

RWTH Aachen University
Fraunhofer FIT

Master's Thesis in Software Systems Engineering

**Tracing Carbon Footprint Across Automobile Supply
Chain And Payment Of Carbon Price Using
Blockchain**

Author:	Samuel Roy
Supervisor and Examiner:	Prof. Dr. Wolfgang Prinz
Co- Examiner:	Prof. Dr. Thomas Rose

August 18, 2021

Eidesstattliche Versicherung Statutory Declaration in Lieu of an Oath

Roy, Samuel
Name, Vorname/Last Name, First Name

391822
Matrikelnummer (freiwillige Angabe)
Matriculation No. (optional)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende ~~Arbeit/Bachelorarbeit/~~
Masterarbeit* mit dem Titel

I hereby declare in lieu of an oath that I have completed the present paper/Bachelor thesis/Master thesis* entitled

Tracing Carbon Footprint Across Automobile Supply Chain and Payment of
Carbon Price using Blockchain.

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

independently and without illegitimate assistance from third parties (such as academic ghostwriters). I have used no other than the specified sources and aids. In case that the thesis is additionally submitted in an electronic format, I declare that the written and electronic versions are fully identical. The thesis has not been submitted to any examination body in this, or similar, form.

Aachen, August 18, 2021
Ort, Datum/City, Date

Unterschrift/Signature
*Nichtzutreffendes bitte streichen
*Please delete as appropriate

Belehrung: Official Notification:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

Para. 156 StGB (German Criminal Code): False Statutory Declarations

Whoever before a public authority competent to administer statutory declarations falsely makes such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment not exceeding three years or a fine.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Strafflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Para. 161 StGB (German Criminal Code): False Statutory Declarations Due to Negligence

(1) If a person commits one of the offences listed in sections 154 through 156 negligently the penalty shall be imprisonment not exceeding one year or a fine.

(2) The offender shall be exempt from liability if he or she corrects their false testimony in time. The provisions of section 158 (2) and (3) shall apply accordingly.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

I have read and understood the above official notification:

Aachen, August 18, 2021
Ort, Datum/City, Date

Unterschrift/Signature

Acknowledgements

At the outset, let me express my deep sense of gratitude to my supervisor, Prof. Dr. Wolfgang Prinz, for identifying this interesting thesis topic and providing me with the opportunity to work under his guidance. I thank him for his time for review meetings, fruitful advice, discussions, patience, and encouragement from time to time during the entire course of my thesis work. His motivation was the driving force behind the completion of this Master thesis.

I would also like to thank my second supervisor, Prof. Dr. Thomas Rose, for his time, suggestions, and encouragement during meetings and reviews.

My thanks are due to my loving parents for their unwavering support as I pursued my studies in Germany.

I also offer my gratitude to God almighty for all His blessings, especially in my difficult times.

Abstract

Climate change is one of the most pressing issues of the twenty-first century, and it is indisputable that human activities drive climate change. Among the various economic sectors, industries such as the automotive industry contribute significantly to greenhouse gas emissions. The automotive supply chain is a complex multi-echelon supply chain with multiple supply chain participants spread globally, all contributing to the different phases of the production of cars and contributing to the greenhouse gas emissions. Automotive supply chains face many challenges that prevent them from effectively managing their carbon emissions and developing a mitigation strategy. The absence of data about the carbon footprint of their products is the most significant of the challenges.

One of the approaches to collecting the carbon emission data is to reliably track the carbon footprint of the product across the automotive supply chain. The main objective of this thesis is to investigate the applicability of blockchain to develop a decentralized carbon emissions tracking system for the automotive supply chain. In addition to this, the thesis proposes a conceptual design to unify the carbon emissions tracking along with the carbon tax payments using carbon credits, which are modeled by tokens on the blockchain network. Furthermore, the thesis proposes using a carbon label with a QR code attached to the product to help in carbon emissions tracking across the supply chain, with the QR code being generated using a self-sovereign digital identity that can be validated on the blockchain. Moreover, a functional prototype of the proposed solution is implemented as a Dapp. Further, evaluations were also performed to validate the scalability and feasibility of the system.

Keywords: Blockchain, Automotive Supply Chain, Smart Contracts, Carbon Emissions Traceability, Carbon Tax, DID, Ethereum.

Contents

Acknowledgement	i
Abstract	iii
1 Introduction	1
1.1 Motivation	1
1.2 Research Purpose and Questions	4
1.3 Thesis Outline	5
2 Related Work	6
2.1 Background	6
2.1.1 Automotive supply chain	6
2.1.2 Distributed Ledger Technology	6
2.1.3 Blockchain	7
2.1.4 Cryptography	9
2.1.5 Addresses	10
2.1.6 Peer-to-Peer Systems	10
2.1.7 Consensus Mechanisms	10
2.1.8 Characteristics of blockchain	11
2.1.9 Types of permissions in blockchains	11
2.1.10 Smart Contracts	12
2.1.11 Tokens	13
2.1.12 Decentralized Identity	13
2.2 Literature Review	14
2.2.1 Blockchain as a Tool for Carbon Emission Management (Liu et al., 2019)	15
2.2.2 Using Blockchain Technology to Re-engineer the Carbon Supply Chain(Banerjee, 2018)	15
2.2.3 Blockchain-based Traceability of Carbon Footprint(Rosado da Cruz et al., 2020)	17
2.2.4 Carbon Labelling- A system for calculating product carbon footprint using Blockchain(Pellen-Pickersgill et al., 2019)	17
2.2.5 A Decentralized Traceability Application for Multi-Tier Automotive Supply Chain Networks (PartChain) (Miehle et al., 2019)	18
2.2.6 Carbon Credits on Blockchain (Patel et al., 2020)	19

2.2.7	Integrating Carbon Footprint into Supply Chain Management for the Automobile Industry (Lee, 2011)	20
2.3	Summary	20
3	Concept Description	22
3.1	Case Background and Definitions	22
3.1.1	Carbon footprint	22
3.1.2	Methodologies of Carbon Footprint Tracing in Supply Chain	22
3.1.3	Carbon Pricing Policies	25
3.1.4	Carbon Labelling	25
3.1.5	Assumptions in the Supply Chain considered for the study	26
3.1.6	Application Scenerio	27
3.2	Proposed System Overview	27
3.2.1	System Objectives	27
3.2.2	System Actors	28
3.2.3	Top-Level Design of the System	28
3.3	Requirements of the System	29
3.4	Use Cases	31
3.5	System Architecture	35
3.5.1	Logical view	35
3.5.2	Process view	36
3.5.3	Development view	37
3.5.4	Physical view	39
3.5.5	Scenerio view	39
3.6	Summary	40
4	Implementation	41
4.1	Implementation Decisions	41
4.1.1	Blockchain	41
4.1.2	Back-end	44
4.1.3	Front-end	44
4.2	Implementation Concepts	44
4.2.1	Ethereum Blockchain	44
4.2.2	Decentralized Application	47
4.2.3	Development Tools	48
4.2.4	Languages	48
4.2.5	Blockchain	49
4.2.6	User Interface	50
4.3	Implementation Details	50
4.3.1	System Actors	50
4.3.2	UML Diagram - Class Diagrams of the Smart Contracts	50
4.3.3	Smart Contract Implementation	52
4.3.4	Architecture of the Implemented System	54

4.4	User Interface	55
4.4.1	User Interfaces Related to Managers	55
4.4.2	User Interfaces for Suppliers	58
4.4.3	User Interfaces for OEMs	61
4.5	Summary	65
5	Evaluation	66
5.1	Theoretical Evaluation	66
5.1.1	Fulfilment of Requirements	66
5.1.2	Limitations of the Implemented System	67
5.2	Implemented Prototype Evaluation	68
5.2.1	Software Testing	68
5.2.2	Scalability Evaluation	69
5.2.3	Quantitative Evaluation	72
5.3	Conclusion	74
6	Conclusion	75
6.1	Summary	75
6.2	Fulfilment of Objectives of the Thesis	76
6.3	Answers to Research Questions	77
6.4	Future work	78
6.4.1	Improvements	78
6.4.2	Possible Future Works	78

1 Introduction

1.1 Motivation

Green House Gas (GHG) emissions originating from human activities are identified as the primary source of climate change, and they are major concerns that must be addressed internationally. Carbon dioxide (CO_2), water vapour (H_2O), methane (CH_4), ozone (O_3), nitrogen oxides (NO_x), hydrofluorocarbon (HFCs), perfluorocarbons (PFCs), Sulphur Hexafluoride (SF_6) and Nitrogen trifluoride($(\text{NF}_3)^3$) are the most frequent GHGs. The Kyoto Protocol, adopted in 1997, was a landmark international treaty that legally mandated country-specific GHG emission reduction target(Oliver, 2005). The United Nations (UN) introduced the Paris Agreement in 2015, marking a significant milestone in establishing a mechanism for countries to monitor, control, and set a GHG emission standard. Signatories to the Paris Agreement have agreed to reduce GHG emissions over time through an economic and social transformation in order to limit the rise of global average temperatures to less than 2 degrees Celsius above pre-industrial levels(Paris Agreement, 2015).

Two widely agreed-upon and efficient approaches that aim to achieve these emission reduction targets are based on a cap-and-trade or a carbon tax system. The cap-and-trade system (also called carbon emission trading) establishes an overall cap on the permitted emissions from a company by issuing a carbon emission allowance (also known as carbon credits). The trade policy in cap-and-trade enables companies to buy or sell carbon emission allowances based on their demand or surplus in the carbon market. The carbon tax is a scheme in which the emitter is held liable for the GHGs emitted, which harms the environment. The tax also encourages and incentivizes carbon reduction efforts. The carbon tax method decreases emissions by giving companies the option of staying within their allowed emission limits by using more efficient processes and less-polluting energy sources or paying for emissions that exceed their allowed emission limits.

Awareness that environmental policies and regulations are becoming increasingly stringent, leading to more financial risk over time, prompts stakeholders and shareholders of Original Equipment Manufacturer (OEM) companies in various sectors and industries to take greater responsibility for the carbon footprint of their products. As a result of the financial implications and the good reputation of their products with a lower carbon footprint, companies are focusing more efforts on reducing carbon emissions. Incorporating environmental criteria into supply chain management (also called green supply chain management) has emerged as a critical strategic issue for many companies because such practices improve the environmental performance of their supply chain partners while also increasing corporate and sustainability compet-

1 Introduction

Production		Use Phase		End of Life	Total
Supply chain	Production	Fuel supply	Fuel consumption	Recycling	
5.7tCO ₂ e	0.8tCO ₂ e	5.5tCO ₂ e	29.0t CO ₂ e	2.7CO ₂ e	43.7t CO ₂ e

Table 1.1: Holistic life cycle approach on carbon footprint over its manufacturing, use and recycle phase of Volkswagen (VW) Group car

itiveness. OEMs are currently requiring their parts or components suppliers in the supply chain to be more transparent in terms of carbon emissions reporting and disclosure.(Hagmann et al., 2019). Enterprises like Walmart, Nike, Ikea in the past have asked their suppliers to disclose their carbon footprint(Gerdes, 2012).

Among the various economic sectors, industries such as the automotive industry contribute significantly to GHG emissions. According to (Stephan et al., 2019), the carbon footprint produced by automotive industry was 9 percent of total GHG emissions in 2018, about 4.8 Gt CO₂e ¹. For comparison, the total GHG emissions of entire European Union was 4.1 Gt CO₂e. In the past, the automobile industry focused primarily on lowering their fuel consumption and using more eco-friendly fuels in order to reduce GHG emissions. However, the complete picture of GHG emissions in the automobile industry can be developed only by understanding the entire life cycle of a vehicle, starting from raw materials, processes, and energy sources employed during the manufacturing, use, and recycling phases.

Volkswagen (VW) Group conducted a recent holistic study on carbon emissions from Volkswagen (VW) Group cars over their lifetime, using a life cycle approach throughout the manufacturing, usage, and recycling phases. The data released by VW can be found in Table 1.1. According to the data, emissions recorded during the production of an average VW group car were approximately 6.5 t CO₂e ². This is a significant amount since this contributes about 15 percent of the total emissions the car will contribute in its lifetime. The majority of the emissions are caused by the supply chain,with only 12 percent being emitted by the VW Group directly(Stephan et al., 2019). The study clearly demonstrates that the amount of GHG emitted for producing a car is determined by the amount of carbon consumed during the various stages of the industrial processes. Companies now recognize that concentrating solely on internal operations is insufficient for mitigating and reducing carbon emissions. They must also be aware of their broader supply chain and its partners, which can be extremely complex, spanning multiple countries and continents. The use of emerging technology such as blockchain will be a robust solution for harmonizing the data from all supply chain partners. Furthermore, blockchain-based emissions tracking will be easy to employ with mutual trust from all partners involved.

Bottcher et al. discusses two key drivers for innovation in the automotive industries to adopt low-carbon operation approaches. One of the drivers is the regulatory pressure

¹giga tons of carbon dioxide equivalent

²tons of carbon dioxide equivalent

exerted by stakeholders (such as governments, NGOs, media and general public, etc.) to reduce the carbon footprint of the products and processes. Regulatory pressure can take the form of setting emission limits for a fleet of cars or demanding accountability and necessary action. The second driver is a competitive advantage that the company gains directly or indirectly by implementing low-carbon operating approaches. The benefits include more efficient use of resources such as energy and increased product sales as a result of its product reputation for being environmentally friendly among the general public(Böttcher, 2013).

However, several challenges prevent companies, especially in the automotive sector, from developing a mitigation strategy to derive insight and reduce product carbon footprint effectively. The majority of accounting of carbon footprints across supply chains are still performed manually. This significantly increases the cost, effort, and time required to track carbon footprints(Liu et al., 2019) (Lee, 2012). Chances of errors are also very likely(Banerjee, 2018). Even when some companies attempt to track their product carbon footprint, the challenge is that each supply chain participant has their own local database representation and data model, which can lead to data inconsistencies and unavailability. These factors contribute to the disparity in data availability between manufacturers. As a result, automobile manufacturers do not publish or release comparable, standardized data on emissions from the supply chain on an annual basis for individual car models(Banerjee, 2018)(Stephan et al., 2019)(Lee, 2012). Furthermore, according to Trucost’s calculations, companies in various sectors tend to under-report their carbon emissions in corporate disclosures by up to 7 percent.(Werner and Brian, 2018).

For the two drivers of innovation for adopting of low carbon operation approaches discussed by(Böttcher, 2013) to make an impact, it is essential that the carbon emission information must be collected throughout the supply chain processes and stored reliably for future retrieval. In Taiwan, such an online database system exists which can record, calculate and disclose the product carbon footprint of local industries(Liu et al., 2019). The use of the traditional database system for carbon footprint calculation across the supply chain poses some limitations. The traditional database system relies on central authority (or a third party) who should be trusted with the control and confidentiality of data stored on the database. The possibilities of a single point of failure are significant because data is kept in a central database system. The access to traditional database systems is controlled and regulated using the access control mechanism, which has its weaknesses and can be exploited. As a result, the security of such database systems is called into question.(Chowdhury et al., 2018).

In our present study, we investigate using a disruptive technology called the blockchain as a potential solution for carbon traceability in the automobile supply chain. It is a distributed ledger technology which can record transactions in a transparent, immutable, verifiable, anonymous and autonomous way. Blockchain can be effectively utilised to track carbon emissions from individual supply chain participants to provide a holistic overview of carbon footprint. Blockchain-based tracking of carbon footprint also provides the potential to the industrial participants to prove their claims of reduced carbon emission and encourage the industry to use improved and efficient

processes and better resource management. We also investigate the possibility of combining a price for the traced carbon emissions using tokens based on regulations. Despite several inherent advantages, blockchain technology for the said application is still in the conceptual stage, or efforts for the implementation are very limited, which is the primary motivation behind the proposed study.

1.2 Research Purpose and Questions

This thesis aims to leverage the capability of blockchain technology to design and prototype a system that addresses the challenges identified in the carbon footprint tracing in the automobile industry. The goal of the thesis can be broken down into the following research questions:

Research Question 1: Can blockchain technologies be used for traceability of carbon footprint data for an automobile supply chain and payment of carbon price? What are the requirements of this system ?(RQ1)

A detailed literature review is necessary to focus on the issues of the currently implemented systems using blockchain technologies such that the result of the study becomes the requirements of the system and the parameters of evaluation of the solution.

Research Question 2: What are the approaches of architecture design for the blockchain-based carbon footprint traceability system?(RQ2)

This research question focuses on developing an architecture and a data model for the system before the implementation phase. The architecture provides a clear view of different components present in the system and how each component interacts with the other. These components can be modules, processes, functions. Especially, since blockchain technology is a decentralized system, the design decision require different approaches than the conventional systems. The decisions on which information are needed to model the supplychain entities, products and processes along with their interactions from different perspectives of the system design.

Research Question 3: How can the designed system architecture be implemented as a blockchain application ?(RQ3)

A prototype can be implemented on the blockchain as a decentralized application (Dapp) using smart contracts. A suitable blockchain will be selected based on the requirements of the system. The supply chain entities and their interactions can be modeled using blockchain tools such as smart contracts.

Research Question 4: How to evaluate the applicability and feasibility of the implemented solution ?(RQ4)

This research question focuses on evaluating the implemented system based on the applicability to the identified problem and evaluating the advantages and disadvantages relative to the existing systems. Secondly, a study using scalability and performance

as the parameters is needed to validate the feasibility of the system.

1.3 Thesis Outline

In this section, we present an outline of the thesis. After the introduction chapter, the related work is presented in Chapter 2. The first part of Chapter 2 discusses the background concepts and related technologies of the thesis. The second part of the chapter gives a review of the related academic literature available. Chapter 3 gives a conceptual description, design and modeling for the solution. The implementation details are explained in Chapter 4. Chapter 5 covers the evaluation of the system. The final chapter, Chapter 6 summarizes the finding and discusses the possible future work.

2 Related Work

This chapter provides a brief overview of the concepts used in this thesis and an overview of existing related works highlighting the issues and gaps that motivate this thesis work.

2.1 Background

2.1.1 Automotive supply chain

Automotive supply chains are complex networks of numerous participants who work together to manufacture finished automobile products such as cars. Original Equipment Manufacturers(OEMs), suppliers, distributors, dealers, service providers, and customers from all over the world are among the many participants in the supply chain (Miehle, 2020). A typical automotive supply chain is shown in Figure 2.1.

The activities in the supply chain can be broadly classified as upstream activities and downstream activities. OEMs do not manufacture the bulk of the parts or components of cars they use in-house. They typically operate as integrators. So, they rely on a large number of suppliers for parts, which are distributed through supplier networks. Upstream activities include selling raw materials, components or services from the suppliers to OEMs. In a typical supply chain, there are three or more tiers of suppliers. The OEMs receive materials straight from the first-tier suppliers. Materials or intermediate components are sent to first-tier suppliers by second-tier suppliers. The third-tier suppliers deliver raw materials to second-tier suppliers. The structure of such a supplier network is shown in the Figure 2.1. Similarly, the downstream activities include selling the finished product from OEMs to customers (Liu and You, 2021). The downstream activities are also divided into tiers such as wholesalers, retailers, and end-users. Management and orchestration of the various heterogeneous activities in the supply chain have many challenges, especially due to the large number of participants involved(Waters, 2003).

2.1.2 Distributed Ledger Technology

A distributed ledger is an append-only distributed database that is immutable and shared, replicated, synchronized by multiple parties across a network(Miehle, 2020). Distributed ledgers are implemented in several different ways, collectively called the distributed ledger technology (DLT). DLTs use the peer-to-peer network to incorporate its characteristics like data immutability, non-repudiation, integrity, fair access, transparency. The implemented DLT systems differ based on data structures used,

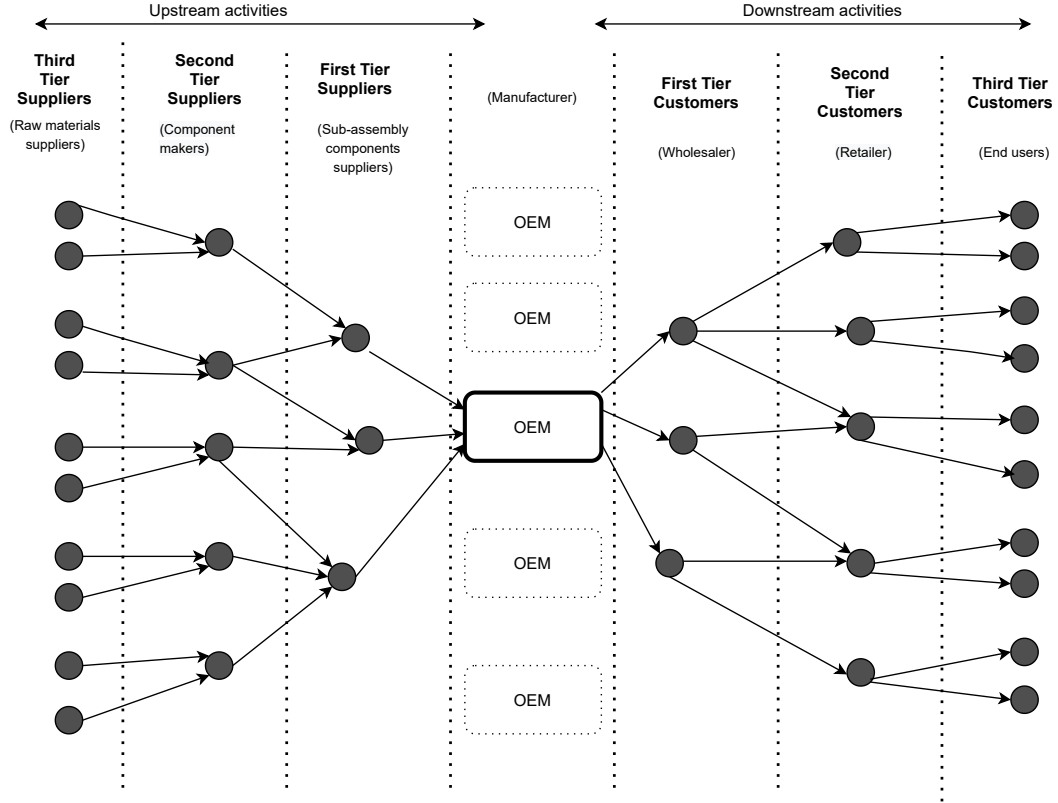


Figure 2.1: A typical Automotive supply chain with one OEM which is represented by solid rectangle. Representation adopted from (Waters, 2003)

consensus mechanism, permissions to access and make changes on the network, ability to use smart contracts (Collomosse, 2020). Blockchains are a subset of the DLTs that have gained popularity in recent years owing to their unique properties and capabilities.

2.1.3 Blockchain

A blockchain is a timestamped, append-only, immutable, decentralized, peer-to-peer distributed data store whose integrity is secured by cryptographic signatures and consensus-based validation protocol (Zheng et al., 2017) (Xu et al., 2017). Satoshi Nakamoto proposed one of the first blockchains in his white paper (Nakamoto, 2008) in 2008, and Bitcoin blockchain was the first implementation based on his white paper. The use case of the Bitcoin blockchain was to serve as a public transaction ledger for the digital currency Bitcoin. Since its inception in 2008, blockchain has continued to evolve, and it now offers a wide range of application and use cases, as opposed to the trust-free tracking of digital currencies in its early days.

The blockchain is a linear growing list of blocks containing transactions. The trans-

2 Related Work

actions are considered fundamental unit in a blockchain which usually represents a transfer of value from one entity to another (Bashir, 2020). The data structure used in the blockchain is a linked list of blocks. This data structure only allows appending new blocks in chronological order and does not support modifying the already appended blocks. The data in the blocks are converted to standardized formats and hierarchically compressed using the hash function. The resulting tree structure is called a Merkle tree or hash tree. The output hash value represents the blocks. Since modifying even one transaction causes a change in the hash value of the block and the subsequent blocks in the Merkle tree to become inconsistent, so the transactions within a block are safe from manipulation. (Weimert et al., 2018).

A block is composed of multiple transactions and multiple blocks are linked together using a hash pointer¹ that uses the hash value of the previous block and root hash of the current block (Miehle, 2020). Every block, as indicated in the Figure 2.2, has a block header and block body which is the ordered sequence of transactions. The metadata about the block is stored in block header, which includes timestamp, previous hash, root hash, block size, version, nonce, and difficulty target. The body of the block contains the transactions. The genesis block is the first block on the blockchain and it has no predecessor blocks, hence this is the only block that does not contain the previous hash (Gallersdörfer et al., 2020) (Zheng et al., 2017). Different blockchain implementations use different block structures. Figure 2.2 shows the Bitcoin block structure.

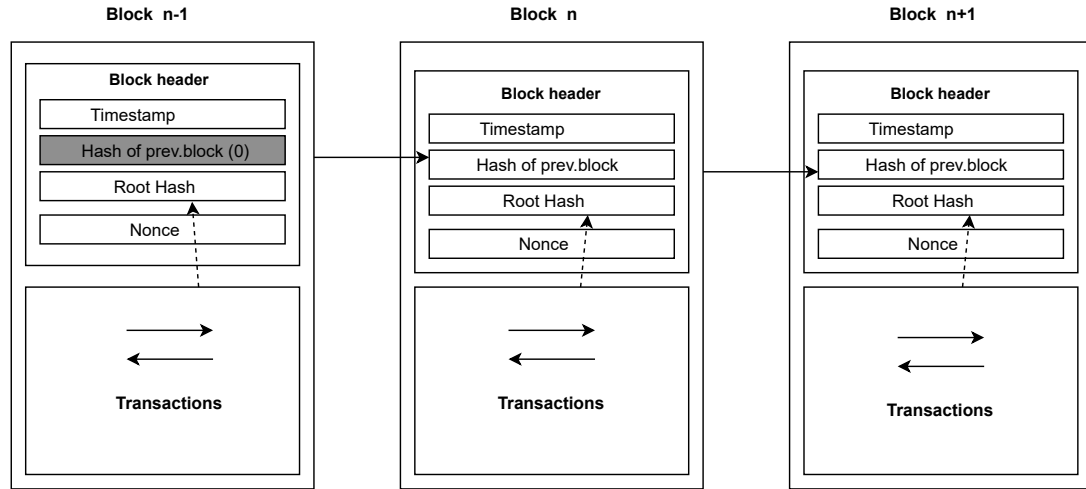


Figure 2.2: A representation of blockchain. Adopted from (Gallersdörfer et al., 2020)

¹A hash pointer contains the information to the location of the data along with the cryptographic hash of it.

2.1.4 Cryptography

Blockchain technology is built on the foundations of cryptography. Different cryptographic methods are employed to ensure the integrity of blocks, validation of the blocks, the authenticity of transactions, and the authentication and non-repudiation of the participants of the blockchain network(Weimert et al., 2018). The hash function and digital signatures are the two cryptographic methods used in the blockchain.

