



Bilkent University

Department of Computer Engineering

Senior Design Project

Project Name: VerifiChain

Analysis Report

Tuna Öğüt - 21803492
Erengazi Mutlu - 21803676
Bartu Teber - 21703460
Ömer Onat Postacı - 21802349
Arda Güven Çiftçi - 21801763

Supervisor: Özgür Ulusoy

Jury Members: Erhan Dolak, Tağmaç Topal

Innovation Expert: Nazan Kurt

October 17, 2022

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

	2
1. Introduction	3
1.1 Description	3
2. Proposed System	5
2.1 Overview	5
2.2 Functional Requirements	6
2.2.1 QR Code Execution and Product Matching	6
2.2.2 Product Transfer and Displaying Transaction History	6
2.2.3 Wallet Integration and Displaying All Products	7
2.2.4 Fake Product Report	7
2.2.5 User Reviews	7
2.3 Non-Functional Requirements	8
2.3.1 Accessibility	8
2.3.2 Availability	8
2.3.3 Backup and Recovery	8
2.3.4 Extensibility	8
2.3.5 Data Integrity	9
2.3.6 Security	9
2.3.7 Exception Handling	9
2.3.8 Usability	9
2.3.9 Maintainability	10
2.3.10 Transparency	10
2.3.11 Legal and Regulatory Requirements	10
2.3.12 Integrability	10
2.3.13 Response Time	10
2.4 Pseudo Requirements	11
2.5 System Models	11
2.5.1 Scenarios	11
2.5.2 Use-Case Model	14
2.5.3 Object and Class Model	15
2.5.4 Dynamic Models	16
2.5.5 User Interface	19
3. Other Analysis Elements	30
3.1 Consideration of Various Factors in Engineering Design	30
3.2 Risks and Alternatives	30
3.3 Project Plan	31
3.4 Ensuring Proper Teamwork	36
3.5 Ethics and Professional Responsibilities	36
3.6 Planning for New Knowledge and Learning Strategies	37
4. Glossary	38
5. References	39

1. Introduction

In recent decades, the production and sale of counterfeit goods has dramatically expanded. The economic situation of manufacturers, suppliers, and distributors, as well as the welfare of customers, are all significantly impacted by this growth. As a result, businesses may experience issues such as diminished customer satisfaction and trust, increased budgets devoted to thwarting fake goods, and economic shrinkage brought on by sales that are likely to decline over time. Meanwhile, consumers may experience difficulties such as health issues that may be brought on by subpar products, a lack of value for their money, and additional costs that may be brought on over time by subpar and fake goods [1]. Although there are QR code-based solutions to address this issue by displaying the product details, these solutions can still be taken advantage of by employing other methods, such as duplicating the QR codes.

By incorporating blockchain and QR code technology into the product supply chain and manufacturing process to improve the traceability and accuracy of product verification, VerifiChain seeks to minimize this issue. VerifiChain's ability to establish an immutable transaction flow in the supply chain by connecting each product with an owner in accordance with blockchain technology sets it apart from other QR code-based solutions. The purpose of this technique is to boost traceability by separating each product from the others using QR code technology.

1.1 Description

VerifiChain is a web application that uses blockchain and QR code technologies. It offers product verification so that people may check the authenticity of the goods they already own or seek to purchase. The method verification mechanism operates is as follows: the product's manufacturer will generate a QR code for the product. At the completion of the manufacturing process, the system will link each product to its individual QR code. Blockchain technology will be used to create the linkage. The

security characteristics that blockchain technology offers and the wallet system that is used are the main reasons for utilizing it. Users can claim ownership of the product using wallet technology. The digital credentials for the product are added to the user's wallet when they purchase it. As a result, the owner of that particular product will be visible whenever a consumer scans the QR code of the product. It will increase the system's security by another level. A product will initially appear to belong to its producer until it is moved to a supplier, user, or distributor. A blockchain transaction will be used to complete the sale, and the product will then be assigned to the buyer until another buyer purchases it. Users should make use of the offered system if they want to sell or acquire goods. Customers can ensure that a product is genuine by using the system. Additionally, customers can guarantee that the goods is not stolen thanks to blockchain technology, which makes it impossible to sell anything you don't own or don't have in your wallet. As a result, anyone trying to resell a product that has been stolen or is false will be found out. If the product is stolen, it will be shown as stolen since it will be assigned to the wallet of another user. The QRcode, on the other hand, won't contain the credentials that show its legitimacy if the goods are phony, making it obvious that it is. This function will also lessen the chances of purchasing fraudulent goods from unreliable vendors. Customers can quickly check the products' legitimacy by scanning the QR codes on them.

While recognized businesses and brands can create distinctive QR codes and link them to certain products, the system will give an interface that only lets consumers buy, sell, monitor, and verify the authenticity of products. Thus, if a product is found on the system, it will be confirmed to have been produced by an accredited company.

Therefore, VerifiChain provides techniques that are safer to buy and sell things by enabling consumers to see the owner and the authenticity of the products. This reduces the risks associated with buying and selling counterfeit or stolen goods.

2. Proposed System

2.1 Overview

VerifiChain will be a web application that utilizes blockchain technology to provide a more transparent and traceable supply chain history for a product. Manufacturers, suppliers, distributors, and consumers can reach every detail of the history of a product using VerifiChain and transparency will be ensured. As a result, VerifiChain will provide all transfer information beginning from manufacturing to the current status of each product and thus counterfeitness will be identified by tracking the history of each product. Manufacturers will have a MetaMask wallet and execute a unique ERC-721 token for each product that includes a URI-encrypted QR code containing the product-specific transfer history. Hereby, they can create an ERC-721 token that belongs to their token and can be seen in EtherScan (a transparent website that provides information for each token and address provided by Ethereum). After creating a product (ERC-721 token) using VerifiChain, manufacturers will transfer the product to the unique wallet address of suppliers, and at every step, VerifiChain will keep track of the information of actions. Suppliers can transfer the product to each other, or they can transfer the product to the distributors. Again, in each step, the information on actions can both be seen in EtherScan, and VerifiChain will keep track of this information. Consumers can scan the QR code of each product and see the information on actions for each product. There might be counterfeitness if there is a mismatch between the information stored in VerifiChain and the information received from other sources. Hereby, transparency and traceability will be ensured for all actors of the application.

VerifiChain will consist of 3 parts, which are respectively; a database to store persistent data, a front-end part to interact with users, Ethereum chain, and back-end service, lastly a back-end part to ensure a bridge between the database, front-end, and smart contracts.

2.2 Functional Requirements

VerifiChain is planned to meet the following functionalities:

2.2.1 QR Code Execution and Product Matching

In VerifiChain, all products will be associated with a single, unique QR code. For reliability and traceability reasons, the access to generate a unique QR code and to match the QR code with a product is given to the manufacturer. Authorizing just the manufacturer to generate a QR code increases the reliability and traceability of VerifiChain since:

- The manufacturer is the most reliable stakeholder to generate a QR code since it is directly related to the brand. Suppliers and distributors may manipulate the authorization of generating QR codes.
- Generating QR codes at the first step of the supply chain increases the traceability of VerifiChain since a supply chain of the product from the manufacturer to a distributor is fully traceable.

2.2.2 Product Transfer and Displaying Transaction History

The products in VerifiChain will be transferred via their QR code. The transactions will execute in the blockchain. The possession of the QR code will be transferred from one user to another user. Using blockchain increases the reliability and traceability of transactions. Every node in the blockchain will be able to see the full transaction history of the product and the history is unalterable. For example, the manufacturer will transfer all products to the supplier and the supplier will transfer all products to the distributor. If the products are not sold in the season, the distributor will transfer these products to the outlet store. Therefore, when a customer goes to an outlet store and scans the QR code,

the customer will be able to see the full transaction history of the product along with if the product was really on the market the previous season even with the information in which store it was on market. Moreover, VerifiChain will allow users to sell products peer to peer on second hand along with a high assurance of originality. However, the main vision of VerifiChain is to make the supply chain of products reliable and traceable rather than handling second-hand product selling. Therefore, the features of second-hand product selling will be limited and basic on delivery but it will be possible in the future to scale the features of second-hand product selling.

2.2.3 Wallet Integration and Displaying All Products

A MetaMask wallet will be integrated with VerifiChain. The stakeholders in the supply chain will be able to list all the products they have and how many of them they have. Moreover, a customer will also be able to see which products they purchase. The QR code of every product will also be available on the product list.

2.2.4 Fake Product Report

In doubtful circumstances, every user regardless of having the product can report the product that can be fake. For example, let someone enter a shop and scan the QR code of a product. Then, he/she sees that the product is supposed to be in another shop in the transaction history. In this doubtful circumstance, if he/she reports the product, that product will be labeled as suspicious and will be investigated.

2.2.5 User Reviews

VerifiChain gives value to the opinion of its customers and uses their feedback for improvement and bug tracking. For this purpose, VerifiChain periodically gets reviews and feedback from users. For example, once a month after the subscription, a pop-up requesting user feedback appears on the window.

2.3 Non-Functional Requirements

2.3.1 Accessibility

- Since as a third-party application MetaMask Wallet will be used, the web browser that VerifiChain is running on must be able to support the requirements of MetaMask Wallet. Therefore, Chromium-based web browsers such as Google Chrome and Brave could be used as well as Mozilla Firefox because such web browsers are integrated with MetaMask Wallet [3].
- React.js will be used for the front-end development of VerifiChain.
- Node.js will be used for the back-end development of VerifiChain.

2.3.2 Availability

- VerifiChain will be operable in any location with an internet connection.
- Operations in the software systems of VerifiChain will be done during periods of least intensity of use to make the application optimally available for the users.

2.3.3 Backup and Recovery

- VerifiChain will use Ethereum Chain for the main chain to store transaction history for each product; therefore, Ethereum Chain undertakes backup and recovery of the transaction history data for each product.
- MetaMask Wallet has its own recovery procedure for each user [5].
- The database of the VerifiChain is PostgreSQL; therefore, the backup and recovery are undertaken by the database service of VerifiChain. Personal Data will be stored in both global and local storage systems to prevent data loss.

