# INDEX

**INTRODUCTION**………………………………………………………………………………………………..…3

**1 BLOCKCHAIN OVERVIEW**……………………………………………………………………..6

1.1 PEER-TO-PEER SYSTEM …………………………………………………………………………….7

1.2 LEDGER……………………………………………………….………………..…………………………7

1.3 SMART CONTRACTS………………….……………………………………………………………….8

1.3 ORDERED AND CONNECTED BLOCKS………………………………………………………….8

1.3 INTEGRITY…………..………………….……………………………………………………………..10

**2 CURRENT BUSINESS-TO-BUSINESS PURCHASING TRANSACTION**…………………………………………………………………………………….……11

**3 TRIPLE-ENTRY ACCOUNTING**……………………………………………………………15

3.1 TRIPLE-ENTRY SYSTEMS………..………………………………………………………………..15

3.2 TRIPLE-ENTRY ACCOUNTING VERSUS TRIPLE-ENTRY BOOKKEEPING…..…..…16

3.3 TRIPLE-ENTRY ACCOUNTING FEATURES………………………………..………………...18

3.3.1 Triple-entry accounting features………………………………………………………..18

**4 MAKE A BUSINESS-TO-BUSINESS DEAL ON BLOCKCHAIN**….21

4.1 DIFFERENCES BETWEEN CURRENT AND FUTURE BUSSINESS-TO-BUSSINESS TRANSACTIONS………………………………………………………………………………………………….21

4.2 PROTOTYPE ANALYSIS………………………………………………………………………………….23

4.3 DATABASE……………………………………………………………………………………………………24

4.3.1 Invoice…………………………………………………………………………………………..24

4.3.2 Blockchain phases…………………………………………………………….…………….25

4.3.3 Account balance…………………….………………………………………….……………25

4.3.4 Participants…………………………………………………………………………….………25

4.3.5 VAT…………………………………………………………………………………….…………25

4.3.6 Public ledger………………………………………………………..…………………………25

4.3.7 Private ledger……….…………………………………………..……………………………26

4.3.8 Asset ID’s……………………………………………………………….………………………26

4.3.9 Balance sheet details……………………….………………………………………………26

4.4 FUNCTION..…………………………………………………………………….……………………………27

4.4.1 Insert table data………………………..……………………………………………………27

4.4.2 Buyer’s bank account……………………….………………………………………………27

4.4.3 Seller’s shipment…………………………………………..…………………………………28

4.4.4 Buyer’s confirmation……………………………………………..…………………………28

4.4.5 Update score…………………………………………………………………………..………28

4.4.6 Paying account transaction…………………………………….…………………………30

4.4.7 Public ledger………………………………..…………………………………………………30

4.4.8 Buyer’s ledger…………………………………………………………………………………31

4.4.9 Update first details……………………………………………….…………………………32

4.4.10 Update details………………………………………………………………….……………32

4.4.11 Export Excel balance sheet………………………………………..……………………33

4.4.12 Update Excel balance sheet………………………………………………….…………33

**5 CONCLUSION**…………………………………………………………………………………………..………34

# *INTRODUCTION*

This research aims to describe how blockchain will affect accounting and auditing through a prototype that consists of a small Python program[[1]](#footnote-1). My study provides a brief introduction to blockchain technology and its main components, a description of triple-entry accounting, and how business-to-business transactions can be affected by the use of blockchain. The second part of the study explains a prototype code that demonstrates how a daily raw material purchasing transaction can be implemented in a blockchain-based system.

Although publications addressing the potential for blockchain to revolutionize accounting and auditing began to emerge in 2016 (Erica Pimentel et al., 2020: 340), most of the existing papers still have a theoretical footprint that gives only a rough overview of the subject (Erica Pimentel et al., 2020: 342; C. W. Cai, 2019, 81). This choice could be explained by the difficulty accessing information and the non-use of this technology by most companies. Also, E. Pimentel et al. (2020) suggest that redundant theoretical papers continue to be published due to the long lag between when papers are submitted to journals and the date of publication (revision time).

On the contrary, this research gives a tangible solution implemented through a Python model. Although the latter contains many simplifications, it highlights advantages of the blockchain-based system in the business area comparing the prototype to current business tools: an increase of transparency which allows accountants/governments to access rapidly information, while the adoption of smart contracts, combined with artificial intelligence, leads to cost decreasing and real-time information (Moinak Maiti et al., 2021: 2; C. W. Cai, 2019, 71; J. Schmitz and G. Leoni, 2019: 331).

The implementation of this Python project leads the research not to focus on all aspects of accounting and blockchain. Consequently smart contracts, the role played by accountants and auditors, and the choice of public or private blockchains are some topics that this publication does not deal with. Nevertheless, they play a fundamental role in the development of this area. On the contrary, this research reinforces the concepts introduced by J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al. (2021) and Ke Wang et al. (2017), adding a practical point of view.

This paper is divided into four main sections, which aim to provide all necessary tools to understand how the code, meaning a business-to-business transaction, works.

Section 1 describes blockchain as a technology. After giving a complete definition of the latter, single parts of the definition itself are described one by one. It is not a detailed description of the topic but aims to define the most critical concepts.

Section 2 explains how a current transaction between two companies is conducted nowadays: from the research of the seller to the record on the ledgers. It points out all possible issues that can arise in terms of fraud, costs, and time consumption.

Section 3 defines triple-entry accounting (TEA), its benefits, and its implementation on a blockchain system. It starts from the birth of this concept and traces progress made. Although many researchers cite Yuji Ijiri as the “inventor” of TEA, his concept was totally different from what it is meant in this paper and had nothing to do with blockchain (C. W. Cai, 2019, 72). Consequently, this research uses Grigg’s definition of TEA, introduced in 2005.

The introduction of Section 4 compares current and blockchain-based business-to-business transactions, while the main body deals with the code itself. Every Python function, representing a portion of the code, is analyzed deeply. The paper provides a description of all parties involved, how the function works, benefits and risks compared to current centralized software, and what simplifications have been made. Furthermore, in order to have a better comprehension of the code, single tables of the database are briefly described.

In conclusion of the foreword, it is necessary to highlight that this research is just a possible way of how accounting will evolve in the future, meaning it is impossible to predict how the sector will evolve. Many simplifications have been made due to a lack of empirical material, as proved by the absence of big firms developing a blockchain-based system focused on this topic.

# BLOCKCHAIN OVERVIEW

Blockchain is viewed as one of the most promising and disruptive inventions, and it is considered to have the potential to significantly change many businesses (Ke Wang et al., 2017: 101). As Figure 1 shows, it is essentially a distributed database, which means that all participants own it, simultaneously shared among participating parties. All blocks are linked to each other, forming “a chain of blocks”, which allows preserving immutability. The majority of the system participants verify each transaction in the public ledger. (Micheal Crosby et al., 2015: 3)

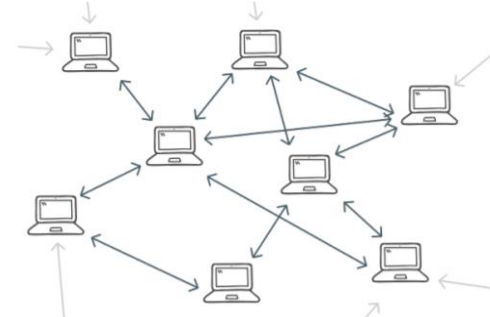


Figure 1 A distributed database: All participants are connected directly or indirectly to each other.