A hash function is a cryptographic algorithm that compresses and maps a data input of arbitrary length to an output of fixed length. A hash function can be used to create unique digital fingerprints that can verify if a transaction has not been manipulated later. For a function to be a hash function, it has to satisfy the following properties(Gallersdörfer et al., 2020):

- Given the data input and hash function, it should be easy to compute the output of fixed length
- The hash function always generates the same hash for a given input data
- It is computationally infeasible to find the input data from the output of the hash function(Hong et al., 2006)
- It is infeasible to find a second input data that generates the same hash output(Hong et al., 2006)
- Also called the Avalanche effect, even a slight change in input changes the output hash value significantly.

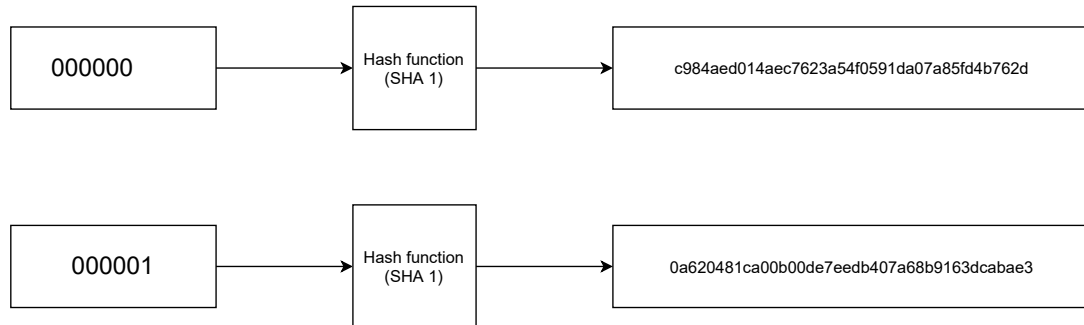


Figure 2.3: An example to show the avalanche effect

Digital signatures is another way the integrity of the blockchain is ensured. Digital signatures are based on asymmetric cryptography algorithms like RSA or ECC that use the cryptographic key pairs (public key and private key) as identities in the system(Gallersdörfer et al., 2020). Digital signatures are used as a means to associate a transaction with an entity from which the transaction has originated so that the origin can be authenticated and provides non-repudiation of the transaction(Bashir, 2020).

2 Related Work

The digital signatures provide important properties. The first property is authenticity, which means the digital signature should be verifiable by the everyone including the receiving party. The second property is unforgeability, which means only the sender of the transaction can create or use the signing functionality of the digital signature using their private key. The third property is the non-reusability which means the digital signature is tied to the transaction that gets signed and it cannot be reused for a different data (Bashir, 2020). Among the cryptographic key pair, the private key is known only to the owner of the key pair. The owner of the key pair uses the private key to sign or encrypt data (as a digital signature), so the access to private key should be protected. In comparison, the public key is known to the other participants in the network, and it is used by other participants to verify the authenticity of the signatures (Liss, 2018).

2.1.5 Addresses

Addresses are unique identifier used in a transaction on the blockchain to identify senders and recipients. The addresses are usually a public key or derived from a public key (Bashir, 2020).

2.1.6 Peer-to-Peer Systems

Most blockchains such as bitcoin use a peer-to-peer network (P2P network), which consists of many nodes also called a distributed network of users that maintains the copy of the blockchain. The distributed network also allows any interested user to participate in the network to verify and validate the blocks by setting up a blockchain node. Since it is a distributed network, no single entity is superior to another (Chowdhury et al., 2018). The nodes are of different types based on its functions. For example in a bitcoin blockchain, based on its role in the blockchain network the nodes are classified as full nodes², light nodes³, miners⁴ and wallet owners⁵.

2.1.7 Consensus Mechanisms

A consensus mechanism utilizes the consensus algorithms so that the participants of the blockchain can agree on the state of the blockchain without the intervention of any trusted central authority. Consensus algorithms provide the rules for selecting a node as the validator that adds the new blocks to the blockchain. The participating nodes in the network can also recognize and protect against invalid blocks added by bad actors following the consensus algorithm (Gallersdörfer et al., 2020). Many consensus algorithms have been proposed using various rules, metrics, and incentives

²A full node stores the complete blockchain. Every transaction and block in a blockchain is verified, authenticated, and stored by all full nodes.

³A light node usually stores the block headers

⁴A miner is responsible for adding a new block to the blockchain after trying to solve the cryptographic puzzle. They store the complete blockchain like full nodes.

⁵A wallet owner signs and publishes new transactions to the network

and addressing the need for different DLT applications. Two of the most widely used consensus algorithms are discussed below.

Proof-of-Work(PoW) is the most commonly known consensus algorithm because it is used in the Bitcoin blockchain. In PoW, every time a new block gets added by a node, the nodes (or the miners) have to spend their computing power which is a scarce resource, to solve a complex cryptographic puzzle for the block to be considered valid. This process is called mining(Iuon-Chang, 2017). Every miner receives an incentive for using their computational power. For every valid block that is added to the blockchain, the miners receive a block reward and the transaction fee. On the other hand, some drawbacks of PoW are low transaction rates, higher energy costs, high initial investment costs, higher carbon footprint.

Proof-of-Stake(PoS) is another consensus algorithm. To propose a new block on the blockchain, the network participants must deposit an amount in native cryptocurrency(Iuon-Chang, 2017). It is a risk-based mechanism, where the participant risks their economic deposit by creating invalid blocks. Everyone can participate in the network and become a validator if they are willing to pay the initial deposit. After a new block is added to the blockchain, a weighted random selection process selects a node as the validator. The penalty for bad behavior such as proposing or validating an invalid block is the deposit getting reduced. The advantage of PoS is that it does not require any energy-intensive mining process.

2.1.8 Characteristics of blockchain

Based on the concepts discussed in Sections 2.1.3, 2.1.4, 2.1.6, 2.1.7 we can conclude on some properties exhibited by blockchain.

- Immutability and irreversibility of the chain state
- Transparency and accountability
- Decentralized data storage
- Distributed consensus and trust
- Data persistence
- Data provenance

It should, however, be noted that these characteristics may or may not be exhibited by all the blockchains and they change based on how the blockchain is designed and implemented.

2.1.9 Types of permissions in blockchains

In the literature, blockchains are classified based on permissions in two key aspects:

The first type of permission determines who can join the blockchain network based on the ownership of the network (i.e., public blockchain or private blockchain). Public

2 Related Work

blockchains (such as Bitcoin and Ethereum main net) allow anyone to join the network. Private blockchains, on the other hand (such as Hyperledger), limit who can join the network. The network can only be accessed by authorized users(Iuon-Chang, 2017). The second type of permission determines if a participant of the blockchain is allowed to enter transactions and execute other operations onto the blockchain. Permissionless blockchains allow all the participants to add new blocks, whereas the permissioned blockchains allow only a set of participants to add new blocks(Xu et al., 2017).

Based on the combination of these two permission types we have four types of blockchain networks:

1. Public-permissionless blockchain: Anyone can participate in the network, read transactions, commit transactions or add blocks to the blockchain following the rules of consensus algorithm. Examples include cryptocurrencies that mainly use the PoW consensus algorithm: Bitcoin(Nakamoto, 2008), Ethereum(Buterin, 2013).
2. Public-permissioned blockchain: Anyone can participate in the network and read the transactions. But only a set of authorized participants can commit transactions or add blocks to the network. Such type of blockchain usually uses the PoS consensus algorithm. Example: Ripple(Chase, 2018), Sovrin(Khovratovich, 2016)
3. Private-permissionless blockchain: Participation on the blockchain is limited to the authorized users who are granted access by the owner of the blockchain. Since it is permissionless, everyone granted access to the blockchain can commit transactions and add new blocks to the blockchain. Example: Private Ethereum Implementation(Go-Ethereum, 2016).
4. Private-permissioned blockchain: Participation on the blockchain is limited to the authorized users who are granted access by the owner of the blockchain. Participants required to be given permission to be able to commit transactions to the blockchain. Example: Hyperledger Fabric(Androulaki et al., 2018), Private Ethereum Implementation(Go-Ethereum, 2016).

2.1.10 Smart Contracts

Smart contract is defined as a computer program with a set of functions that other users or contracts can call. It runs on the blockchain to automatically enforce the terms and conditions of agreements between untrusted parties without the need for a trusted third party. Nick Szabo(Szabo, 1997) was the first to define the notion of smart contracts, but it was only with the development of Ethereum that use of smart contracts became possible. Smart contracts can be used to execute functions, send or receive cryptocurrencies, update account balances, and store data. Execution of the smart contract is deterministic and immutable once it is deployed on the blockchain. Every smart contract is an account holding object, so it has its own address. Sending

transactions or messages to the address of the smart contract can trigger it(Christidis and Devetsikiotis, 2016). When a contract is deployed, it becomes publicly accessible to all network participants or blockchain nodes. This emphasizes the importance of security considerations when building smart contracts. One of the largest use case of smart contracts are token systems (such as Initial coin offerings).

2.1.11 Tokens

In the context of blockchain, a token is a piece of data that grants an access right and/or represents an asset that is collectively regulated on a blockchain using the smart contract(Voshmgir, 2019). All tokens are not cryptocurrencies, but cryptocurrencies are tokens that can be used as money that holds value in the real world. Tokens are one of the core elements of the internet of value idea, which allows even the rights to real-world values to be digitally mapped on the blockchain(Weimert et al., 2018). Before the origin of Ethereum blockchain, each new token required the creation of a blockchain that implemented the token. It is possible to build multiple token systems on Ethereum. As a result, many token-based systems, such as ICOs, are built on Ethereum to benefit from the existing Ethereum ecosystem.

The tokens are classified into three types based on their functionality(Angelo and Salzer, 2020):

- Utility tokens: These tokens are closely tied to the functionality of the issuing network or application platform with a defined benefit or service by using a blockchain-based infrastructure.
- Security tokens: These tokens behave like a security deposit such as equities(in an issuing company), bonds, or derivatives.
- Cryptocurrencies: These tokens are majorly used as payment methods or medium of exchange.

Based on the design, the tokens can be classified into two classes(Gallersdörfer et al., 2020):

- Fungible token: The value of all the tokens is identical, and they are indistinguishable from one another.
- Non-fungible tokens: Every token is identifiable and has a unique ID; hence they are not mutually interchangeable.

2.1.12 Decentralized Identity

Decentralized identity(DID) is a novel identifier that provides a data-model for decentralized, verifiable digital identification. It is a unique identifier that maps the subject to a DID document containing information about the identity of the subject](Brunner et al., 2020). The subject in this context is like a person, organization, items etc, as determined by the controller of DID. The DID has the following syntax:

2 Related Work

“did:” + <did-method> + “:” + <method-specific identifier>

It is divided into three sections. The URI scheme identifier (i.e. did) is the first component of DID string. The did method, on the other hand, is a reference to a specific distributed ledger or network. The method-specific identifier, the third portion, is used to resolve the DID within the supplied DID method. The actual implementation is provided by the DID method. The DID method accepts DID as an argument and returns a document containing all of the metadata about the identity shown in Figure 2.4 including the public keys. The returned document is most commonly expressed in the JavaScript Object Notation for Linked Data(JSON-LD) structure(Drummond Reed, 2019). The DID document contains public key and other metadata about the subject.

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    // used to authenticate as did:...fghi
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyMultibase": "zH3C2AVvLMv6gmMnam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

Figure 2.4: An example DID document adopted from (Drummond Reed, 2019)

2.2 Literature Review

The topic of this thesis involves three areas: blockchain technology, supply chain traceability, and environmental sustainability in the automotive industry. The relevant literature for this work was primarily identified from literature that discussed various approaches for tracing carbon footprint across the supply chain using blockchain and some literature that provides the methodology for tracing carbon footprint in a complex supply chain, such as the automotive supply chain. The scientific databases and academic search engines used for searching the literature were Google Scholar, ScienceDirect, Scopus, Web of Science, and Research Gate. The identified literature included research papers from published journals, conferences, technology white papers, poster presentations. We have used the keywords “carbon footprint traceability” AND “blockchain” AND “automotive supply chain” to search these databases.

Limited studies were found in the literature that discusses the area of carbon footprint traceability in supplychains using blockchain. Despite this, in the following

subsections, we describe the most relevant works that are found related to the thesis problem. Each subsection describes the problem the related work tries to solve, goals achieved, and the described implementation approach.

2.2.1 Blockchain as a Tool for Carbon Emission Management (Liu et al., 2019)

Liu et al. discussed a framework wherein blockchain is used as a tool for the management and governance of carbon emission. The framework is a conceptual idea, and it has not been implemented nor prototyped as per this manuscript. This paper aimed to demonstrate how blockchain can be used for emission management applications in conjunction with a database to complement it with its unique features such as decentralization, irreversibility, data immutability, non-repudiation, and transparency.

The framework under discussion has three tiers. The tiers are given various names depending on their function. The calculation layer is the initial layer. In an enterprise, the calculation layer is the data collection layer. This data can be collected in two ways: manually from inventory or through IoT devices or sensors that capture data such as consumption of raw materials, energy, and resource (water, electricity, fuel, or gas). This information is sent to the blockchain as transactions. The blockchain layer is the next layer to be addressed, and it performs three critical functions: validation, consensus, and supply chain monitoring. The carbon footprint of the product is estimated using smart contracts disseminated to every node in the blockchain network, and data transmitted to the blockchain as transactions are validated using a consensus protocol. This carbon footprint information can be released to the general public. The authors claim that by including the blockchain layer into the system, the general public, industry, and local governments can become stakeholders and take necessary actions to cut carbon emissions. The integration layer is the third layer covered in this paper. This layer retrieves the carbon footprint data of every entity involved in the supply chain from the blockchain. For quick data retrieval, this study recommends using a database. When new transactions are added to the blockchain, the database can be updated.

We think this work provided an abstract framework to use blockchain as a carbon emission management tool. This work gives a conceptual view of the three layers but does not specify how they can be implemented. The blockchain in this work is visualized like another central database.

2.2.2 Using Blockchain Technology to Re-engineer the Carbon Supply Chain(Banerjee, 2018)

Banerjee et al. explored how to use blockchain to re-engineer the supply chain and build an uniform ecosystem for tracking carbon footprint across the supply chain and intra-organizational decentralized carbon offset trading. This is a conceptual idea because it is a technology white paper. This concept has yet to be implemented or prototyped. This paper proposes two distinct blockchain architectures as a way for

2 Related Work

using carbon credits as a standard measurement of carbon footprint. The product carbon footprint across multiple players in a supply chain is another crucial concept addressed in the article. To gather data across the supply chain, the work suggests connecting systems like ERP (Enterprise Resource Planning) and SCADA (Supervisory Control and Data Acquisition) to the blockchain. In this situation, the blockchain serves as a uniform, integrated platform for tracking carbon emissions, calculating carbon emissions, creating carbon labels, pricing carbon credits, and sharing carbon credits.

The first approach discussed proposes to use an open blockchain such as Bitcoin, Ethereum, or Multichain. There are two types of actors interacting with the open blockchain system. All manufacturers, suppliers, distributors belonging to the supply chain can join the blockchain network as members, which is the first type of actor. The second type of actor will be independent authenticators who join the blockchain network. The emission data from the members will be from their ERP systems and recorded on the blockchain. The supply chain managers can use the blockchain to perform carbon emissions calculations. The participants of the blockchain issue and consume carbon credits in accordance with carbon emissions trading regulations. These transactions are validated by independent authenticators who check the emission from the members and tradings are happening between the members as per the regulations. Some disadvantages of the open blockchain are: The transactions are openly available to all participants of the blockchain. Since the blockchain used is a public blockchain, access to this information cannot be restricted because anyone can join the open blockchain. The infrastructures necessary for this implementation are a large-scale distributed ledger and high computational power.

The second approach discussed proposes to use a private or a permissioned blockchain such as Oracle Blockchain Cloud or Hyperledger Fabric. An OEM creates a blockchain network and invites all suppliers, service providers, distributors, and other parties involved in the supply chain to join it. To join this blockchain network, the participants have to agree to the emission calculation standards, terms and conditions of the carbon credit system, etc. The smart contract ensures that the agreed regulations are enforced, removing the need for an independent external authenticator.

We think this work focused on providing the architectures of a blockchain application that could be used for carbon emission tracking and emissions trading as a unified platform. The data recorded on the blockchain is accessed through oracles from other existing ERP and SCADA systems. The paper also does not specify which emission data, such as direct stack emissions and indirect emissions, are used to calculate the product carbon footprint. The work assumes that these IT systems are trustworthy. The work also mentions carbon labeling as a concept but did not mention how it is part of the two given architectures.

2.2.3 Blockchain-based Traceability of Carbon Footprint(Rosado da Cruz et al., 2020)

Rosado da Cruz et al. discussed a blockchain-based platform for traceability of carbon footprint across the supply chain. Consumers of the product can use this platform to track information about the product carbon footprint. The paper firstly tries to define the carbon footprint tracing on the basis of Hybrid EIO-LCA approach (EIO: Environmental input-output, LCA: Life Cycle Analysis). The study analyzes the possibility that a product may have several components that different suppliers may supply, and several activities may be involved in creating these components. These activities also contribute directly and indirectly to the total carbon footprint of the product. The paper proposes calculating the carbon footprint every month for each product—the activities involved in making the product contribute to the carbon footprint. The carbon footprint from these activities can be recorded for a month. Each supplier or manufacturer is a node. They insert the transactions to a public blockchain such as the Ethereum blockchain.

A working prototype for this research paper is developed using a Decentralized application connection to Ethereum blockchain⁶. A smart contract is written in Solidity⁷ which controls the insertion of the transaction into the blockchain. The Ethereum network is run locally using a Truffle suite during the development⁸. A NodeJS⁹ middleware is used to interact with the smart contract using Web3.js libraries¹⁰. The front end of the application is developed using React framework¹¹.

We think this work demonstrates one of the approaches to implement carbon tracing in a blockchain for supply chains having large number of suppliers. However, public blockchain usage may not be possible for some enterprises since it needs to maintain a large-scale distributed ledger and need high computation power. Also, there is no control over data access on a public blockchain. This platform is only for carbon tracing; it does not offer support for regulation such as emissions trading, cap and trade, or carbon taxes.

2.2.4 Carbon Labelling- A system for calculating product carbon footprint using Blockchain(Pellen-Pickersgill et al., 2019)

Pellen-Pickersgill et al. discussed a system that allows the companies and their suppliers to log their carbon footprint data on the blockchain for individual products and to retrieve the logged information by a mobile application for the customers to view this information. The use of blockchain allows for monitoring and storage of carbon data across the supply chain.

⁶<https://ethereum.org/en/>

⁷<https://docs.soliditylang.org/en/v0.7.4/>

⁸<https://www.trufflesuite.com/>

⁹<https://nodejs.org/en/>

¹⁰<https://web3js.readthedocs.io/en/v1.3.0/>

¹¹<https://reactjs.org>

As stated before, there are two separate components in the work. Carbon labelling or a barcode that can track the product across the supply chain is attached to the product. The OEM that sells the final product is responsible to provide the barcode information. The OEM adds suppliers as participants to the blockchain so that they can insert transactions. The suppliers log the carbon footprint data on the blockchain as transactions after they have finished working on the product within the supply chain. Suppliers have additional rights such as adding or removing more suppliers (their downstream suppliers) and adding or removing senders. Each supplier is conceptualized as a node of the blockchain. The transactions on the blockchain will record information like carbon footprint data, supplier details and location.

The second component of the system is a mobile application that users may use to view information about a product carbon footprint and provenance by scanning the barcode on the goods. The mobile application will get the data from a database that gets updated every day from the current state of the blockchain.

This work is prototyped using an Ethereum blockchain¹² running on Ganache¹³. The work was not deployed on the Ethereum main net since it is a prototype and the TestRPC simulate the main net pretty well.

We think this work provides an idea that carbon labelling could be used to track the carbon footprint of an individual product. However, the report does not state which supplier tier the carbon labeling is applied to.

2.2.5 A Decentralized Traceability Application for Multi-Tier Automotive Supply Chain Networks (PartChain) (Miehle et al., 2019)

Miehle et al. discussed PartChain, a decentralized supply chain traceability application that creates a digital representation or a digital twin of the physical part (component) of the car produced by the suppliers and enables monitoring and tracking on the blockchain network. Authors present this approach as a cost effective and easy to operate system especially for small and medium size automobile industries as it can bridge the gap between actual part and digitalisation through a mobile device. If a shipment contains defective parts, OEMs and suppliers can quickly identify the parts using the PartChain.

The system aids in establishing the provenance and authenticity of a part. To achieve this, the system uses a QR code on the parts and a PartChain mobile app that scans the QR code, the current location and timestamp. So using the app the supplier can register their car parts. The transfer of ownership from suppliers to OEMs and delivery is handled digitally by smart contracts. The network also tracks, traces, and records all events related to the part. The network participants can validate the authenticity of the part as if it was manufactured and delivered by a registered supplier by scanning their QR code. The records on PartChain digitize the documentation, removing the need for manual signatures throughout the supply chain. The system is built on a permissioned blockchain, so only registered users can access the network.

¹²<https://ethereum.org/en/>

¹³<https://www.trufflesuite.com/ganache>

In the prototype, the application subsystem contains two components. A mobile app and a web client. The web client is developed using Angular 5¹⁴ framework. The blockchain network used for the prototype is Hyperledger Fabric¹⁵.

We think this study provides a reference framework for developing a blockchain solution for managing multi-tier supply chain networks in the automotive industry. The study makes no mention of dealing with carbon footprint data while tracking the parts across the blockchain.

2.2.6 Carbon Credits on Blockchain (Patel et al., 2020)

Dhiren Patel et al. presented their blockchain-based carbon credit trading solution and discussed its advantages over the Clean Development Mechanism registry in terms of usefulness, participation, and cost. The paper proposes a token-based economy for decentralization and transparency, with digital tracking of carbon emission allowances using blockchain globally. It also suggests a safeguard against carbon credit hoarding. This token-based cap-and-trade scheme proposes three participants: generators, consumers, and issuers. Generators are corporations that have been granted permits to offset carbon emissions through innovative offsetting methodologies or that have a surplus of carbon credits due to lower consumption. Corporates with short of carbon credits given to them are regarded as consumers and hence, can opt to procure the permits from generators. Issuers are the entities that set the cap for every year based on the auditing of emissions in the industry and also validate the carbon offset of the generator.

During the initial phase of the cap-and-trade system, the permits are issued by the issuer and distributed among the organizations; alternatively, the permits can be auctioned, and the proceeds from the auction can be used to fund environmentally friendly technologies or research. Based on the positive contributions of the generators to the environment, they can approach the issuer for the issuance of new tokens after proper verification and recording in the blockchain. In the event that consumers exceed their carbon emission limits, they can purchase tokens from the unified trading platform with specified expiry dates. Tokens become invalid after their expiry dates or can be returned to the generator, thereby eliminating hoarding. The industry is subjected to a progressive reduction in the emission cap in order to encourage investments in environmentally friendly technologies that generate carbon credits rather than purchasing them. The consumer will be able to emit only the equivalent amount of purchased carbon credits, after which the consumer will not be able to spend the credits and all further transactions will be locked. Blockchain also allows for independent auditing of tokens to prevent the sale of bogus tokens. Buyers can also verify the digital signatures of the issuer to rule out any potential scams. We think this study provides a basic foundations of carbon credits and how to leverage the token systems on blockchain and implement them digitally.

¹⁴<https://angular.io/>

¹⁵<https://www.hyperledger.org/>

2.2.7 Integrating Carbon Footprint into Supply Chain Management for the Automobile Industry (Lee, 2011)

Ki-Hoon Lee et.al reported a carbon footprint management system through a case study of Hyundai Motor Company. According to the study, the critical steps in incorporating the carbon footprint into supply chain management begin with identifying and measuring carbon emissions from direct and indirect sources, followed by the evolution of the carbon footprint map of product carbon footprint. In the study, the authors collected the primary data through site visits and interviews with officials of Hyundai Motor Company and demonstrated identification and measurement of the carbon footprint of the major parts from important suppliers; the car manufacturer can estimate the total carbon footprint of a car. Such an estimate will enable the OEM to plan and fix the targets for the participating suppliers to mitigate the cumulative burden of the carbon footprint.

This study provides an improved understanding of carbon footprint in supply chain management from the automobile industry perspective. However, the study relies heavily on self-reported measurements and empirical calculations, which limits generalizability. The empirical case study of HMC and its key supplier's front bumper product provides some evidence regarding how to practice carbon footprint measurement and improve the environmental performance of CO₂ emissions. By identifying and measuring the carbon footprint of main parts and products from key suppliers, an OEM can identify and measure the total carbon footprint.

This paper benefits academics and managers by providing a new way to integrate carbon emissions in supply chain management since climate change and carbon footprint measurement present challenges to many industries. We think this paper provides a framework to trace carbon footprint across the automotive supply chain with multiple suppliers.

2.3 Summary

Researchers have already worked on carbon traceability in supply chains using blockchain, as evidenced by the papers and research works discussed in this section. However, there is a scarcity of publicly available literature on implementation or prototyping. The majority of the literature is comprised of articles or technology white papers describing carbon traceability in a supply chain using blockchain. Many proposed solutions lack concrete implementation and evaluation, and the use of private permissioned blockchain networks has received little attention. Various approaches discussed in this section, perceives the problem from different point of view. Some of the discussed works focused on providing a solution from the customer's point of view. The customer can use the blockchain-based platform to check the carbon footprint of the product and make better decisions. The other perspective is centered on the stakeholders' point of view. Stakeholders use the blockchain-based platform to view their carbon footprint data and make changes to their supply chain activities.

Majority of the discussed works use blockchain have used the open blockchain or public blockchain such as Ethereum, without giving the reasoning for doing so. Public blockchains do not offer control on access to data inserted on them. Some enterprises many not want their data to be available to the public. Adopting the public blockchains in an enterprise scenerio would require high computational power and infrastructure to maintain a large-scale distributed ledger. Only one of the works discussed describes a concept for using carbon labeling to trace carbon at the product level. The majority of the discussed work also did not trigger any further actions using the data on blockchain using the capability of the smart contract such as payments of carbon taxation. Only, one of the discussed work gives a framework to represent carbon credits on blockchain using tokens. A comparison table that summarizes the discussion in this section is provided in Table 2.1.

In addition, the literature review assisted in identifying literature gaps, requirements, and evaluation parameters for our system, which serves as the basis for the concept description in the following chapters. This chapter showed that blockchain technology could be used for traceability of carbon footprint across the automobile supply chain and payment of carbon price by using the carbon credit which could be modelled by a token on the blockchain. Hence answering the research question 1(RQ1).

