once," and it is a number appended to a hashed block in a Blockchain that matches the difficulty level limitations when rehashed.
6. **CreatedAt**: the block's creation date.
7. **UpdatedAt**: date of update and modification of data.

The description of the hash, previous hash, nonce, createdat, updatedat, and index are all the same for money transfer. The only variation is in the array of data and ChainID which are as follows:

Data: the data is an array that contains transaction information such as the ID of who performed the action, whether it was the client, employee, or administrator, as well as the sender and receiver IDs and the amount sent.

ChainID: the id of the chain.

4.3.2 System interfaces

This section provides an overview of our system by displaying its key web pages and providing a full description of each one.

Home interface

Home **(Figure 4.5)**, this is the initial or main page of our system for all users (administrators, employees, and clients), and it also displays the application's best services. When we click on the login button, we will be sent to the login page **(Figure 4.6)**.

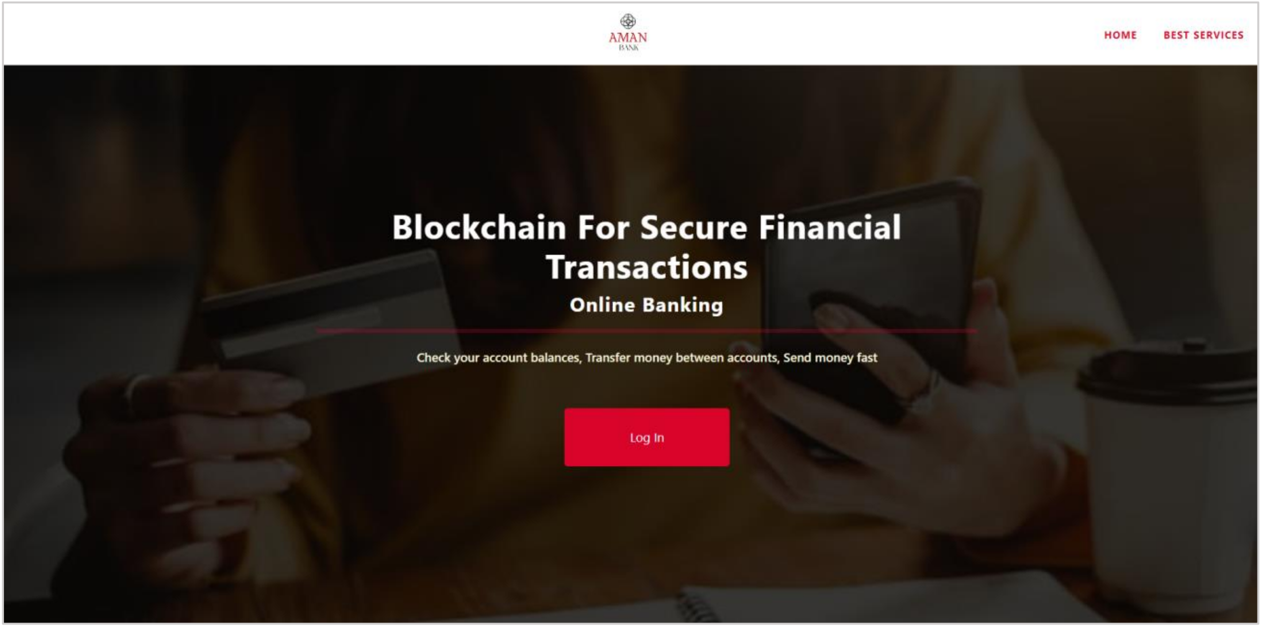


Figure 4. 5: Home interface

When the user gets this interface **(Figure 4.6)**, he will select whether he is a client, employee, or administrator, and then enter his email address and password to log in.

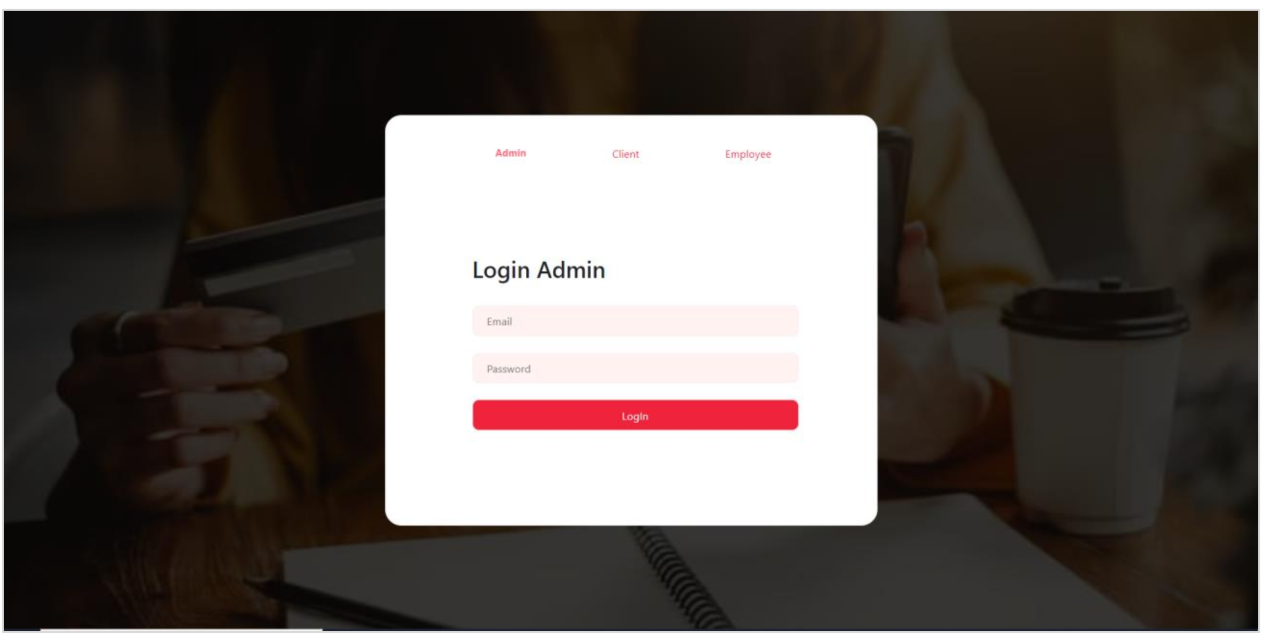


Figure 4. 6: Login interface

Profile of the Client

Once the client is connected, he will receive all of his information **(profile Figure 4.7)**, he will be able to check his balance and change his password **(Figure 4.8)**, and he will be able to view his depositing and withdrawing history by clicking on the button history **(Figure 4.9)**.