Source: Soares (2018)

A more exhaustive definition is the following:

*“The blockchain is a purely distributed peer-to-peer system of ledgers that utilizes a software unit that consists of an algorithm, which negotiates the informational content of ordered and connected blocks of data together with cryptographic and security technologies in order to achieve and maintain its integrity” (Daniel Drescher, 2017: 35)*

The following paragraphs focus on the main concepts contained in the definition above[[2]](#footnote-2). Furthermore, smart contracts are quickly described to understand the code found in Section 4.

## Peer-to-Peer System

Peer-to-peer systems are a special kind of distributed system. Daniel Drescher (2017) and J. I. Ibañez (2017) describe such system as a set of independent computers or nodes (with a local memory) which communicates to each other through message passing. This comes in opposition to the alternative of having one shared memory, i.e., of being a “parallel system,” such that the collection of independent nodes appears to the users as a single computer. Peer-to-peer systems have a further feature: each node (i.e., computing device) has the same roles and rights. Furthermore, they share their computing power over the network and coordinate their efforts to complete the goal the network has been created for. All the nodes are both suppliers and consumers of resources (Daniel Schreder, 2017: 23, 146).

If such a system is secure enough, it has the potential to replace intermediaries with the consequence of decreasing transactional costs (N. Elommal et R. Manita 2022: 39).

## Ledger

Ledgers can be seen as containers that gather every data occurring for a given account. For instance, bookkeepers preserve accounting records, which consist of debits, credits, and transaction amounts, on a ledger (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 8). In a blockchain, after a transaction occurs, it is reordered on the ledger managing to maintain chronological order. Furthermore, since blockchain is a peer-to-peer system, all participants keep their own ledger with the consequence that it must be shared with everyone after transactions are recorded. As described in paragraph 4, every block contains a ledger.

## Smart Contract

A smart contract is a computer code stored on the blockchain that executes actions under specified circumstances. Their auto-execution allows reducing the risk of error or manipulation and not needing a third-party intermediate. (Erica Primentel et al., 2020, 331). The ability of these contracts to self-execute[[3]](#footnote-3) is one of the main advantages of adopting them. In order to achieve self-execution, all terms and conditions must be pre-defined (J. Schmitz and G. Leoni, 2019, 336; C. W. Cai, 2019, 76). These two features enable to improve efficiency of business processes and save costs.

For instance, they could represent the terms and conditions of legal contracts, such as commercial agreements, governed by a set of rules established by the contractual parties (Erica Primentel et al., 2020, 331). Hamilton (2020: 10) provides an overview of several smart contract applications, including simplifying insurance claim processing, efficiency in mortgage approval and loan servicing, copyright protection, and smart identities.

## Ordered and Connected Blocks[[4]](#footnote-4)

As the name “Blockchain” suggests, this technology is made by many blocks connected to each other, as shown in Figure 2.

Every block contains, among the other features, four kinds of data:

* Time: this technology is timestamped, meaning that all nodes can check when a block was mined, i.e., when it was accepted.
* Hash: it is a string of fixed length which is related uniquely to any kind of data[[5]](#footnote-5). All data contained in a block are transformed into this kind of string. Hash functions are change-sensitive, meaning that any change in data outputs a completely different string. Block hash is calculated by miners/validators[[6]](#footnote-6) who are rewarded for this task (Daniel Schreder, 2017: 72).
* Previous hash: it is fundamental to keep immutability. As Figure 2 shows, one data change in any block would make the block hash invalid, but the latter is also contained to the next block; consequently, data of the next block would change, making its hash invalid. Including the previous hash prevents people from changing previous blocks.
* Ledger: Blocks obviously contain data stored in the ledgers.

The first block is called the ”genesis block”, and it differs from the others because it does not contain the previous block’s hash.

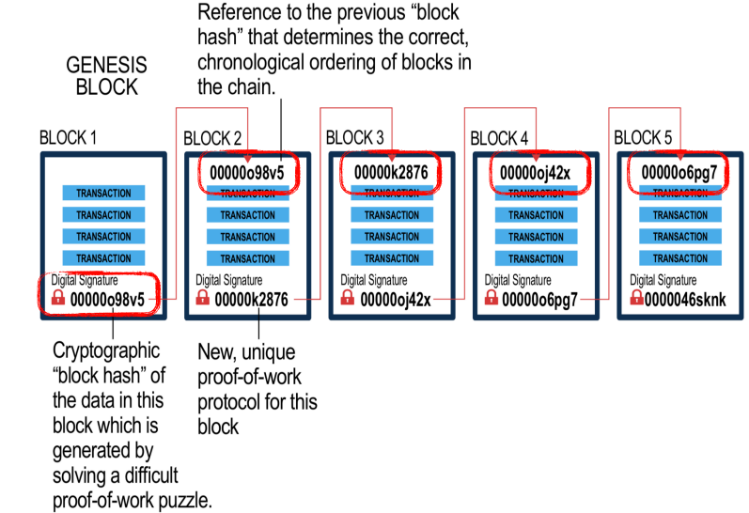


Figure 2: Connection between blocks: all blocks contain the unique code of the previous block, allowing to link all blocks to each other   
Source: Natarajan, Krause, and Gradstein (2017)

## Integrity

Integrity is a non-functional aspect of a system to be safe, consistent, correct, and free of corruption and errors (Daniel Schreder, 2017: 6; EY blockchain report, 2016).  
In essence, blockchain system integrity is the ability to make true and verified statements about ownership. Integrity is a fundamental aspect of all software systems: a loss of data, illogical software behavior, or realizing that strangers were able to access your private data are some examples of integrity lack (Daniel Schreder, 2017: 6).

# CURRENT BUSINESS-TO-BUSINESS PURCHASING TRANSACTION

After a brief overview of how blockchain works, a specific application of this technology is now described: how it will affect the accounting and auditing sector. This paper focuses only on the business-to-business purchasing process for the following motivations: (I) it allows to explore deeper a specific aspect; (II) it highlights not exclusively how the process itself could change but also changes in company and auditor’s roles; (III) the concept of triple-entry accounting, probably one of the most analyzed concepts in this field, fits well in this process; (IV) it highlights how it could prevent VAT evasion which is one of the most unpaid taxes: in Italy, the government estimates almost 27 billion per year (NADEF 2021[[7]](#footnote-7)).

Before studying the impact of blockchain, this section tracks current business-to-business purchase steps in order to highlight all issues that could occur.

In the last twenty years, the huge adoption of the internet allowed companies to access a worldwide market and increase the efficiency of many activities. However, still, most of the supply process is costly and time-consuming.

According to G. Vicari (2019)[[8]](#footnote-8), every time a new purchase is made, the first step for the buyer (company “A”) consists of finding a seller (company “B”). Depending on the kind of goods, “A” searches either on business-to-business online platforms or directly contacts the firm. The latter’s advantage is to avoid platform fees and might have a more modifiable offer, but it is mostly impossible to trust the other party. On the other hand, “B” cannot be sure that “A” pays for the goods either. In this scenario, both parties bear transaction costs including all activities to get information about company reliability. If “A” chooses to use an intermediate, the latter guarantees any inconvenience and resolves possible disputes, but costs increase due to fees.

Stipulating a contract is the next phase of the purchase process. Terms and conditions are often interpretable, and, in the case “A” and “B” belong to different countries, they might be subject to two separate jurisdictions. Furthermore, in the best scenario where both parties behave properly, potential problems arise when an inconvenience turns into a dispute. Indeed, lawyers might get involved, or the company might choose to avoid legal action due to extra costs[[9]](#footnote-9), while being aware of having suffered damage.