References ¹⁶	Discussed Aspects				Implementation		
	Customer POV ¹⁷	Stakeholders POV ¹⁷	Carbon Tracing	Carbon Pricing	Prototype	Public Blockchain	Platform
(Liu et al., 2019)	✗	✓	✓	✗	✗	N/A	N/A
(Banerjee, 2018)	✗	✓	✗	✓	✗	N/A	N/A
(Rosado da Cruz et al., 2020)	✓	✗	✓	✗	✓	✓	Ethereum
(Pellen-Pickersgill et al., 2019)	✓	✗	✗	✗	✓	✓	Ethereum
(Miehle et al., 2019)	✗	✓	✗	✗	✓	✗	Hyperledger
(Patel et al., 2020)	✗	✓	✗	✓	✗	N/A	N/A

Table 2.1: Comparison between discussed works

¹⁶(Lee, 2011) is not included in the comparison table since it is not based on blockchain

¹⁷point of view

3 Concept Description

In this chapter, we will present a conceptual description of the unified system that uses blockchain to trace the carbon footprint in an automobile supply chain and pay carbon prices based on the carbon footprint. It will discuss the background of the case, the application scenario, the methodology, the components, the requirements, the use cases, and the architecture. This chapter attempts to answer research question 2 mainly, concretely define the requirement of the system to answer the second part of research question 1 and identify evaluation parameters to answer research question 4.

3.1 Case Background and Definitions

3.1.1 Carbon footprint

Although the term carbon footprint is extensively used, it is not well defined in the literature. Academics, consultancies, businesses, non-governmental organizations, and governments all have their own definitions of carbon footprint. The definition of carbon footprint proposed by Wiedmann and Minx et al. is a measurement of the total amount of GHGs produced directly or indirectly by an activity or accumulated over the life stages of a product(Wiedmann and Minx, 2008). The carbon footprint is quantified in tonnes of carbon dioxide equivalents (t CO₂e). Product carbon footprinting refers to standardization efforts for measuring and reducing GHG emissions across the supply chain. There is also debate over which gases should be included in the definition of carbon footprint(Wiedmann and Minx, 2008). Some definitions only consider CO₂ emissions when defining their carbon footprint. However, recent carbon footprint standardization efforts have led to widespread agreement that all GHG emissions should be included(Jensen, 2012).

3.1.2 Methodologies of Carbon Footprint Tracing in Supply Chain

The green supply chain management literature mainly discusses two fundamental methodologies for tracking and tracing the carbon footprint of a product in a supply chain. It can be performed as a bottom-up or top-down analysis. The bottom-up analysis, also known as process analysis(PA), was developed to understand better the environmental impacts of specific products from conception to disposal. The top-down analysis, also known as environmental input-output analysis, is an alternative approach that employs economic input-output tables for analysis(Wiedmann and Minx, 2008). These methodologies strive to capture the full life cycle analysis of a product. Both of these approaches have advantages and disadvantages. PA has clear advantages when

looking at micro systems such as a specific process or a specific product made of several components(Wiedmann and Minx, 2008). As a result, different carbon footprint calculation standards generally agree that calculating the carbon footprint of a product should be done as a bottom-up PA (Jensen, 2012).

One of the drawbacks of using the PA approach to study life cycle emissions of a product is the lack of a defined scope and boundary within which emissions must be considered(Wiedmann and Minx, 2008). Because of the ambiguous definition of boundary, OEMs in the supply chain may only evaluate the carbon footprint of their products based on direct emissions (onsite and internal emissions), ignoring indirect emissions (upstream and downstream emissions), which can significantly decrease the calculated total carbon footprint of the product.

The standards developed by the WBCSD¹ and WRI² for tracing and reporting carbon emissions is called the Greenhouse Gas (GHG) Protocol. Greenhouse Gas Protocol Product Life Cycle Accounting and Reporting Standard(Bhatia et al., 2011) is one of several standards proposed as part of the GHG Protocol. It takes a product life cycle perspective and provides a framework to standardize indirect GHG emissions and set boundaries for PA. The report categorizes carbon emissions into three scopes. Scope 1 emissions are defined as direct carbon emissions from sources that a company owns or controls. Scope 2 emissions are defined as the indirect carbon emissions from the purchased electricity, heating, steam, cooling. Scope 3 is defined as indirect emissions from activities such as upstream and downstream activities in the total supply chain(Bhatia et al., 2011) (Lee, 2011).

Bhatia et al. give the steps to set the boundary for PA. The first step is to identify attributable processes throughout the life cycle (like raw materials acquisition, components production, product production, distribution, etc.) that are directly related to the product under consideration and group them into different life cycle stages as shown in Figure 3.1.

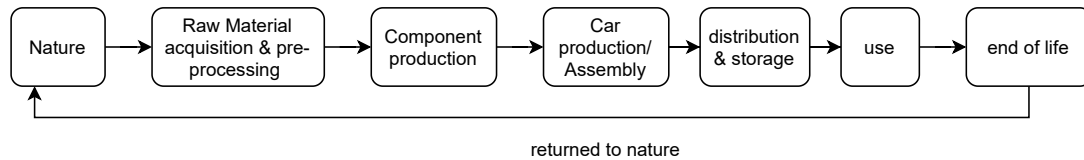


Figure 3.1: The stages of product life cycle in automobile industry. Based on (Bhatia et al., 2011)

The second step is the development of a process map. This map identifies the defined life cycle stages and the corresponding attributable process in each stage, and the flow of the product through the life cycle. A sample process map of the automobile industry is shown in Figure 3.2.

With the carbon map and the three scopes defined in the preceding paragraph, OEMs can define the boundary and scope of which carbon emissions in their supply

¹World Business Council for Sustainable Development (<https://www.wbcsd.org/>)

²World Resources Institute (<https://www.wri.org/>)

3 Concept Description

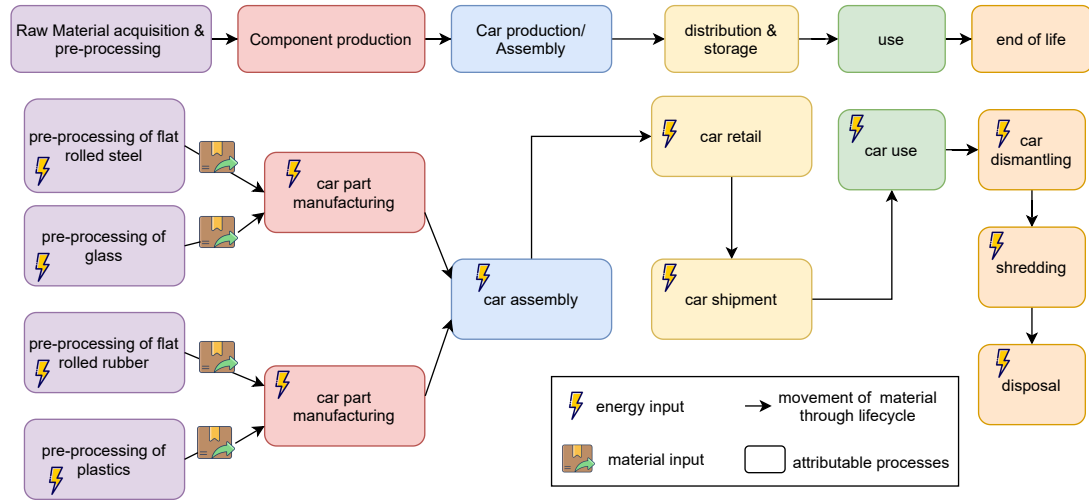


Figure 3.2: The carbon map. Based on (Bhatia et al., 2011)

chain are taken into account when calculating the carbon footprint of their product. From the perspective of the OEM, the upstream and downstream activities fall under the scope 3 emissions as shown in Figure 3.3. They are typically not recorded and reported in product carbon footprint disclosures as discussed in the motivation Section 1.1.

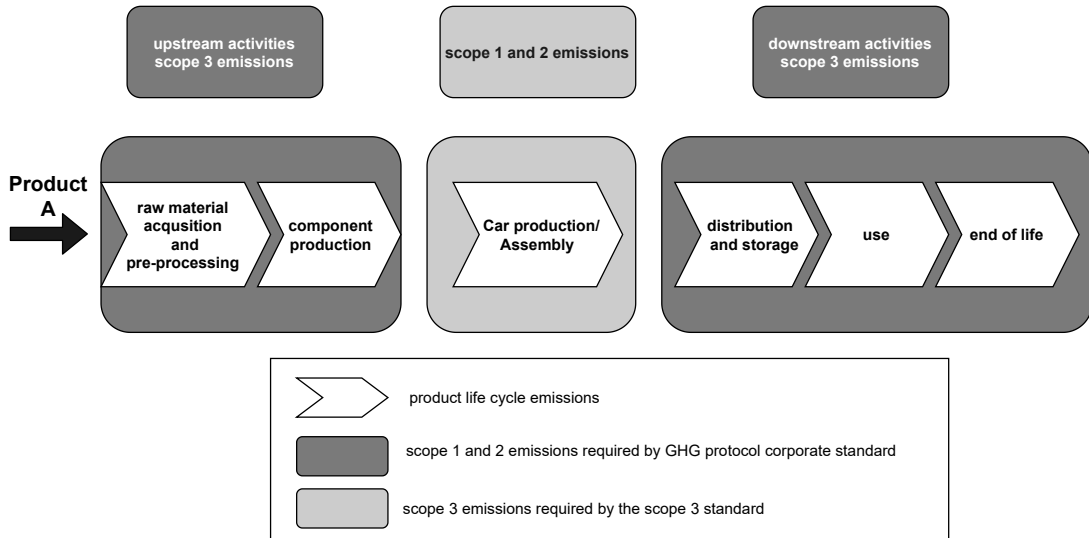


Figure 3.3: Relationship between life cycle of a product and emission standards Based on (Bhatia et al., 2011)

3.1.3 Carbon Pricing Policies

Carbon pricing schemes in the perspective of supply chains may be classified into two major types:

- **Emission trading system (ETS):** In a carbon emission trading system, the government allocates a reasonable amount of carbon credit allowance (carbon emission quota) to supply chain firms such as suppliers, OEMs, and others. When the emissions are fewer than the allotted carbon credit allowance in a company, it can sell the excess allotment on the market. The government also ensures that carbon emission trading are reasonably regulated. The carbon emission trading market determines the price of the carbon credit (Yang et al., 2014).
- **Carbon tax:** In a carbon tax system, the government distributes a fair amount of carbon credit allowance (carbon emission quota) to supply chain firms such as suppliers, OEMs, etc. If the carbon emissions produced by the firms exceed its allowance, the firm should pay for the additional carbon emissions they emit. However, suppose the quantity of carbon emissions is below the carbon credit allowance. In that case, unlike the carbon emission trading, the excess carbon credits cannot be sold on the market (Yang et al., 2014). The carbon tax policy is implemented in different forms in different countries and the carbon taxes are also collected through various mechanisms (Chen et al., 2020). For example, a carbon tax might be implemented such that it is paid exclusively by upstream supply chain members who use fossil fuels, with the tax expense subsequently passed on to downstream supply chain members.

During our literature review, we noticed that a significant number of literature work had studied carbon emissions trading using blockchain, and there is only limited research work done on carbon emissions tax using blockchain. As a result, the focus of this research will be on the carbon tax policy. However, for our study, we take a generic carbon tax system such as the system discussed in (Yang et al., 2014) with assumed values of carbon credit pricing and allowances, which are usually decided by the regulatory authority where the system is used.

3.1.4 Carbon Labelling

During our literature review, we came across the notion of using carbon labeling to identify every product and store its carbon footprint on the blockchain using the identifier on the label. In the poster (Pellen-Pickersgill et al., 2019), the authors have conceptualized a product-level carbon footprint tracking using the bar code identifier, which they call the carbon label. Since we are performing a product-level carbon footprint tracing in our system, it is critical that we use an identifier to map the physical product on the blockchain. For this, we use the Decentralized Identifier (DID), which was introduced earlier in Section 2.1.12. We can use the DID as the identifier, generate a QR code of the DID identifier, and attach it to the product. This

will help to track the product across the supply chain. We can use the carbon labeling for the car parts produced by the tier-1 suppliers and the car assembled by the OEM.

3.1.5 Assumptions in the Supply Chain considered for the study

Since the automobile supply chain is a complex network of supply chain participants. It is critical to define the scope and formalize assumptions about the supply chain that is considered for this study. Following are the list of defined scope and assumptions :

1. The life cycle stages of a car considered in this study are limited to the upstream activities and production activities. Specifically, we consider the component production and car production/assembly stages.
2. Among the suppliers involved in upstream phase, only the tier 1 suppliers are considered for the study. There can be 'n' number of suppliers.
3. Every tier 1 supplier is assumed to provide a component (car part) attached with a carbon label(QR code) that also contains a unique DID Identifier.
4. The component produced by tier 1 supplier is assumed to be assembly part which can directly be assembled to the car.
5. Many manufacturing units (managed by a single OEM) are assumed to be present in the study.
6. OEMs are assumed to be an assembly unit or integrator that assembles the parts received from several hundreds of suppliers.
7. OEMs are assumed to produce the end product which is the car.
8. The cars assembled by the OEMs are also attached with a carbon label(QR code) that also contains a unique DID Identifier.
9. The supply chain is assumed to be ideal with no constraints in the availability of parts, no lack of orders, capacity limitations.
10. The demands of the participants in the supply chain are assumed to be fulfilled.
11. Emissions during transportation between the outlets are not considered in this study.
12. The emission rate is not fixed, and it is constantly monitored during manufacturing operations at facilities of both suppliers and OEM.
13. The carbon pricing scheme used in this study will be the carbon tax policy. It is assumed that the carbon price is fixed and there are no price variations in different levels of taxation.

3.1.6 Application Scenerio

In this section, we will present a scenario in which the system might be used in order to provide a better understanding of the context and design considerations of the proposed solution .

The system will be used in the supply chain of an automobile (car) manufacturer. Consider a supply chain with ‘n’ suppliers and ‘n’ OEMs. The facilities of both suppliers and OEMs are equipped with sensors to monitor both direct emissions and indirect emissions. These two facilities coordinate to perform two production tasks. Tier 2 or tier 3 suppliers provide the raw materials or intermediate components to the tier 1 suppliers. The first production task that is performed is the production of components or car parts by tier 1 suppliers. After the component is produced, the total carbon footprint (including direct and indirect emissions) used for producing the component is recorded. The component is attached with a QR-code that can be used to track the recorded carbon emission data. Then the product is transported to the OEM. Once the OEM receives components from all its suppliers, it starts its assembly activity. The OEM performs the second production task to assemble and produce a car from the received components. Following the production of the car, the total carbon footprint (including direct and indirect emissions) is recorded. The car is attached with a QR-code that can be used to track the carbon emission of all the components attached in the car and the total carbon emission emitted. Finally, the finished cars are delivered to facility of the distributors.

3.2 Proposed System Overview

3.2.1 System Objectives

The proposed system’s objectives are based on the challenges identified in the automotive supply chain during the case background study as well as the issues identified with the currently implemented solutions during the literature review :

- The first objective of the system is to utilize the blockchain as an infrastructure to support recording carbon emissions reliably at the facilities of both the suppliers and OEM.
- The second objective of the system is to provide a blockchain-based digital identity to the manufactured parts and produced cars which will be attached to them as a QR code.
- The third objective of the system is that using the blockchain-based digital identity, the records of the carbon footprint of the component and the car should be traceable.
- The fourth objective of the system is that the system implements a price on carbon which can be paid using the native cryptocurrency.

3 Concept Description

- The fifth objective of the system is that a user interface should be built to interact with the system and access data on the blockchain.
- The sixth objective of the system is that it should only be accessible to the registered participants. The participants should be able to view the transactions on the blockchain according to their permissions.

3.2.2 System Actors

The four main actors identified and their role in the system are given below:

- **Suppliers:** The companies that produces the components of a car. Only the tier 1 suppliers are actors in this system.
- **OEMs:** The companies that produces the car from the supplied components.
- **Manager:** Issue the annual allowance of carbon credits to the supply chain network participants. They also mint token allowances. They control the supply chain and keep track of the carbon emission data. They exert pressure on the supply chain participants to reduce their carbon footprint. They also add or remove other managers.
- **Admin:** The admin is a manager and has all the roles and abilities of the manager. The admin is the only manager in the beginning when the system is started.

3.2.3 Top-Level Design of the System

Our system is built to work alongside the existing automotive supply chain. There are mainly three primary layers of the system.

- **Physical layer:** This layer records the production, movement of the components from suppliers and the assembly of components in the OEMs and record the carbon emissions in all these activities.
- **Web Application:** It is the application used by different actors to interact with the system to store and retrieve information on the blockchain. It consists of a front-end application and a blockchain client application.
- **Blockchain:** The system uses blockchain to store and manage the data generated by the automotive supply chain participants like suppliers and OEMs and also track their carbon tax payments.

Figure 3.4 shows a top-level diagram of the system layers and the flow of data.

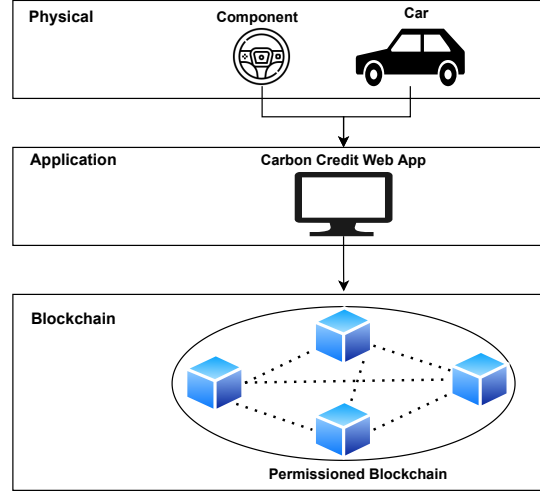


Figure 3.4: Top-level design of the envisioned system.

3.3 Requirements of the System

This section attempts to answer the second part of the first research question i.e. the requirements of the system. Requirements are derived from the objectives defined in Section 3.2.1:

Requirement 1: The designed system should be able to reliably record the carbon emission data from the facilities of OEM and suppliers on the blockchain.

The carbon emission data should be recorded by suppliers after producing their components and the carbon emission data should be recorded by the OEM after producing the car from the components. By enabling the suppliers and OEM to record the carbon emission data on the system reliably, the system can solve the problems like inconsistent and unavailability of carbon emission data that was discussed in the motivation Section 1.1.

Requirement 2: The designed system should be able to assign a unique blockchain-based digital identity to the manufactured parts and cars at facilities of OEM and suppliers.

The use of blockchain-based digital identity to be assigned to the manufactured parts and cars will allow their identities to be verified on the blockchain.

Requirement 3: The carbon footprint of the components and cars should be traceable in the designed system using the blockchain-based digital identity (DID).

The carbon footprint traceability using the blockchain-based digital identity (DID) ensures that the carbon footprint can be traced across the product life cycle of the supply chain from component production by suppliers to car assembly by the OEM. This also helps to get an estimated carbon footprint of the car, considering only the

3 Concept Description

production phase.

Requirement 4: The designed system should perform carbon pricing (Carbon tax) by implementing a representation of carbon credit allowance on the blockchain that can be transferred and surrendered.

The representation of carbon credit allowance acts a right to pollute that is issued by an issuer(in this study it is issued by managers). So they represent an amount of pollution that the polluters can legally emit. Every time the suppliers or OEM enters their carbon emissions on the blockchain, the carbon credit allowance is reduced based on the emissions. Finally, the unused tokens can be surrendered to get some incentives. In a typical carbon taxation system, the allocated carbon credits cannot be surrendered. However in our study based on the findings of (Yang et al., 2014), the carbon credit can be surrendered for getting incentives because it motivates the firms with strong ability to reduce carbon emissions to actually reduce the carbon emissions.

Requirement 5: The annual cap on carbon credit allowance offered yearly are decided based on the regulations and can be changed.

The annual cap on carbon credit can be decided by the regulatory authority in case of a carbon tax. This allowance can be changed every year based on the decisions of the regulatory authority or the OEM managers.

Requirement 6: The price of the carbon credits are fixed, and the payments have to be made using the native cryptocurrency of the blockchain.

The prices of carbon credits are fixed according to the regulations and the payment has to be made using the cryptocurrency of the blockchain. So once the carbon credit allowance is depleted, the suppliers and OEM can pay the price of carbon credit allowance and request for more credits. The price of carbon does not vary according to the taxation levels. Also according to (Yang et al., 2014) a fixed carbon price benefits the supply chain profits.

Requirement 7: The designed system should only be accessible to the registered participants. The participants should be able to view the transactions on the blockchain according to their permissions.

Since carbon footprint data are highly confidential for the suppliers and OEM, it cannot be released in the public domain. So the blockchain implementation that supports only permissioned access to data on the blockchain should be used. Participants like OEMs or suppliers should not be able to access the carbon footprint details of the other participants.

All the requirement of the designed system discussed in this section is summarised in the Table 3.1.

Sl No.	Requirements
R1	The designed system should be able to reliably record the carbon emissions data from the facilities of OEM and Suppliers on the blockchain.
R2	The designed system will assign a unique blockchain-based digital identity(DID) to the manufactured parts and cars at facilities of OEM and Suppliers.
R3	Carbon footprint of the components and cars should be traceable in the designed system using the blockchain-based digital identity(DID).
R4	The designed system should perform carbon pricing (Carbon tax) by implementing a representation of carbon credit allowance on the blockchain that can be transferred and surrendered.
R5	The annual cap on carbon credit allowance offered yearly are decided based on the regulations and can be changed.
R6	Price of the carbon credits are fixed and the payments have to be made using the native cryptocurrency of the blockchain.
R7	The designed system should only be accessible to the registered participants. The participants should be able to view the transactions on the blockchain according to their permissions.

Table 3.1: Summary of the requirements of the designed system

3.4 Use Cases

This section attempts to define the use cases of the system derived from the requirements of the system in the previous Section 3.3. Listing the use cases helps to define the core feature of the system. :

Use case 1: Suppliers and OEMs need to be approved by the manager to participate in the network.

The use case begins when the suppliers and OEMs sign up by putting their details such as the name of the company, location, their blockchain wallet address. Once they have submitted these details on the blockchain, they wait until the managers or the admin approve their participation.

Use case 2: The admin or managers can approve the suppliers or OEMs to participant in the system.

The use case begins when the suppliers or OEMs have already signed up, and the managers or the admin approves their participation by supplying them with an initial yearly carbon credit allowance.

Use case 3: The admin or managers or admin can ban a user or refuse participation in the system.

If a user who signs up is not part of the supply chain, he may be refused participation or banned permanently from the system by suppliers or OEMs.

3 Concept Description

Use case 4: The admin or managers can allocate the initial supply of carbon credits allowance every year to the participant suppliers and OEMs accounts. This allocation is based on the carbon tax regulations.

The admin or managers decide on the amount of carbon credits allowance to be given to suppliers or OEMs at the beginning of the year based on regulation and previous year usages statistics.

Use case 5: OEMs and suppliers should be able record their carbon emission data to the system during the production of components or assembly of the car. Once they enter the carbon emission data, an equivalent amount of the carbon credit allowance is debited.

The use case begins when the OEMs and suppliers have already received the carbon credit allowance for a year. As they produce their products, they emit carbon emissions. When they record their product carbon footprint, it is essential that their carbon credit allowance equivalent to the emitted emissions are debited. The carbon credits allowance can be debited from their wallets as long as there is enough carbon credit allowance. If the carbon credit allowance is finished, they have to stop their production activities.

Use case 6: When the OEMs and suppliers complete their production activities, the produced or assembled product should be given a unique blockchain-based identity(DID).

The use case begins when the suppliers produce the component or when the OEM assembles the car. Whenever these activities are successfully occurring, an entry with the carbon emissions to complete this activity is recorded on the system. The system should assign these components or assembled cars a unique blockchain-based digital identity (DID) that can be verified on the system for authenticity.

Use case 7: The unique blockchain-based digital identity (DID) assigned to components or assembled car could be used to trace their carbon footprint in the system.

The unique blockchain-based digital identity (DID) assigned to the products produced by suppliers and OEMs could be used to trace the carbon footprint on the system. In the case of a component produced by a supplier, the DID could be used to get the exact carbon emissions emitted during the production activities by the supplier. In the case of an assembled car by an OEM, the DID could be used to get the exact carbon emissions emitted during the assembly activities and also the carbon emissions by all the components attached in the car. So it is possible to trace the complete carbon footprint during the production of the car.

Use case 8: The unique DID generated by the system should be converted into a QR-code which can be later attached by the suppliers or OEM on to the product.

The unique DID generated by the system should be converted into a QR-code such that it can be later attached by the suppliers or OEM on to the product. This helps to map the produced component or car with its digital record on the blockchain.

Use case 9: When the carbon credit allowance in the accounts of suppliers and OEMs are depleted, they send a buy order to the system along with an amount equal to the price in ether.

The use case begins when the carbon credit allowance of suppliers or OEMs has gone low or finished. Then they place a buy order and send the equivalent price in ether for more carbon credit allowance to be issued to them.

Use case 10: The designed system automatically sends the carbon credit allowance equivalent to the ether amount sent.

The use case begins when the suppliers or OEMs have already sent a buy order for the issue of more carbon credit allowance. The system automatically sends a carbon credit allowance based on the price set by the admin or suppliers. This is based on the crowd sale or initial token offering concept of blockchain.

Use case 11: The price of carbon credit allowance is fixed based on the regulation of carbon taxations set by the regulatory authority.

The system has a fixed carbon credit pricing based on the regulations of carbon taxation. This also ensures that hoarding the carbon credit allowance for using it later does not provide advantage to the supply chain participant.

Use case 12: The unused carbon credit allowance can be submitted back to the system at the end of the year by the suppliers or OEM and receive an the price of the token in return.

The use case begins when the suppliers or OEMs have some carbon credit allowance remaining in their wallet that goes unused at the end of the year. To avoid this the suppliers or OEMs can submit their excess tokens to the system and receive the price of the token in native cryptocurrency in return.

Use case 13: When the unused carbon credit allowance is submitted back by suppliers and OEMs to the system at the end of the year, the price of the carbon credit allowance is sent to them in return by the system automatically.

The use case begins when the suppliers or OEMs submit their carbon credit allowance at the end of the year. The system automatically sends them the price of the carbon credit allowance in return.

Use case 14: The suppliers and OEMs can view the transactions recorded by them on the blockchain.

The suppliers and OEMs can view the list of transactions recorded by them on a dashboard.

Use case 15: The manager can view and monitor the activities performed by all blockchain participants.

3 Concept Description

The manager can monitor the list of transactions recorded by all the participants of the system on a dashboard.

Use case 16: Managers can be added or removed as participants in the system.

The admin can add or remove managers whenever required.

All the use cases of the designed system discussed in this section is summarized in the Table 3.2.