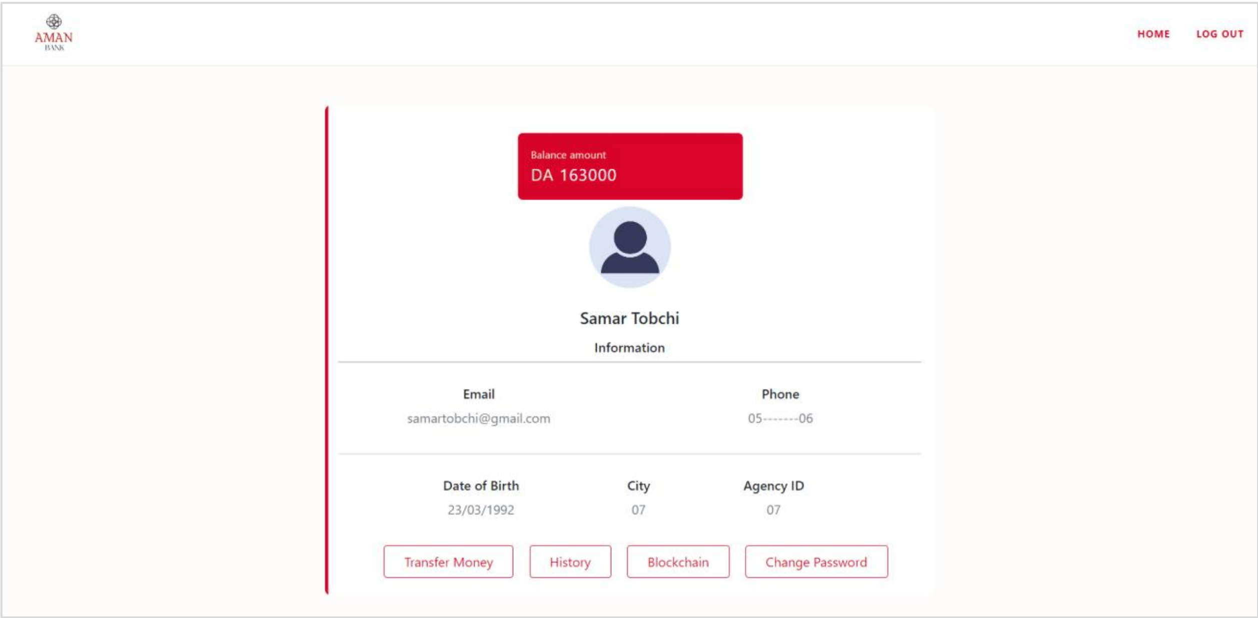


Figure 4. 7: Client profile interface

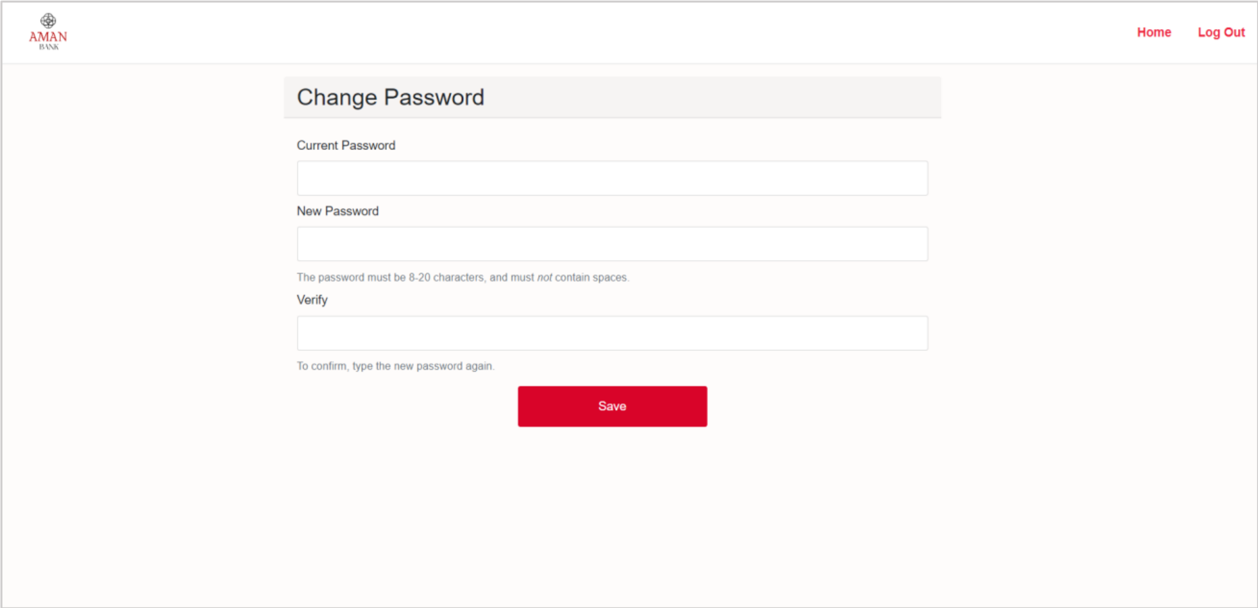


Figure 4. 8: Change password interface

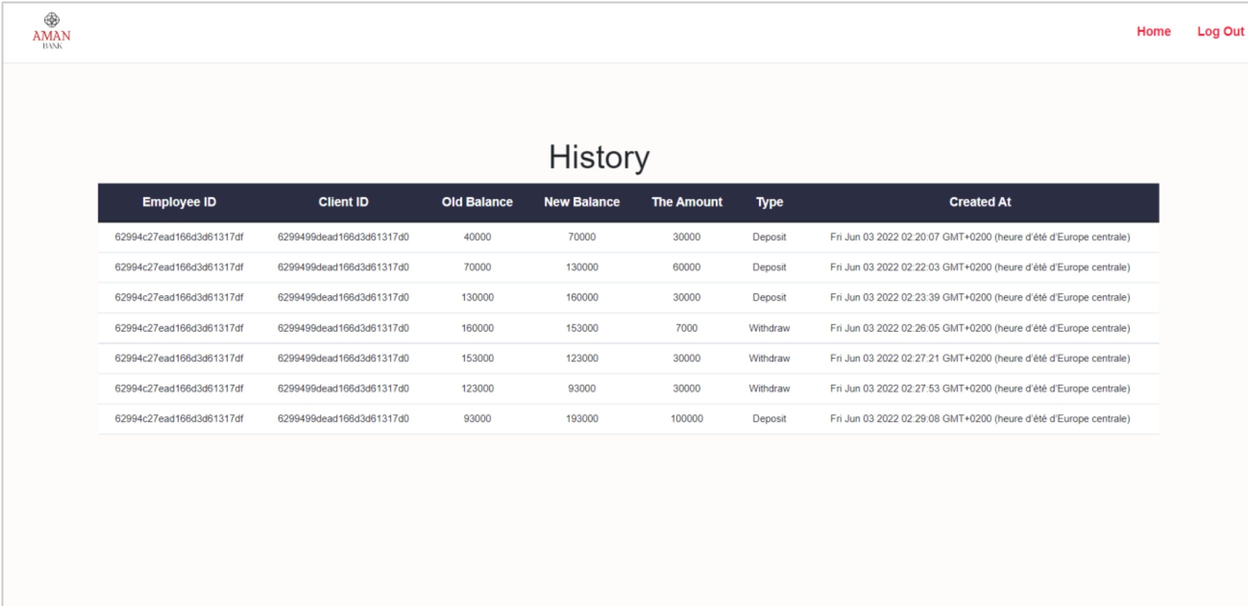


Figure 4. 9: History interface

He can transfer money by clicking the transfer money button **(Figure 4.7)**, and because this action can also be executed by an employee or administrator, he must specify whether he is a client, an employee, or an administrator. Whoever performed this action must enter his ID and password to validate this action, and he must also enter the sender and receiver's IDs, as well as the amount **(Figure 4.10)**.

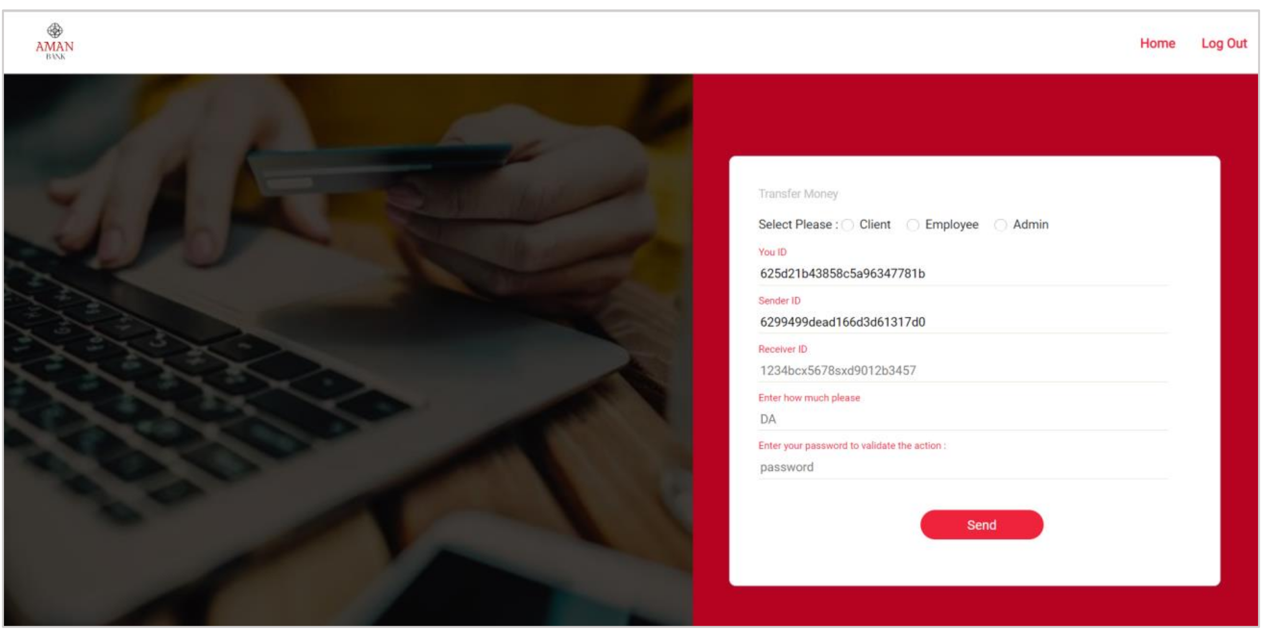


Figure 4. 10: Transfer money interface

When he presses the send button, the funds are transferred and a new block is added to the sender and recipient chains, and he gets the page **(Figure 4.11)** confirming that the new block was successfully added.

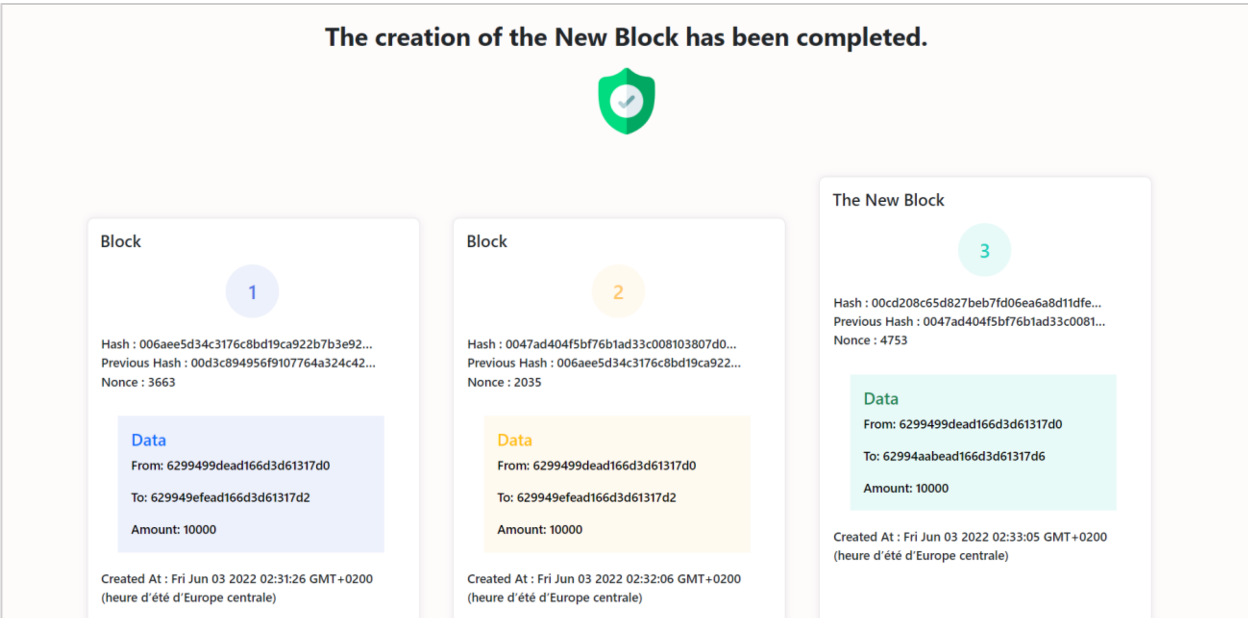


Figure 4. 11: Money transfer the creation of new block

When he clicks on the Blockchain button in Figure **4.7**, he will see all of his chain's blocks beginning with the genesis block, as well as the hash of each block and the hash of the preceding block, which he may compare in order to validate the chain as shown in Figure **4.12**.