2.3.4 Extensibility

- VerifiChain must be open to developments for adopting new features with new functionalities such as a second-hand exchange.

- Other blockchains such as the Solana chain, can be integrated along with Ethereum.

2.3.5 Data Integrity

- Since the system will use Ethereum Chain to store transfer and transaction data for each product, the maintenance and the assurance of the data accuracy is undertaken by Ethereum Chain and be visible in EtherScan [2].

2.3.6 Security

- There are mandatory private credentials for the users to be signed up, connect their wallets, and make a transaction.
- Since VerifiChain uses blockchain technologies, the data storage system will be decentralized in terms of wallet information. Therefore, a more secure system is provided compared to a centralized data storage system.

2.3.7 Exception Handling

- If any error occurs during the execution of the program, users will be informed in regards to what is the error, why the error occurs, and how to solve the error. Hereby, users will be guided to handle the exceptions during a flow of actions.

2.3.8 Usability

- The GUI of the VerifiChain will be clear and user-friendly to direct the users to take accurate actions.
- Users will be able to score determined parameters and convey their thoughts via a feedback system.

2.3.9 Maintainability

- Integrated sub-systems will work harmoniously to provide functionality as complementary factors; therefore, the maintenance of VerifiChain can be ensured.
- Since each sub-system will serve a goal, new sub-systems can be added to the cumulative system and maintained easily.

2.3.10 Transparency

- VerifiChain is a web3-based application. Transactions are kept under the Ethereum network. Therefore, all of the product's transaction history will be visible in both VerifiChain and EtherScan in which all transactions on the Ethereum chain are visible [2].

2.3.11 Legal and Regulatory Requirements

- A contract signed by the users will prevent exploiting the application for their own benefit.

2.3.12 Integrability

- VerifiChain uses a third-party application, which is MetaMask Wallet, to integrate users into the Ethereum blockchain.
- Ethereum Chain will be used as an integrated blockchain to store data and interact with MetaMask Wallet.
- Such sub-systems will be brought together into the system of VerifiChain.

2.3.13 Response Time

- Since VerifiChain uses the Ethereum blockchain, Ethereum's transactions per second will apply to all the transactions. Therefore, tps relies on Ethereum's tps speed.

2.4 Pseudo Requirements

- Version Control

Git will be used for version checking and tracking.

- Implementation Language

VerifiChain will be a web application using React software and Ethers framework (a Web3 Framework) for the front-end implementation.

REST service of VerifiChain will be implemented using Node.js environment for creating the bridge between the front-end and database

PostgreSQL database will be used for storing, processing, and managing the persistent data.

- Frameworks, Libraries, External API's
- Target Platform

2.5 System Models

2.5.1 Scenarios

Scenario 1: Sign Up

Actors: User

Entry Condition: The user clicks the sign up button on the home page.

Exit Condition: The user clicks the cancel button to exit the register page

The Flow of Events:

1. Click the sign up button on the home page.
2. Page for registration opens.
3. User enters the required information.
4. User selects one of the boxes depending on the type of account (manufacturer/receiver).
5. User click the sign up button.

6. System enters the information to the database.

Scenario 2: Log In

Actors: User

Entry Condition: User click on the login button on the home page.

Exit Condition: User closes the app.

The Flow of Events:

1. User enters the required information to login.
2. Clicks on the login button.
3. System checks the required information and matches with the database.
4. User enters the application.

Scenario 3 Scan QR Code

Actors: User

Entry Conditions: User scans the QR.

Exit Condition: Approves transaction.

The Flow of Events:

1. User sees the approved transaction notification.
2. User approves the transaction.
3. Metamask Wallet opens up.
4. If the funds are sufficient, the transaction completes.

Scenario 4: Execute QR Code

Actors: Manufacturer

Entry Conditions: Manufacturer user type clicks on the execute QR code button.

Exit Conditions: QR code occurs.

The Flow of Events:

1. Manufacturer clicks on the execute QR code button.
2. Manufacturer enters how many unique QR codes they want.
3. Manufacturer receives the QR code(s) to his wallet.
4. Manufacturer navigates to the home page.

Scenario 5: Connecting Metamask Wallet

Actors: User

Entry Conditions: User clicks on the connect wallet button.

Exit Conditions: User clicks on the Metamask Wallet icon.

The Flow of Events:

1. User clicks on the connect wallet button on the homepage.
2. User connects Metamask Wallet to the application.

Scenario 6: Add For Sale Tag

Actors: User

Entry Conditions: User clicks on the add sale tag button.

Exit Conditions: User clicks on the add for sale page's confirm button.

The Flow of Events:

1. User clicks on the add sale button on the home page.
2. User selects which QR code they would like to sell.
3. User sets a price tag on the specific QR code.
4. User confirms the selected items to be sold.
5. User navigates to the home page.