Sl No.	Use Cases
UC1	Suppliers and OEMs need to be approved by the manager to participate in the network.
UC2	The admin or managers can approve the suppliers or OEMs to participant in the system.
UC3	The admin or managers or admin can ban a user or refuse participation to the system.
UC4	The admin or managers can allocate the initial supply of carbon credits allowance every year to the participant suppliers and OEMs accounts. This allocation is based on the carbon tax regulations.
UC5	OEMs and suppliers should be able record their carbon emission data to the system during the production of components or assembly of the car. Once they enter the carbon emission data, an equivalent amount of the carbon credit allowance is debited.
UC6	When the OEMs and suppliers complete their production activities, the produced or assembled product should be given a unique blockchain-based digital identity (DID)
UC7	The unique DID assigned to components or assembled car could be used to trace their carbon footprint in the system.
UC8	The unique DID generated by the system should be converted into a QR-code which can be later attached by the suppliers or OEM on to the product.
UC9	When the carbon credit allowance in the accounts of suppliers and OEMs are depleted, they send a buy order to the system along with an amount equal to the price in ether.
UC10	The designed system automatically sends the carbon credit allowance equivalent to the ether amount sent.
UC11	The price of carbon credit allowance is fixed based on the regulation of carbon taxations set by the regulatory authority
UC12	The unused carbon credit allowance can be submitted back to the system at the end of the year by the suppliers or OEM and receive an the price of the token in return.
UC13	When the unused carbon credit allowance is submitted back by suppliers and OEMs to the system at the end of the year, the price of the carbon credit allowance is sent to them in return by the system automatically.
UC14	The suppliers and OEMs can view the transactions recorded by them on the blockchain.
UC15	The manager can view and monitor the activities performed by all blockchain participants.
UC16	Managers can be added or removed as participants in the system.

Table 3.2: Summary of the use cases of the designed system

3.5 System Architecture

Software architectures are concerned with the design and implementation of the high-level structure of the software (Kruchten, 1995). There are numerous perspectives to software systems. A single view of the architecture does not always adequately capture the gist of the software system. The use of multiple architecture views in software engineering aids in addressing the concerns of various stakeholders of the system. The “4+1” view model of software architecture developed by Philippe Kruchten in his paper (Kruchten, 1995) defined a suitable methodology to capture the description of software architecture into multiple concurrent views. We will use the “4+1” view model to describe the architecture of the designed system in this master thesis.

The “4+1” view model gives four different views that organize the description and perspective of software systems and these four views are integrated together with a final fifth view:

- **Logical view:** The logical view decomposes the application into functional requirements of the architecture.
- **Process view:** The process view concentrates on the execution time of system components, their behaviors, communication, synchronization, and interactions. Non-functional requirements such as performance and availability are also taken into account. It mainly decomposes the software system in terms of groups of tasks that can form executable units called processes.
- **Development view:** The development view focuses on the organization of the software components and the modules in the development environment.
- **Physical view:** The physical view describes a relationship between the software components and the physical hardware they are running on. It takes into account the non-functional aspects of the system such as availability, reliability, performance and scalability.
- **Scenario view:** The scenario view captures stakeholder perspective based on use cases and complements all other views.

3.5.1 Logical view

As previously stated, the logical view divides the application into functional requirements. In the case of data-driven applications, (Kruchten, 1995) recommends using E-R diagrams. Since our system is a data-driven system that employs smart contracts, we can use the E-R diagram to depict the logical view of the system. Figure 3.5 depicts an E-R diagram of the envisioned system based on the use cases and requirements identified. The entities are represented in the diagram by rectangles, and diamonds represent the relationship between the entities. The entities include all the actors involved in the system and the resources involved in the system like carbon credit, the component of a car, the assembled car etc.

3 Concept Description

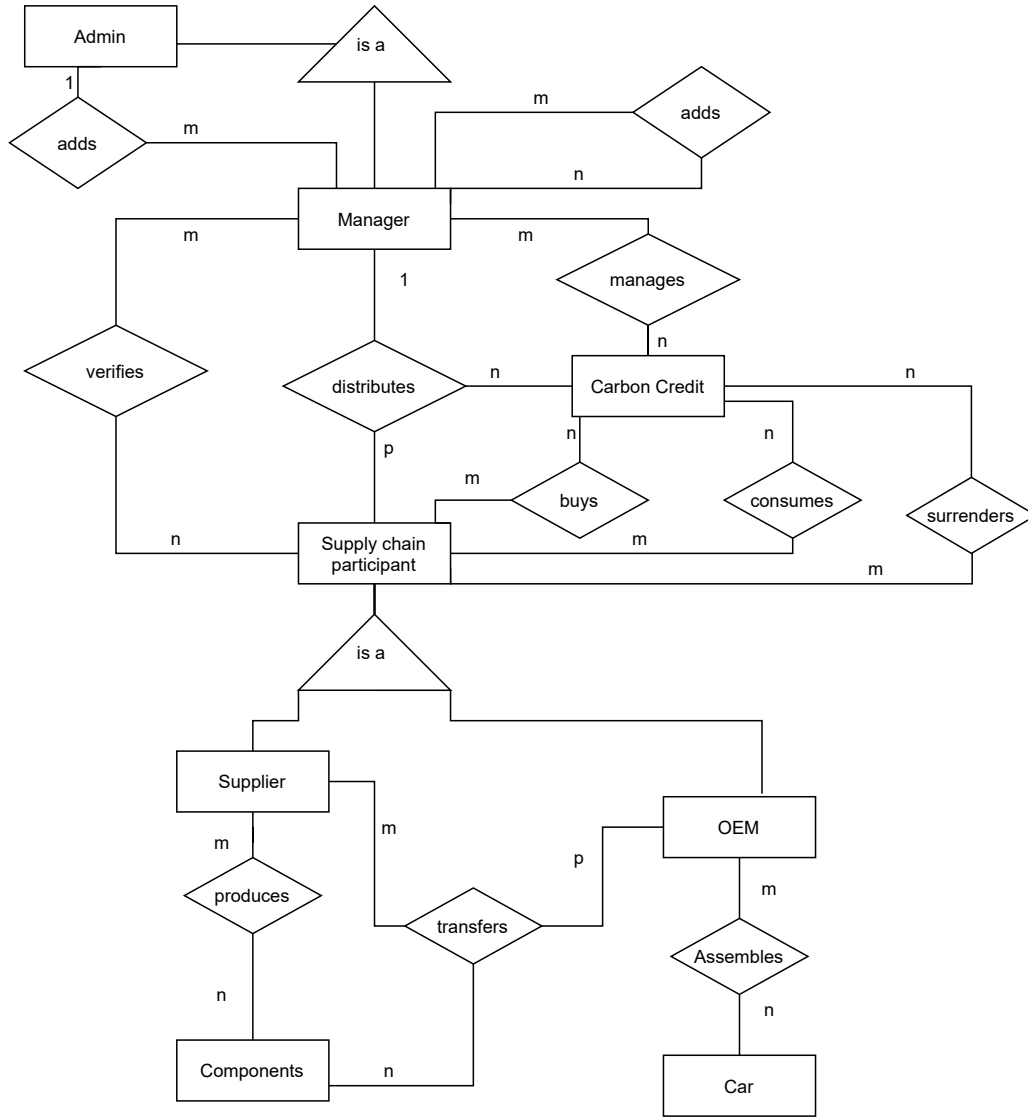


Figure 3.5: Entity relationship diagram of the designed system

3.5.2 Process view

As previously stated, the process view captures the flow of a process across several modules, physical layers, and entities, their behavior, task concurrency, information flow, and the run time behavior of the system. UML diagrams used to represent process view are sequence diagram, communication diagram, and activity diagram. The process view mainly divides the software system into groups of tasks. Since there can be many processes in the system, there can be several process views for a system. Figure 3.6 depicts an activity diagram of a group of tasks in the designed system. The activity diagram shows the interaction between supplier, OEM and blockchain. In this

diagram, it is assumed that only one supplier exist that produces one component and one OEM exist that uses the one component from supplier to produce a car.

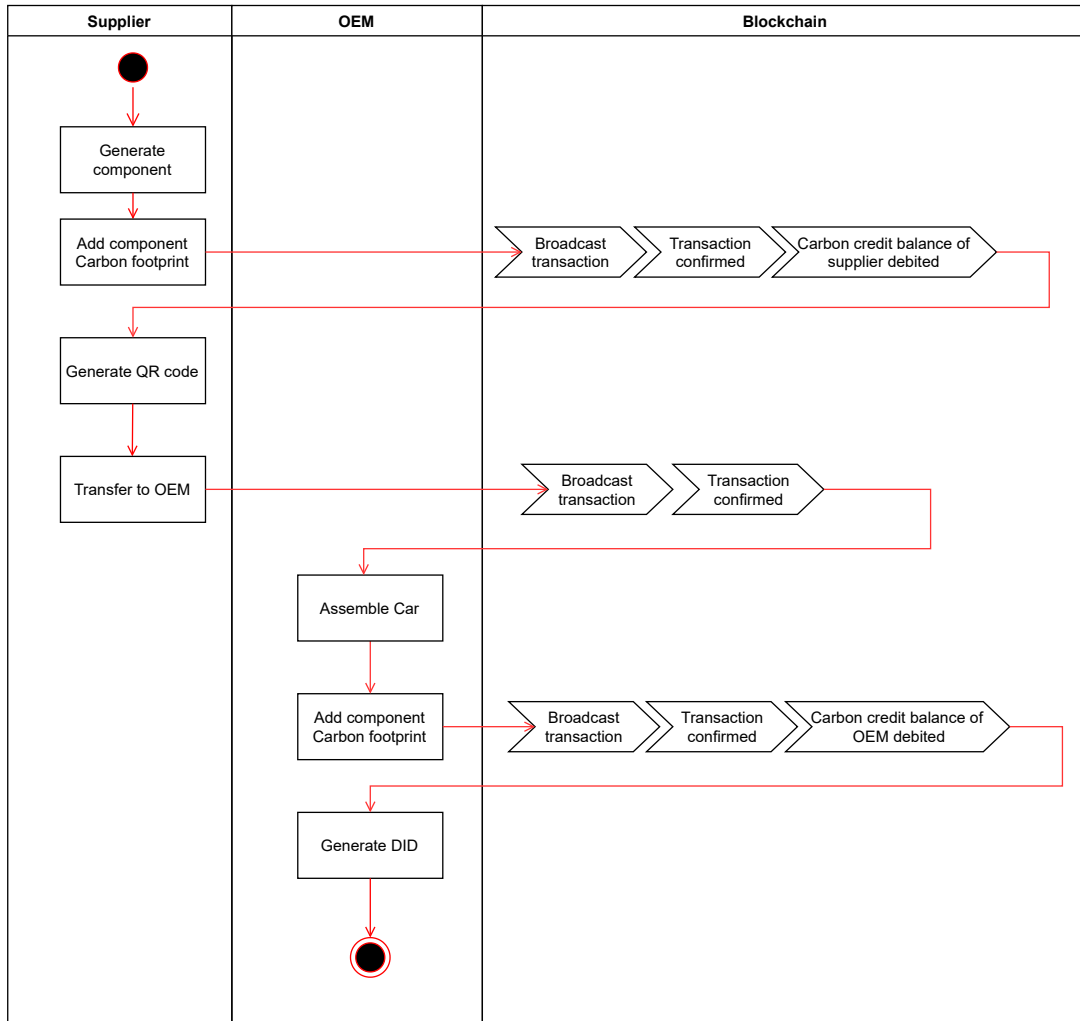


Figure 3.6: Activity diagram of one of the scenerios of the system.

3.5.3 Development view

The development view is concerned with how the software modules are organized in the development environment from the perspective of a developer. The system is decomposed into subsystems, modules and components. (Liss, 2018) discusses a structure of components which consist of modules or other components that groups use cases and requirements of the system that have been identified. The use cases and requirements identified for the designed system are grouped together as functional responsibilities. Figure 3.7 shows the use cases and requirements grouped together

3 Concept Description

based on their functional responsibilities. The requirements identified in Section 3.3 and the use cases identified in Section 3.4 are used in the diagram. Figure 3.8 shows the overview of the designed system architecture from a development view. It shows how different modules and components interact with each other.

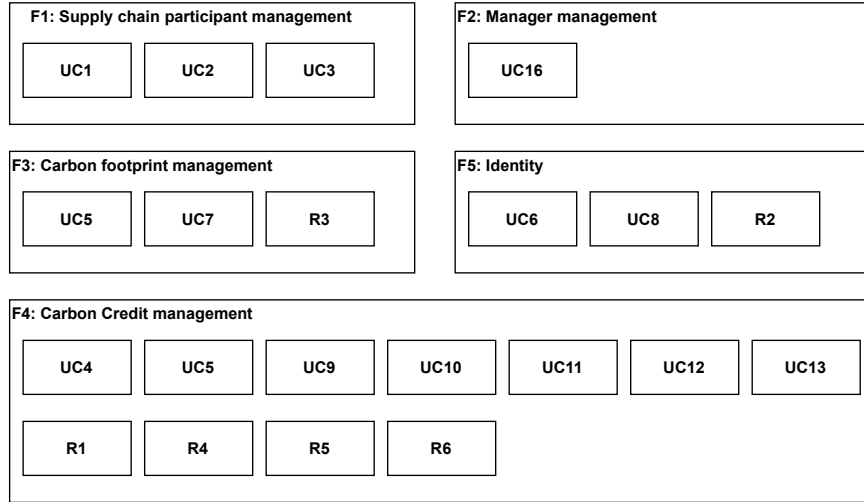


Figure 3.7: Grouping use cases and requirements according to functionalities

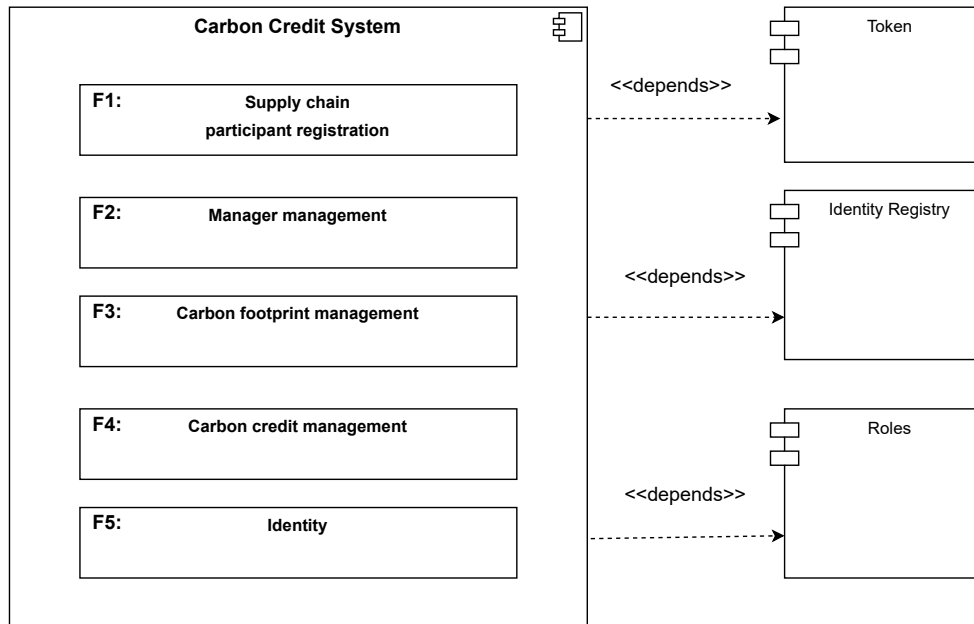


Figure 3.8: Component diagram of the designed system

3.5.4 Physical view

The physical view describes the deployment layout or the infrastructure necessary to deploy the software system. It captures a hardware mapping of different system components. Figure 3.9 shows the designed system deployed on two different physical layers. A front end running on the user machine and the smart contracts running on the blockchain.

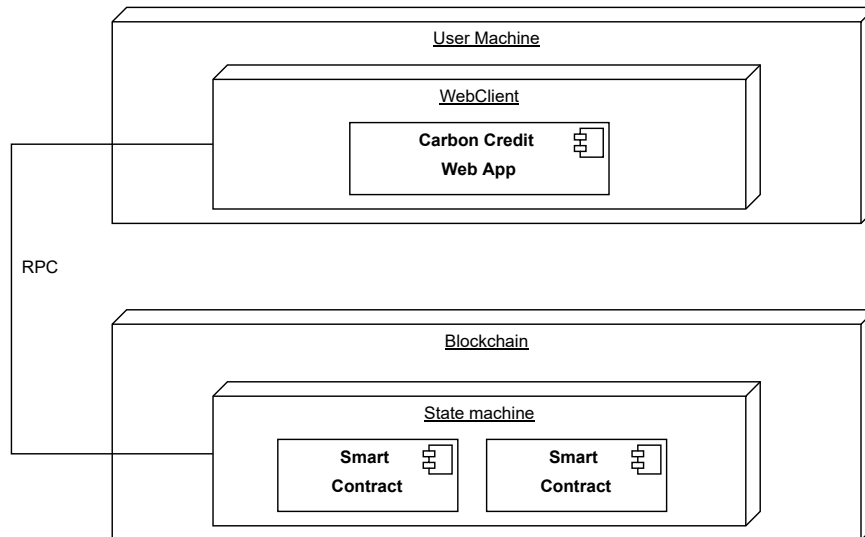


Figure 3.9: Deployment diagram of the designed system

3.5.5 Scenerio view

The scenario view is created using use cases, which describe the functionalities and processes of the system. The actors who interact with the system are used to identify use cases. We have already discussed the use cases of the system in Section 3.4. Figure 3.10 is a use case diagram built from the identified use cases. The stick figures in the diagram represent the different actors in the system. The identified use cases of the system are written into oval-shaped circles. The box that envelopes the use cases is called a system boundary box. The associations are the lines between actors and use cases.

3 Concept Description

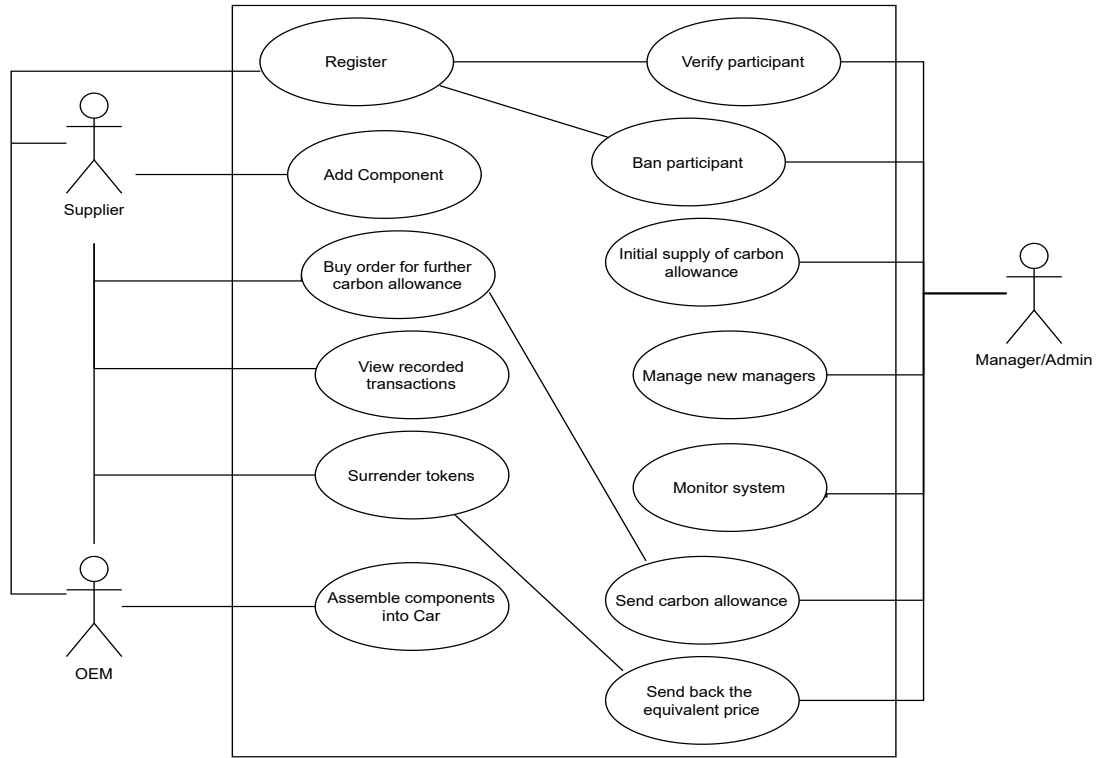


Figure 3.10: Use case diagram of the designed system

3.6 Summary

This chapter covered the conceptual description of the proposed unified blockchain-based carbon product footprint tracing and carbon tax payment into a single system for the automotive supply chain (which is a multi-echelon supply chain) where the products are tracked using a carbon label (QR code with a digital identity called DID). The chapter starts with a review of the case background to provide the groundwork for conceptual design. The case background section discussed carbon footprint tracking techniques in the automotive supply chain, as well as carbon pricing strategies. The following section establishes the baseline assumptions of the proposed system, followed by a description of a scenario in which the system may be deployed. The requirements, use cases, and design of the system were thoroughly described in the following sections. The primary contribution of this chapter was that it presented abstract modeling and architecture of the different parts of the system based on the “4+1” view model by (Kruchten, 1995). This chapter also addressed research question 2 (RQ 2) by presenting the architecture design of the system. It also addressed the second half of research question 1 (RQ 1) by concretely defining the requirements of the system. It also provided some requirements and use cases based on which the applicability of the implemented system can be examined, therefore answering the research question 4 (RQ4).

4 Implementation

In this chapter, we discuss the design decisions and technical details for implementing the software solution hypothesized in the previous chapter. This chapter primarily addresses research question 3 by describing the software frameworks, tools, and methodologies that may be utilized to construct the envisioned system.

4.1 Implementation Decisions

4.1.1 Blockchain

Since there are so many different blockchain implementations to choose from, one of the most crucial decisions before implementing the application is to choose which blockchain is appropriate for our project. Firstly, we evaluate which type of blockchain is suitable for our application based on the use cases, requirements, and objectives. The major types of blockchain are discussed in Section 2.1.9. To select the type of blockchain suitable for our application we rely on the decision tree provided in Figure 4.1 adapted from (Wüst and Gervais, 2018).

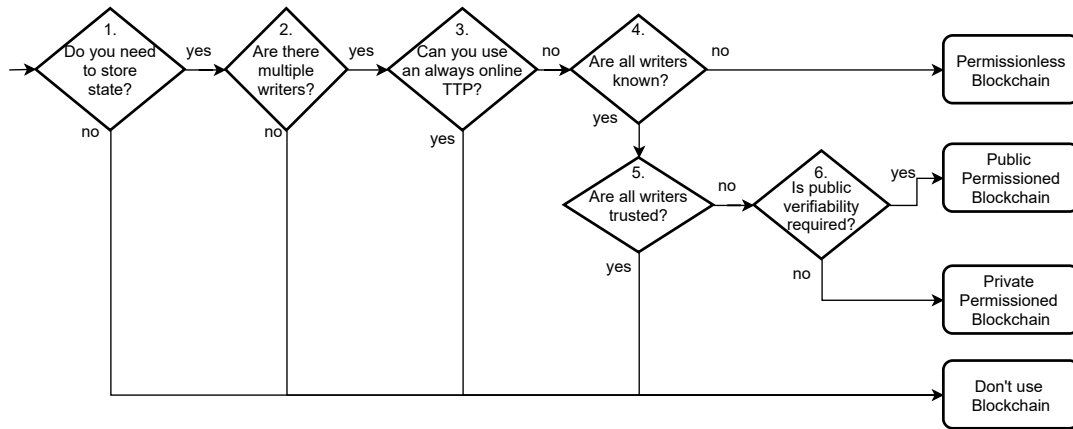


Figure 4.1: Decision tree to select the type of blockchain adapted from (Wüst and Gervais, 2018)

The first question asked in the decision tree is whether the application requires the state of the system to be stored. As discussed in the motivation section, the application must reliably store data, especially carbon emission data. It is also required to keep the data such that the managers and participants of the system can monitor their carbon footprint and the carbon tax payments. So, we follow the decision path yes

4 Implementation

and reach the second question in the decision tree. The second question is if multiple actors write data into the system. As discussed in Section 3.2.2 multiple actors write and read data to or from the system. So we follow the decision path “yes” for the second question. The third question in the decision tree is if an online trusted third party(TTP) could be used as a certificate authority in the system. As we already discussed in the motivation section, a third party cannot be trusted with the control and confidentiality of the data. So we follow the decision path “no” for the third question. The fourth question in the decision tree is if all the writers who write data on the system are known. In our system, as per the requirement R7, all the writers have to be first validated before they can access the system. So, we follow the decision path “yes” to find another decision question. The fifth question is if all the writers can be trusted. As we know, the carbon footprint and carbon tax payments are very sensitive data, and it is in the interest of the participants to manipulate or misrepresent the data. So, the writers of the system cannot be trusted. The sixth and final decision question is if the data on the system should be publicly verifiable. Since the data is very sensitive, it is not in the interest of the suppliers and OEMs to release their data to the public for verification. Finally, from the decision tree, we conclude that a private-permissioned blockchain will be suitable for the proposed system.

SL No.	Blockchain	Private/Public	Permissioned	Smart Contract	Native Currency	Selected for comparision
1.	Private Ethereum ¹	Private	Permissioned	Yes	Yes	✓
2.	Hyperledger Fabric ²	Private	Permissioned	Yes	No	
3.	Ripple ³	Public	Permission-less	No	Yes	
4.	Openchain ⁴	Private	Permissioned	No	No	
5.	Big Chain DB ⁵	Private/Public	Permission-less	No	No	
6.	Bitcoin ⁶	Public	Permission-less	No	Yes	
7.	Diem (Libra) ⁷	Private	Permissioned	Yes	Yes	✓
8.	Quoroum ⁸	Private	Permissioned	Yes	No	

Table 4.1: review of various Blockchain implementations

Further, we review few blockchain implementations based on their features to select

¹Go Ethereum: <https://geth.ethereum.org>

²Hyperledger Fabric: <https://www.hyperledger.org/>

³Ripple: <https://ripple.com/>

⁴Openchain: <https://www.openchain.org/>

⁵BigchainDB: <https://www.bigchaindb.com/>

⁶Bitcoin: <https://bitcoin.org/en/>

⁷Diem: <https://www.diem.com/en-us/>

⁸Quoroum: <https://consensys.net/quorum/>

them for implementing our proposed application. We compare the major blockchain-based on whether it is private or public, has permissioned access, supports smart contracts, and supports a native currency. To achieve the objectives of our application, it is essential that the blockchain supports the smart contract and uses a native currency. Based on the results of the review in Table 4.1, two possible candidate blockchains satisfy all the conditions needed for our application. The two blockchains are Private Ethereum and Diem (Libra). We compare these two candidate blockchains in Table 4.2 based on their features that are found in the white papers of both the blockchain projects(Buterin, 2014)(Wood, 2014)(Diem, 2020)(Warski, 2019).