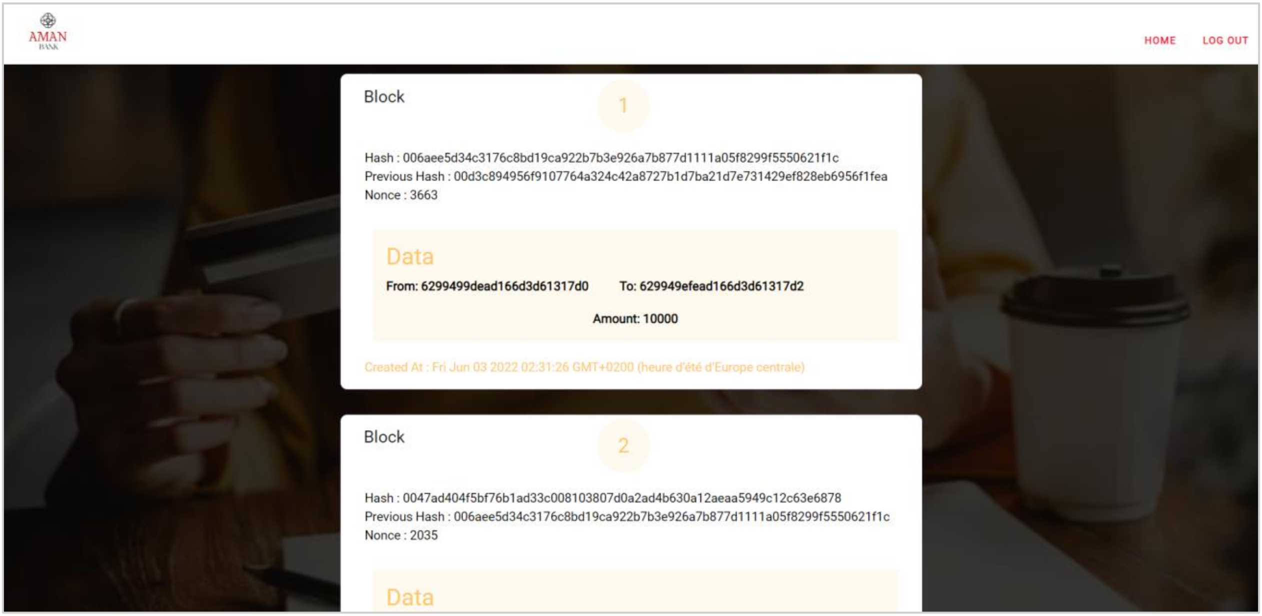


Figure 4. 12: Money transfer Blockchain

Profile of the Administrator

The administrator has access to his profile **(Figure 4.13)** and may edit or modify his data **(Figure 4.14)** or password.

He has access to the whole list of administrators **(Figure 4.15)**, employees **(Figure 4.16)**, and clients **(Figure 4.17)**, and he has the ability to delete an administrator, employee, or client, as well as add one **(Figure 4.18, Figure 4.19)** or update the employee and client data.