The third step consists of recording the transaction. “A” documents it in its own ledger, while “B” does the same in its own ERP database resulting in four entries being recorded: it is redundant. Moreover, since ledgers are kept private, companies could manipulate the books. Governments, after some scandals[[10]](#footnote-10), have taken some precautions in order to lower this kind of episode (Lori Zulauf et al., 2002: 52); external auditors or board of auditors are some measures which have been adopted (Lori Zulauf et al., 2002: 52). They might be effective, but they increased bookkeeping costs. In addition, separated ledgers are prone to human error (J. Schmitz and G. Leoni, 2019: 338).

After a while from the moment the transaction occurred, accountants have the task of reconciling accounts (N. Elommal et R. Manita 2022: 50). They are supposed to ask journals or general ledgers to the company they are working for and bank accounts to the bank itself. Once they have collected all papers, they start to verify sampling transactions (N. Elommal et R. Manita 2022: 52). If they find an error, accountants ask company account employees what has happened; the latency between the transaction and the reconciliation causes a certain difficulty to trace it backward. Furthermore, sampling transactions involve at least three different parties who could be reluctant to share information to each other. Also, such accounts could be manipulated. Reconciliation becomes harder for bank transactions that do not link to an invoice or when the latter is paid in cash.

Finally, accountants are responsible for paying taxes. Value-added tax (VAT[[11]](#footnote-11)) is one of the most used indirect taxes, which affects mostly all transactions. VAT reconciliation is costly, time-consuming, and is prone to tax evasion[[12]](#footnote-12). Micheal Keen et al. (2007: 8) state that the most common kinds of VAT evasion are the following:

* Parties agree not to issue an invoice which is still possible using cash payments. Due to VAT deductibility, companies operating in the European context should be disincentivized to conduct business-to-business payments off the books. In reality, there are many cases where it is convenient for both parties. Under-the-table payments are even more convenient when a private customer is involved: VAT theoretically falls fully on them, but companies can increase their prices, consequently, their margins, not including it.
* Although an invoice is regularly issued, a company merely does not pay VAT. Revenue agencies struggle to trace it due to the time gap and the huge number of transactions. The introduction of electronic invoices combined with the increase in digital payments has partially resolved this problem. Although many companies are aware of risks, they consider it a sort of loan: sanctions are compared to an installment with higher interests.

J. Schmitz and G. Leoni (2019: 336) state that most financial statement evaluations are made at the end of the fiscal year. Several months of latency could be a problem for authorities who struggle to investigate events that happened several months before and for companies that could not realize promptly if their economic-financial situation is compromised. To obtain more often updated data on the actual health of the company, huge firms draw up monthly reports, which cause extra costs.

In conclusion, even when all steps are carried out meticulously, this process is labor-intensive and prone to human error; for instance, an invoice could be paid twice (Moinak Maiti et al., 2021: 1; J. Schmitz and G. Leoni, 2019: 336): Pacioli's double-entry accounting allowed to decrease mistakes, but auditors' reconciliation and verification are still costly from both an economic and timing point of view (Erica Pimentel et al., 2020: 328; Ke Wang et al., 2017:104, J. Schmitz and G. Leoni, 2019: 336). The birth of blockchain has brought to light a new concept called triple-entry accounting which has the potential to resolve most of the problems highlighted (C. W. Cai, 2019: 71, 77).

# TRIPLE-ENTRY ACCOUNTING

## Triple-Entry Systems

Any triple-entry system is an inter-party bookkeeping system that keeps a single transaction record by sharing a repository (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 9).

One of the main issues when the concept was first introduced was security threats: it was unclear who would have acted as a third party (i.e., who would have enabled, notified, stored, and timestamped transactions in a shared ledger) (C. W. Cai, 2019: 77; J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 17). Nowadays, however, blockchain technology seems to be the most efficient method to maintain a shared record in a three-dimensional manner (C. W. Cai, 2019: 72; J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 16): its immutability[[13]](#footnote-13) allows not to corrupt transactions being recorded, being public and distributed permits anybody to participate, while cryptography can be used to restrict access to sensitive data, and its consensus mechanism incentivizes nodes to be honest[[14]](#footnote-14) (C. W. Cai, 2019: 77; Ian Grigg, 2016).

In order to update the shared record with a new transaction, two parties need to be involved: one party adds its signature to the transaction entry draft, and the counterparty accepts it by countersigning (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021:15).

Before the entry gets processed, the share transaction repository, i.e., nodes of the blockchain system, checks the validity of the signatures and then, if everything is in order, signs off on the entry (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021:15). Actually, this process is carried out by validators or miners whose task consists of confirming transaction blocks as in any other blockchain. The three entries in TEA are not three mirroring records of a transaction but three signature entries or signed messages. This means that TEA records may be compatible with both a single-entry and a double-entry representation of transactions (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021:14; C. W. Cai, 2019: 72)

## Triple-Entry Accounting versus Triple-Entry Bookkeeping

It is often stated that the Bitcoin blockchain is a triple-entry accounting system. Yet, not only most of the blockchain developments are about finance, but also it would be redundant to explain what triple-entry accounting is (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 4). As Grigg (2019) suggests, it is more appropriate to differentiate between triple-entry bookkeeping (TEB) and TEA.

As Figure 3 shows, TEB simply records the transaction in a share transaction repository (e.g., Bitcoin blockchain), while TEA involves an accounting software built on top of a TEB (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 14; Moinak Maiti et al., 2021: 2). Consequentially, accounting and bookkeeping records differ since transactions are recorded analytically in the former, sequentially in the latter: transactions are classified in order to show information meaningful for business life, for instance, it facilitates decision-making or financial reporting (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 14; Moinak Maiti et al., 2021: 2). Thus, the accounting process takes the information of the journal and posts it in a second book, where information is organized analytically. This book is known as a ledger. As a consequence, accounting happens in ledgers, whereas bookkeeping is limited to journals (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 14; Moinak Maiti et al., 2021: 2).

For example, suppose an asset transaction happens. In that case, a TEB system simply records it (Figure 3), giving information about the parties involved, the amount of money, the kind of assets, and a brief description. While in a TEA system, the asset bought could be assigned to an asset ledger which allows real-time calculation of amortization, a sophisticated system could automatize tax payments involved, such as VAT, or payment reconciliation could be done immediately: it could either be connected to the “payment” blockchain layer if cryptocurrencies have been used or ask bank confirmation (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 14; Moinak Maiti et al., 2021: 3).

Immagine che contiene tavolo

Descrizione generata automaticamente

Figure 3: Triple-entry bookkeeping: it shows private and common ledgers. Source: Ledgerium

TEA systems work similarly to TEB systems meaning that they will allow participants to record transactions in a shared ledger, but they will take a further step: they provide some additional services consisting of organizing information and automatizing some mechanic operations (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 14; Moinak Maiti et al., 2021: 3).

In conclusion, triple-entry accounting systems have two different goals. The first consists of creating an immutable shared ledger where transactions are recorded chronologically, which aims to prevent fraud and lead to lower redundancy. On the one hand, a single source of truth can resolve many disputes between companies; on the other hand, it helps the government’s control because the agreement of a second company makes falsifying books harder and it prevents tax reconciliation, in particular VAT evasion (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 13). Indeed, many companies do not pay taxes even once invoices are issued because they know that revenue agencies struggle to track all transactions. This goal can be achieved by TEB systems as well. However, it is not true to state that it can automatically prevent fraud or money laundering, but “it is tough to lie when everybody is watching” (Ian Grigg).