Sl No.	Features	Diem (earlier Libra)	Private Ethereum
1.	Type of blockchain	Permissioned blockchain	Permissioned blockchain
2.	Data model	Account based model	Account based model
3.	Smart contracts	Supports smart contracts	Supports smart contracts
4.	Data structure	Core data structure are transactions	Core data structure are blocks
5.	Consensus	Proof of authority (less energy intensive)	Proof of work(High energy intensive)
6.	Gas cost	Lower gas costs	Higher gas costs
7.	Transactions per second	More transactions per sec	Lower transactions per sec
8.	Native cryptocurrency	Diem coin is stable currency	Ether coins fluctuate in their price
9.	Smart Contract Language	Uses Move (functional programming language)	Uses Solidity (object oriented language)
10.	Wallet	Calibra is the wallet	Metamask is the most popular wallet/private key manager
10.	Blockchain implementations	Only test net is available yet	Main net, various implementations and test nets are available
11.	Resources and documentations	Documentation and resources are limited	Documentations and resources are abundantly available

Table 4.2: Comparison between Diem blockchain and Private Ethereum blockchain based on major features

According to our analysis, after comparing the two candidate blockchains, we have reached the following conclusions. Diem is beneficial when considering gas costs, the number of transactions per second, and a less energy-intensive consensus mechanism. Also, since the Diem association plans to release the Diem coins as a stable coin, it is easier to map the carbon tax payments in Diem coins to recognized fiat money (such

4 Implementation

as US dollars) in our application. There are also some reasons to favor the private Ethereum blockchain. The smart contract language used in private Ethereum is solidity is based on the object-oriented paradigm, which is very familiar to the developer community. At the same time, Diem uses Move, which is based on a functional programming paradigm that is not very well known. Metamask wallet manager (private key manager) used in Ethereum is very robust and relatively superior to Calibra in Diem. Currently, Diem provides only a test network to deploy the application. Private Ethereum has various implementations and test networks that could be used to deploy the application. Finally, plenty of resources, well-maintained documentation, and community support are much better in the case of Ethereum. After evaluating all of the reasons in favor of the two blockchain implementations presented in this paragraph, the decision was made to develop the proposed system on a private Ethereum platform. It should be noted that Diem would have also been a good fit for the proposed system.

4.1.2 Back-end

Choosing which data will be saved on-chain (or on the blockchain) and which will be stored off-chain in a back-end data store is another decision to make during the implementation. Since the managers only validate the suppliers and OEMs known to them, they do not need to store a lot of information about the users on the system. Also, the system only considers the tier 1 suppliers which are quite smaller in number compared to the complete supplier network. So, a decision was taken to store all the information on-chain (on the blockchain).

4.1.3 Front-end

The front-end plays a vital role by providing an interface for the actors to interact with the system. The front-end will display the data on the blockchain and help the actors write data to the blockchain. In the proposed work, a decision was made to use React to create the user-interface because it is most commonly used in developing Dapps with Ethereum. React is a JavaScript framework developed by Facebook. The communication between the front-end and the Ethereum nodes will utilize the Web3.js framework.

4.2 Implementation Concepts

4.2.1 Ethereum Blockchain

Vitalik Buterin began his work on the first edition of the Ethereum platform in his white paper in November of 2013. Ethereum is an open-source, decentralized computing platform that uses the blockchain to synchronize and store the changes that occur in the state of the system (Gallersdörfer et al., 2020).

In the Ethereum universe, there is a single prime computer known as Ethereum World Computer. Furthermore, everyone participating in the Ethereum network agrees on the state of this machine. Every Ethereum network participant has a copy of the state of the world computer. Any network participant can initiate and broadcast a request for the computer to perform arbitrary computation. Whenever such a request is initiated, other participants of the Ethereum network verify, validate and execute the computation. Such requests change the state of EVM, and this state change is committed and propagated throughout the entire network. The requests are called transactions; participants in the network are called nodes(Wood and Antonopoulos, 2018). Basically, the Ethereum blockchain is a transaction-based state machine. Since it is a computing platform, it also has the capability to run programs called smart contracts(discussed in Section 2.1.10). The platform has also introduced a cryptocurrency called Ether to track and limit the costs of computation resources. Ethereum currently uses the Proof-of-Work algorithm called Ethash for all the nodes to agree on the state of the machine (also called consensus mechanism).

Ethereum Virtual machine

Ethereum is a deterministic state machine, and it has a virtual machine called the Ethereum Virtual Machine (EVM) that specifies the execution model for state changes in the blockchain(Wood and Antonopoulos, 2018). The state of the blockchain includes all Ethereum accounts, all smart contracts, and transactions. Basically, the EVM handles the smart contract deployment and execution of the smart contract bytecode. The EVM is a global singleton, which means it functions as if it were a single-instance, global computer that runs everywhere(Wood and Antonopoulos, 2018). Every node in the network runs a local copy of the EVM.

Accounts and Ethereum Addresses

Ethereum has two types of accounts. The first type of account is called the externally owned accounts (EOA). In addition to sending and receiving ether, EOAs can deploy smart contracts and trigger them. EOAs are also used to sign transactions sent by the user using the associated private key of EOA. The second type of account is the contract accounts. The contract accounts have smart contracts associated with them. Every smart contract deployed in the Ethereum network gets an account with a unique Ethereum address owned by the logic of the smart contract. The contract account does not have a private key that controls it(Wood and Antonopoulos, 2018). Ethereum addresses are hexadecimal numbers, unique identifiers, and derived from the last 20 bytes of the Keccak-256 one-way hash function of the public key of a EOA. In the case of a contract account, the address is derived from Keccak-256 one-way hash function of the address of the contract deployer and the number of transactions sent from that address (also called nonce)(Wood and Antonopoulos, 2018).

Smart Contracts

A detailed description of smart contracts was given in Section 2.1.10. The smart contracts in Ethereum are written in Solidity language and are stored as files with file

4 Implementation

extensions .sol. The Solidity compiler takes high-level Solidity code as input and produces low-level EVM bytecode. The bytecode is sent as a transaction on the blockchain network. After the transaction is mined, the smart contract is usable, and a contract address is created(Wood and Antonopoulos, 2018).

Clients

An Ethereum client is a software implementation of an Ethereum node based on the specifications and communicates with other Ethereum nodes. Since Ethereum is open-source and has defined reference standards and communication protocols, there are different implementations of Ethereum clients, and they can interoperate with each other on the network. There are also several Ethereum Networks such as Ethereum, Ethereum Classic, Ella, Expanse, Ubiq, Musicoin(Wood and Antonopoulos, 2018). There are mainly six implementations of Ethereum protocol written in different languages:

- Go-Ethereum (written in Go)
- Parity (written in Rust)
- cpp-ethereum (written in C++)
- pyethereum (written in python)
- Mantis (written in Scala)
- Harmony (written in Java)

Wallets

Wallets in Ethereum serve as a primary user interface to the Ethereum network, managing keys, addresses, controlling access, tracking account balances, and signing transactions(Wood and Antonopoulos, 2018). The Ethereum wallets hold the private key and provide a user with a public Ethereum address that can be used to transfer Ether. The wallets help users send a transaction to the network by signing the transactions with the associated private keys in their wallets. A popular wallet used for Ethereum is Metamask.

Transactions

A transaction is a signed instruction that is sent from an EOA using a wallet and transmitted on the Ethereum network, and recorded on the Ethereum blockchain(Gallersdörfer et al., 2020). The transactions are the causes of change in the state of the blockchain. Transactions are also used to execute a contract in EVM. The structure of transactions in Ethereum is given below:

- A sequence number of transactions that are sent by the sender account (Nonce)
- The fee that the sender is willing to pay for each step of computation (Gas price)

- The maximum amount of gas the sender is willing to pay for the transaction. (Gas limit)
- The Ethereum address of the recipient of the message. (Recipient)
- The amount of Ether to transfer from sender to receiver (Value)
- The optional data field (Data)
- The three components of an ECDSA digital signature of the sender (v,r,s)

The transactions can be of two types. The first type of transaction is a wallet-to-wallet transaction where there is no execution of the smart contract, and it is a transfer of Ether from one wallet to another. The second type of transaction is the wallet-to-smart contract transaction, which is used to trigger the smart contract. The logic in the smart contract then decides further flow.

Transaction fee

Every operation (write or read) performed on the blockchain network is referred to as a transaction, and each transaction has a fee, which is paid in Ether and is called as Gas Cost. Gas is a unit of measurement for the amount of computational work necessary to carry out specific operations on the Ethereum virtual machine. The price of gas is measured in Ether. Usually the gas cost is measured in terms of gwei instead of ethers for convenience. Gas cost of every operation is predetermined and fixed. The smart contract code, on the other hand, can be complicated. A small piece of code can create a lot of computational work, whereas a long piece of code can generate less. This is why the Ethereum Virtual Machine expenses are subjected to the measure of work being finished. The gas is also a way in which EVM ensures the terminable nature of Ethereum World Computer i.e. when the allotted gas for an operation is consumed, and the operation gets stopped regardless of being completed or not.

Transaction Fee (paid in Ether) = Gas Limit * Gas Price

4.2.2 Decentralized Application

An application that runs on a decentralized computing system is considered a decentralized application. Even applications such as BitTorrent, Tor project are all considered decentralized applications. In the context of this thesis we consider a decentralized application (Dapp) based on smart contracts with a blockchain-based data storage that is accessible via a dedicated front-end software. The two essential properties that must hold are that the core data records of the application are stored on the blockchain, and the functions to change the core data records must be executed on the blockchain using smart contract (Wood and Antonopoulos, 2018).

The benefits of using decentralized applications usually depend on the use case or the problem it solves. But the general advantages of using Dapp are (Gallersdörfer et al., 2020):

4 Implementation

- **Trust:** All the recorded transactions with the Dapp will be stored on the blockchain forever.
- **Payment:** Payment is implemented by default, and anyone can transfer or receive ether using the wallet or Dapp. No need to use external third-party payments implementations.
- **Accounts:** Dapp makes use of the Ethereum account system. So there is no necessity to implement a user account management system for handling user data such as passwords etc.
- **Storage:** Dapp can use the blockchain as data storage which is quite expensive.
- **Resiliency:** Since the Dapp backend is completely distributed, they will have zero downtime.

4.2.3 Development Tools

- **Remix IDE:** Remix is an open-source, browser-based integrated development environment used to develop, debug and deploy smart contracts written in Solidity for Ethereum applications(Remix, 2018). During the development of smart contracts, we used Remix for easier development and testing.
- **Truffle Framework:** Truffle is a popular framework that facilitates the Ethereum smart contract development and is part of a set of tools provided by Truffle Suite(Truffle, 2018). It is an open-source framework. The framework provides tools to compile, debug, deploy and test Solidity smart contracts that help the developers to deploy their smart contracts on various Ethereum networks or test nets(Truffle, 2018). After the contracts are deployed, the framework creates smart contract abstractions to interact with the smart contracts using the front-end technologies like React. Truffle framework provides the project structure or project scaffolding for our application.
- **Visual Studio Code:** Visual Studio Code was used as the code editor to build the user interface using React.

4.2.4 Languages

- **Solidity:** Solidity is the Turing complete high-level programming language designed for writing Ethereum smart contracts(Solidity, 2014). It is syntactically similar to JavaScript and is a statically typed language. It is an object-oriented language. The contract definition is seen as similar to the class definition, and the contract instances are seen as objects similar to the traditional object-oriented programming languages like Java. Similar to the class which contains methods, the contract definition consists of functions. Contracts also have data and behavior just like objects, modeled by variables and functions defined in the

definition. Solidity also supports inheritance such that a contract definition can inherit from other contract definitions. It also supports complex user-defined types. The contracts written in solidity are finally compiled to bytecode(Wohrer and Zdun, 2018). Once it is compiled and deployed to the blockchain, it is immutable. We used Solidity language to write our smart contract logic.

- **JavaScript:** Javascript is a high-level programming language that is one of the World Wide Web's fundamental technologies(Javascript, 2008). It is utilized in our application for client-side scripting, particularly when utilizing third-party frameworks such as React, Web3.js, and others.
- **JSX:** JSX is an embeddable XML-like syntax(JSX, 2013). It is transformed into valid JavaScript. However, the semantics of that transformation is implementation-specific. JSX is used with React to specify how the user interface should look like.

4.2.5 Blockchain

- **Go-Ethereum:** Go-Ethereum, also known as Geth, is the official Ethereum implementation developed by Ethereum Foundation(Go-Ethereum, 2016). Geth is implemented using the language Go. Geth is standalone client software that connects with the other nodes in a blockchain network. Geth comes with a JavaScript console and a JSON RPC server. It is possible to create a private blockchain using Geth.
- **Ganache:** Ganache is an open-source lightweight local blockchain for Ethereum smart contract development. It is used to deploy, simulate and test smart contracts(Ganache, 2018). Ganache has two versions, a GUI version, and a CLI version. Ganache is also part of the set of tools provided by Truffle Suite for Ethereum development. So it can be easily linked with Truffle projects to automate tests for smart contracts. The GUI version has a developer-friendly block explorer and additional debugging features. Instead of connecting to Go-Ethereum for deployment and testing, the ganache GUI provides a faster and cleaner version of a real Ethereum blockchain during the development phase.
- **MetaMask:** Metamask is a browser plugin that acts as a cryptocurrency wallet and allows users to interact with any Ethereum network(Metamask, 2016). MetaMask lets users to store and manage keys, broadcast transactions to Ethereum nodes, send and receive Ethereum-based cryptocurrencies and tokens. The private key associated with an Ethereum account is used to sign transactions and establish a connection to the Ethereum node. MetaMask is a Hierarchical Deterministic Wallet (HD wallet). HD wallets derive keys from a single seed. The keys are derived in a tree structure where the parent key is derived from a single seed, the child keys are derived from parent keys, and the grandchild keys are derived from child keys.

4 Implementation

- **Web3.js:** Web3.js is a suite of libraries that may be used to interact with a local or remote Ethereum node through HTTP, IPC, or WebSocket(Web3.js, 2014). It is also the official Ethereum JavaScript API. It implements the JSON RPC protocol by providing a wrapper for the JSON RPC interface to connect and interact directly with the blockchain. Web3.js provides methods for the front-end website to interact with smart contracts, sign and send transactions, access data and execute actions on the blockchain. It can be used both server-side and client-side.

4.2.6 User Interface

- **ReactJS:** ReactJS is a open-source JavaScript library to build interactive user interfaces(React, 2013).
- **Node JS:** Node.js is an open-source and cross-platform JavaScript runtime environment which executes JavaScript. Node.js allows developers to dynamic web page content using different JavaScript libraries(Node.js, 2009).
- **Material-UI:** It is an open-source library to create easier and better-looking components and front-end elements in a React web application(Material-UI, 2014).
- **Ant Design:** It is also a React UI library to build interactive user interfaces(Ant-design, 2019).

4.3 Implementation Details

4.3.1 System Actors

The four actors of the system identified in Section 3.2.2 will act as EOAs in Ethereum. All the actors will be assigned unique keys, digital signatures, and Ethereum addresses. The admin will deploy the smart contract on the EVM, and they will be automatically assigned the role of the first manager of the system.

4.3.2 UML Diagram - Class Diagrams of the Smart Contracts

The UML Class diagrams of smart contracts implemented in our system is created using sol2uml (Addison, 2021) which is a UML class diagram generator for Solidity contracts. The first UML class diagram shows the smart contracts implementing the carbon tracing and payments system. It can be seen in Figure 4.2. The second UML class diagram shows the smart contract implementing the EthereumDIDRegistry, the registry that handles the Decentralized Identity in the system. It can be seen in Figure 4.3.

4.3 Implementation Details

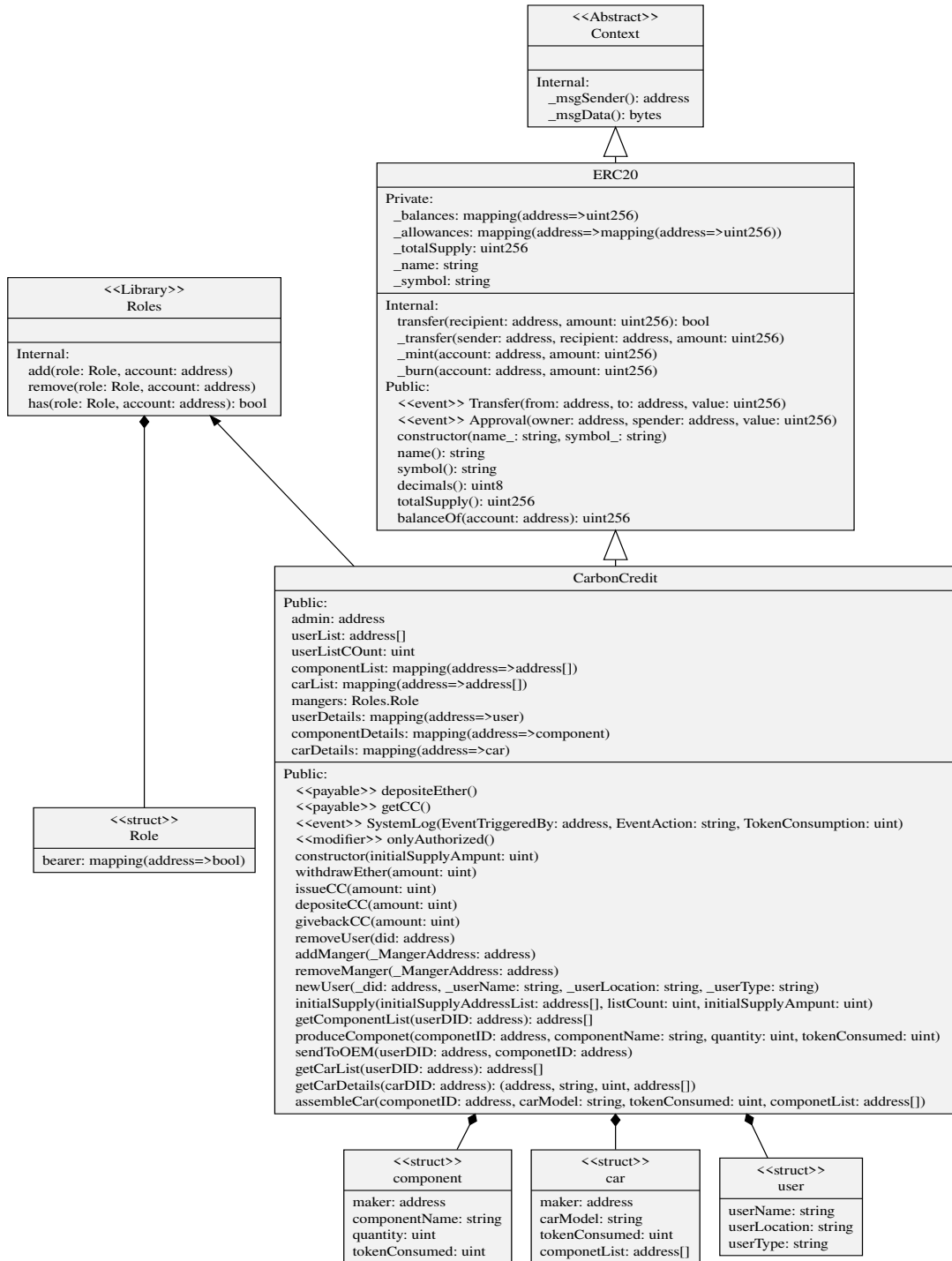


Figure 4.2: Class diagram of the system

4 Implementation

EthereumDIDRegistry
Public: owners: mapping(address=>address) delegates: mapping(address=>mapping(bytes32=>mapping(address=>uint))) changed: mapping(address=>uint) nonce: mapping(address=>uint)
Internal: checkSignature(identity: address, sigV: uint8, sigR: bytes32, sigS: bytes32, hash: bytes32): address changeOwner(identity: address, actor: address, newOwner: address) addDelegate(identity: address, actor: address, delegateType: bytes32, delegate: address, validity: uint) revokeDelegate(identity: address, actor: address, delegateType: bytes32, delegate: address) setAttribute(identity: address, actor: address, name: bytes32, value: bytes, validity: uint) revokeAttribute(identity: address, actor: address, name: bytes32, value: bytes)
Public: <<event>> DIDOwnerChanged(identity: address, owner: address, previousChange: uint) <<event>> DIDDelegateChanged(identity: address, delegateType: bytes32, delegate: address, validTo: uint, previousChange: uint) <<event>> DIDAttributeChanged(identity: address, name: bytes32, value: bytes, validTo: uint, previousChange: uint) <<modifier>> onlyOwner(identity: address, actor: address) identityOwner(identity: address): address validDelegate(identity: address, delegateType: bytes32, delegate: address): bool changeOwner(identity: address, newOwner: address) changeOwnerSigned(identity: address, sigV: uint8, sigR: bytes32, sigS: bytes32, newOwner: address) addDelegate(identity: address, delegateType: bytes32, delegate: address, validity: uint) addDelegateSigned(identity: address, sigV: uint8, sigR: bytes32, sigS: bytes32, delegateType: bytes32, delegate: address, validity: uint) revokeDelegate(identity: address, delegateType: bytes32, delegate: address) revokeDelegateSigned(identity: address, sigV: uint8, sigR: bytes32, sigS: bytes32, delegateType: bytes32, delegate: address) setAttribute(identity: address, name: bytes32, value: bytes, validity: uint) setAttributeSigned(identity: address, sigV: uint8, sigR: bytes32, sigS: bytes32, name: bytes32, value: bytes, validity: uint) revokeAttribute(identity: address, name: bytes32, value: bytes) revokeAttributeSigned(identity: address, sigV: uint8, sigR: bytes32, sigS: bytes32, name: bytes32, value: bytes)

Figure 4.3: Class diagram of the EthereumDIDRegistry

4.3.3 Smart Contract Implementation

In this section, we discuss the technical details regarding the smart contract implementation on the blockchain. We mainly discuss the three major smart contracts in the system.

ERC-20 tokens

Based on the requirements identified in Section 3.3, i.e. R4, R5, R6, a decision was made to represent the carbon credit allowance on blockchain using fungible, utility tokens (discussed in Section 2.1.11). Tokens are smart contracts that implement the standardized interface, and many different token standards are available. ERC-20⁹ tokens define the interface for a fungible token that a contract must implement to be compliant to ERC-20 standard interface that is proposed in EIP-20¹⁰ (EIP-20, 2015).

The smart contract ERC20 in our system is taken from Open Zeppelin (OpenZeppelin, 2018). Open Zeppelin provides contract implementations of standardized use cases with the best security practices. The ERC-20 contract from Open Zeppelin could not be used for our application directly. So, the contract has been modified based on the requirements of the system and implemented to represent the carbon credit allowance in the system. For instance, according to the ERC-20 standard, the imple-

⁹ERC: Ethereum Request for Comment

¹⁰EIP: Ethereum Improvement Proposals

mented tokens should be transferable between the network participants. According to the carbon tax policy (discussed in Section 3.1.3) it is not possible to transfer or trade carbon credit allowances between the participants of the supply chain, such as OEMs and suppliers. So we only implement the tokens transfer functionality from managers to suppliers/OEMs and vice-versa.

Some functionalities relevant to the system were used from the ERC-20 standard contract, such as annual distribution of tokens by managers, minting of tokens, transfer of tokens to the admin account when OEMs or suppliers consume carbon credit to produce car/components. The ERC20 contract and CarbonCredit contract are shown in the class diagram in Figure 4.2.

In implementing a carbon credit allowance, it is not possible to represent carbon credits as a floating-point because Solidity does not support floating point numbers. So in the system, the lowest unit of carbon credit allowance is 1 token. We consider that 1 token equals to 1/1000 or 0.001 t CO₂e. So, a car manufactured by emitting 4 t CO₂e will spend 4000 carbon credit allowance. To map the carbon footprint to the carbon tax, the price of each carbon credit allowance in our system is set as 10⁻⁸ ethers. That also means that if a supply chain participant buys 1 ether worth of tokens, they receive 10⁸ carbon credit allowance(tokens).

Also, when the leftover tokens are returned by the supply chain participant back to the contract account, they receive back the price of the tokens. The fixed price also ensures that hoarding the tokens for the next year will not benefit the supply chain participant, assuming the price of Ether does not fluctuate.

Roles contract

The roles contract is also used from the OpenZeppelin contracts for access control in the application. The roles contract can be used to define multiple roles and their members. The contract also can add, remove and validate if someone belongs to a particular role. So this functionality was mainly used to manage the roles of the manager in the system.

Identity

Based on the requirement R3 identified in Section 3.3 a decision was made to use the Decentralized identity standard proposed as EIP-1056 (EIP-1056, 2018) as a unique identifier for the actors, the components, and the cars in the supply chain that will be verifiable on the blockchain.(introduction to Decentralized identity is discussed in Section 2.1.12).

The EIP-1056 specifies a standard for registry called EthereumDIDRegistry that can be deployed once and used commonly by everyone. The registry holds the identity information about all the entities, such as the actors, the components, and the cars. It maintains a mapping between the custom-defined DID and the addresses on the blockchain. So entities will have two addresses, one from the Ethereum address and the DID(Panait Drăgnoiu et al., 2020). There are numerous DID methods that follow the specification and implement EIP-1056. We utilize the open-source implementation of ethr-did-registry by uPort. The EthereumDIDRegistry is the smart contract developed

4 Implementation

by uPort that manages the DIDs, and it allows the entities to be the owners of their identity, which uses Ethereum addresses as self-managed identities. The public-private key pair corresponding to the Ethereum address maintains the ultimate control over the identity represented by DID (Panait Drăgnoiu et al., 2020). The registry also has several other features, such as allowing the identity owner to edit customized attributes, delegate identity, and change the identity owner; however, none of these features were required for our system. As a result, we did not utilize them.

We utilized the `ethr-did-resolver` library to resolve the DID Document for a DID identity developed by the Decentralized Identity Foundation. The library utilizes the Ethereum address and looks at the read-only functions and contract events on the `ethr-did-registry` smart contract to create a DID document.

4.3.4 Architecture of the Implemented System

The actors (suppliers, OEMs and managers) of the system can access the implemented Dapp using a browser. The browser is connected to a Nodejs web server which renders the user interface of the application on the browser. The system consists of a private key manager called Metamask. When the actors interact with the user interface, unsigned transactions are broadcasted to the Metamask. Metamask signs the transactions using the private key of the actor who is logged in the system. The private key manager also helps to retrieve the state of the blockchain and register for event subscriptions. The signed transactions are then passed to the Go-Ethereum client via the JSON RPC interface, where the transaction may trigger some function defined in the smart contract. The Metamask injects a web3 provider object into the Go-Ethereum client. The transaction triggers the bytecode in the EVM and is based on the triggered function, and data is written or read from the blockchain.