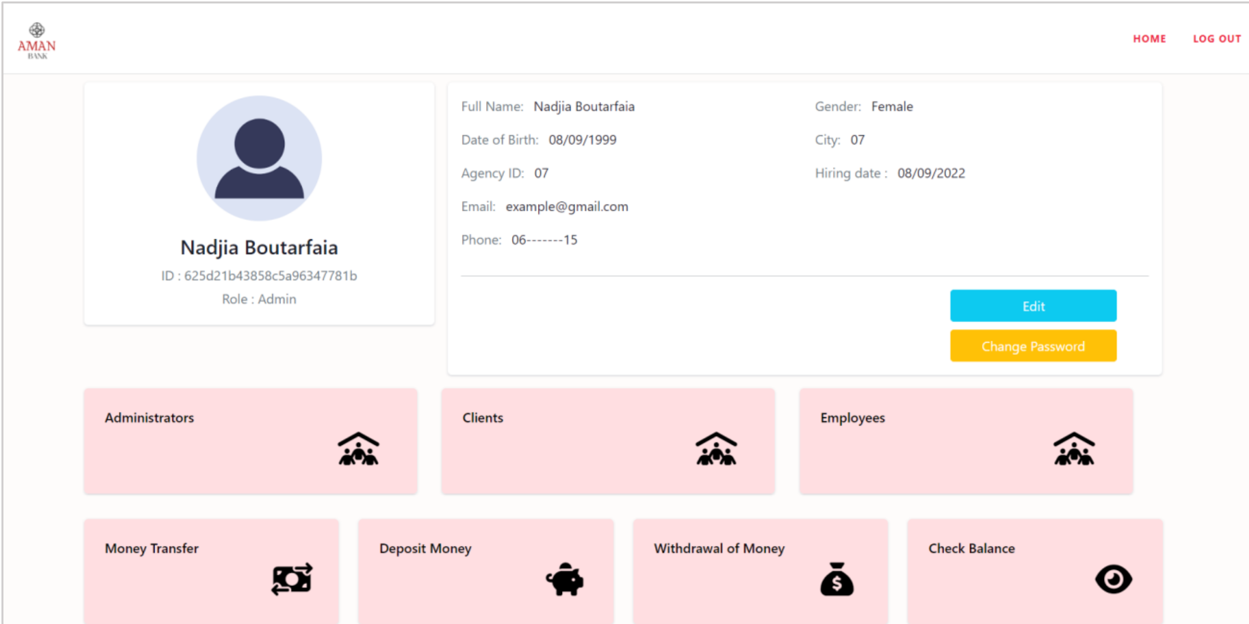


Figure 4. 13: Profile of the Administrator interface

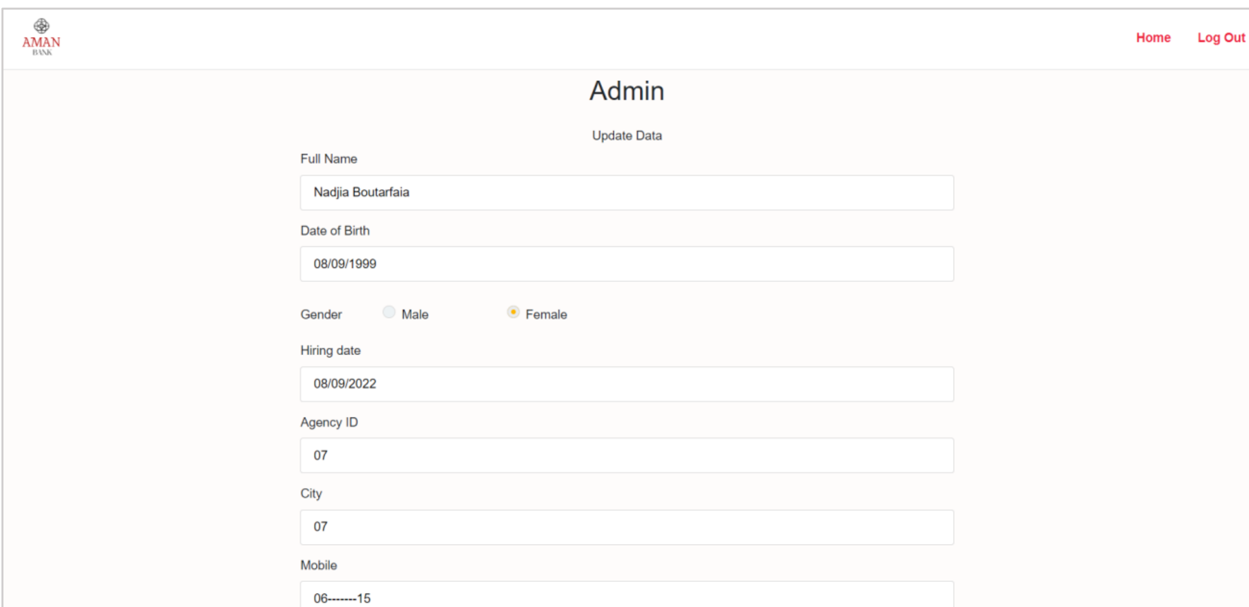


Figure 4. 14: Update admin data interface

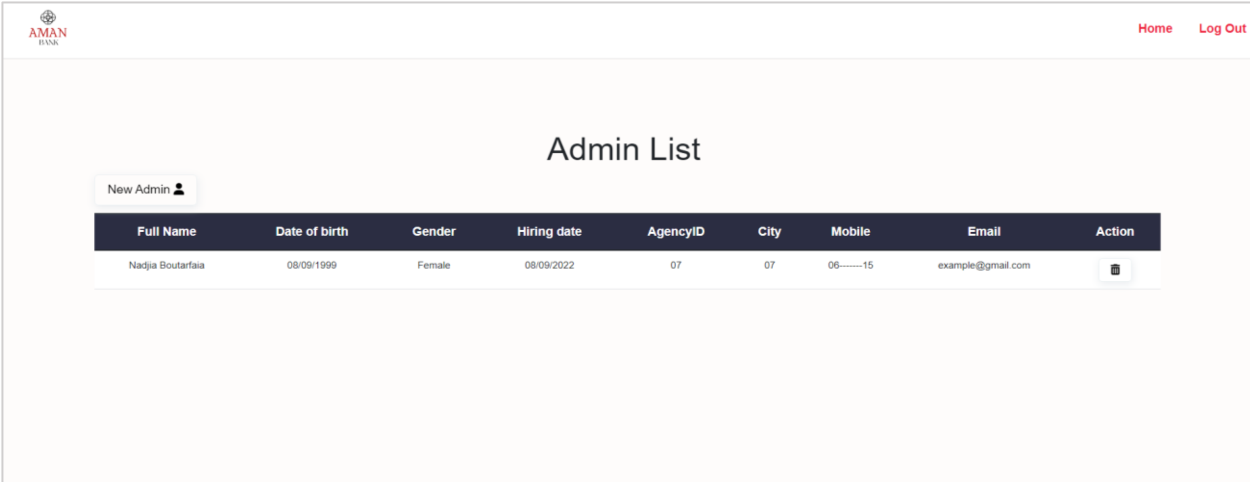


Figure 4. 15: Administrators list interface

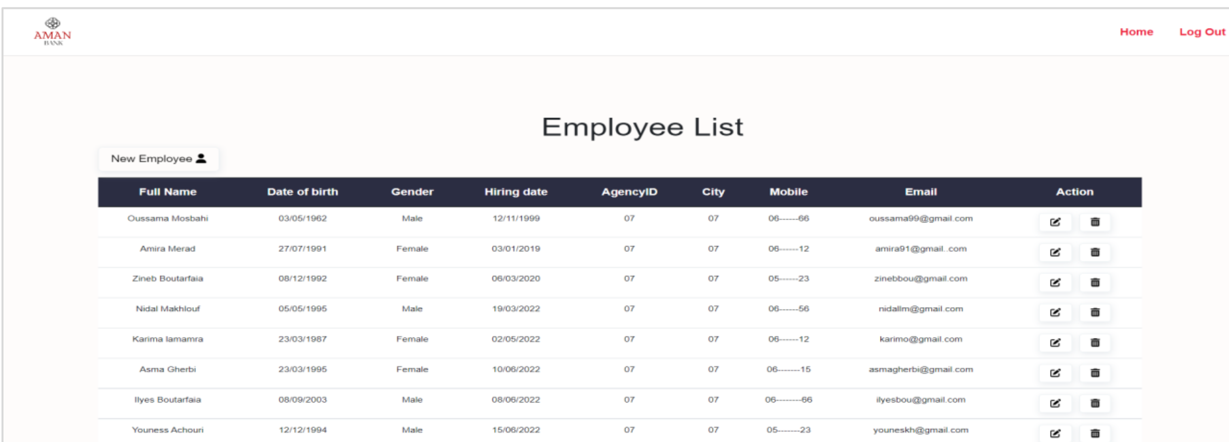


Figure 4. 16: Employee List interface

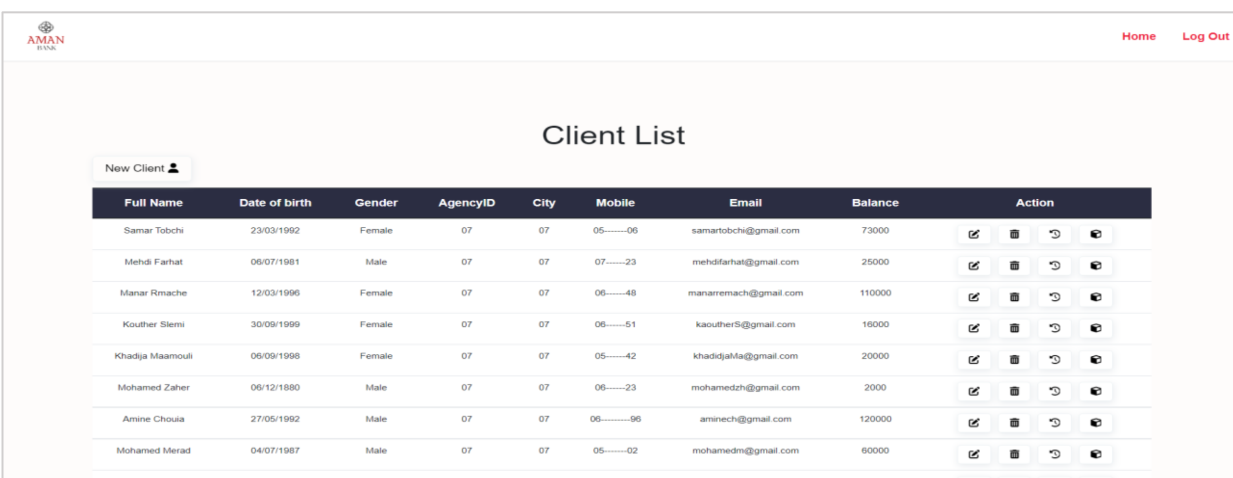


Figure 4. 17: Client list interface

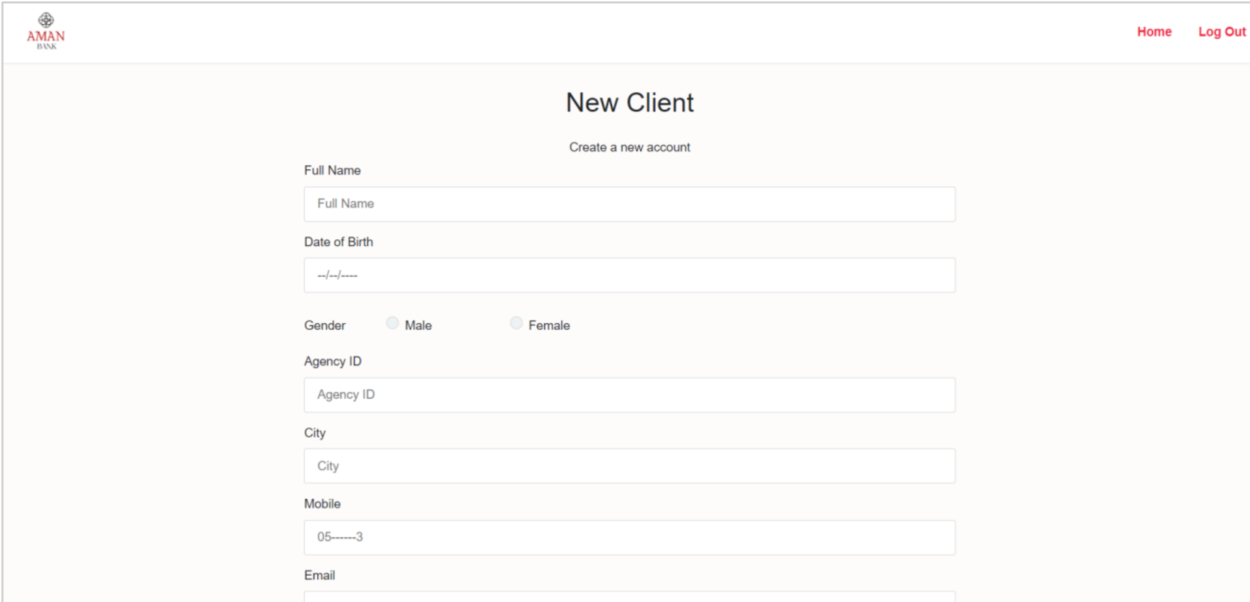


Figure 4. 18: Add new client interface

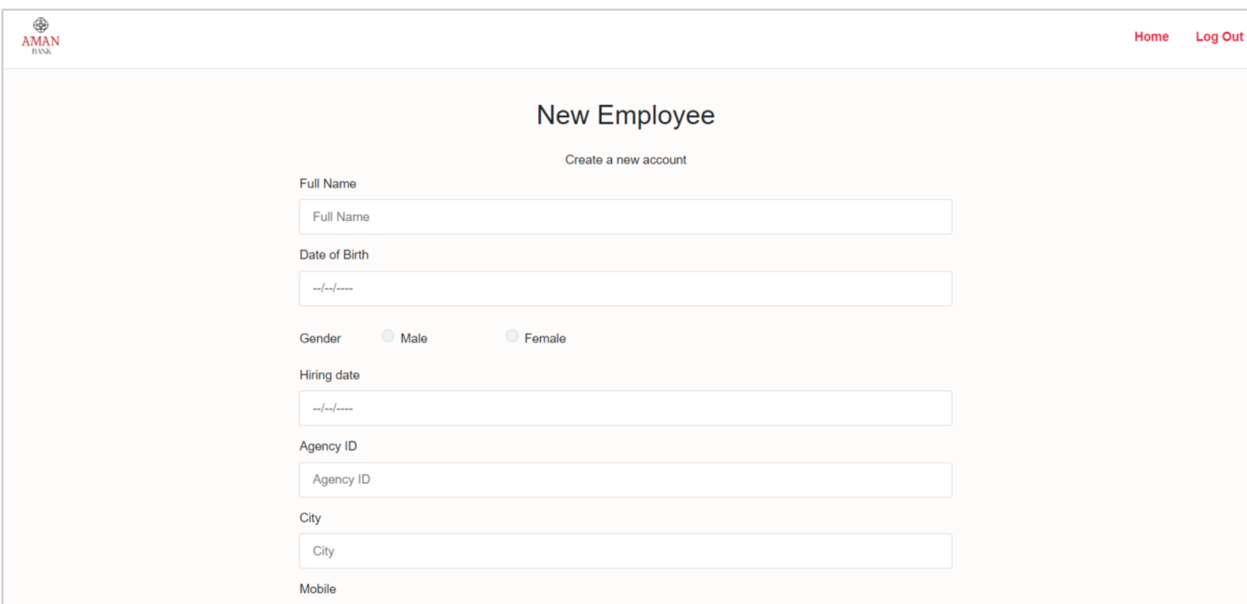


Figure 4. 19: Add new employee interface

The administrator has the ability to perform the four primary client operations of checking the balance, transferring money, withdrawing money, and depositing money. The money transfer interface is the same as we saw before **(Figure 4.10)**. Deposit and withdrawal activities may only be performed by the employee and the administrator, thus each of them must enter their personal ID and password to authenticate the transaction.