TEA systems also aim to facilitate auditor’s and accountant’s work. Instantaneous reconciliation, lower redundancy, low-cost and real-time auditing, and invoice automatization are some benefits where TEA can have an impact (Moinak Maiti et al., 2021: 7; EY report, 2016; Ke Wang et al., 2017:104; J. Schmitz and G. Leoni, 2019: 336)

## Triple-Entry Accounting Features[[15]](#footnote-15)

In order to implement a triple-entry system that records transactions, some essential characteristics are needed (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 15). However, additional features build a TEA ecosystem. They can be considered non-essential characteristics, but some of them are necessary to build a full-fledged TEA network that does not only record transactions but is also able to analyze them (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021:15). On the contrary, building a complex system due to the adoption of many characteristics could hinder its wide adoption. At the birth of blockchain-based platforms, it is necessary to implement user-friendly systems which can be integrated with current ERP databases and tools, which means that “distant concepts[[16]](#footnote-16)” should not be considered at first (J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al., 2021: 18; C.W. Cai, 2019: 82; Deloitte, 2017: 2).

### Features

J. I. [Ibañez](http://blockchain.cs.ucl.ac.uk/user/jibanez/) et al. (2021: 16) and Ke Wang et al. (2017:104) gathered the following features as the most important:

* Validation by three parties: all receipts are signed (i.e., confirmed) by both companies involved and blockchain nodes through consensus mechanisms.
* Distributed Ledger: the shared ledger is set up on a peer-to-peer system with independent nodes communicating through message passing. The distributed ledger is a decentralized, blockchain-based system. This feature is essential for nodes and companies lacking mutual trust.
* Bookkeeping and accounting layer: transactions should be first recorded in chronological order. Then, data should be assigned to each party’s general ledger (e.g., bank payables are collected altogether).
* Immutability of transactions: Achieving immutability is fundamental to trust that the other party will not alter the dominating record in their favor. In principle, a TEA system without this security feature is conceivable. However, Grigg’s three-way consensus through digitally signed receipts should ensure immutability by design. To be sure, such systems must be public, and validators are independent of the companies involved.
* Digital identity verification: As immutability, this characteristic is essential to avoid record alterations. It can be easily obtained through public and private keys as any other crypto wallet works, but problems could emerge anyway. The absence of a central authority implies that forgotten passwords are unrecoverable; on the contrary, discovered passwords are a serious threat to company data.
* Privacy issues: While private blockchains are considered less efficient due to access restrictions to validators and companies, public blockchains struggle with privacy concerns. A strong cryptography system must be implemented to allow only involved parties to see data while allowing nodes to validate transactions without knowing the content inside it. There are even some “imaginative” solutions: for instance, trusted computing which is a software able to know more than users[[17]](#footnote-17). Another privacy issue consists of what documents are consultable to participants and how to hide some content while keeping the system public.
* Compliance with financial and accounting regulations: In theory, a TEA does not need to be compliant with local laws. However, higher chances of the spread of this system require to be user-friendly to accountants and companies.
* Network-based payment: a TEA system that enables cryptocurrency payments can partially automatize transaction records and decrease reconciliation costs. However, cryptocurrencies are not ready yet due to high volatility and the absence of a dominant cryptocurrency.
* Smart contracts-enabling network: network-based settlement/payment does not presuppose smart contracts, but the latter might enhance the appeal of the former by making the network more versatile. Furthermore, smart contracts may play a role in automating accounting practices.
* Actual tools network: In order to have a wider adoption, it is requested to develop an intuitive platform. The possibility to communicate with current accountant tools makes it more user-friendly.
* Linked to private ledgers: companies would still need to keep private ledgers to have data stored. C. W. Cai (2019: 71) suggests that “business entities will only need to perform a single-entry internally, and the opposite entry will be recorded in a shared public ledger.” Linking private and public ledgers keeps both sides secure and immutable. A further implementation layer[[18]](#footnote-18) is supposed to be created where these private journals can be held.

# IMPLEMENTING A BUSINESS-TO-BUSINESS DEAL ON BLOCKCHAIN

After having defined current transactions, blockchain-based business-to-business deals are now described. This section is divided into two parts: the first aims to give a general overview of the TEA prototype, while the second analyzes single functions.

The project was developed in Python, one of the most famous and simple code languages.

## Differences between Current and Future Business-To-Business Transactions

This paragraph retraces most of the steps analyzed in Section 2 from a blockchain-based system point of view. Some phases could not be covered, but it does not mean they will not be affected by this technology.

In order to avoid third-party platforms while still keeping trust between parties, smart contracts are involved: they would allow encoding all contract terms and conditions, making lawyers useless[[19]](#footnote-19) (Ke Wang et al., 2017, 102). In this scenario, smart contract coders, supported by auditors, play a fundamental role since such contracts could regulate most of the transactions. Since terms and conditions are automized[[20]](#footnote-20), the contract, i.e., the blockchain where it runs, should be able to track all steps of the transaction: from the beginning of the deal to the conclusion. Consequently, in order to decrease the number of disputes, it is supposed to recognize at what stage parties are (e.g., if goods were sent, received, and the payment was carried out) involving artificial intelligence to detect good damages and third parties as transport companies. These contracts should reduce the costs of fraud, reconciliation, and transaction (N. Elommal and R. Manita, 2022: 52): for instance, banks are no longer required. Finally, smart contracts resolve evasion problems regarding the payment of taxes. Indeed, the amount of VAT could be previously calculated either by parties or artificial intelligence, and governments should force companies to implement a system that is able to automatize VAT payments (Van-Cam Nguyen 2019: 77). In the latter scenario, revenues agencies could focus on checking smart contract compliance rather than tracking single transactions.

Currently, once the deal is over, both companies record the transaction in their ledgers. At the same time, a blockchain-based system should have a TEB system that allows having a shared ledger where transactions should be signed (i.e., confirmed) by both parties and validators/miners. Giving access to the public ledger to auditors would reduce information latency and, at the same time, would increase transparency (Moinak Maiti et al., 2021: 2; C. W. Cai, 2019: 76). This prototype contains private ledgers in order to facilitate companies to keep books as they are currently used. In order to preserve immutability, they are linked to the public ledger. Three ledgers might seem redundant, but it has an advantage: it is closer to current Pacioli’s double-entry concept, which means it favors a wider adoption. In the future, private ledgers will probably disappear.

At this point, accountants and auditors are supposed to reconcile bank statements with “economic transactions” such as invoices and payrolls. As highlighted in Section 2, auditors and accountants use sampling techniques due to a lack of time. Blockchain, combined with artificial intelligence, could bypass this process because the full transaction history and supporting evidence would always be available to auditors (N. Elommal et al. R. Manita 2022: 49, 51). Furthermore, blockchain transparency and immutability will decrease data threats: anybody should not be able to change either financial transactions or public/private ledgers (C. W. Cai, 2019: 81).

Finally, a blockchain-based system brings companies and their auditors toward a continuous audit process: monitoring all steps of any transaction, paying taxes immediately, updating asset values and inventories, and drawing up instantaneous financial statements. This last feature is crucial for both government and companies to figure out respectively law and economic issues on time.

As with any blockchain system, validators/miners, who are paid through fees, certificate the correctness of all the process steps: for instance, they ensure that both firms have signed smart contracts and check that buyers can afford the payment.

In conclusion, such a system does not only make the transaction process more efficient, but it changes most of the player’s roles. Auditors and accountants will focus on consultancy and item evaluation rather than mechanical processes, while new figures will emerge as smart contract coders and lawyers, miners, and validators (Deloitte, 2017:12; Erica Pimentel et al., 2020: 344).