2.5.3 Object and Class Model

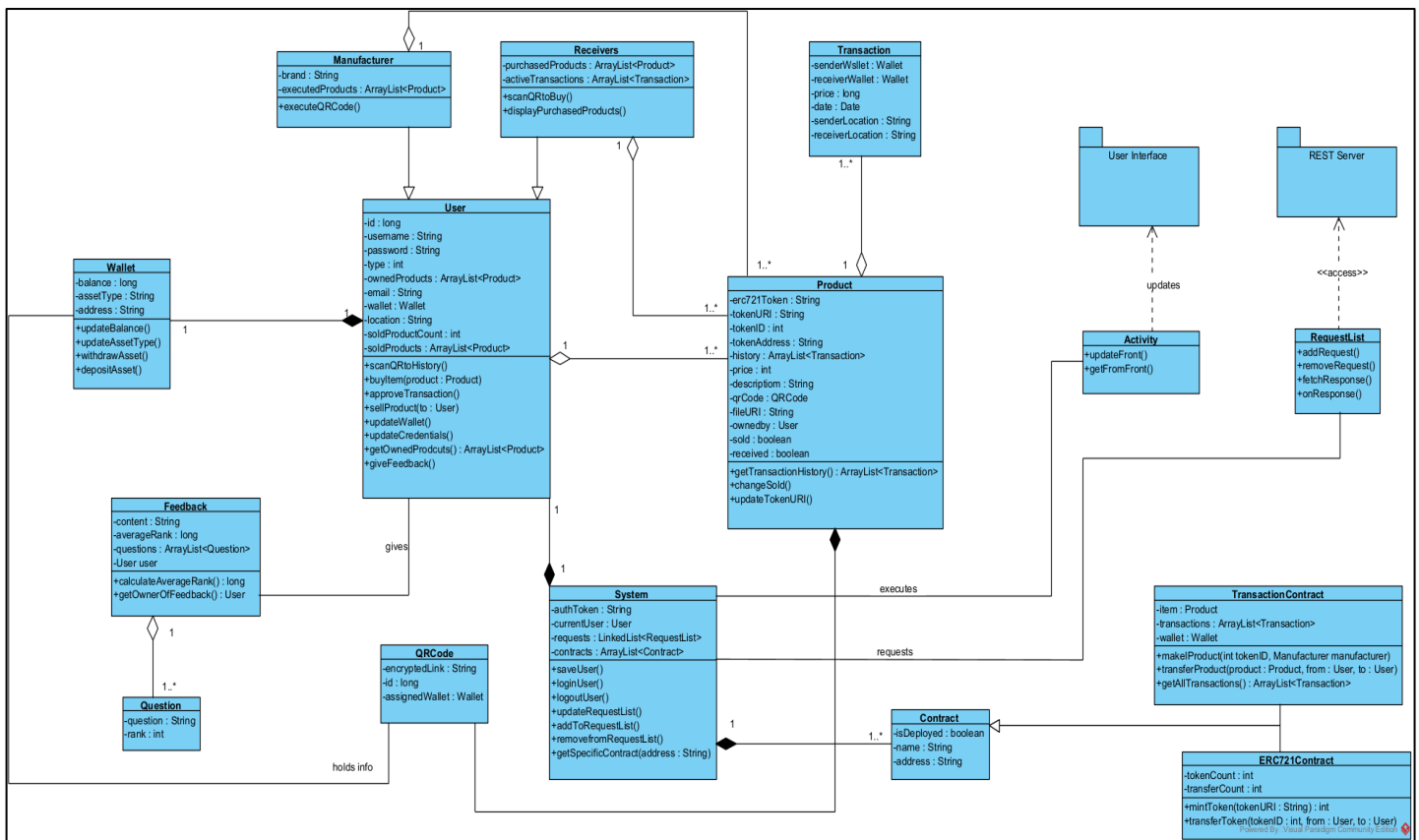


Figure 2: Object and Class Model of VerifiChain System

2.5.4 Dynamic Models

2.5.4.1 Supply Chain Activity Diagram

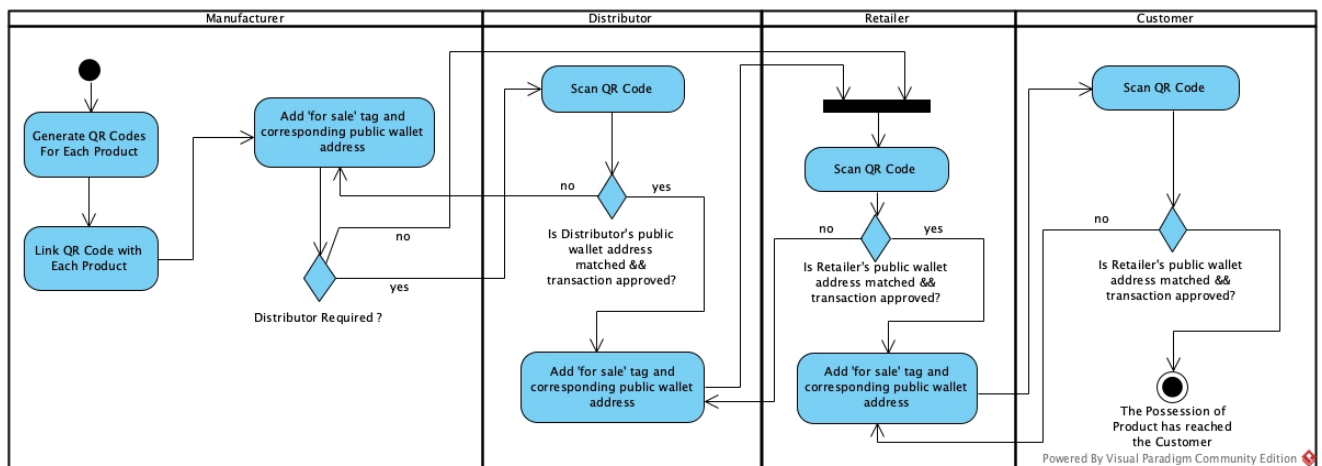


Figure 3: Activity Diagram of Supply Chain in VerifiChain

The activity diagram above demonstrates the supply chain process from the manufacturer to the customer. The diagram does not include customer to customer transaction which is a second hand trading rather than a supply chain process. Initially, the manufacturer produces the QR code for each product and links the QR codes for each product. In the supply chain, the manufacturer is supposed to sell the product either to the distributor or directly to the retailer. In both options, the manufacturer tags the product as 'for sale' and adds the receiver's public wallet address. Then, when the receiver receives the product, the QR code is scanned, if the receiver's public wallet address and the product's destination wallet address matches and the receiver approves the transaction, the transaction occurs immediately and the owner of the product is also transferred in the blockchain. Note that the owner does not change when the product is sent, the owner changes when the receiver receives the product and approves the transaction. If the receiver receives the product and does not approve the transaction, the owner of the product is still unchanged, and receiver is legally obliged to resend the product. The process of approving the transaction, labeling the product 'for sale' and

adding destination wallet address continues until the customer receives the product and supply chain is ended. It is important that the activity diagram only includes the actions conducted by the corresponding actor. VerifiChain is not responsible of the communication of supply chain actors or the transportation methods. VerifiChain is a lightweight tool to ensure the reliability of the supply chain and ensure the originality of each product in the chain.

2.5.4.2 Product Verification State Diagram

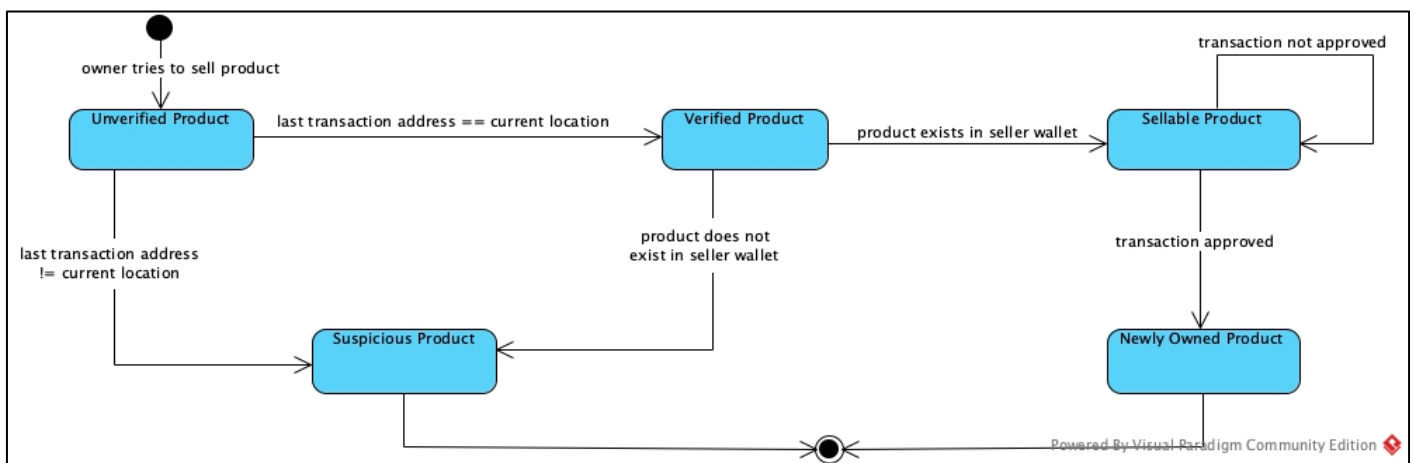


Figure 4: Product Verification in VerifiChain State Diagram

The state diagram indicates the process of a product to be verified and sold. A user or store with an unverified product tries to sell his/her product, and the candidate customer scans the qr code of the item that he/she wants to buy. Returned information indicates if the product is at the right store. If it is not, the product is either fraud or stolen. If the owner of the product in the transaction history matches the current location of the product, it means that the product is at the right place. Also if the wallet id of the seller and the wallet id that is assigned to the product is the same, that means that product actually belongs to him/her or store and the seller has the authority to sell the product. If both parties approve the transaction and no further problems arise then product credentials will be transferred to the customer.

2.5.4.3 Buy Product Sequence Diagram

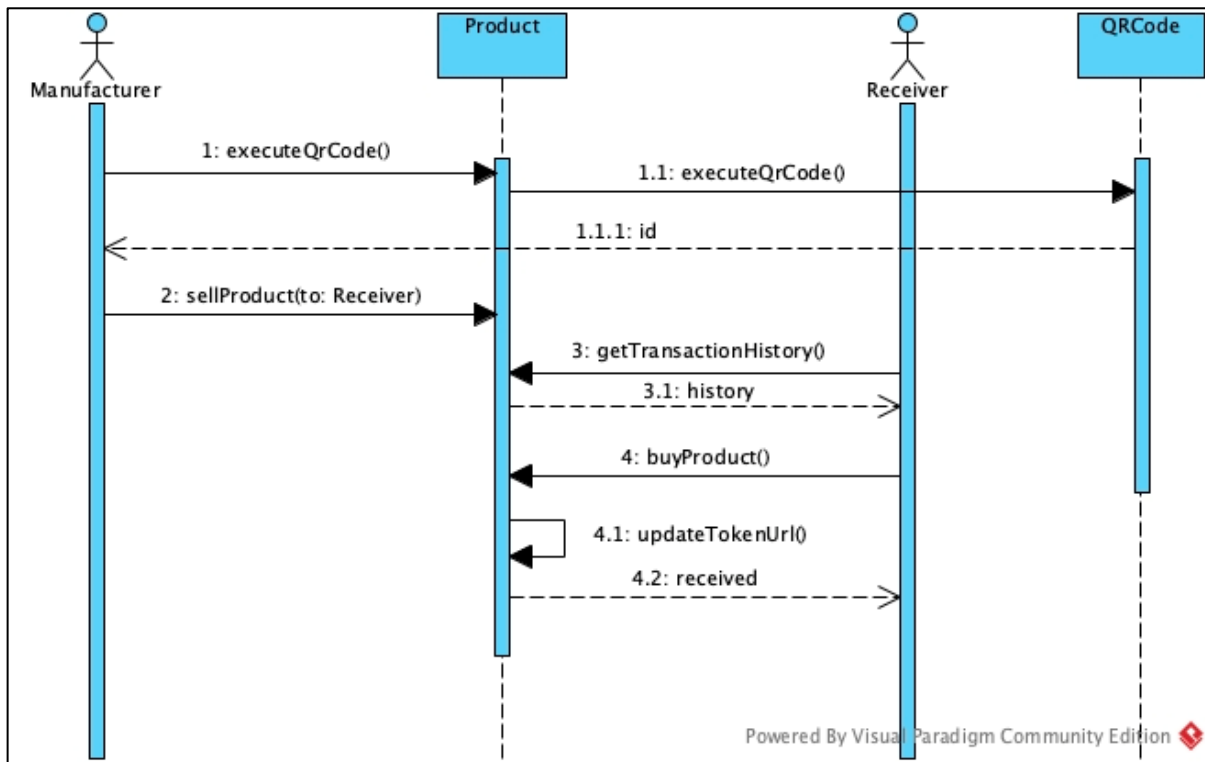
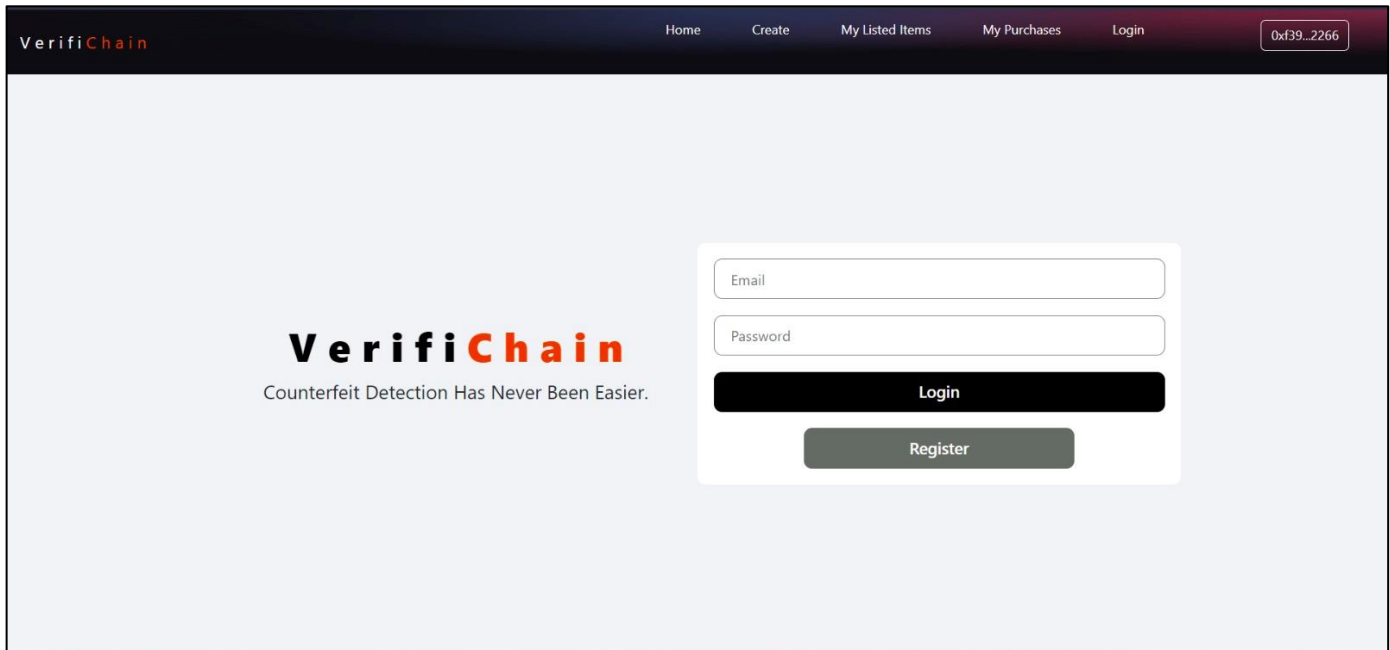