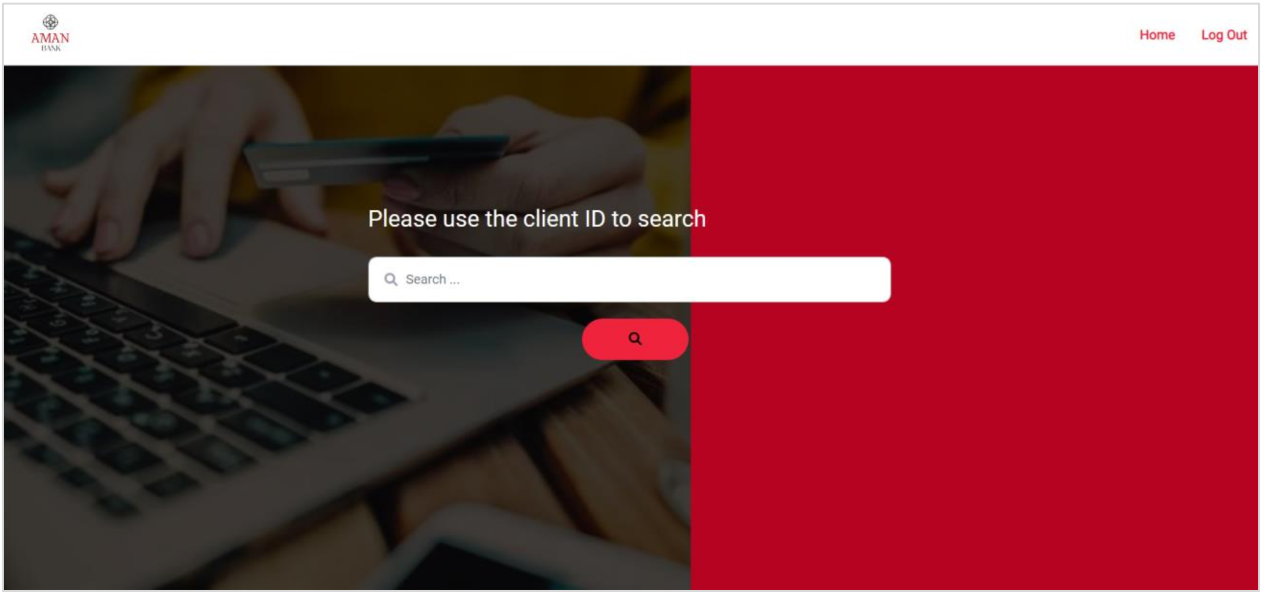


Figure 4. 20: Client search interface

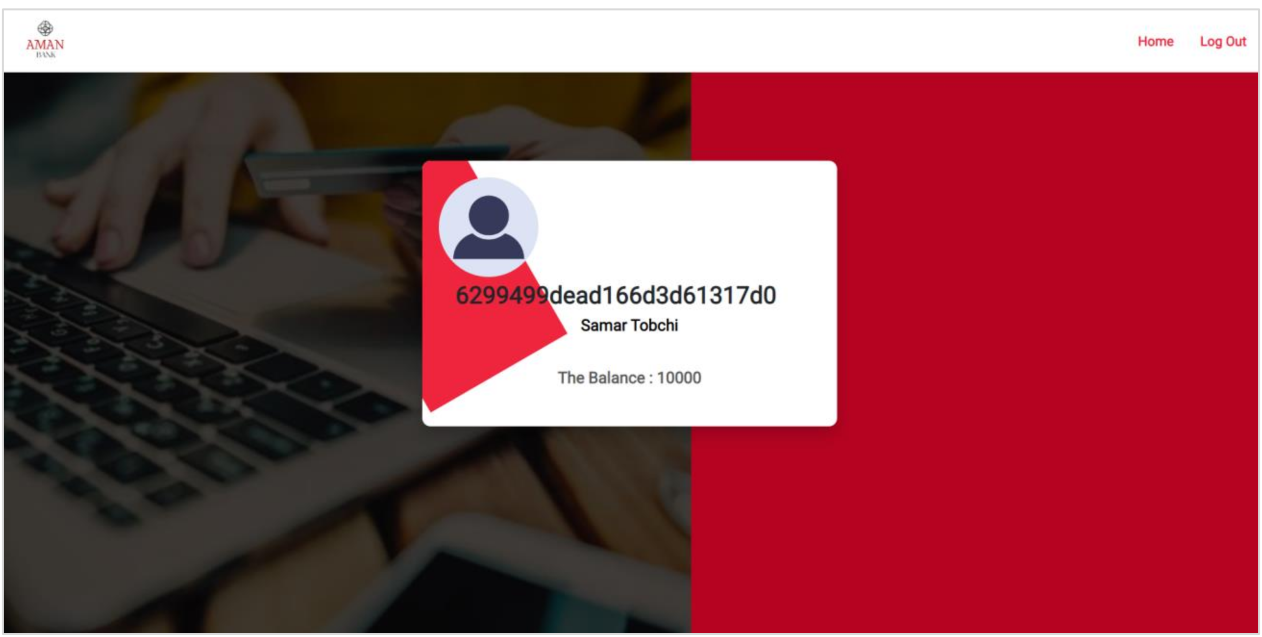


Figure 4. 21: Check the balance of the client interface

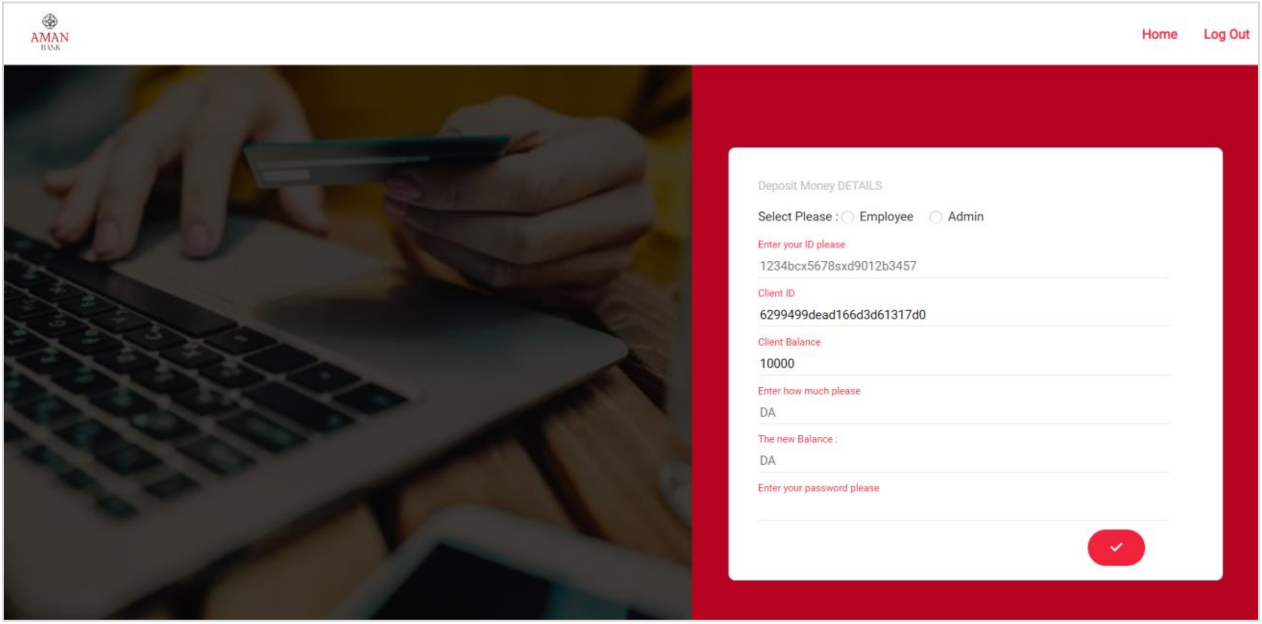


Figure 4. 22: Funds deposit details interface

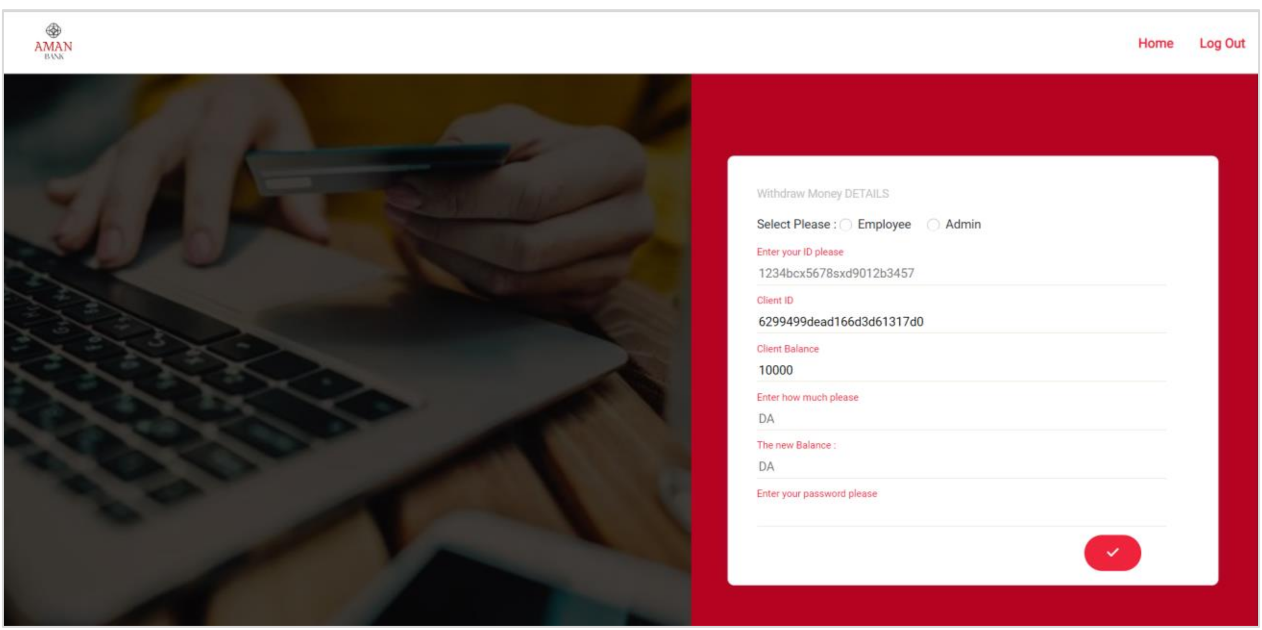


Figure 4. 23: Funds Withdrawal Details Interface

After the administrator or employee enters all the data and the system confirms the transaction, a new block is added to the client chain and gets an interface indicating that adding a new block was successful **(Figure 4.24)**.

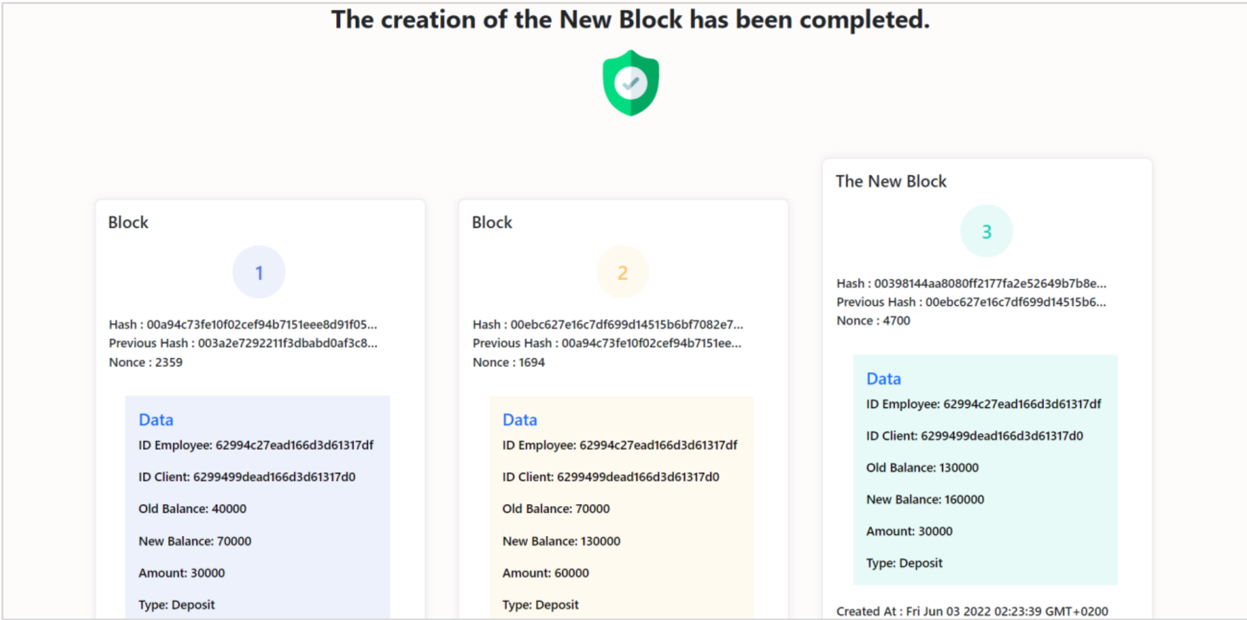


Figure 4. 24: The creation of a new block in the event of a deposit or withdrawal of funds

Profile of the Employee

The employee does not have the same powers as the administrator, but he can for example, check the balance of the client, deposit and withdraw funds. He can also access to the list of clients and view their data, history, and Blockchain. He can also delete or update client data and add new clients. The Figure **4.25** represents an employee's profile.