After a complete overview of this system, the next paragraph’s purpose consists of code prototype analysis.

## Prototype Analysis

The following sections provide a comparative analysis of the stylized system illustrated above and present a Python code set to implement this system, developed as part of the thesis work.

To develop the case study, I made several simplifying assumptions. First, the model presumes that companies carry out all transactions in such system, and cryptocurrency payments are immediately made.

Furthermore, the code replaces blockchain ledgers with database tables due to complexity reasons. Consequentially, it does not run on a blockchain, and a central user (i.e., who runs the code) has the power to control all data. In contrast, in a “real” system, validators/miners should have that specific task in order to guarantee data immutability and safety. It is also obvious that there is no consensus mechanism since the project is a single-user system.

Also, some features are partially or not automatized: blockchain and artificial intelligence combined should make the process leaner, while this project requires massive human intervention. Finally, the code deals with just a few anomalous situations.

## Database[[21]](#footnote-21)

Before the analysis of the code, the nine[[22]](#footnote-22) tables of the database are explained. A buyer’s point of view was chosen. All companies have access to their tables which substitute blockchain ledgers, but instead of having many blocks which contain a fixed amount of data, they contain all of them. As an ordinary ledger, tables contain headers and data related to those headers. As a public blockchain, this kind of table does not contain data related to a specific company but all data of the blockchain participants. Unlike this project, in a blockchain system, having access to a database does not mean having access to all data contained. For example, the “invoice” table, which contains all invoices, shows only transactions concerning the company. It would be redundant to implement a personal ledger linked to the general ledger.

### *Invoice*

As shown in Figure 4, this table gathers all information collected in current invoices, such as VAT, amount, description of the goods, and date. The “InvoiceID” voice is fundamental because it belongs to most of the other tables linking them to each other. If the TEA software presupposes smart contracts, invoices can be issued by the system itself; in the case considered, the user must fill out all data except for the date, which is automatically determined. Both parties, their auditors, and the government have access to it.

Immagine che contiene tavolo

Descrizione generata automaticamente

Figure 4: Invoice Database

### *Blockchain phases*

This table shows all transaction steps. Once a phase is completed, the related voice is timestamped. The company is able to follow all procedures, from the invoice issuing to the final reconciliation. Only parties involved have access to it.

### Account balance

It can be compared to a bank statement. Although it is not required, it would be useful to be linked to an invoice; for instance, a stockholder's deposit does not issue any invoice. Consequently, the "TransactionID" voice has been created, which could be slightly different from the "InvoiceID". A smart contract system is supposed to be able to automatize invoice payments fully. Being private, only the buyer can have access to it.

### *Participants*

It contains all actors[[23]](#footnote-23) to the network. It is the only table which data are available to all participants. The only data constantly updated is “score”: automatically calculated by the system, it indicates the party’s trustworthiness.

### *VAT*

It stores all details of indirect taxes for every invoice, including the total amount. If the system allows smart contracts, taxes could be automatically paid for every transaction. The buyer, the government, and the auditors should be able to see the data inside this table.

### *Public ledger*

It is the core part of the TEA system. Figure 5 shows that it contains the linked invoiceID, data, debit, credit, and amount. It differs from a current business journal because credits gather Pacioli’s credits of the seller while debits gather Pacioli’s debits of the buyer. Both parties and their auditors are supposed to see it. Furthermore, companies should confirm every transaction.

Immagine che contiene tavolo

Descrizione generata automaticamente

Figure 5: Private ledger database

### *Private ledger*

As the name suggests, it is the buyer’s private ledger. It seems redundant to have it, but it is fundamental to implement financial statements. It is linked directly to the “public ledger” table, and such link is supposed to be “unbreakable”. Artificial intelligence could help to fill up the opposite Pacioli’s entry.

### *Asset IDs*

It consists of a table that tracks all company assets. Consequently, it is not involved in all transactions. In the case that governments use blockchain as the land register, it would be easier to update this ledger. As it happens nowadays, access restrictions depend on the kind of asset.

### *Balance sheet details*

It contains all details about an asset. It updates real-time to calculate asset value based on annual amortization, bought date, and today’s date. It also allows companies to reevaluate it or depreciate the asset itself. Companies and their auditors are supposed to have access to it.

## Function[[24]](#footnote-24)

The second part of the paragraph deals with the code. All functions[[25]](#footnote-25), which are a bunch of code that does a specific task, are analyzed: firstly describing how it works in this project, then highlighting differences between code and a real TEA system[[26]](#footnote-26).

### *Insert table data*

As the name suggests, its task consists of inserting data into any table[[27]](#footnote-27). Data are either asked directly to the user or filled out automatically by the system. When the user fills up an invoice, the system creates a data row in the “Blockchain phases” table and inserts the public signature and the invoice ID. The next functions work only in the case that an invoice is issued by two parties who belong to the “Participants” table.

While in the code most of the parts are filled out by the user, in a TEA system, participants should be able to insert their data for themselves. For example, they should sign up through a website, and their data should be recorded in the “Participants” table. In contrast, other ledgers, such as the “Account balance” or the “Blockchain phases”, should not be modifiable by participants. Furthermore, a smart-contract integration would fully automate the "Invoice” table or the "VAT" table.

### *Buyer’s bank account*

The second function compares the buyer’s bank balance and the total invoice amount. If the first is greater, the system records a transaction equal to the second and holds the money in the “Blockchain phases” ledger. Indeed, this kind of transaction presupposes immediate payment, which can be considered a simplification of most current business-to-business transactions. Also, the system is able to fill out all the table fields autonomously. In the case that buyer’s balance is insufficient, code outputs: "Buyer's account has not enough money".

This function task is close to a blockchain-based function: the latter involves a smart contract that holds the money until the transaction is over to grant both parties’ security. The same smart contract will be used for the next functions.

### *Seller’s shipment*

This short function is supposed to ask the seller[[28]](#footnote-28) to notice once they sent goods[[29]](#footnote-29). An affirmative answer is registered in the “Blockchain phases” ledger linked to the invoice.

The same smart contract carries out this task as above. Indeed once the seller has been notified that money is held on the blockchain, they should send goods and update it.

In the case that a transport company is involved, it could update the ledger alone.

### *Buyer’s confirmation*

Its goal is simply to verify that goods are arrived by asking the buyer. As the function above, an affirmative answer is registered on the “Blockchain phases” ledger.

To avoid any inconvenience, the transportation company should accomplish this task as well; for the same reason, if the transaction object is a service, both parties should confirm that the service has been provided; in the case that an asset is involved, the notary or the blockchain itself[[30]](#footnote-30) should update both functions.

### *Update score*

System asks again to confirm that no problems have been raised so far (e.g., goods are not damaged). After an affirmative answer, it updates both parties’ scores by adding one to them as the code below shows; in the opposite scenario, scores would fall by 10 points. As highlighted in the first line of the code, the “Update score” function asks the user parties’ VAT number as a parameter instead of invoice ID.

Parties’ score indicates the reliability of a company.