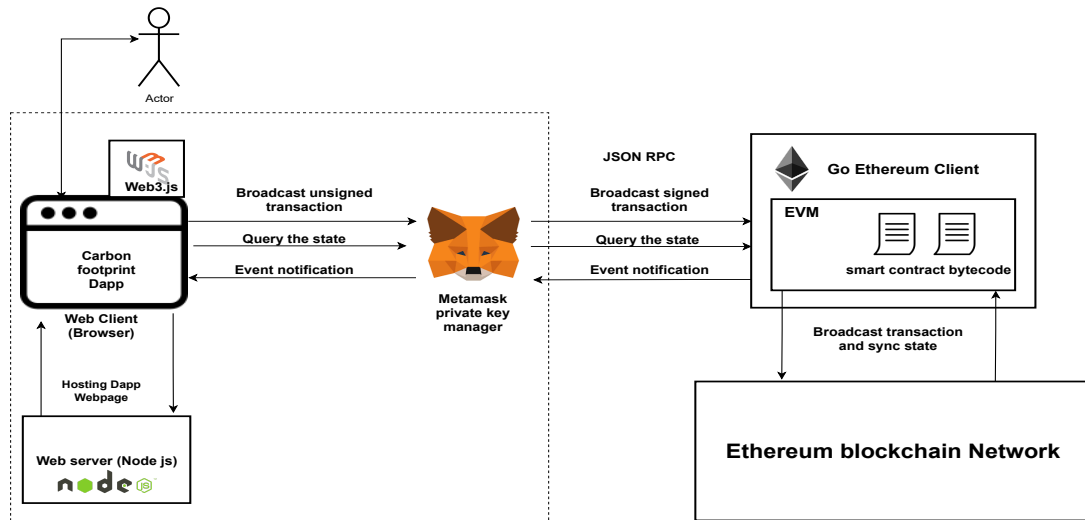


Figure 4.4: Architecture of the implemented system (inspired from (Gallersdörfer et al., 2020))

4.4 User Interface

In this section, we list the user interface of our application based on the scenario of their functionality.

4.4.1 User Interfaces Related to Managers

As mentioned before, the deployer of the smart contract (who is also called admin throughout this document) is also allotted the role of manager. The admin is the only manager at the beginning.

Dashboard of Managers

The dashboard interface is shown in Figure 4.5. The interface displays the Ether balance and the Carbon credit balance. The interface also displays the Ether balance and Carbon credit balance in the contract account. The supply chain participants request for carbon credits and surrender carbon credits to this contract account. The dashboard also shows the log of all the events taking place in the system.

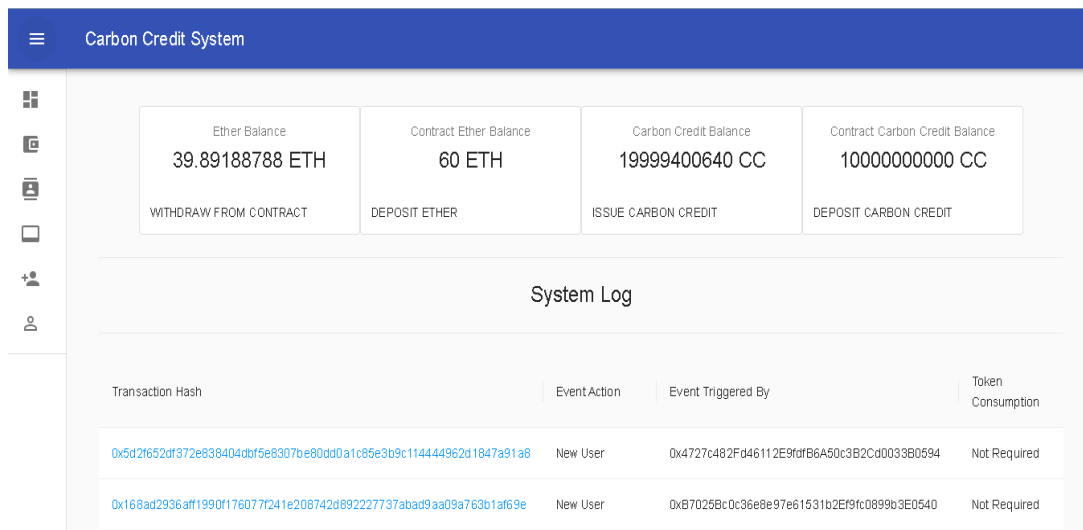


Figure 4.5: Dashboard of the manager

The dashboard also offers the following functionalities. The first functionality is to withdraw Ether from the contract account. It will allow the managers to withdraw the amount collected as carbon tax for the emissions by supply chain participants like suppliers and OEMs. The second functionality is to deposit Ether into the contract account. This is used when the Ether balance in the contract falls low and the supply chain participants surrendering carbon credits should be returned incentives in Ether. The third function is to issue carbon credit, which means to mint more carbon credits. The fourth functionality is to deposit carbon credits to transfer carbon credits from the manager's account to the contract account.

4 Implementation

Initial Supply Interface

The initial supply interface is used at the start of the year to distribute the annual allowance of carbon credit allowance that the regulatory authority decides to the supply chain participants. The initial supply interface is shown in Figure 4.6

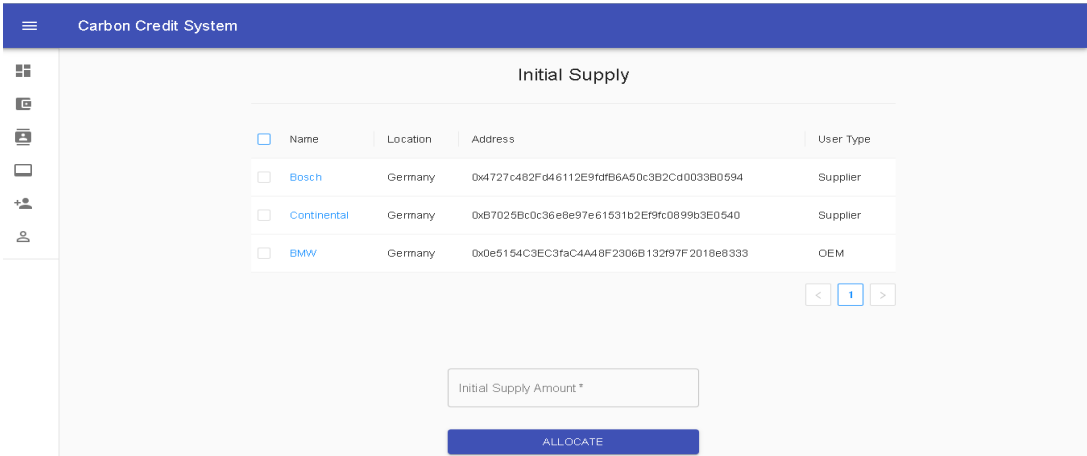


Figure 4.6: Initial supply interface

User List Interface

The user list interface is used to ensure all the participants are valid. Incase any invalid user signs up in the system, they will be set as banned user and they cannot access the system. The user list interface is shown in Figure 4.7.

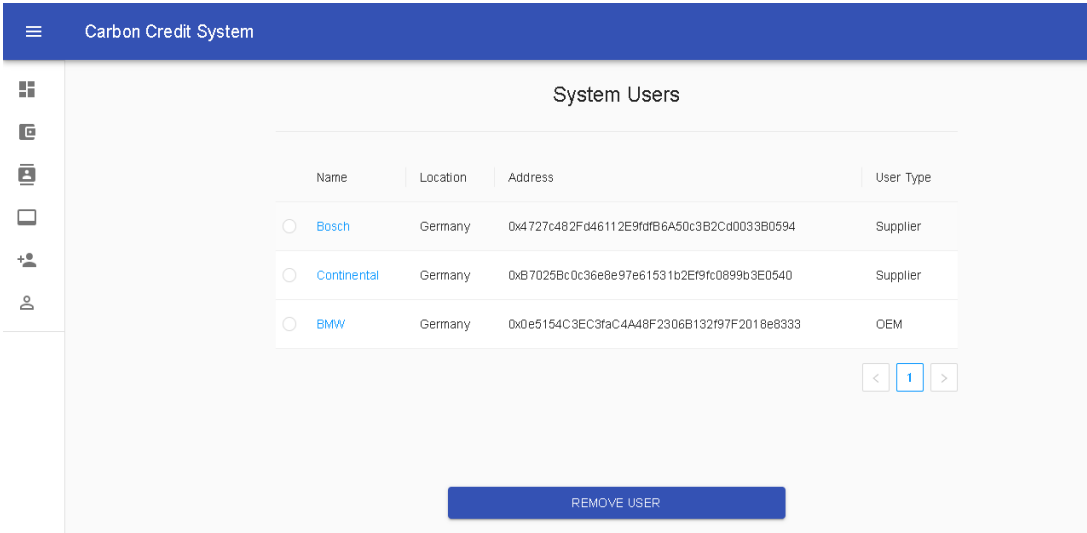


Figure 4.7: User list interface

DID resolver

In this interface the DID is resolved into the corresponding DID document. The DID resolver is shown in the Figure 4.8.

DID Details

DID *

0x4727c482Fd46112E9fdB6A50c3B2Cd003c

GET DID DETAILS

```
{
  "didDocumentMetadata": {},
  "didResolutionMetadata": {
    "contentType": "application/did+json"
  },
  "didDocument": {
    "@context": [
      "https://www.w3.org/ns/did/v1",
      "https://identity.foundation/EcdsaSecp256k1RecoverySignature2020/lds-ecdsa-secp256k1-recovery2020-0.0.json"
    ],
    "id": "did:ethr:development:0x4727c482Fd46112E9fdB6A50c3B2Cd003c380594"
  }
}
```

Figure 4.8: DID resolver

Add Manager interface

In this interface a manager can add an ethereum address to add them also as a manager of the system. Figure 4.9 shows the add manager interface.

Add Manager

Ethereum Address *

ADD MANAGER

Figure 4.9: Add Manager interface

Remove Manager interface

In this interface a manager can add an ethereum address to the interface to remove a manager from the system. Figure 4.10 shows the add manager interface.

Remove Manager

Ethereum Address *

REMOVE MANAGER

Figure 4.10: Remove Manager interface

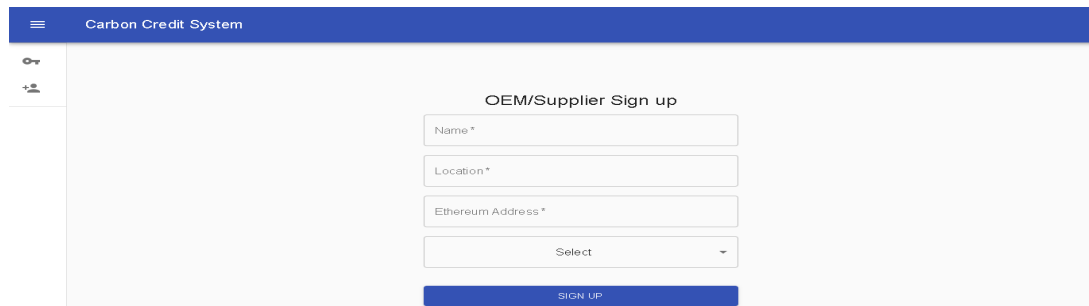
4 Implementation

4.4.2 User Interfaces for Suppliers

In this subsection we discuss about the user interfaces that suppliers use to interact with the system.

Supplier Sign-up

The suppliers who have not registered in the system already have to register in the system through this interface. Figure 4.11 shows the supplier sign-up interface.



Carbon Credit System

OEM/Supplier Sign up

Name *

Location *

Ethereum Address *

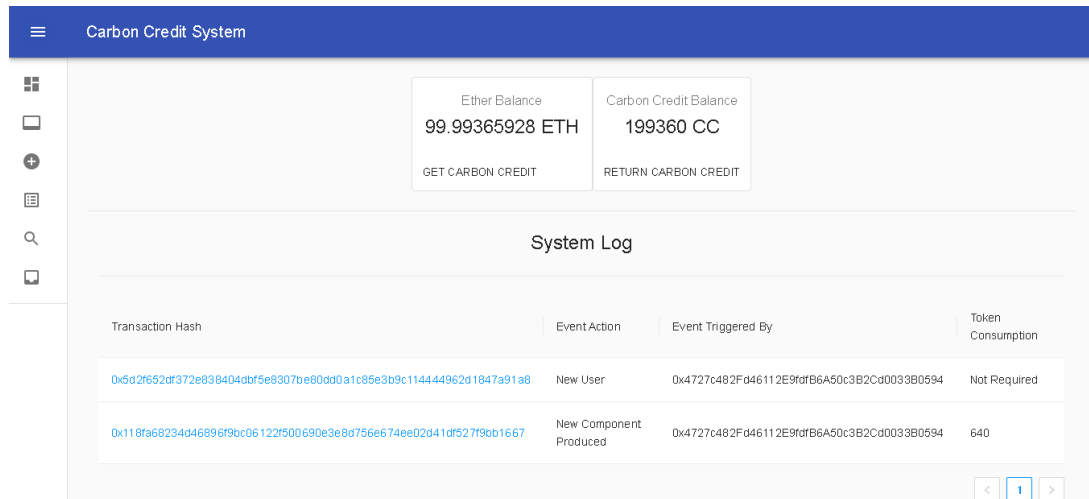
Select

SIGN UP

Figure 4.11: Suppliers Sign-up interface

Supplier Dashboard

The suppliers who are registered in the system have access to the dashboard. The dashboard displays ether balance and carbon credit allowance balance. The interface also has two functionalities: get carbon credit, where the supplier requests a carbon credit by sending an equivalent amount of Ether. The second functionality is return carbon credit which is to surrender the excess carbon credit to the system. There is also a log of all the events that show all the supplier interactions on the system. Figure 4.12 shows the supplier dashboard interface.



Carbon Credit System

Ether Balance
99.99365928 ETH
GET CARBON CREDIT

Carbon Credit Balance
199360 CC
RETURN CARBON CREDIT

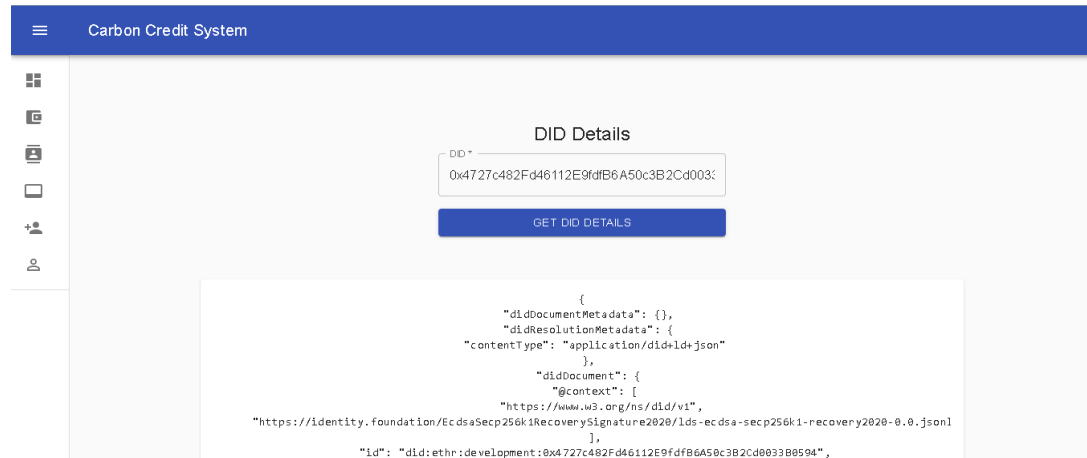
System Log

Transaction Hash	Event Action	Event Triggered By	Token Consumption
0x5d2f652df372e838404dbf5e8307be80dd0a1c85e3b9c114444962d1847a91a8	New User	0x4727c482Fd46112E9fdfB6A50c3B2Cd0033B0594	Not Required
0x118fa68234d46896f9bc06122f500690e3e8d756e674ee02d41df527f9cb1667	New Component Produced	0x4727c482Fd46112E9fdfB6A50c3B2Cd0033B0594	640

Figure 4.12: Suppliers Dashboard interface

DID resolver

In this interface the DID is resolved into the corresponding DID document. The DID resolver is shown in the Figure 4.13.



Carbon Credit System

DID Details

DID *

0x4727c482Fd46112E9fdFB6A50c3B2Cd003:

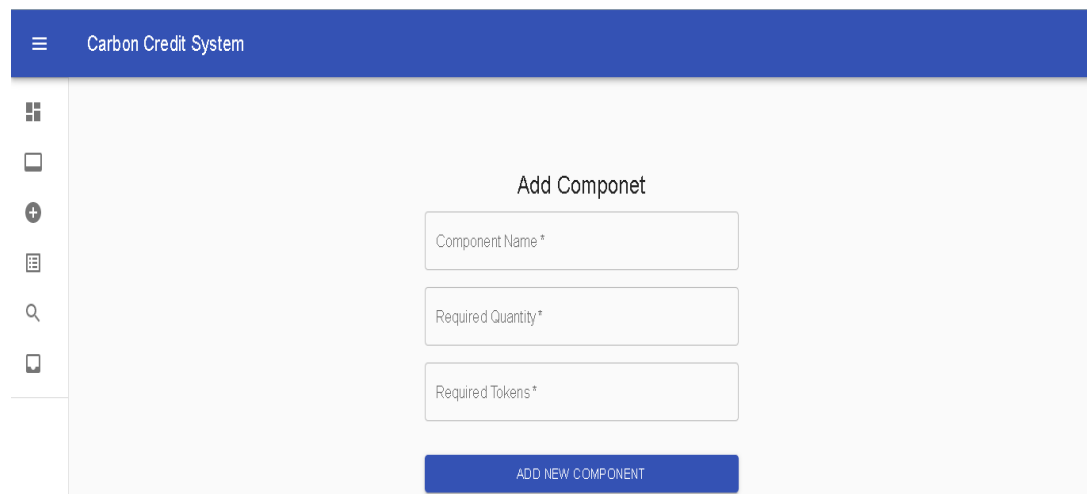
GET DID DETAILS

```
{
  "didDocumentMetadata": {},
  "didResolutionMetadata": {
    "contentType": "application/did+json"
  },
  "didDocument": {
    "@context": [
      "https://www.w3.org/ns/did/v1",
      "https://identity.foundation/EcdsaSecp256k1RecoverySignature2020/lds-ecdsa-secp256k1-recovery2020-0.0.json"
    ],
    "id": "did:ethr:development:0x4727c482Fd46112E9fdFB6A50c3B2Cd003388594"
  }
}
```

Figure 4.13: DID resolver

Add Component interface

In this interface the supplier can record the component produced, the number of quantity and the carbon credit allowance spent to produce the component. The Add component interface for suppliers is shown in the Figure 4.14.



Carbon Credit System

Add Component

Component Name *

Required Quantity *

Required Tokens *

ADD NEW COMPONENT

Figure 4.14: Add Component Interface

4 Implementation

Component list interface

The component list interface lists all the components that were produced by the supplier. The supplier can also generate the QR code for the component on this interface. The component list interface for suppliers is shown in the Figure 4.15.

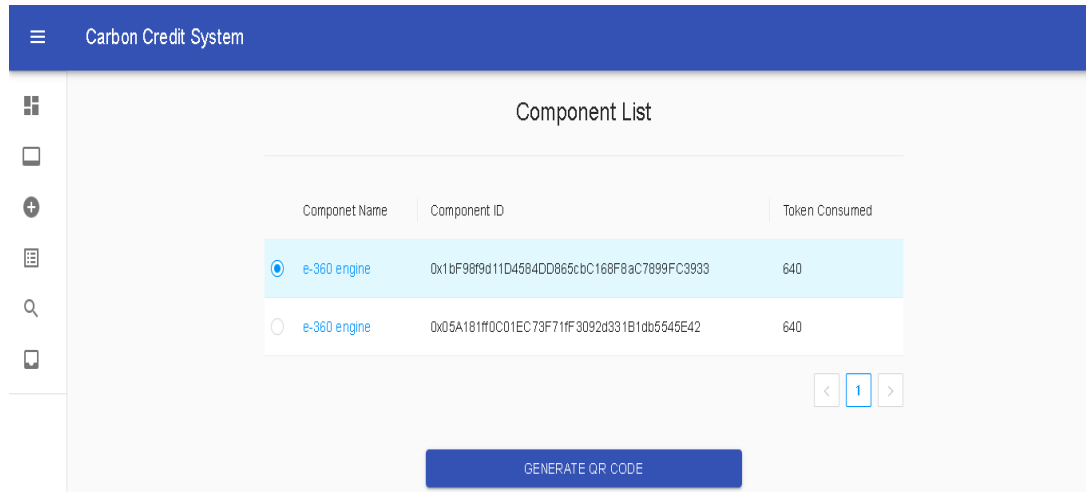


Figure 4.15: Component list Interface

Search Component Interface

The search component interface allows the supplier to search for any component produced by the supplier and the carbon emission emitted to produce. The search uses the DID of the component to search it. The supplier can also use the information from the QR code of the component on this interface. The search component interface for suppliers is shown in the Figure 4.16.

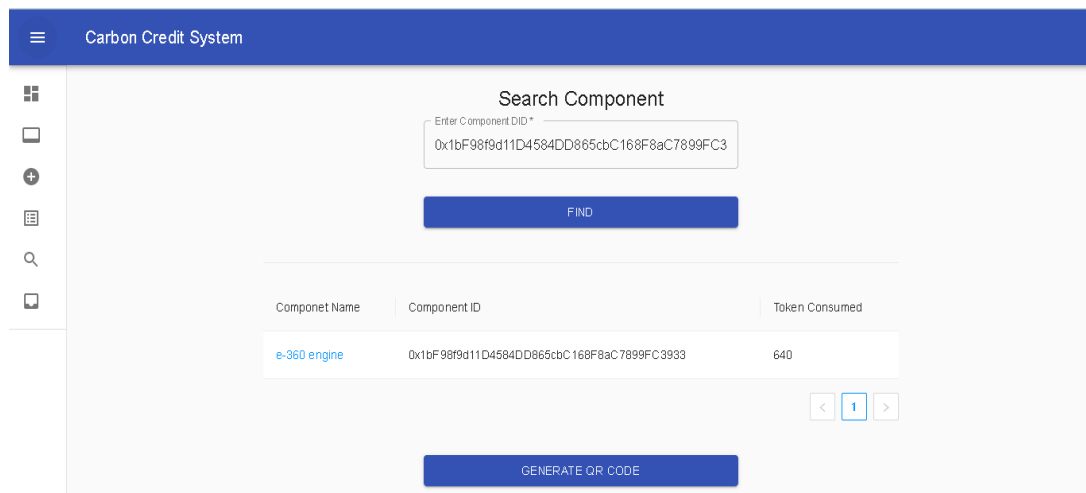


Figure 4.16: Search Component Interface

Transfer Component interface

The transfer component interface allows the suppliers to record that the component is transferred to which OEM. First the component is chosen which needs to be sent and then the OEMs are selected. The transfer component interface for suppliers is shown in the Figure 4.17.

OEM Name	Location	Address
<input checked="" type="radio"/> BMW	Germany	0x0e5154c3ec3faC4A48F2306B132f97F2018e8333

Componet Name	Component ID	Token Consumed
<input checked="" type="radio"/> e-360 engine	0x1bF98f9d11D4584DD865cbC168F8aC7899FC3933	640
<input type="radio"/> e-360 engine	0x05A181ff0C01EC73F71F3092d331B1db5545E42	640

TRANSFER

Figure 4.17: Transfer Component Interface

4.4.3 User Interfaces for OEMs

In this subsection we discuss about the user interfaces that are used by OEMs to interact with the system.

OEM Sign-up

The OEMs who have not registered in the system already have to register in the system through this interface. Figure 4.18 shows the OEM sign-up interface.

OEM/Supplier Sign up

Name *

Location *

Ethereum Address *

Select ▼

SIGN UP

Figure 4.18: OEM Sign-up interface

4 Implementation

OEM Dashboard

The OEM who are registered in the system have access to the dashboard. The dashboard displays ether balance and carbon credit allowance balance. The interface also has two functionalities such as get carbon credit where the supplier request a for carbon credit by sending equivalent amount of Ether. The second functionality is return carbon credit which is to surrender the excess carbon credit to the system. There is also a log of all the events that show all the interactions of supplier on the system. Figure 4.19 shows the OEM dashboard interface.

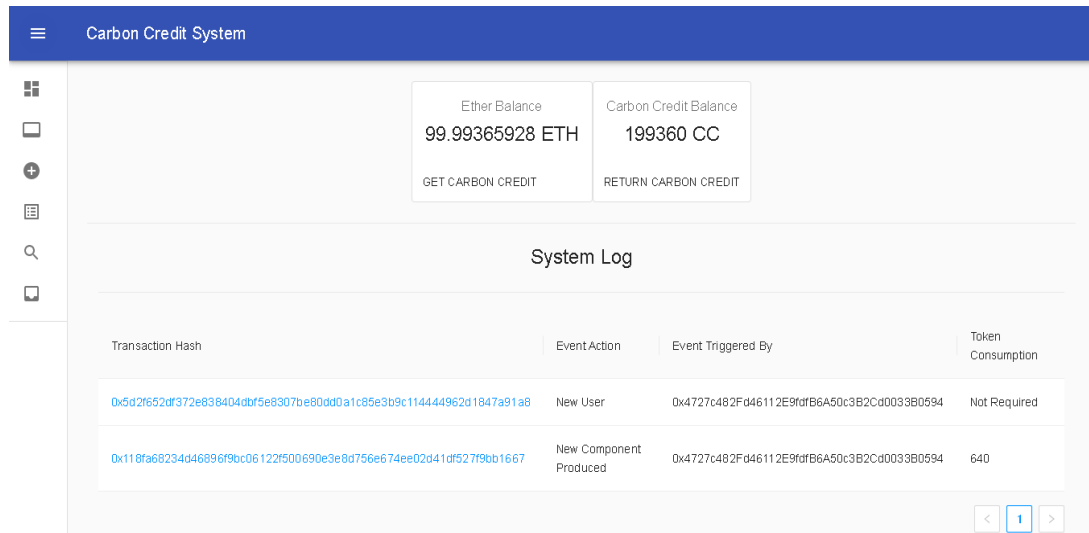


Figure 4.19: OEM Dashboard interface

DID resolver

In this interface the DID is resolved into the corresponding DID document. The DID resolver is shown in the Figure 4.20.