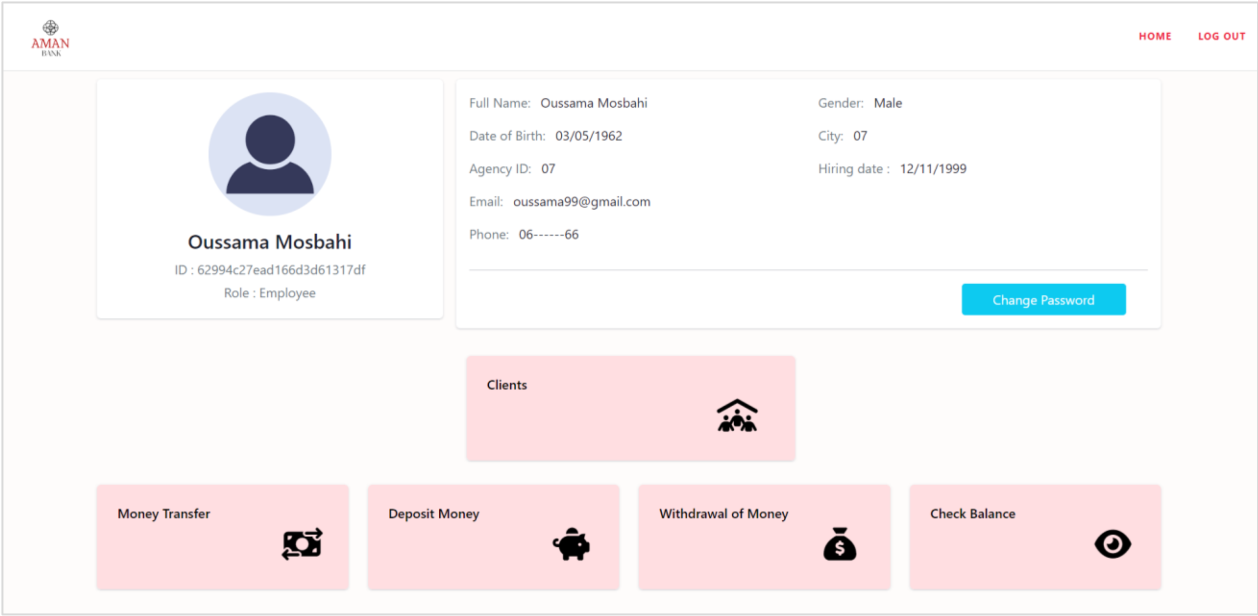


Figure 4. 25: Employee profile page

4.4 Conclusion

We focused on offering an overview of the system in this chapter by presenting the tools and platforms that were used to develop our system and we gave a complete description of the system and the most essential system interfaces in the implementation section. Furthermore, we focus on demonstrating how the Blockchain protocol is used with the process of adding a new block to the chain to provide the user with an overview of how data is saved and safeguarded.

General Conclusion

General Conclusion

Blockchain is one of the most essential and modern data security technologies employed in recent years. Blockchain is used to record transactional information as a digital ledger. The data is saved as signed blocks that are linked together, allowing data to be stored in a secure and immutable manner. The use of Blockchain as a database is quite intriguing, particularly in financial transactions, which contain sensitive information that must be stored in a safe database. The small problem that may arise is the large amount of data that is created every second. The integration of Big Data and Blockchain will create a chain of data that is reliable and trusted, facilitating the analytics phase and leading to better and trusted insights through the transparency and traceability of Blockchain.

As a result, we picked the banking industry to apply the notion of merging Big Data with Blockchain because of the vast volume of data generated by banks every day, as well as the simplicity of transferring funds from anywhere and at any time using a safe system. Using the Blockchain as a database will provide security for banks as the data will be shared with the network that is allowed to view the data (bank employees). In addition, users can check their chain to see if it is valid or not because the Blockchain can be traced using block hashes. Moreover, this technology supports the basic functions of the bank of securely depositing, transferring, and withdrawing funds.

Finally, we can outline some perspectives:

- Including the stage of evaluating the data stored in the Blockchain database and making predictions for clients and banking staff,
- The use of multi-agent systems,
- Analyze stored data in real-time,
- Artificial intelligence tools will be integrated to prevent fraud.

References

[1]. *Statista," Statista,"2020. [Online]. Available:https://www.statista.com/statistics/871513/worldwide-data-created/.*

[2]. *Doug Laney, "3D Data Management: Controlling DataVolume, Velocity, and Variety", Gartner, file No.949. 6 February 2001, http://blogs.gartner.com/doug- laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity- and-Variety.pdf*

[3]. *Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Ullah Khan, S. (2015). The rise of "big data" on cloud computing: Review and open research issues. Information Systems, 47, 98–115. doi:10.1016/j.is.2014.07.006*

[4]. *Bello-Orgaz, G., Jung, J. J., & Camacho, D. (2016). Social big data: Recent achievements and new challenges. Information Fusion, 28, 45–59. doi: 10.1016/j.inffus.2015.08.005*

[5]. *Khan, M. A., Uddin, M. F., & Gupta, N. (2014). Seven V's of Big Data understanding Big Data to extract value. Proceedings of the 2014 Zone 1 Conference of the American Society for Engineering Education. doi:10.1109/aseezone1.2014.682068*

[6]. *Nawsher Khan et al: "The 10 Vs, Issues and Challenges of Big Data", March 2018 DOI:10.1145/3206157.3206166*

[7]. *Arockia Panimalar.S et al: "The 17 V's Of Big Data"International Research Journal of Engineering and Technology (IRJET) Volume: 04 Issue: 09 | Sep-2017.*

[8]. *Abou_el_ela Abdou Hussein: "How Many Old and New Big Data V's Characteristics, Processing Technology, And Applications (BD1)" International Journal of Application or Innovation in Engineering & Management (IJAIEM) Volume 9, Issue 9, September 2020.*

[9]. *Koo, J., Kang, G., & Kim, Y.-G. (2020). Security and Privacy in Big Data Life Cycle: A Survey and Open Challenges. Sustainability, 12(24), 10571. doi:10.3390/su122410571.*

[10]. *Big Data Tools and Techniques: A Roadmap for Predictive Analytics Article · December 2019 DOI: 10.35940/ijeat. B2360.129219.*

[11]. *Apache Spark: A Big Data Processing Engine Conference Paper · November 2019 DOI: 10.1109/MENACOMM46666.2019.8988541.*

[12]. *Big Data Analysis: Apache Storm Perspective Article in International Journal of Computer Trends and Technology · January 2015 DOI: 10.14445/22312803/IJCTT-V19P103.*

[13]. *Hashem, I. A. T., Anuar, N. B., Gani, A., Yaqoob, I., Xia, F., & Khan, S. U. (2016). MapReduce: Review and open challenges. Scientometrics, 109(1), 389–422. doi:10.1007/s11192-016-1945-y.*

[14]. *Ajah, & Nweke. (2019). Big Data and Business Analytics: Trends, Platforms, Success Factors and Applications.Big Dataand Cognitive Computing, 3(2), 32. doi: 10.3390/ bdcc3020032.*

[15]. *Sandra Lovrencic et al, " BIG DATA TRENDS IN DOMAINS OF APPLICATION, " https://doi.org/10.31410/itema.2018.71*

[16]. *Moody Amakobe, " The Impact of Big Data Analytics on the Banking Industry, " July 2015, DOI:10.13140/RG.2.1.1138.4163*

[17]. *Iliashenko, O., Iliashenko, V., & Lukyanchenko, E. (2021). Big Data in Transport Modelling and Planning. Transportation Research Procedia, 54, 900–908. doi: 10.1016/j.trpro.2021.02.145*

[18]. *Chen, J., He, S., & Li, X. (2021). A Study of Big Data Application in Agriculture. Journal of Physics: Conference Series, 1757(1), 012107. doi:10.1088/1742-6596/1757/1/012107*

[19]. *Hossein Hassani et al. Big Data and Climate Change, Big Data Cogn. Comput. 2019, 3, 12; doi:10.3390/bdcc3010012*

[20]. *El-Seoud, S. A., El-Sofany, H. F., Abdelfattah, M. A. F., & Mohamed, R. (2017). Big Data and Cloud Computing: Trends and Challenges. International Journal of Interactive Mobile Technologies (iJIM), 11(2), 34. doi:10.3991/ijim. v11i2.6561*

[21]. *B.Sloot, D.Broeders, and E.Schrijvers, Exploring the Boundaries of Big Data. Amsterdam: Amsterdam University Press, 2016.*

[22]. *In book: HCI Challenges and Privacy Preservation in Big Data, Chapter: Cryptography in Big Data Security, January 2018 DOI:10.4018/978-1-5225-2863-0.ch004.*

[23]. *Website: https://www.proofpoint.com/us/threat-reference/encryption.*

[24]. *Sahai A, Waters B. Fuzzy identity-based encryption. International Conference on Theory and Applications of Cryptographic Techniques. Nice, French Riviera, France: Springer-Verlag, 2005:457-473.*