Figure 5: Sequence Diagram of Buy Product in VerifiChain

The sequence diagram indicates the process starting from a manufacturer creating a product until a user buys that product. Initially, the manufacturer executes a QR code and assigns it to the product. At the end of this process, the QR code is assigned to the wallet of the manufacturer. Then, when he/she wants to sell the product, he/she calls the `sellProduct()` method giving the specific customer id. While the customer is buying the product, he/she can scan the QR code to see the transaction history and verify its originality. After that, if the user chooses to buy that product, the product will be transferred to his/her wallet. Product token url will be updated respectively.

2.5.5 User Interface Mock-Up



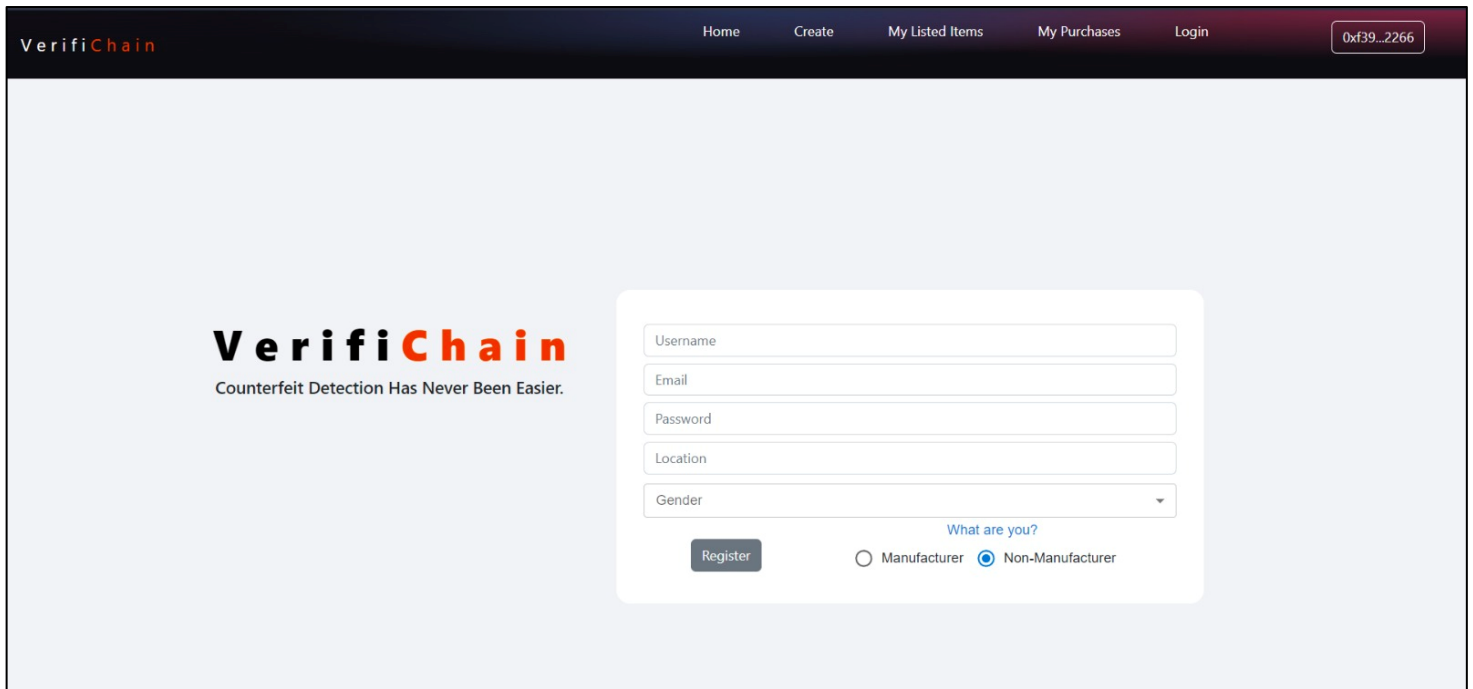
The image shows a user interface mock-up for the VerifiChain login page. At the top, there is a dark navigation bar with the VerifiChain logo on the left and links for Home, Create, My Listed Items, My Purchases, and Login on the right. A user address '0xf39...2266' is displayed in the top right corner. The main content area has a light blue background. On the left, the VerifiChain logo is displayed with the tagline 'Counterfeit Detection Has Never Been Easier.' On the right, there is a white login form with two input fields for 'Email' and 'Password'. Below these fields are two buttons: a black 'Login' button and a grey 'Register' button.

Figure 6: Login Page

Users have their own credentials that are stored in the database of VerifiChain; therefore, they have to enter necessary credentials to sign in to the system. The necessary information for entering VerifiChain are respectively:

- Email and
- Password.

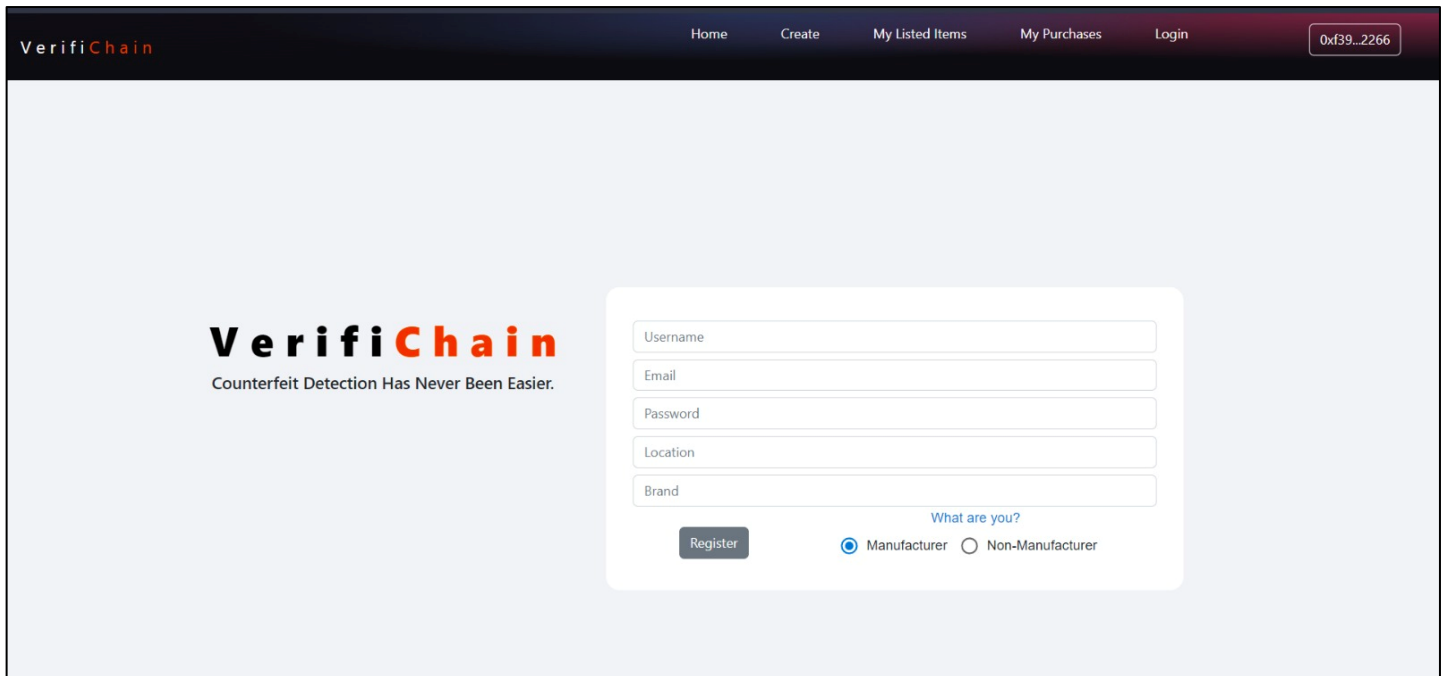
After they enter the required information to the login page, they will enter the application by clicking on the login button.



The screenshot shows the VerifiChain application's registration page for non-manufacturers. The header features the VerifiChain logo on the left and navigation links (Home, Create, My Listed Items, My Purchases, Login) on the right, along with a user ID '0xf39...2266'. The main content area has a light blue background. On the left, the VerifiChain logo is displayed with the tagline 'Counterfeit Detection Has Never Been Easier.' To the right, a white registration form is centered. The form includes input fields for Username, Email, Password, Location, and Gender. Below these fields is a 'Register' button and a section titled 'What are you?' with radio buttons for 'Manufacturer' and 'Non-Manufacturer', where 'Non-Manufacturer' is selected.