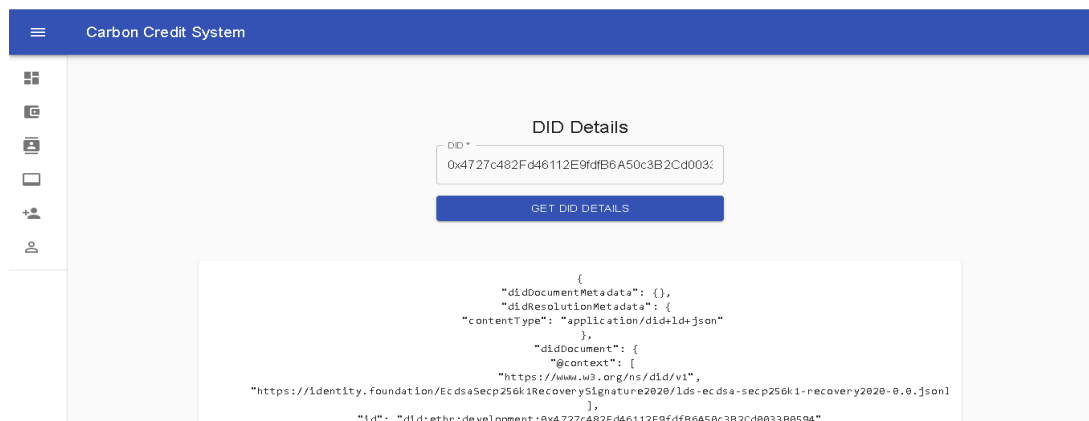


Figure 4.20: DID resolver

Assemble car interface

In this interface, the car is assembled using all the components available from the suppliers. The carbon emissions related to the assembly of the car is also recorded. The car assembly interface is shown in the Figure 4.21.

Carbon Credit System

Assemble Car

<input type="checkbox"/>	Component Name	Component ID	Token Consumed
<input checked="" type="checkbox"/>	e-360 engine	0x1bF98f9d11D4584DD865cbC168F8aC7899FC3933	640
<input type="checkbox"/>	e-360 engine	0x05A181ff0C01EC73F71fF3092d331B1db5545E42	640
<input checked="" type="checkbox"/>	t-100 wheels	0x26B98252D9c2c93C8f803799805E03f5D6853278	45

< 1 >

Model Name *
BMW X2

Required Tokens *
700

ASSEMBLE

Figure 4.21: Assemble car interface

Car list interface

The car list interface lists all the cars that were produced by the OEM. The OEM can also generate the QR code for the car on this interface. The car list interface for OEMs is shown in the Figure 4.22.

Carbon Credit System

Car List

<input type="radio"/>	Car Model	Car DID	Maker DID	Token Consumed
<input type="radio"/>	BMW X2	0xeaa48B8290cf865145feBa9e7Ddc5470fe0843D4	0x0e5154C3EC3faC4A48F2306B132f97F2018e8333	700
<input type="radio"/>	BMW X2	0x7B1CE66d76ab85936775442918B9a67b364A2c2A	0x0e5154C3EC3faC4A48F2306B132f97F2018e8333	700

< 1 >

GENERATE QR CODE

Figure 4.22: Car list Interface

4 Implementation

Car showroom interface

The car showroom interface lists all the cars and the component used to build the car and also the carbon emissions emitted during the production of both the components and the car that were produced by the OEM. The OEM can also generate the QR code for the car on this interface. The car show interface for OEMs is shown in the Figure 4.23.

Car Model	Car DID	Maker DID	Token Consumed
BMW X2	0xea348B8290cf865145feBa9e7Ddc5470fe0843D4	0x0e5154C3EC3faC4A48F2306B132f97F2018e8333	700

Component Name	Component ID	Maker DID	Token Consumed
e-360 engine	0x1bf98f9d11D4584DD865cbC168F8aC7899FC3933	0x4727c482Fd46112E9fdfB6A50c3B2Cd0033B0594	640
t-100 wheels	0x26B9825209c2c93C8f803799805E03f5D6853278	0xB7025Bc0c36e8e97e61531b2Ef9fc0899b3E0540	45

Figure 4.23: Car list Interface

Search Car Details interface

The search car details enables the OEMs to search for the cars they produced using the DID of the car and the interface also lists all the component used to build the car and the carbon emissions emitted during the production of both the components and the car that were produced by the OEM. The OEM can also use the information from the QR code of the car on this interface. The car show interface for OEMs is shown in the Figure 4.24.

Car Model	Car DID	Maker DID	Token Consumed
BMW X2	0xea348B8290cf865145feBa9e7Ddc5470fe0843D4	0x0e5154C3EC3faC4A48F2306B132f97F2018e8333	700

Component Name	Component ID	Maker DID	Token Consumed
e-360 engine	0x1bf98f9d11D4584DD865cbC168F8aC7899FC3933	0x4727c482Fd46112E9fdfB6A50c3B2Cd0033B0594	640

Figure 4.24: Search Car Interface

4.5 Summary

This chapter provided the technical details of implementing the designed system architecture in the previous chapter (Chapter 3). This chapter started with implementations decisions. The first decision was about which type of blockchain should be used. Based on the decision tree provided in (Wüst and Gervais, 2018), we decided on working with the private-permissioned blockchain. Also, the private permissioned blockchain was less explored in terms of carbon footprint traceability in the literature. The second decision was to find which blockchain implementation fits our requirements. The selected blockchain implementations were then compared to decide on which blockchain should be used to implement the system. We decided on using the private Ethereum blockchain. The third decision was if it was necessary to use a back-end data store and which front-end framework could be used for our system. The chapter then discusses the major implementation concepts related to Ethereum blockchain and decentralized applications. Further, the chapter discusses the development tools and frameworks used to implement the system. The next section dealt with implementation details such as the actual architecture with all the components and the information flow between them, and the smart contract implementations are discussed in detail. Finally, the user interface of the implemented application is also discussed, and the implemented interfaces are shown as results. This chapter primarily answers research question 3 (RQ3) by discussing the implementation decisions, describing the software frameworks, tools, and methodologies that may be utilized to construct the system architecture envisioned in the previous chapter.

5 Evaluation

In this chapter, we discuss and present an evaluation of the conceptual design (presented in Chapter 3) and the implemented prototype (presented in Chapter 4). This chapter primarily aims to answer the fourth research question (RQ4) by analyzing the applicability of the implemented solution to our identified problem and the feasibility and scalability of the solution.

5.1 Theoretical Evaluation

It is critical to evaluate the implemented solution in order to determine to what extent the thesis artifacts fulfill the requirements set forth by the identified problem in this thesis. Therefore, the implemented prototype will be evaluated in the following subsection to determine if it fits the identified requirements in Section 3.3 and if the implemented prototype fulfills the requirements satisfactorily. In addition, the Section 5.1.2 covers the limitations in conceptual design as well as in the implemented prototype.

5.1.1 Fulfilment of Requirements

In this subsection, we discuss if the requirements of the system defined in Section 3.3 are achieved through the implemented system. We evaluate and justify how every requirement is fulfilled.

Requirement 1: The designed system should be able to reliably record the carbon emission data from the facilities of OEM and suppliers on the blockchain

The requirement has been fulfilled, the prototype is built over a private Ethereum blockchain: Go-Ethereum, which allows both the suppliers and OEM to reliably record their carbon emissions on the blockchain as transactions using the user interface.

Requirement 2: The designed system should be able to assign a unique blockchain-based digital identity to the manufactured parts and cars at facilities of OEM and suppliers.

In the implemented system, we use the DID as the digital identity which is assigned to every manufactured component and produced car, and these DIDs can be generated as a QR code on the web client to be attached to the produced product.

Requirement 3: The carbon footprint of the components and cars should be traceable in the designed system using the blockchain-based digital identity (DID).

In Section 4.4, there are two user interface discussed with the title “Search Component Interface” and “Search Car Details”. The Search Component Interface is available to the suppliers to search for the carbon footprint of the product using DID of the product. The Search Car Details is available to the OEMs to search for the carbon footprint of the produced car using DID.

Requirement 4: The designed system should perform carbon pricing (Carbon tax) by implementing a representation of carbon credit allowance on the blockchain that can be transferred and surrendered.

We implemented a token based on the ERC-20 standard to implement the carbon credit. The ERC-20 standard has been implemented in the CarbonCredit.sol smart contract. This carbon credit allowance can only be bought using the native cryptocurrency. Since we use the Go-Ethereum blockchain, the price is paid using Ethers. It is also possible to transfer and surrender the tokens according to the carbon tax policy.

Requirement 5: The annual cap on carbon credit allowance offered yearly is decided based on the regulations and can be changed.

In Section 4.4, the initial supply interface is used by the managers to allocate carbon credit allowance annually for a year to the supply chain participants based on the regulations set by the regulatory authority.

Requirement 6: The price of the carbon credits is fixed, and the payments have to be made using the native cryptocurrency of the blockchain.

The price of buying the carbon credit allowance from the managers is fixed, and they are paid in Ethers.

Requirement 7: The designed system should only be accessible to the registered participants. The participants should be able to view the transactions on the blockchain according to their permissions.

After a supplier or OEM registers in the system, the managers can validate their membership and allot their carbon credit allowance. If they are not a valid member, they can be removed from the system. We use the Roles.sol contract from OpenZeppelin to handle the roles in the system.

5.1.2 Limitations of the Implemented System

This subsection discusses the limitations of the conceptual design and the prototype, which can also be the gaps identified for future work.

1. The system did not consider and model an entire automobile supply chain (the entire production life cycle of the car) from the first upstream activity to the last downstream activity. The system only modeled the supply chain consisting of the first-tier suppliers and OEMs.

5 Evaluation

2. The considered supply chain is elementary, and it ignores many possible work-flows and process flows.
3. The supply chain participants enter their carbon footprint data. However, it is possible to cross-check and validate their emissions data at a later stage by retrieving the data from the blockchain. Using a human technician to enter the carbon footprint data increases the chances of errors.
4. Since the application is managed by metamask private key manager, losing the seed phrase or private key means permanently losing access to the system.
5. Since the system is implemented in a private blockchain; the Know-Your-Customer validation check is not performed. So, there is a possibility that an anonymous user may act as a supply chain entity and try to become a participant in the blockchain. So, that will be a future work to improve the KYC.
6. Once the smart contracts are deployed, the system logic cannot be changed. So if the regulations of the taxation system change drastically from the deployed system, then the system needs to be redeployed. The system can only handle the changes in annually distributed carbon credit allowance.
7. Every transaction, interaction with the smart contract costs a transaction fee (gas-cost).
8. The performance of the system is determined by the Blockchain configuration and the Blockchain implementation utilized for the prototype. So, some key performance parameters like the throughput¹ and the latency² depend on the implementation and configurations of the blockchain.

5.2 Implemented Prototype Evaluation

5.2.1 Software Testing

Testing the software is also another way of evaluation where the built software is tested if the system is actually working. We test the smart contracts which forms the core logic of our implemented software. To test the smart contracts we use the built in automated testing framework in Truffle framework that we used to perform unit tests on the smart contracts. The Truffle uses the Mocha testing framework which is used to write unit tests for the smart contract. To perform the testing, the truffle framework is used to run a local ganache blockchain and then the testing framework performs unit tests for every defined test cases. For every test case, a new instance of contract is deployed (also called the clean-room environment), only after which the test cases are executed. The result of the executed automated unit test is given in Figure 5.1.

¹Transaction Throughput: It is the measure of the rate at which valid transactions are committed to the blockchain.

²transaction Latency: The time until the transaction is processed after submission.


```

Contract: Carbon Credit App contract test
Deployed Address: 0x3581E556a3c42fE82cAD18C0c9d247b27f2Bd58b
✓ 1. The contract is deployed correctly
✓ 2. Is able to deposit Ether to contract (320ms)
✓ 3. Is able to withdraw Ether from contract (681ms)
✓ 4. Is able to issue new carbon credit (350ms)
✓ 5. Is able to deposit carbon credit to contract (372ms)
✓ 6. Is able to buy carbon credit (243ms)
✓ 7. Is able to return carbon credit for incentive (480ms)
✓ 8. Is able add & remove user(s) (548ms)
✓ 9. Is able add & remove manager(s) (530ms)
✓ 10. Is able assign initial supply (284ms)
✓ 11. Supplier is able to produce new components (1284ms)
✓ 12. Supplier is able to send produced components to OEM (1932ms)
✓ 13. OEM is able to assemble car using received component (2240ms)

13 passing (11s)

```

Figure 5.1: Unit test results

5.2.2 Scalability Evaluation

The scalability evaluation of the prototype can be performed by simulating the sending of the same flow of transactions to the private blockchain that the implemented prototype sends. The testing will be automated by a script and will include a large number of transactions flows. The transaction processing time or the time taken to process the complete flow of transactions is recorded. Finally, we try to understand the behavior and performance parameters of transactions from our system on the private blockchain.

The script to send a flow of transactions was built using the JavaScript and web3.js library. Node.js was used to run the script and send several bunches of transactions to the private blockchain. For calculating the scalability performance of the prototype, we run the private blockchain implementation of Go-Ethereum. We run the exact implementation on two different computer systems with different configurations, but the script to send the flow of transactions is the same. The first system (PC1) configuration is 12GiB RAM, Core i7-6700HQ CPU, and the second system (PC2) configuration is 8GiB RAM, core i5-5200U CPU (It should be noted that both these computers are generic personal computers and are in no way optimized to run blockchain). According to (Rouhani and Deters, 2017) in a comparable test performed in literature, it is concluded that the more RAM a system has, the faster the Geth blockchain can process and execute transactions.

However, while deploying the smart contracts of the implemented system on Geth using truffle, the deployment failed because gasLimit set by default in Geth was 4800000 by default, and the EthereumDIDRegistry.sol smart contract surpassed that limit. As a result, we adjusted the gasLimit in the genesis configuration JSON file and redeployed the system, which worked fine.

The simulation script tests the scalability of two modules in the prototype of the

5 Evaluation

supply chain participants' registration and the supply chain automobile production. The first flow (i.e., the supply chain participants registration) is tested using a script to register a supplier or OEM to the Blockchain network as well as the system. It involves only one transaction, that is, registration of supplier or OEM. Figure 5.2 shows the graph of the time taken for transaction execution during the simulation of user sign-up transactions for 1, 10, 50, and 100 when performed on PC1 and PC2.

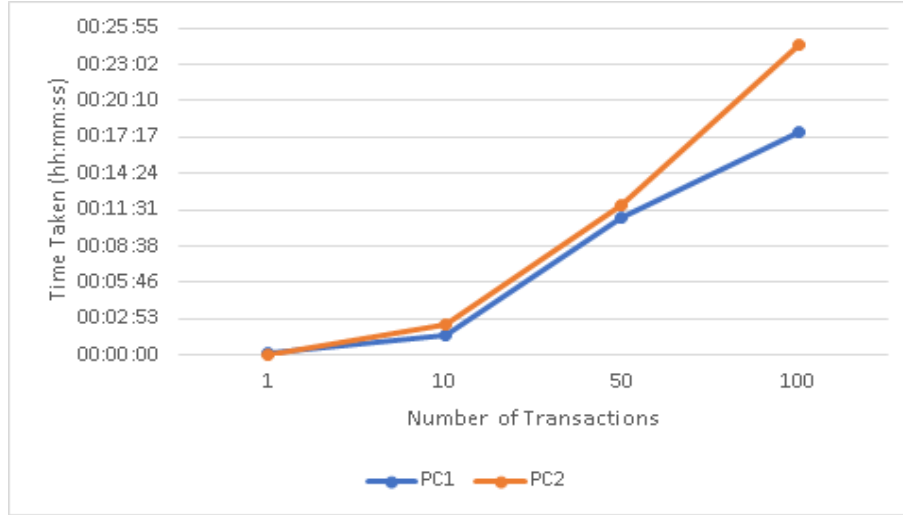


Figure 5.2: Total time for processing different amounts of User sign-up transactions by different systems PC1 and PC2

The second flow that is simulated in the scalability evaluation is automobile production. It involves three different transactions, i.e., the production of a component, sending component to OEM, and the assembly of component into a car. So, we perform this test in two phases. In the first phase, we write the script for an automobile production flow with two suppliers and one OEM. So the total number of transactions to build a car in this scenario will be five transactions, i.e., produce component 1, produce component 2, send component 1 to OEM, send component 2 to OEM and assemble the car. In the Figure 5.3 the graph shows the time taken for transaction execution during the simulation of automobile production flow with two suppliers to produce: 1 car(5 transactions), 10 cars(50 transactions), 50 cars(250 transactions), 100 cars(500 transactions) when performed on PC1 and PC2.

In the second phase of simulating the second flow, i.e., evaluation of automobile production, we write the script for an automobile production flow with ten suppliers and one OEM. So the total number of transactions to build a car in this scenario will be 21 transactions, i.e., produce component (1,2,3,...10), send component (1,2,3...10) to OEM, and assemble the car. In the Figure 5.4 the graph shows the time taken for transaction execution during the simulation of automobile production flow with ten suppliers to produce: 1 car(21 transactions), 10 cars(210 transactions), 50 cars(1050 transactions) when performed on PC1 and PC2.

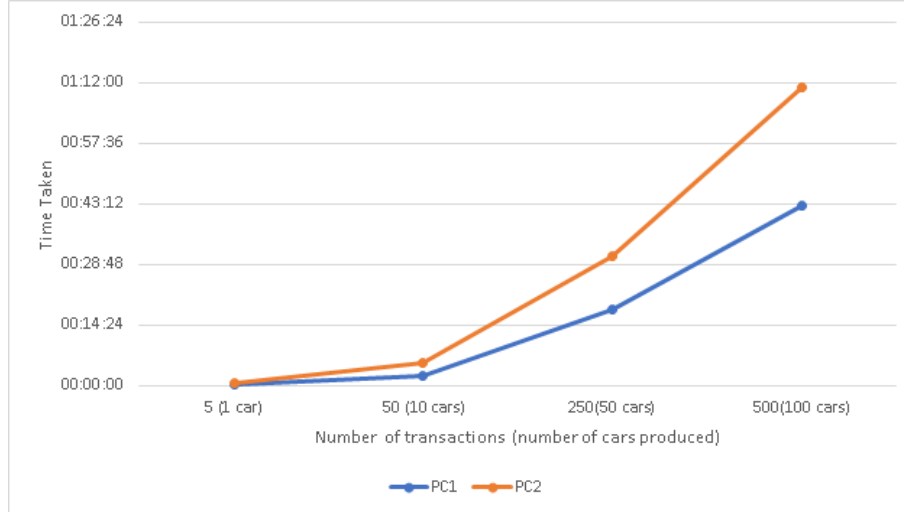


Figure 5.3: Total time for processing different amounts of transactions from automobile production with 2 suppliers by different systems PC1 and PC2

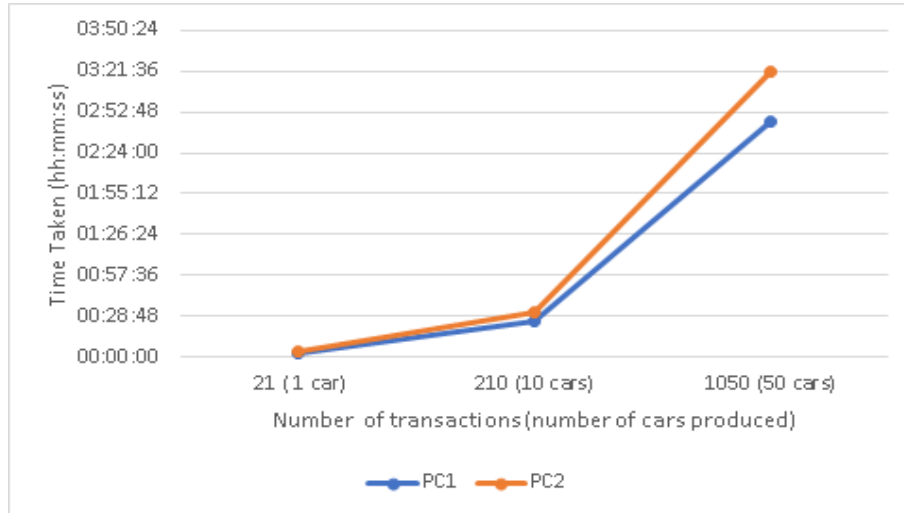


Figure 5.4: Total time for processing different amounts of transactions from automobile production with 10 suppliers by different systems PC1 and PC2

The average transaction time is calculated for all the three simulated flow scenarios by taking the total time taken to execute the transactions and the total number of transactions. These values are plotted and shown in Figure 5.5. Some observations that can be derived from the results of the simulations are: It is observed that the average transaction execution times for the different sets of transactions are different. For example, the average transaction time taken to execute the user signup is greater than automobile production. It is also observed that having a better computation

5 Evaluation

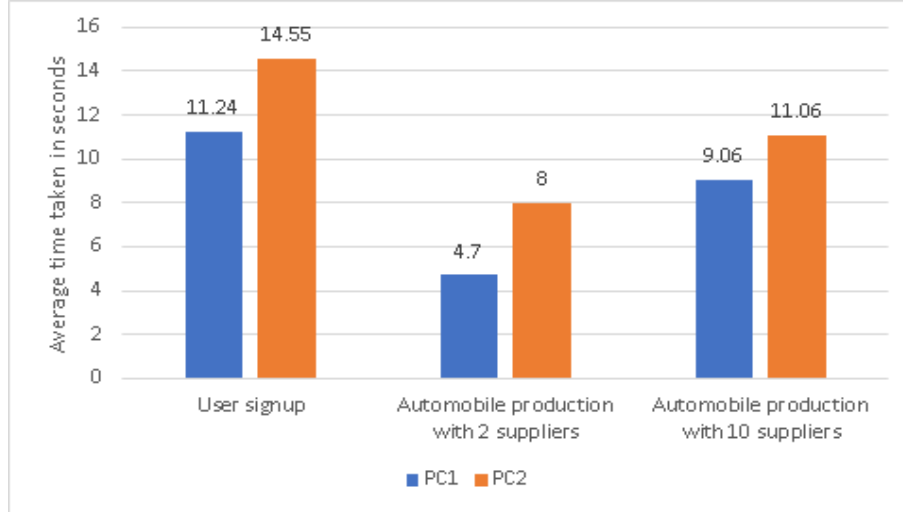


Figure 5.5: Average time for each transaction by different systems PC1 and PC2 for the three simulations performed

infrastructure (like more RAM) helps to get a lower average transaction execution time. From the plotted graphs in Figures 5.2,5.3,5.4 it can also be noted that the underlying implementation and the consensus mechanism in the blockchain may affect the latency³ when there is a large number of transactions, i.e., the increase in time to execute a transaction is not proportional to the increase in the number of transactions. As a result of the observations, we can conclude that the performance of our system when scaled up will be totally dependent on the blockchain implementation. It may be noted that the observations are based on the simulations carried out specifically for this study and may not reflect the real-world scenarios and hence cannot be generalized.

5.2.3 Quantitative Evaluation

Quantitative evaluation can also be performed using blockchain-related metrics such as gas consumption, throughput, latency, etc. The majority of the metrics that may be used to evaluate a blockchain quantitatively are dependent on how it is implemented. However, gas consumption (transaction cost) is unaffected by this. We have already discussed the transaction fee, gas price, the gas cost in the transaction fee section in Section 4.2.1 in detail. Since the prototype we built uses the private permissioned Ethereum blockchain which is a consortium model, it is not possible to map the gas cost to the Ether value in the Ethereum mainnet. However, this evaluation may be helpful to understand how computationally expensive each of these transactions is. Also, it can be a reference if, in the future, the supply chains participants want to become more transparent and adopt public blockchain like Ethereum mainnet.

One of the most costly transactions in terms of gas cost is the deployment of smart

³Transaction Latency: The time until the transaction is processed after submission.

5.2 Implemented Prototype Evaluation

contracts on the blockchain. Table 5.1 shows the deployment cost of Smart Contracts of the prototype on Geth. This data was returned by truffle development after the smart contracts were deployed on the Geth. The gas price is 1 gwei.

Sl No.	Smart Contract	Deployment Cost	Price in Ether	Price in Euro ⁴
1.	EthereumDIDRegistry	1763851	0.001763851	4.90
2.	CarbonCredit	3503658	0.003503658	9.74

Table 5.1: Deployment Gas cost of the Smart Contracts of the implemented system and their conversion to Ether and Euro

The other transactions in the system also cost gas. The average gas cost of most commonly used transactions produced from calling the writing functions in the smart contract are shown in Table 5.2. These values vary according to the bytes of memory of data that are sent along with them. These values were recorded from the Remix IDE after executing these transactions and an average transaction cost is calculated by performing many such transactions. The gas cost is taken to be 1 Wei.

Sl No.	Function Called	Average Deployment Cost	Price in Ether	Price in Euro ⁵
1.	Adding New User	76215	0.000000000000076215	0.00000000021
2.	Produce Component	116202	0.000000000000116202	0.00000000032
3.	Send Component to OEM	91146	0.000000000000091146	0.00000000025
4.	Assemble Car	219074	0.000000000000219074	0.00000000061
5.	Add Manager	49042	0.00000000000049042	0.00000000014
6.	Remove Manager	27095	0.00000000000027095	0.000000000075
7.	Remove User	34080	0.0000000000003408	0.000000000095

Table 5.2: Gas cost of the transactions of the implemented system and their conversion to Ether and Euro

In a supply chain scenario with 1000 suppliers and an OEM to produce a car, we can estimate the deployment cost and the price in Ether and price in Euro as shown in Table 5.3. To produce one car approximately 0.00000057061 Euros will be spent. If 1000 cars are produced in the OEM in a day, then approximately 0.00057061 Euros will be spent. If the same production continues for a year, 0.2054196 Euros may be spent on producing 360,000 cars. It should be noted that these calculations are based

⁴Price as on 16 Aug, 04:38 UTC

⁵Price as on 16 Aug, 04:38 UTC

5 Evaluation

on average deployment cost, and there may be a significant increase or decrease in cost based on the amount of data that is stored.

Function Called	Number of times function called	Average Deployment Cost	Price in Ether	Price in Euro ⁶
Produce Component	1000	116202000	0.000000000116202	0.00000032
Send Component to OEM	1000	91146000	0.000000000091146	0.00000025
Assemble Car	1	219074	0.00000000000219074	0.00000000061

Table 5.3: Gas cost of the transactions of the implemented system and their conversion to Ether and Euro in a supply chain scenerio

5.3 Conclusion

This chapter provided the details about the evaluations that are carried out to test the conceptual design (Chapter 3) and the implemented prototype (Chapter 4). Firstly, the theoretical evaluation was performed to check if the requirements identified during the development of the thesis are fulfilled. It was observed that most of the requirements were met by the implemented prototype. The second theoretical evaluation identified and listed the limitations of the system. In the second part of the chapter, we perform the implemented prototype evaluation. The first prototype evaluation performed was the automated unit testing of smart contracts to ensure the functionality of the smart contract. The smart contracts of the prototype system passed all the unit tests. The second prototype evaluation performed a scalability evaluation using simulation to see how the system will handle a large number of transactions. We concluded from the simulation that if the system should be scaled up, the performance of the system largely depends on the blockchain implementation used to build the system. Finally, the last quantitative evaluation of the gas consumption for smart contract deployment and the transaction cost of the smart contract functions are listed. This chapter on evaluation answered the fourth research question (RQ4) by analyzing the applicability of the implemented solution to the identified problem of the thesis and also answered about the scalability and feasibility of the solution.