[25]. *Chaudhari, N., Saini, M., Kumar, A., & Priya, G. (2016). A Review on Attribute Based Encryption. 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN). doi:10.1109/cicn.2016.81.*

[26]. *Lee, C. C., Chung, P. S., & Hwang, M. S. (2013). A Survey on Attribute-based Encryption Schemes of Access Control in Cloud Environments. IJ Network Security, 15(4), 231-240.*

[27]. *Milanov E., (2009), "The RSA Algorithm" https://sites.math. washington. edu/~morrow/336_09/ papers/Yevgeny.pdf.*

[28]. *Bansal, Neha & Singh, Sukhdeep. (2020). RSA Encryption and Decryption System. International Journal of Scientific Research in Computer Science, Engineering and Information Technology. 109-113. 10.32628/CSEIT206520.*

[29]. *Website: https://101blockchains.com/history-of-blockchain-timeline/*

[30]. *Daniela Mechkaroska, Aleksandra Popovska-Mitrovikj, & Vesna Dimitrova. (2018). Secure big data and iot with implementation of blockchain. INTERNATIONAL SCIENTIFIC JOURNAL « SECURITY & FUTURE », 183-185.*

[31]. *Javaid, M., Haleem, A., Pratap Singh, R., Khan, S., & Suman, R. (2021). Blockchain technology applications for Industry 4.0: A literature-based review. Blockchain: Research and Applications, 100027. doi: 10.1016/j.bcra.2021.100027.*

[32]. *Berdik, D., Otoum, S., Schmidt, N., Porter, D., & Jararweh, Y. (2021). A Survey on Blockchain for Information Systems Management and Security. Information Processing & Management, 58(1), 102397. doi: 10.1016/j.ipm.2020.102397.*

[33]. *A. Lastovetska, "Blockchain architecture basics: Components, structure, benefits and creation." https://mlsdev.com/blog/ 156-how-to-build-your-own-blockchain-architecture, January 31, 2019.*

[34]. *Zhonghua Zhang, Xifei Song, Lei Liu, Jie Yin, Yu Wang, and Dapeng Lan. Recent Advances in Blockchain and Artificial Intelligence Integration: Feasibility Analysis, Research Issues, Applications, Challenges, and Future Work.*

[35]. *Website:https://www.foley.com/en/insights/publications/2021/08/types-of-blockchain-public-private-between.*

[36]. *Paul, P.K., Aithal, P.S., Saavedra, R. and Ghosh, S. (2021). Blockchain Technology and its Types—A Short Review. IJASE., 9(02): 189-200.*

[37]. *Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. 2017 IEEE International Congress on Big Data (BigData Congress). doi:10.1109/bigdatacongress.2017.*

[38]. *Nascimento S. (ed), Pólvora A. (ed), Anderberg A., Andonova E., Bellia M., Calès L., Inamorato dos Santos A., Kounelis I., Nai Fovino I., Petracco Giudici M., Papanagiotou E., Sobolewski M., Rossetti F., Spirito L., Blockchain Now And Tomorrow: Assessing Multidimensional Impacts of Distributed Ledger Technologies – executive summary, EUR 29813 EN, Publications Office of the European Union, Luxembourg, 2019, ISBN 978-92-76-08972-8, doi:10.2760/305417, JRC117255*

[39]. *Website: https://medium.com/@yazannajjar44/what-is-blockchain-how-dose-blockchain-work-4ee99c0d4e9b*

[40]. *B. Rawat, D., Chaudhary, V., & Doku, R. (2020). Blockchain Technology: Emerging Applications and Use Cases for Secure and Trustworthy Smart Systems. Journal of Cybersecurity and Privacy, 1(1), 4–18. doi:10.3390/jcp1010002.*

[41]. *Book: Fusing Big Data, Blockchain and Cryptocurrency.https://link.springer.com/chapter/10.1007/978-3-030-31391-3_2*

[42]. *Epstein, J. 2017. "The CMO Primer for the Blockchain World". Silver Spring, MD: Never Stop Marketing.*

[43]. *J. Chen, Z. Lv, and H. Song, "Design of personnel big data management system based on blockchain," Future Generation Computer Systems, vol. 101, pp. 1122–1129, Dec. 2019.*

[44]. *A. A. Monrat, O. Schelén, and K. Andersson, "A survey of blockchain from the perspectives of applications, challenges, and opportunities," IEEE Access, vol. 7, pp. 117 134–117 151, 2019.*

[45]. *N. DEEPA et al, "A Survey on Blockchain for Big Data: Approaches, Opportunities,and Future Directions, " ACM Comput. Surv., Vol. 1, No. 1, Article. Publication date: September 2020.*

[46]. *M. A. Rahman, M. M. Rashid, M. S. Hossain, E. Hassanain, M. F. Alhamid, and M. Guizani, "Blockchain and iot-based cognitive edge framework for sharing economy services in a smart city," IEEE Access, vol. 7, pp. 18 611–18 621, 2019.*

[47]. *Yue, X., Wang, H., Jin, D., Li, M., & Jiang, W. (2016). Healthcare Data Gateways: Found Healthcare Intelligence on Blockchain with Novel Privacy Risk Control. Journal of Medical Systems, 40(10). doi:10.1007/s10916-016-0574-6.*

[48]. *Cocîrlea, Dobre, C., Hîrţan, L.-A., & Purnichescu-Purtan, R. (2020). Blockchain in Intelligent Transportation Systems. Electronics, 9(10), 1682. https://doi.org/10.3390/electronics9101682.*

Appendix A

Appendix

We will introduce the back end of our application and the strategies used to build the system in this section; we will not take all of the application code, but instead we will focus on the security approaches covered in Chapter 03's security layer in the global architecture.

A.1 The MVC model

We used the MVC model or model-view-controller to create our application, which is referred to as a software architectural model. This architecture, which is currently extensively utilized in web application projects, due to its popularity to its high maintainability and ease of use in collaborative work. As shown in Figure **A.1**, our project (FinalApp) is divided into three folders: models, controllers, and views.

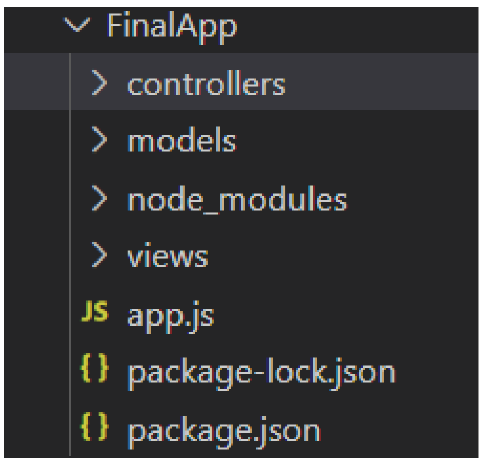


Figure A. 1: Project Structure

A.1.1 System models

In this section, we will offer our data models (or schemas):

- Figure **A. 2** shows the admin schema,
- Figure **A. 3** shows the employee schema,
- Figure **A. 4** shows the client schema,
- Figure **A. 5** shows the transfer schema,
- Figure **A. 6** shows the withdrawal and deposit schema.

```javascript
const mongoose = require('mongoose');
const { required } = require('nodemon/lib/config');
const Schema = mongoose.Schema;

const adminSchema = new Schema({

    fullname : {
        type: String,
        required: true
    },

    dateofbirth: String,

    gender : {
        type: String,
        required: true
    },

    dateofhiring : {
        type: String,
        required: true
    },

    AgencyID : {
        type: String,
        required: true
    },

    city : {
        type: String,
        required: true
    },

    mobile : {
        type: String,
        required: true
    },

    email : {
        type: String,
        required: true
    },

    password : {
        type: String,
        required: true
    },

}, {timestamps: true});

const Admin = mongoose.model('Admin', adminSchema);

module.exports = Admin;
```

Figure A. 2: The Admin model

```javascript
const mongoose = require('mongoose');
const { required } = require('nodemon/lib/config');
const Schema = mongoose.Schema;

const employeeSchema = new Schema({

    fullname : {
        type: String,
        required: true
    },

    dateofbirth: String,

    gender : {
        type: String,
        required: true
    },

    dateofhiring : {
        type: String,
        required: true
    },

    AgencyID : {
        type: String,
        required: true
    },

    city : {
        type: String,
        required: true
    },

    mobile : {
        type: String,
        required: true
    },

    email : {
        type: String,
        required: true
    },

    password : {
        type: String,
        required: true
    },

}, {timestamps: true});

const Employee = mongoose.model('Employee', employeeSchema);

module.exports = Employee;
```

Figure A. 3: The Employee model

```javascript
const mongoose = require('mongoose');
const { required } = require('nodemon/lib/config');
const Schema = mongoose.Schema;

const clientSchema = new Schema({

    fullname : {
        type: String,
        required: true
    },

    dateofbirth: String,

    gender : {
        type: String,
        required: true
    },

    AgencyID : {
        type: String,
        required: true
    },

    city : {
        type: String,
        required: true
    },

    mobile : {
        type: String,
        required: true
    },

    email : {
        type: String,
        required: true
    },

    password : {
        type: String,
        required: true
    },

    balance : {
        type: String,
        required: true
    },

}, {timestamps: true});

const Client = mongoose.model('Client', clientSchema);

module.exports = Client;
```

Figure A. 4: The Client model

```javascript
const mongoose = require('mongoose');
const { required } = require('nodemon/lib/config');
const Schema = mongoose.Schema;

const TransactionSchema = new Schema({

    index : {
        type: String,
        required: true
    },

    Hash : {
        type: String,
        required: true
    },

    PreviousHash : {
        type: String,
        required: true
    },

    ChaineID : {
        type: String,
        required: true
    },

    Data : [{ Whodid : String ,From: String, To: String, Amount: String }],

    Nonce : {
        type : Number,
        required: true
    },

}, {timestamps: true});

const Transaction = mongoose.model('Transaction', TransactionSchema);

module.exports = Transaction;
```

Figure A. 5: The Transfer model

```
const mongoose = require('mongoose');
const { required } = require('nodemon/lib/config');
const Schema = mongoose.Schema;

const moneySchema = new Schema({


    index : {
        type: String,
        required: true
    },

    Hash : {
        type: String,
        required: true
    },

    PreviousHash : {
        type: String,
        required: true
    },

    Data : [{ idemployee: String, idclient: String, oldbalance: String, newbalance: String, Amount: String, ttype: String }],

    Nonce : {
        type : Number,
        required: true
    },
}, {timestamps: true});

const Money = mongoose.model('Money', moneySchema);

module.exports = Money;
```