Figure 7: Non-Manufacturer Register Page

User uses the register page to sign up to the application. Username, e-mail, password, location and gender information are required for the user to register. After providing necessary information, the user chooses whether to be a manufacturer or non-manufacturer. After selecting the user type, the user clicks on the “Register” button to register to the database. User navigated to the homepage. This is an example for non manufacturers.



The screenshot shows the VerifiChain website's manufacturer registration page. The header features the VerifiChain logo on the left and navigation links (Home, Create, My Listed Items, My Purchases, Login) on the right, along with a user ID '0xf39...2266'. The main content area displays the VerifiChain logo and tagline 'Counterfeit Detection Has Never Been Easier.' on the left. On the right, a registration form is presented within a white box. The form includes input fields for Username, Email, Password, Location, and Brand. Below these fields is a 'Register' button and a section titled 'What are you?' with radio button options for 'Manufacturer' (selected) and 'Non-Manufacturer'.

Figure 8: Manufacturer Register Page

User uses the register page to sign up to the application. Username, e-mail, password, location and gender information are required for the user to register. After providing necessary information, the user chooses whether to be a manufacturer or non-manufacturer. After selecting the user type, the user clicks on the “Register” button to register to the database. User navigated to the homepage. This is an example for manufacturers.

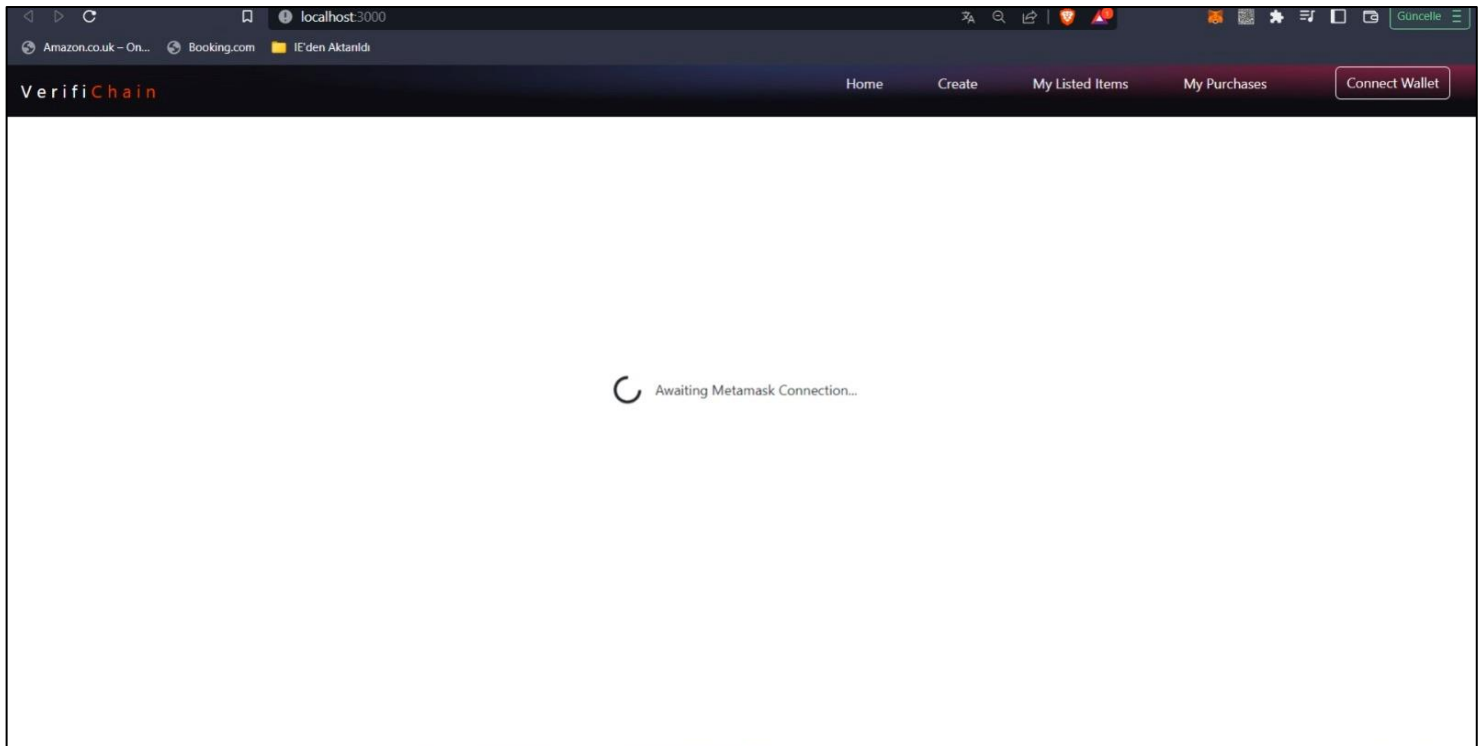


Figure 8: Metamask Connection

When a user first login to the system, they will firstly be introduced with the homepage for each actor type. However, to view the elements on the homepage, they have to connect their wallet because without connecting a wallet, VerifiChain cannot find an address to demonstrate the listed products in the connected chain. Therefore, the system will not demonstrate anything until users connect their wallet. When they hit the “Connect Wallet” button at the navigation bar, the MetaMask Wallet application will prompt user to connect the wallet to the VerifiChain system. After VerifiChain successfully connects the wallet of a user, the listed products of the connected chain will be demonstrated to the users.

VerifiChain

[Home](#)[Create](#)[My Listed Items](#)[My Purchases](#)[Login](#)

0xf39...2266

Dosya Seç3655320_0.jpg

Product #1

A simple product.

2

Create Product

Generate QR Code




Figure 9: Create Product Page

Only manufacturers will view the product creation and QR code execution part. In this page, manufacturers can choose an image, give a name, write a description, determine a price, and generate a QR code for each product before they list and assign the product.

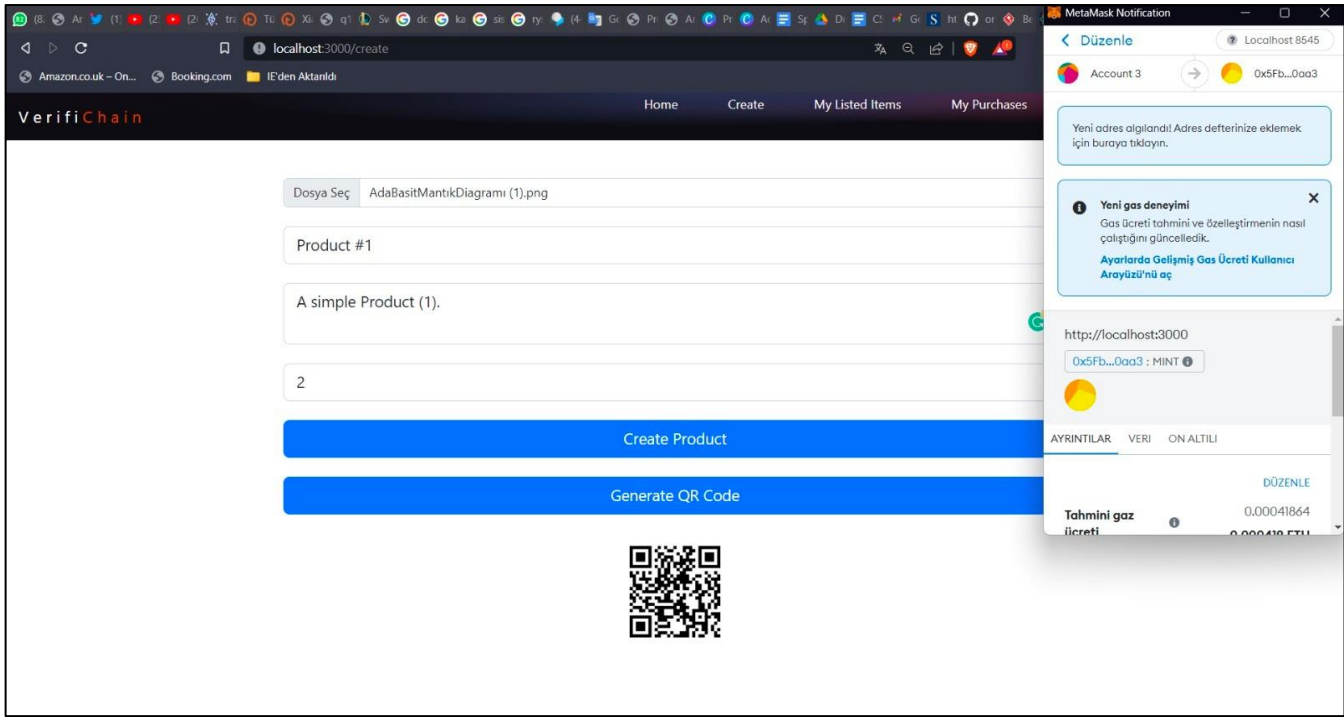


Figure 10: Create Product Page (Cont.)

After manufacturers hit the create product button, they will interact with the MetaMask Wallet application. They will be prompted that their wallet is interacting with a contract deployed on the chain that they are connected to (Ethereum Chain). Furthermore, MetaMask will demonstrate the gas fees for the interaction of the contract and their wallet. Users have to accept the contract interaction and gas fee to proceed for listing and deploying the product onto blockchain.