This function presents many simplifications. First, all anomalies are processed the same way: for instance, if the buyer declares that goods are damaged, both scores drop automatically. In reality, parties probably try to reach an agreement. Secondly, artificial intelligence linked to the system could be used to detect any damage in order to resolve the case considered. Furthermore, a more complex algorithm should be developed to determine parties’ scores considering many factors. Finally, considering what happened, whose fault it is, and how the transaction concluded is crucial to state the number of points to subtract from the score.

def update\_score(seller\_vat, buyer\_vat):  
 old\_score = cur.execute("SELECT \* FROM Participants")  
 rows = old\_score.fetchall()  
 # Re-asking parties if everything went as expected  
 response = input('Do you both confirm that everything went good so far? (Y/N) ')  
 if response == 'Y':  
 # if so, adding 1 point to both parties  
 for row in rows:  
 if row[0] == seller\_vat:  
 new\_seller\_score = row[3] + 1  
 elif row[0] == buyer\_vat:  
 new\_buyer\_score = row[3] + 1  
 else:  
 pass  
 else:  
 # if not, differencing 10 points  
 for row in rows:  
 if row[0] == seller\_vat:  
 new\_seller\_score = row[3] - 10  
 elif row[0] == buyer\_vat:  
 new\_buyer\_score = row[3] - 10  
 else:  
 return "party doesn't exist"  
  
 # updating buyer & seller's score  
 update\_buyer\_score = "UPDATE Participants SET Score = " + str(new\_buyer\_score) \  
 + " WHERE VATNumber = " + str(buyer\_vat)  
 cur.execute(update\_buyer\_score)  
 conn.commit()  
 update\_seller\_score = "UPDATE Participants SET Score = " + str(new\_seller\_score) \  
 + " WHERE VATNumber = " + str(seller\_vat)  
 cur.execute(update\_seller\_score)  
 conn.commit()  
 return 'Score has been updated successfully'

### *Paying account transaction*

At this point, the system detects the total indirect tax[[31]](#footnote-31), by reading it on the invoice, and pays it. Also, it records on the “VAT” ledger all information required: the transaction ID, the percentage of VAT, the amount, and the date. Finally, the system makes money available to the seller’s account[[32]](#footnote-32).

The smart contract ends with paying the seller and taxes. In order to automatize all this process, cryptocurrencies are required, and government should be an active participant in the system as well.

### *Public ledger*

As the name suggests, this function aims to fill out data on the public ledger as a current journal. The buyer’s credit and the seller’s debit are typed in by the user while the system figures out the opposite entry, the date, the amount, and the invoice ID. The user also responds if an asset is involved; in such case, the latter is registered on “Asset

ID” ledger as shown below.

Most current systems are more efficient than the one described: an artificial intelligence spoils the invoice to fill entries automatically. They could be able to understand if an asset is involved. A real TEA system should use the same methods as current software to implement the public ledger. This function can be regarded as the core of a TEA system; therefore the public ledger is supposed to be viewable and confirmed by the three parties.

def public\_ledger(invoice\_id):  
 # asking seller credit and buyer debit (it is supposed to use budget items which can be found in the financial statements)  
 buyer\_debit = str(input("What is the buyer's debit? "))  
 assets = str(input('Is it an asset?(Y/N) '))  
 cur.execute('SELECT COUNT(\*) from Asset\_IDs')  
 cur\_result = cur.fetchone()  
 asset\_id = cur\_result[0] + 1  
 if assets == 'Y':  
 items = [asset\_id, buyer\_debit, invoice\_id]  
 sql = '''INSERT INTO Asset\_IDs(Asset\_ID, Name, Invoice\_ID) VALUES(?,?,?)'''  
 cur.execute(sql, items)  
 conn.commit()  
 else:  
 pass  
 seller\_credit = str(input("What is the seller's credit? "))  
 time = str(datetime.today())  
 invoice\_cost = cur.execute("SELECT \* FROM Invoice WHERE InvoiceID =?", str(invoice\_id))  
 row = invoice\_cost.fetchall()  
 total\_amount = row[0][4]  
 # recording all items needed  
 items = [invoice\_id, time, buyer\_debit, seller\_credit, total\_amount]  
 sql = '''INSERT INTO Public\_ledger(InvoiceID, Date, Buyer\_debit, Seller\_credit, Amount) VALUES(?,?,?,?,?)'''  
 cur.execute(sql, items)  
 conn.commit()  
 return 'Credit/Debits successfully recorded to Public Ledger'

### *Buyer’s ledger*

This function could become unnecessary in the future. Indeed, it simply links public and private ledger: the first entry equals public ledger debit, while the user writes down the second. Finally, the system fills out the remaining fields of the ledger.

A private ledger is useful to automatically update financial statements (both entries are automatically taken from it), but it can be considered redundant. Also, as with most of the current blockchains, validators/miners are rewarded through fees[[33]](#footnote-33) which means that its implementation has a cost.

### *Update first details*

This function and the next concern only assets: trying to pass a non-asset object raises an error because that object cannot be found in the “Asset ID” ledger. This function is supposed to run only once an asset is purchased. It aims to store in the “Balance sheet detail” ledger the annual amortization and a possible depreciation/revaluation of the asset.

The user has to fill out both information, while the current value is automatically calculated as follows:

Paid Value ± Revaluation – Annual Amortization \* (Last Update Date – Purchase Date)

In a blockchain-based system, a ledger that contains all registered assets might be created. In such a scenario, a business-to-business purchase could be partially automatized: annual amortization already exists[[34]](#footnote-34). As regards depreciation, it is much more complicated to automatize, artificial intelligence could be useful, but auditors are supposed to calculate it. Consequently, blockchain will never be able to replace auditors, it will just change their tasks. This function represents another revolution: from annual financial statements to real-time reports.

### *Update details*

This function does not require any input[[35]](#footnote-35), system automatically updates all assets found in the ledger. As the code below shows, it calculates updated amortization to evaluate the current value. As said previously, real-time reports are one of the main benefits of switching to a blockchain system.

def update\_details():  
 date = datetime.today()  
 transactions = cur.execute("SELECT \* FROM Balance\_sheet\_details")  
 rows = transactions.fetchall()  
 for row in rows:  
 bought\_date = row[1]  
 annual\_amortisation = row[4]  
 datetime\_obj = datetime.strptime(bought\_date, '%Y-%m-%d %H:%M:%S')  
 total\_time = datetime\_obj - date  
 # Count how many days are passed from the last update  
 days = abs(int(str(total\_time).split()[0]))  
 tot\_amortisation = annual\_amortisation \* (days / 365)  
 final\_value = row[5] - tot\_amortisation  
 updated\_data = '''UPDATE Balance\_sheet\_details SET Updated\_date = ''' + str(date) + \  
 ''', Total\_current\_amortisation = ''' + tot\_amortisation + ''', Final\_value = ''' + final\_value \  
 + ''' WHERE AssetID = ''' + str(row[0])  
 cur.execute(updated\_data)  
 conn.commit()  
 return 'details updated'

### *Export Excel balance sheet*

Most of the current accounting methods involve Excel to draw up financial statements. In order to be closer to current methods, this function aims to let the user to get an Excel pre-compiled financial statement. In reality, it makes a copy of the “Buyer’s balance sheet”[[36]](#footnote-36) and lets the user choose its name. The latter is a real Italian financial statement. The same Excel sheet is supposed to be used all fiscal year, while the next function keeps it updated.

The software is more likely linked to the blockchain in a real TEA system, drawing up financial statements rather than working on Excel sheets.

### *Update Excel sheet*

This function is divided into two main sections. The first one aims to update the purchase transaction, while the second updates asset amortizations. It requires the user just to type in the Excel sheet chosen, while balance sheet items are automatically updated: asset names and private ledger entries must coincide with items found in the Excel sheet.