⁶Price as on 16 Aug, 04:38 UTC

6 Conclusion

6.1 Summary

This thesis investigates the applicability and feasibility of using blockchain as a platform to track carbon emissions of manufactured items (such as cars or car components) across an automotive supply chain. Then we attempted to integrate the carbon tracking system with the carbon tax payment mechanism also by leveraging the capabilities of the blockchain.

After introducing the topic, the thesis identified various obstacles in the automotive supply chain that prevent firms from measuring the product carbon footprints of their manufactured goods, such as cars and components. One of the challenges was that the different supply chain participants had their own local database representation and data model, leading to inconsistencies and unavailability. It is also very common that the suppliers do not track the carbon footprints of their products. Also, the under-reporting of the carbon emissions in corporate disclosures was relatively high. These factors fueled the motivation for the investigation and exploration of blockchain as a viable solution for harmonizing carbon emission data from all supply chain participants, such as suppliers and OEMs, and storing them reliably.

Following that, we conducted a literature review to find other blockchain-based traceability systems that trace carbon footprint across supply chains. Also, we explored the literature for carbon price payments using blockchain. Despite the limited number of literature on these areas, the review revealed that academics had previously worked on carbon tracking across the supply chain using blockchain. The literature on blockchain applications for carbon price payment mainly had concentrated on the carbon emissions trading scheme. The usage of private-permissioned blockchain had received little attention in these literatures. Another notion we came across during our literature study was using carbon labeling to identify each product and store their carbon footprint on the blockchain using the identifier on the label. Based on the analysis from the literature review, we conceptualized and designed a unified carbon product footprint tracing and carbon tax payment into a single system for the automotive supply chain (which is a multi-echelon supply chain) where the products are tracked using a carbon label (QR code with a digital identity called DID). All of these are conceptualized using a private-permissioned blockchain. This concept of unifying these approaches into a single system is the main contribution of this thesis.

Since the automotive supply chain is very complex, we made many assumptions about the supply chain. Based on the assumptions and scenario of the system, we defined the objectives of the system, gathered the requirements, and developed the use cases. Blockchain-based software architecture was developed based on the use

6 Conclusion

cases using the “4+1” view model by (Kruchten, 1995). The next stage was to create a prototype utilizing smart contracts on the Ethereum blockchain network, based on the designed software architecture. The prototype is a Dapp that provides interfaces for the supply chain participants to interact with the blockchain. Finally, we performed evaluation tests of the conceptual design and the implemented prototype to show the applicability of the implemented solution to the identified problem and also to validate the scalability and feasibility of the blockchain-based solution.

6.2 Fulfilment of Objectives of the Thesis

In this subsection, we discuss if the objectives of the system defined in Section 3.2.1 are achieved through the implemented system which can be found in Table 6.1.

Objective	Fulfillment
The first objective of the system is that it utilizes a blockchain as infrastructure to support recording carbon emissions reliably at the facilities of both the suppliers and OEM.	To achieve this objective, the system is built over a private Ethereum blockchain which allows both the suppliers and OEM to reliably record their carbon emissions on the blockchain as transactions.
The second objective of the system is to provide a blockchain-based digital identity to the manufactured parts and produced cars which will be attached to them as a QR code.	In the implemented system, we use the DID which is assigned to each and every manufactured parts and produced cars and the DIDs can be generated as a QR code on the web client to be attached on the produced product.
The third objective of the system is that using the blockchain-based digital identity, the records of the carbon footprint of the component or car should be traceable.	In Section 4.4, there are two user interface discussed with the title “Search Component Interface” and “Search Car Details”. The first interface is available to the suppliers to search for the carbon footprint of the product. The second interface is available to the OEMs to search for the carbon footprint of the produced car.
The fourth objective of the system is that the system implements a price on carbon which can be paid using the native cryptocurrency.	We implemented a token based on standard ERC-20 to implement the carbon credit and the this carbon credit allowance can only be bought for the native cryptocurrency. Since, we use Go-Ethereum blockchain, we pay the price using Ethers
The fifth objective of the system is that a user interface should be built to interact with the system and access data on the blockchain.	In Section 4.4, we discuss about all the interfaces built for different roles of the system, like the suppliers, managers and OEMs to interact with the system and access data on the blockchain.
The sixth objective of the system is that the system should be only accessible to the registered participants and they should be able to view transactions according to their permissions.	After a supplier or OEM registers in the system, the managers can validate their membership and allot their carbon credit allowance. If they are not a valid membership, they can be removed from the system. We use the Roles.sol contract from OpenZeppelin to handle the roles in the system.

Table 6.1: Fulfilment of the objectives of the system

6.3 Answers to Research Questions

At the outset of the thesis, we broke down the goal of the thesis into 4 research questions specified in Section 1.2.

Research Question 1: Can blockchain technologies be used for traceability of carbon footprint data for an automobile supply chain and payment of carbon price? What are the requirements of this system?(RQ1)

The first research question looked at the feasibility of employing blockchain technology for carbon footprint data traceability and carbon price payment in the automotive supply chain. To answer the research question and to examine the state-of-the-art systems in the literature that implement such a carbon traceability system in blockchain, we conducted a comprehensive literature review in Section 2.2. The literature review also aimed to identify the gaps, requirements, and evaluation parameters for our proposed system. The review revealed that blockchain technology could be used because of its unique characteristics like immutability, traceability, and reliability. The literature review also helped to identify that it is possible to unify the carbon footprint tracing and carbon tax payment using the carbon credit which a utility token could model on the blockchain. The requirements of the system are identified and answered in Chapter 3.

Research Question 2: What are the approaches of architecture design for the blockchain-based carbon footprint traceability system?(RQ2)

The second research question was designing the system architecture for the proposed unified blockchain-based carbon footprint tracing and carbon tax payment system. Firstly, in Chapter 3, the assumptions about the supply chain were presented, which influenced the design of the system. Based on the scenario, the requirements of the system are identified, and use cases of the system are developed. Many design decisions are also discussed in these sections. Finally, the architecture design of the system was modeled based on the “4+1” view model by (Kruchten, 1995).

Research Question 3: How can the designed system architecture be implemented as a blockchain application?(RQ3)

Research question 3 looks at the implementation details of the designed system architecture. In Chapter 4 the implementation details of the decentralized application are explained in detail. The decisions about which blockchain should be used, if the system requires a backend and frontend, are presented. The implementation decisions while implementing different system components, such as the smart contracts, are discussed in detail.

Research Question 4: How to evaluate the applicability and feasibility of the implemented solution?(RQ4)

Research question 4 deals with the evaluation of the applicability and feasibility of the implemented system. In Chapter 5, we evaluate the implemented system in two different evaluation methods. The first is the theoretical evaluation, where we assess if the implemented system fulfils the research questions, objectives defined during the thesis and the requirements of the system, along with the limitations of the implemented system. The second form of evaluation is an implemented prototype evaluation, where we perform unit tests of the smart contracts, quantitative evaluation, and the simulation of the system.

6.4 Future work

6.4.1 Improvements

Different aspects could be improved upon this work, which forms the basis for future work. Some of these improvements were either out of scope or difficult to finish in this thesis. The section on the list of limitations (Section 5.1.2) will serve as the foundation for future work. The first part of the thesis with the scope of improvement is that our system did not consider the entire automotive supply chain, including the upstream, manufacturing, and downstream activities. The methodologies and findings of this study can be extended in the future to the downstream phase of the automotive supply chain and can also be extended to include the other life cycle phases like the use phase, recycling phase, and disposal phase for estimating the total life cycle carbon footprint of an automobile. The second area with the scope of improvement is that the human technicians currently enter the carbon footprint data in this system in the suppliers or OEMs. Instead, we propose deploying a set of sensors and IoT devices on-site to directly measure and collect carbon emissions data and send it to the blockchain instead of a human technician. Chances of errors can also be reduced. The third area with the scope of improvement that we propose is to explore the possibility of using the public blockchain instead of the private blockchain and to store the private information about the companies off-chain indexed by their assigned DID identity, and they can interact with the public blockchain using their DID identity for storing the carbon emission data. This guarantees their anonymity, and it is also feasible to verify if the firm is registered on the blockchain using the DID resolver, and the Dapp will have access to all of the capabilities of a public blockchain like Ethereum mainnet in this situation.

6.4.2 Possible Future Works

Blockchain technology is not limited to the realm of computer science and is a multidisciplinary subject with a flexible ecosystem that can incorporate mathematical concepts, cryptography, relevant guidelines and laws, finance, and rewarding mechanisms. Thus developing a deep understanding of different domains, followed by consistently

integrating these areas, provides much scope for multidisciplinary research. Thus, gaining a comprehensive grasp of these fields followed by appropriately integrating them opens up many possibilities for blockchain applications.

A future work that may be inspired from this thesis could be to integrate other parts of the supply chain such as the upstream, manufacturing, downstream, recycling, use phase, recycling, disposal into the system for an end to end fair trade, transparent carbon footprint tracking, carbon price payment and traceable ledger for efficient e-waste management. Another future work could also explore using a hybrid blockchain in a similar traceability system in a supply chain scenario like an automotive supply chain.

Another extended application of blockchain demonstrated can address several global and local environmental challenges such as recycling of wastes, climate change, and energy. This is due to the capability of blockchain to generate transparent record-keeping and value-based transactions using the tokenized ecosystem. A good example is the blockchain application that can be developed in the future to offer a reward in the form of cryptocurrency for the sustainable recycling of used plastics and e-waste. Such an ecosystem will encourage people to deposit plastics and e-waste in exchange for currency with improved sustainability.

A few areas that have the potential for blockchain technology to promote sustainability are mentioned above. However, the technology needs to fill several technical gaps to realize its full potential to provide scalable solutions. The financial sector is an early adopter, and the potential is established well. In many of the vital sectors, the proof of concept is established. However, blockchain is still in the early stages of development, and several areas still remain unexplored when it comes to deploying blockchain solutions providing broad scope for future research.

Bibliography

- Addison, Nick (2021), “Solidity 2 UML [online].” URL <https://github.com/naddison36/sol2uml>. Original-date: 2019-01-29T11:11:51Z.
- Androulaki, Elli, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick (2018), “Hyperledger fabric: a distributed operating system for permissioned blockchains.” In *Proceedings of the Thirteenth EuroSys Conference, EuroSys ’18*, 1–15, Association for Computing Machinery, New York, NY, USA, URL <https://doi.org/10.1145/3190508.3190538>.
- Angelo, Monika and Gernot Salzer (2020), “Tokens, Types, and Standards: Identification and Utilization in Ethereum.” In *2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, 1–10, URL https://www.researchgate.net/publication/339843828_Tokens_Types_and_Standards_Identification_and_Utilization_in_Ethereum.
- Ant-design (2019), “Ant Design - The world’s second most popular React UI framework [online].” URL <https://ant.design/>. (Last accessed on 2021-07-31).
- Banerjee, A (2018), “Re-engineering the carbon supply chain with blockchain technology.” *Infosys Ltd.* <http://www.infosys.com/oracle/white-papers/documents/carbon-supply-chain-blockchain-technology.pdf>.
- Bashir, Imran (2020), *Mastering Blockchain 3rd Edition*. URL <https://github.com/ethereumbook/ethereumbook>.
- Bhatia, Pankaj, Cynthia Cummis, Laura Draucker, David Rich, Holly Lahd, and Andrea Brown (WBCSD) (2011), “Greenhouse Gas Protocol Product Life Cycle Accounting and Reporting Standard.” URL <https://www.wri.org/research/greenhouse-gas-protocol-product-life-cycle-accounting-and-reporting-standard>.
- Brunner, Clemens, Ulrich Gellersdörfer, Fabian Knirsch, Dominik Engel, and Florian Matthes (2020), *DID and VC: Untangling Decentralized Identifiers and Verifiable Credentials for the Web of Trust*, 61–66. Association for Computing Machinery, New York, NY, USA, URL <https://doi.org/10.1145/3446983.3446992>.

- Böttcher, Müller Martin, Christian (2013), “Drivers, Practices and Outcomes of Low-carbon Operations: Approaches of German Automotive Suppliers to Cutting Carbon Emissions.” *Business Strategy and the Environment*, 24, URL <https://onlinelibrary.wiley.com/doi/epdf/10.1002/bse.1832>.
- Buterin, Vitalik (2013), “Ethereum Whitepaper.” URL <https://ethereum.org>.
- Buterin, Vitalik (2014), “A next-generation smart contract and decentralized application platform.” *white paper*, 3, URL https://cryptorating.eu/whitepapers/Ethereum/Ethereum_white_paper.pdf.
- Chase, Ethan, MacBrough (2018), “Analysis of the XRP Ledger Consensus Protocol.” *arXiv:1802.07242 [cs]*, URL <http://arxiv.org/abs/1802.07242>. ArXiv: 1802.07242.
- Chen, Xu, Huan Yang, Xiaojun Wang, and Tsan-Ming Choi (2020), “Optimal carbon tax design for achieving low carbon supply chains.” *Annals of Operations Research*, URL <https://doi.org/10.1007/s10479-020-03621-9>.
- Chowdhury, Mohammad, Alan Colman, Ashad Kabir, Jun Han, and Paul Sarda (2018), *Blockchain Versus Database: A Critical Analysis*. URL <https://ieeexplore.ieee.org/document/8456055>. Pages: 1353.
- Christidis, Konstantinos and Michael Devetsikiotis (2016), “Blockchains and Smart Contracts for the Internet of Things.” *IEEE Access*, 4, 2292–2303, URL <https://ieeexplore.ieee.org/document/7467408>. Conference Name: IEEE Access.
- Collomosse, John (2020), “Technical overview of blockchain and distributed ledger technology (dlt) -lecture slides.” URL https://www.youtube.com/watch?v=N95h2e_tUes.
- Diem (2020), “Diem official white paper.” URL <https://www.diem.com/en-us/white-paper/>.
- Drummond Reed, Dave Longley Christopher allen Ryan Grant Markus Sabadello, Manu Sporny (2019), “Decentralized Identifiers (DIDs) v1.0.” URL <https://www.w3.org/TR/did-core/>.
- EIP-1056 (2018), “EIP-1056: Ethereum Lightweight Identity [online].” URL <https://eips.ethereum.org/EIPS/eip-1056>.
- EIP-20 (2015), “EIP-20: ERC-20 Token Standard [online].” URL <https://eips.ethereum.org/EIPS/eip-20>.
- Gallersdörfer, Ulrich, Patrick Holl, and Florian Matthes (2020), “Blockchain-based systems engineering – lecture slides.” URL <https://github.com/sebischair/bbse>.
- Ganache (2018), “Ganache [online].” URL <https://trufflesuite.com/ganache>. (Last accessed on 2021-07-31).

Bibliography

- Gerdes, Justin (2012), “Major Corporation Making Transportation Shifts to Cut Back On Costs & Carbon Footprint ...” URL <http://mangoworldmagazine.blogspot.com/2012/02/major-corporations-making.html>.
- Go-Ethereum (2016), “Go Ethereum [online].” URL <https://geth.ethereum.org/>. (Last accessed on 2021-07-31).
- Hagmann, David, Emily H. Ho, and George Loewenstein (2019), “Nudging out support for a carbon tax.” *Nature Climate Change*, 9, 484–489, URL <https://www.nature.com/articles/s41558-019-0474-0>.
- Hong, Deukjo, Donghoon Chang, Jaechul Sung, Sangjin Lee, Seokhie Hong, Jaesang Lee, Dukjae Moon, and Sungtaek Chee (2006), “A new dedicated 256-bit hash function: Fork-256.” In *Fast Software Encryption, 13th International Workshop, FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, 195–209, Springer, URL <https://iacr.org/archive/fse2006/40470198/40470198.pdf>.
- Iuon-Chang, Liao, Lin (2017), “A Survey of Blockchain Security Issues and Challenges.” *International Journal of Network Security*, 19, URL https://www.researchgate.net/publication/329683614_A_survey_of_blockchain_security_issues_and_challenges.
- JavaScript (2008), “JavaScript | MDN [online].” URL <https://developer.mozilla.org/en-US/docs/Web/JavaScript>.
- Jensen, Jesper (2012), “Product carbon footprint developments and gaps.” *International Journal of Physical Distribution & Logistics Management*, 42, 338–354, URL https://www.researchgate.net/publication/235321965_Product_carbon_footprint_developments_and_gaps.
- JSX (2013), “Introducing JSX – React [online].” URL <https://reactjs.org/docs/introducing-jsx.html>.
- Khovratovich, Jason, Dmitry (2016), “Sovrin: digital identities in the blockchain era.” 5, URL <https://sovrin.org/wp-content/uploads/AnonCred-RWC.pdf>.
- Kruchten, Philippe (1995), “The 4+1 view model of architecture.” *IEEE Software*, 12, 45–50, URL https://www.researchgate.net/publication/220018231_The_41_View_Model_of_Architecture.
- Lee, Ki-Hoon (2011), “Integrating carbon footprint into supply chain management: the case of Hyundai Motor Company (HMC) in the automobile industry.” *Journal of Cleaner Production*, 19, 1216–1223, URL <https://www.sciencedirect.com/science/article/pii/S0959652611000862>.
- Lee, Ki-Hoon (2012), “Carbon accounting for supply chain management in the automobile industry.” *Journal of Cleaner Production*, 36, 83–93, URL <https://www.sciencedirect.com/science/article/pii/S0959652612000996>.

- Liss, Florian (2018), *Blockchain and the EU ETS: An architecture and a prototype of a decentralized emission trading system based on smart contracts*. Ph.D. thesis, URL https://www.researchgate.net/publication/328354712_Blockchain_and_the_EU_ETS_An_architecture_and_a_prototype_of_a_decentralized_emission_trading_system_based_on_smart_contracts.
- Liu, Hu-Chen and Xiao-Yue You (2021), *Green Supplier Evaluation and Selection: Models, Methods and Applications*. Springer Singapore, URL <https://www.springer.com/gp/book/9789811603815>.
- Liu, Kun-Hsing, Shih-Fang Chang, Wun-Hui Huang, and I-Ching Lu (2019), “The Framework of the Integration of Carbon Footprint and Blockchain: Using Blockchain as a Carbon Emission Management Tool.” In *Technologies and Eco-innovation towards Sustainability I: Eco Design of Products and Services* (Allen H. Hu, Mitsutaka Matsumoto, Tsai Chi Kuo, and Shana Smith, eds.), 15–22, Springer Singapore, Singapore, URL https://doi.org/10.1007/978-981-13-1181-9_2.
- Material-UI (2014), “Material-UI: A popular React UI framework[online].” URL <https://material-ui.com/>. (Last accessed on 2021-07-31).
- Metamask (2016), “MetaMask [online].” URL <https://metamask.io/>. (Last accessed on 2021-07-31).
- Miehle, Daniel, Dominic Henze, Andreas Seitz, Andre Luckow, and Bernd Bruegge (2019), “PartChain: A Decentralized Traceability Application for Multi-Tier Supply Chain Networks in the Automotive Industry.” In *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, 140–145, URL <https://ieeexplore.ieee.org/document/8783173>.
- Miehle, Daniel Simon (2020), “Distributed Ledger Technologies in the Automotive Value Chain.” 102, URL http://mediatum.ub.tum.de/doc/1545336/mj2y9fx5s19pu6s4o42kdx8nc.20201206_Diss_Daniel_Miehle.pdf.
- Nakamoto, Satoshi (2008), “Bitcoin: A Peer-to-Peer Electronic Cash System.” 9, URL <https://bitcoin.org/bitcoin.pdf>.
- Node.js (2009), “Node.js [online].” URL <https://nodejs.org/en/>. (Last accessed on 2021-07-31).
- Oliver, John E. (2005), “Kyoto Protocol.” In *Encyclopedia of World Climatology* (John E. Oliver, ed.), 443–443, Springer Netherlands, Dordrecht, URL https://doi.org/10.1007/1-4020-3266-8_118.
- OpenZeppelin (2018), “OpenZeppelin [Online].” URL <https://openzeppelin.com/>.
- Panait Drăgnoiu, Andreea, Ruxandra Olimid, and Alin Stefanescu (2020), “Analysis of uPort Open, an identity management blockchain-based solution.”

Bibliography

- Paris Agreement, United Nations (2015), “United nations treaty collection, chapter xxvii 7. d.” URL https://treaties.un.org/pages/ViewDetails.aspx?src=TREATY&mtdsg_no=XXVII-7-d&chapter=27&clang=_en.
- Patel, Dhiren, Benita Britto, Sanidhya Sharma, Kaustubh Gaikwad, Yash Dusing, and Mrinal Gupta (2020), “Carbon Credits on Blockchain.” In *2020 International Conference on Innovative Trends in Information Technology (IC-ITIIT)*, 1–5, URL https://www.researchgate.net/publication/340813301_Carbon_Credits_on_Blockchain.
- Pellen-Pickersgill, Jack, Raja Naeem Akram, and Konstantinos Markantonakis (2019), “Carbon Labelling - Blockchain based product carbon footprint system.” 1, URL https://scc.rhul.ac.uk/files/2019/09/2018-Jack_Pellen-Pickersgill_ComputerScienceSoftware-EngineeringWithYearInIndustry.pdf.
- React (2013), “React – A JavaScript library for building user interfaces [online].” URL <https://reactjs.org/>. (Last accessed on 2021-07-31).
- Remix (2018), “Remix - Ethereum IDE [online].” URL <http://remix.ethereum.org/>. (Last accessed on 2021-07-31).
- Rosado da Cruz, A. M., Francisco Santos., Paulo Mendes., and E. Cruz (2020), “Blockchain-based traceability of carbon footprint: A solidity smart contract for ethereum.” In *Proceedings of the 22nd International Conference on Enterprise Information Systems - Volume 2: ICEIS*, 258–268, INSTICC, SciTePress, URL <https://www.scitepress.org/Link.aspx?doi=10.5220/0009412602580268>.
- Rouhani, Sara and Ralph Deters (2017), “Performance analysis of ethereum transactions in private blockchain.” In *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 70–74. ISSN: 2327-0594.
- Solidity (2014), “Solidity — Solidity 0.8.6 documentation [online].” URL <https://docs.soliditylang.org/en/v0.8.6/>. (Last accessed on 2021-07-31).
- Stephan, Benjamin, Insung Lee, and Jiseok Kim (2019), *Crashing the climate - How the car industry is driving the climate crisis*. URL https://www.greenpeace.de/sites/www.greenpeace.de/files/publications/gp_cleanairnow_carindustryreport_full_v5_0919_72ppi_0.pdf.
- Szabo, Nick (1997), “Formalizing and Securing Relationships on Public Networks.” *First Monday*, 2, URL <https://firstmonday.org/ojs/index.php/fm/article/view/548>.
- Truffle (2018), “Truffle framework for ethereum development [online].” URL <https://trufflesuite.com/truffle>. (Last accessed on 2021-07-31).
- Voshmgir, Shermin (2019), “Token Economy - Second Edition.” URL <https://blockchainhub.net/blog/blog/token-economy-book-shermin-voshmgir/>.

- Warski, Adam (2019), “Comparing Ethereum and the Libra blockchain.” URL <https://blog.softwaremill.com/comparing-ethereum-and-the-libra-blockchain-64bec7dd70c0>.
- Waters, C. D. J. (2003), *Logistics: an introduction to supply chain management*. Palgrave Macmillan, Houndmills, Basingstoke, Hampshire ; New York, URL https://juancarlosvergaras.files.wordpress.com/2013/06/waters_d-logisticsc_an_introduction_to_supply_chain_management_2003en354s.pdf.
- Web3.js (2014), “web3.js - Ethereum JavaScript API — web3.js 1.0.0 documentation [online].” URL <https://web3js.readthedocs.io/en/v1.4.0/>. (Last accessed on 2021-07-31).
- Weimert, Birgit, Wolfgang Prinz, Nils Urbach, Steffen Holly, Axel Schulte, Gilbert Fridgen, Thomas Rose, Julian Schütte, Mathias Dalheimer, Markus Wenzel, Boris Otto, Christian Schwede, Ulrich Leiner, Michael Fritz, Michael Kreutzer, Alexander Nouak, Thomas Hoeren, Christian Welzel, Philipp Sprenger, and Nikolas Guggenberger (2018), *BLOCKCHAIN AND SMART CONTRACTS - Technologies, research issues and applications*. URL https://www.researchgate.net/publication/325846979_BLOCKCHAIN_AND_SMART_CONTRACTS_-_Technologies_research_issues_and_applications.
- Werner, Steve and Bolton Brian (2018), “Corporate Carbon Disclosure in North America.” 7, URL <https://www.spglobal.com/spdji/en/documents/research/research-corporate-carbon-disclosure-in-north-america.pdf>.
- Wiedmann, Thomas and Jan Minx (2008), “A Definition of Carbon Footprint.” *CC Pertsova, Ecological Economics Research Trends*, 2, 55–65, URL https://www.researchgate.net/publication/247152314_A_Definition_of_Carbon_Footprint.
- Wohrer, Maximilian and Uwe Zdun (2018), “Smart contracts: security patterns in the ethereum ecosystem and solidity.” In *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, 2–8, URL <https://ieeexplore.ieee.org/document/8327565>.
- Wood, Gavin (2014), “Ethereum: A secure decentralised generalised transaction ledger.” URL <https://ethereum.github.io/yellowpaper/paper.pdf>.
- Wood, Gavin and Andreas M Antonopoulos (2018), “Mastering ethereum: Building smart contracts and dapps.” URL <https://www.oreilly.com/library/view/mastering-ethereum/9781491971932/>.
- Wüst, K. and Arthur Gervais (2018), “Do you Need a Blockchain?” *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, URL <https://ieeexplore.ieee.org/document/8525392>.

Bibliography

- Xu, Xiwei, Ingo Weber, Mark Staples, Liming Zhu, Jan Bosch, Len Bass, Cesare Pautasso, and Paul Rimba (2017), “A Taxonomy of Blockchain-Based Systems for Architecture Design.” In *2017 IEEE International Conference on Software Architecture (ICSA)*, 243–252, URL <https://ieeexplore.ieee.org/document/7930224>.
- Yang, Lei, Chenshi Zheng, and Minghui Xu (2014), “Comparisons of low carbon policies in supply chain coordination.” *Journal of Systems Science and Systems Engineering*, 23, 342–361, URL <https://doi.org/10.1007/s11518-014-5249-6>.
- Zheng, Zibin, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang (2017), “An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends.” URL https://www.researchgate.net/publication/318131748_An_Overview_of_Blockchain_Technology_Architecture_Consensus_and_Future_Trends.