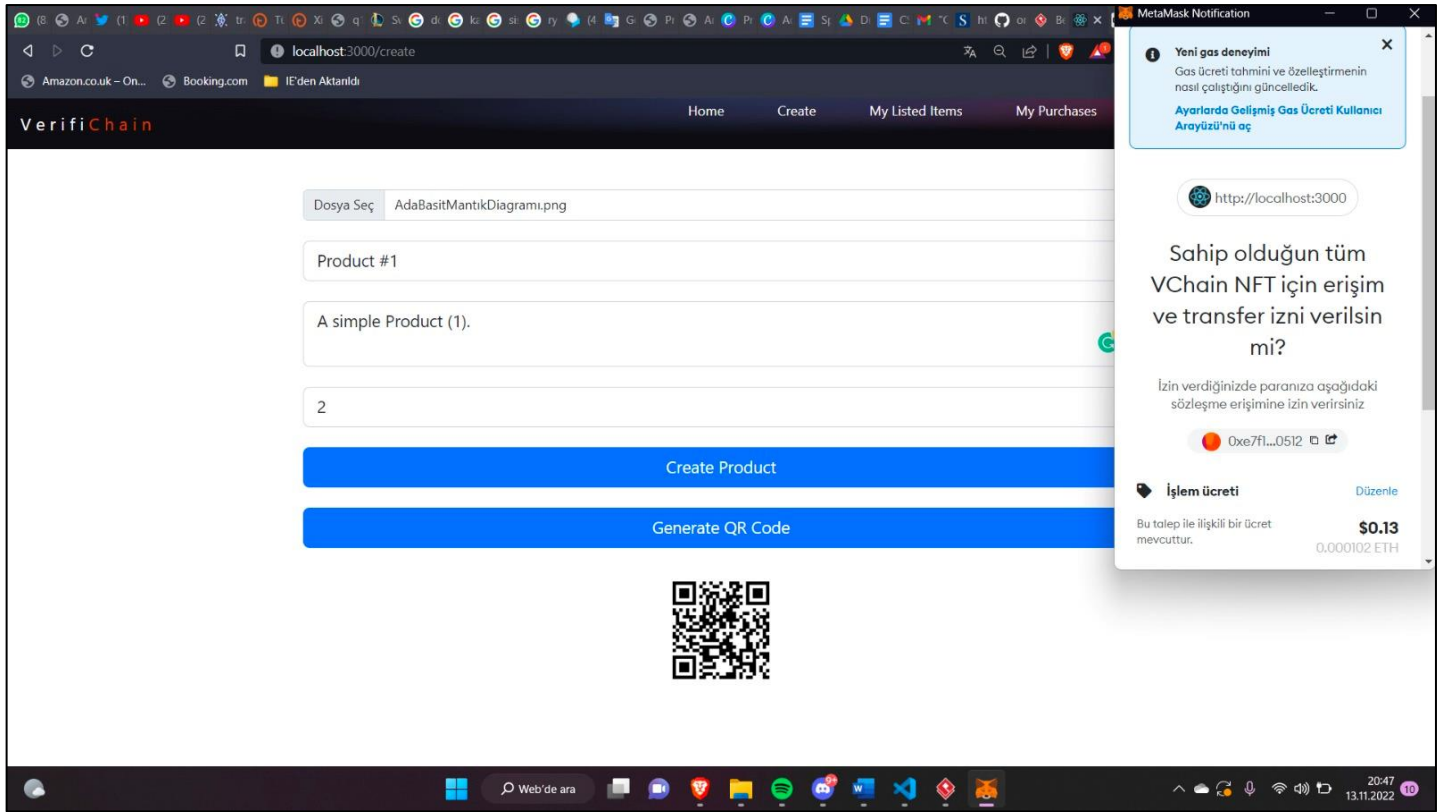


Figure 11: Create Product Page (Cont.)

After accepting the request that comes from Metamask wallet, users will be prompt about the access permissions. Hereby, the deployed contracts will interact their wallet addresses; therefore, MetaMask wallet requires a permission for increasing the security. Furthermore, as can be seen, users also will be informed about the gas fees that they have to compensate for by MetaMask. If they cannot afford the minimum requirement gas fee, they cannot create a product and deploy it onto blockchain.

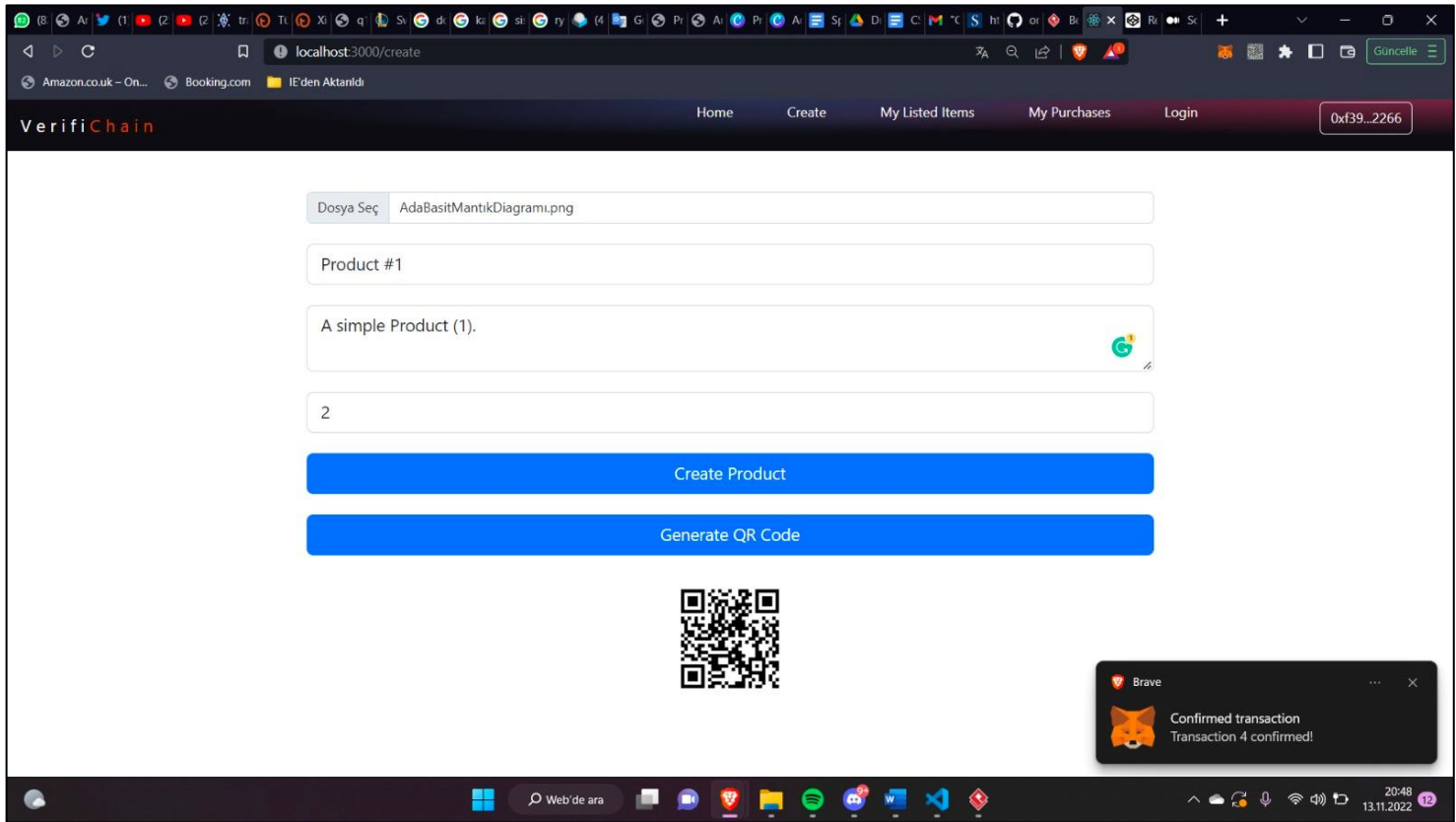


Figure 12: Create Product Page (Cont.)

After the product has been deployed on blockchain, users will be informed via pop-up notification from browser that the transaction has been successfully confirmed.