# *CONCLUSION*

The aim of this study is to explore how blockchain could affect the accounting and auditing profession by examining the potential use of blockchain technology in business-to-business purchasing transactions. Blockchain as a technology has the potential to disrupt business deals (Erica Pimentel et al., 2020: 326). Its immutability and timestamping allow disintermediation, a transparency increase, and real-time auditing. Furthermore, this technology enabled Triple-entry accounting to emerge (C. W. Cai, 2019, 71). The latter has the potential to modify a cornerstone of classic accounting: double-entry accounting. This paper analyses what it is, what changes from the current system, and additional features which add value to the system. This system could automatize most of the current time-consuming and costly activities auditors carry out nowadays (C. W. Cai, 2019, 72). This study describes the basics of blockchain technology and the most important steps of current purchasing transactions and develops a practical case study of a realistic blockchain-based accounting transaction using Python. The proposed stylized project contains many simplifications, such as the lack of the blockchain-based system itself. Nevertheless, this publication helps both auditors and developers to analyze both sides. Still, more research needs to be conducted by both subjects in order to develop prototypes or beta version projects which focus on different aspects of this area (Erica Pimentel et al., 2020: 326).

Although this prototype focuses on business-to-business transactions, blockchain has the potential to affect most accountants’ and auditors’ roles. Indeed, N. Elommal et R. Manita (2022: 37, 39) state that TEA systems could facilitate companies from many points of view:

* saving time and improving the efficiency of their audit spoiling transparency and real-time auditing;
* bank automatic reconciliation favors an audit covering the whole population instead of an audit based on sampling techniques;
* focusing auditors on testing controls rather than testing transactions, especially if cryptocurrencies are involved;
* setting up a continuous audit process; and
* letting auditors play a more strategic role and develop new advisory services. For instance, they could verify the existence of digital assets and attest to the consistency between information in the physical world and on the blockchain.

Consequently, blockchain will revolutionize rather than delay auditors’ work. It indeed will allow auditors to focus on tasks at a higher level; for instance, they could verify the existence of digital assets and attest to the consistency between information in the physical world and on the blockchain (Erica Pimentel et al., 2020: 340; J. Schmitz and G. Leoni, 2019: 331; N. Elommal et R. Manita, 2022: 44).

In addition, auditing bases its principles on the law, which means governments play a fundamental role in spreading blockchain in this field. Indeed, as with all technologies which involve the public sector, the first technology “push” is usually done by private companies; however, governments must implement it to promote its spread (EY, 2016). Consequently, this paper intends to incentivize companies and auditors to invest in real blockchain projects in this area.

# *REFERENCES*

Cai C. W., 2019. Triple-entry accounting with blockchain: How far have we come? Available at: <https://doi.org/10.1111/acfi.12556>

CPA and AICPA 2017, Blockchain Technology and Its Potential Impact on the Audit and Assurance Profession. Available at:   
[https://www.cpacanada.ca/en/business-and-accounting-resources/audit-and-assurance/canadian-auditing-standards-cas /publications/impact-of-blockchain-on-audit](https://www.cpacanada.ca/en/business-and-accounting-resources/audit-and-assurance/canadian-auditing-standards-cas%20/publications/impact-of-blockchain-on-audit)

Crosby M. et al., 2015.​ BlockChain Technology. Available at:   
<https://scet.berkeley.edu/wp-content/uploads/BlockchainPaper.pdf>  
Cryptopedia, 2022. How a Block in the Bitcoin Blockchain Works. Available at:  
<https://www.gemini.com/cryptopedia/what-is-block-in-blockchain-bitcoin-block-size>

Deloitte, 2016a. Blockchain technology: A game-changer in accounting? Available at: [https://www2.deloitte.com/content/dam/Deloitte/de/Documents/Innovation/Blockchain\_A%20game-changer%20in% 20accounting.pdf](https://www2.deloitte.com/content/dam/Deloitte/de/Documents/Innovation/Blockchain_A%20game-changer%20in%25%2020accounting.pdf)

Deloitte, 2016b. Blockchain: Enigma, Paradox, Opportunity. Available at: <https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/Innovation/deloitte-uk-blockchain-full-report.pdf>  
Drescher D., 2017. Blockchain Basics: A Non-Technical Introduction in 25 Steps. Available at: <https://link.springer.com/book/10.1007/978-1-4842-2604-9>  
Elommal N. et al., 2022. How Blockchain Innovation could affect the Audit Profession: A Qualitative Study. Available at:   
<https://www.cairn.info/revue-journal-of-innovation-economics-2022-1-page-37.htm>

Ethereum official website. Consensus mechanism. Available at:  
<https://ethereum.org/en/developers/docs/consensus-mechanisms/>  
EU official documentation. What is VAT. Available at:   
<https://ec.europa.eu/taxation_customs/what-vat_en>

EY 2016, ‘Building Blocks of the Future’. Available at: <https://www.ey.com/Publication/vwLUAssets/EY-building-blocks-of-the-future/%24FILE/EY-building-blocks-of-the-future.pdf>  
Ibañez J. I. et al.., 2021. Triple-entry Accounting, Blockchain and Next of Kin: towards a Standardization of Ledger Terminology. Available at: <https://doi.org/10.2139/ssrn.3760220>

Italian Government NADEF, 2021. Available at:  
<https://www.dt.mef.gov.it/export/sites/sitodt/modules/documenti_it/analisi_progammazione/documenti_programmatici/nadef_2021/NADEF_2021.pdf>  
Keen M. et al., 2007. VAT Fraud and Evasion: What Do We Know, and What Can be Done? Available at:

<https://www.imf.org/en/Publications/WP/Issues/2016/12/31/VAT-Fraud-and-Evasion-What-Do-We-Know-and-What-Can-be-Done-20215>  
Nguyen V.-C- et al., 2019. Digitizing Invoice and Managing VAT Payment Using Blockchain Smart Contract. Available at:

<https://www.researchgate.net/publication/334167437_Digitizing_Invoice_and_Managing_VAT_Payment_Using_Blockchain_Smart_Contract>

Pimentel E. et al.. 2020. Systemizing the challenges of auditing blockchain-based assets. Available at: [https://ssrn.com/ abstract=3359985](https://ssrn.com/%20abstract=3359985)

Pinto [R](https://www.forbes.com/sites/forbestechcouncil/people/rohanpinto1/)., 2019. A Blockchain-Based Digital Notary: What You Need To Know. Available at: <https://www.forbes.com/sites/forbestechcouncil/2019/11/12/a-blockchain-based-digital-notary-what-you-need-to-know/?sh=3f3b397e4557>

Schmitz J. and Leoni G., 2019. Accounting and auditing at the time of blockchain technology: a research agenda. Available at: [https://doi.org/ 10.1111/auar.12286](https://doi.org/%2010.1111/auar.12286)

[Vermaak](https://coinmarketcap.com/alexandria/author/werner-vermaak) W., 2021. What Are Application Layer Protocols? Available at: <https://coinmarketcap.com/alexandria/article/what-are-application-layer-protocols>

Wang K. et al., 2021. A Conceptual Model for Blockchain-based Auditing Information System. Available at: [https://doi.org/10.1145/3409934.3409949](https://doi.org/10.1145/3409934.3409949%20%20)

Wijk C. V., 2019. How to keep data private on a blockchain. Available at:

<https://medium.com/ing-blog/how-to-keep-data-private-on-a-blockchain-252848ae228e>

Zulauf L. et al., 2002. Enron: The Good, The Bad, The Lessons. Available at:

<https://doi.org/10.19030/iber.v1i11.3998>

# *APPENDIX*

**Code**

Link to the code, the Excel sheet, and the database file: https://github.com/jacopo-urbani/B2B-blockchain-deals-Triple-entry-accounting

**Libraries**

A Python library contains bundles of code that can be used in different programs. In other words, downloading and using a library allows one to call a function that carries out a specific task. The advantages consist of saving time and using functions hard to create. For example, the “random” package contains the “randrange” function, which outputs a number between a range: implementing it would require coding several rows while calling it requires just one row.