Figure A. 6: The withdrawal and deposit model

A.2 The Connection to the MongoDB Atlas

We enter the URL of the database, the database name, and the password to connect to the mongodb cloud Atlas.

```
//DataBase Mongodb

const dbURI = 'mongodb+srv://DataBank:1234@cluster0.tzdse.mongodb.net/Bankdata?retryWrites=true&w=majority';
mongoose.connect(dbURI, { useNewUrlParser: true, useUnifiedTopology: true })
 .then((result) => console.log("connected to database"))
 .catch((err) => console.log(err));
```

Figure A. 7: Database connection

A.3 Creating a new account and logging in

When the client controller receives a request from the view to add a new client, it creates a model object (client) and saves the data, then we use the bcrypt library to hash the password, first generating the salt, then hashing the password (plain text) with the salt and saving it with data.

```
const client_create = (req, res) => {
    const client = new Client(req.body);

    bcrypt.genSalt(10, (err, salt) =>
    bcrypt.hash(client.password, salt, (err, hash) => {
      if (err) throw err;
      client.password = hash;
      client
        .save()
        .then(res.redirect("/add-client"))
        .catch((err) => console.log(err));
    })

    )
}
```

Figure A. 8: The creation of a new account for client

Figure **A.9** shows the login function. After the client enters his email and password, we obtain the client's data using his email if it exists, then compare the hashed password to the inputted password using bcrypt compare to see if they match or not.

(Note that the employee and the administrator will be treated in the same way.)

```javascript
const login_cleint = (req, res) => {
    const { email, password } = req.body;
    if (!email || !password) {
      console.log("Please fill in all the fields");
      res.render("loginclient", {
        email,
        password,
      });
    } else {
        Client.findOne({ email: email })
        .then((user) => {
          if (!user) {
            console.log("wrong email");
          }
          //Match Password
          bcrypt.compare(password, user.password, (error, isMatch) => {
            if (error) throw error;
            if (isMatch) {
              res.render("clientprofile", {pdata: user});
            } else {
              console.log("Wrong password");
            }
          });
        })
        .catch((error) => console.log(error));
    }
}
```

Figure A. 9: Login function for client

A.4 Blockchain

The primary functions needed to accomplish the Blockchain process will be discussed in this section.

To begin developing the Blockchain, we must first produce the first block referred to as the "genesis Block." The function below (CreateGenesisBlock) will be in responsible of constructing the genesis block for a chain.

```
const CreateGenesisBlock = () => {
    var h = CalculateHash(0, "null", "0000", 0);
    var nonce = 0;
    var hash = MineBlock(2, h, nonce, 0, "null", "0000");
    const transaction = new Transaction({
        index : '0',
        PreviousHash : '0000',
        ChaineID : "null",
        Data: [{Whodid: "null", From: "null", To: "null", Amount: "null"}],
        Hash : hash,
        Nonce : nonce
    });

    transaction.save()
    .then(console.log('The creation of the genesis Block has been completed.'))
    .catch((err) => console.log(err));

}
```

Figure A. 10: Create the Genesis Block function

We employ a Proof of Work method because if we don't include it, a new block will be created quickly if we don't have any security measures in place. To begin, we must first develop a function that generates the block's hash based on the nonce, index, previous hash, and the data (**Figure A. 11**).

```
const CalculateHash = (index, data, previousHash, nonce) => {
    return SHA256(index + previousHash + JSON.stringify(data) + nonce ).toString();
}
```

Figure A. 11: Calculate hash function

MineBlock is a function that identifies a number that is supplied as a difficulty parameter. The difficulty determines how many zeros are appended at the front of the hash. If a machine tries to mine the block, the added degree of protection will slow down the operation significantly.

```
const MineBlock = (difficulty, hash, nonce, index, data, previousHash) => {
    while (hash.substring(0, difficulty) !== Array(difficulty+1).join("0")){
        nonce++;
        hash = CalculateHash(index, data, previousHash, nonce);
    }

    return hash;

}
```

Figure A. 12: MineBlock function

Moving on to the validation procedure in the event of money transfer, we examine the request's validity and return an error; we have five situations as discussed before, and if everything is OK, we will return zero (**Figure A.13**).

```javascript
const Validation = async (doer, sender, receiver, amount, password, type) => {

    var error ;
    var dr ;
    const rec = await Client.findById(receiver);
    const sed = await Client.findById(sender);

    if(type == "Client"){
        dr = await Client.findById(doer);
    }else{
        if(type == "Employee"){
            dr = await Employee.findById(doer);
        }
        else{
            dr = await Admin.findById(doer);
        }
    }

    if(!dr){error = 5 ; return error ;
    }else{
        const isMatch = await bcrypt.compare(password, dr.password);
        if(isMatch === false){error = 3 ; return error};
    }
    if(!rec){error = 1 ; return error ;}
    if(!sed){error = 2 ; return error ;
    }else{
          if(amount >= parseInt(sed.balance)){
            error = 4 ; return error ;
        }
    }
    return 0;
}
```

Figure A. 13: Validation function

After the validation is done, the result set will determine whether we continue with the process of adding a new block, or something is wrong. If the verification returns zero, we will add the transfer amount to the recipient and withdraw the amount from the sender. After that, we'll use the addblock function (**Figure A.15**).

```
const AddNewBlock = async (req, res) =>{

    const Doer = req.body.doer;
    const From = req.body.from;
    const To = req.body.to;
    const Amount = req.body.amount;
    const Password = req.body.password;
    const Type = req.body.type;
    var r = await Validation(Doer, From, To, Amount, Password, Type);

    if(r == 0){
        Client.findOne({ _id: From })
        .then((client) => {
          client.balance = parseInt(client.balance) - parseInt(Amount);
          client.save(client);
        });

        Client.findOne({ _id: To })
        .then((client) => {
          client.balance = parseInt(client.balance) + parseInt(Amount);
          client.save(client);
        });

        addblock(Doer, From, To, Amount);
        var threebloks = await getthethreedocuments(From);
        res.render('doneBlock', {id : From, data: threebloks});
    }else{
        var message ;
         if(r == 1) { message ="Receiver does not exist"};
         if(r == 2) { message ="Sender does not exist"};
         if(r == 3) { message ="Your password is incorrect"};
         if(r == 4) { message ="Oops your balance is not enough"};
         if(r == 5) { message ="Your ID is incorrect"};

        res.render('TransferMoney', {id : Doer, data : From , message : message});
    }
};
```

Figure A. 14: AddNewBlock function

We'll obtain the last block in the function addblock, or the genesis block if it doesn't exist, and the block's hash will be added to the new block as the previous hash, we'll use the function BlockTransaction (**Figure A.16**) to add the new block to the chain.

```
const addblock = (Doer, From, To, Amount) =>{
    var Block = GetTheLastBlock(From);
    Block.then((result) => {
        if(!result){
            var Genesis = GetGenesisBlock();
            Genesis.then((res) => {
               var prvhash = res.Hash;
               var ind = parseInt(res.index)+1;
               var data = [{Whodid: Doer, From: From, To: To, Amount: Amount}];
               var h = CalculateHash(ind, data, prvhash, 0);
               var nonce = Math.floor(Math.random() * (5000 - 1000) + 1000);
               var hash = MineBlock(2, h, nonce, ind, data, prvhash);
               BlockTransaction(ind, prvhash, From, data, hash, nonce);
            })
        }else{
            var prvhash = result.Hash;
            var ind = parseInt(result.index)+1;
            var data = [{Whodid: Doer, From: From, To: To, Amount: Amount}];
            var h = CalculateHash(ind, data, prvhash, 0);
            var nonce = Math.floor(Math.random() * (5000 - 1000) + 1000);
            var hash = MineBlock(2, h, nonce, ind, data, prvhash);
            BlockTransaction(ind, prvhash, From, data, hash, nonce);

        }
    });
    var Block = GetTheLastBlock(To);
    Block.then((result) => {
        if(!result){
            var Genesis = GetGenesisBlock();
            Genesis.then((res) => {
               var prvhash = res.Hash;
               var ind = parseInt(res.index)+1;
               var data = [{Whodid: Doer, From: From, To: To, Amount: Amount}];
               var h = CalculateHash(ind, data, prvhash, 0);
               var nonce = Math.floor(Math.random() * (5000 - 1000) + 1000);
               var hash = MineBlock(2, h, nonce, ind, data, prvhash);
               BlockTransaction(ind, prvhash, To, data, hash, nonce);
            })
        }else{
            var prvhash = result.Hash;
            var ind = parseInt(result.index)+1;
            var data = [{Whodid: Doer, From: From, To: To, Amount: Amount}];
            var h = CalculateHash(ind, data, prvhash, 0);
            var nonce = Math.floor(Math.random() * (5000 - 1000) + 1000);
            var hash = MineBlock(2, h, nonce, ind, data, prvhash);
            BlockTransaction(ind, prvhash, To, data, hash, nonce);

        }
    });
}
```

Figure A. 15: addblock function

```
const BlockTransaction = (ind, prvhash, chID, data, hash, nonce) =>{
    const transaction = new Transaction({
        index : ind,
        PreviousHash : prvhash,
        ChaineID : chID,
        Data: data,
        Hash : hash,
        Nonce : nonce
    });

    transaction.save()
    .then(console.log('The creation of the New Block has been completed.'))
    .catch((err) => console.log(err));
}
```

Figure A. 16: BlockTransaction function

```
const GetGenesisBlock = () => {
    return Transaction.findOne();
}

const GetTheLastBlock = (id) => {
    return Transaction.findOne({ChaineID : id}).sort({'_id':-1});
}
```

Figure A. 17: Get the genesis block and last block functions

```javascript
const GetalllistofBlock = (req, res) => {
    const id = req.params.id;

    Transaction.find({ $or: [{ChaineID : id}, {index : "0"}] })
    .then((result) => {
       res.render('BlockchainTransfer', { data: result });
    })
    .catch((err) => {
        console.log(err);
    })
}
```

Figure A. 18: Get Blockchain

Note: in the case of money transfer, all of the preceding phases were applied, and the blocks for withdrawing and depositing money are generated in the same way.