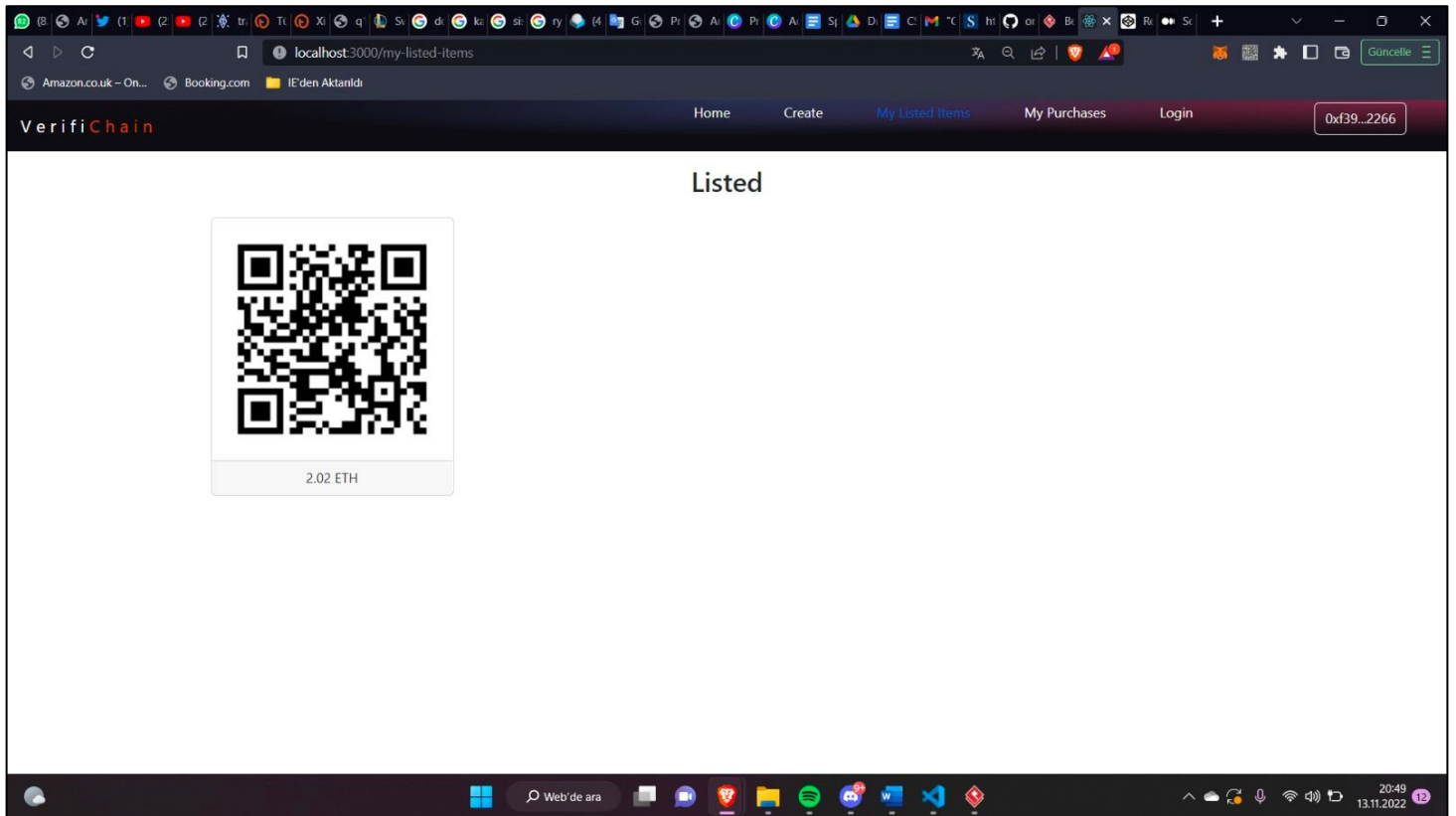


Figure 13: Listed Products Page

After creating a product with smart contracts. When a user hits the My Listed Items navigation link, a new page will be opened. Inside of this page all the created products will be listed by their QR codes alongside with their prices beneath them. This will show users how many products that they have created via smart contracts and price of each one of them.

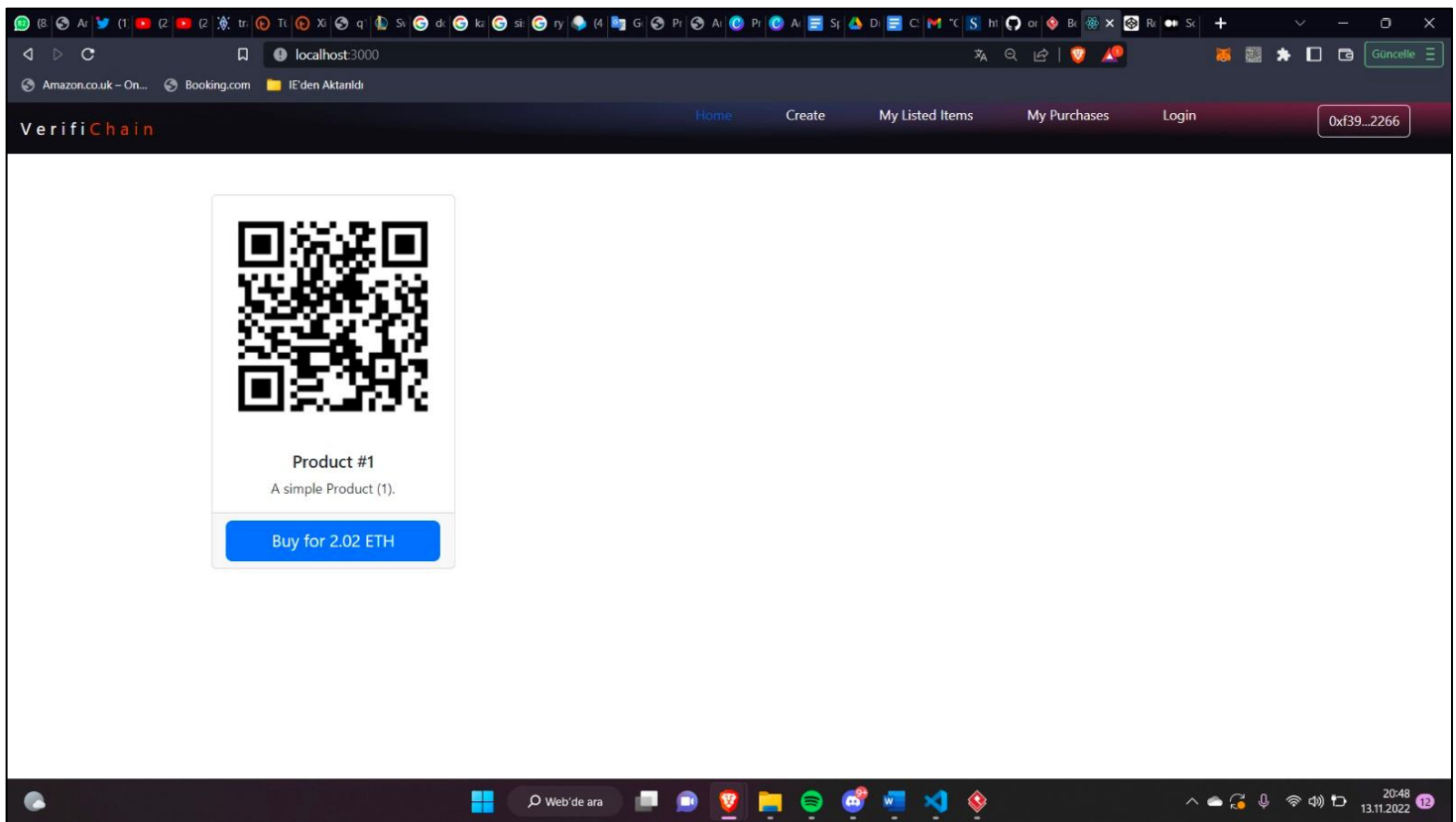


Figure 14: Listed Products Page

If a user wants to buy a specific product, they can do it on the homepage. User simply clicks on the “Buy for 2.02 ETH” button and if the user’s Metamask wallet have the sufficient value, a transaction successfully occurs. However, the product only will appear on the user that is assigned to.

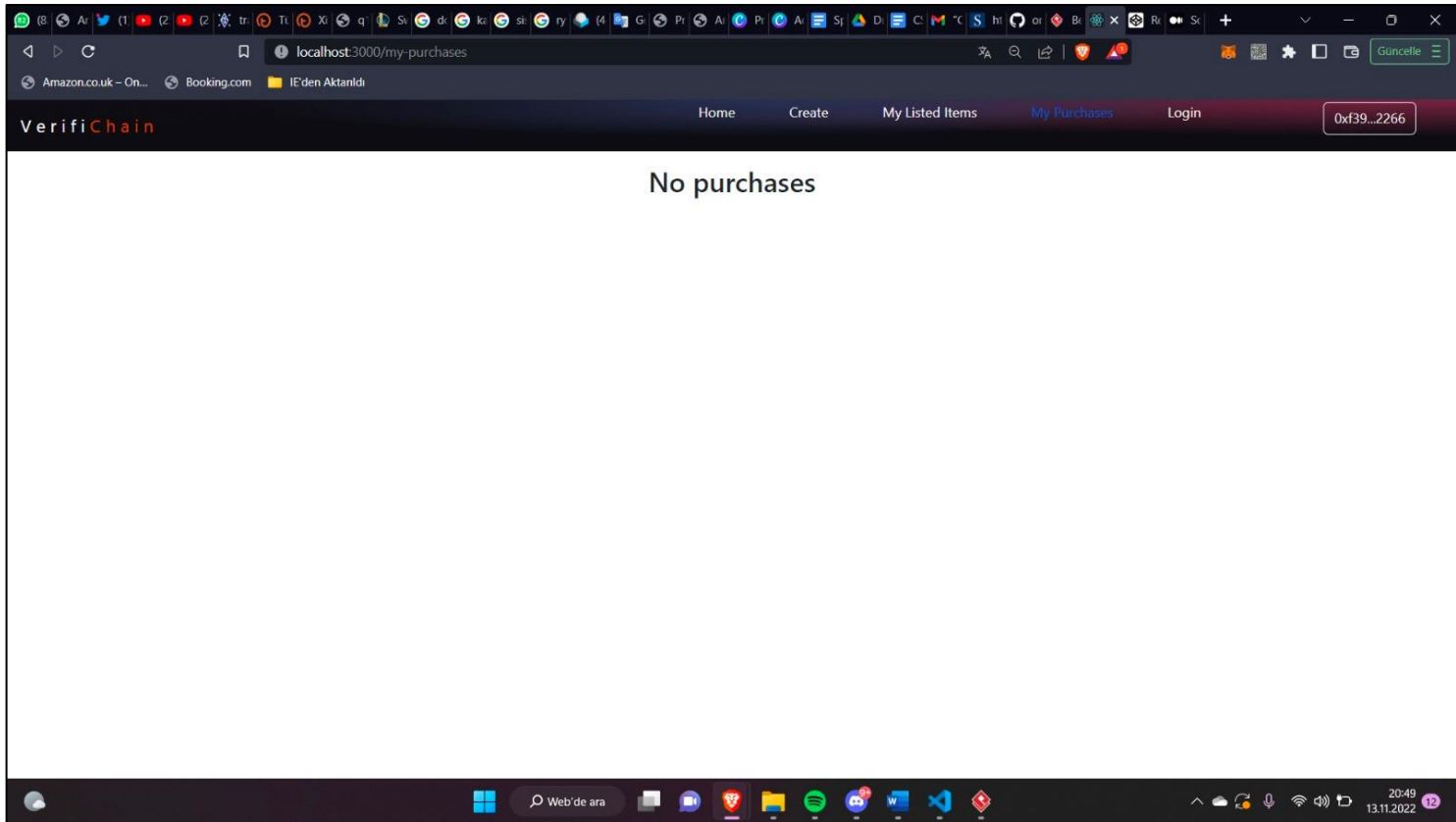


Figure 15: Purchased Products Page

If a user bought a product, it displays in the “My Purchases” page. The product’s information and transaction history can be seen in this page. Product items will occur here after the transaction completes.