Packages are so popular because they make Python Programming simpler and more convenient for programmers[[37]](#footnote-37)

**SQLite3**

It is one of the most famous database libraries. The user can interact with the database app, which is user-friendly, and it allows the creation of tables and interaction with them. Python is used when more complex operations, such as extraction, of data are executed. In this program, blockchain tables are replaced by SQLite tables.

To find out more, see official documentation: <https://www.sqlite.org/index.html>

**Datetime**

This library is a basic Python library that “supplies classes to work with date and time”.  
It is useful because it is able to timestamp data, and date operations are simpler.

To find out more, see: <https://docs.python.org/3/library/datetime.html>

**Openpyxl**

This library allows interaction with Excel. Most ERP databases can extract data onto Excel sheets, so it could be useful to implement the same feature in a blockchain-based system. Also, this module updates financial statements in real-time on Excel.

To see more: <https://openpyxl.readthedocs.io/en/stable/>

**Random**

It is one of the most famous Python modules. As the name suggests, it generates random numbers or items. In this project, the random function is used to generate blockchain signatures which are used mainly to authenticate blocks.

To see more: <https://docs.python.org/3/library/random.html>

**Shutil**

The “Shutil” module provides many functions of high-level operations on files and collections of files. In this project, it is used just to generate a pre-compiled financial statement.

To see more: <https://docs.python.org/3/library/shutil.html>

**Supporting function**

Since libraries are just a bundle of code, users can either download them from the internet or create theirs. Indeed, this library was created to contain only one function since it is repetitively called (i.e., used). This function aims to update a financial statement item on an Excel sheet containing the balance sheet and income statement.

1. Although the code is simple and will be thoroughly analyzed, a basic knowledge of Python or any other programming language allows one to understand the paper better. [↑](#footnote-ref-1)
2. For further information: “Blockchain Basics: A Non-Technical Introduction in 25 Steps”, Daniel Drescher, 2017 [↑](#footnote-ref-2)
3. It works like a vending machine. By inserting a coin, people implicitly conclude an agreement that is self-executed. [↑](#footnote-ref-3)
4. All the paragraph information is based on: <https://www.gemini.com/cryptopedia/what-is-block-in-blockchain-bitcoin-block-size> [↑](#footnote-ref-4)
5. It is a function of one-to-many. A single hash is related to many kinds of data, but finding two data with the same hash is computationally unfeasible. [↑](#footnote-ref-5)
6. They are a special kind of network participants whose task consists of validating blocks (i.e., verifying that transactions are legal). Miners are proof-of-work (PoW) participants, while validators participate in proof-of-stake (PoS) blockchains. PoW and PoS are two of the most famous consensus mechanisms. For further information: <https://ethereum.org/en/developers/docs/consensus-mechanisms/> [↑](#footnote-ref-6)
7. It is similar to a report of the year budget law; for further information, visit: [https://www.dt.mef.gov.it/export/sites/sitodt/modules/documenti\_it/analisi\_progammazione/documenti\_programmatici/nadef\_2021/NADEF\_2021.pdf](https://www.dt.mef.gov.it/export/sites/sitodt/modules/documenti_it/analisi_progammazione/documenti_programmatici/nadef_2021/NADEF_2021.pdf%20) [↑](#footnote-ref-7)
8. [↑](#footnote-ref-8)
9. Extra costs could exceed company damage. [↑](#footnote-ref-9)
10. Enron is the most famous case regarding manipulation of financial statements. [↑](#footnote-ref-10)
11. There are many kinds of indirect taxes. For example, the USA uses a different type called “Retail tax”, which is still exposed to the same problems.  
    To know more about VAT: <https://ec.europa.eu/taxation_customs/what-vat_en> [↑](#footnote-ref-11)
12. In Italy, it is estimated that VAT evasion is up to 21,3% (2019). [↑](#footnote-ref-12)
13. Note that since no system can be regarded to be absolutely immutable, the word ‘immutability’ should be applied within reasonable bounds to systems that are significantly harder to modify than their competitors. [↑](#footnote-ref-13)
14. The consensus mechanism rewards miners and validators in order to incentivize them to participate. Being honest is crucial to holding a crypto value, which implicitly raises rewards. [↑](#footnote-ref-14)
15. Most of the features come from “Triple-entry accounting, blockchain and next of kin: Towards a standardization of ledger terminology” (Ibañez et al., 2021). [↑](#footnote-ref-15)
16. As “distant concept” is meant a feature that is not close to any current method (e.g., Smart Contracts). [↑](#footnote-ref-16)
17. to know more: <https://medium.com/ing-blog/how-to-keep-data-private-on-a-blockchain-252848ae228e> [↑](#footnote-ref-17)
18. An application layer is an abstraction level that masks the nitty-gritty technical details of a communication channel and serves as a user interface on a network. For instance, HTTP is a primary layer that only developers understand, whereas web pages are the application layers that the user can easily interact with.  
    To learn more: https://coinmarketcap.com/alexandria/article/what-are-application-layer-protocols [↑](#footnote-ref-18)
19. They could be useless in that specific context, not in all scenarios. [↑](#footnote-ref-19)
20. As described in the “blockchain” section, smart contracts automatically carry out a specific action when a specific event occurs. [↑](#footnote-ref-20)
21. It can be found on Github. The link is on the “appendix” section. [↑](#footnote-ref-21)
22. “sqlite\_sequence” is a default table that is not to be considered. [↑](#footnote-ref-22)
23. By actor, it is meant not only nodes but also all subjects interacting with the network. For instance, in the Bitcoin blockchain, an actor could be either a miner or whoever buys/sells that cryptocurrency. [↑](#footnote-ref-23)
24. As well as for the database, the code can be found on Github. The link is in the “appendix” section. [↑](#footnote-ref-24)
25. Some function uses libraries: to know more about what a library is and its features, consult the “appendix” section. [↑](#footnote-ref-25)
26. As a real TEA system, it is meant the definitive version of this code, which should run on blockchain. [↑](#footnote-ref-26)
27. The user writes the choice of the table when it calls the function. [↑](#footnote-ref-27)
28. User answers instead of them. [↑](#footnote-ref-28)
29. This function applies to services and assets as well. [↑](#footnote-ref-29)
30. A short Forbes article explains why blockchain could revolutionize the land registry system by replacing notaries: https://www.forbes.com/sites/forbestechcouncil/2019/11/12/a-blockchain-based-digital-notary-what-you-need-to-know/?sh=3f3b397e4557 [↑](#footnote-ref-30)
31. VAT is used as an indirect tax. [↑](#footnote-ref-31)
32. Since the database shows only “buyer’s tables”, the code is limited to taking out money from the buyer’s bank. [↑](#footnote-ref-32)
33. In some cases, they can be very low, but they still exist. [↑](#footnote-ref-33)
34. Because the previous owner was a company that was already amortizing it. [↑](#footnote-ref-34)
35. As specified at the beginning, most functions require invoice ID to work properly. [↑](#footnote-ref-35)
36. The Excel sheet can be found on Github. The link is in the “appendix” section. [↑](#footnote-ref-36)
37. libraries are used in all modern programming languages [↑](#footnote-ref-37)