3. Other Analysis Elements

3.1 Consideration of Various Factors in Engineering Design

The purpose of VerifiChain is to improve the verification of products. With the help of VerifiChain exchange and shopping for the unique products will be more reliable. Since the purpose of this project is to increase reliability and near perfection, the main consideration is to create a fully functional platform to check QR codes which will be placed on products that allow users to check the quality and previous transactions of their products or products that they are going to buy.

Since Verifichain will be using metamask wallets which are integrated with Ethereum Network, the second consideration of this project is to have smooth transactions between these wallets. Due to that this project aims to create a system that first checks both sides of transaction approvals and then makes the transaction.

3.2 Risks and Alternatives

Verifichain creates an improved verification system that checks the quality and reliability of products that are created and inserted inside the system. While creating a platform that provides users with better shopping and transaction options, this system should not be seen as a marketplace. This system will be used to ensure the quality of the product and show transaction history when an agreement has been done between both sides. However, there is only one case that our system fails to prevent, which is copying a QR code and inserting a fake product into our system. Although this problem has been nearly fixed by creating holographic QR codes which will be harder to copy and making copied products must be bought before the QR code is copied and used to create the fake product. This affects the marketing value of the product since it's not a firsthand product this also makes this risk less profitable.

3.3 Project Plan

WP#	Work package title	Leader	Members involved
WP1	Project Specification	Onat	All Members
WP2	Analysis Report	Arda	All Members
WP3	High-Level Design	Tuna	All Members
WP4	First Prototype	Bartu	All Members
WP5	Low-Level Design	Erengazi	All Members
WP6	Database Implementation	Tuna	All Members
WP7	User Interface Implementation	Bartu	All Members
WP8	Second Prototype	Erengazi	All Members
WP9	Final Implementation	Arda	All Members
WP10	Final Report	Onat	All Members

WP 1: Project Specification			
Start date: 10.10.2022 End date: 17.10.2022			
Leader:	Onat Postacı	Members involved:	Tuna Öğüt, Erengazi Mutlu, Bartu Teber, Arda Çiftçi
Objectives: Describe the project briefly, determine the functional, non-functional requirements and constraints. Discuss some professional and ethical issues.			
Tasks: <ul style="list-style-type: none"> Identify functional requirements Identify constraints Discuss professional and ethical issues 			
Deliverables: Project Specifications Report			

WP 2: Analysis Report			
Start date: 6.11.2022 End date: 13.11.2022			
Leader:	Arda Çiftçi	Members involved:	Tuna Öğüt, Erengazi Mutlu, Bartu Teber, Onat Postacı
Objectives: Create first mock-ups, create work packages, think about certain professional ethics, design the systems using diagrams.			
Tasks: <ul style="list-style-type: none"> Construct class, state, activity, and sequence diagrams to further explain Draw initial mock-up for user interface Take into account the project's moral and professional ramifications Define work packages and elaborate them 			
Deliverables Analysis Report			

WP 3: High-Level Design			
Start date: 14.11.2022 End date: 24.11.2022			
Leader:	Tuna Öğüt	Members involved:	Arda Çiftçi, Erengazi Mutlu, Bartu Teber, Onat Postacı
Objectives: Create a software architecture for the system, identify the boundary conditions and subsystems, and take various social elements into account.			
Tasks: <ul style="list-style-type: none"> Determine the subsystem decomposition, HW/SW mapping, persistent data management, etc. to propose a software architecture Establish the boundary circumstances For Wheelancer, take into account various social factors 			
Deliverables High-level Design Report			

WP 4: First Prototype			
Start date: 25.11.2022 End date: 30.11.2022			
Leader:	<i>Bartu Teber</i>	Members involved:	<i>Tuna Öğüt, Erengazi Mutlu, Arda Çiftçi, Onat Postacı</i>
Objectives: Create each module in the project with care and detail to ensure that the coding goes smoothly.			
Tasks: <ul style="list-style-type: none"> • Writing the modules in detail • Writing the architecture in detail 			
Deliverables First Prototype Code			

WP 5: Low-Level Design Report			
Start date: 1.12.2022 End date: TBD			
Leader:	<i>Erengazi Mutlu</i>	Members involved:	<i>Tuna Öğüt, Arda Güven Çiftçi, Bartu Teber, Onat Postacı</i>
Objectives: Create each module in the project with care so that the coding can go smoothly.			
Tasks: TBD			
Deliverables Low-Level Design Report			

WP 6: Database Implementation			
Start date: 12.12.2022 End date: TBD			
Leader:	Tuna Öğüt	Members involved:	Arda Güven Çiftçi, Erengazi Mutlu, Bartu Teber, Onat Postacı
Objectives: <i>Database Implementation</i>			
Tasks: <i>TBD</i>			
Deliverables D1: Database code in PostgreSQL			

WP 7: User Interface Implementation			
Start date: 20.12.2022 End date: TBD			
Leader:	Bartu Teber	Members involved:	Tuna Öğüt, Erengazi Mutlu, Arda Çiftçi, Onat Postacı
Objectives: <i>User Interface Implementation</i>			
Tasks: TBD			
Deliverables D1: User Interface code			

WP 8: Second Prototype			
Start date: TBD End date: TBD			

Leader:	Erengazi Mutlu	Members involved:	Tuna Öğüt, Arda Güven Çiftçi, Bartu Teber, Onat Postacı
Objectives: Build the second prototype of the VerifiChain by implementing some complex functionality			
Tasks: TBD			
Deliverables D1: Second Prototype			

WP 9: Final Implementation			
Start date: TBD End date: TBD			
Leader:	Arda Güven Çiftçi	Members involved:	Tuna Öğüt, Erengazi Mutlu, Bartu Teber, Onat Postacı
Objectives: Finalize the implementation according to the feedbacks from previous prototypes			
Tasks: TBD			
Deliverables D1: Final Implementation			

WP 10: Final Report			
Start date: TBD End date: TBD			
Leader:	Onat Postacı	Members involved:	Tuna Öğüt, Erengazi Mutlu, Bartu Teber, Arda Güven Çiftçi
Objectives: Summarize the entire project, the requirements that were satisfied, and the improvements that were made. Provide a user manual for the project, implementation information, and a maintenance schedule.			
Tasks: TBD			

Deliverables <i>D1: Final Report</i>
--

3.4 Ensuring Proper Teamwork

For most of our group members, the senior design project is one of the largest-scale projects they have done in Bilkent. Due to that, we understand that everything needs to go planned and organized. We ensure every member is following the project and deadlines through our continually weekly meetings. In our meetings, we plan everything on the project as a team and we make sure that every team member understands and approves what will be added or subtracted from our project. Also in these meetings, we assign tasks to team members which they want to complete. While doing these assignments we make sure that every team member has a balanced workload. If a team member has a huge portion of the work that needs to be done this will also be divided between team members which again makes sure that we have a properly balanced workload share.

For our project, a Google Drive folder is used for our reports including this one. This will allow every team member to have access to reports and update them simultaneously which will save more time while creating reports.

While developing our project we will use Git for version controls and GitHub to store our development process. This will provide us with a better development process while creating our project.

3.5 Ethics and Professional Responsibilities

Since in our project we will be using Metamask wallets there will be transparency in wallets which allows developers to see assets of the wallet but these assets will not be presented in our platform. Therefore users will not see assets of other users' Metamask wallets. However, since these wallets have been addressed, if users have the

address of the wallet they can be searched and find these Metamask wallets by Etherscan.

Because our platform Verifichain is a product verification application rather than a marketplace, users that have access to the QR code of the product will be able to see the transaction history of the product which helps them to understand the verification of the product. We will provide an agreement about the access permissions to wallets and transaction history to our users when they enter VerifiChain. This will provide a better understanding to every user of how we process their data.

Since our main goal in VerifiChain is to assure the user that they will buy or sell a verified product, manufacturers and distributors who want to have transactions about their products through our platform need to have some requirements and restrictions about their products. One of the most important requirements is they must include Holographic QR codes in their products since these unique holographic QR codes will be created for every product.

3.6 Planning for New Knowledge and Learning Strategies

For the UI design we will use React.js. Some of the team members already used this framework and have knowledge from their past internships. Also, we plan to use React.js documentation [1] whenever we need and there are many open source projects developed by using React.js. Therefore, we plan to use official documentation and samples from the internet to develop our application UI.

The major functionality of our application will be handled by the backend service, which will be a REST API running on the Express.js framework for Node.js. Also, to keep the transfer history of the products and be integrated with the ethereum blockchain[eth], we will use smart contracts written in the solidity language[2]. To do all of these, knowledge of javascript is essential to implement necessary functions.

Lastly, we plan to use PostgreSQL for keeping the registered user information and show the transaction of the unique item by using solidity language. We plan to use

official documentation of PostgreSQL[4] and Solidity language. Also, as we discussed above, there are many open source projects that can be used as an example.

4. Glossary

User: All of the types of account accepted as a user.

Receiver: The user who is a distributor, retailer or customer.

Manufacturer: The user who is executing the QR code for a specific product.

Metamask Wallet: Ethereum integrated wallet.

Ethereum: Ethereum is a decentralized, open-source blockchain with smart contract functionality.

5. References

- [1] "Getting started," *React*. [Online]. Available: <https://reactjs.org/docs/getting-started.html>. [Accessed: 13-Nov-2022].
- [2] "Solidity," *Solidity*. [Online]. Available: <https://docs.soliditylang.org/en/v0.8.17/>. [Accessed: 13-Nov-2022].
- [3] "Home," *ethereum.org*. [Online]. Available: <https://ethereum.org/en/>. [Accessed: 13-Nov-2022].
- [4] "Documentation," *PostgreSQL*. [Online]. Available: <https://www.postgresql.org/docs/>. [Accessed: 13-Nov-2022].